Collaborative Execution of Exploration and Tracking using Move Value Estimation for Robot Teams (MVERT)

Ashley W. Stroupe

CMU-RI-TR-03-07

Thesis submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy in Robotics.

The Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania, 15213

September 2003

Abstract

This work presents Move Value Estimation for Robot Teams (MVERT), an architecture specifically designed for selecting low-level actions for multi-agent teams. The design goal for MVERT is to produce reasonable performance that takes advantage of a heterogeneous team while maintaining computational efficiency. MVERT is fully distributed – each agent selects actions based on its knowledge and knowledge provided it by teammates. Each robot approximates the expected next-step teammate contributions and, given these predictions, each robot can select its action to maximize the team's progress. MVERT represents progress with mathematical *value functions* that map state and robot task performance models to a numerical value representing mission utility.

Many action selection approaches (optimal trajectory planning, for example) in large state-spaces may be computationally prohibitive, particularly for online mission replanning. However, taking advantage of a team's multi-agent nature to provide efficiency requires consideration of teammate contributions. Thus, in selecting an action with MVERT, each robot approximates the next-step contributions of teammates by applying their sensing models, task capabilities, and current poses in the value functions. The robot then evaluates its candidate actions by applying the value functions and its own sensor models. The action resulting in the overall highest-valued pose is selected and executed.

Performance in each task is described by an individual value function. *State* includes current locations of teammates and objects in the environment. *Performance models* include task capabilities and sensor models. *Value functions* may be any mathematical representation of task performance. To determine an actions' overall value, independent task values are combined by weighted average. Weighting each task's value allows prioritizing tasks in accordance with desired performance. As progress reduces potential for improving value on some tasks, the weights automatically shift focus to the other tasks. Weights can be dynamically adapted as mission needs change.

MVERT has been applied in simulation and on physical robots for mapping, dynamic target tracking, and complex multi-task missions (planetary exploration). MVERT improves team mission performance time and completeness compared to individual action selection and greatly improves computation time compared to a one-step optimal. MVERT produces contextually appropriate actions for successfully performing complex multi-task, multi-robot missions.

Acknowledgements

I would like to thank my advisor, Tucker Balch, who helped me find my path and has provided continual support, advice, wonderful research opportunities, and interesting conversation during the last four years. Thanks also to my committee members. Manuela Veloso has provided a tremendous amount of support, great opportunities to participate in amazing research projects with RoboCup, and helpful insights into my thesis work. Tony Stentz and Lynne Parker have also provided wonderful feedback on my dissertation work, and great suggestions to contribute to the work.

I would like to thank the many researchers who directly contributed to this work (you each know what you did), listed roughly in order of appearance: Martin Martin, Rosemary Emery, Scott Lenser, Jim Bruce, Kevin Sikorski, Hans Moravec, Frank Dellaert, Maayan Roth, Sonia Chernova, Sarjoun Skaaf, Ram Ravichandran, and Michael Kaess.

I would also like to thank the many people at Carnegie Mellon who were helpful and supportive throughout my time here. Thanks to Matt Mason and Illah Nourbakhsh for serving on my qualifier committee and for the help and guidance you provided when I needed it. Thanks to Reid Simmons and Sanjiv Singh for providing me an interesting research opportunity with NASDA. Thanks also to Suzanne Lyons-Muth, Deb Cavlovich, Jean Harpley, and Cynthia Bryant for providing fantastic support. Thanks also to Daniel Huber for serving on my qualifier committee. Bernardine Dias, Joelle Pineau, Vandi Verma, Paul Tompkins, Sarjoun Skaaf, Nick Roy, Aaron Courville, Gita Sukthankar, and Chris Urmson, thank you for all the feedback and help with various papers, talks, and classwork. Thanks to Kim Shillcutt and Matt Deans for getting me involved in FIRST.

Finally, I want to thank family and friends. Deep thanks to my parents without whom I would not be here and who made me believe I could do anything. Thanks to my dad Clark who passed along his love of solving problems and who gave me a different perspective and feedback on my dissertation. Thanks to my mom Sharon who gave me great emotional support and helped me keep balance in my life (and without whom my apartment doors would be unlocked). Thanks to my brother Clark, whose talents, skills, and achievements made me strive to do well. Thanks to Heidi Pullen for her support and encouragement. Thanks to Lisa Waldstein for being a great friend and for providing me a safe haven in Atlanta during those long visits.

This work was supported by Carnegie Mellon University and the DARPA Mobile Autonomous Robot Software Program, with additional support from the DARPA Control of Agent Based Systems Program, Georgia Institute of Technology, and the Northrop Grumman Corporation.

Table of Contents

List of Figures

List of Tables

Chapter 1 Introduction and Problem Statement

In this work, it is asserted that in order to maximize the utility of heterogeneous teams, each agent on a team should consider the future contributions of teammates based on their different capabilities to select actions that will most help overall team progress toward mission completion. How to accomplish this is currently an active area of research. Planning full trajectories of actions for large teams in a complex, changing environment may be prohibitively costly in computation, particularly if the teams are heterogeneous. Up-front optimal planning may cause unacceptable delays before starting a mission, and replanning due to changes in the team or environment may further delay mission completion or fail to allow robots to adequately react to dynamic circumstances. Thus, efficiently maximizing the *team* contribution remains an open area of research. This work addresses the following research question:

How can a heterogeneous team of robots maximize team progress in a multi-task mission in a scalable, efficient manner at execution?

This can be broken into several subsidiary questions, concerning the execution level of teams:

- *How can robot teams effectively integrate multiple mission tasks?*
- *How should robot teams be distributed dynamically to best improve the uncertainty of maps and target locations?*
- *How should dynamic elements of the environment be treated?*
- *How can performance of multi-robot systems be evaluated in this context?*

This work specifically seeks to investigate the following thesis statement:

Applying value functions in action selection can provide efficient team execution of multiple parallel tasks.

Algorithms that address these questions could be used to control multi-robot teams for such complex tasks as large-scale planetary exploration (Figure 1) [30], [34], [79], [96], surveillance networks [56], [98], and construction [53]. Substantial research has been completed on individual components for such systems. Separately, tasks of mapping, deploying communications or sensor networks, and foraging/consumption (the tasks of locating objects in the environment and interacting with them or collecting them) have been investigated with productive results. Additionally, task allocation for multiple robot teams has been demonstrated using market-based approaches, model-based approaches, behavior-based approaches, and traditional optimal planning approaches. Execution of tasks using multiple robot teams have also been proven, including action selection for information gain in an exploration or mapping scenario. Lacking in the above related research is an efficient approach for action selection integrating many different types of tasks for a heterogeneous multi-robot team. Most of these approaches also have limitations in scalability for large teams and many tasks.

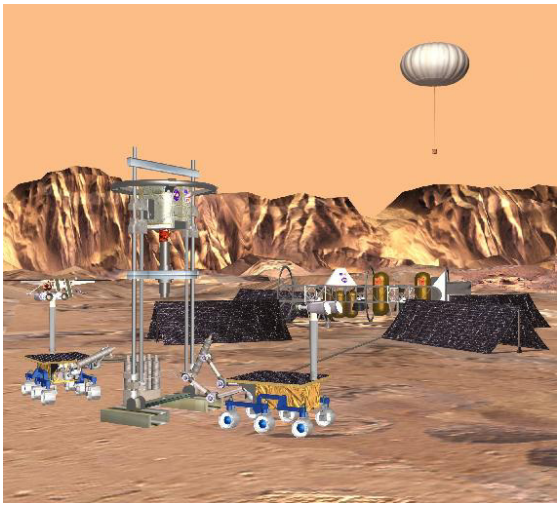

Figure 1. NASA/CMU Conception of a Martian Multi-Robot Colony.

This work presents Move Value Estimation for Robot Teams (MVERT), an architecture for selecting low-level actions at the execution layer of a robot control system. MVERT is specifically designed for multi-agent (robot) teams. The design goal for MVERT is to produce reasonable performance that takes advantage of the heterogeneous multi-robot team while maintaining computational efficiency. Here, *actions* are defined as selecting a pose to move to in the next time step. The MVERT action selection architecture represents progress toward mission goals with mathematical *value functions* that map current state and performance models to a numerical value representing progress. *Current state* includes the current locations of teammates as well as objects in the environment. *Performance models* include models of which tasks can be performed and sensor models. Using the current state and models of the teammates' capabilities, approximations of their next-step progress can be made by applying the value functions. Given these predictions, each robot can select the action for which the *team's* overall progress will be maximized. Computation is made scalable by adjusting the number of candidate moves considered by each robot and the complexity of the models/value functions to be evaluated at each step. Three assumptions are fundamentally inherent in the MVERT approach. First, team agents must be operating in a common coordinate frame. This can be established a prior or may be established during execution using traditional map registration techniques. Second, agents must be able to obtain information about relevant aspects of teammate state, either through communication or observation and inference. Third, changes in teammate state from one step to the next are small and can be reasonably approximated by a single average.

The MVERT architecture is considered behavior-based in the sense that separate behaviors (designed for each task) independently contribute to the selection of an action. The selection of an action is based only on current state of the team and the environment, not on any historical information, and selects actions for only one step. However, unlike a purely behavior-based system, some reasoning is applied to assist in the decision. Each robot makes a prediction, based on models, of teammate contributions and then reasons about the expected results of its own different candidate actions. This greedy search is essentially a one-step plan in the low-level action space.

The potential applications of MVERT include any tasks that can be represented by some computable mathematical function. These functions need not be smooth, continuous or differentiable, as they are evaluated at the current state and not optimized. Values may depend on any aspects of the current state of the world, and will typically depend on robot poses, object locations, and robot capabilities and sensing models. These functions may also be time dependent, changing mission priorities as time elapses. The total value of a state is a weighted average of the individual values for each task. The weights assigned to each task determine the resulting behavior of the team by prioritizing some tasks relative to others. As progress reduces potential for improving value on some tasks, the weights automatically shift focus to the other tasks. Weights may differ among robots to reflect differing capabilities or to diversify the team by providing subteams different priorities.

The MVERT architecture is implemented in simulation and on physical robot teams. Value functions are developed for several tasks: target position estimation for multiple static or dynamic targets, target sampling tasks, maintaining line of sight connectivity, and exploration. As the goal of mapping (or other target location tasks) is to minimize the uncertainty on the locations of targets, the target location value function represents target uncertainty with an uncertainty area; larger areas are lower value than smaller areas. The target sampling task requires moving to a specified location, and remaining at that location to perform a task such as analysis or manipulation. The value function for target sampling encourages moving toward the target sampling location by giving positive value to reductions in distance, and then encourages maintaining the sampling location by shifting to a forced local optimum for the duration of task completion. The value function for exploration is identical to the target sampling value, though it does not require remaining at the location. Lastly, the network connectivity value function assumes a line-of-sight model for communication. It varies from 0 to 1 with the fraction of robots that can be seen directly or indirectly (by relay via one or more teammates).

These value functions are applied to action selection for robot teams in four mission types. The first two mission types specifically address the second subsidiary research question by focusing on mapping tasks. The first of these missions is an idealized mapping task, called *target location*. The basic performance of MVERT and the isolated effects of varying parameters, team size, and robot capabilities are examined using this simplified mission. The second is a traditional mapping task, *mapping*, in an unknown environment with noisy observation and motion. The third mission type, addressing the third subsidiary

research question, is *dynamic target tracking*, in which robots locate dynamic objects (with unknown motions) within an environment with known localization landmarks. These first three missions utilize the fundamental value function for minimizing target location uncertainty to select actions. The last mission type, addressing the first subsidiary question, is designed to investigate the utility of MVERT in more than one task simultaneously: *planetary exploration*. All value functions are applied to this complex, multi-task mission that requires mapping and exploring an unknown environment, target sampling (performing some task) at specified locations, and maintaining line of sight communications for sharing information.

To address the fourth and final subsidiary research question, several metrics have been developed and suggested as relevant for evaluating the performance of teams in an exploration-like mission. These metrics are related to mission goals: map uncertainty, map accuracy, total mission completion time, percentage of mission completeness, and robot localization accuracy. These metrics have been applied to evaluate and compare the performance of MVERT in the various missions. MVERT performance is compared to several other approaches qualitatively and quantitatively using these metrics. The first approach is individual action selection, in which robots attempt to maximize their own progress toward mission completion without regard to teammate contributions. The second approach is the one-step optimal, in which the set of moves for the team that maximizes mission progress (through maximizing value) is selected, rather than the approximation used by MVERT. The third approach for comparison is a commonly-used approach to mapping, the division of the exploration area into smaller areas, each of which is assigned a robot to map it using a coverage pattern.

The results of this work indicate that the MVERT architecture can successfully select actions that approximately maximize mission progress for a team. For mapping tasks, robots automatically select actions that will make the most contribution, given their heterogeneous sensing capabilities. Thus, with asymmetrical sensors that require multiple points of view to fully localize objects, robots as a team move to produce observations from complementary visual axes. To reduce uncertainty on all objects simultaneously, robots automatically distribute themselves among the targets. These results are also reflected in the dynamic target tracking task. In the complex planetary exploration mission, MVERT demonstrates the ability to integrate multiple mission tasks, selecting actions that prioritize the mission tasks as desired by tuning the relative weights of each task's value. MVERT provides advantages over individual action selection by better covering the space and by providing higher value (lower uncertainty) target position estimates. MVERT also provides advantages over the one-step optimal by producing similar results with far less computation, as it scales linearly with team size rather than exponentially. Computing the optimal for large teams or complex missions (such as the planetary exploration mission investigated here) would be impossible in real time. Lastly, MVERT provides advantages over coverage patterns in some cases, such as when applying small teams in environments with non-uniformly distributed targets, and in the ability to prioritize and perform multiple tasks.

The contributions of this work can be summarized as follows:

- An architecture for selecting actions for a heterogeneous team at execution level during a complex mission, including:
  - A method for iterative, near-optimal distribution of a robot team for mapping multiple distributed static and dynamic objects;
  - A method for selecting actions to maximize mission progress in a multi-task mission.
- Validation of the approach in simulation for target location, mapping, target tracking, and a multi-task complex mission with heterogeneous team.
- An implementation of this framework on a real multi-robot system for mapping and target tracking.
- Metrics proposed for evaluating the performance of such multi-robot systems.
- Suggested simple value functions for some common tasks.

While a powerful tool for the low-level execution systems described in this work, MVERT performance may potentially benefit from several future research directions. MVERT is envisioned as part of the execution layer of a more complex multi-robot control system that includes higher-level reasoning. By taking some aspects of low-level planning into account at the MVERT level, the computation required at the reasoning level may be reduced. Additionally, the use of traditional learning approaches during design

and execution may allow improving the value functions' ability to evaluate progress, providing better team performance. Lastly, some of the assumptions about simple environment models must be relaxed for MVERT to be applied in practical systems.

This document is organized as follows. In Chapter 2, research in areas related to this work is summarized and the limitations relevant to this thesis are discussed. The theory of the MVERT architecture, including some of the fundamental design choices, is described in Chapter 3. Chapter 4 presents the mathematical treatment of uncertainty in this work, including representation and methods for combining information. Performance evaluation metrics are suggested in Chapter 5 to address the last subsidiary research question. Experimental platforms, environments, and mission scenarios are also detailed in Chapter 5. Chapter 6 presents a method of localization, developed as part of this work, that is applied to some of the experiments presented here. Chapter 7 though Chapter 10 present the detailed experimental descriptions, results, and discussion, divided by mission task. Chapter 7 presents results of the Target Localization Mission, a simulated mapping-related task used to explore the basic performance of MVERT and the effects of varying different parameters in simulation. A more realistic mapping mission is presented in Chapter 8, with results for several environments in simulation and on a physical multi-robot team. Chapter 7 and Chapter 8 directly address the second subsidiary research question. Experiments, in simulation and on a physical robot team, in dynamic target tracking missions using MVERT are presented in Chapter 9. This chapter specifically addresses the third subsidiary question. Lastly, a complex multi-task mission representing planetary exploration (in simulation) is presented in Chapter 10. The experiments in Chapter 10 are designed to address the first subsidiary research question. Chapter 11 concludes with a summary of the approach and discussion of results, including a reiteration of the contributions of this work and suggested future research directions.

Chapter 2 Background and Related Work

Several areas of robotics research are relevant to this thesis. These are robot localization, mapping of environments, dynamic target tracking, distributed sensing, task allocation, and action selection. In particular, approaches geared toward teams of multiple robots are most relevant to this work.

2.1 Robot Localization

Robot Localization is the process of determining a robot's pose (position and orientation) within its environment as defined by an external frame. Relative localization allows teammates to localize relative to each other without respect to an external reference frame. Global localization is the process of determining a robot's pose within an environment, as defined by an external frame, without any prior estimate of pose.

2.1.1 Single-Robot Approaches to Robot Localization

The most basic approach to localization is triangulation using reference measurements of range and bearing to landmarks with known positions. This is the approach most frequently used by humans [44]. Localizing with the Global Positioning System (GPS) is a form of triangulation, using satellites as reference landmarks. Many robotic systems use types of triangulation, including GPS, in order to localize within a known environment [34], [41], [54], [55], [69], [90], [91], [101], [108], [122], [127]. Triangulation is also frequently performed using only bearing measurements to reference landmarks [22], [23], [38], [116]. It can be performed in both continuous and discrete coordinate frames. Typically, localization by triangulation does not directly provide quality measures on position estimates.

Kalman-Bucy Filters (KBFs) are the most commonly used method of localization for robots [32], [33], [41], [46], [48], [49], [51], [52], [64], [70], [88], [110], [111], [112]. KBFs use statistical methods to incorporate measurement uncertainty in position estimates [14], [65], [80]. A KBF update consists of a process update, in which an estimate of robot motion and motion error models are incorporated into the robot's pose, and a sensor update, in which corrections are made to the pose estimate based on sensed landmarks. Pose estimates are typically represented by a single multi-dimensional Gaussian distribution and updates are a weighted multiplication of these distributions. The future sensor measurements are predicted based on the motion of the robot and the location of the landmarks. The weights applied to new sensor readings are determined by how well the sensor readings agree with the prediction. KBFs can keep robots localized when the Gaussian models hold, but can only recover from errors within bounds because highly unlikely sensor readings are given little weight, even though possibly correct [46].

Maximum Likelihood localization chooses the most likely pose as the current estimate. Updates use Bayes' Rule, which relates a new pose belief distribution to the prior pose belief distribution and a sensor or motion model; this can also be easily implemented to track multiple hypotheses of robot position and does not require Gaussian distribution [120]. In my previous work, Probabilistic Constraint-Based Localization, a Bayes' Rule update is combined with a KBF approach to use Gaussian multiplication and geometric constraints to simplify computation of pose estimates [127]. CBL identifies lost robots when sensors do not match pose estimates and increases uncertainty to allow recovery and global localization.

Markov Localization (or Monte Carlo Localization) can also perform global robot localization, and various implementations have been applied to many systems [10], [15], [25], [26], [36], [37], [66], [75], [81], [117], [135]. ML and MCL represent robot positions as a set of sample particles, each sample representing a possible pose, allowing arbitrary distributions over pose belief. The density of particles within an area of pose space represents the likelihood that the true pose lies in that area of pose space. Pose estimates are the center of mass of the particles. Updates are performed using Bayes' Rule. This approach trades off computational complexity (in terms of number of particles to update) with accuracy of pose estimates. By maintaining a small number of particles distributed throughout the space, robots can recover from being lost. Initialization with a uniform distribution of particles within the environment allows for localization within the environment without any knowledge of initial pose. In one implementation, Sensor Resetting Localization, lost robots, identified when sensor readings do not match expectations, are reset to a state of unknown pose to allow faster recovery [73].

A final commonly used approach to robot localization is the use of template matching. A priori, poses within the environment are sampled and represented by the sensor readings made from those poses. Typically, templates are geometric, such as a set of edges from an image. Localization compares current sensor readings to the stored templates and the closest match or interpolation between templates provides a location for the robot [20], [45], [49], [76], [141]. Another commonly used type of template is Cell Occupancy Grids [18], [43], [55], [62], [87]. Cell Occupancy grids discretize the space, marking cells as occupied or free. Sensor input is a scan of local occupied/free space that can be matched to the discrete map. Monte Carlo Localization [81] and Kalman-Bucy filters [49], [51], [52] can also be applied to templates rather than range-bearing information to landmarks. Locations represented by templates can either be in a continuous representation (as in the case of a single map) or may be distinct locations linked in a topological map. In one case, time-dependent models of the environment are matched against symbolic interpretations of sensor input to handle environmental changes [104].

2.1.2 Multi-Robot Approaches to Robot Localization

In a perfectly known environment, multiple robots may localize independently using the map. However, the performance may be improved by allowing robots to help each other localize. In this case, as robots observe each other and the environment, sharing these observations increases the information available to each robot for localization. By using teammates as landmarks, in some cases dependence on environmental landmarks for localization can be reduced or eliminated.

Without the need for an external reference frame, such as in formation following, robots may perform relative localization with their teammates. There are a few examples of using only relative positioning to localize robots [57], [59], [90].

One of the simplest approaches to team localization is the “leap-frog” approach in which some robots remain stationary to serve as landmarks while other robots move through the environment. Robots take turns serving as landmarks and explorers [43], [66], [69], [90], [91], [108], [122]. Updates are performed using the geometrical relationships between the moving robot and the stationary robots. Without external references, accumulated error from imperfect motion can still grow as robots explore. The upper limit on pose uncertainty is bounded [69]. This approach requires robots to travel in groups, limiting the ability to take advantage of the multiple platforms.

To eliminate the need to remain in groups, many approaches allow robots to localize relative to each other when they meet. Simple geometric localization using range and bearing measurements [24], [62] and using only bearing measurements from multiple robots to multiple landmarks [28] allow robots to update their poses. MCL has been implemented on a small team of robots operating in an indoor environment using a grid cell occupancy map [36]. Single-robot KBFs can also incorporate opportunistic information from teammates' localization by relating themselves through commonly observed landmarks [51], [52]. A single KBF can simultaneously track the positions of all robots in the team, including the cross-correlations between robots [110], [111]. Computation is distributed across the robots to reduce the considerable computational load. When teammates meet and observe each other, robots use their observations to cooperatively update their poses. Maximum Likelihood can also incorporate observations by and of teammates in the Bayes’ update [58]. CBL can also accommodate observations of and by teammates to refine pose estimates [128].

2.1.3 Limitations of Robot Localization Algorithms

While these approaches have typically been implemented on real systems with good results, localization alone cannot deal with unknown environments; if the locations of landmarks are not known, robots have nothing with which to localize. Addressing this problem is the technique of Simultaneous Localization and Mapping (2.3). Additionally, these approaches do not address the problem of choosing where a robot should move to remain well localized. Lastly, the only approach to localization which inherently provides a sliding scale of computational complexity versus accuracy is MCL, which typically is very computationally expensive for high accuracy. MVERT directly addresses the issues of where robots should move, for example, to remain localized. MVERT allows for adjusting complexity at several levels, and can integrate with localization alone or with simultaneous localization and mapping approaches (2.3).

2.2 Environment Mapping

Mapping is the task of autonomously locating objects and/or occupied space within an environment that is previously unknown or partially unknown.

2.2.1 Single-Robot Approaches to Environment Mapping

Approaches to mapping are similar to those applied to localization; instead of using relative measurements to known objects to locate the robot, relative measurements to unknown objects are used to locate the objects.

Basic geometric approaches can be applied to robot systems, where the relative positions of objects in the environment are combined with the robot's pose to locate them on the map. This can be employed for discrete targets in a continuous map space [22], [23], [28]. It can also be implemented using grid cell occupancy maps to locate occupied cells in the map [15], [17], [18], [43], [55], [62], [85], [86], [115], [134]. Occupied cells may be binary (0 is clear, 1 is occupied), or probabilistic (0-1 probability that the cell is occupied). Probabilistic grid cell maps are updated each time a cell is observed using simple probabilistic methods to combine the previous belief with the new sensor reading. Grid cell maps are typically applied to indoor environments, where objects (such as walls) tend to be large and continuous, rather than discrete. In some instances, grid cell maps represent navigability instead of occupancy. In these cases, each cell contains a number that represents whether that cell is navigable by the robot which can be binary (0 is perfectly navigable, 1 is completely impassible) or continuous (0-1 for level of navigability). This is similar to the typical grid cell map, with a slightly richer representation of space that is free for the robot to enter [39], [119].

Provided a robot can track its pose through an environment, the KBF approach can be used to locate objects within the robot's environment and generate a map [7], [32], [75], [89], [121]; this can also be implemented using only bearing measurements [22]. For large environments, this can become computationally expensive. To reduce computational requirements, maps may be broken into smaller, local maps that are later registered together to create a single large map [20], [46]. Similarly, MCL can be applied to localize objects relative to a known robot position, representing each object's position by a set of samples [15], [22], [134]. Maps are thus multiple sets of particles, with each object's position estimate computed as the average of its samples. E-M can also be used to generate maps in a similar manner, with positions represented in continuous space [15]. *Structure from motion* uses the relative change in object positions from one robot location to another to reproduce the structure of the environment and build a map [27]. Maximum Likelihood can also be applied to object positions and can track multiple hypotheses for each object [120].

Templates that can be used for localization of robots can also be used for map building, adding the templates online while the mapping continues [60], [104].

2.2.2 Multi-Robot Approaches to Environment Mapping

It may be desirable to apply multiple robots to the mapping problem in many cases because multiple robots can theoretically cover the space in a more efficient manner than an individual robot.

The typical approach to mapping with multiple robots is to divide the problem into several smaller, single-robot problems, referred to as "divide and conquer." The individual maps are later combined by joining common edges, or by registration techniques. The "divide and conquer" approach has been used to generate grid cell maps from multiple maps generated by individual robots [12], [16], [30], [60], [134]. This has also been applied using navigability grid cells [100]. The "divide and conquer" approach has also been applied to mapping using KBFs [32], [33], [46]. Each robot generates a map using KBF updates and the maps are later combined.

Multiple robots can update a common map (each robot maintaining its own copy) by adding data received by their own sensors and that communicated from teammates. Shared grid cell maps update probabilities of occupancy using the same process as in the single-robot approach [16], [17], [18], [85], [140]. This exchange may occur via general broadcast of all information, or a sharing of information when robots meet. The Information Filter form of the KBF, which allows multiple robots to update a common map by communicating less information, can also generate shared maps [130]. Finally, template approaches allow

multiple robots update to a common map by regularly registering individual robot local maps and broadcasting the updates [68], [135].

Another common approach to multi-robot mapping is the "leap-frog" approach [43], [55], [108]. To apply "leap-frog" to mapping, robots trade roles of explorer and landmark as in the single-robot case. An exploring robot updates the free space of the map, positioning it using its relative position to the free space and to the robots serving as landmarks. This has been in indoor environments with a grid cell map.

Templates (or models) for maps can be also updated using inputs from multiple robots [60], [104]. Bearing only measurements on many landmarks from multiple robots can also be used to map landmarks by doing a simultaneous optimization on error [28].

2.2.3 Limitations of Environmental Mapping Algorithms

Mapping approaches alone do not deal with robot pose uncertainty, though this problem has been dealt with using Simultaneous Localization and Mapping (Section 2.3). Additionally, like localization, the only approach that inherently allows for adjusting the computational complexity at the cost of accuracy is MCL. Mapping approaches do not address the issue of where robots should go in order to improve map quality. MVERT is designed to determine the best move to improve map quality, provided a measure of map quality.

2.3 Simultaneous Localization and Mapping (SLAM)

Simultaneous Localization and Mapping (SLAM), also called *Concurrent Mapping and Localization* (CML), is the process of simultaneously determining robot pose and landmark location within an initially unknown environment. The SLAM approach may be required for robots to operate in unknown environments efficiently, for long periods of time, or over large distances. This requires iteratively refining estimates over time and more relative measurements are made between objects in the environment. Most mapping approaches can be adjusted to perform localization concurrently.

2.3.1 Single-Robot Approaches to SLAM

The inherent ability of KBFs to track cross-correlations between observations and incrementally add new information makes them a common choice for SLAM. Each measurement can update the position of all landmarks and the robot due to the cross-correlations. This has been implemented for robots using typical sensors that provide range and bearing [7] [33] [75], as well as for sensors that provide bearings only [22]. Templates can be applied in a SLAM-type KBF update [60], [104]. A more efficient (and thus scalable) approach to SLAM uses factorability and a hybrid particle filter/EKBF approach to reduce computation [82].

2.3.2 Multi-Robot Approaches to SLAM

Multi-Robot SLAM allows teammates to not only take advantage of shared information about landmark locations, but to also use teammate positions. "Leap-Frog" can be applied to SLAM by assigning a single robot's initial position as the origin of the global frame [43], [55], [108]. Simultaneous optimization of error on multiple views from multiple robots on multiple landmarks can localize a team [28]. Multi-robot KBFs have also been applied to SLAM [35], [98], as has another filter that estimates rather than computing cross-correlations when they are unknown [113]. Common grid cell maps, generated by registering local grid cell maps, can similarly simultaneously localize individual robots [68], [135].

2.3.3 Limitations of SLAM Algorithms

Much work in mapping has relied on using coverage patterns to map the space, which may make sense for indoor environments or cases where a complete representation of the navigable space is needed. For outdoor environments and landmark-based maps, however, coverage patterns may not produce the most efficient path for exploration. Additionally, implementations of SLAM do not provide for adjusting the balance of computational complexity with accuracy except in the case of MCL, which does not scale well with large team sizes or large areas. Lastly, SLAM (and other mapping approaches) does not consider the actions of teammates beyond dividing the space or using teammates as temporary landmarks. MVERT, while not directly addressing the process of updating maps from measurements, uses the model for updating maps to choose which next step would provide the most improvement in the map, given a

measure of map quality. This choice includes choosing the move that will best improve the map *in conjunction* with the expected contributions of teammates. MVERT also allows for easily adjusting the level of complexity and performance to suit the task and platform capabilities.

2.4 Dynamic Target Observation and Tracking

Dynamic target observation is the process of keeping a moving target under observation. *Dynamic target tracking* is the process of continuously locating a moving object as it moves through the environment. Tracking may also include estimating the object's motion parameters.

2.4.1 Single-Robot Approaches to Observation and Tracking

The ability of the KBF to include time dependence by making predictions of future positions allow KBFs to be easily applied to the target tracking problem. The KBF has been applied to predict and observe the behavior of moving objects from robot platforms [31], [50], [54], [109], [132]. Multiple objects can be tracked [120], [132], and multiple hypotheses can be tracked for a single object [19], [131]. They have also been applied to real-time object tracking and camera pointing for object observation [139] and to tracking for grasp using proximity sensors [103]. Particle filters (MCL) have also been applied in an efficient manner to tracking single objects (and theoretically multiple) [83].

2.4.2 Multi-Robot Approaches to Observation and Tracking

In some cases, objects have been tracked using multiple robots, combining the information from the different robots. The KBF is applied to multi-robot target tracking by simply performing an update for each robot observing the target. This was applied to tracking a soccer ball for a RoboCup team [31]. Simple multiplication of Gaussian observations from multiple robots, without doing prediction, creates an instantaneous observation in my work in observation merging [124], [126].

2.4.3 Limitations to Observation and Tracking Algorithms

Approaches to dynamic target tracking do not take into account where robots should move to obtain the best measurements on the targets, nor do they consider the expected future contributions of teammate measurements. As in the case of mapping, MVERT addresses the choice of where a robot should move to, along with its teammates, produce the best set of observations on the set of targets.

2.5 Distributed Sensing

Distributed sensing is the task of combining the sensory inputs from multiple sensors. In this context, the multiple sensors are mounted on multiple robotic platforms, and the sensory information integrated is the location of objects within the environment.

The simplest approach to distributed sensing is to fill in parts of the environment that have not been observed by a robot with information from teammates. For example, robots can locate a ball that is not directly seen in robot soccer [13], [138]. Combining multiple views to improve the estimated positions of static targets can be computed efficiently by using multiple robot platforms to collect the multiple views [13], [21], [47].

A different type of multi-robot target tracking is the coverage task. The coverage task entails using a team of robots to keep a set of targets under observation. The problem is the choice of where robots should move to maintain coverage. Robots can maintain observation of several static targets in simulation [8] or moving targets in a real system [98], [97]. In both cases, locations of targets are obtained by the single observing robot rather than by combining observations from multiple platforms. Coverage may also be applied to completely covering a space for observation, such as a sensor net [56].

2.5.1 Limitations to Distributed Sensing Approaches

As with the case of dynamic target tracking, distributed sensing approaches do not consider where to move or the expected future contributions of teammate measurements except in the case of covering a space, whereas this is the specific goal of MVERT.

2.6 Multi-Robot Task Allocation

Task allocation is the process of assigning of a set of tasks among a set of agents. *Multi-robot task allocation* is the specific case in which the agents are a group of robots, which may be homogeneous or heterogeneous. For some types of tasks, such as mapping or foraging, tasks can be allocated by geometrically dividing up the space KBFs [12], [16], [30], [32], [33], [46], [100], [134], [114].

Another commonly used approach is to assign a value or cost to each robot's ability to perform a task and assign task based on optimizing this value or cost. Differences arise in the methods for developing these value functions and the amount of information that is shared. Values may be determined using pre-defined functions [29], [30], [40], [79], [93], [143], or may be learned through experience [94], [95]. These values may also include models of teammate performance in the case of distributed task allocation [94], [95]. For optimal solutions, a central processor will take the robot's individual estimates of cost and value and assign tasks to globally optimize performance. This can be performed dynamically by using an auction-based market approach where the central processor is the auctioneer [29], [30], [40], [143]. Local negotiation between agents allows for coordination or execution of tasks that require multiple robots [63]. The simple mathematical nature of value functions allows for quick computation. In a more behavioral framework [94], [95], robots can make decisions quickly by looking only at their own state and relevant teammate states to choose a single task. In systems that look for a simultaneous global optimization of all robots and all tasks [29], [30], [143], the complexity for determining a solution increases quickly with the number of robots and tasks.

A third approach is to use more traditional planning methods applied to the high-dimensional multi-robot space. This requires pre-defined models of each robot's performance, and may be computed in a distributed or centralized manner [12], [118], [133]. As the number of tasks and robots increase, the search space for planners increases quickly, making it impractical to implement for on-line planning in large teams or systems with large numbers of tasks.

2.6.1 Limitations to Multi-Robot Task Allocation Algorithms

These approaches to multi-robot task allocation presently look only at very simplified problems: either allocating a group of identical types of tasks (such as mapping a target area) of limiting the team size and number of possible tasks and planning at a high-level. This is due to the increased complexity for considering all possible allocation schemes in order to find the best. While MVERT does not address the issues of optimizing task allocation among robots, it is designed to integrate with a task allocation algorithm by altering the value functions used for decision making to reflect assigned tasks. MVERT can, however, perform simple greedy task allocation by allocating tasks to the robot with the highest instantaneous ability to contribute to that task (based on simple mathematical value).

2.7 Action Selection

Action selection is the process by which a robot chooses which action it should next choose. Actions may be considered at two levels, high and low. High-level actions may be defined as choosing between abstract activities (go to a location, analyze target, etc), while low-level action selection deals with the next physical action (drive forward, turn, sense). In this context, low-level action selection is primarily considered. It is focused on selecting the choice of movement actions and in selecting actions for multiple agents on a team.

2.7.1 Multi-Robot Action Selection

For the multi-robot domain, due to its high complexity, many systems use only a behavior-based framework, like motor schemas [2] or Subsumption [9], which maps the current belief to an action in a pre-defined way. Behavior-based approaches have been used to apply multi-robot teams to simple domains such as foraging [5], [77], [114] and exploration [143], and to complex dynamic domains such as robotic soccer [13], [31], [137], [138], box pushing [78], [94], [95]. In some cases, traditional planning approaches have been applied to high-level action selection [12], [118], [133], while the low-level is handled by a behavior-based execution level. Planning approaches may be applied also to minimize interference between agents while executing tasks [142]. Learning sequences of actions to maximize overall team reward has also been applied to robot action selection [136]. Behavior-based systems are extremely efficient and fast, making them ideal for dynamic environments.

2.7.2 Action Selection for Information Gain

Action selection for information gain implies that the robot will choose its action in order to maximize the information gained as a result of that action. Two types of information gain are explored: area covered by the system and quality of map generated by the system.

For coverage, the task is to choose a point from which to take an observation to add the most area to the map. Typically, this means choosing a point on the frontier, or on the edge of the known map, that maximally extends to the map [92], [115] or extends to the map in a random way [143]. For the particular application of (planetary) exploration, types of information may be prioritized to give weight to areas that might be "interesting" as defined by pre-existing criteria [84]. A slightly different approach to coverage is the sensor-net, in which a team of robots must choose where to go to ensure that the entire task space is observed by the sensors [56] using dispersion methods, such as potential fields. Similarly, coverage may deploy multiple robots to keep a set of moving targets under observation by using sensor footprints to estimate optimal robot locations [97], [98]. Since each of these approaches is looking ahead only one step, they can be extremely computationally efficient.

For map quality, the task is to choose the next measurement that will best improve the uncertainty in the current map, rather than that which most expands the map. For a single robot, Next Best View chooses the next camera angle to most improve a three-dimensional model of an observed object [105]. Two approaches to multi-robot map building have applied information gain techniques. The first represents targets by a particle filter (similar to that of MCL) and searches the space over all possible sets of actions for the team to most reduce the uncertainty in the next step, provided sensor models [123]. The second allows each robot to choose the action that will most improve the Gaussian uncertainty in the map using the Information form of the KBF. Robots iteratively exchange best moves until a consensus is agreed upon for the team [130]. A related approach in earlier work related to this thesis chooses the globally optimum set of observation points in a discrete space for a team of robots to observe a set of targets [124], [125]. All of these approaches are extremely expensive in computation (the fully optimal approaches in particular), making them impractical to apply to large teams or large numbers of targets. This is especially true for dynamic environments or robots moving at faster speeds, which need to be more reactive and able to select actions more quickly.

2.7.3 Limitations to Action Selection Approaches

These approaches to action selection do attempt to account for teammate contributions. However, these approaches are complex, requiring a large amount of computation for large teams or large numbers of actions. This is true both for planning over a large space or for optimizing high-dimensional value functions. MVERT attempts to directly address action selection problems, looking specifically at actions as the low-level next motion step. MVERT chooses a move action from a discrete set of candidate move actions to best complement the current state and the expected contributions to the state of teammates in the next step.

2.8 Summary and Discussion

The large body of work in each of the areas of localization, mapping, simultaneous localization and mapping, target tracking, multi-robot task allocation, and multi-robot action selection has provided a solid foundation for implementing practical physical systems. However, at present each of these has only been implemented as stand alone tasks. For systems to handle missions such as planetary exploration, search and rescue, and other complex tasks, it may be impossible to perform each of these alone. Instead, it may be necessary and desirable to concurrently localize, map, track moving targets, maintain communications, and perform other mission-related tasks. Thus, it is necessary to develop a framework for integrating these individual tasks that can allow robots to select actions that are good for the team without planning beyond the time available.

While choosing an action that benefits the team has been investigated and applied, these approaches are typically very computationally expensive because they try to find optimal solutions. It is desirable to find methods of approximating an optimal solution with a sliding scale of complexity that can be adapted to the accuracy demanded in the task and to the computational resources allowed while still taking advantage of the robot as part of a team. Some preliminary work related to this dissertation has looked at choosing an

action to best improve map quality for the team [124], [129], [130]. This work (Value-Based Observation for Robot Teams, or VBORT) approximates the one-step team optimal and discretizes the search space in order to make the action selection tractable enough for large environments.

Chapter 3 Approach: MVERT Architecture

3.1 Theory

Real multi-robot missions include multiple tasks, each of which may call for the robot to move in different directions. Examples of such tasks potentially important in many multi-robot missions include mapping, exploration, target sampling or manipulation (target analysis, target manipulation, etcetera), and maintaining communication. Conflicts arise in attempting to satisfy each of these tasks: target sampling tasks require remaining stationary while mapping encourages fast movement; communication may require remaining nearby or in line-of-sight, while exploration can occur faster if robots spread out. Additionally, there may be the added complexity of heterogeneity of robots in sensing, mobility, or task capabilities. Action selection (action defined as a movement) becomes difficult.

The goal of MVERT is to enable robots on a team to efficiently choose actions that contribute most to overall mission achievement and to better take advantage of teammates and transitory opportunities without requiring detailed planning or replanning. Many execution approaches are behavior-based and select actions by choosing a dominant goal to take control (perhaps defined by a planner), by combining the best moves for each goal into a single average move, or by allowing actions to assign votes to each action (most votes wins). By assigning (dynamic) values to different tasks one can optimize overall performance at each step by choosing the action that produces the best total result. In MVERT, each action is assigned a value (utility) by each task, and the total value of an action is a combination of the individual values. This value is relative to the expected teammate contributions in the next step. By evaluating total expected value for a discrete set of possible actions, overall performance can be optimized in the discrete space at each step. This type of compromise result can include any task or goal that may be represented or approximated by a mathematical function, making computation simple and fast as in behavior-based approaches.

MVERT is explicitly a distributed multi-robot approach to choosing a next-step move action for each robot team member. The value of potential actions can be dramatically affected by the actions of teammates; MVERT approximates the expected contributions of teammates at the next step to optimize relative to teammates. Working at the execution level allows MVERT to take greater advantage of the team without requiring explicit low-level planning for team interaction. MVERT may be integrated with a planner or pre-defined plan for additional capability (particularly for optimizing initial parameters and task allocation), but planning is not an explicit part of the MVERT execution architecture. Integration of MVERT with a typical robot control system is shown in Figure 2. This example control system consists of a World Model, a Plan, a Sequencer, and Executive/Execution modules. World Model updates may include robot pose localization, target and/or landmark location updates, and incorporating information from teammates. The

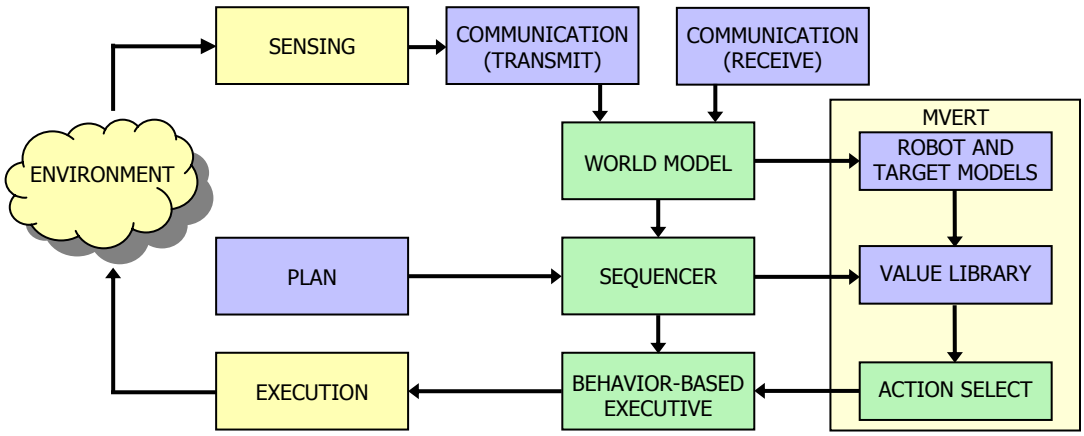

Figure 2. Robot Control System Block Diagram

MVERT may integrate into a robot control system. Blue (dark gray) represents data, green (medium gray) computation, and yellow (white) interaction with the environment. MVERT takes information from the world model and current priorities (from a plan) and recommends an action based on expected teammate contributions. Details of the robot control system implemented with MVERT for this work are shown in Figure 3. Low-level details of MVERT Action Select implementation are shown in Figure 4.

Plan is provided externally by a human or AI planner. The Sequencer (or scheduler) selects a high-level action to execute while the Executive selects how to execute it. MVERT recommends a low-level action to the execution level based on current state and expected teammate contributions. By making this evaluation at each step, given the current state, MVERT is adaptive and potentially self-correcting.

MVERT assumes a sequencer/scheduler (human or AI) has selected which tasks should be active at a particular time (which values should be considered). The MVERT approach assigns explicit mathematical value functions to each task. These value functions may depend on the state of the robot and its teammates (including pose uncertainty), the current map and its quality, specific mission goals (and their priorities), and time. These value functions, which take estimated future teammate contributions into account, may improve teamwork if robots have overlapping tasks or skills. Overall value for an action is a combination of the expected contributions by teammates and the expected contribution from executing the action. Teammate contributions are approximated using the known current state of the teammate and pre-defined models of its capabilities. In the absence of communication, if teammates can be observed and their poses estimated, teammate contributions may still be estimated, though with less certainty. If data is later combined, the full advantage of the team's contribution can be obtained.

In order to make optimization over actions relative to teammates tractable for practical implementation, a design goal of MVERT is to make an approximation to the one-step optimal. This approximation occurs by making the action space discrete and by approximating expected teammate contributions. Additionally, MVERT can provide a sliding level of complexity to adjust the computational complexity versus accuracy tradeoff along several dimensions to fit with the task, domain, and available computation. Examples of this sliding complexity include: the level of discretization of the action space, the prediction model for teammate contributions and candidate actions, the update model for incorporating observations, and the type of localization. Prediction and update models may be performed using any method that uses the desired representation: Gaussian distribution multiplication, Kalman-Bucy Filter (KBF), Monte Carlo (MC), SLAM, or Constraint-Based Localization (CBL, Chapter 6), for examples.

The current implementation incorporates MVERT into a simple robot control system. The implementation, including particular modules used, the types of data, and data flow, are shown in Figure 3.

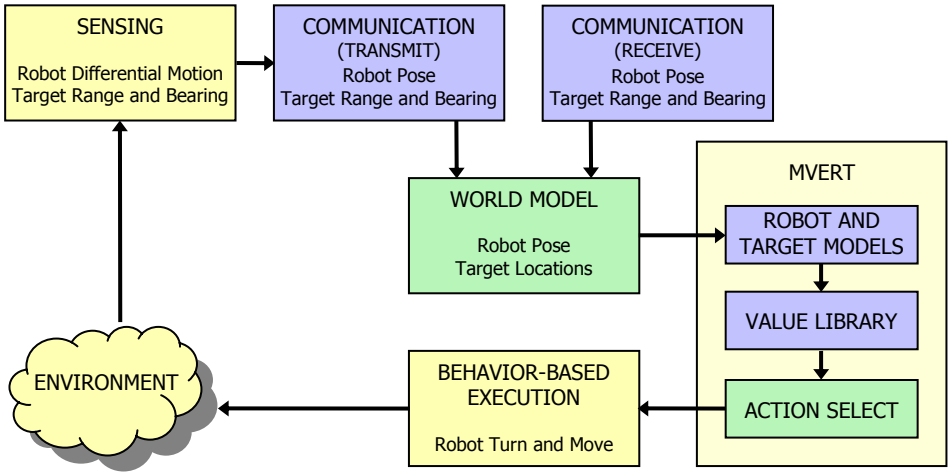

Figure 3. Implemented Robot Control System Block Diagram

The current robot control system implementation. Blue (dark gray) is data, green (medium gray) computations, and yellow (white) interaction with the environment. The plan is human provided and designed into the value library. MVERT alone selects actions. Details of data types are shown in each module. Details of MVERT Action Selection are shown in Figure 4.

MVERT may be considered behavior-based in the sense that individual task behaviors independently contribute to action selection. Action selection is based only on current state of the team and the environment, not on any historical information, and selects actions for only one step. However, unlike a purely behavior-based system, some reasoning is applied to assist in the decision. Each robot predicts teammate contributions using models and then reasons about the expected results of selecting different candidate actions. This greedy search is essentially a one-step plan in the low-level action space.

Three important assumptions must be satisfied for the MVERT approach:

- Robots have a common coordinate frame. This can be established a priori or by registering maps as robots come into contact with one another.
- Robots can obtain information about the state (at least pose) of teammates, either through communication or through observation and inference.
- Teammate state changes slowly with respect to cycle time and environment scale. This makes approximation of future teammate state by current state reasonable.

3.2 Action Selection

The purpose of MVERT is to select the action (move) that will produce the greatest contribution to the team for all active tasks, including an expectation of the contributions of the teammates in the next step. Rather than attempting to perform individual tasks sequentially (for example, first mapping, then target tracking), it is desired to find the move that allows the team to make the most progress towards all tasks simultaneously. Typical implementations that attempt to combine multiple objectives will choose the action based on "winner-take-all" (as in Subsumption [9]) or will average the actions selected by each task (as with motor schemas [2]). However, the first of these approaches may select actions that are very good for one task without making any contributions to the other tasks while the second, by averaging actions, may choose an action that is not highly productive for any of the tasks.

Ideally, the goal is to choose a sequence of team actions that optimizes the performance of the team over all tasks simultaneously. This requires looking at the total contribution of each potential set of actions toward all tasks and then choosing the action sets that produce the best total result through time. To examine an action's total contribution, the results of performing each task after taking that action must be considered. To optimize in this multi-dimensional space over all possible paths in order to find a global optimal is too computationally complex to implement on a real-time system in a dynamic environment. Planning even a one-step optimal over complex spaces may be impractical for large teams or large numbers of tasks.

The MVERT action selection approach takes information from a plan (generated either by a human or by an AI planner) that provides high-level task allocation to reduce mission cost. MVERT is applied during execution to create paths that attempt to optimize a robot's performance on all of its current tasks simultaneously, including anticipating teammate participation in overlapping tasks. In order to perform this optimization in an efficient manner so that it may be applied at each time-step during execution, the true optimal must be approximated. A summary of the present MVERT implementation is shown in Figure 4.

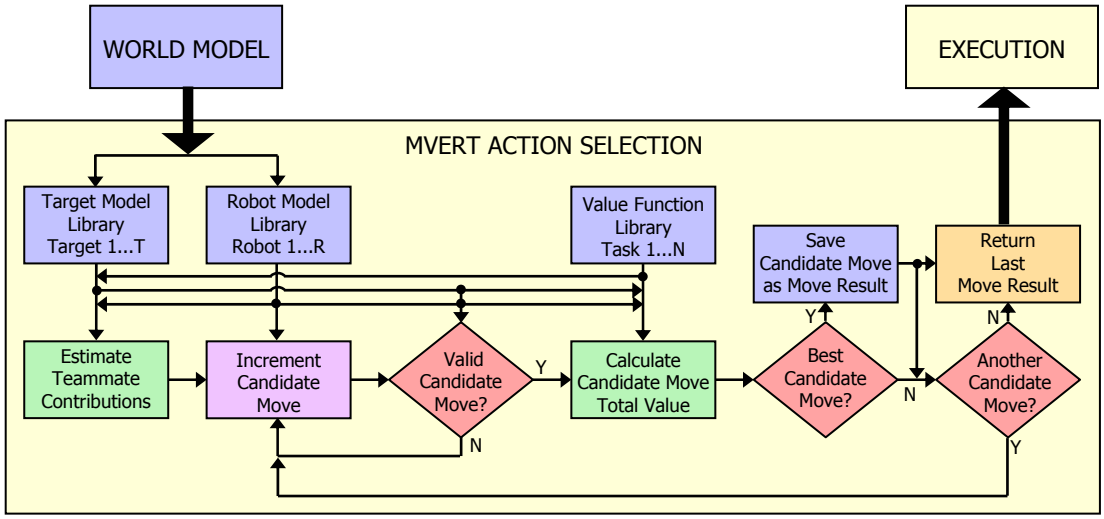

Figure 4. MVERT Action Selection Block Diagram
MVERT estimates teammate contributions, then iterates through candidate moves to choose the best, the one that provides the highest value over all tasks. Diamond blocks are decision points.

To detail the procedure in Figure 4, an example is provided here. The MVERT action selection process is called by the robot control architecture after the World Model module has been updated. This *world model* consists of target locations (with uncertainty) and robot poses (with uncertainty). The world model feeds into the current *target model library* (the current locations of all types of targets with uncertainty). The world model also feeds into the *robot model library*, which includes robot poses (with uncertainty) as well as pre-defined models of each robot's capabilities (the task they can perform) and their sensor models (noise, maximum range, angular field of view, etc.). Using the target and robot model library, the expected approximate contributions of teammates are computed in the *estimate teammate contributions* step. This computation applies data in the *target model library* and *robot model library* to the value functions in the *value function library* and obtains an expected belief and expected value. This expected belief and value therefore reflects the expected contributions from all teammates in the next step.

Once teammate contributions have been predicted, Figure 4 continues with the action selection process. Candidate moves are (in this implementation) defined as moving along a desired heading for one step. The robot individually takes each candidate move in the *increment candidate move* step, and first tests to see whether this move is valid in the *valid candidate move* test. This test uses the *robot model library* to test for violations of the robot's motion capabilities and uses the current *target model library* and *robot model library* to test for potential collisions with targets, obstacles, or teammates. Candidate moves that violate motion or collision constraints are discarded. For valid actions, the robot applies the current robot's pose information from the *robot model library* to functions in the *value function library* to obtain a value for that candidate action in the *calculate candidate move value* step. If this new candidate action provides a higher value (relative to the expected teammate contributions), this action and its value are saved. This iterative process continues until all of the candidate moves have been investigated. MVERT then returns the move that is currently saved as that with the highest value. This move is provided as input to the Execution module, which then translates the desired move into robot motion commands and monitors progress toward action completion.

MVERT approximates the one-step optimal by using several simplifications and assumptions. The central assumption is that the time between decisions at the execution level is small. During these small steps, teammates will move only small distances from their current locations. Thus, a teammate's expected contribution in the next time step can be approximated by an *average* expected contribution. This average contribution is computed as the contribution that the teammate would make from their current location, which is the average of its possible locations in the next step. The computation of this average contribution will only include the teammates' contributions in tasks that are relevant to the deciding robot's tasks. Teammates' contributions to the current robot's active tasks are estimated, while their contributions in other tasks are not estimated. Teammate actions are not predicted, only approximated by the *current* state. Thus, the irrelevant tasks have no effect on the prediction and can be ignored. This approximation is reasonable as long as teammates move a small distance relative to the scale of the environment, at each time step.

A second approximation to the one-step optimal is made by making the action space of each robot discrete. The amount of discretization can be geared to the task and to the computational resources available. If resources are limited, potential actions may be of a fixed step size with only a few candidate moves examined on the one-step circle. If resources are plentiful, many candidate moves may be considered, and may, if desired, consider multiple step sizes.

MVERT assumes that a measure of task performance exists that can be applied as a value function for evaluating moves. Individual value functions are designed for each task. They are designed to provide robots some guidance in performing tasks and are relevant to the tasks. These value functions may depend on the current state, time, and task/target priority. Dependence on state may include, among other things, the current uncertainty in the map, current uncertainty in robot pose, and relative positions of teammates. Dependence on time may allow for finite mission time. Task or target priority can affect value functions by increasing value of some tasks over others; for example important tasks may be heavily weighted for early execution. The detail of a value function must match the precision required for a task. The value functions proposed and used for this work are described in the following section.

3.3 Value Functions

In MVERT, for each candidate move, values for different tasks are combined using a weighted sum to compute the final value, V, of the candidate move as shown in Eq. 1:

$$V = \sum_i \kappa_i V_i \qquad \textbf{Eq. 1}$$

where V_i are individual value function values and κ_i are the weights given each value. Designing an individual value function for each task simplifies this aspect of the design process. This also allows flexibility in how the different tasks are combined and the number of tasks that are included in any single decision step.

Weights for different task value functions, κ_i, must be assigned. The weights of the individual values have been determined through hand-tuning and experimentation. Here, it is assumed that these weights are provided at run time by a human operator and remain static throughout the experiment. However, weights can be dynamically changed online to reflect changing mission goals or environmental conditions. Changes might be introduced by humans or by a planner, even making the instantaneous weighted sums follow non-linear functions if desired. Learning algorithms might also be introduced in order to reduce the burden of value function design, improve performance, and provide online adaptability. A detailed investigation of the effects of varying these parameters is beyond the scope of this thesis.

In the MVERT implementation used in this work, the primary task is mapping targets. The evaluation for performance is the uncertainty in the resulting position estimate of the targets, and the value function must minimize uncertainty. Additional tasks considered in this work include target sampling, exploration, and maintaining communication; each requires a value function to promote performing the task which reflects how well or poorly each action can contribute to the task.

3.3.1 Target Location Value Function

The purpose of the target location task is to produce the highest quality estimate of the position of all targets in the map. This means attempting to minimize the overall uncertainty in the map. In order to minimize uncertainty, any representation of uncertainty might be used. Some obvious possibilities include minimizing the area of the one-sigma ellipses (or 3-sigma ellipses) of the Gaussian distributions of targets or the worst-case uncertainty (the largest axis of the one- or three-sigma ellipses) of targets. It may be desirable to simultaneously reduce the uncertainty in the whole map (looking at the total area or total largest uncertainty) or to reduce the largest uncertainty in the map (the least well known target).

In this work, preliminary consideration of different value functions was made. For the implementation of MVERT in experimentation for this these, it is assumed that the mission goal is to minimize the area of uncertainty on all targets. Thus, the value function selected is the total area of the one-sigma ellipses for all targets, and is shown in Eq. 2.

$$V_{TL} = -\sum_i \pi \sigma_{i\,maj} \sigma_{i\,min} \qquad \textbf{Eq. 2}$$

where $\sigma_{i\,maj}$ and $\sigma_{i\,min}$ are the standard deviations of the Gaussian distribution along its major and minor axes (the largest and smallest axes of the one-σ ellipse) for the i^{th} target, and π=3.14 to complete the area formula. This equation represents the area of an ellipse, though relative values would be equally valid without the constant π. The negative sign penalizes large areas greater than small areas, giving more certain distributions higher value. Thus, as uncertainty is reduced and the area of the ellipse decreases, the value increases. The total value is zero if there are no targets observed. The prediction of what will be observed from a location uses the sensor model and, if desired, models of occlusion by obstacles. The prediction of the resulting measurements also uses the sensor model and the predicted result is combined with the prior result to predict the effect of the measurement.

As an example application of this value function, consider two targets that have been observed and have positions and uncertainty represented by the two-dimensional Gaussian distributions shown in Figure 5.

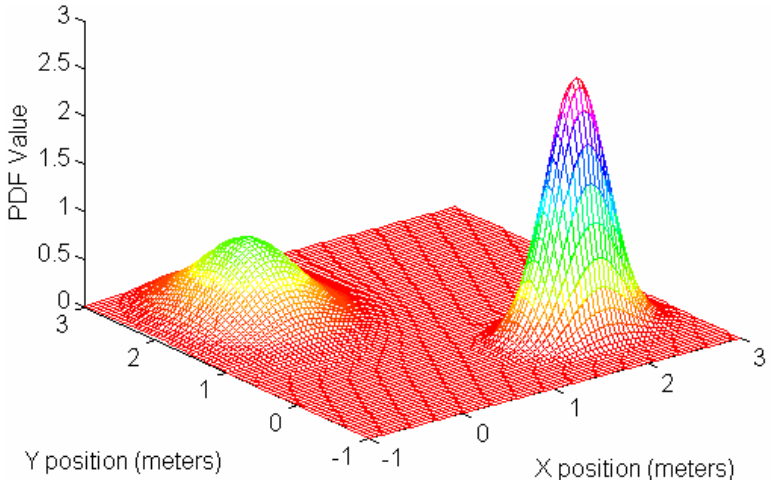

Figure 5. Target Location Example PDF
The summed PDF for two example targets is shown. One target (left) is less well localized than the other target (right), indicated by the wider spread and lower peak of the Gaussian distribution compared to the other target.

Applying V_{TL}, Eq. 2, the value surface shown in Figure 6 is obtained. The maximum observation value (discrete space with resolution of 0.05 meters) occurs at (1, 1.05). This point maximizes the angle of each observation relative to the major axis of each distribution, and minimizes distance to each target with a slight bias toward the less well-localized target.

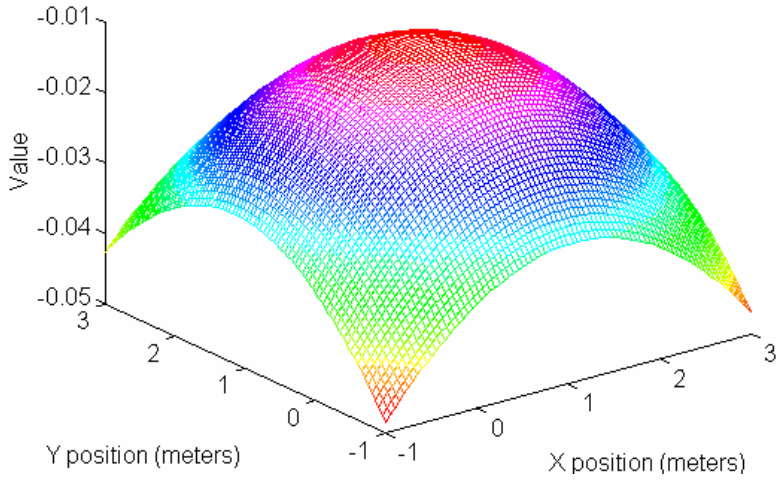

Figure 6. Target Location Example Value Surface
The value of V_{TL} at each point in a discrete space (resolution 0.05 meters), showing a maximum value of -0.0106 for observing from the point (1, 1.05). This point is aligned to see across the long axes of both distributions and is slightly closer to the less well known target.

Maximizing value serves to reduce the product of the major and minor axis standard deviations as far as possible. Sensitivity of V_{TL} to changes in the standard deviations is illustrated in Figure 7. The direction of steepest increase in gradient of V_{TL} (Θ) is given by the partial derivatives as shown in Eq. 3:

$$\Theta = \text{atan2}\left(-\pi\sigma_1, -\pi\sigma_2\right) \qquad \textbf{Eq. 3}$$

where σ_1 and σ_2 are either σ_{maj} or σ_{min}. Thus, maximizing V_{TL} will decrease σ_1 more strongly as σ_2 increases. Observe that for σ_2 larger than σ_1, the steepest ascending slope moves σ_1 more toward zero than σ_2 ($\pi\sigma_2 > \pi\sigma_1$), minimizing the smaller σ_1 more. If σ_1 and σ_2 are equal, both are minimized equally ($\pi\sigma_2 = \pi\sigma_1$). As σ_2 decreases toward zero, changes in σ_1 do not significantly affect value, while small changes in σ_2 produce large effects ($\pi\sigma_2 \ll \pi\sigma_1$). Thus maximizing value minimizes the smaller σ_2 more. Note the nearly flat slope along the line where each standard deviation is almost zero, and steep slope in value moving away from both of these lines.

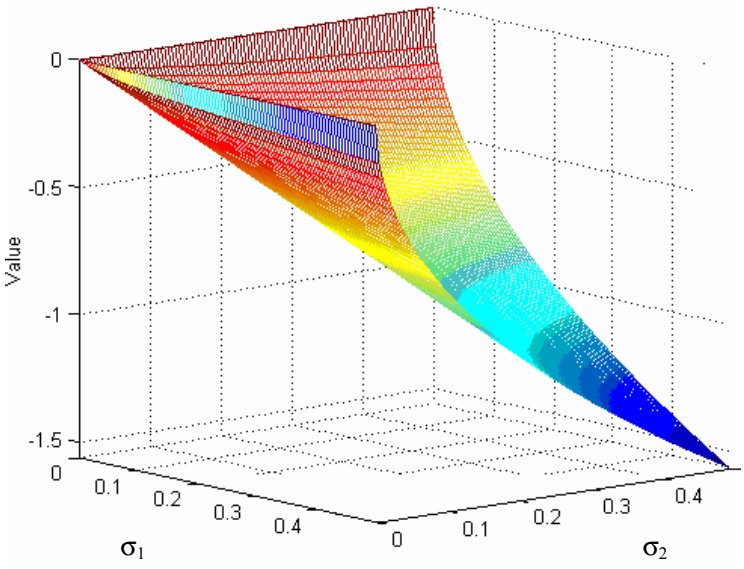

Figure 7. Value as a Function of Standard Deviations

The value of V_{TL} as a function of major and minor axis standard deviations. Due to the product in Eq. 2, maximizing value will minimize the smaller standard deviation more. The smallest standard deviation is minimized most when it is close to zero (observe the steep slope away from σ_1=0 and σ_2=0 lines).

3.3.2 Target Sampling Task Value Function

Target sampling is a model for several types of tasks, and is analogous to consumption described by Balch and Arkin [5]. Any task that requires a robot to remain at a fixed location for a time to perform work at a target can be modeled as a target sampling task. Here, target sampling is used to model target analysis, where robots must perform some kind of non-destructive analysis of the specified target position. Target sampling tasks have two phases: 1) approaching the target, and 2) sampling the target. Approach ends when the robot is within a threshold, λ, of the target. Sampling the target requires that the robot must remain in place until the task is completed. Once the target is sampled, the target sampling task for that target should contribute no further value in the decision process, as no further mission value can be obtained in that task. Until completion, sampling value V_S is computed as in Eq. 4, where d is the distance from the robot to the target and λ is a threshold distance below which sampling can be performed.

$$V_S = \begin{cases} V_{SA}, & \text{for } d > \lambda \\ V_{SE}, & \text{for } d <= \lambda \end{cases} \qquad \textbf{Eq. 4}$$

For approaching a target, the maximizing the value must draw the robot closer to the target. Thus, positive value is assigned for moving closer to the task location and negative value is assigned if moving away from the task location. The larger the change is in distance, the larger the change is in value. This value is thus based on change in distance from the robot to the target, and shown as V_{SA} in Eq. 5:

$$V_{SA} = \sqrt{(X_O - X_T)^2 + (Y_O - Y_T)^2} - \sqrt{(X_C - X_T)^2 + (Y_C - Y_T)^2} \quad \textbf{Eq. 5}$$

where (X_O, Y_O) is the robot's current location, (X_C, Y_C) is the candidate location, and (X_T, Y_T) is the location of the currently assigned sampling task location. In this way, if the candidate location represents a reduction in distance, a positive value is obtained while locations further away produce a negative value. Thus, locations moving further from the sampling point are penalized, but not eliminated from consideration.

Once the target has been reached, within a given threshold, target sampling must begin execution. For target sampling, value must hold the robot in place until the task is completed. Thus, for the target sampling value during execution, V_{SE}, there is a large positive constant value (K_S) for the robot's current location and a large negative value for all other points, Eq. 6. This large constant can be designed to be large enough to override the values of any other tasks, except where desired.

$$V_{SE} = \begin{cases} +K_S & \text{for } (X_C, Y_C) = (X_O, Y_O) \\ -K_S & \text{for } (X_C, Y_C) \neq (X_O, Y_O) \end{cases} \quad \textbf{Eq. 6}$$

The selection of the current target can be assigned by a plan (generated by human or AI) or determined using MVERT as the target providing the highest greedy *team* value in the next step. This is the approach implemented here, and is described in 3.4.5.

3.3.3 Network Connectivity (Communications)

In order to take advantage of the team at execution time, robots must maintain communication whenever possible. This typically requires remaining within line of sight (not occluded by obstacles) of teammates. Effects due to limited range and directional variation are ignored here. The network connectivity value function (V_N) is defined in Eq. 7.

$$V_N = \frac{R_C}{R_T} \quad \textbf{Eq. 7}$$

where R_C is the total number of teammates that are reachable by the robot (including the robot) and R_T is the total number of robots on the team. A teammate is considered reachable if it is within line of sight of the robot or within line of sight of one other reachable teammate (one step relay only). Obstacles can occlude line of sight. A teammate is not visible if the obstacle subtends an angle the covers the angle to the teammate. Given a candidate robot position R, a teammate position T, and an obstacle position O with radius O_r, visibility, S, is determined as in Eq. 8 - Eq. 11.

$$A_{TR} = \tan^{-1}\left(\frac{T_y - R_y}{T_x - R_x}\right) \quad \textbf{Eq. 8}$$

$$A_{OR} = \tan^{-1}\left(\frac{O_y - R_y}{O_x - R_x}\right) \quad \textbf{Eq. 9}$$

$$dA = \left|\sin^{-1}\left(\frac{O_r}{\sqrt{(O_x - R_x)^2 + (O_y - R_y)^2}}\right)\right| \quad \textbf{Eq. 10}$$

$$S = \begin{cases} 0, \text{ if } |A_{TR} - A_{OR}| \leq dA \\ 1, \text{ otherwise} \end{cases} \quad \textbf{Eq. 11}$$

where S=1 indicates the teammate can be seen by the robot and S=0 indicates that the teammate cannot be seen by the robot.

An example of the network value computation is shown in Figure 8. A fully connected network (left) has a value of 1.0 and a partially connected network (right) has a value below 1.0.

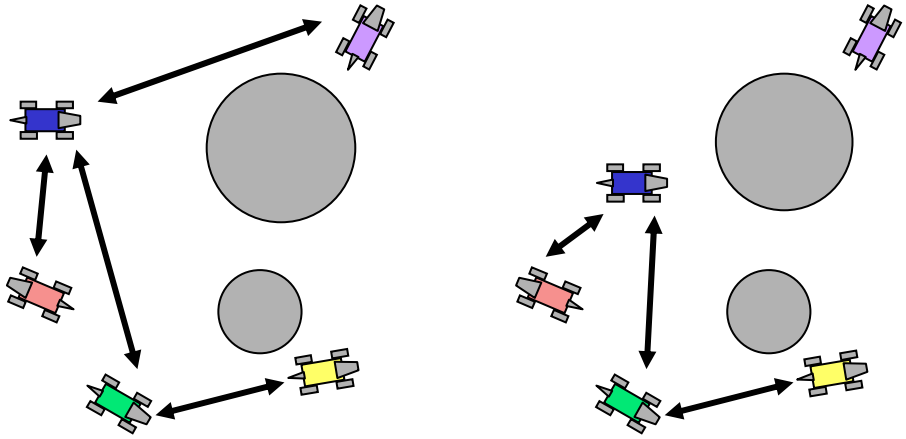

Figure 8. Computing the Network Value Function
Computing network connectivity from the point of view of the upper left robot (blue). *Left:* Three teammates are directly visible by the top left (blue) robot, and one (lower right) is visible through one-step relay. This full connectivity leads to a value of 1.0. *Right:* Two teammates are directly visible by the top left (blue) robot and one is visible through one-step relay (bottom right). A last teammate (top right) is no longer visible, directly or indirectly through one step relay, making the value only 0.8.

3.3.4 Exploration

Exploration encourages robots to enter unknown territory within the scope of the mission. Exploration is encouraged by assigning exploration points, locations to which robots should move in order to guarantee that the space is covered. These exploration points are defined similarly to target sampling task locations, but do not require the robot to remain for any duration. Also, the tolerance for reaching these exploration points is much greater, as the objective is to move robots into a specified region, not exactly to a specific point. The value for exploration is also based on distance from the target points, as in Eq. 12.

$$V_E = \sqrt{(X_O - X_T)^2 + (Y_O - Y_T)^2} - \sqrt{(X_C - X_T)^2 + (Y_C - Y_T)^2} \qquad \textbf{Eq. 12}$$

where (X_O, Y_O) is the robot's current location, (X_C, Y_C) is the candidate location, and (X_T, Y_T) is the location of the current exploration point. As in the case with sampling, this value function favors locations that move the robot closer to the exploration point and penalizes locations that move the robot further away.

3.3.5 Discussion of Value Function Selection

The value functions presented here are specifically designed to be very simple representations of the quality of the team's performance at each task. The precise nature of these value functions can be tuned to match the needs of the mission, adding accuracy or detail at the potential cost of losing efficiency. For example, predicting a SLAM update could more precisely predict mapping contributions when using SLAM, and a more complex model of communication (including range, signal strength, errors, and losses) could provide better overall network connectivity.

3.4 Implementation Issues

3.4.1 Code

MVERT is specifically intended to be a distributed multi-robot architecture. In the present implementation of MVERT, the distributed nature of computation is simulated in a single *Matlab* process that independently performs computations for all robots; the computation for each robot only uses information that would be explicitly communicated by teammates. The single process is used to increase portability between platforms and ease of implementation. Robots move synchronously; this choice is made for simplicity of implementation on robots using the single Matlab process. Each step is executed simultaneously for each robot: collecting measurements and localization, sharing information, action selection, and action execution. This use of synchronous motion is not an inherent restriction of MVERT. To accommodate asynchronous motion, each robot would make decisions based on the information currently available. As new measurements arrive from teammates, they can be accounted for in the map (using either Gaussian distribution multiplication, or SLAM) to keep the map common to all robots. Data that arrives during the decision making process would be ignored until the decision was completed to keep the decision making process consistent. For dynamic targets, old measurements cannot be easily accommodated by either the Gaussian multiplication or KBF tracking approaches and must be discarded during on-line operation. A posteriori, all of the individual measurements can be incorporated into the model in order to make a final, single track for each target.

3.4.2 Representation

In order to reduce the computational requirements of MVERT, the present implementation uses Gaussian distribution multiplication of predicted measurements to predict teammate contributions and resulting position estimates of targets from candidate positions (Section 4.2). In target location and dynamic target tracking tasks, which assume known landmarks, Probabilistic Constraint-Based Localization (Chapter 6) is used for robot pose updates and Gaussian multiplication (Section 4.2) is used for target position updates. For mapping tasks, SLAM (Section 4.6) is used for target position and robot pose estimates, but Gaussian multiplication is used for the teammate contribution and candidate move contribution prediction stage because of its lesser complexity than SLAM updates.

3.4.3 Local Optima

In order to prevent oscillations around local optima, robots are prevented from considering moving back to a point it occupied in the previous step. This is implemented by not allowing consideration of differential headings of π, moving exactly backward to its previous location, even if the robot is considered holonomic. Additionally, except when remaining stationary to complete a target sampling task, robots are not allowed to consider their present location as a potential move. Robots only remain in place if no valid moves are available; all moves lead to a collision or violate robot motion constraints. Robots are only allowed to take one set of measurements from a location to prevent artificially reducing uncertainty through repetitive dependent measurements.

When evaluating potential moves for value, multiple moves may provide the same overall value. This may be due to symmetry of target and robot positions, and may often occur when some known targets are out of visual range of robots with limited sensing. In such cases, ties are broken by choosing the candidate move that minimizes the total distance to all known targets (whether or not they are observable). In this way, robots may improve contributions at a later time, by bringing the unobserved targets into view. In the case when no targets are visible and no targets are known, making a knowledgeable decision on movement impossible, each robot chooses a move at random from a list of potential moves. This list of moves includes: move straight, turn left, and turn right. The distribution is biased so as to produce movement straight ahead 80% of the time, turn left 10% of the time, and turn right 10% of the time. The bias toward moving ahead encourages robots to explore farther in hopes of finding targets sooner.

3.4.4 Collision Avoidance

Buffer zones around known targets, landmarks, teammates and obstacles, prevent collisions by not permitting a robot to enter the buffer zones. In simulation, targets and landmarks are treated as points, and thus require a buffer equal to the robot's size. Simulated obstacles and robots have size; their buffers are

their radius/size plus the robot's size. In the physical system, robots cannot observe targets at close distances, as the targets are larger than the field of view at close range. In this case, the buffer is set to prevent the robot from moving closer than where objects are observable.

3.4.5 Target Sampling Target Selection

Using the MVERT approach, robots estimate team contributions in the next step and select a target for sampling that will best serve the team. The allocation begins by determining the closest sampling target to the robot. If no teammates able to perform the target sampling task are closer to this target than the robot, this target is assigned to the robot. If closer teammates exist, more detailed computation is performed iteratively until the robot can claim an unassigned target. In this case, all robot-target distances are computed for sample-capable robots and sampling targets. The smallest robot-target distance assigned that target to that robot; similarly, the next-smallest robot-target distance determines the next target allocation. For each robot, the process ends when it reaches and claims an unassigned target. By greedily assigning tasks in this manner, each robot is uniquely assigned a target that will benefit the team performance. This approach to target selection is chosen to fit within the MVERT framework. Each robot can predict which targets teammates will most contribute to based on proximity and the value of selecting those targets expected to be sampled by teammates becomes zero. This process is illustrated in Figure 9.

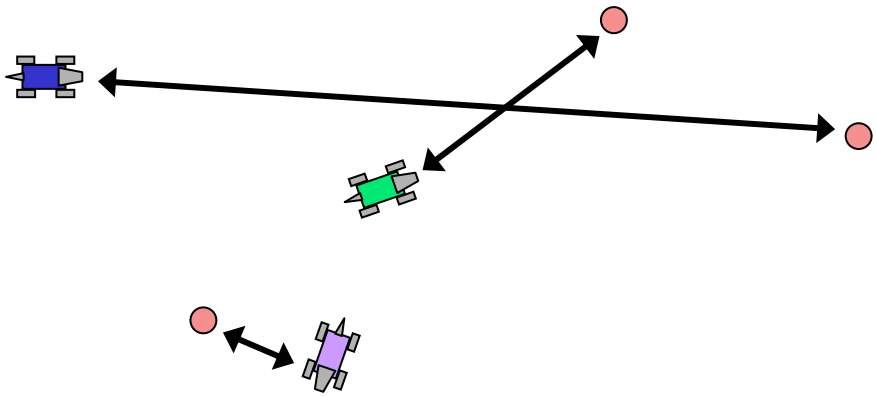

Figure 9. Sampling Task Allocation Approach

Task allocation example for three robots and three targets. The lower left target is closest to all robots. If the middle (green) robot is selecting a task, it will disregard the closer lower left target because the lower robot (purple) and that target are each other's closest match. The middle robot can then assume the lower robot will contribute sampling that target.

3.4.6 Terrain

This implementation assumes a smooth, locally flat terrain that can be represented two-dimensionally. Large discontinuities in terrain are modeled as obstacles. This simplifies the representations and reduces complexity of visibility (line of sight) computations. This assumption is reasonable for many planetary environments (including Earth), such as the smooth terrain with rocks illustrated in Figure 1.

3.4.7 Measurement Independence

To ensure that measurements are independent (an assumption of both Kalman updates in Section 4.2 and Simultaneous Localization and Mapping updates in Section 4.6), the robots must move before taking additional measurements on target landmarks and obstacles. If a robot does not move (to avoid collision or to make progress in a sampling task, for example), additional visual observations on targets are not incorporated into the world model.

3.5 Summary and Discussion

MVERT is a distributed multi-robot architecture for low-level action selection during execution. It is a hybrid behavior-based and planning approach that predicts approximate teammate contributions and selects an action that maximizes team progress using a greedy search. Mission progress is defined by a weighted sum of value functions representing performance for each individual mission task. The work presented here primarily addresses exploration-type missions, focusing on the tasks of mapping, exploration, target tracking, target sampling, and maintaining network connectivity. Value functions for these tasks are simple representations of task performance, such as reduction in uncertainty for mapping and tracking tasks, proximity to target locations for sampling and exploration tasks, and fraction of team network connectivity.

Chapter 4 Treatment of Uncertainty

In robotic systems, there is inherent uncertainty in the world. The types of uncertainty addressed in this work are:

- Uncertainty in robot sensing,
- Uncertainty in robot motion,
- Uncertainty in robot pose,
- Uncertainty in target position.

Uncertainty in robot sensing arises due to pixel resolution in camera images, imperfect color calibration, lighting variations, and imperfect sensor modeling. Uncertainty in robot motion arises due to slippage against the surface and imperfect modeling of robot joint motions. Uncertainty in robot pose arises from initial uncertainty in placing the robot, subsequent uncertainty in the robot's differential motion used for the process update in localization, and uncertainty in sensing of landmarks used for the sensor update in localization. Uncertainty in target position arises due to the uncertainty in robot sensing of targets and uncertainty in the robot pose.

4.1 Assumptions

Some assumptions are made in the treatment of uncertainty in order to simplify the mathematical complexity of the approach. The first assumption is that all uncertainty models can be approximated as Gaussian distributions, over one or more variables. This includes individual measurements from the sensors (range and bearing), measurements of differential motion (distance and heading), landmark and target positions (X and Y), and robot pose (X, Y, and θ). This assumption allows use of simple Kalman-Bucy equations to combine estimates.

The second assumption is that each measurement is independent of all other measurements and variables. Each measurement (range and bearing to a single target, position estimates of the same target from different platforms or different locations) is independent of each other measurement.

The third assumption is the structure of the sensor and motion models. The sensor is assumed to provide range and bearing, and that the uncertainty in range varies with range, while uncertainty in bearing is a constant angle. This matches well with a single camera, as a single pixel represents a larger distance at greater ranges, but subtends a constant angle. The robot is assumed to be commanded to move a differential amount in its local coordinate frame (x, y, and θ). The robot reports its measured differential motion, which is assumed to increase in uncertainty with increasing differential motion. Additionally, when position does not change but heading does, position is assumed to increase in uncertainty relative to the size of the turn; similarly, when heading does not change but position does, heading is assumed to increase in uncertainty relative to the distance traveled. These assumptions only affect the precise representation of the covariance matrices and Jacobians used; any other type of model might be implemented using this same formulation.

4.2 Combining Estimates

To combine multiple estimates on the state of a single object, Kalman-Bucy equations as in Eq. 13 and Eq. 14 are employed [121].

$$C = C_1 - C_1(C_1 + C_2)^{-1} C_1 \qquad \textbf{Eq. 13}$$

$$X = X_1 + C_1(C_1 + C_2)^{-1}(X_2 - X_1) \qquad \textbf{Eq. 14}$$

where X_1 and X_2 are position estimate means, C_1 and C_2 are the corresponding covariance matrices, and X and C are the mean and covariance matrix resulting from combining the estimates. By using the KB formulation shown, greater weight is automatically given to more certain estimates. These equations are symmetrical, and produce the same result regardless of the choice of order.

Transforming uncertainty as measured in one frame to uncertainty in another frame requires the local uncertainty in the frame in which the measurement is taken and the global uncertainty of the location of the measuring frame in the global frame. This converts uncertainty in sensor measurements to uncertainty in (x, y, θ) in the global frame. This uses the traditional transformation equation, Eq. 15:

$$C = JC_j J^T \quad \textbf{Eq. 15}$$

where Cj is the joint covariance matrix of all the components, J is the Jacobian of the transformation between frames, and C is the resulting covariance in the new coordinate frame.

4.3 Motion Model

The uncertainty in the robot pose is represented as a three-dimensional Gaussian distribution, which has a mean position (X) and covariance (C) as shown in Eq. 16 and Eq. 17.

$$X = \begin{bmatrix} X_R \\ Y_R \\ \theta_R \end{bmatrix} \quad \textbf{Eq. 16}$$

$$C = \begin{bmatrix} v_x & \rho_{xy}\sigma_x\sigma_y & \rho_{x\theta}\sigma_x\sigma_\theta \\ \rho_{xy}\sigma_x\sigma_y & v_y & \rho_{y\theta}\sigma_y\sigma_\theta \\ \rho_{x\theta}\sigma_x\sigma_\theta & \rho_{y\theta}\sigma_y\sigma_\theta & v_\theta \end{bmatrix} \quad \textbf{Eq. 17}$$

where v_x, v_y, and v_θ are the variances of x, y, and θ, respectively, where variance is the square of the standard deviation; σ_x, σ_y, and σ_θ are the standard deviations of x, y, and θ, respectively; ρ_{ij} values represent correlations between variables i and j.

Differential motion (ΔX) and covariance ($C_{\Delta X}$) are reported in the local frame of the robot as shown in Eq. 18-Eq. 19. The parameters of $C_{\Delta X}$ are experimentally determined.

$$\Delta X = \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta\theta \end{bmatrix} \quad \textbf{Eq. 18}$$

$$C_{\Delta X} = \begin{bmatrix} v_{\Delta x} & 0 & 0 \\ 0 & v_{\Delta y} & 0 \\ 0 & 0 & v_{\Delta\theta} \end{bmatrix} \quad \textbf{Eq. 19}$$

The geometry of the robot, differential motion, and global coordinate frames are shown in Figure 10.

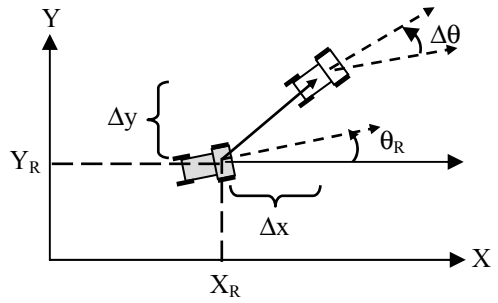

Figure 10. Robot Motion Coordinate Frames
The robot coordinate frame and differential motion are shown. The global frame is shown as (X, Y). The robot has a pose of (X_R, Y_R, θ_R) in the global frame (gray). It moves by (Δx, Δy, Δθ) to a new location (white).

Differential motion, defined in the local robot frame at the start of the motion, must be transformed into the global frame as X_R. In this case, the differential motion, measured in the robot's local frame, is a rotation by $\Delta\theta_T$ (to point toward the target point). The desired Δx_T and Δy_T are zero. This leads to the pose update in Eq. 20-Eq. 22, where $\theta = \theta_R$. J_M is the Jacobian of the rotation move.

$$X_R = X_R + \begin{bmatrix} \Delta x_M \cos(\theta) - \Delta y_M \sin(\theta) \\ \Delta x_M \sin(\theta) + \Delta y_M \cos(\theta) \\ \Delta\theta_M \end{bmatrix} \quad \textbf{Eq. 20}$$

$$J_M = \begin{bmatrix} 1 & 0 & -\Delta x_M \sin(\theta) - \Delta y_M \cos(\theta) & \cos(\theta) & -\sin(\theta) & -\Delta x_M \sin(\theta) + \Delta y_M \cos(\theta) \\ 0 & 1 & \Delta x_M \cos(\theta) - \Delta y_M \sin(\theta) & \sin(\theta) & \cos(\theta) & \Delta x_M \cos(\theta) - \Delta y_M \sin(\theta) \\ 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} \quad \textbf{Eq. 21}$$

$$C_R = J_M \begin{bmatrix} C_R & \{0\} \\ \{0\} & C_{\Delta X_M} \end{bmatrix} J_M{}^T \quad \textbf{Eq. 22}$$

The turn is followed by one forward step Δx_M, where commanded Δy_M and $\Delta\theta_M$ are zero. Eq. 23-Eq. 25 is the update, where $\theta = \theta_R + \Delta\theta_T$, and X_R and C_R are from Eq. 20 and Eq. 22. J_M is the move Jacobian.

$$X_R = X_R + \begin{bmatrix} \Delta x_M \cos(\theta) - \Delta y_M \sin(\theta) \\ \Delta x_M \sin(\theta) + \Delta y_M \cos(\theta) \\ \Delta\theta_M \end{bmatrix} \quad \textbf{Eq. 23}$$

$$J_M = \begin{bmatrix} 1 & 0 & -\Delta x_M \sin(\theta) - \Delta y_M \cos(\theta) & \cos(\theta) & -\sin(\theta) & -\Delta x_M \sin(\theta) + \Delta y_M \cos(\theta) \\ 0 & 1 & \Delta x_M \cos(\theta) - \Delta y_M \sin(\theta) & \sin(\theta) & \cos(\theta) & \Delta x_M \cos(\theta) - \Delta y_M \sin(\theta) \\ 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} \quad \textbf{Eq. 24}$$

$$C_R = J_M \begin{bmatrix} C_R & \{0\} \\ \{0\} & C_{\Delta X_M} \end{bmatrix} J_M{}^T \quad \textbf{Eq. 25}$$

4.4 Sensor Model

A measurement is represented by a range and bearing, each of which is represented by a one-dimensional Gaussian distribution. The coordinate frames are illustrated in Figure 11.

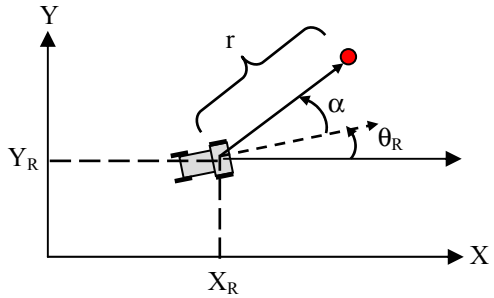

Figure 11. Robot Sensor Coordinate Frames

The sensor model parameters are shown. The global frame is shown as (X, Y). The robot has a pose of (X_R, Y_R, θ_R) in the global frame. It measures a range of r and a bearing of α to a target (red circle).

The parameters of this distribution are standard deviations (σ) and variance (v), computed as in Eq. 26 where a and b are constants that are experimentally determined. The measurement covariance is C_m.

$$C_m = \begin{bmatrix} v_r & 0 \\ 0 & v_\alpha \end{bmatrix} = \begin{bmatrix} (\sigma_r)^2 & 0 \\ 0 & (\sigma_\alpha)^2 \end{bmatrix} = \begin{bmatrix} (ar)^2 & 0 \\ 0 & b^2 \end{bmatrix}, \quad \textbf{Eq. 26}$$

To transform covariance into the global coordinate frame, the Jacobian of the transformation from sensor to global coordinates in Eq. 27 is used. Position and covariance are transformed using Eq. 28 and Eq. 29.

$$J = \begin{bmatrix} 1 & 0 & -r\sin(\theta_R + \alpha) & \cos(\theta_R + \alpha) & -r\sin(\theta_R + \alpha) \\ 0 & 1 & r\cos(\theta_R + \alpha) & \sin(\theta_R + \alpha) & r\cos(\theta_R + \alpha) \end{bmatrix} \quad \textbf{Eq. 27}$$

$$X = X_R + \begin{bmatrix} r\cos(\theta_R + \alpha) \\ r\sin(\theta_R + \alpha) \\ \alpha \end{bmatrix} \quad \textbf{Eq. 28}$$

$$C = J \begin{bmatrix} C_R & \{0\} \\ \{0\} & C_m \end{bmatrix} J^T \quad \textbf{Eq. 29}$$

4.5 Transforming Covariance Matrix Coordinate Frames

Two coordinate frames are used for covariance (Figure 12). First is the global frame, which defines covariance relative to the global axes and has been presented in 4.3 and 4.4. Second is the major-minor axis frame. This frame is aligned with the primary axes of the two-dimensional Gaussian distribution (elliptical in shape). In this frame, the standard deviation along the major axis (σ_{maj}) is the largest uncertainty and standard deviation along the minor axis (σ_{min}) is the smallest uncertainty.

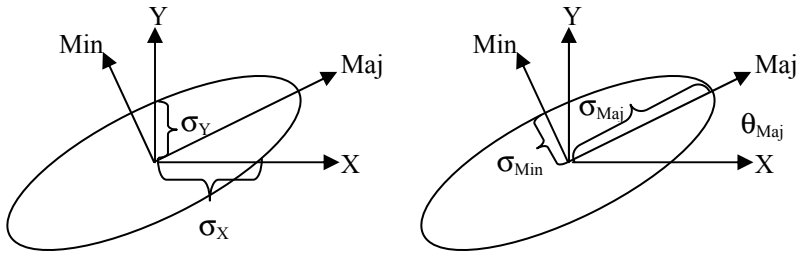

Figure 12. Covariance Matrix Coordinate Frames

Left: Covariance in the global frame, with standard deviations relative to X and Y axes. *Right*: Covariance in the major-minor frame, rotated by θ_{maj} relative to major and minor axes.

JCJ^T reflects rotation by $\varphi=+\theta_{Maj}$ from global to major-minor and $\varphi=-\theta_{Maj}$ from major-minor to global using J as defined in Eq. 30.

$$J = \begin{bmatrix} \cos(\varphi) & -\sin(\varphi) \\ \sin(\varphi) & \cos(\varphi) \end{bmatrix}$$

4.6 SLAM

In several experiments, the MVERT approach is integrated with a traditional Simultaneous Localization and Mapping (SLAM) approach so as to improve the resulting map SLAM produces. The formulation for SLAM used here is that used by Leonard and Durrant-Whyte, as presented in [74]. SLAM is integrated into the overall architecture shown in Figure 2 and Figure 3 (3.1) as part of the Update World Model step.

The process update for the SLAM update is identical to that described in detail for the Constraint-Based Localization in 6.3 (position mean) and 6.4 (covariance).

For the SLAM update, the typical transformation equations must be inverted, as in Eq. 31 and Eq. 32.

$$r_E = \sqrt{(\Delta x)^2 + (\Delta y)^2} \quad \textbf{Eq. 31}$$

$$\alpha_E = \tan^{-1}\left(\frac{\Delta y}{\Delta x}\right) \quad \textbf{Eq. 32}$$

where r is range, α is bearing, and subscript E indicates the expected values, and (Δx, Δy) are distances from the object at (X_L, Y_L) to the robot at (X_R, Y_R).

$$\Delta x = X_L - X_R \quad \textbf{Eq. 33}$$

$$\Delta y = Y_L - Y_R \quad \textbf{Eq. 34}$$

For each observed landmark or target, 1 to n, the Jacobian elements are determined, to create the joint Jacobian matrix, H, which varies from observation to observation (Eq. 35). Δx and Δy are distances in x and y, and is the range (two dimensional distance) to the specified object.

$$H = \begin{bmatrix} -\frac{\Delta x_1}{r_1} & \frac{\Delta y_1}{r_1} & 0 & \frac{\Delta x_1}{r_1} & \frac{\Delta y_1}{r_1} & 0 & \cdots & 0 & 0 & 0 \\ \frac{\Delta y_1}{r_1^2} & -\frac{\Delta x_1}{r_1^2} & -1 & -\frac{\Delta y_1}{r_1^2} & \frac{\Delta x_1}{r_1^2} & 0 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & 0 & \cdots & 0 & 0 & 0 \\ \frac{\Delta x_n}{r_n} & \frac{\Delta y_n}{r_n} & 0 & 0 & 0 & 0 & \cdots & 0 & \frac{\Delta x_n}{r_n} & \frac{\Delta y_n}{r_n} \\ \frac{\Delta y_n}{r_n^2} & \frac{\Delta x_n}{r_n^2} & 0 & 0 & 0 & 0 & \cdots & -1 & -\frac{\Delta y_n}{r_n^2} & \frac{\Delta x_n}{r_n^2} \end{bmatrix} \quad \textbf{Eq. 35}$$

Innovation is computed for each observed landmark to weight the measurements (subscript m) by how well they agree with expectations (better agreement leads to higher weight). The joint Innovation, I, is a column vector with all landmark innovations (Eq. 36).

$$I = \begin{bmatrix} r_{E1} \\ \alpha_{E1} \\ \vdots \\ r_{En} \\ \alpha_{En} \end{bmatrix} - \begin{bmatrix} r_{m1} \\ \alpha_{m1} \\ \vdots \\ r_{mn} \\ \alpha_{mn} \end{bmatrix} \quad \textbf{Eq. 36}$$

A joint covariance matrix, N, is computed to represent the covariance for each observed landmark as in Eq. 37. The range and bearing variance parameters v_{r1} and $v_{\alpha 1}$ are determined using the relationships in Eq. 26.

$$N = \begin{bmatrix} v_{r1} & 0 & 0 & \cdots & 0 \\ 0 & v_{\alpha 1} & 0 & \cdots & 0 \\ 0 & 0 & \ddots & \cdots & 0 \\ \vdots & \vdots & \vdots & v_{rn} & 0 \\ 0 & 0 & 0 & 0 & v_{\alpha n} \end{bmatrix} \quad \textbf{Eq. 37}$$

The Kalman-Bucy filter update uses H, I, and N to compute the final resulting position (X) and covariance (C) as shown in Eq. 38-Eq. 41.

$$S = HCH^T + N \quad \textbf{Eq. 38}$$

$$K = CH^T S^{-1} \quad \textbf{Eq. 39}$$

$$C = C - KSK^T \quad \textbf{Eq. 40}$$

$$X = X + KI \quad \textbf{Eq. 41}$$

X is the joint state matrix, as in Eq. 42,

$$X = \begin{bmatrix} X_R \\ Y_R \\ \theta_R \\ X_{L1} \\ Y_{L1} \\ \vdots \\ X_{Ln} \\ Y_{Ln} \end{bmatrix} \quad \textbf{Eq. 42}$$

and C is the joint covariance matrix, as in Eq. 43,

$$C = \begin{bmatrix} C_R & \{0\} & \{0\} & \cdots & \{0\} \\ \{0\} & C_{L1} & \{0\} & \cdots & \{0\} \\ \{0\} & \{0\} & \ddots & \cdots & \{0\} \\ \vdots & \vdots & \vdots & \ddots & \{0\} \\ \{0\} & \{0\} & \{0\} & \{0\} & C_{Ln} \end{bmatrix} \quad \textbf{Eq. 43}$$

with

$$C_{Li} = \begin{bmatrix} (\sigma_{XLi})^2 & \rho_{XLiYLi}\sigma_{XLi}\sigma_{YLi} \\ \rho_{XLiYLi}\sigma_{XLi}\sigma_{YLi} & (\sigma_{YLi})^2 \end{bmatrix} \quad \textbf{Eq. 44}$$

where σ is a standard deviation and ρ is a correlation factor.

For multiple robot updates, this process is repeated, replacing the robot entries with the current observing robot. If collaborative localization is desired, a full joint X and C would be required, with the appropriate place holders in the H, I, and N matrices.

4.7 Additional Computations

Two additional computations are used in this work. The first is the standard deviation (σ) or variance (σ^2), which is computed traditionally as shown:

$$\sigma = \frac{\sum_{i=1:N}(x_i - \mu)}{N - 1} \quad \textbf{Eq. 45}$$

where x_i are individual measurements, μ is the mean of the x_i measurements, and N is the total number of x_i measurements.

The second is the 95% confidence interval on a mean, CI, which indicates the range within which it is 95% confident that the true mean lies. This is also computed traditionally as shown:

$$CI = 1.96\frac{\sigma}{N} \quad \textbf{Eq. 46}$$

4.8 Summary and Discussion

In the treatment of uncertainty in this work, Gaussian distributions are used to represent the uncertainty locations of robots, targets, and obstacles within the environment. Updates to position estimates either use Gaussian multiplication of (independent) beliefs or a traditional SLAM approach to track correlations and uncertainties.

Chapter 5 Experimental Methodology

The MVERT approach is intended to improve the performance of a distributed multi-robot team at the execution level without requiring complex planning. The particular applications to which MVERT is applied in this thesis are mapping and exploration. Several types of experimental domains are desirable for testing the hypothesis that team performance may be improved, and are described in 5.6. The tasks executed in these domains, including mapping and exploration, are detailed in 5.3. MVERT can be compared to current traditional approaches within these domains; these comparative approaches are described in Section 5.2. Additionally several performance metrics must be applied in order to quantitatively compare and evaluate the MVERT performance, which are described in 5.1. The specifics of the platform used in these experiments are described in 5.4, with details of the objects in the environment described in Section 5.5. Performance and evaluation in the various domains, using the specific implementation parameters listed in Section 5.7, are presented in Chapter 7 through Chapter 10.

5.1 Performance Evaluation Metrics

One of the primary goals of MVERT in the applications presented here is to obtain a map with high accuracy and high certainty. This map should be generated as quickly as possible. Additionally, the generation of this map may be required while simultaneously attempting to fulfill other mission tasks as quickly as possible. Thus, the metrics for comparison focus on map quality (both accuracy and certainty) and on mission completion time. In some cases, the relative metric (ratio of the value of the metric in the compared experiment to the value of the metric in MVERT) is used to show comparative performance.

5.1.1 Map Accuracy

Map accuracy is defined as how well the position estimates of the targets in the map agree with the ground truth. Distance between the position estimate and the true position is one metric for evaluating accuracy. The maximum distance is used here to quantify the map accuracy, demonstrating the worst-case performance. If a target is not known, the uncertainty in its position is infinite. This can be determined by evaluating the Gaussian distribution at the ground truth location value. In Eq. 47, (X_E, Y_E) is the position estimate and (X_A, Y_A) is the actual position. Mean E_1 is computed replacing *max* with *mean*.

$$E_1 = \max_{i=1:T} \sqrt{(X_{Ei} - X_{Ai})^2 + (Y_{Ei} - Y_{Ai})^2} \qquad \textbf{Eq. 47}$$

5.1.2 Map Certainty

Map certainty is defined as the confidence of the map that is produced. Certainty is inverse uncertainty, and can therefore be directly related to the target location value function (Eq. 2), which is one method of quantifying the level of certainty in an estimate (Eq. 48).

$$E_2 = V_{TL} \qquad \textbf{Eq. 48}$$

An additional measure of map certainty is the maximum error in any target (Eq. 49). If a target is unknown, this uncertainty is infinite. By definition, σ_{maj} is the largest uncertainty in a two-dimensional Gaussian distribution. If a target is now known, its σ_{maj} is considered to be infinite.

$$E_3 = \max_{i=1:T} \sigma_{maj} \qquad \textbf{Eq. 49}$$

5.1.3 Time to Mission Completion

To measure the efficiency of mission completion, the metric applied is the time to a map quality of a pre-defined certainty. This metric measures the number of steps required until all targets are known to a desired certainty, where certainty is defined by the largest target uncertainty along any dimension and as computed by the target location value function. In Eq. 50, U is the specified measure of uncertainty (metrics E_2 or E_3) and G is the goal uncertainty level. If each step is a fixed size, this metric can be used to evaluate both time and path length, where path length is the time (in steps) times the step size.

$$E_4 = \text{Time}(U <= G) \qquad \textbf{Eq. 50}$$

5.1.4 Robot Localization Accuracy

A fifth metric is the quality of localization for the robot team members. This is computed as the sum of distance between robots' actual positions (X_A, Y_A) and robots' estimated positions (X_E, Y_E), as in Eq. 51.

$$E_5 = \max_{i=1:R} \sqrt{(X_{Ei} - X_{Ai})^2 + (Y_{Ei} - Y_{Ai})^2} \quad \textbf{Eq. 51}$$

5.1.5 Mission Completeness

The last metric is mission completeness. In mapping tasks, this is the percentage of the total number of targets located by the end of the mission.

$$E_6 = 100 * \frac{T_{foud}}{T_{true}} \quad \textbf{Eq. 52}$$

5.2 Compared Approaches

5.2.1 SLAM: Divide and Conquer versus MVERT

The most commonly used method for mapping an unknown area is to use SLAM. This comparison will demonstrate the contribution MVERT can make to the overall mapping task. Typically, regions are divided in order to break the problem into multiple single-robot tasks. Where measurements overlap, they are used to improve estimates and to combine maps together. Here, SLAM is used with and without MVERT.

The performance will be compared by examining the resulting quality of the map and the time required to generate a map of the specified quality. The specified quality for these experiments is that the maximum uncertainty on any target must be less than 0.01 meter. The baseline SLAM approach assigns a sub-region for each robot to map, and within that region the robot follows a rectangular coverage pattern (defined by waypoints). Measurements taken in other sub-regions while traveling to the designated sub-region are also added to the common global map. Robots are given the same starting locations.

5.2.2 One-Step Optimal

The one-step optimal computes all *sets* of moves for a team. The resulting value of each move set is compared to that for all other move sets, and the set with the highest value is chosen and executed. The number of computations required scales exponentially with number of robots. One-step optimal is applied to several target location missions and the resulting trajectories and map qualities are observed. The one-step optimal is implemented using the same value function and candidate moves as the MVERT approach, with evaluation of candidate move *sets* rather than of individual candidate moves for each robot.

Optimal performance with the same parameters is compared to MVERT. Qualitatively, the similarity or dissimilarity of the trajectories is discussed in order to illustrate the difference between MVERT and one-step optimal. Additionally, the final map quality after a fixed number of steps is compared. Lastly, the time required to produce a map of the desired quality is investigated. The specified map quality is such that the maximum uncertainty on any target must be less than 0.01 meter.

5.2.3 Individual Action Selection in MVERT

MVERT attempts to improve the team performance by including expected teammate contributions in action selection. The effects of including expected teammate contribution are investigated by comparing the performance of MVERT with and without teammates. In this comparison, robots are always trying to maximize information gain in terms of value, but without teammate consideration robots only maximize their own contributions. This is implemented by applying MVERT to robot one at a time, each with the same starting pose as a team member in the multi-robot experiment. Individual maps are later recombined using the raw data as if they had been shared, enabling direct comparison of action selection methods.

Performance is compared by investigating the final map quality and time to desired map quality in each case. Desired map quality is that no maximum target uncertainty should exceed 0.01 meter. Additionally, a qualitative comparison is made of the robot trajectories to those produced using teammate contributions in order to express the effects of considering teammates.

5.3 Experimental Mission Definitions

5.3.1 Target Location Mission

Target location is a simplified version of a mapping mission. The robots are assumed to be operating in an environment with landmarks whose locations are known exactly. The robot team is to locate targets, whose locations are not known a priori, while avoiding collisions. This task is primarily used to evaluate performance as different parameters are varied, such as sensing footprint, robot motion, team size, and number of candidate actions considered. This task is also used to compare performance to that of one-step optimal action selection. Value for this task is the target location value, defined in 3.3.1. As no noise is present, only metrics for certainty (E_2, E_3) and mission time (E_4) are used.

5.3.2 Mapping Mission

The mapping mission is to map an unknown environment. The map is the locations of all teammates and all targets in the environment while avoiding collisions. Localization and mapping are performed using the targets with traditional Simultaneous Localization and Mapping (SLAM) as described in 4.6. Initially, robots know only the poses of all teammates in the arbitrary global frame, not the locations targets. For the robot implementation of the mapping task, robots pause and scan their heads to collect measurements. Once all robots have communicated observation and localization results, robots, synchronously and distributed, compute their next move and, then, synchronously execute the moves. Synchronization is handled automatically by the code. The value for this task is the target location value, defined in Eq. 2, Section 3.3.2 . Comparisons are made to SLAM with and without MVERT for action selection. Coverage patterns are used to direct motion to contrast with MVERT. Comparisons are also made within MVERT, with and without consideration of teammate contributions. All metrics are used to evaluate this task.

5.3.3 Dynamic Target Tracking Mission

The dynamic target tracking mission is to observe multiple moving targets within a known environment and avoid collisions. The environment is known in the sense that landmarks are known exactly. The characteristics of the moving targets are not known a priori. In simulation, the targets are moved according to pre-defined paths. These paths are generated to be random motion or to follow smooth trajectories. Targets and robots are moved at the same time; since robots are assumed to be synchronized, measurements occur simultaneously. This allows the simplification of performing only one target update per cycle. In physical experiments, the targets (remote controlled vehicles) are moved by hand to approximately follow pre-determined paths. For simplicity in handling the dynamic targets, the targets are individually moved while the robots are still. This motion occurs after the robots collect data and before they move to their new locations. As in the static mapping case, the momentary stationary nature of the robots allows simulation of an omnidirectional camera through rotating in place. Synchronization among robots is handled within the code. Synchronization with the moving targets is enforced by forcing robots to wait for human input before executing the next step. Probabilistic Constraint-Based Localization (6.4) is used to localize the robots without collaboration, as robots cannot accurately observe the positions of teammates with their sensors. Target location value, defined in Eq. 2, is used for action selection in the dynamic target tracking mission. The metric applied to this mission is target certainty (E_2).

5.3.4 Planetary Exploration Mission

The planetary exploration task simulates the types of goals that may be required of a planetary robot team. These goals include exploring and mapping an area, examining specified targets of varying priority in an area with known or unknown landmarks while maintaining communication and avoiding collisions. Obstacles in the environment interfere with motion, observation, and communication. The robots are heterogeneous in sensing and/or motion capabilities. Target mapping/localization is modeled by the target location value function, described in Eq. 2, Section 3.3.1, and target examination is modeled by the target sampling value function, described in Eq. 3, Section 3.3.2. Similarly the values of communication and exploration are described in Eq. 5, Section 3.3.2, and Eq. 6, Section 3.3.4, respectively. For the planetary exploration task, the allocation of target sampling tasks is made based only on proximity (the closest robot to the target when it is first seen), as this is not a research focus. Target priority is assigned a priori at random or evenly. Due to the complexity of this task, it is only implemented in simulation. All metrics are applied to this task.

5.4 Robot Platform

The bulk of this work has been implemented using the Sony Quadruped robot platform (Figure 13). This is a four-legged platform that contains a single color camera. Additional sensors on the robot include sensors to report joint angles, used in determining differential motion, and an IR proximity sensor, unused in these experiments. The command code for interfacing with the robot (motions and sensing) is based on the code from the CMU 2002 RoboCup team [138].

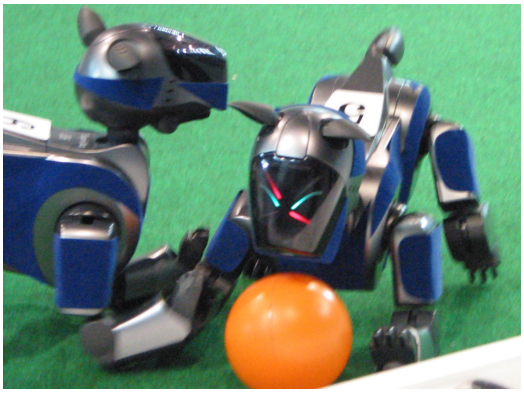

Figure 13. Sony Quadruped Robots

5.4.1 Motion Model

The Quadruped motions are specified as differential motions as described in Figure 10, 4.3: distance in x, distance in y, and change in heading. The robot has positional and heading uncertainty when walking and when turning in place. The basic motions use the walking gait developed by Carnegie Mellon for CMPack in 2002. The parameters of the motion model for a step of 0.254 meters (as in Eq. 18, 4.3) are:

$$\sigma_x = 0.034 \quad \text{m}$$
$$\sigma_y = 0.015 \quad \text{m}$$
$$\sigma_\theta = 0.056 \text{ rad}$$

For turns, the motion parameters differ for single turns (5º, 10º, 45º) and compound turns (other turns):

$$\sigma_x = 0.0040 \text{ m (single)}; 0.0075 \text{ m (compound)}$$
$$\sigma_y = 0.0027 \text{ m (single)}; 0.0045 \text{ m (compound)}$$
$$\sigma_\theta = 0.0235 \text{ rad (single)}; 0.0563 \text{ rad (compound)}$$

σ_x, σ_y, and σ_θ are (respectively) the standard deviations in x, y and θ in the robot's local coordinate frame.

5.4.2 Sensor Model

The sensor is a single camera reporting target range and bearing as shown in Figure 11, Section 4.4. The single camera can provide range to targets is available through calibration on well-modeled targets of known characteristics. These experimentally determined model parameters for Eq. 26 (Section 4.4) are used for both simulation experiments and physical experiments, except as noted. The camera has a range of 2.75-3.0 meters for the pre-defined targets and a field of view of 360°. While the field of view of the camera is not a full 360 degrees (single frame of 58° horizontal, 48° vertical), the effective field of view of the robot is 360 degrees by allowing the robot to scan its head (increasing the field of view to 220°) and by turning the body 180° once to bring the full circle into view for measurements. Camera noise model parameters are shown below. Robot cameras have a bias on high measured range (R_m), corrected in Eq. 53.

$$\sigma_r = 0.102 \text{ r}$$
$$\sigma_\alpha = 0.029$$
$$R = (1 - 0.1195) R_m, \; R_m > 2.3\text{m} \qquad \textbf{Eq. 53}$$

5.5 Targets and Landmarks

In physical experiments, static targets for mapping and landmarks for dynamic target tracking are the bi-colored, 0.1 meter cylindrical RoboCup legged-league landmarks (Figure 14).

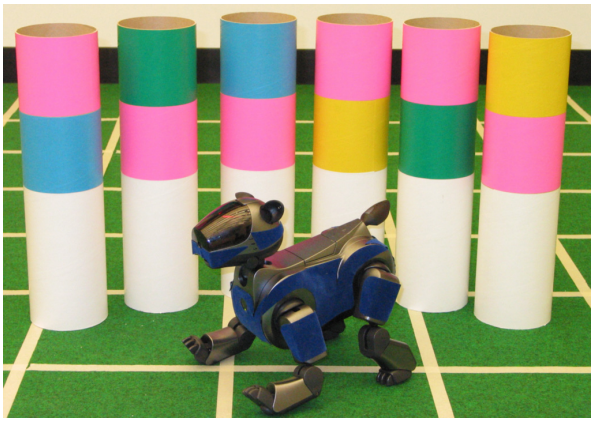

Figure 14. Stationary Targets
Six bi-colored RoboCup landmarks are used as stationary targets.

Remote-controlled tanks with similar bi-colored, cylindrical markers are used as dynamic targets (Figure 15). Dynamic targets are 0.08 meters in diameter. Landmarks and targets are treated as points. In simulation, the targets are points, but a buffer is defined around each target into which the robot may not pass. In physical experiments, sensing is calibrated to measure range and bearing to the center of the landmark, and a buffer around the landmark is defined to ensure the robot can see the entire target.

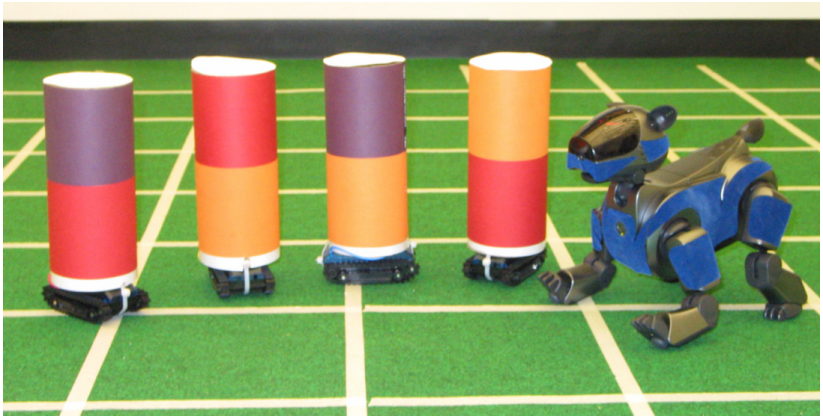

Figure 15. Dynamic Targets
The dynamic targets are remote-controlled tanks with bi-colored, cylindrical markers.

5.6 Experimental Environments

5.6.1 Code Environment

The code for both simulation and physical experiments is written in *Matlab*. The MVERT implementation is identical for both simulation and physical experiments, though physical experiments require an additional interface. This interface consists of additional Matlab functions and basic behaviors written in C++ using the Carnegie Mellon CMPack02 architecture [138].

5.6.2 Simulation Environment

The simulation environment approximates the robots, targets, and landmarks as points for computational purposes. The size of the environment and the step size used by the robots vary by task. The vision and motion models of the simulated robots is identical to that used for the robots, as described in 5.4. Up to 25 robots, 50 static targets, and 10 dynamic targets are deployed in the simulation environment. An example environment is shown in Figure 16, with four targets (two known and two unknown), two robots (schematic Sony Quadrupeds), two known landmarks, and two sample targets.

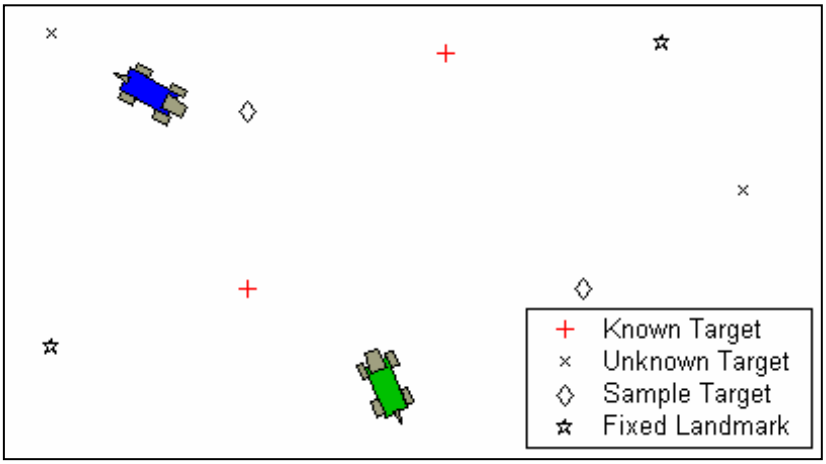

Figure 16. Simulation Environment
Simulation environment with two robots, two fixed landmarks (☆), two known targets (+), two unknown targets (x), and two sampling targets (◊).

5.6.3 Robot Environment

The Robot Environment is a carpeted arena 3.0 meters by 4.6 meters and is shown in Figure 17. This arena is based on that used for the RoboCup soccer four-legged league, with walls and goals removed from the environment. Up to four robots, six static targets, and four dynamic targets are deployed in the environment. Targets and landmarks are treated as points for computation, and sensing is calibrated to treat targets and landmarks as points. A pair of overhead cameras observes experiments for determining ground truth for comparison. A white grid (0.254 meter resolution) is used for camera calibration and tracking.

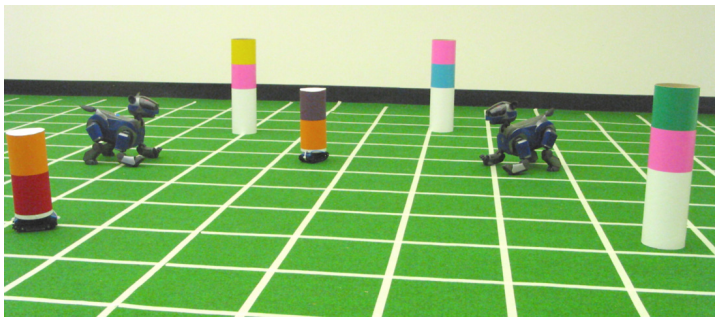

Figure 17. Robot Environment
The robot environment with two robots, three static targets/landmarks (the larger, taller cylinders) and two dynamic targets (the smaller, shorter cylinders mounted on small robots).

To obtain ground truth in the physical robot experiments, two overhead cameras (one overlooking each half of the field) were used. An example of the overhead images is shown in Figure 18.

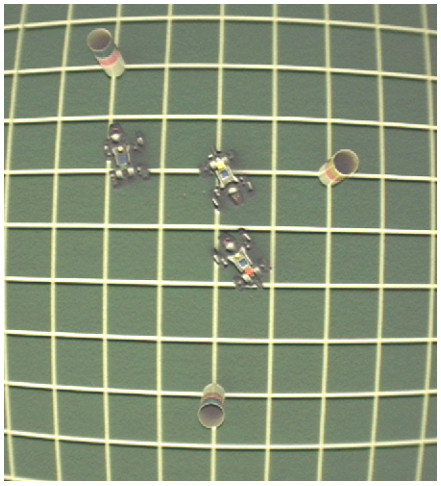

Figure 18. Robot Environment Overhead View
Overhead view (used for ground truth) of one half of the physical environment. Three robots and three landmark targets are shown.

5.7 MVERT Experimental Parameters

Several parameters in MVERT must be defined for these experiments, including parameters to select candidate moves and parameters for value functions and combining value functions. Except where noted, the default values described here are used.

The baseline parameters for this implementation of MVERT, used throughout with the exception of Chapter 7, are based on the Aibo Robot (5.4). These parameters are detailed below and are summarized here:

- Robot step size: 0.25 meters,
- Robot candidate move angular resolution: 5 degrees,
- Obstacle avoidance buffer around teammates: 0.5 meters,
- Obstacle avoidance buffer around targets and landmarks: 0.5 meters

Candidate moves are selected to be on a circle at one fixed step size away from the robot's current location. Robots are assumed to move along the line from the initial position to the desired position, making the resulting heading align with this direction. The step size (radius of the circle for candidate moves) is 0.25 meters. This value is chosen to be approximately the length of the robot. Candidate moves on this circle are spaced at an angular resolution of 5 degrees and can be at any relative heading (except as noted). This value is chosen so as to be small enough to provide high resolution of potential moves (distance between candidate points is approximately 0.02 meters) yet large enough to make different candidate moves differentiable given the noise of robot motion. Motions are executed by first turning the robot to face the selected point, and then moving the robot forward one step.

For collision avoidance, robot size is considered to be 0.25 meters. In the physical system, target/landmark size 0.1 meters, with an additional buffer of 0.15 meters, preventing robots from moving closer than 0.5 meters to a landmark, to ensure entire targets/landmarks are visible.

In Chapter 7, the parameters are determined to have high resolution in moves and low noise, and are somewhat varied to explore their effects. These are detailed in that chapter.

In addition to the parameters of MVERT, parameters of the value functions must also be determined. The relative weights of values must be selected. For target location, mapping, and dynamic target tracking missions, only one value is applicable (V_{TL}, Eq. 2, Section 3.3.2) and no weighting is required. For the

planetary exploration mission, two or more of the different tasks are combined. The total value is determined using the weighted average with experimentally chosen weights with Eq. 1, Section 3.3, which becomes:

$$V = \kappa_{TL} V_{TL} + \kappa_E V_E + \kappa_N V_N + \kappa_S V_S \quad \textbf{Eq. 54}$$

For the target sampling value function, the constant $K_S = 100000$ in Eq. 6, Section 3.3.2. This value is chosen to ensure that it overrides all other values applied in the planetary exploration mission simulations. The closest sampling target is selected current sampling target. If a sampling target is closest to two or more robots, the closest robot gets the target, and the other robots select the next-closest, iteratively.

5.8 Summary and Discussion

To evaluate performance, metrics reflecting map accuracy and uncertainty, time to mission completion, and robot localization accuracy are proposed and defined. MVERT will be compared to several approaches, including individual action selection, one-step optimal team action selection, and mapping with coverage patterns. Several experimental mission types are defined including mapping, target tracking, and planetary exploration. The implementation of the MVERT architecture is in *Matlab*, both for simulation and for control of the physical robot team of Sony Aibo robots.

Chapter 6 Collaborative Localization

If robots are operating within an environment that has known landmarks, robots may perform localization rather than simultaneous localization and mapping. An approach to landmark localization is used to localize the robot within the known environment for the Dynamic Target Tracking Mission experiments (Chapter 9); the approach used here is Constraint-Based Localization, an approach designed to be fast, simple, functional with few landmarks, and approximate pose with Gaussian distributions.

6.1 Introduction to Constraint-Based Localization (CBL)

Several commonly used approaches to landmark-based localization for robots have been proven to perform well: Kalman-Bucy filters and Monte Carlo Localization are the most common. Kalman-Bucy filter localization has been noted to be sensitive to noise, though it is relatively low in computational complexity. Monte Carlo Localization can be highly robust to noise at the cost of increasing computational complexity by increasing the number of particles used to track robot pose. For some platforms and for some highly dynamic environments, the need for very low computational overhead and robustness to noise and dynamics may not make these approaches practical for implementation.

Constraint-Based Localization (CBL) is designed to address these specific applications, and was developed in the domain of Robot Soccer for the Minnow middle-sized league platform (Figure 19). This platform has very little processing time available due to internal overhead, requiring the control system to be as simple as possible. Additionally, for the middle-sized league in 2000, very few landmarks were available (only the goals), and these are difficult to measure distances and bearings with accuracy, due to the complex geometry. CBL is inspired by Kalman-Bucy filter localization, though the computational requirements for updates are reduced through several methods. There is also a sliding level of complexity to allow CBL to perform as the platform and environment demands.

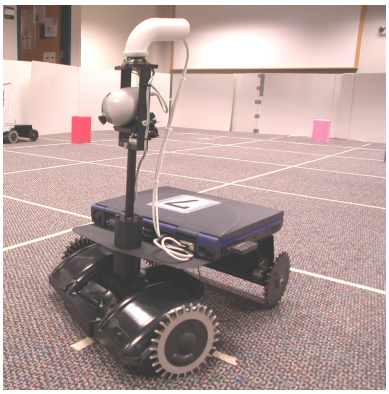

Figure 19. The Minnow Robot

CBL separates contributions of range and bearing to landmarks, each update being far simpler than a full KBF update. The estimate resulting from a measurement is determined using simple geometric constraints. Sliding complexity use of simple geometric representation and combination of estimates or, like the KBF, Gaussian distributions may used to model both sensor measurements and robot pose when processing is available. Lastly, only single updates are made per cycle: heading or position; this eliminates some computation at each step, and eliminates the need to simultaneously optimize both heading and position.

In addition to allowing for simpler computation, CBL introduces some natural flexibility. Different types of landmarks may provide different types of information (or different qualities of information). For example, landmarks may be too far to measure range but can still provide bearing, or some sensors may provide only range or bearing. The separation of different types of measurements easily allows for missing measurements or data quality inequity without special treatment.

CBL allows for several different types of landmarks, also motivated in part by the robot soccer domain. These are range-to-point landmarks, bearing-to-point landmarks, and range-to-surface landmarks. These landmarks may be markers, walls, or teammates (for example) and landmarks may be of more than one

type. CBL does, however, assume that the locations of the landmarks are known exactly or with little uncertainty. CBL cannot accommodate simultaneous localization and mapping.

6.2 CBL Approach

CBL relies on using simple geometric constraints to compute estimates from sensor measurements. Different types of measurements lead to different types of constraint. The robot's current pose estimate is represented as (X_O, Y_O, θ_O). Landmarks are located at or pass through (X_L, Y_L). Range measurements are labeled r, and bearing measurements are labeled α. Sensor-based estimates are labeled (x', y', θ').

A range-to-point landmark places the robot on a circle around the landmark with a radius equivalent to the range measurement (Figure 20). Range-to-point provides no information on heading. The point on this constraint that most closely agrees with the robot's prior pose estimate is computed in Eq. 55 and Eq. 56. This is on the line from the robot's pose estimate to the landmark, at a distance r from the landmark.

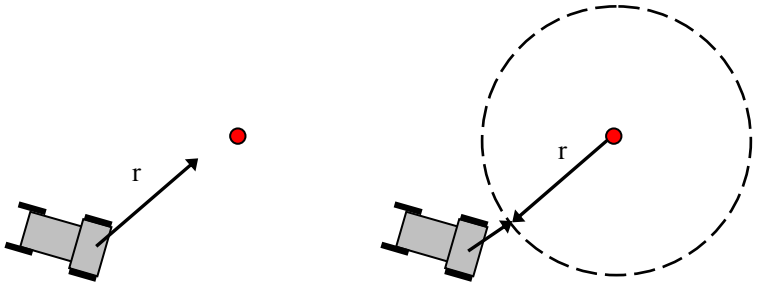

Figure 20. CBL Position Estimate from Range to a Point
Left: A noisy range is measured to a point. *Right*: The range places the robot on a circle. The closest point on the circle to the robot's pose estimate is the sensor-based estimate.

$$x' = X_O + (X_L - X_O)\frac{r - \sqrt{(X_L - X_O)^2 + (Y_L - Y_O)^2}}{r}$$

$$y' = Y_O + (Y_L - Y_O)\frac{r - \sqrt{(X_L - X_O)^2 + (Y_L - Y_O)^2}}{r}$$

A bearing-to-point landmark (assuming heading is correct) places the robot on a line, the slope of which is defined by the bearing α and the robot's heading θ_O (Figure 21). The point on this constraint that most closely agrees with the robot's prior heading estimate is computed in Eq. 57and Eq. 58.

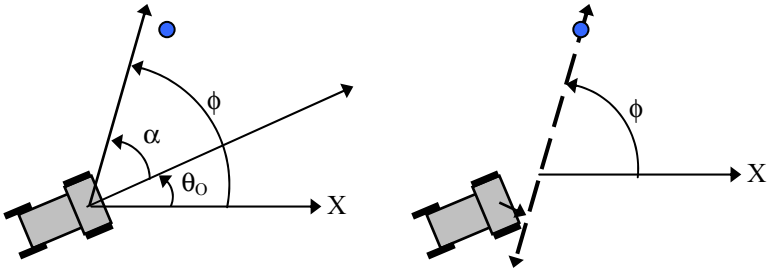

Figure 21. CBL Position Estimate from Bearing to a Point
Left: A noisy bearing is measured to a point. *Right*: The bearing and heading place the robot on a line. The closest point on the line to the robot's pose estimate is the sensor-based estimate.

$$x' = \frac{X_O + mY_O + m^2X_L + \frac{m^2 r}{\sin(\phi)}}{1 + m^2} \qquad \textbf{Eq. 57}$$

$$y' = mx' + mX_L + Y_{L-}\frac{mr}{\sin(\phi)} \qquad \textbf{Eq. 58}$$

where $m = \tan^{-1}(\phi)$. In the case where $\phi = \pm\frac{\pi}{2}$, these equations simplify to:

$$x' = X_L \quad \textbf{Eq. 59}$$

$$y' = Y_O \quad \textbf{Eq. 60}$$

A bearing-to-point landmark also defines a single heading, assuming position is correct (Figure 22). This value is computed in Eq. 61 using the pose (X_O, Y_O, θ_O) and the bearing α.

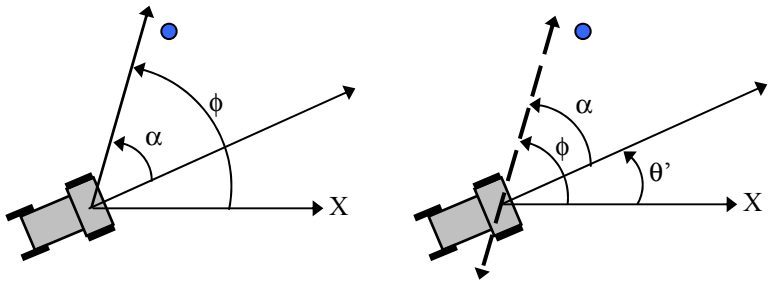

Figure 22. CBL Heading Estimate from Bearing to a Point
Left: A noisy bearing is measured to a point. *Right*: Relative position to the landmark provides ϕ. The bearing α and ϕ place the robot's heading at a single value.

$$\theta' = \tan^{-1}\left(\frac{Y_L - Y_O}{X_L - X_O}\right) - \alpha \quad \textbf{Eq. 61}$$

A range-to-surface landmark, given the range r is perpendicular to the surface at the detection point, defines a curve parallel to the surface at the measured distance (such as a wall, Figure 23). For a linear surface (a flat wall), the point on this constraint that most closely agrees with the robot's current position estimate is computed using Eq. 57-Eq. 58, where (X_L, Y_L) is a point on the wall. Range-to-surface could also supply a heading, but the complexity of this limits its usefulness and it is ignored here.

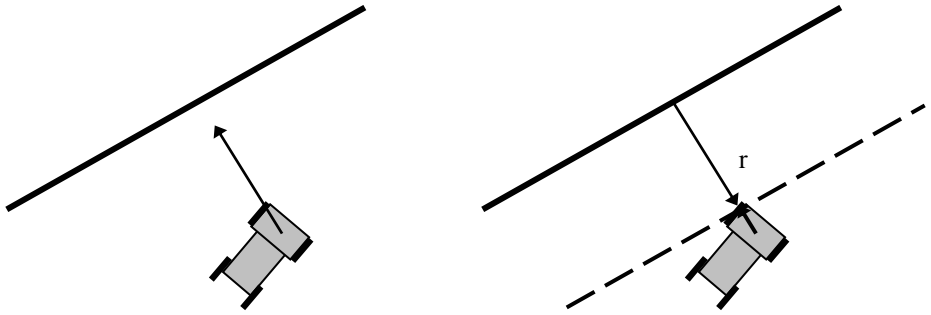

Figure 23. CBL Position Estimate from Range to a Wall
Left: A noisy range is measured perpendicular to a wall. *Right*: Range provides a line parallel to the wall, and the sensor-based estimate is the closest point on it.

The basic implementation of CBL uses two stages in the update: process update (based on odometry) and sensor update (based on observed landmarks). The basic process is illustrated in Figure 24; pose estimates are shown as light gray (subscript 0) and actual poses are shown as dark gray. An update using one bearing landmark and one range landmark is shown.

The algorithm for the updates can be summarized as follows:

1. Use differential motion and motion model to update pose.
2. Take measurements on visible landmarks (range and bearing).
3. Use geometry to generate constraints on pose.
4. Compute sensor-based estimates as closest points on constraints (with or without uncertainty).
5. Combine estimates with prior belief using appropriate probabilistic or non-probabilistic method.

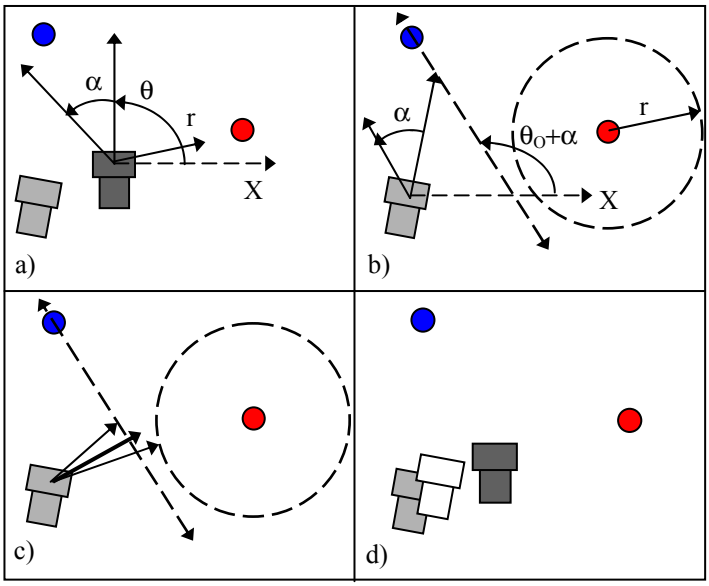

Figure 24. Constraint-Based Localization Sensor Update
a) Noisy range is measured to one landmark, and noisy bearing is measured to another (Step 2).
b) The constraints (circle and line, respectively) are determined for each measurement (Step 3).
c) The closest point on each constraint to the pose estimate (in light gray) is determined (Step 4).
d) The combined sensor estimate is combined with the pose estimate to generate the new pose estimate (in white) using weighted average or probabilistic techniques (Step 5).

6.3 Non-Probabilistic CBL (NPCBL)

NPCBL is the simplest implementation. In this non-probabilistic implementation, position is represented as a point. The process update moves the robot to (X, Y, θ) according to the motion model (Eq. 62-Eq. 66, Figure 10) given previous pose (X_O, Y_O, θ_O) and differential motion (Δp, Δθ); Δp is differential position and Δθ is differential heading.

$$\Delta x = \Delta p \cos(\Delta\theta + \theta_O) \quad \textbf{Eq. 62}$$

$$\Delta y = \Delta p \sin(\Delta\theta + \theta_O) \quad \textbf{Eq. 63}$$

$$X = X_O + \Delta x \quad \textbf{Eq. 64}$$

$$Y = Y_O + \Delta y \quad \textbf{Eq. 65}$$

$$\theta = \theta_O + \Delta\theta \quad \textbf{Eq. 66}$$

The sensor model is that shown in Figure 11 in 4.4. For the sensor update, individual sensor measurement estimates are determined using the equations in 6.2. Multiple estimates from individual sensor measurements are combined into a single sensor-based estimate, Xs, by a simple weighted average. Uncertainty in measurements (σ) is applied only through the weight of the measurements' corresponding estimates. This weighted average of all sensor-based estimates is computed in Eq. 67. The final sensor-based estimate is combined with the prior estimate using the same type of weighted average in Eq. 68.

$$X_S = \sum_i \frac{1}{\sigma_i^2} X_{S_i} \quad \textbf{Eq. 67}$$

$$X = X_O + wX_S \quad \textbf{Eq. 68}$$

NPCBL was implemented on the Minnow robots and in simulation. The choice of weight values and thresholds are based on experimental performance to provide a high weight to good quality sensor readings, and low weight to low quality sensor readings. From experimentation, it was determined that the quality of range information dropped off quickly after 3.75 meters and that heading results were of much better quality when provided by averaging across more than one landmark. For position, w = 0.5 if the closest landmark is less than 3.75 meters away and 0.2 if the closest landmark is more than 3.75 meters away. For heading, w = 0.5 if two or more landmarks are seen, and 0.2 if only one landmark is seen. The standard deviation of uncertainty on sensor measurements, used for weighting in the overall sensor-based estimate, was experimentally determined: $\sigma_r = 0.1r$. As the uncertainty in bearing scaled with range, the same weight is used for bearing and for range estimates when determining the final sensor-based estimate.

6.4 Probabilistic CBL (PCBL)

In many cases, an estimate of pose uncertainty is desirable and processing is available for this purpose. PCBL adapts CBL to handle probabilistic representation by using two-dimensional Gaussian distributions to represent position, and single-dimensional Gaussian distributions to represent heading, range, and bearing. Two-dimensional Gaussian position allows for arbitrary orientation and proportion to indicate correlation, and differing certainty along some dimensions. To accommodate the addition of uncertainty, odometry and sensor uncertainty estimates must be known. Additionally, the point estimates produced by CBL must have error introduced by mapping sensor uncertainty to positional uncertainty.

Motion updates the pose based on the reported odometry, as in Eq. 62-Eq. 66. For uncertainty, the transformation from distance and direction to Cartesian space is computed using $J^T C J$ with C shown in Eq. 69. Here, c is a constant, Δr is the change in distance, and $\Delta\theta$ is the change in heading.

$$C_L = \begin{bmatrix} (c\Delta r)^2 & 0 \\ 0 & (\Delta\theta)^2 \end{bmatrix} \quad \textbf{Eq. 69}$$

First, the uncertainties are rotated to align with the global frame using Eq. 70-Eq. 74. (X, Y, θ) is the robot's updated position, (X_O, Y_O, θ_O) is the robot's previous position, J is the Jacobian of the transformation, and Cm is the pose covariance resulting after the move.

$$X = X_O + \Delta r \cos(\Delta\theta + \theta_O) \quad \textbf{Eq. 70}$$

$$Y = Y_O + \Delta r \sin(\Delta\theta + \theta_O) \quad \textbf{Eq. 71}$$

$$\theta = \theta_O + \Delta\theta \quad \textbf{Eq. 72}$$

$$J = \begin{bmatrix} \cos(\Delta\theta + \theta_O) & \sin(\Delta\theta + \theta_O) \\ \Delta r \sin(\Delta\theta + \theta_O) & \Delta r \cos(\Delta\theta + \theta_O) \end{bmatrix} \quad \textbf{Eq. 73}$$

$$C_m = J C_L J^T \quad \textbf{Eq. 74}$$

Once aligned, compounding uncertainty with the robot's uncertainty, covariance C_R, using JCJ^T becomes a sum due to the structure of J for Eq. 75 and Eq. 76.

$$J = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} \quad \textbf{Eq. 75}$$

$$C = J \begin{bmatrix} C_R & \{0\} \\ \{0\} & C_m \end{bmatrix} J^T = C_R + C_m \quad \textbf{Eq. 76}$$

Sensor updates are treated similarly. The means of the distributions for the sensor-based estimates are determined using the constraint equations from 6.2. To add uncertainty to position estimates, Gaussian distributions are aligned tangent to the circle for range, and along the line for bearing. For heading, a one-dimensional Gaussian is centered on the constraint-based mean. These are shown in Figure 25.

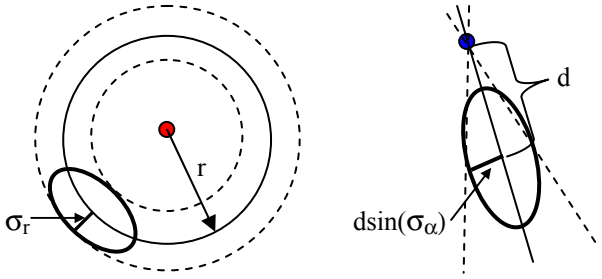

Figure 25. Adding Uncertainty to CBL Sensor Estimates
Left: A Gaussian is centered at the estimate, with width σ_r and length arbitrary.
Right: A Gaussian is centered at the estimate, with width $d\sin(\sigma_\alpha)$ and length arbitrary.

Transforming from range-bearing to Cartesian space for uncertainty is computed using Eq. 77-Eq. 78:

$$C_{mr} = \begin{bmatrix} v_l & 0 \\ 0 & (c\Delta r)^2 \end{bmatrix}, \quad \theta_{maj} = \frac{\pi}{2} - (\alpha + \theta_O) \qquad \textbf{Eq. 77}$$

$$C_{mb} = \begin{bmatrix} v_l & 0 \\ 0 & (d\sin(\sigma_\alpha))^2 \end{bmatrix}, \quad \theta_{maj} = \alpha + \theta_O \qquad \textbf{Eq. 78}$$

where θ_{maj} is the angle of the major axis relative to the global X axis and v_l is the arbitrary variance applied to the long axis of the Gaussian distribution. In this case, since the landmark's position is constraining the robot, the robot's own uncertainty need not be added to the estimate. Multiple sensor-based estimates are combined using the Kalman-Bucy formulation described in 4.2.

In order to prevent robots from becoming lost when teleported or after long periods without observing landmarks, if a sensor-based estimate disagreed by more than 3-σ from the previous estimate, uncertainties were increased to greater than the field size without moving position. In the next step, if this large discrepancy remained, the robot would quickly converge to the new value. If the large discrepancy was anomalous, the robot would quickly converge to the former value.

PCBL was implemented in simulation (for Minnow Robots) and on the Sony quadruped robots. The Sony quadruped implementation was applied to CMPack-02, the CMU Legged-League team for 2002 and the 2002 Legged-League world champions. For the Minnow Robot simulation, $\sigma_r = 0.1r$ and $\sigma_\alpha = 2^o = 0.0349$ radians. For the Sony quadrupeds, $\sigma_r = 0.03r$ and $\sigma_\alpha = 2^o = 0.0349$ radians.

6.5 Collaborative CBL (CCBL/CPCBL)

For robot teams that can communicate and observe one another, there is potential for cooperative localization among teammates. Such collaboration may improve the localization of team members by adding additional observations of pose and, in the case of heterogeneity, spreading information from more robots with greater observational accuracy to those with lower observational accuracy. CPCBL allows for two types of collaborative localization: passive and active.

Passive collaborative localization occurs when an observing robot uses the known location of a teammate as an additional landmark. The observing robot computes range and bearing to the teammate, each of which produces a constraint and subsequent sensor-based estimate that is incorporated into the pose estimate. Active collaborative localization occurs when an observing robot tells a teammate where it observes the teammate to be. The observing robot uses range and bearing to the teammate and its own pose estimate to estimate the teammate's position. If the observing robot can observe relative heading as well as range and bearing, an estimate of heading can also be provided; as our robots cannot detect relative

heading, this update is not performed. Both types of collaboration can be used in either the non-probabilistic or probabilistic implementations of CBL. All investigations have been executed using the probabilistic implementation and are also called CPCBL.

For collaboration using the probabilistic representation, uncertainty estimates must be added to both passive and active estimates. In passive updates, the teammate is treated as a landmark with position known with uncertainty. The initial sensor-based estimate is determined using the approach described in the previous section. Due to the uncertainty of the landmark's position, C_T, the Gaussian distribution provided by the uncertainty in range or bearing is combined with uncertainty in landmark position using JCJ^T. For uncertainties defined in the global frame, covariance matrices are combined as in Eq. 26 and Eq. 27 (Section 4.4), which becomes a sum of the two covariance matrices.

In active updates, the teammate is treated as a target rather than a landmark. The Gaussian distribution provided by the combination of uncertainty in range or bearing, Cm, must be combined with the observing robot's pose uncertainty, C_R, using JCJ^T as in Eq. 26 and Eq. 27 in Section 4.4. Again, multiple observed estimates are combined using the Kalman-Bucy approach from Section 4.2.

6.6 Evaluation

6.6.1 Quantitative Evaluation

Both simulation and physical experiments were conducted using various implementations of CBL. In each case, the robot was commanded to traverse a cyclic path among the landmarks as shown in Figure 26. NPCBL was tested in this manner on the Minnow Robot, using landmarks that can provide both range and bearing (interior, red) and/or landmarks that can provide only bearing (exterior, blue). NPCBL, PCBL and CPCBL were also tested in simulation on the Minnow robots using the four range-bearing landmarks. In each case, results are compared to the odometry reported by the robot or simulated robot.

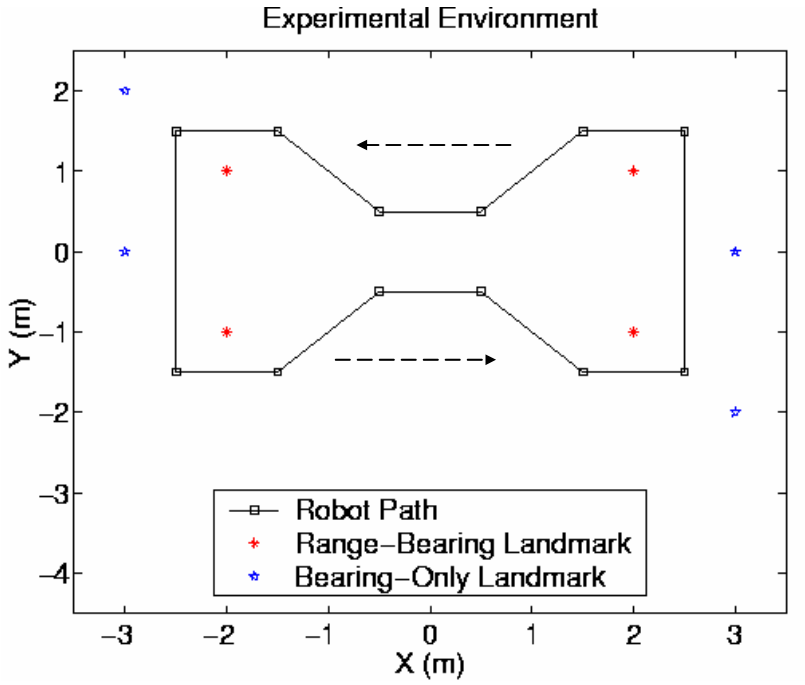

Figure 26. CBL Test Environment
The test environment for CBL is a cyclic path with eight landmarks of various types. Dotted arrows indicate direction of motion along the path.

To test the performance of NPCBL, the Minnow was command to follow the cycle a total of five times, with a total path length of approximately 88 meters. The quality of the position estimate provided by NPCBL is compared to that reported by the Minnow odometry in Figure 27. Total error in position is reported as the sum-squared errors in x and y position.

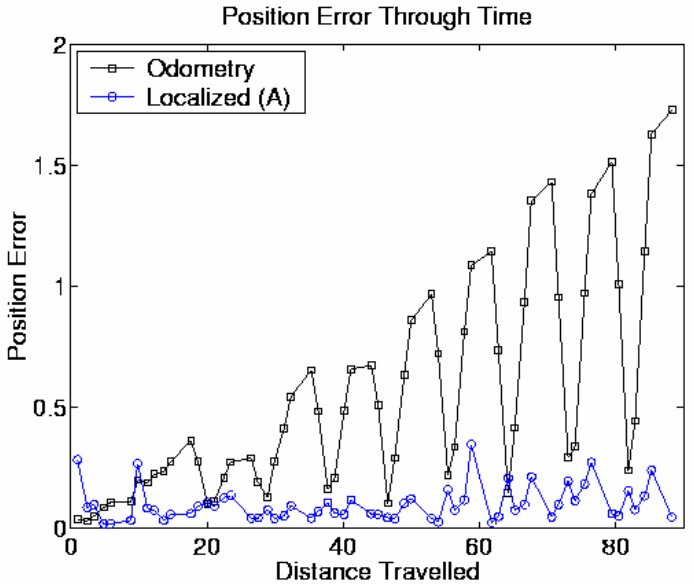

Figure 27. NPCBL Performance Compared to Odometry
Total error in position is compared for NPCBL localization and odometry. Notice odometry error grows unbounded while CBL localization error remains constrained.

As shown, despite the limited representation provided by NPCBL, the robot is able to remain well localized, without the uncertainty growing unbounded, as in odometry. After nearly 90 meters, the robot odometry is lost by more than 1.5 meters, a significant portion of the 3 by 5 meter arena. The CBL results however, never exceed 0.30 meters. Peaks in localization error occur when the robot is furthest away from the landmarks, corner points of the cycle at (-2.5, -1.5) and (2.5, 1.5) where the robot is looking the length of the field to observe landmarks, and only one landmark is clearly observed. The oscillatory nature of the odometry values, which produces minima at the loops ends, is likely due to the cyclical nature of the heading errors as the robot follows a closed path; the errors due to turns cancel out as the robot makes equal left and right turns, which could account for these artificially accurate points.

The quantitative results are summarized in Table 1.

Table 1. NPCBL Position Estimation Results

	Odometry (Best)	Localization (Range Only)	Localization (Range-Bearing)	Localization (Range-Bearing + Bearing Only)
Mean Position Error	0.5592m	0.0994m	0.0719m	0.0627m
$\sigma_{\Delta x}$	0.4378m	0.0822m	0.0578m	0.0562m
$\sigma_{\Delta y}$	0.5624m	0.0920m	0.0620m	0.0482m
Max Position Error	1.7275m	0.3459m	0.2326m	0.1652m
Mean Heading Error	29.6°	5.1°	2.3°	1.7°
Max Heading Error	42°	11°	5°	4°

From Table 1, it is clear that even with little information available NPCBL is able to maintain robot localization to a high level of certainty. Performance improves as more information is available from each landmark (demonstrated by moving from range only to range-bearing) and as more landmarks are provided (range only and range-bearing compared to range-bearing and bearing-only together).

In simulation, using four range-bearing landmarks, different implementations of CBL are compared. In Figure 28, the blue (lower left) robot travels along the cyclic path for five loops while the red (upper center) robot observes and is observed from a stationary location; this allows its positional uncertainty to be discounted. Each robot's field of view is indicated by the arc of the same color.

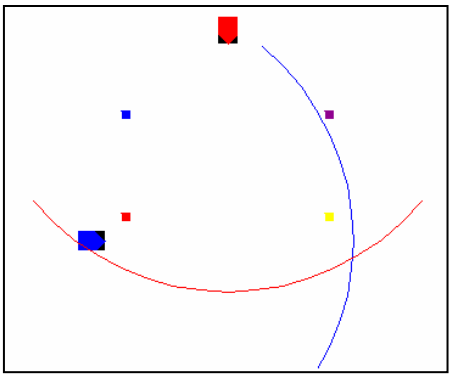

Figure 28. CBL Simulation Setup
Two robots (one driving, one observing) with four landmarks.

For these experiments, the observing robot was provided a sensor that is more accurate than that of the navigating robot. The navigating robot's vision has standard deviations 5% of range and 5.0 degrees, while the observing robot has standard deviations of 0.5% of range and 0.5 degrees. Here, $v_l = (c\Delta r)^2$ in C_{mr} (Eq. 77 above), and $v_l = (d\sin(\sigma_\alpha))^2$ in C_{mb} (Eq. 78 above). This leads to circular sensor-based estimate distributions, which can significantly simply computation.

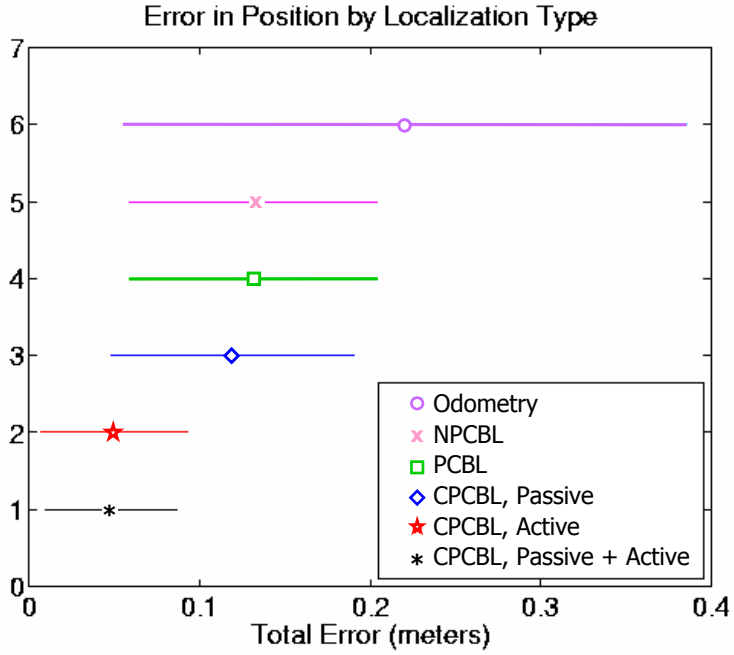

Figure 29. CBL Comparative Performance
CBL, even in its minimal NPCBL implementation, provides significant improvement in maintaining localization. The addition of probabilistic representation provides confidence, but does not improve performance. Collaboration improves performance, but most significantly when high quality information flows to robots with lower quality sensing.

As an additional landmark for passive collaboration, the collaboration of the observing robot produces minimal effect. This is due to the high uncertainty of the navigating robot's sensor. However, the addition of a highly accurate measurement from the observing robot for active collaboration, there is a significant improvement in performance. This result may be specifically useful in such systems with a few high-quality sensors. Not only can robots benefit from their teammates' higher quality measurements (even if only intermittent), but the lack of improvement using those teammates as landmarks may allow for simplifying computation by not performing both active and passive updates. For heading, this improvement is not as dramatic, as robots cannot in this case observe each other's heading and therefore cannot inform teammates about heading. However, since the heading updates use position to infer heading (Eq. 61), improvement in position also improves heading. These results are summarized in Figure 29.

The addition of a probabilistic representation does not improve performance over the hand-tuned weighted average parameters. Thus, NPCBL can provide good localization results in the event that additional computation for uncertainty is not available or when confidence estimates are not required. The probabilistic representation does, however, provide good confidence limits on the estimates. Pose estimates always remained within 3-σ of the actual values.

6.6.2 Qualitative Performance

While ground-truth is not available for comparison during robot soccer games, some qualitative results can be described from these applications. NPCBL was applied to the Minnow Robots in the middle-sized league in 2001. On observing the "go home" behavior (in which the robot returns to a pre-defined point, facing down field), performance using odometry caused the robot to drift significantly, quickly. After only five minutes of play, the robot would be returning to points up to a meter away from home position with 20-degree heading errors, and after ten minutes would be completely lost and it would have to be restarted. In contrast, NPCBL allowed the robots to consistently return to home within 0.5 meters of home and oriented correctly down field, regardless of length of play.

PCBL was applied to the Sony quadrupeds for the legged-league in 2002. Reported positions of the robot agreed within 0.02 meters of the actual truth during laboratory tests when landmarks were not obscured.

6.7 Summary and Discussion

CBL provides a method of localization that can be adapted to the environment, platform, and requirements. A sliding scale of computational complexity allows tuning of computational demands in representation, combining observations, treatment of uncertainty, and collaboration. Representation may be a point or multiple Gaussian distributions. Combining observations may be computed as a weighted average or using the Kalman-Bucy equations. Uncertainty may be ignored, treated as symmetrical Gaussian distributions, or treated as asymmetrical Gaussian distributions. Lastly, collaboration may be implemented bi-directionally, uni-directionally (high quality to low), or not at all. Landmarks may contribute different types of information with different levels of certainty. Performance in each case prevents the robots from growing errors unbounded, with improved bounds as number of landmarks and quality of measurements are increased.

Chapter 7 Target Location Mission

This chapter addresses several key issues regarding the performance of MVERT in a mapping like task:

1. Can MVERT produce trajectories qualitatively appropriate for the task?
2. Can MVERT produce trajectories that improve on individual action selection?
3. Can MVERT produce trajectories that compare in performance to optimal?
4. How does team size affect the performance of MVERT?
5. How do the various parameters of MVERT and the robot affect performance?

7.1 Experimental Summary

The Target Location Mission is to localize stationary targets within an environment, as detailed in 5.3.1. In these experiments, perfect robot localization and motion is assumed. Target location value is applied (Eq. 2, Section 3.3.1). To isolate the impact of other factors on performance, noise is modeled but not introduced into sensor measurements. This provides clarity of interpreting robot trajectories and comparison to individual action selection and the one-step optimal for addressing the first three questions. Key parameters are varied to address the fourth and fifth questions, including step size, sensor model, robot capabilities, and team size. Performance metrics used to evaluate this task are certainty (E_2, E_3) and mission time (E_4), including relative ratios of these metrics. Except for variations listed in the experimental series, the default parameters and capabilities are used. The experimental series, all in simulation, are:

T1. MVERT Performance: MVERT with default parameters and capabilities;
T2. Sensor Noise Model Parameters: MVERT with various sensor noise parameters;
T3. Candidate Move Resolution: MVERT with varied number of candidate moves;
T4. Robot Limitations: MVERT limiting visual range, visual angular field of view, and turning angle;
T5. Individual Actions: MVERT disregarding teammate contributions;
T6. One-Step Optimal: Comparison of MVERT approximation to *set* of moves maximizing value.

Each experiment series is run in one or more of the following environments:

- small: 1 target and up to 4 robots in a 3 by 3 meter area,
- medium: 4 targets and up to 6 robots in a 8 by 6 meter area,
- large: 10 targets and up to 12 robots in a 30 by 30 meter area.

Four different sensor models are used. Bearing uncertainty corresponds to the uncertainty perpendicular to line of sight from robot to target and range uncertainty corresponds to uncertainty parallel to line of sight.

- perpendicular much more accurate than parallel: σ_α=0.5º and σ_r = 0.1r (default),
- perpendicular more accurate than parallel: σ_α =0.5º and σ_r = 0.03r,
- perpendicular and parallel equally accurate: σ_α =0.03(180/ π)º and σ_r = 0.03r,
- perpendicular much less accurate than parallel: σ_α =10º and σ_r = 0.01r.

Several resolutions of candidate move headings are compared, while step size is fixed at 0.4 meters:

- high: 180 points at 2º (default),
- good: 72 points at 5º,
- medium: 36 points at 10º,
- low: 12 points at 30º,
- very low: 8 points at 45º.

Robot capabilities are varied and investigated in several combinations. Capabilities include motion:

- holonomic motion: 360 degree turns permitted (default),
- non-holonomic motion: limited turn per step of ±50 degrees;

vision range:

- unlimited range (default),
- limited range of 2.25, 3.0, and 10.0 meters;

and visual angular field of view:

- unlimited angular field of view of ±180 degrees (default),
- limited angular field of view of ±50 degrees.

Each experimental series is discussed to highlight the impact of specific parameters. In each Section, representative figures are shown. In some illustrations, robots are shown larger than actual size or omitted for clarity of presentation; apparent overlap in robot positions is an artifact of this and does not represent actual collisions. Experiments are run until equilibrium is reached with robots in local optima: 20 steps for the small environment, 40 steps for the medium environment, and 150 for the large environment.

7.2 MVERT Performance (T1)

The MVERT Performance experiments are meant to validate the ability of MVERT to generate trajectories that provide good team observations, evaluate the effects of varying team size, and to serve as a basis of comparison with other approaches. It is intended to test the hypothesis that the benefits of making *team* decisions can be reaped without resorting to more computationally expensive methodologies and address questions 1 and 4 regarding appropriateness of trajectories and affects of team size. MVERT performance is evaluated in all environments, and the following specific experiments were conducted for series T1:

A. 1, 2, 3, and 4 robots observing in the small environment,
B. 2, 3, 4, and 6 robots observing in the medium environment,
C. 2, 3, 4, 6, 8, and 12 robots observing in the large environment.

Small environments enable individual behaviors to be clearly observed while larger environments provide a venue for demonstrating complex behavior. This series also provides an opportunity to establish whether there is a maximum team size above which additional robots do not substantially contribute and, in cases, interfere with teammates. The medium environment has three configurations: symmetrical, asymmetrical, and perpendicular (2 robots). Typical MVERT trajectories are illustrated in Figure 30-Figure 33.

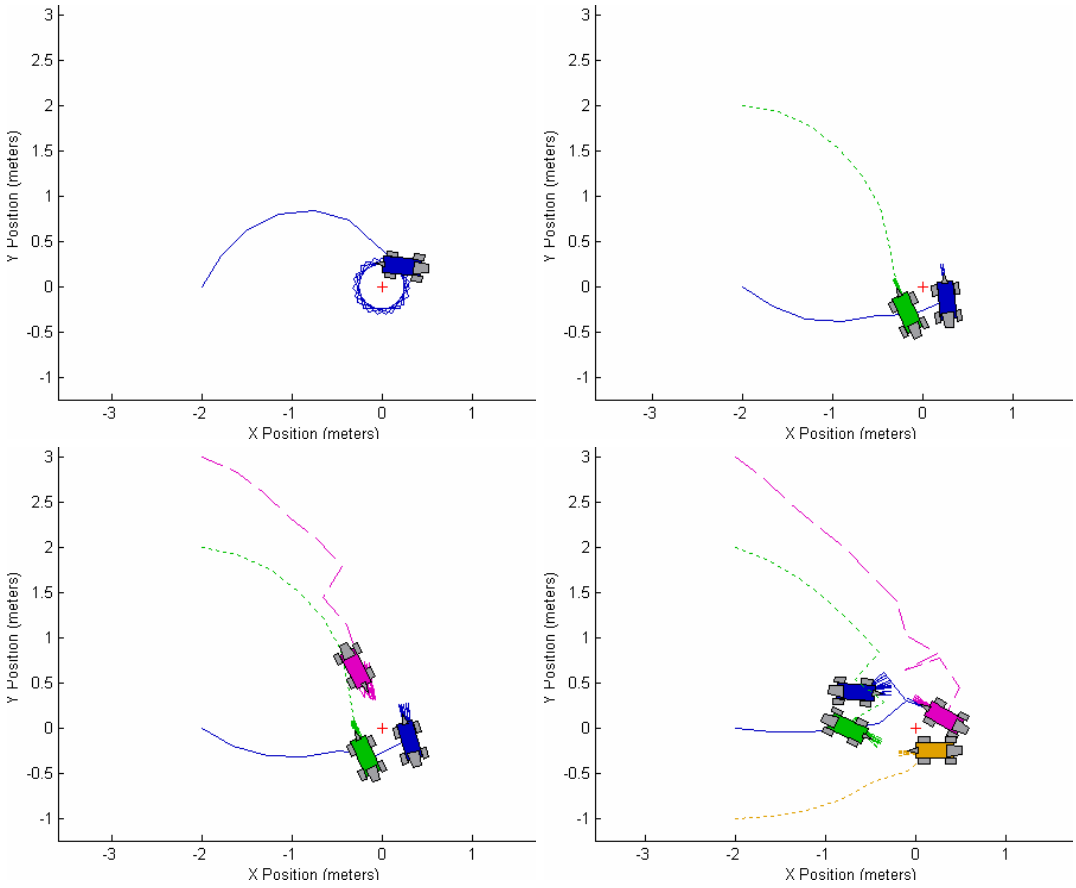

Figure 30. Experiment A, T1: Small Environment Target Location Trajectories
In maximizing value (minimizing uncertainty area), measurements from multiple points of view produce the largest improvements. These measurements are made by the same robot when required (upper right), and concurrently by multiple robots when possible (upper right and lower).

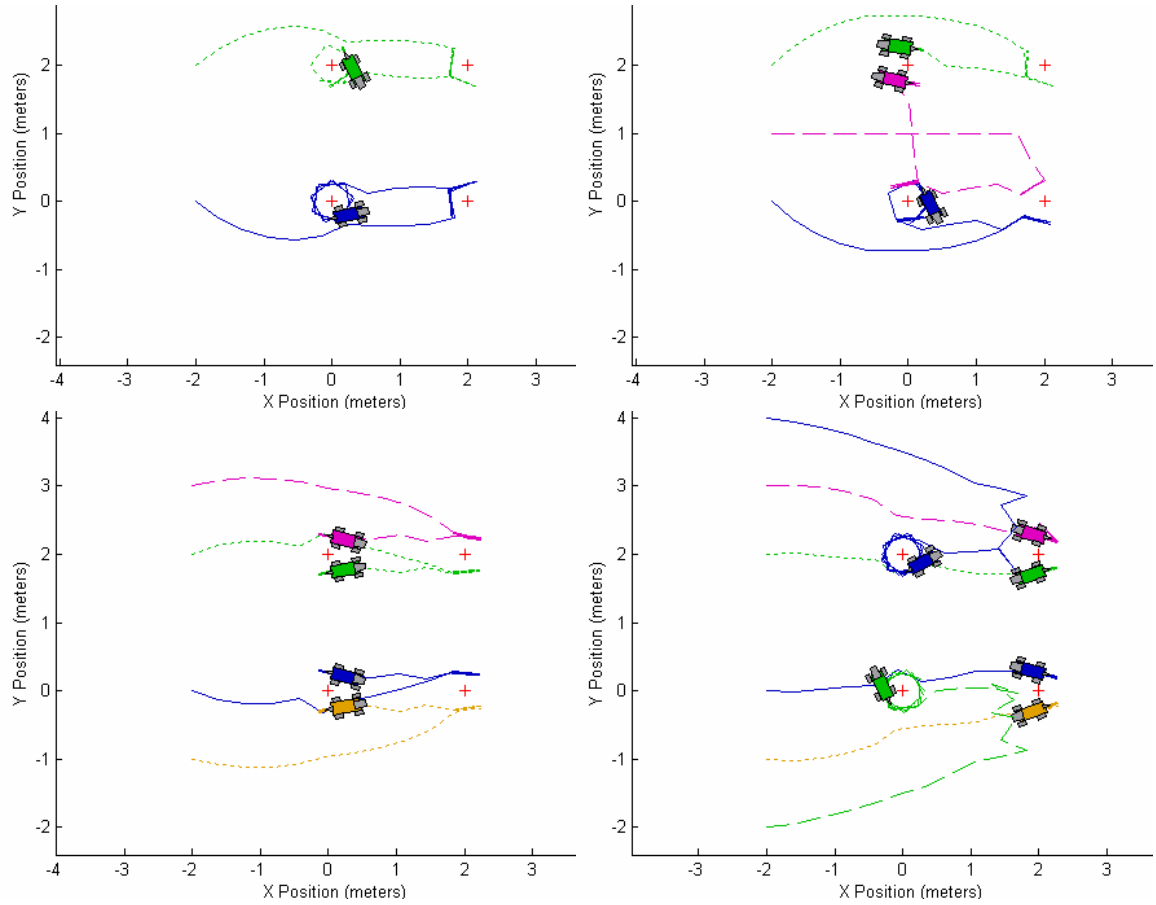

Figure 31. Experiment B, T1: Medium Environment Target Location Trajectories
Symmetrical configuration shown. With multiple targets, maximizing value requires reducing uncertainty on all targets. To reduce uncertainty most quickly, multiple points of view must be obtained on each target. To thus maximize value, multiple robots are distributed among the targets. With small teams (top), additional increase in value is obtained by arcing around targets, increasing the variety in point of view, then circling single targets. With larger teams (bottom) and diversity in of point of view, need for this additional arc is partially overridden by the improved measurements obtained by moving more directly close to targets. Here, robots each observe from points on half the circle rather than fully circling.

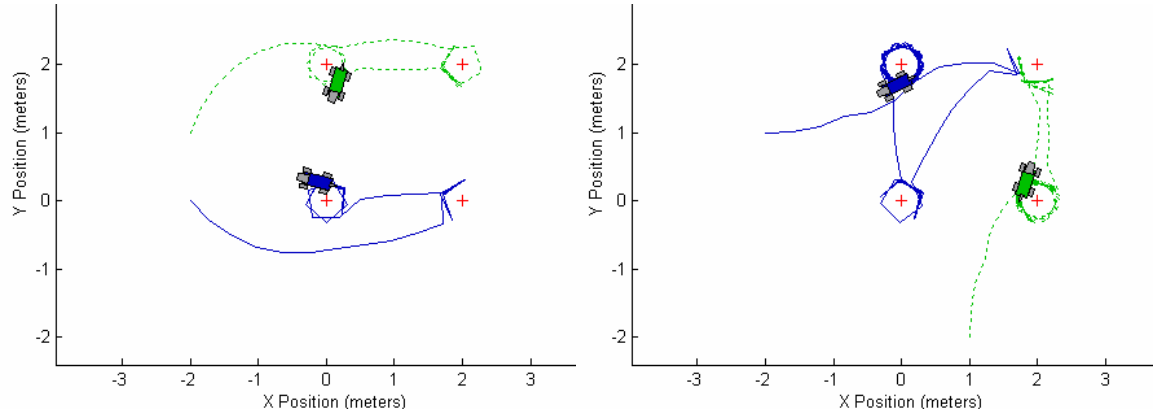

Figure 32. Experiment B, T1: Medium Environment Target Location Trajectories
Value is maximized by taking measurements from different points of view and by taking less noisy measurements closer to targets. *Left:* If robots start close together with similar points of view (asymmetrical), maximizing uncertainty moves them apart. *Right:* If robots start with very different points of view, maximizing uncertainty moves them more directly toward targets (perpendicular).

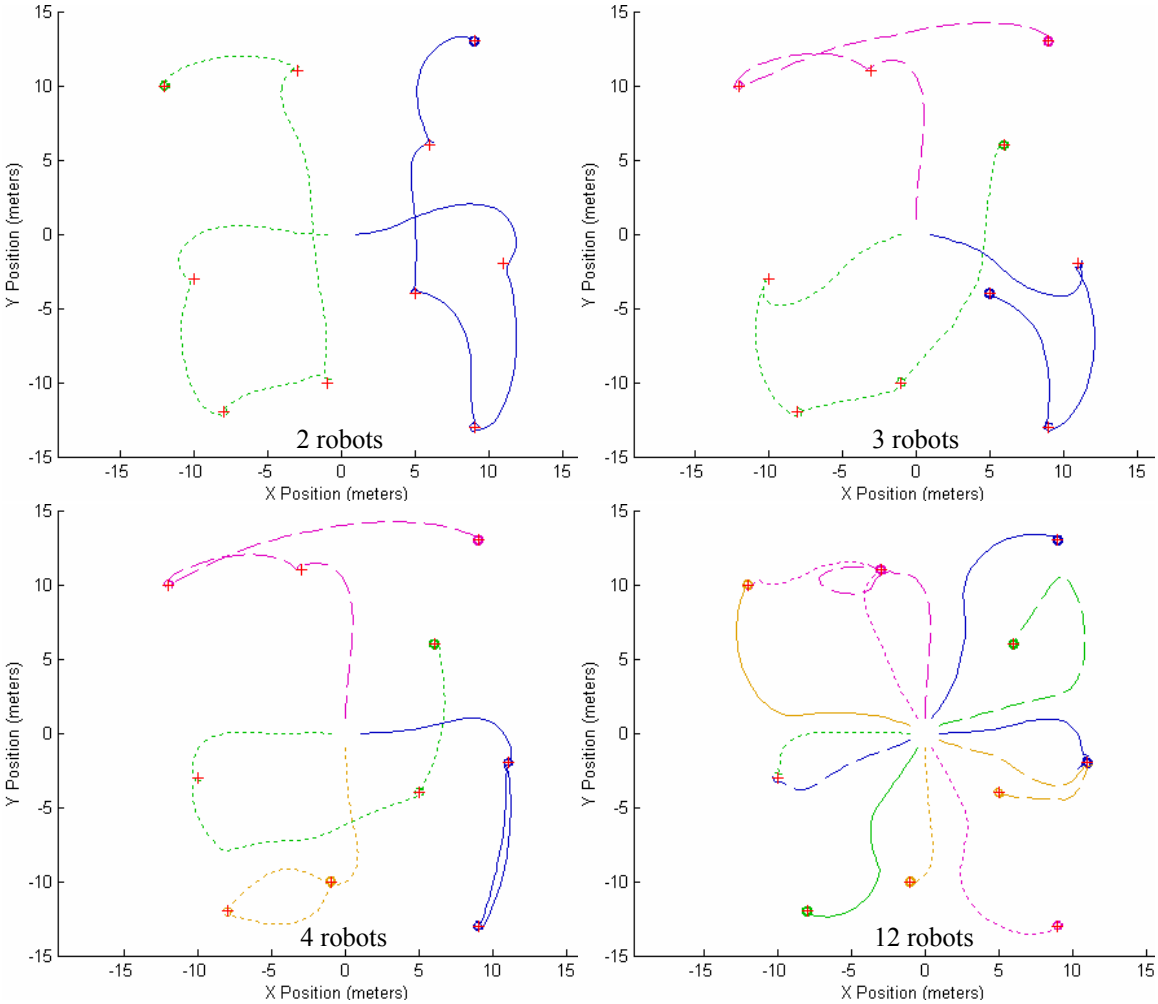

Figure 33. Experiment C, T1: Large Environment Target Location Trajectories
To maximize value, close observations of all targets must be made from multiple view points of view. In small teams (top), robots can obtain these measurements by each moving close to different targets. In larger teams (bottom), robots move close to different targets, but further improve value by taking simultaneous observations from different points of view on single targets (note pairs of robots converging on individual targets in bottom right).

Additional experiments were run (3-6 robots in the asymmetrical medium environment, and 6 and 8 robots in the large environment). These experiments provide similar results to those shown in the example cases.

A key observation is made in both environments, addressing question 4 regarding the effects of team size on performance. As team size increases, the performance of MVERT improves. In terms of E_2 (value) at any given step, E_3 (maximum uncertainty) at any given step, and E_4 (time to mission completion, defined as reaching a maximum uncertainty below a desired threshold), the rates of improvement of these metrics increase with team size. Additionally, it is observed that for each environment there is team size above which a decrease in the rate of improvement resulting from the additional team members. This point is reached at four robots in the medium environment and six or eight robots in the large environment. Thus, as team size increases performance improves, but at some team size (determined by the environment), a point of diminishing returns is reached where additional robots do not drastically improve performance.

Quantitative results are summarized in Table 2 and Table 3. The steps chosen for illustration in the tables were selected at a point prior to the robots reaching local optima circling individual targets. In the tables, a dashed entry indicates that the desired uncertainty level was not reached during the experiment. Larger values (values closer to zero) and fewer steps indicate better performance.

Table 2. MVERT Target Location Performance: Medium Environment

Robot Configuration	E_2 at Step 12	E_4: Steps to $E_3 < 0.01$	E_4: Steps to $E_3 < 0.005$	E_4: Steps to $E_3 < 0.002$	E_4: Steps to $E_3 < 0.001$
2 robots, symmetrical	-7.66×10^{-5}	9	11	24	-
2 robots, asymmetrical	-9.17×10^{-5}	10	12	31	-
2 robots, perpendicular	-11.52×10^{-5}	7	12	28	-
3 robots, symmetrical	-7.67×10^{-5}	8	10	27	-
3 robots, asymmetrical	-5.04×10^{-5}	8	10	28	-
4 robots, symmetrical	-3.65×10^{-5}	7	9	17	29
4 robots, asymmetrical	-3.68×10^{-5}	7	10	12	32
6 robots, symmetrical	-2.97×10^{-5}	6	9	11	28
6 robots, asymmetrical	-2.98×10^{-5}	6	9	12	28

Table 3. MVERT Target Location Performance: Large Environment

Robots	E_2 at Step 60	E_4: Steps to $E_3 < 0.1$	E_4: Steps to $E_3 < 0.01$	E_4: Steps to $E_3 < 0.005$	E_4: Steps to $E_3 < 0.001$
2	-58.4×10^{-4}	16	165	170	-
3	-19.0×10^{-4}	14	123	142	-
4	-12.4×10^{-4}	12	199	128	-
6	-3.70×10^{-4}	10	57	116	-
8	-1.04×10^{-4}	9	42	48	-
12	-0.0314×10^{-4}	7	38	47	72

Figure 34 and Figure 35 illustrate the effects of team size on value. Value improves with larger teams.

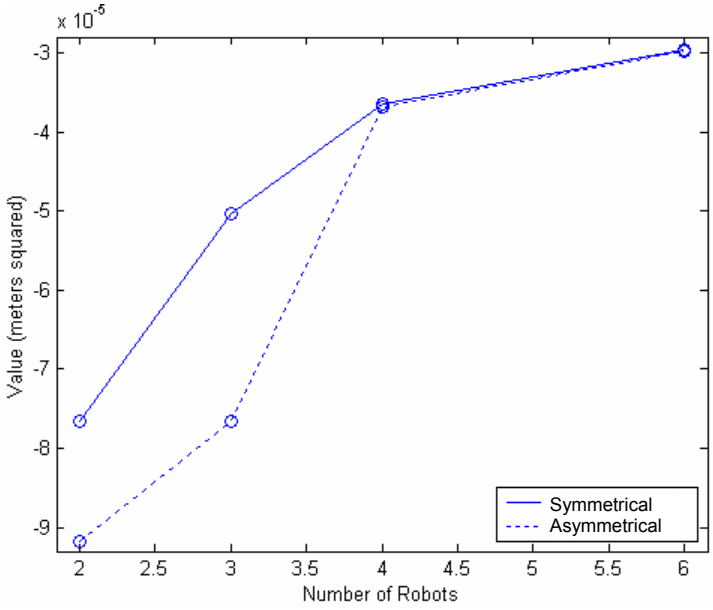

Figure 34. Value Versus Team Size in Medium Environment Target Location

The total value of the map is shown for step 12. As team size increases, value increases, improving certainty of the map. As team size increases, the effects of different initial conditions are reduced, resulting in smaller differences in final value. Note initial symmetry improves results.

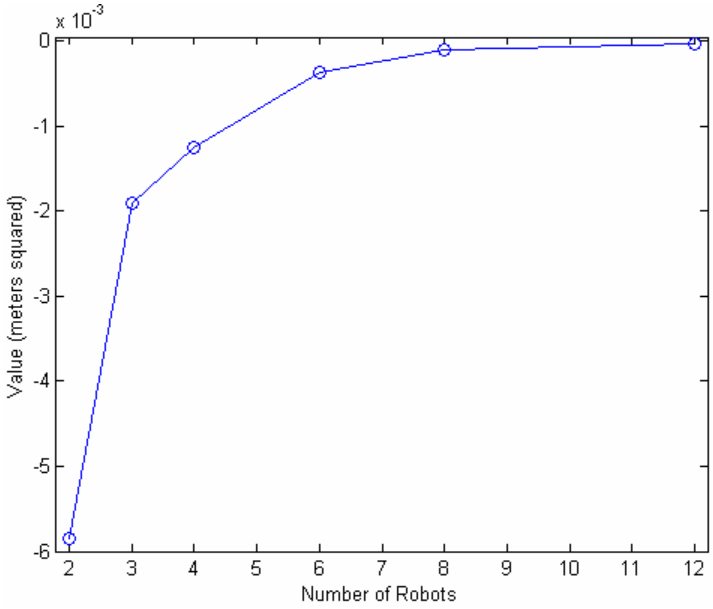

Figure 35. Value Versus Team Size in Large Environment Target Location
The total value of the map is shown at step 60. As team size increases, value increases, improving certainty of the map.

E_2, value, rises quickly with time, at faster rates for larger team size, and then begins to level off. This is illustrated in Figure 36 (for the symmetrical robot configuration) for the medium environment and Figure 37 for the large environment. This same qualitative result occurs for the other configuration in the medium environment.

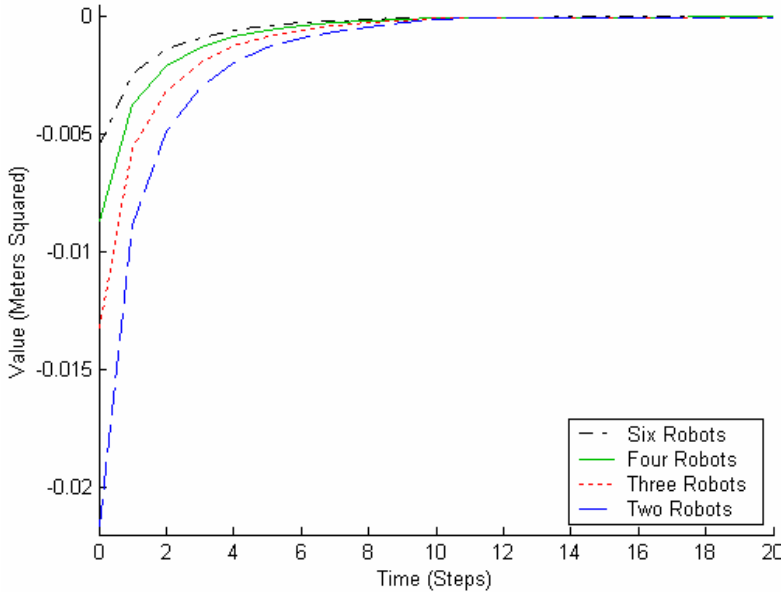

Figure 36. Target Location Value (E_2) Versus Time in Medium Environment Target Location
Value increases more quickly as team size increases. Higher values (near zero) indicate better performance.

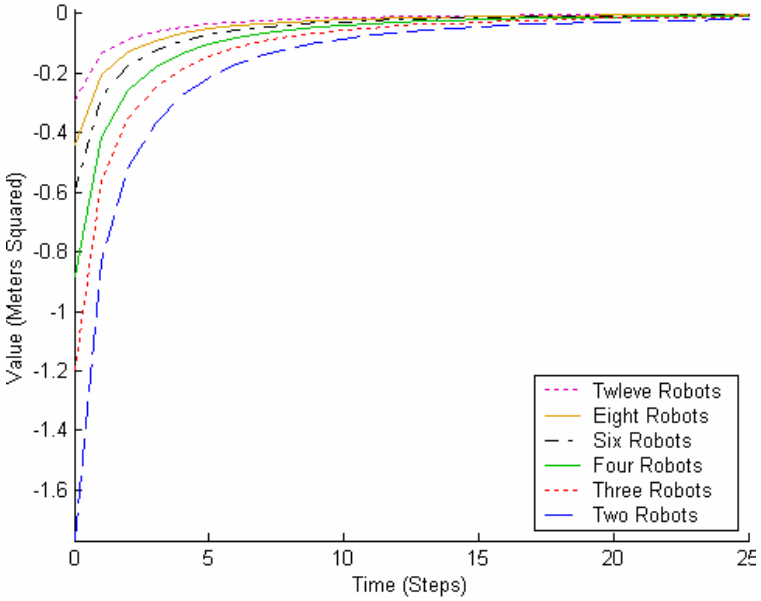

Figure 37. Target Location Value (E_3) Versus Time in Large Environment Target Location
Value increases more quickly, improving performance, as team size increases.

For both Value (E2) and Maximum Uncertainty (E3), numbers closer to zero indicate better performance. A similar result is obtained looking at E_3, maximum uncertainty, as shown in Figure 38 and Figure 39.

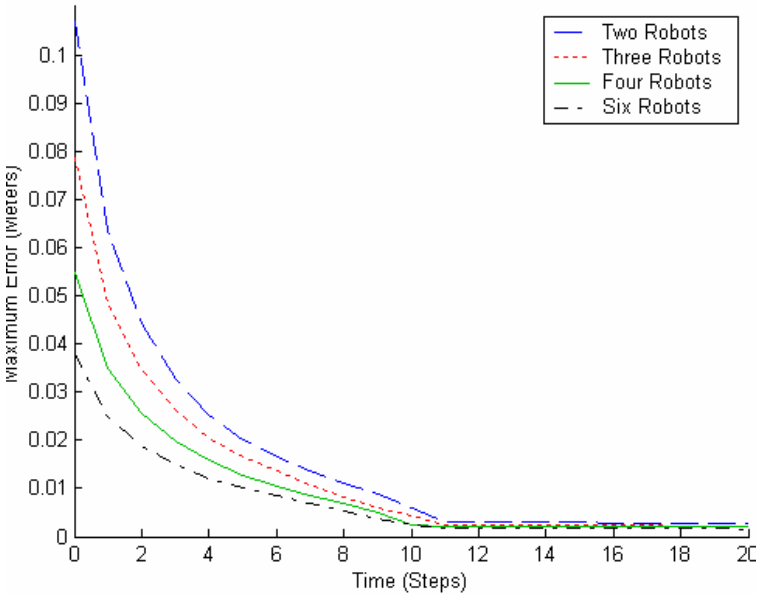

Figure 38. Maximum Uncertainty (E_3) Versus Time in Small Environment Target Location
Maximum Uncertainty decreases more quickly as team size increases. Smaller uncertainties (nearer to zero) indicate targets are better known, and thus indicate better performance.

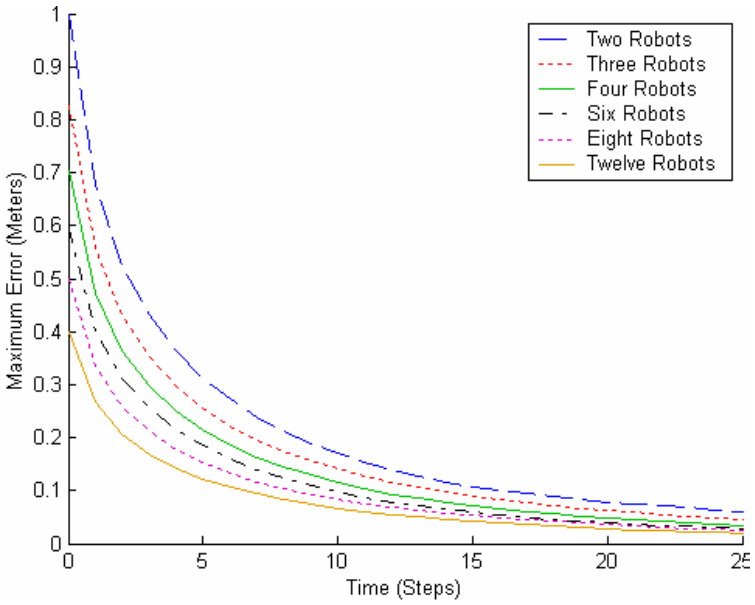

Figure 39. Maximum Error (E_3) Versus Time in Large Environment Target Location
Maximum uncertainty decreases more quickly with larger teams. Lower uncertainty is better performance.

The results of these experiments show that by maximizing value, robots automatically spread out, producing complementary views of targets, triangulating to compensate for the lower quality of range measurements (parallel to line of sight) compared to bearing measurements (perpendicular to line of sight). As team size increases, this spreading results in more effective triangulation, reducing uncertainty in all dimensions. Robots using individual action selection travel around targets, triangulating with respect to observations over time (Figure 30, top left). Multiple robots spread out and observe targets from complementary axes (orthogonal for two, triangular for three). This can be most clearly seen in Figure 30, where robots are distributed around the single target at equal angles (90º for two and four robots, 120º for three). In Figure 31, two robots arc to move to perpendicular views of the four targets. A third balances between to equally contribute; prior to passing beyond targets, this robot turns to obtain a closer view on the target with the larger uncertainty. As robots are added, less arcing is required, as more diverse axes are observed at each measurement. At targets, robots circle around the target, obtaining complementary views and additional measurements, until another target becomes more attractive (providing a greater effect on value) due to its higher uncertainty. Circling may manifest as full circles (Figure 31, top left) or as partial circles (Figure 31, lower left); these cases are equivalent. Multiple robots circle in half circles (Figure 31, lower right), remaining distributed and avoiding collisions (paths are obscured by the robot, but are similar to those in Figure 31, lower left). Slight asymmetries in paths are due to small differences in rounding.

Generally, more robots allow for more specialization. Use of MVERT results in automatic distribution of the targets among the robots, with each target being closely examined by at least one robot (Figure 31 and Figure 33). If targets outnumber robots, targets are divided up among robots (Figure 31, bottom right). If robots outnumber targets, robots automatically form temporary groups to quickly obtain complementary observations (Figure 33, bottom right). As the team moves, uncertainties on close targets drop quickly, giving the more distant targets higher priority (greater potential effect on value). This greater effect on value makes robots concentrate on these far targets, even if multiple robots initially head toward the same target, and later return to the early targets as the far targets become more certain than the early ones. This can be most clearly seen in Figure 33, where robots initially move past targets, such as the one at (5, -4), only to return later to improve estimates.

The results of these experiments also indicate that varying the initial position relative to the targets leads to markedly different choices of path, illustrating the ability of MVERT to select team measurements that most benefit the overall uncertainty in all targets. At each step, robots move to complement the existing pdf. Different initial conditions alter the size and orientation of the Gaussians, requiring different complementary observations. Different relative teammate positions must also be complemented with

different motions. When starting closer together, robots move apart to maximize value, allowing them to obtain perpendicular views. When starting with orthogonal views, robots can proceed more directly toward targets. This is most clearly illustrated in Figure 31; robots starting near each other (left) must move apart to best improve uncertainty, while robots starting with more perpendicular views (right) move more directly toward the targets, already benefiting from complementary views.

7.3 Sensor Noise Model Parameters (T2)

Varying sensor noise model parameters allows investigation of MVERT adaptability to differences in sensing, partially addressing question 5 (effects of robot capabilities on performance). As MVERT minimizes uncertainty on all targets at each step, it favors measurements that reduce the uncertainty the most. The position best able to do may depend highly on the sensor footprint. The specific experiments conducted for series T2 are:

D. 1 robot observing in the medium environment with four sets of sensor noise parameters,

E. 2 robots observing in the medium environment with four sets of sensor noise parameters.

The chosen parameters reflect a wide range of possible models, with standard deviation ratios of 18, 12, 3, and 1. Data for sensor parameters "perpendicular much more accurate than parallel" is from Experiment B (Figure 31). Trajectories produced using two of the sensor parameter models (perpendicular more accurate than parallel and perpendicular and parallel equal) sets are shown in Figure 40. Trajectories for the sensor noise model "parallel much more accurate than perpendicular" are very similar to those for "perpendicular much more accurate than parallel," and are therefore not shown. The symmetrical configuration is used.

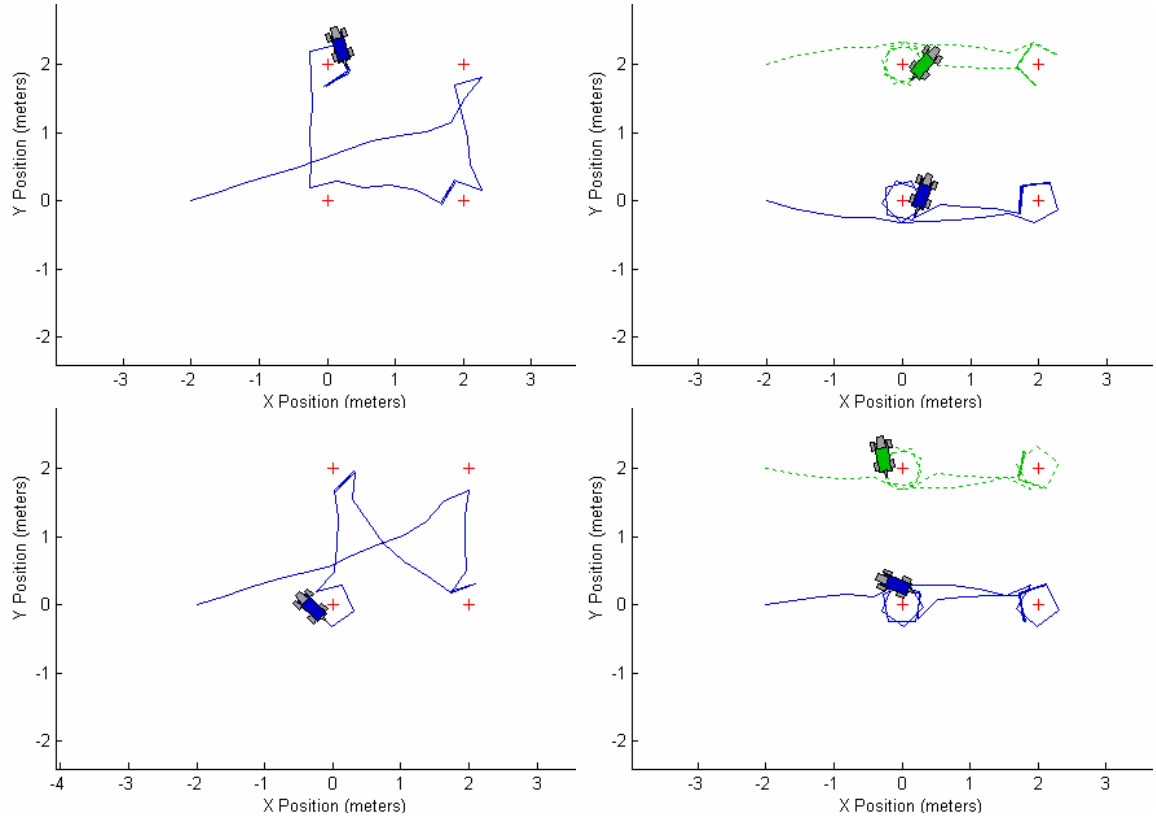

Figure 40. Experiments D and E, T2: Varying Sensor Model in Target Location
Top: With perpendicular slightly more accurate than parallel, maximizing value depends less on point of view and more on proximity. Thus, robots move more directly to targets and circle to obtain additional close measurements. Slight arcing is evident in the team of two. *Bottom*: With equally accurate perpendicular and parallel, no benefit to value arises from varying point of view; thus, robots move even more directly toward targets and arcing is eliminated entirely in the team.

Quantitative results are summarized in Table 4. Value differences arise from the different areas produced by the sensor parameters. Relative E_2 is the ratio of current value to value upon initialization (step 0, after observing and prior to moving). Ratios closer to zero indicate greater improvement. Relative improvement at 12 steps is approximately the same for each sensor model, with an improvement by a factor of 195-285.

Table 4. Target Location Performance with Varying Sensor Noise Parameters

Sensor Noise Parameters	E_2 at Step 12	Relative E_2 at Step 12
σ_α=0.5º, σ_r = 0.1r	-7.66×10^{-5}	3.54×10^{-3}
σ_α =0.5º, σ_r = 0.03r	-5.64×10^{-5}	3.77×10^{-3}
σ_α =0.03(180/ π)º, σ_r = 0.03r	-10.3×10^{-5}	3.51×10^{-3}
σ_α =10º and σ_r = 0.01r	-33.5×10^{-5}	5.13×10^{-3}

With large differences in quality (greater certainty along one dimension than another), MVERT selects positions to reduce the long dimension of the Gaussian uncertainty distribution. Robots make large arcs, rapidly varying the axis of observation and triangulate. Circling behavior around individual targets both allows triangulation and keeps the robot as close to the target as possible, obtaining good quality measurements. Smaller differences in quality (such as the low bias toward bearing) encourages some triangulation, but paths are much straighter and focus more quickly on the less well known targets that are further from the observer. When the uncertainty is exactly equal (producing circular Gaussian distributions with each measurement and update) robots drive straight toward targets, not needing to triangulate, and focusing on the least well known (far) targets. In this case, circling targets occurs to keep the robots close to the targets, to provide the best quality measurement at each step.

7.4 Candidate Move Resolution (T3)

Varying candidate move resolution enables evaluating robustness of MVERT to reducing computational complexity at each step by reducing the number of candidate moves, investigating question 5 (parameter effects). It also illustrates effects of this reduction as a trade-off with quality. The T3 experiments are:

F. 2 robots observing in the medium environment with five resolutions,

G. 4 robots observing in the medium environment with five resolutions.

The data for resolution of 2º is taken from Experiment B, T1. The choices are meant to represent the range of options while remaining physically realizable. 2º is slightly smaller than the smallest turn that both the Sony Quadruped and the Minnow Robot can accurately resolve. 45º reduces candidate points to fewer than 10 and still samples the space. Example trajectories, symmetrical configuration, are shown in Figure 41.

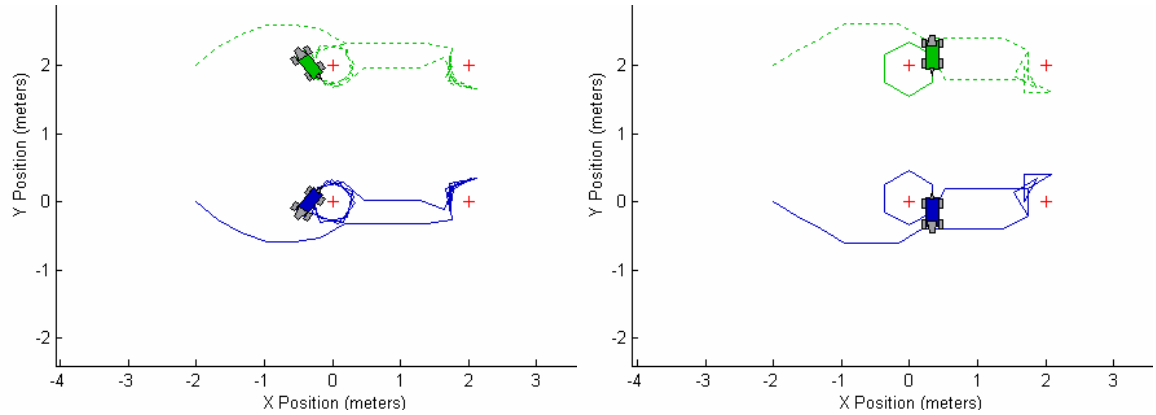

Figure 41. Experiment F, T3: Varying Candidate Move Resolution in Target Location
Left: Resolution at 5º. *Right:* Resolution at 45º. Maximizing value requires robots to arc around targets, thus producing complementary observations. Varying resolution affects smoothness and accuracy of selecting the maximal-value point, but not the qualitative approach to target location.

Quantitative results are shown in Table 5. With exception of the coarsest resolution, 45º, reducing resolution does not substantially alter MVERT performance. Occasionally, the different trajectory puts the robot in a position to take a single better measurement, producing higher value than at higher resolution, as for 2 robots with 5º and 10º. Dashes represent uncertainty levels not reached. Ratio is value for the new resolution divided by value at resolution 2º; ratios over 1 indicate higher uncertainty/performance drop.

Table 5. Effects of Candidate Move Resolution on MVERT Target Location Performance

Robots	Resolution	E_2 Ratio at Step 12	E_4 Ratio: Steps to E_3 <0.01	E_4 Ratio: Steps to E_3 <0.005	E_4 Ratio: Steps to E_3 <0.002	E_4 Ratio: Steps to E_3 <0.001
2	5º	1.017	1	1	1.042	--
4	5º	1.017	1	1	0.882	1.103
2	10º	1.018	1	1	1.042	--
4	10º	1.112	1	1	1.294	1.172
2	30º	1.155	1	1	0.958	--
4	30º	1.301	1	1	1.353	1.345
2	45º	1.413	1	1	1.125	--
4	45º	1.491	1	1.111	1.353	1.310

Changing candidate move resolution changes the smoothness and quality of trajectories, but not the general approach. Robots initially arc to maximize value by obtaining complementary observations, triangulating with teammates. After targets are partially known, robots circle targets to improve estimates on these targets. The primary effect of lowering the resolution is that trajectories become less smooth, and the resulting quality of the map is reduced, as the quality of each measurement is slightly reduced. Additionally, robots move more directly toward targets, because larger turns do not reduce the distance to the target as quickly, and thus do not reduce uncertainty as quickly. This impacts larger teams more; as each measurement is further from the maximal-value point, divergence compounds with more robots.

7.5 Robot Limitations (T4)

Varying robot capabilities enables investigating MVERT's robustness to sensing and motion limitations, addressing question 5 regarding effects of parameters, and illustrates how limitations are automatically handled within the MVERT architecture. The experiments performed for series T4 are as follows:

H. 2, 3, 4, and 6 robots with vision limited to 3.0 meters in the medium environment,

I. 2, 3, 4, and 6 robots with vision limited to 2.25 meters in the medium environment,

J. 2, 3, 4, and 6 robots with vision limited to 50º in the medium environment,

K. 2, 3, 4, and 6 robots with turn angle limited to 50º in the medium environment,

L. 2, 3, 4, and 6 robots with vision limited to 3.0 meters and 50º and turn angle limited to 50º in the medium environment,

M. 2, 3, 4, 6, and 8 robots with vision limited to 10.0 meters n the large environment,

N. 2, 3, 4, 6, and 8 robots with vision limited to 5.0 meters in the large environment,

O. 2, 3, 4, 6, and 8 robots with vision limited to 50º in the large environment,

P. 2, 3, 4, 6, and 8 robots with turn angle limited to 50º in the large environment,

Q. 2, 3, 4, 6, and 8 robots with vision limited to 10.0 meters and 50º and turn angle limited to 50º in the large environment.

The range limitations of 3.0m in the medium environment (C1) and 10.0m in the large environment were chosen to cover average distance between targets, including the buffer zone on either side. This does not allow robots to see all targets initially, but does allow robots observe new targets after approaching known targets for closer observation. Limiting vision to 2.25m in the medium environment limits observation of

multiple targets to positions between targets. As the Aibo has an effective 220º angular field of view and 360º turn angle, limitation values are chosen based on the Minnow Robot. The 50º visual angle limitation is based on the Minnow Robots' 55º field of view. The 50º turning angle limitation is based on the sharpest in-place turn the Minnow can make without driving forward. Trajectories are in Figure 42 - Figure 45.

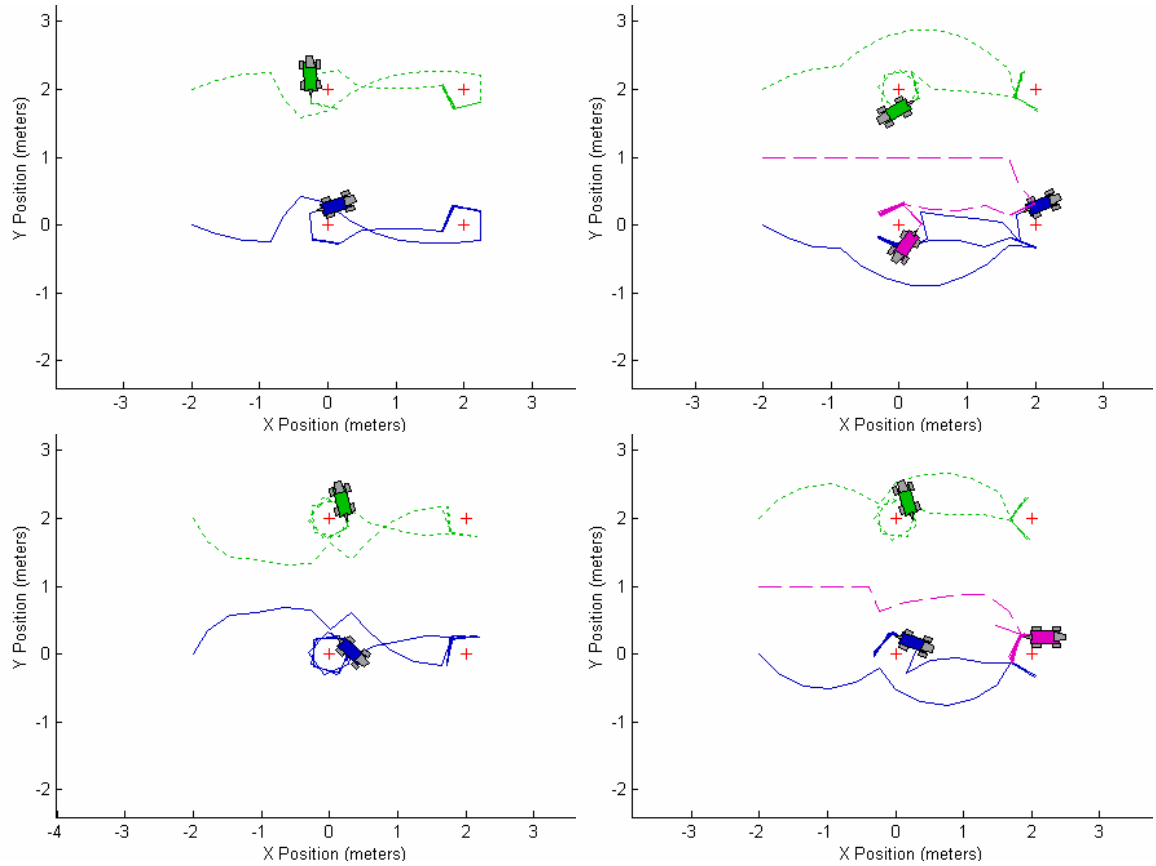

Figure 42. Experiments H and I, T4: Limiting Visual Range in Target Location
Top: Vision range 3.0 meters. *Bottom*: Vision range 2.25 meters. Uncertainty reduction is better achieved by obtaining multiple measurements on targets at each step; thus robots select actions which maintain overlap of field of view. Larger teams have more ability to spread out and obtain multiple points of view while maintaining visual overlap to maximize value.

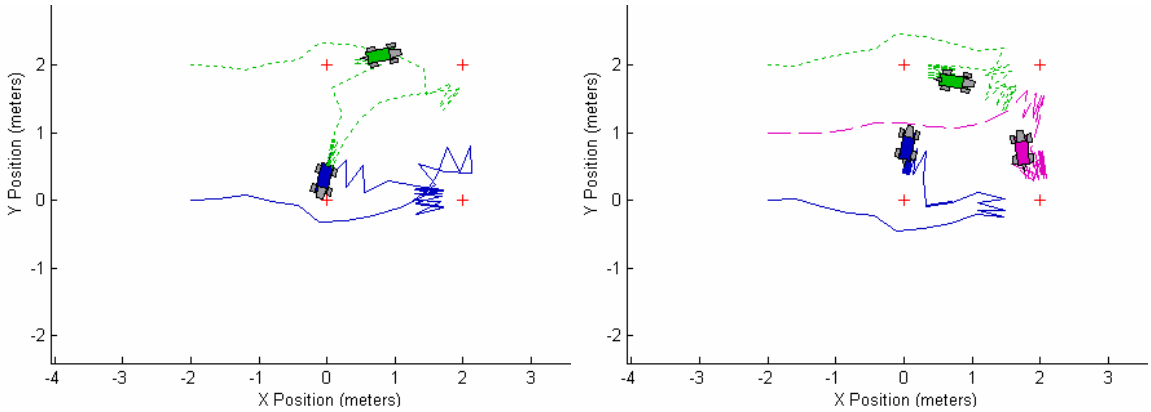

Figure 43. Experiment J, T4: Limiting Visual Angle in Target Location
Visual angle is limited to 50º. Robots can maximally improve value within the local area by turning to effectively increase the angular field of view. Once the most productive observations are made, maximizing uncertainty requires moving closer to less well-known targets.

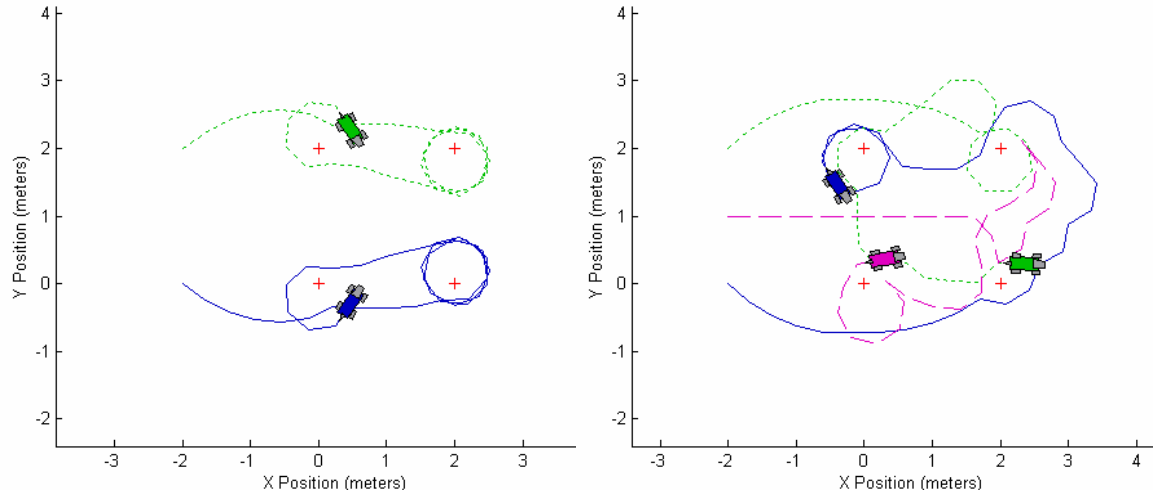

Figure 44. Experiment K, T4: Limiting Robot Turning Angle in Target Location
Turning angle is limited to 50º. Robots unable to reach the higher-value observations that could occur at a sharp change in heading must turn more gradually and improve value more slowly.

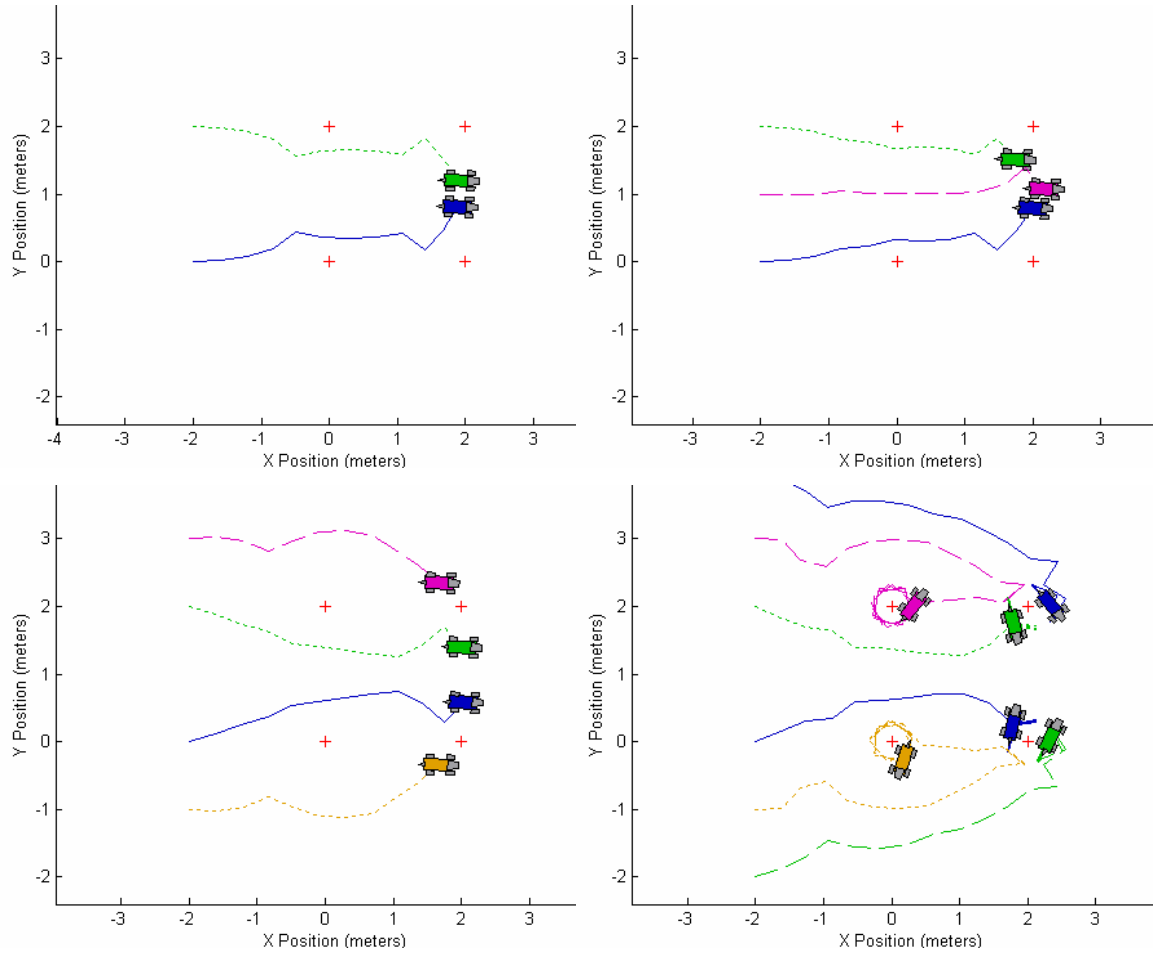

Figure 45. Experiment L, T4: Combining Robot Limitations in Target Location
With multiple limitations, robots become stuck in local optima when no valid moves result in observable targets. Otherwise, robots behave similarly, spreading to maximize value while maintaining visual overlap.

Quantitative results for the medium environment are summarized in Table 6. Limitations result in reduced values, though MVERT still produces maps with high value. E_2 ratio is value the limited robots over value for unlimited robots (larger values indicate greater relative performance). Where the E_2 ratio is less than one, fewer targets were observed by limited robots and thus fewer targets contribute to the total area in E_2.

Table 6. Robot Limitations in Medium Environment Target Location: Value (E_2)

Robots	E_2 Ratio at Step 12				
	Range = 3.0m	Range = 2.25m	Bearing = 50º	Turn = 50º	All (Range = 3.0m)
2	0.838	1.23	1.79	1.07	3.59
3	1.60	1.22	2.07	1.59	3.37
4	1.50	1.26	1.95	1.13	2.96
6	1.71	1.40	1.90	1.14	2.58

Additional quantitative results are shown in Table 7. Each limitation slows the uncertainty reduction somewhat. Adding all limitations, without introducing additional parameters to force exploration, leads robots to a local optimum where no moves can contribute to reducing error, as all targets will remain out of view regardless of action selected. Range most effects the time, as a limited bearing can be quickly overcome by turning within the local area, but limited range may require longer distance travel to cover the space. The more limited the range, the longer it will take to cover the space and reach a given map quality.

Table 7. Robot Limitations in Medium Environment Target Location: Time to Map Quality (E_4)

Robots	E_4 Ratio: Steps to $E_3 < 0.01$				
	Range = 3.0	Range=2.25	Bearing = 50º	Turn = 50º	All (Range = 3.0)
2	1.11	1.22	1.00	1.00	1.33
3	1.00	1.13	1.00	1.00	1.50
4	1.14	1.29	1.00	1.00	1.14
6	1.17	1.33	1.00	1.00	1.17

Limiting the visual range of the robots forces robots to concentrate on targets sequentially as they are observed. The collaboration of teammates is limited, as not all robots can contribute to the observation of all targets. While robots are close enough together to share information, robots act as in the unlimited case, arcing around targets to get complementary observations. As robots move further apart, they compensate for the loss of teammate contributions by triangulating alone. The desirability to share information with teammates is most obvious in the case Experiment I (Figure 42, bottom). Here, robots stay within information sharing range rather than to more fully triangulate, producing an overall greater improvement in knowledge (higher value). This is compared to Experiment H (Figure 42, top), where triangulation and information sharing can occur without conflict due to the slightly larger vision range.

Limiting the visual angular field of view causes robots to move locally to cover an area more thoroughly before moving on to other locations (Figure 43) Obtaining new views from the present location maximizes value at first, then is overshadowed by moving to a new (less well known) location. Limiting vision angle prevents robots from moving beyond where targets are in range, and instead back up and turn to maintain visual observation. This is apparent in the jagged motions of robots as they circle around a target before moving on to another target. When multiple robots can observe a target together, they take up complementary observation angles on the target, and then turn approximately in place to improve coverage.

Limiting the robot's turn angle primarily limits the amount of arcing robots can do around targets, potentially causing trajectories to flatten slightly, and causing robots to circle targets at a greater distance (Figure 44). With multiple robots observing, if robots move past targets to obtain a good observation location, they must then loop around in order to reacquire observations. This is enabled by giving value to locations that move robots closer to targets, even if they are not observable, when moves have equal value.

Combining all of these limitations together produces the strongest effects on robot performance (Figure 45). To maximize value, robots compensate for limited visual angle by moving back and forth (as described above). When this option is removed by limiting turn angle (disallowing moving backward), robots reach a local optimum: moving forward moves all targets out of range (an undesirable result), and moving laterally or backward is not permitted. This local optimum can be averted by redesigning the value function to bias points that are close to targets, even if they will not be observable from the location. This allows robots to move past targets and gradually turn until they can be brought back into view.

Trajectories demonstrating the effects of these robot limitations in the larger environment are shown in Figure 46 and Figure 47.

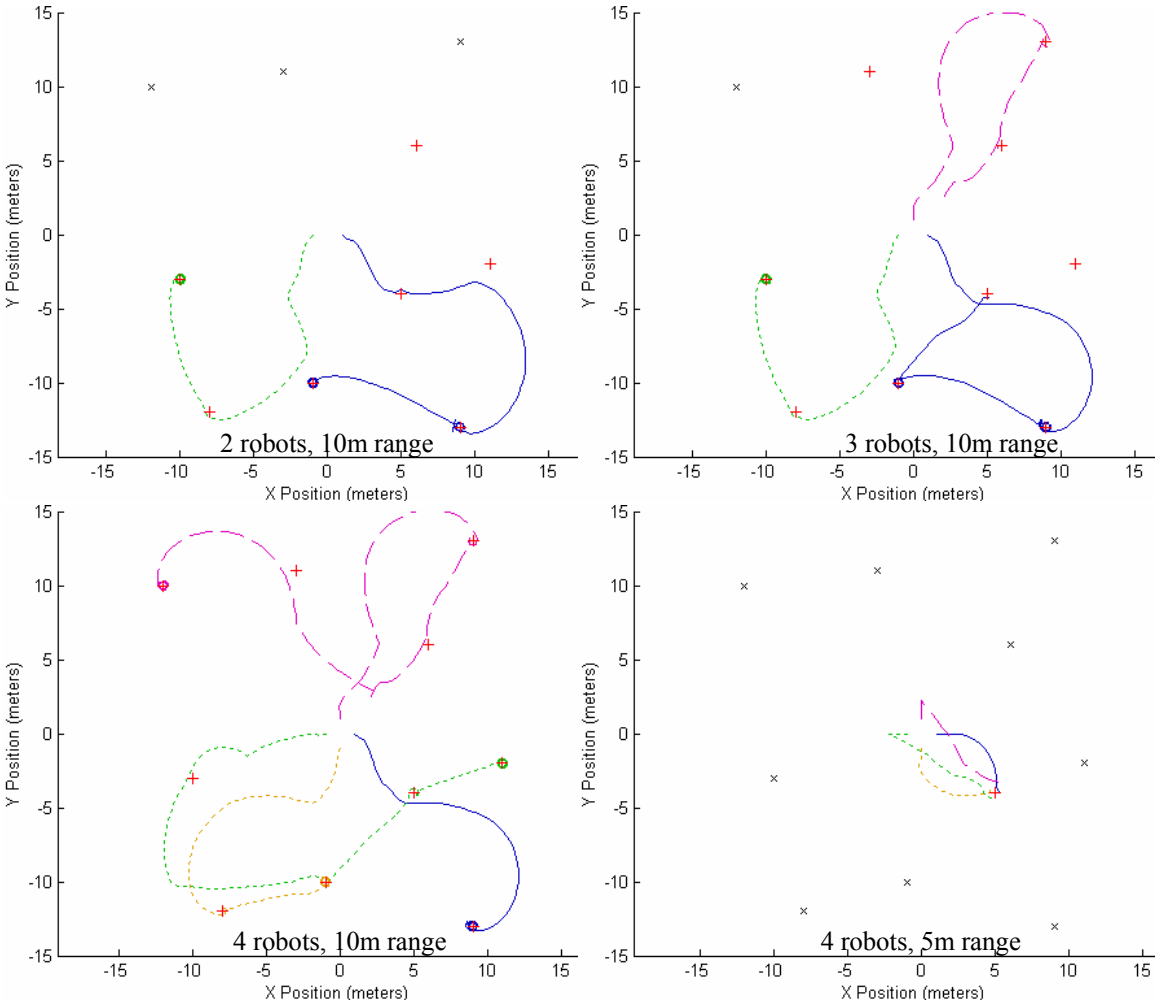

Figure 46. Experiments M and N, T4: Limited Vision Range in Large-Scale Target Location
Unseen targets are shown as black x's. *Top, Bottom left:* Range 10 meters. *Bottom right:* Range 5 meters. Limiting field of view in this case allows robots to see only a small fraction of the space. To maximize value, the uncertainty must be decreased on as many targets as possible, thus robots spread out. If, while approaching known targets, new targets are discovered, robots will attempt to maximize the value of these new targets. Unknown targets not observed in the course of maximizing uncertainty on known targets remain undiscovered.

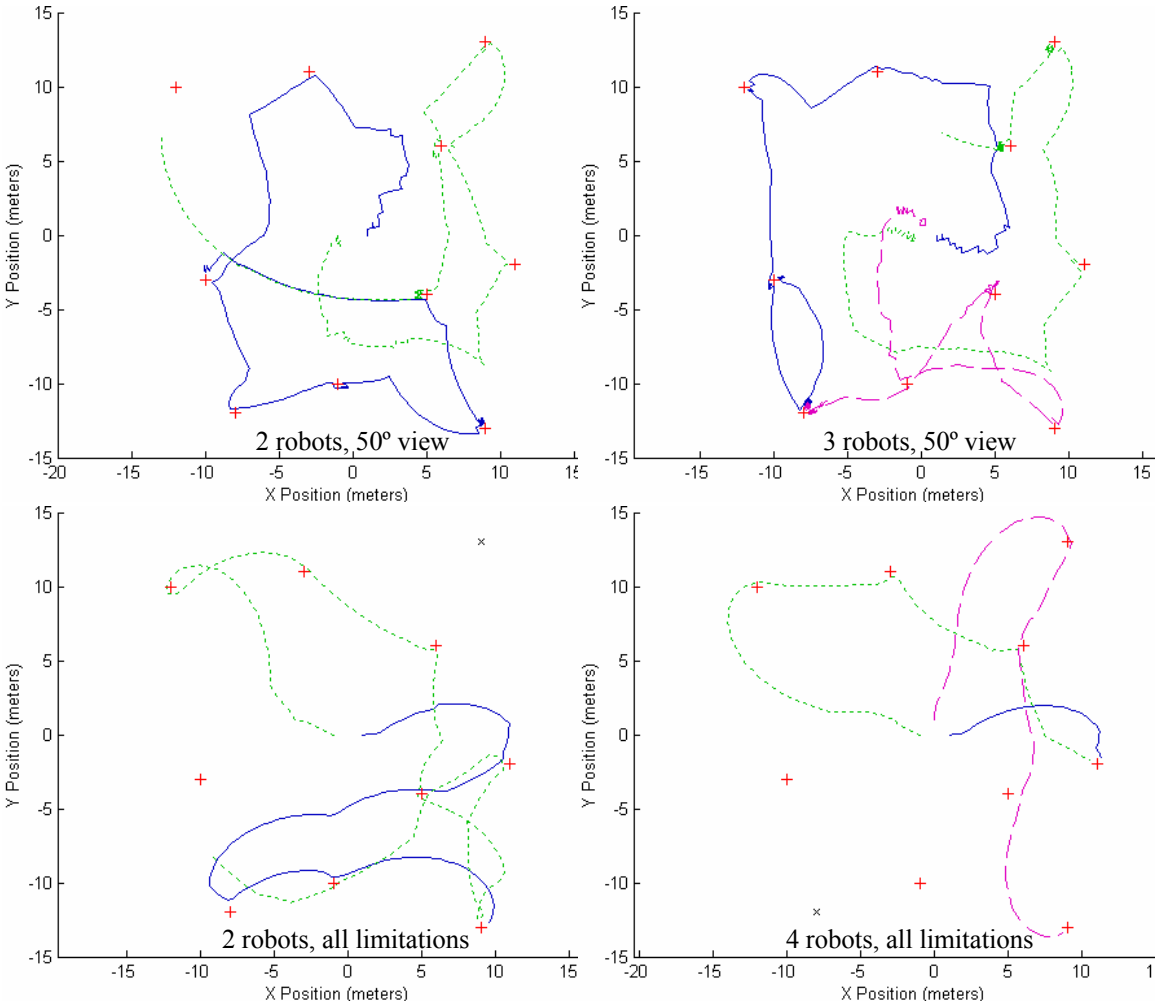

Figure 47. Experiments O, P, and Q, T4: Robot Limitations in Large-Scale Target Location Unobserved targets are black x's. *Top:* Angular field 50º *Bottom:* All limitations (range 10 m). Limited field of view may prevent robots from finding all targets. If forced to explore more (with a larger turning radius), robots increase number of targets found.

Results for the large environment are shown in Table 8 and Table 9. As before, the largest effect comes from applying all limitations together. Bearing has a stronger effect when team size increases, as robots can share fewer overlapping observations. As in previous cases, ratios below 1 occur when fewer targets are observed (and therefore fewer contribute to value) by the limited robots than by the unlimited robots.

Table 8. Robot Limitations in Large Environment Target Location: Value (E_2)

Robots	E_2 Ratio at Step 60			
	Range = 10.0m	Bearing = 50º	Turn = 50º	All
2	0.49	1.49	1.00	4.05
3	0.95	3.27	1.05	2.70
4	1.22	2.21	0.89	3.78
6	8.89	2.93	0.74	124
8	31.5	4.25	1.30	8.21

Table 9. Robot Limitations in Large Environment Target Location: Time to Map Quality (E_4)

Robots	E_4 Ratio: Steps to E_4<0.01			
	Range = 10.0m	Bearing = 50º	Turn = 50º	All
2	15.7	15.7	9.9	15.7
3	17.9	10.9	9.0	17.9
4	17.5	7.3	10.7	13.9
6	10.4	6.5	5.4	25.1
8	11.6	6.6	4.7	7.7

A similar pattern emerges in looking at the case of the large environment (Figure 46). As robots begin to explore, some targets are observed. Observation by all the robots concentrates on these targets. In the process of moving to observe these targets, more targets may become known. Limiting range to 10 meters (approximately the same as the average inter-target spacing) allows the robots to explore, but teams do not find all the targets until the team size reaches four robots (Figure 46, bottom left). When the range is limited further to 5.0 meters, robots quickly converge on the first target found, but the short vision range does not allow them to observe other targets; thus, robots become stuck on the first target and do not fully explore the space (Figure 46 bottom right). Adding more robots to the system does not improve the situation, as the effective coverage from the starting location is not largely increased by adding robots to the same location. This is observed when the vision range is much less than the average spacing between targets. The effect could be circumvented for forcing robots to explore, even when targets are know. The addition of an explicit additional value function for exploration is discussed in 5.3.4 as part of the more complex Planetary Exploration Mission.

Other limitations also produce similar results to those in the medium environment (Figure 47). Robots move back and forth in an area to compensate for a limited angular field of view. Limiting turning angle causes robots to circle targets at larger ranges, but otherwise trajectories appear to be identical to those shown in. Combining all the limitations (range, angular field of view, and turning angle) produces trajectories similar in quality to those in the medium environment case, as illustrated by Figure 47. However, one beneficial side effect is observed. The limitation on the turning angle does not allow robots to circle back toward the first known target as quickly, potentially allowing them to explore more area, as in the case of two robots (Figure 47, top right). Here, the limitation forces robots to turn around more slowly at each target, allowing them to eventually observe all but one of the targets.

As robots explore, the team maximizes the value of known targets, causing them to converge on these known targets. Additional targets are only considered if they are observed in the course of this exploration. As team size increases, the spreading to maximize value on the known targets may allow robots with a limited field of view to explore a greater area and find more targets. Limited turning angle may similarly improve exploration with limited range relative to limited range without limited turning angle. By turning more slowly robots may travel further and observe more targets. Increasing team size only improves performance to the extent that it improves the value on known targets by adding more measurements to the map and may increase the explored area as robots spread out. This increase in explored area allows the robots to find more targets if the intra-target distance is within the sensor range.

7.6 Comparison to Individual Action Selection (T5)

Individual action selection forces each robot to make decisions on how to improve map quality based only on its own observations. The only interaction between teammates is that of collision avoidance. By comparing this individual action selection to MVERT's inclusion of teammate contributions, the hypothesized performance improvement generated by making *team* decisions instead of individual decisions can be tested, and question 2 (regarding comparison to individual action selection) is addressed. To reach equilibrium (completion) requires 40 steps in the medium environment and 250 in the large environment. Data for comparison is taken from Experiments B and C (T1). Three experiments are performed for series T5:

R. 2, 3, 4, and 6 robots observing in the medium environment,

S. 2, 3, 4, 6, 8, and 12 robots observing in the large environment.

Trajectories for maximizing value for individual action selection are shown in Figure 48 and Figure 49.

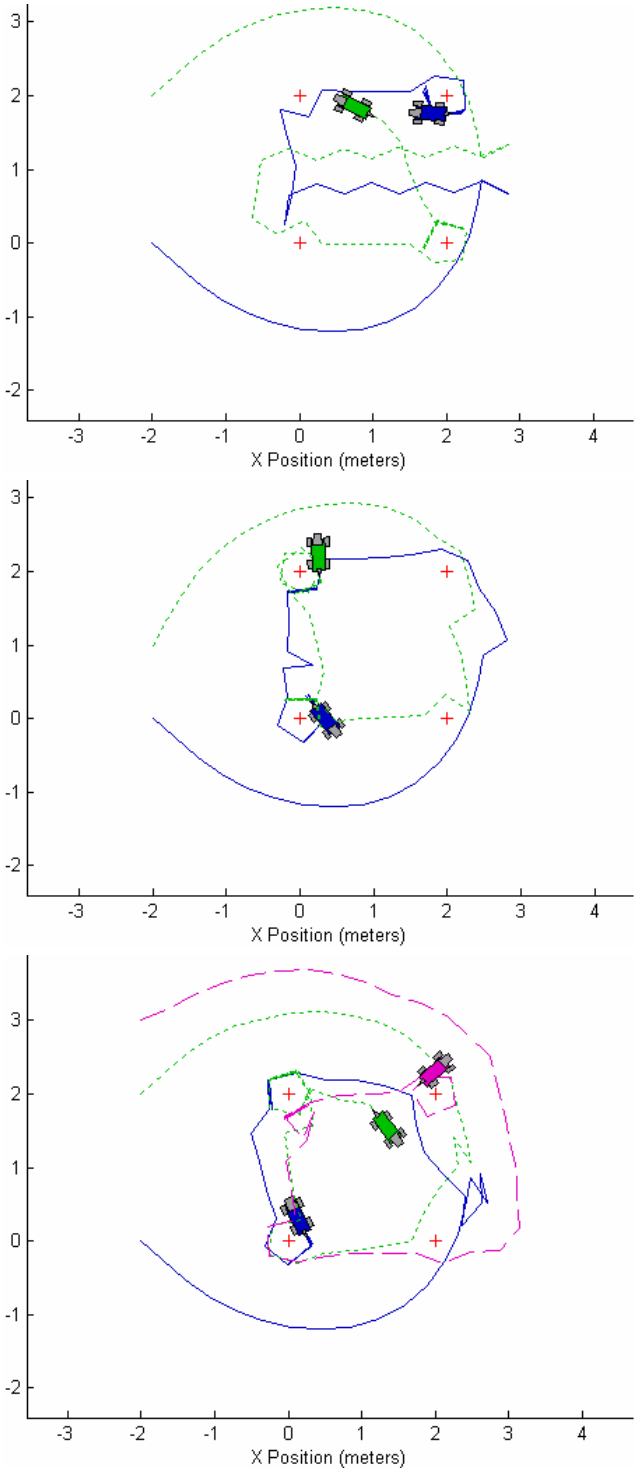

Figure 48. Experiment R, T5: Individual Action Selection in Medium Environment Target Location To maximize individual value, robots must take their own complementary measurements. This results in robots arcing more, driving much further around targets, and a slower reduction of uncertainty. *Top:* Symmetrical two-robot configuration. *Middle, Lower:* Asymmetrical two- and three-robot configuration.

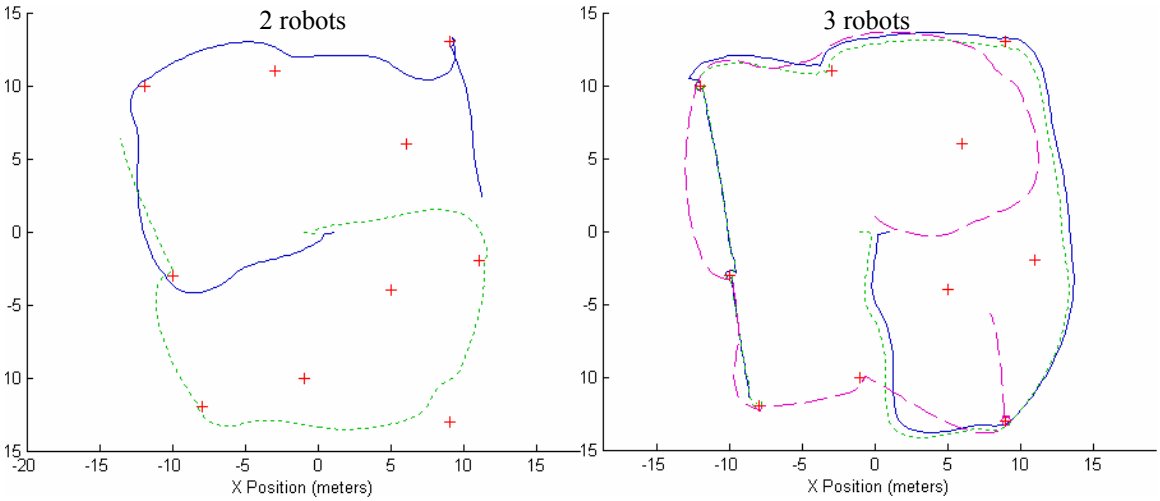

Figure 49. Experiment S, T5: Individual Action Selection in Large Environment Target Location
As in the medium environment, robots follow similar paths, maximizing value in a similar manner.

Value at any step is higher using MVERT. Maps of a desired quality are made in fewer steps using MVERT. Table 10 and Table 11 show value and time required to obtain a given maximum uncertainty. Dashes indicate the desired uncertainty was not reached. Ratios over 1 indicate better MVERT results.

Table 10. Individual versus Team Action Selection: Medium Environment Target Location

Robots	E_2 at Step 12	E_2 Ratio at Step 12	E_4 Ratio: Steps to E_3<0.01	E_4 Ratio: Steps to E_3<0.005	E_4 Ratio: Steps to E_3<0.002	E_4 Ratio: Steps to E_3<0.001
2	-2.02×10^{-4}	2.64	1.00	1.09	1.29	--
2	-2.14×10^{-4}	2.33	1.00	1.08	0.90	--
2	-2.72×10^{-4}	2.37	1.57	1.75	0.92	--
3	-1.23×10^{-4}	1.60	1.13	1.20	1.00	--
3	-1.59×10^{-4}	3.15	1.13	1.20	1.04	--
4	-1.25×10^{-4}	3.42	1.00	1.33	1.59	1.62
4	-1.08×10^{-4}	2.93	1.14	1.20	2.42	--
6	-1.01×10^{-4}	3.40	1.00	1.22	2.45	1.63
6	-0.778×10^{-4}	2.61	1.00	1.22	2.08	1.61

Table 11. Individual versus Team Action Selection: Large Environment Target Location

Robots	E_2 at Step 60	E_2 Ratio at Step 60	E_4 Ratio: Steps to E_3<0.01	E_4 Ratio: Steps to E_3<0.005	E_4 Ratio: Steps to E_3<0.002
2	-7.12×10^{-3}	1.22	1.94	1.18	--
3	-4.91×10^{-3}	2.58	1.86	1.17	--
4	-3.95×10^{-3}	3.17	1.92	1.31	--
6	-2.80×10^{-3}	7.57	2.10	2.39	1.44
8	-1.84×10^{-3}	17.65	1.78	3.45	3.63

MVERT's performance relative to individual action selection is illustrated in Figure 50 to Figure 56.

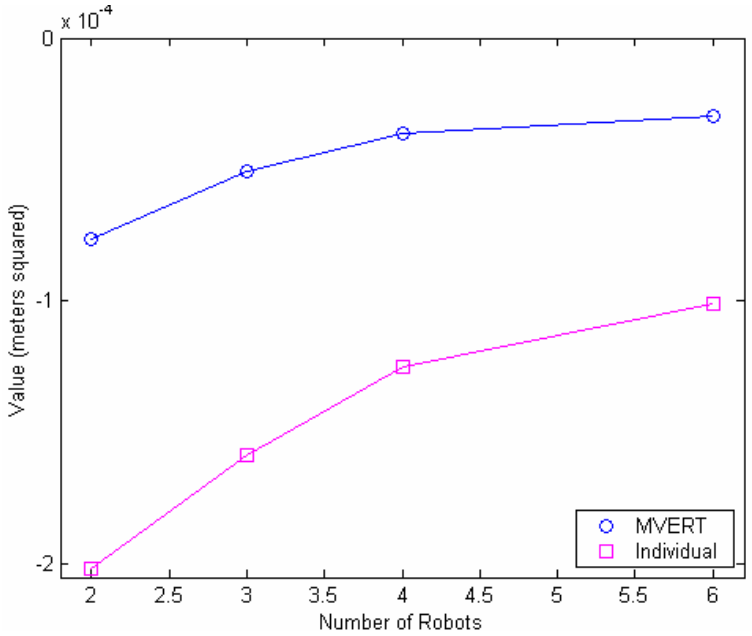

Figure 50. Value (E_2) Versus Team Size: Medium Environment Target Location
Absolute value increases (improves) with team size. Also, relative value increases with team size.

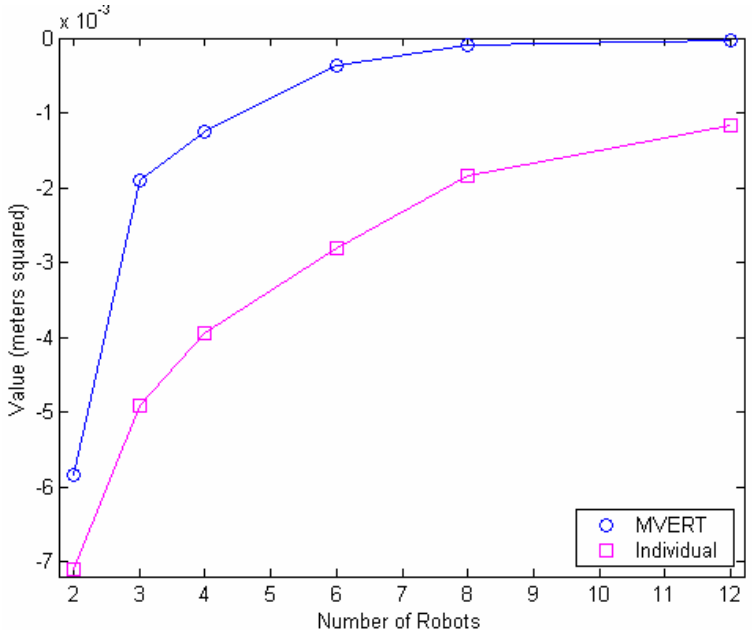

Figure 51. Value (E_2) Versus Team Size: Large Environment Target Location
As in the medium environment, absolute and relative values increase (improve) with team size.

Medium environment data are using the symmetrical configuration. Figure 50 and Figure 51 address question 4, regarding impact of team size, by looking at total value (for one configuration) versus team size for MVERT and individual action selection. They illustrate that while value continues to improve with each additional team member, the amount of improvement begins to drop off as the team size further increases. This indicates that there is a team size above which the amount of improvement does not greatly affect performance. This limit is close to five robots in the medium environment and nine in the larger, and is clearly a function of the size of the environment and the number of targets in the environment.

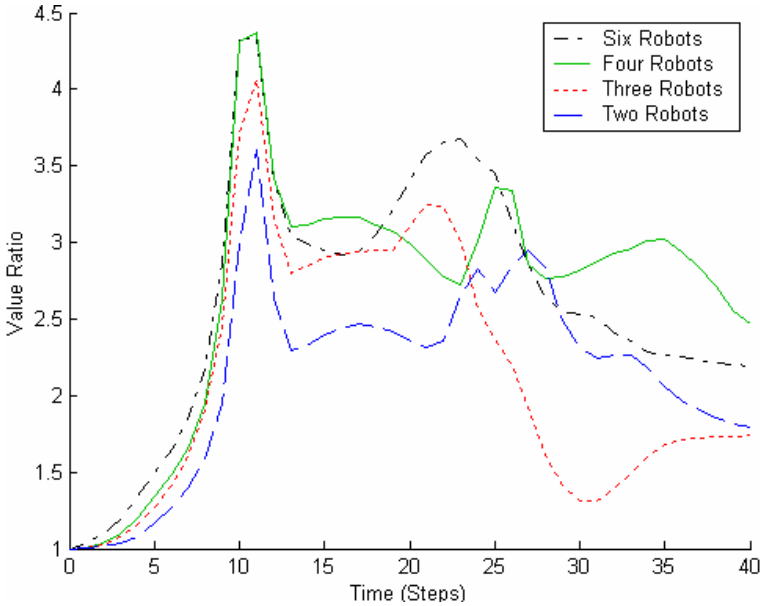

Figure 52. Value (E_2) Ratio Versus Time: Small Environment Target Location
The higher peaks for curves for larger team size indicate better relative value over time for MVERT compared to individual action selection. The largest relative benefit is gained early, when few measurements have been taken.

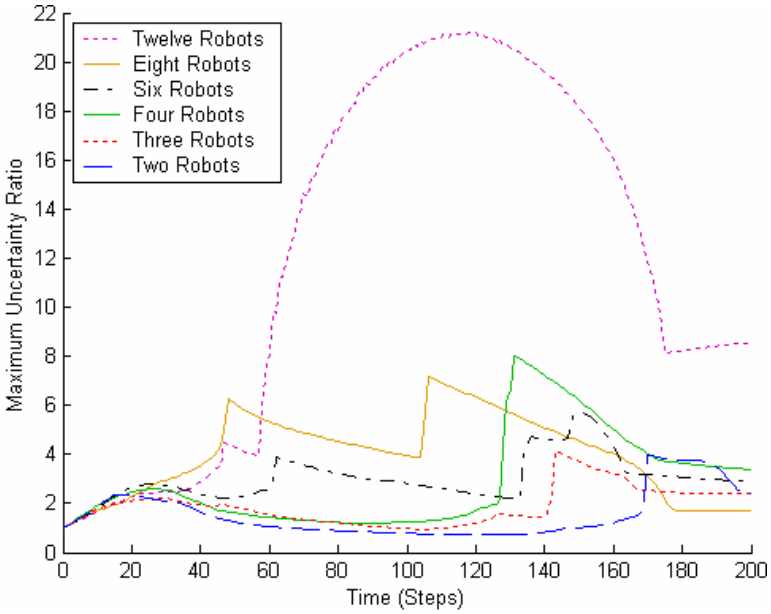

Figure 53. Value (E_2) Ratio Versus Time: Large Environment Target Location
Relative performance increases (improves) as team size increases, as in the case of the medium environment. Initial peaks occur early, after few measurements.

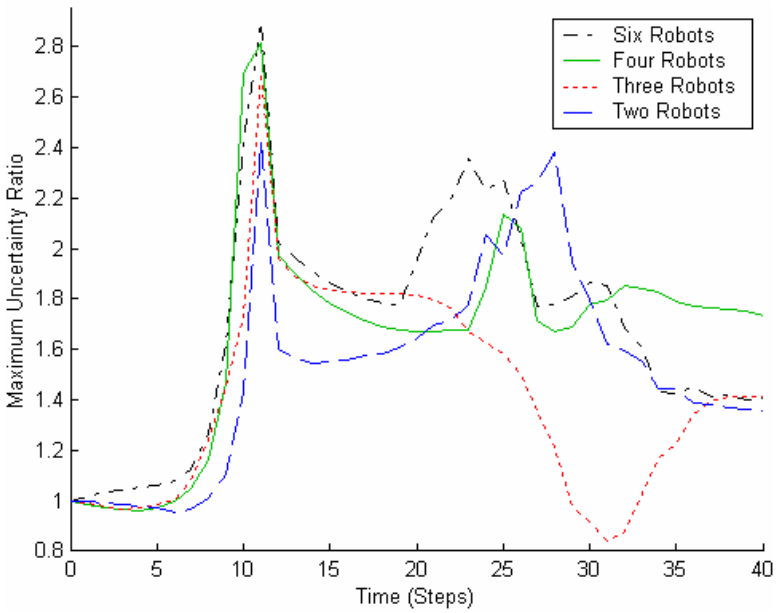

Figure 54. Maximum Uncertainty (E_3) Ratio Versus Time: Small Environment Target Location
As with value, relative performance in maximum uncertainty increases (improves) with team size, with the largest benefit occurring early in the task..

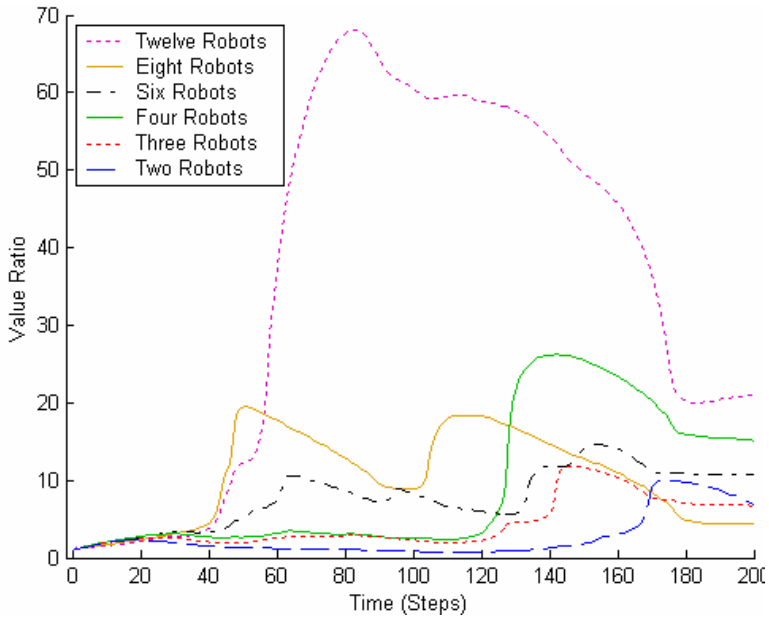

Figure 55. Maximum Uncertainty (E_3) Ratio Versus Time: Large Environment Target Location
Again, relative performance improves (larger ratios of uncertainty area) with team size.

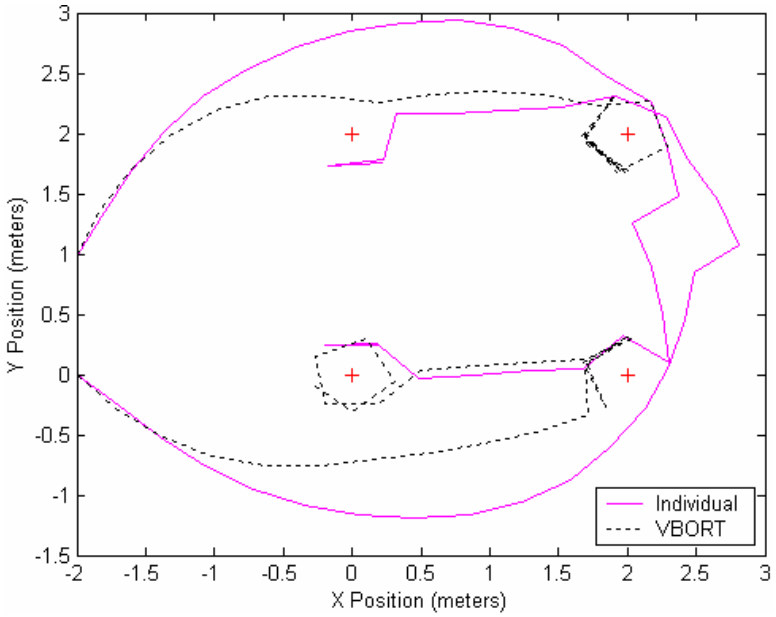

Figure 56. Target Location: MVERT Versus Individual Action Selection (Medium)
Individual robots must go much further out of their way to obtain the same complementary observations. MVERT allows robots to remain closer to targets, obtaining higher quality measurements and multiple points of view. Example shown for the asymmetrical configuration.

MVERT, including teammate contributions in action selection, always produces final maps of higher value (E_2) than those maps generated without including teammate contributions. Typically, the relative improvement increases with team size, as each improvement in observation point further improves the estimates. In the extreme case of twelve robots in the large environment, the very dramatic improvement is due to the fact that robots using individual action selection investigate the same part of the environment first, only producing good estimates in that area, where robots using MVERT cover the space quickly. As the robots eventually turn to the other part of the environment and reduce uncertainty there, the relative performance of MVERT drops but remains better than individual action selection. In contrast, for more than two robots, robots distribute more initially (partly to maximize value and partly to avoid collisions).

In almost all cases, despite optimizing for value, MVERT produces lower maximum uncertainty than when actions do not consider teammate contributions. The value ratio of individual to MVERT eventually drops below 1 for a time in the two-robot, 10-target case. This is due to the fact that the two robots reach local optima and circle targets many steps before the individual robots, which continue to reduce uncertainty on multiple targets. Occasionally, the maximum uncertainty of the individual robots is lower than that from MVERT. In the individual approach, the robots must reduce area only with a single measurement. This is typically accomplished by lowering the maximum uncertainty. In contrast, MVERT reduces area with multiple measurements. The result of the joint measurement may significantly reduce area in targets without significantly reducing the single maximum uncertainty. This occurs when the maximum uncertainty is at a larger distance, and the area reduction of closer targets dominates the value function.

The qualitative differences between including and disregarding teammate contributions are most visible in Figure 56, which shows that individual robots must go much further out of their way to produce the same complementary measurements, and thus do not remain as close to targets.

7.7 Comparison to One-Step Optimal (T6)

The implementation of a one-step optimal maximizes the same value function used by the MVERT approach. Instead of approximating expected teammate contributions, the one-step optimal simultaneously optimizes over the *set* of moves the team may make in order to select the best one. Comparison with this approach investigates the ability of MVERT to produce good results (similar to optimal) while also reducing the computational complexity to a manageable, scalable level. This addresses question 3,

comparing to one-step optimal. Due to the high level of computation required for the one-step optimal, only experiments with two robots are run using the optimal. The detailed experiments are:

T. 2 robots observing in the medium environment;

U. 2 robots observing in the large environment.

Example trajectories, compared to those produced by MVERT, are shown in Figure 57 and Figure 58.

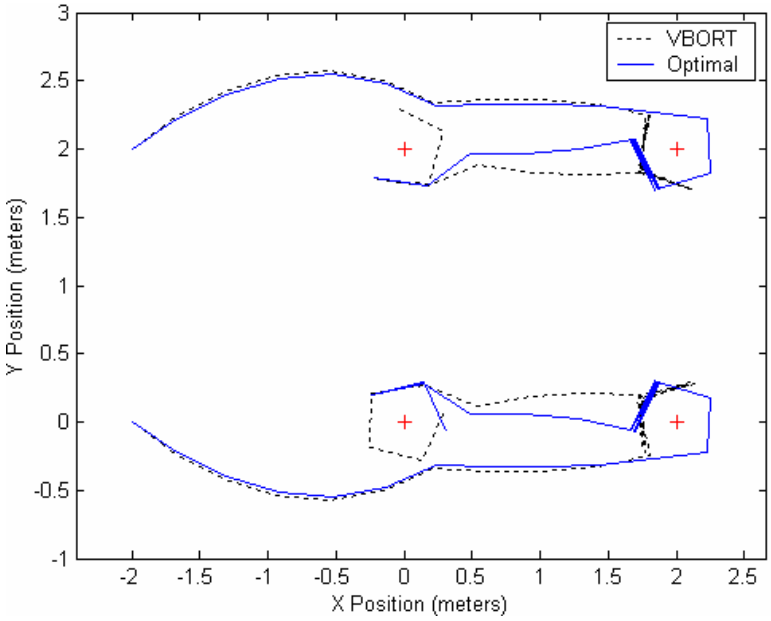

Figure 57. Experiment T, T6: MVERT Versus One-Step Optimal, Medium Target Location
MVERT closely approximates one-step optimal trajectories. MVERT and one-step optimal circle targets differently, but for a similar number of steps. Example shown for the symmetrical configuration.

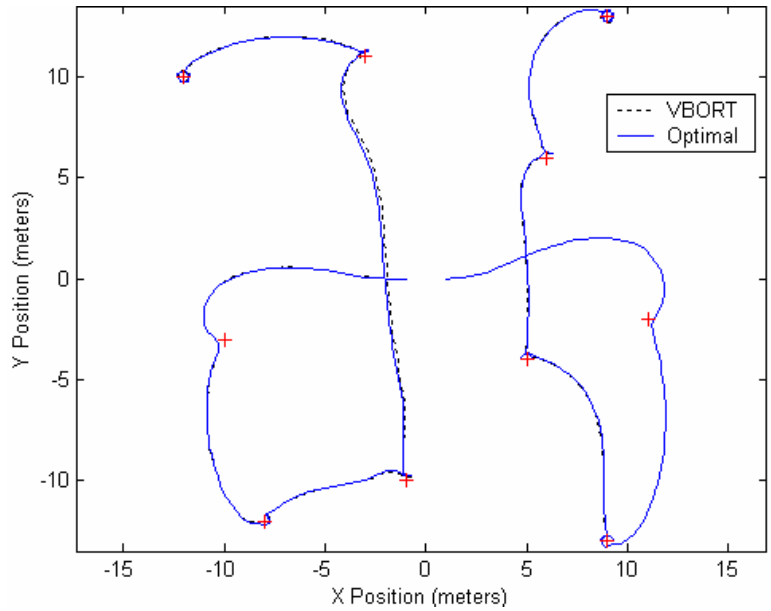

Figure 58. Experiment U, T6: MVERT Versus One-Step Optimal, Large Target Location
Two-robot comparison to one-step optimal. In the larger environment, MVERT trajectories are nearly indistinguishable from one-step optimal.

A comparison of the resulting performance for MVERT and the one-step team optimal are provided in Table 12. These results include the number of steps to a desired map quality (maximum uncertainty), value, and the computation time. Computation time reflects the time for one robot to compute one action, averaged over ten trials. Complexity is the number of Gaussian distribution multiplications that must be performed for one robot to select one action, where m is the number of candidate moves, R is the number of robots on the team and T is the number of mapping targets observed by each robot at each step. MVERT performs almost identically to the one-step optimal in both value and time to a given maximum error. Both approaches perform much better (2.6 times in the medium environment, 1.2 times in the large) than individual action selection, where teammate contributions are not considered. Despite the similarity in quantitative and qualitative performance, MVERT provides a drastic reduction in complexity and time of a factor of 350.

Table 12. Two-Robot Target Location: One-Step Optimal, MVERT, and Individual Action Selection

		One-Step Optimal	MVERT	Individual
Medium Environment	Value at Step 14	$-7.55\text{x}10^{-5}$	$-7.66\text{x}10^{-5}$	$-20.2\text{x}10^{-5}$
	E_4: Steps to E_3<0.001	9	9	9
	E_4: Steps to E_3<0.005	11	11	12
	E_4: Steps to E_3<0.002	27	24	31
	Computation Time	225.9 ± 0.3 s	0.645 ± 0.005 s	0.645 ± 0.008 s
Large Environment	Value at Step 60	$-6.05\text{x}10^{-3}$	$-6.06\text{x}10^{-3}$	$-7.47\text{x}10^{-3}$
	E_4: Steps to E_3<0.1	16	16	31
	E_4: Steps to E_3<0.01	165	165	194
	E_4: Steps to E_3<0.005	170	170	--
	Computation Time	529.6 ± 0.7 s	1.498 ± 0.005 s	0.673 ± 0.011 s
Complexity		$m^R T$	(R-2)T+ mT	mT

In comparison to the one-step team optimal, MVERT causes robots to spread slightly more, since simultaneous teammate movement is not considered. However, trajectories and values are quite similar (as illustrated by Figure 57 and Figure 58). MVERT and the one-step team optimal produce values nearly identical, while almost twice those produced by individual action selection.

7.8 Summary and Discussion

In this chapter, the trajectories produced by MVERT for locating targets in the absence of sensing and motion noise are described. MVERT takes advantage of teammates when selecting actions, providing triangulation and multiple points of view on targets when needed to minimize uncertainty and maximize value. MVERT automatically optimizes action selection for different sensing and motion capabilities by applying models to predict results of actions. MVERT provides a sliding scale of computational complexity through adjusting the number of candidate moves each robot considers; the resulting trajectories make sharper turns as the choices are limited, but overall performance drops only slightly.

In comparison to individual action selection, MVERT reduces uncertainty in target estimates more quickly, and this improvement increases with team size, as more measurements are simultaneously optimized. By distributing robots to produce complementary measurements, MVERT provides a higher level of coverage and faster mission completion.

In comparison to one-step team optimal, MVERT produces trajectories that are quantitatively and qualitatively similar while reducing computation time significantly. MVERT scales linearly with team size, while one-step optimal scales exponentially.

Chapter 8 Mapping Mission

The experiments reported in this chapter are intended to investigate the following questions:

6. Can MVERT be applied to mapping an unknown real environment?
7. Are maps produced by MVERT of higher quality than those made using other approaches?
8. How do motion and sensing noise affect MVERT performance?

8.1 Experimental Summary

In the Mapping Mission, the environment is completely unknown to the robots. Robots only know the initial poses of all teammates prior to the experiment. The purpose of experiments in this task is to evaluate the integration of MVERT with a practical mission task and to investigate the ability of MVERT to improve the quality or efficiency of the mapping process. Target location value is used to selection actions (Eq. 2, Section 3.3.1). Performance metrics from Section 5.1 are map error (E_1), map certainty (E_2 and E_3), mission time (E_4), and robot localization uncertainty (E_5). The experimental series are:

M1. MVERT Small-Scale Simulation Mapping Performance;
M2. MVERT Small-Scale Physical Robot Mapping Performance;
M3. MVERT Large-Scale Simulation Mapping;
M4. Comparison to Mapping with Coverage Patterns.

Experiments are run in four environments with sensing and motion noise:

- small mapping simulation : 6 targets and up to 8 robots in a 5 by 5 meter area,
- middle clustered mapping simulation: 100 targets and up to 25 robots in a 80 by 80 meter area,
- large uniform mapping simulation: 30 targets and up to 25 robots in a 500 by 500 meter area,
- physical: 6 targets and up to 4 robots in a 5 by 3 meter area.

8.2 MVERT Small-Scale Mapping Simulation (M1)

Simulation experiments for the Mapping Mission are designed to illustrate how MVERT may be applied to mapping unknown environments and comparing performance, questions 6 and 8 above. These experiments are also designed to investigate the impact of using predictions of teammate contributions during action selection on the quality and efficiency of map generation, question 7. The M1 simulation experiments are:

A. 2, 3, 4, 6, and 8 robots mapping the small environment;
B. Experiment series A using individual action selection.

Ten configurations of landmarks are mapped by the robot team (Figure 59), two runs per configuration.

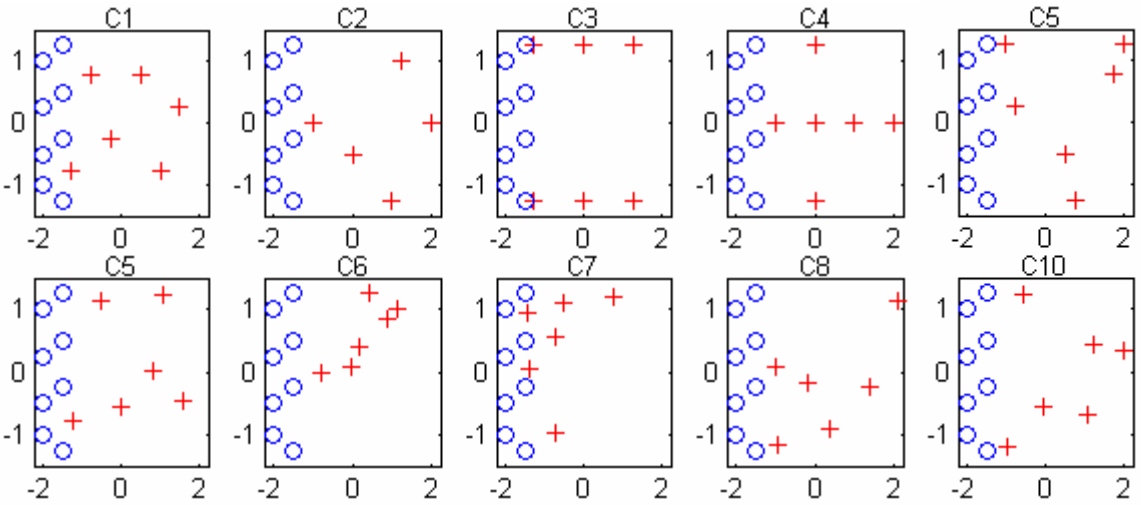

Figure 59. Simulation Small-Scale Mapping Environment Configurations
Red +'s are target locations. Blue circles are robot starting locations. C5-C10 were generated randomly.

In these experiments, the sensor characteristics of the Aibo are used to generate motion and sensor noise as well as to model pose uncertainty and measurement uncertainty (5.4). Experiments are run for 16 steps, allowing robots to explore the space. The small environment and small physical size of the robots allows use of a small step size. A step of 0.254 meters (10 inches) is used in these experiments. Example trajectories are shown for the C1 configuration in Figure 60.

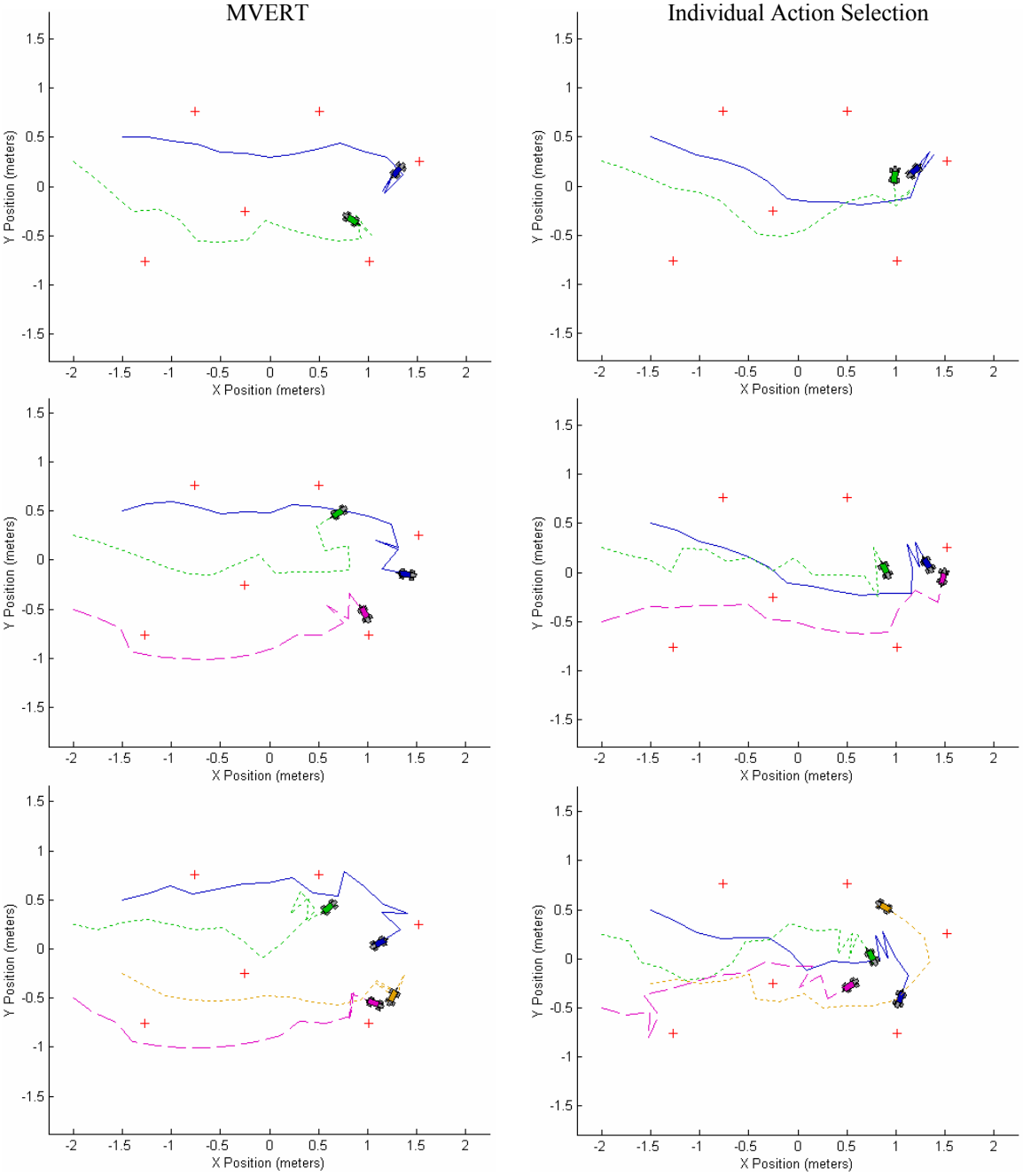

Figure 60. Experiments A and B, M1: Sample Mapping Trajectories
Two, three, and four robots map the environment. *Left:* Trajectories produced by MVERT. *Right:* Trajectories produced using individual action selection. MVERT, by maximizing joint value, causes robots to distribute themselves spatially. Robots using individual action selection tend to follow similar trajectories, because they are maximizing value from similar locations, and interfere with each other due to closer clustering.

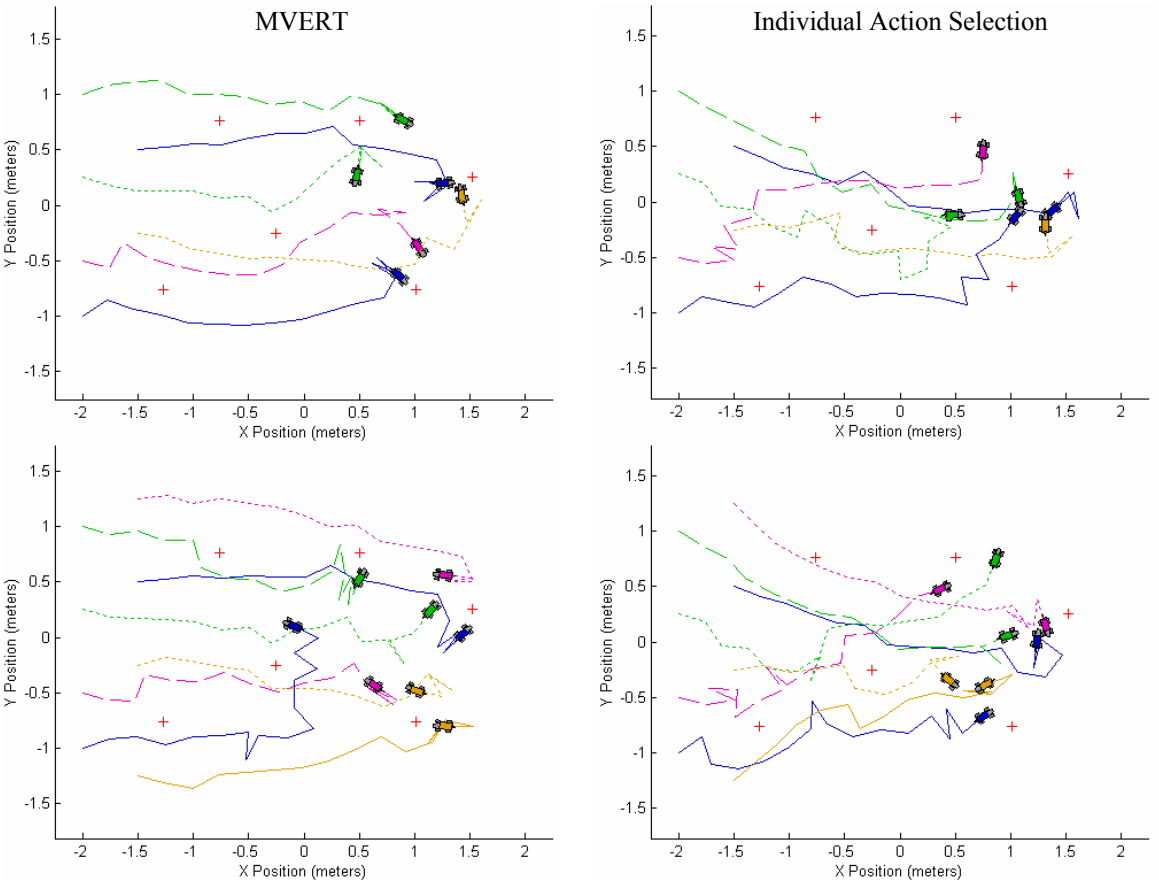

Figure 60 (Cont). Experiments A and B, M1: Sample Mapping Trajectories
Six and eight robots map the environment. *Left:* Trajectories produced by MVERT. *Right:* Trajectories using individual action selection. MVERT reduces interference, even with larger team size, by spreading robots more and distributing them in the environment.

MVERT performance compared to the individual approach is shown in the following figures and tables. The 95% percent confidence interval is shown in parentheses. Ratios are the metric for individual action selection divided by the metric for MVERT. At step 12 (prior to robots moving past the last targets), individual action selection produces a total uncertainty area 1.36 times that generated by MVERT, as shown in Table 13. The improvement in performance is statistically significant.

Table 13. Mean E_2 (Value): Small-Scale Mapping Simulation, Step 12

Robots	2	3	4	6	8
MVERT E_2	-0.00919 m^2 (±0.0013 m^2)	-0.00609 m^2 (±0.0007 m^2)	-0.00416 m^2 (±0.0004 m^2)	-0.00290 m^2 (±0.0003 m^2)	-0.00206 m^2 (±0.0002 m^2)
Individual E_2	-0.01196 m^2 (±0.0024 m^2)	-0.00877 m^2 (±0.0016 m^2)	-0.00575 m^2 (±0.0016 m^2)	-0.00401 m^2 (±0.0006 m^2)	-0.00294 m^2 (±0.0004 m^2)
Ratio E_2	1.30 (±0.10)	1.40 (±0.15)	1.36 (±0.12)	1.37 (±0.10)	1.40 (±0.12)

Table 14 shows that in every case, the average performance of MVERT reaches a fixed uncertainty level in fewer steps than individual action selection. The performance is relatively constant across team size in this small environment. MVERT improves its relative performance slightly as the desired maximum uncertainty decreases. There is a slight trend for greater improvement with larger team size.

Table 14. Time to Desired Maximum Uncertainty (E_4): Small-Scale Mapping Simulation

Robots		2	3	4	6	8
E_4 : Mean Steps to $E_3 < 0.1$m	MVERT	6.3	5.7	4.9	4.6	4.1
	Individual	6.4	6.0	5.1	4.7	4.4
	Ratio	1.02	1.05	1.04	1.02	1.07
E_4 : Mean Steps to $E_3 <$ 0.05m	MVERT	9.9	8.6	7.4	6.8	5.9
	Individual	10.6	9.7	8.2	7.2	6.6
	Ratio	1.07	1.12	1.11	1.06	1.12
E_4 : Mean Steps to $E_3 <$ 0.02m	MVERT	15.9	13.8	11.7	10.7	9.4
	Individual	16.5	15.2	13.4	11.7	10.5
	Ratio	1.04	1.10	1.15	1.09	1.18

In every case, the final map produced by the robot team is of higher quality using MVERT than using individual action selection. This is illustrated by Table 15, which shows the map error at step 16, the end of the trial. The map error relative to individual action selection improves as the team size increases, as the MVERT approach maximizes joint value directly. The 95% confidence limits are shown in parentheses. Confidence intervals indicate that MVERT performance improvement is statistically significant for teams larger than two. For two robots in a short mission, there are fewer measurements to maximize, slowing divergence of performance from individual action selection. Performance improves with larger teams. Thus, not only can MVERT be successfully applied to mapping unknown environments (question 6), but it does improve the map quality relative to individual action selection (question 7).

Table 15. Mean Final Map Landmark Error (E_1): Small-Scale Mapping Simulation

Robots	2	3	4	6	8
E_1 MVERT	0.0291 m (±0.0056 m)	0.0264 m (±0.0072 m)	0.0233 m (±0.0063 m)	0.0224 m (±0.0054 m)	0.0165 m (±0.0037 m)
E_1 Individual	0.0296 m (±0.084 m)	0.0282 m (±0.0091 m)	0.0280 m (±0.0067 m)	0.0286 m (±0.0069 m)	0.0237 m (±0.0069 m)
E_1 Ratio	1.02 (±0.09)	1.07 (±0.13)	1.20 (±0.07)	1.28 (±0.07)	1.44 (±0.06)

The maximum localization error in each robot can be compared similarly. The localization of robots using MVERT is better because the landmarks used to localize have been localized themselves by multiple robots and more total observations. Individual action selection does not improve localization as team size increases. Occasional high errors may occur due to a single bad landmark; the multi-robot SLAM approach prevents single measurements from producing bad landmarks and leading to poor localization. This is demonstrated by Table 16; the first number indicates pose error (distance from actual location), and the second indicates heading error magnitude.

Table 16. Maximum Final Localization Error (E_5): Small-Scale Mapping Simulation

Robots	2	3	4	6	8
E_5 MVERT	0.0524 m 1.41º	0.0488 m 2.01º	0.0406 m 1.40º	0.0425 m 0.72º	0.0252 m 0.77º
E_5 Individual	0.0531 m 4.74º	0.0540 m 2.95º	0.0638 m 3.90º	0.0549 m 30.28º	0.0461 m 2.25º
E_5 Ratio Individual to MVERT	1.01 3.34	1.11 1.45	1.57 2.79	1.29 41.83	1.83 2.93

Examples of MVERT performance compared to individual action selection during a series of experiments (those in Figure 60) are shown in Figure 61 - Figure 64. The high level of noise in the system often makes predicted results differ from expected. In most cases, occasional higher value of individual action selection is due to the fact that MVERT has observed more targets, adding more terms to the total uncertainty area.

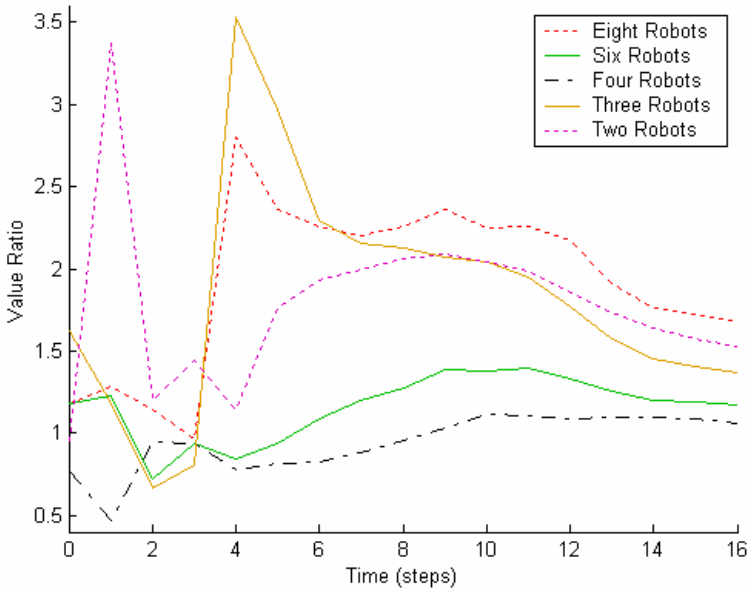

Figure 61. Value (E_2) Ratio Over Time: Small-Scale Mapping Simulation
MVERT quickly does much better than individual action selection. Numbers greater than 1.0 indicate better MVERT performance. Dips in value ratio below 1 result from MVERT observing more targets, thus adding more uncertainty area to the total value.

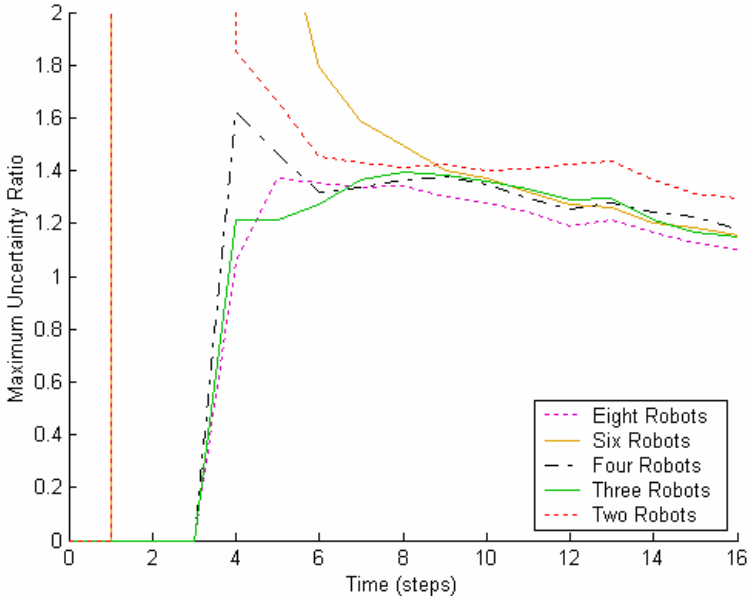

Figure 62. Maximum Uncertainty (E_3) Ratio Over Time: Small-Scale Mapping Simulation
Numbers greater than 1.0 indicate better MVERT performance. MVERT quickly does much better than individual action selection. The spikes in six and eight robots are due to MVERT observing all the targets before individual action selection does, leaving it with an infinite maximum uncertainty.

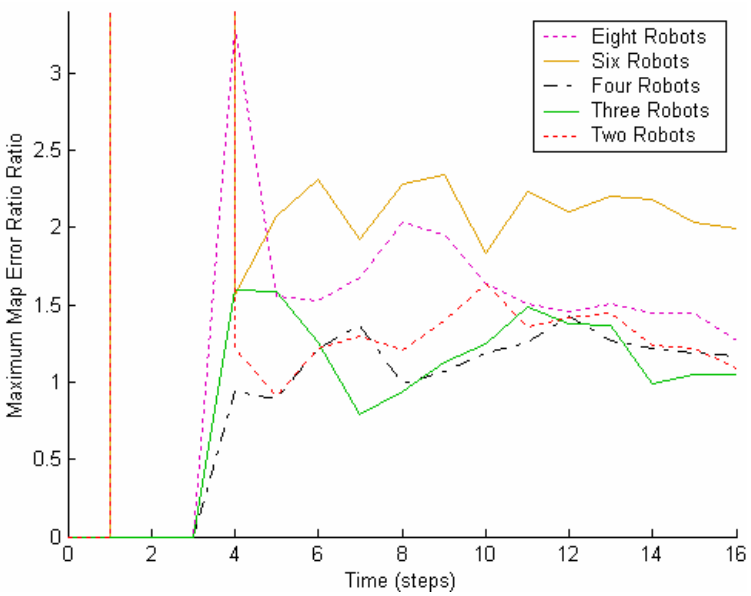

Figure 63. Maximum Map Error (E_1) Ratio Over Time: Small-Scale Mapping Simulation
Numbers greater than 1.0 indicate better MVERT performance. MVERT quickly does much better than the single-robot approach. The spikes in six and eight robots are due to MVERT observing all targets before individual action selection does, leaving individual action selection with an infinite maximum error.

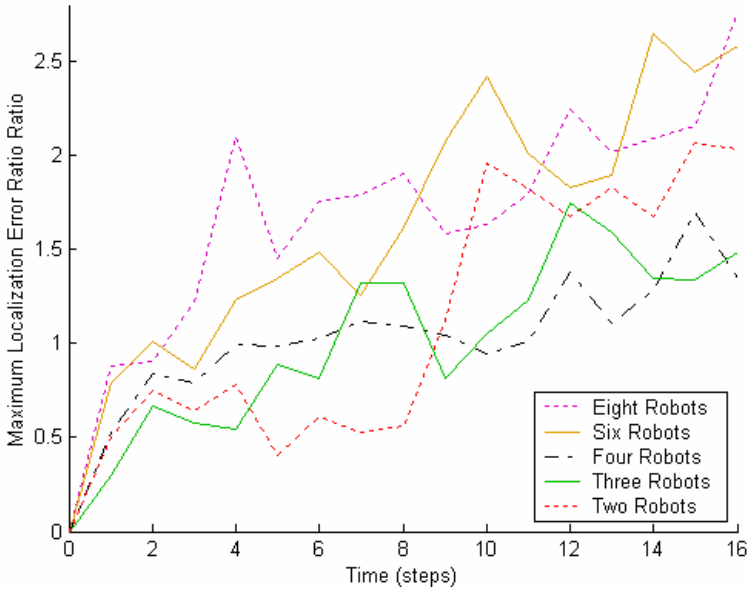

Figure 64. Maximum Localization Error (E_5) Ratio Over Time: Small-Scale Mapping Simulation
Numbers greater than 1.0 indicate better MVERT performance. Over time, the relative localization of MVERT improves, as independent robots rely only on their own map, which diverges due to odometry errors over time. MVERT, using a joint map, remains converged or diverges much more slowly.

It is evident that MVERT (as with individual action selection) performance suffers due to high levels of sensing and motion noise in the system. However, the use of team information quickly allows the map to converge and produce accurate maps despite the presence of these sources of noise. Thus MVERT can succeed in noisy systems (question 8).

On average, over all trials over all configurations, the performance of MVERT in map accuracy (E_1), value (E_2), maximum uncertainty (E_3), time to mission completion (E_4), and robot localization (E_5) is much higher than that of individual action selection after a few steps of settling time. The mean value is shown versus time in Figure 65.

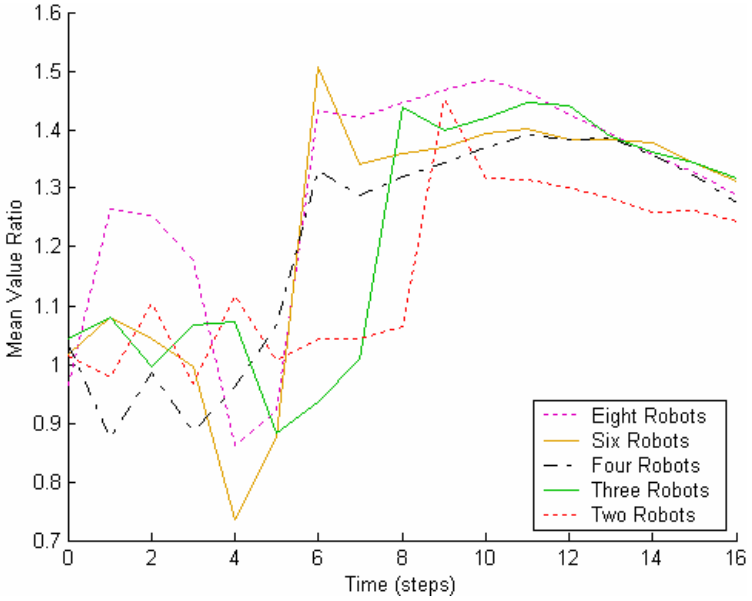

Figure 65. Mean Value (E_2) Ratio Over Time: Small-Scale Mapping Simulation
Numbers greater than 1.0 indicate better MVERT performance. Averaging over all trials, smoothing out noise, MVERT performs better than individual action selection after initial settling time due to high noise. Peak relative performance occurs early, and levels off slightly over time. Mean Value (E2) Ratio Over Time: Small-Scale Mapping Simulation.

The trajectories produced by MVERT, maximizing the joint value, spread out more and require less arcing around targets, as illustrated in the case of the Target Location Mission (Chapter 7). Individual action selection causes clustering and leads to multiple robots following the very similar trajectories, duplicating (rather than complementing) teammate observations. In smaller teams (Figure 60, two and three robots), robots often take trajectories that pass closer to teammates (rather than spreading more) if it provides an overlapping view with a teammate (as shown in the two and three robot cases). In larger teams, the greater number of observers makes this less necessary, and robots distribute over a greater area. Where in the smaller teams, the topmost robot moves below the top left target, in groups of four and more, this robot moves above; the bottom most robot moves blow the lower left target rather than above in teams of 3 and more. With the larger teams, MVERT only occasionally must select an alternate move to avoid a teammate (as in the case of six robots, in the lower left). Disregarding teammates produces larger clusters of robots, and sets of nearly identical trajectories.

8.3 MVERT Small-Scale Mapping with Physical Robots (M2)

As in simulation, the physical system experiments are designed to show how MVERT can be applied to mapping and assess any benefits derived from predicting teammate contributions (addressing questions 6 and 8). Additionally, these experiments illustrate how MVERT may be applied on a physical multi-robot system (addressing question 7). Experiments series M2, conducted in the physical environment, are:

C. 2, 3, and 4 robots mapping one configuration of landmarks;

D. Experiment series C using individual action selection.

The physical mapping environment is a single configuration of the simulation small environment, Figure 59 C1. Experiments are run for 16 steps.

Snapshots from the physical robot experiments are shown in Figure 66 and Figure 67. The first figure is a snapshot from an experiment applying MVERT action selection. Note that robots are spread out in the environment, moving on either side of the central target.

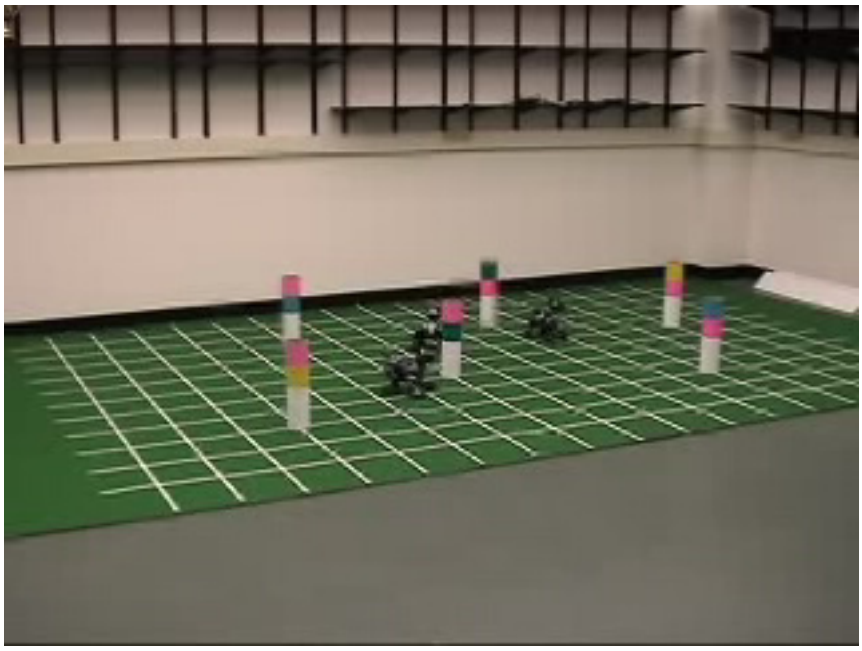

Figure 66. Physical System Small-Scale Mapping with MVERT
A snapshot from a trial of three robots mapping in the physical environment using MVERT action selection. In this case, the robots spread out in the environment, with one robot moving on the close side of the central target, to vary point of view and maximize value.

The second figure shows a snapshot from an experiment applying individual action selection. Note all the robots move on the far side of the central target and do not spread out as much as in MVERT (above).

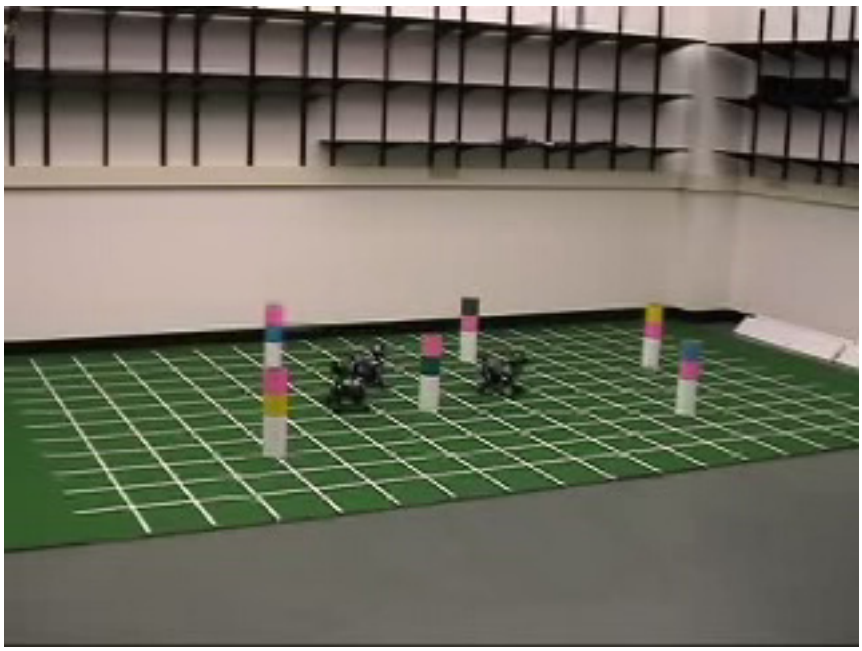

Figure 67. Physical System Small-Scale Mapping with Individual Action Selection
A snapshot from a trial of three robots mapping in the physical environment using individual action selection. Robots do not spread out; all robots move on the far side of the central target.

Examples of the trajectories produced using MVERT and using the individual action selection approach to map the small-scale physical environment are shown in Figure 68. Trajectories shown are those reported by the robot, not ground truth (error in localization is addressed later in this section).

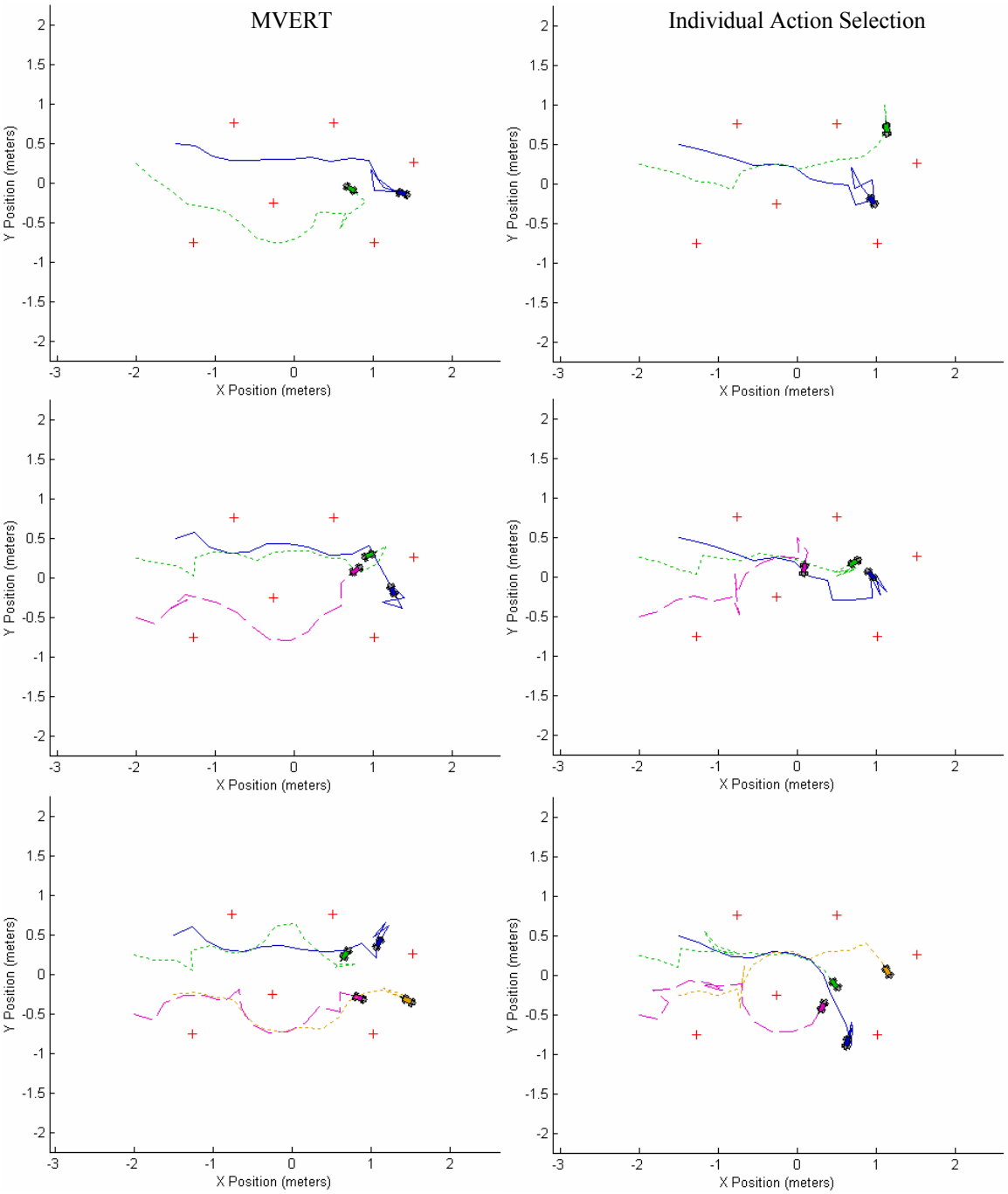

Figure 68. Experiments C and D, M2: Physical System Mapping Trajectories
Left: Trajectories generated by MVERT. *Right:* Trajectories generated by individual action selection. MVERT causes robots to spread out more than individual action selection, which causes robots to follow the same paths. This spreading accidentally cause crowding, as in the case of three-robots, where robot 3 (dashed) must turn around at step 3 to let robot 2 clear the narrow alley.

Quantitative results for small-scale mapping are in Table 17 - Table 19. Confidence intervals are not computed due to the low number of samples in each experiment.

Table 17. Mean Map Uncertainty (E_2 and E_3): Physical System

Robots		2	3	4
Mean E_2 (Step 12)	MVERT	-0.0099	-0.0073	-0.0050
	Individual	-0.0107	-0.0076	-0.0057
	Ratio	1.083	1.037	1.133
Mean E_3 (Step 12)	MVERT	0.0319	0.0284	0.0239
	Individual	0.0366	0.0309	0.0287
	Ratio	1.1490	1.0893	1.2031

Table 18. Final Map Landmark Error (E_1): Physical System

Robots		2	3	4
Mean E_1 (Step 16)	MVERT	0.0774	0.1025	0.0690
	Individual	0.0605	0.1055	0.0507
	Ratio	0.7816	1.0290	0.7346
Maximum E_1 (Step 16)	MVERT	0.1204	0.1620	0.1719
	Individual	0.0960	0.3364	0.0729
	Ratio	0.7970	2.0764	0.4244

Table 19. Maximum Robot Localization Error (E_5): Physical System

Robots	2	3	4
E_5 MVERT	0.29 m	0.44 m	0.18 m
E_5 Individual	0.86 m	0.31 m	0.29 m
E_5 Ratio Individual to MVERT	3.00	0.70	1.61

Mean value ratio of all configurations over time is shown in Figure 69. The mean maximum uncertainty ratio of all configurations is shown in Figure 70.

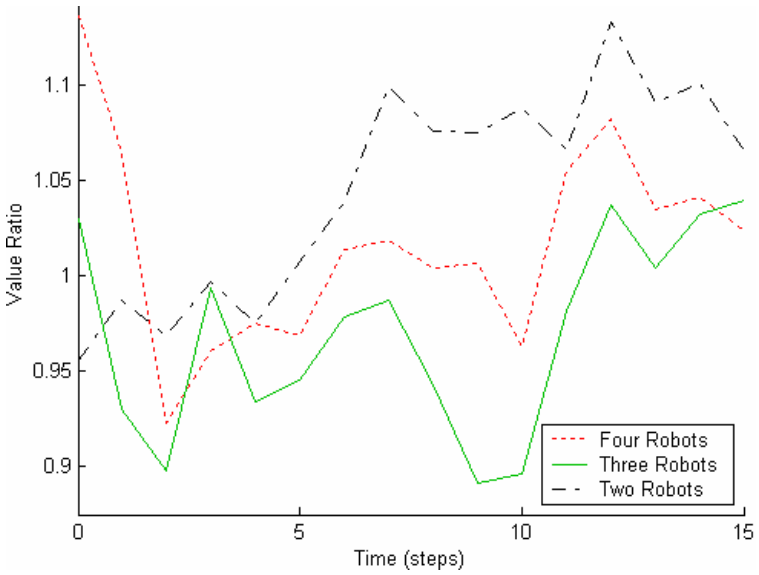

Figure 69. Physical System Value (E_2) Ratio Over Time: Individual vs MVERT
Numbers greater than 1.0 are better MVERT performance. After difficulties due to noise and backing up to avoid teammates, MVERT achieves and maintains a higher value than that produced using the individual robot approach.

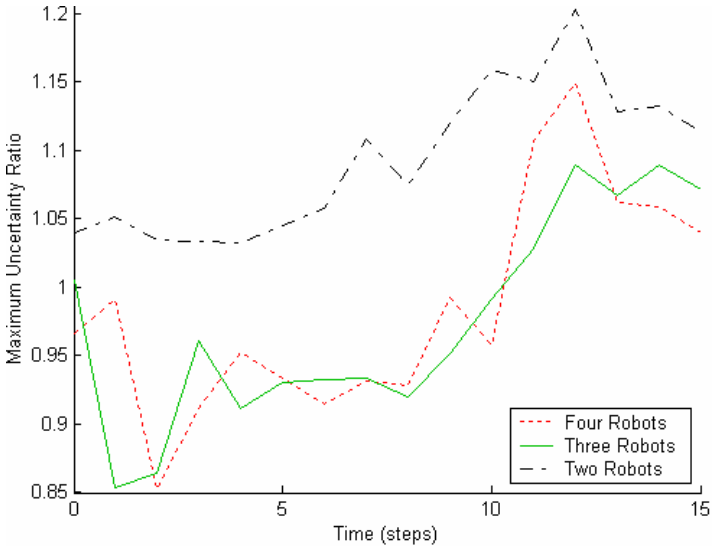

Figure 70. Physical System Maximum Uncertainty (E_3) Ratio Over Time: Individual vs MVERT Numbers greater than 1.0 indicate better MVERT performance. After initial noisy results, MVERT maintains lower maximum uncertainty than individual action selection.

A limitation of MVERT becomes apparent when examining the final map errors. In most cases, map error is lower in the individual action selection experiments than in the MVERT. This is a result of a larger number of total measurements being taken on each landmark in the individual action selection approach. As MVERT attempts to share information, each robot may rely on its teammate to take measurements on the other landmarks rather than obtaining them directly. These fewer observations are intended to be of higher quality, but due to the high level of noise this is often not the case. This is quite different from the simulation result, which indicates lower map errors for MVERT and is probably due to the discrepancy between the noise models and actual noise. MVERT is more sensitive to mismatch between sensor model and sensor performance in the presence of high noise, as it relies more heavily on predicting future results.

As with the Mapping Mission simulation experiments and the Target Location Mission experiments, there is a tendency toward improved relative performance as team size increases. As in the case of the simulation experiments, the robots using MVERT distribute themselves more within the environment. Trajectories vary dramatically with team size, as illustrated in Figure 68. In the two-robot case, the robots initially spread slightly more than in the individual action selection approach. Measurements are similar, and value remains nearly the same (though MVERT performs just slightly worse). Once reaching the middle of the field, the MVERT robots split around the center target while the individual robots do not. This results in an immediate improvement in value, both absolute and relative to the individual approach.

In the three-robot case, an interesting degredation of performance is observed. Early in the experiment (step 3), robot 2 (middle, dotted) has moved significantly away from robot 1 (top, solid) in order to maximize value with a complementary observation. This leaves robot 3 (bottom, dashed) no room to move forward at this step. Thus, robot 3 turns around to take a closer measurement on the lower-left landmark, causing it to observe only one or no landmarks in that step. In the individual approach, robot 2 has remained directly behind robot 1, leaving room for robot 3 to continue forward. Thus, robot 3 contributes measurements on many landmarks, reducing value more quickly than MVERT. After several steps, however, MVERT catches up and surpasses the value of the individual robot approach (Figure 69).

In the four-robot MVERT case, robot 2 (upper middle, dotted) remains slightly closer to robot 1 (top, solid), moving between it and the fourth robot (lower middle, dotted). This leaves room for robot 3 (bottom, dashed) to follow behind it, rather than turn around early in the experiment. In this case, robot 3 contributes more fully at this step (observing more landmarks), and allows the team to reduce value faster than the individual approach at every step. Clutter in the environment prevents MVERT from spreading quite as much as desired. Because of the narrow corridor between the upper left and middle left landmarks, robots 1 and 2 follow the same path until more space opens up. Similarly, due to the narrow passage between the middle left and upper left landmarks, robots 3 and 4 also temporarily follow the same path.

In some cases, individual action selection results in temporarily better values. With individual action selection approach, robots tend to follow similar trajectories. This causes a larger problem of obstacle avoidance, leading to a need to move in a direction that is less desirable overall. This is evident particularly in robot 3 (dashed) in the three- and four-robot experiments, which remains near the beginning of the experiment longer, and then lingers near the middle of the field. This ultimately leads to more total measurements, and measurements of a higher quality (shorter range) than those produced by the less-hindered MVERT trajectories. As a result of taking more measurements, the absolute error in the positions of these targets are lower in the individual action selection approach than in the MVERT approach. As in the simulations, occasionally avoiding obstacles can send individual robots in a more fortuitous direction.

Robots are typically better localized using MVERT than individual action selection. Ground truth is hand measured robots relative to gridlines in overhead photographs (within 0.05 m). Robots using individual action selection frequently collided with each other and landmarks. Bad localization in MVERT arises when robots cannot see landmarks for multiple steps, while larger map errors also contribute to localization errors in individual action selection. The shared map rapidly converges towards the ground truth with more measurements relative to the individual maps generated by the independent robots. The more accurate locations of landmarks in turn provide more accurate localization for the robots and a better foundation for their measurements.

The differences in the physical system performance compared to that of the simulation are most likely due to the differences in error. While both systems use the same error models, these models are only approximating the true error in the physical system, while they are exact in the simulation. Additionally, due to randomness, the individual robot approach obtains individual sensor measurements of higher quality. Despite this, MVERT is qualitatively attempting to collect complementary measurements to maximize value, and as the noise averages out over number of measurements, it ultimately produces maps with higher value, if not higher accuracy in this case.

8.4 MVERT Large-Scale Mapping in Simulation (M3)

In order to evaluate the scalability of the MVERT approach, the mapping task is implemented in larger environments. Two simulation environments are explored: a middle-scale mission with dense clustered targets and a large-scale mission with sparse, uniformly distributed targets. The experiments for M3 are:

E. 4, 9, 16, and 25 robots mapping the dense clustered environment;

F. Experiment series E with individual action selection;

G. 4, 9, 16, and 25 robots mapping the sparse uniform environment;

H. Experiment series G with individual action selection.

Two environments are explored to investigate performance in different types of scenarios. The middle-size dense clustered environment has many closely spaced targets that are distributed in a non-uniform manner. For the cluttered environment, sensors with relatively limited vision range and high accuracy are used for this task. A wide (but not omnidirectional) field of view is used, assuming a panning camera. This might model an outdoor urban disaster area with clusters of rubble and people. The large environment is an area sparsely populated by fewer, uniformly distributed targets. To be appropriate for this larger, sparse environment, a long-range omnidirectional sensor (high uncertainty) is used. This might model planetary exploration, mapping rocks. The sensor model parameters are summarized in Table 20. Only the target location value is used for these simple mapping tasks; robots are not forced to specifically explore. Ranges are selected to cover the typical inter-landmark spacing to allow natural discovery of new landmarks without inducing exploration. Robots start in a corner of the environment and run until equilibrium is reached. For each experiment, five runs are conducted in order to provide data for statistical analysis.

Table 20. Large-Scale Mapping Simulation Sensor Parameters

Environment	Steps	Vision Range	Range Noise Standard Deviation	Vision Angular Field of View	Bearing Noise Standard Deviation
Clustered, Dense	150	15 m	0.04r m	220º	0.25º
Uniform, Sparse	450	150 m	0.10r m	360º	1.00º

Example trajectories for the middle-sized clustered, dense environment are shown in Figure 71. Example trajectories for the large-scale uniform, sparse environment are shown in Figure 72.

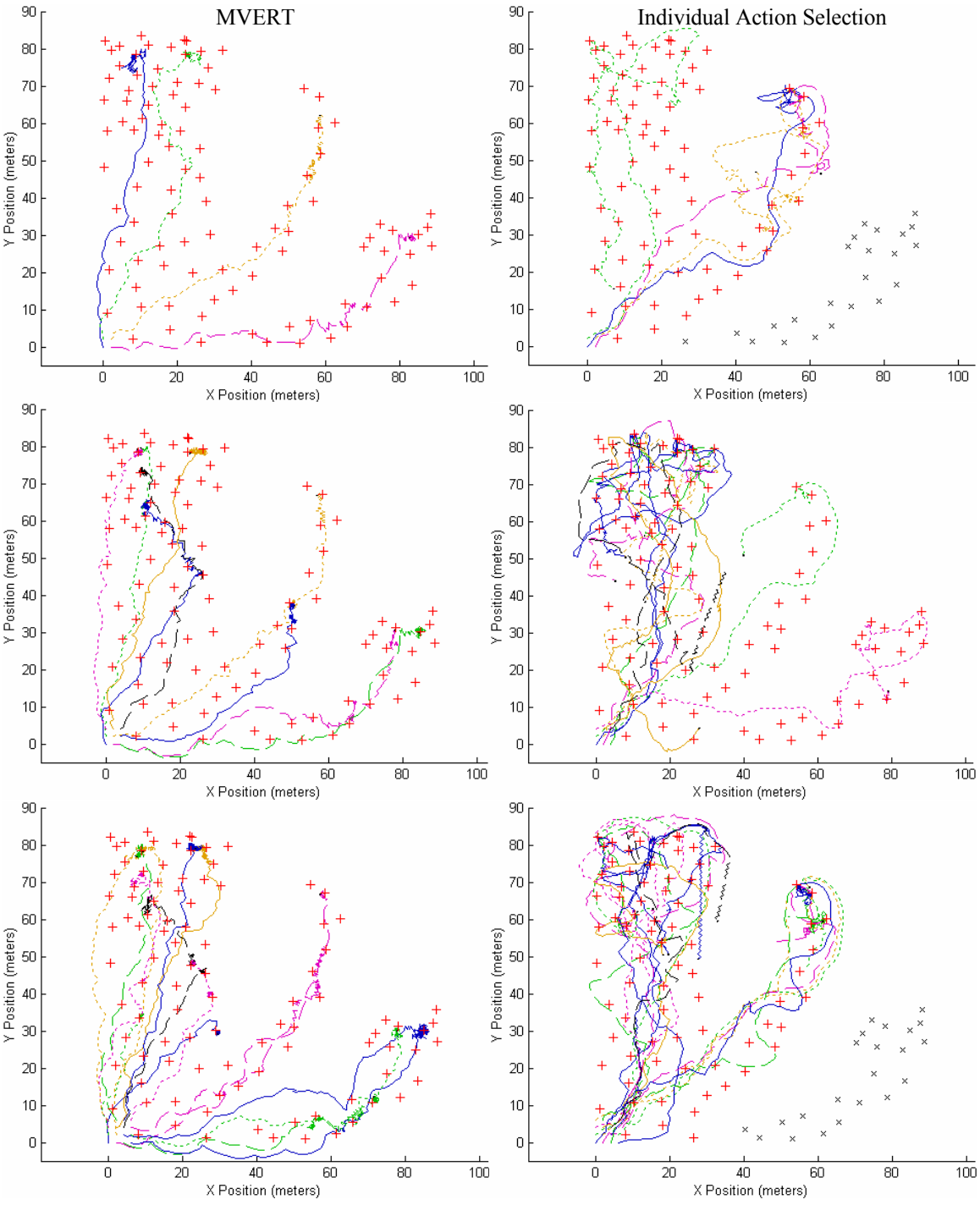

Figure 71. Experiments E and F, M3: Clustered, Dense Environment Simulation Trajectories Mapping with 4, 9, and 16 robots. The black x's are unobserved targets; red +'s are observed targets. *Left:* MVERT trajectories. MVERT causes robots to spread out in the environment and cover most or all of the space. Note opportunistic overlap of field of view with arcing, as in the lower left example, far left (dotted and dashed robots). *Right:* Individual action selection trajectories. Note tendency to take similar paths initially; these paths diverge as the map differences grow due to noise. Robots do not necessarily cover the space.

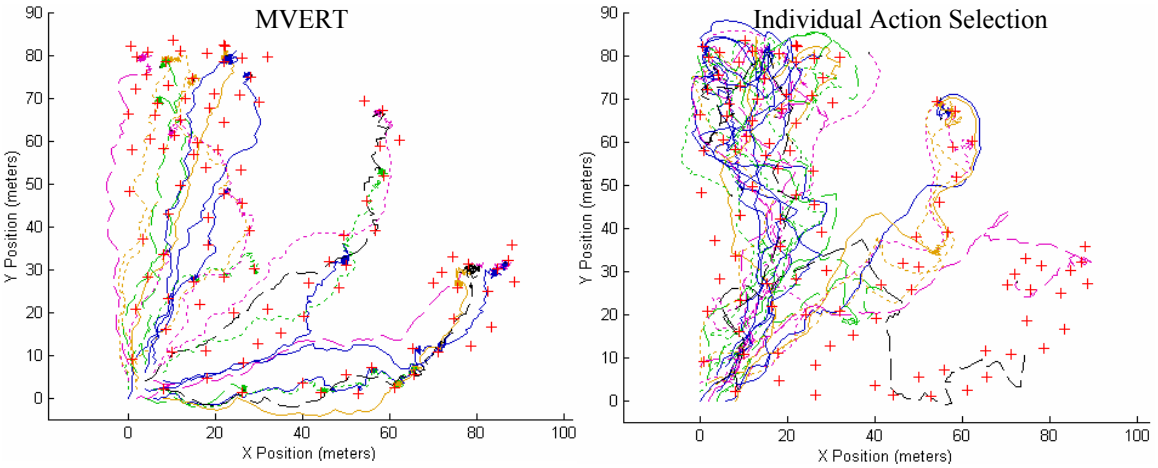

Figure 71 (Cont). Experiments E and F, M3: Clustered, Dense Environment Simulation
Left: MVERT with 25 robots. *Right:* Individual action selection with 25 robots. Note the random exploration (trajectories cross regularly) of individual action selection versus MVERT.

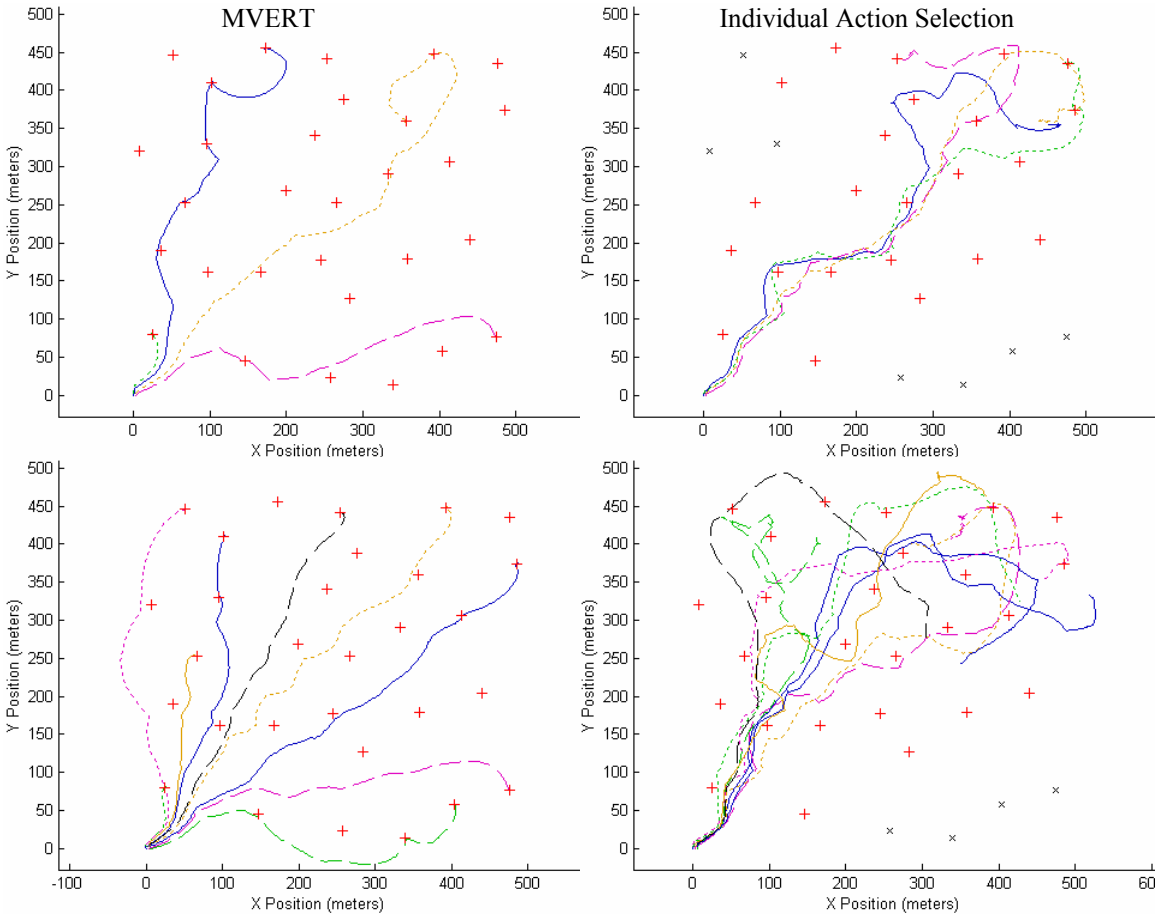

Figure 72. Experiments G and H, M3: Uniform, Sparse Environment Simulation Trajectories
Mapping with 4 and 9 robots. *Left:* MVERT trajectories. Robots spread out and cover the space. Note the opportunistic move to share information in the upper left example, solid and dotted robots at the top move toward each other to overlap field of view. *Right:* Individual action selection trajectories. Note very similar paths in upper right and large number of unknown targets (black x's).

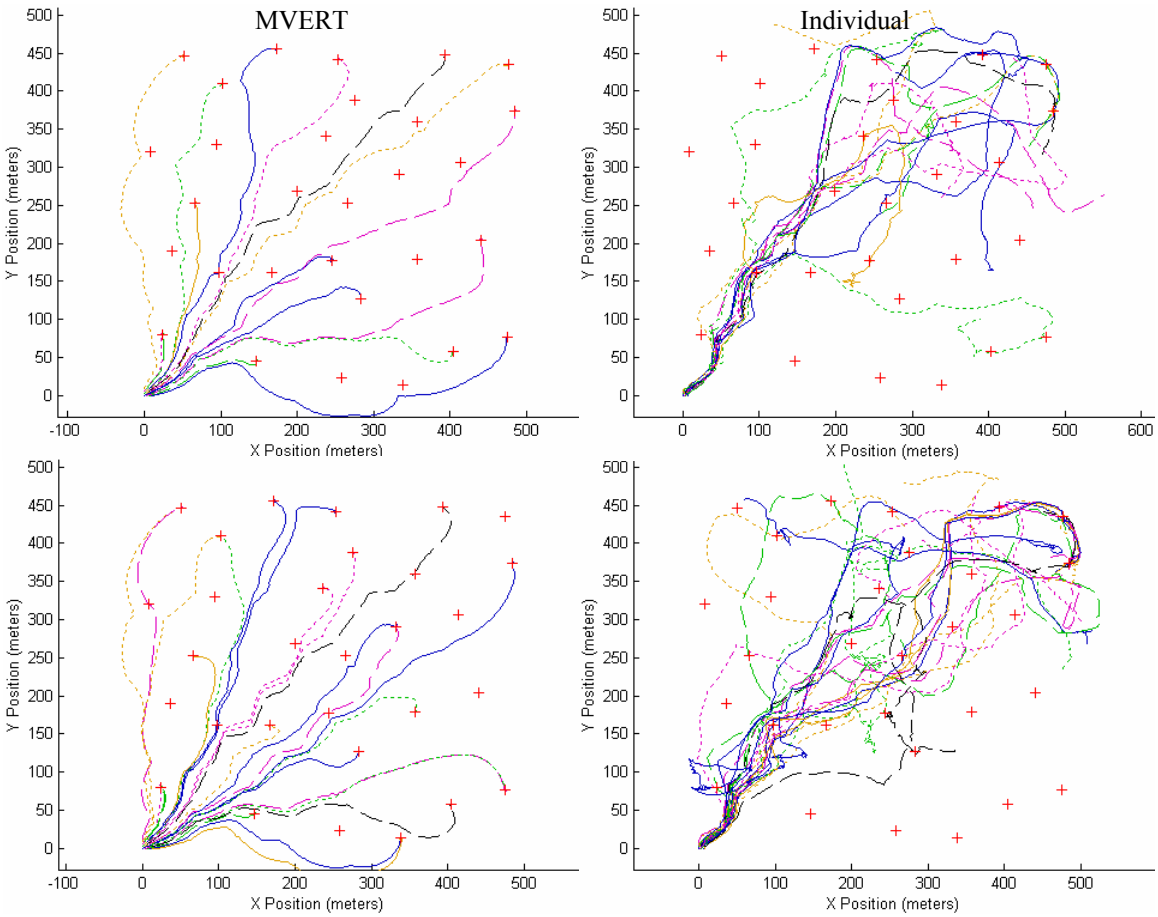

Figure 72 (cont). Experiments G and H, M3: Uniform, Sparse Environment Simulation Mapping with 16 and 25 robots. *Left:* MVERT, with distributed robots, moving to overlap fields of view. *Right:* Individual action selection, with chaotic exploration and non-uniform coverage.

Quantitative results are shown in Table 21 - Table 25. The mean over five trials is shown, with ratios representing ratio of the mean results at the trial end (steps 150 and 450, respectively). Standard deviations and confidence intervals (95%, in parentheses) of the five trials are provided. For Value, Maximum Uncertainty, and Map Error, only the known map is considered. This may or may not contain all targets, or contain the same number of targets for MVERT and individual action selection.

Table 21. Value (E_2): Large-Scale Mapping

Environment	Approach	Robots			
		4	9	16	25
Clustered, Dense	MVERT	-0.961±0.097 m^2 (±0.038)	-0.401±0.032 m^2 (±0.012)	-0.195±0.007 m^2 (±0.003)	-0.134±0.005m^2 (±0.002)
	Individual	-0.502±0.103m^2 (±0.040)	-0.409±0.148m^2 (0.058)	-0.232±0.132m^2 (±0.052)	-0.139±0.038m^2 (±0.015)
	Ratio	0.522	1.02	1.19	1.03
Uniform, Sparse	MVERT	-8.14±0.46m^2 (±0.18)	-2.89±0.15m^2 (±0.06)	-1.52±0.05m^2 (±0.02)	-0.928±0.06m^2 (±0.02)
	Individual	-15.98±10.64 m^2 (±4.2)	-8.82±9.66m^2 (±3.8)	-11.9±14.7m^2 (±5.8)	-7.93±8.1m^2 (±3.2)
	Ratio	1.96	3.06	7.81	8.55

Despite the fact that often fewer targets have been observed by the individual action selection, the total uncertainty of the fewer targets may still be much greater than that for more targets with MVERT. Robots concentrate on fewer targets and yet produce maps that are less certain than MVERT. The relative improvement increases rapidly with team size. Large standard deviations on individual action selection, particularly in the large environment, reflect some trials with very low values. The confidence limits, however, indicate that the improvement is significant. Relative value (Individual divided by MVERT) versus team size for both environments is shown in Figure 73. Relative value increases with team size.

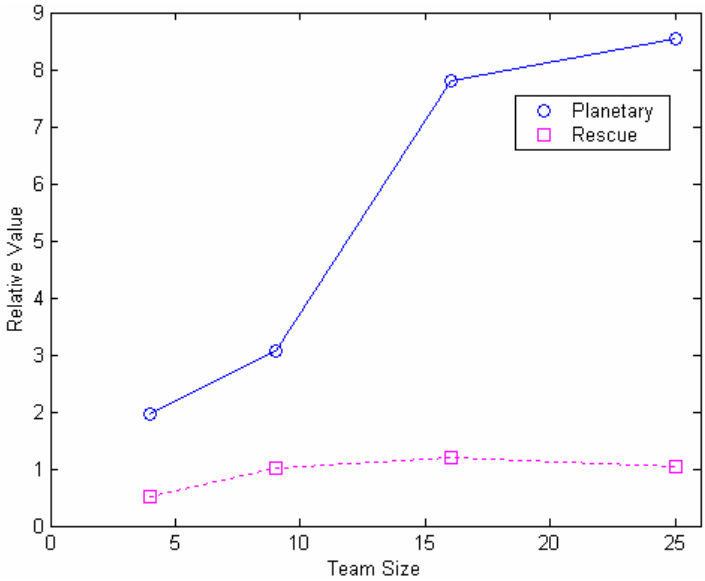

Figure 73. Relative Value Versus Team Size: Large-Scale Mapping Simulation
Relative value is value provided by individual action selection divided by the value provided by MVERT. Results are shown for the final step in each run. In both environments, the general trend is an improvement in relative value as team size increases.

Table 22. Maximum Uncertainty (E_3): Large-Scale Mapping

Environment		Approach	Robots			
			4	9	16	25
Clustered, Dense	Mean	MVERT	0.064±0.024m (±0.0001)	0.040±0.016m (±0.00006)	0.027±0.011m (±0.00004)	0.022±0.010m (±0.00004)
		Individual	0.054±0.016m (±0.00008)	0.040±0.019m (±0.00008)	0.030±0.016m (±0.00007)	0.021±0.011m (±0.00004)
		Ratio	0.843	1.00	1.09	0.979
	Maximum	MVERT	0.194 m	0.104 m	0.0656 m	0.0461m
		Individual	0.117 m	0.130 m	0.117 m	0.104 m
		Ratio	0.60	1.25	1.79	2.26
Uniform, Sparse	Mean	MVERT	0.351±0.092m (±0.001)	0.218±0.057m (±0.0007)	0.158±0.045m (±0.0006)	0.124±0.035m (± 0.005)
		Individual	0.453±0.561m (±0.008)	0.296±0.257m (±0.004)	0.275±0.450m (±0.006)	0.234±0.351m (±0.005)
		Ratio	1.29	1.36	1.75	1.89
	Maximum	MVERT	0.606 m	0.370 m	0.275 m	0.228 m
		Individual	6.15 m	2.43 m	5.38 m	3.66 m
		Ratio	10.16	6.57	19.51	16.08

In Table 22, *Maximum* indicates the largest E_3 (maximum uncertainty) over all five trials. *Mean* is the mean E_3 (maximum uncertainty) over all trials. Statistics can thus only be determined for the mean. In almost all cases, the final value of E_3 is improved in MVERT versus individual action selection, and the amount of relative improvement increases with team size. In the four-robot and twenty-five-robot cases for the clustered dense environment, the final maximum uncertainty is lower for individual action selection. This is due to the fact that individual action selection has continued to focus on a subset of the map, while MVERT has continued to add new targets, and works toward reducing their uncertainty. Thus, this lower maximum uncertainty does not indicate a failure of MVERT in performance. The larger relative improvement in the uniform environment over the clustered environment is due to differences in the sensing. With the much smaller field of view, MVERT robots have less opportunity to share information and mostly cover the space. Additionally, a larger number of the targets are not seen by the individual approach, which adds less uncertainty to the total value term, making the difference artificially reduced.

Table 23. Steps to Map Quality (E_4): Large-Scale Mapping

Environment	E_3 <	Approach	Robots			
			4	9	16	25
Clustered, Dense	∞	MVERT	148.4±9.1	101.0±5.4	90.4±3.5	83.2±2.7
		Individual	--	130.4±23	117.5±29	116.8±31
		Ratio	--	1.29	1.30	1.40
	0.5	MVERT	149.2±8.8	102.0±5.4	91.4±3.5	84.2±2.7
		Individual	--	131.2±23	126.8±31	116.4±32
		Ratio	--	1.29	1.41	1.38
	0.1	MVERT	--	146.0±40	101.6±3.4	94.8±4.3
		Individual	--	148.4±17	151.0±22	125.2±30
		Ratio	--	1.03	1.49	1.32
Uniform, Sparse	∞	MVERT	293.2±29.4	271.8±23.7	262.2±4.5	258.6±3.3
		Individual	418.2±62.8	397.8±97.5	309.6±80.4	304.4±80.9
		Ratio	1.43	1.46	1.18	1.18
	2.0	MVERT	320.4±26.0	287.0±32.5	274.4±7.3	268.6±4.0
		Individual	425.0±58.1	390.0±83.6	326.2±72.8	346.2±87.1
		Ratio	1.33	1.36	1.19	1.29
	0.5	MVERT	--	318.2±8.2	308.6±6.5	298±3.1
		Individual	447.0±8.9	444.5±15.9	439.8±25.0	425.0±58.1
		Ratio	--	1.40	1.43	1.43

The number steps to a given uncertainty (E_4) includes all targets; unknown targets are assigned a maximum uncertainty of ∞. To reach a maximum uncertainty of ∞ implies that all targets have been observed at least once. The mean is computed assigning a time of one step longer than the experiment to those runs that did not reach the desired uncertainty, thus means may be artificially low for the individual action selection. If no runs reached the desired uncertainty, a dash is entered. Relative improvement has a slight trend to decrease as the number of robot increases. In larger teams, randomness due to measurement and motion noise as well as collision avoidance with teammate and targets can encourage the robots to explore more, and thus find more of the targets. The consistency also improves with team size, as demonstrated by the decreasing standard deviations as the number of robots increases.

Table 24. Completeness (E_6): Large-Scale Mapping

Environment	Approach	Robots			
		4	9	16	25
Clustered, Dense	MVERT	99.0% (95-100)	100%	100%	100%
	Individual	76.0% (71-78)	95.6% (78-100)	92.0% (78-100)	99.4%(97-100)
	Ratio	1.30	1.05	1.09	1.01
Uniform, Sparse	MVERT	100%	100%	100%	100%
	Individual	91% (77-100)	96% (87-100)	98% (90-100)	100%
	Ratio	1.10	1.04	1.02	1.00

Completeness considers the entire true map, comparing the known map to the true map. The range of values for Completeness (if non-zero) is shown in brackets. In only one case, MVERT fails to completely explore the space in the middle-scale clustered, dense environment with the four robot team, as the robots become stuck in a local optimum. This failure is due to robots reaching a local optimum from which no other targets can be observed by moving one step away. Individual action selection fails to find all the targets in at least one trial for all team sizes less than 25. MVERT provides a larger completeness benefit for smaller teams because it results in distributing robots through other means than collision avoidance and the small differences in covariance due to measurement errors. MVERT performance is also much more consistent, with much smaller ranges in results than for individual action selection.

Table 25. Map Error (E_1): Large-Scale Mapping

Environment		Approach	Robots			
			4	9	16	25
Clustered, Dense	Mean	MVERT	0.140±0.068m (±0.0003)	0.130±0.066m (±0.0003)	0.121±0.058m (±0.0002)	0.130±0.062m (±0.0002)
		Individual	4.48±5.75m (±0.030)	3.40±7.16m (±0.030)	45.4±86.1m (±0.37)	438.5± 77.5m (±0.31)
		Ratio	32.1	26.4	377	295
	Maximum	MVERT	0.462 m	0.333 m	0.286 m	0.261 m
		Individual	25.6 m	82.5 m	628 m	447 m
		Ratio	55.5	247	2197	2112
Uniform, Sparse	Mean	MVERT	2.68±1.02m (±0.013)	2.66±1.03m (±0.013)	2.77±1.08m (±0.014)	2.76±1.10m (±0.014)
		Individual	57.3±88.9m (±1.27)	618±1267m (±17.2)	4442±6576m (±87.7)	$4x10^7$±$1x10^7$m ($\pm 1x10^5$)
		Ratio	21.4	232	1602	1453941
	Maximum	MVERT	5.27 m	5.04 m	5.13 m	5.24 m
		Individual	645 m	10053 m	41150 m	$5x10^7$ m
		Ratio	122	1993	8021	9.6 $x10^6$

Final map errors for known targets are in Table 25. *Maximum* is the largest error over all trials. *Mean* is the average error of all targets, then over all trials. Maximum error is approximately constant for MVERT across team sizes, while the addition of lost robots' measurements rapidly degrades quality for individual action selection. Very large standard deviations and confidence intervals for the large-scale environment result from lost robots integrating erroneous measurements into the map. Robots do not always become lost, leading to a large range in performance. Performance consistency is much higher in MVERT.

Example maps generated by MVERT and individual action selection are shown in Figure 74. Trials with twenty-five robot are illustrated because they show the greatest completeness of maps, though accuracy is reduced for individual action selection.

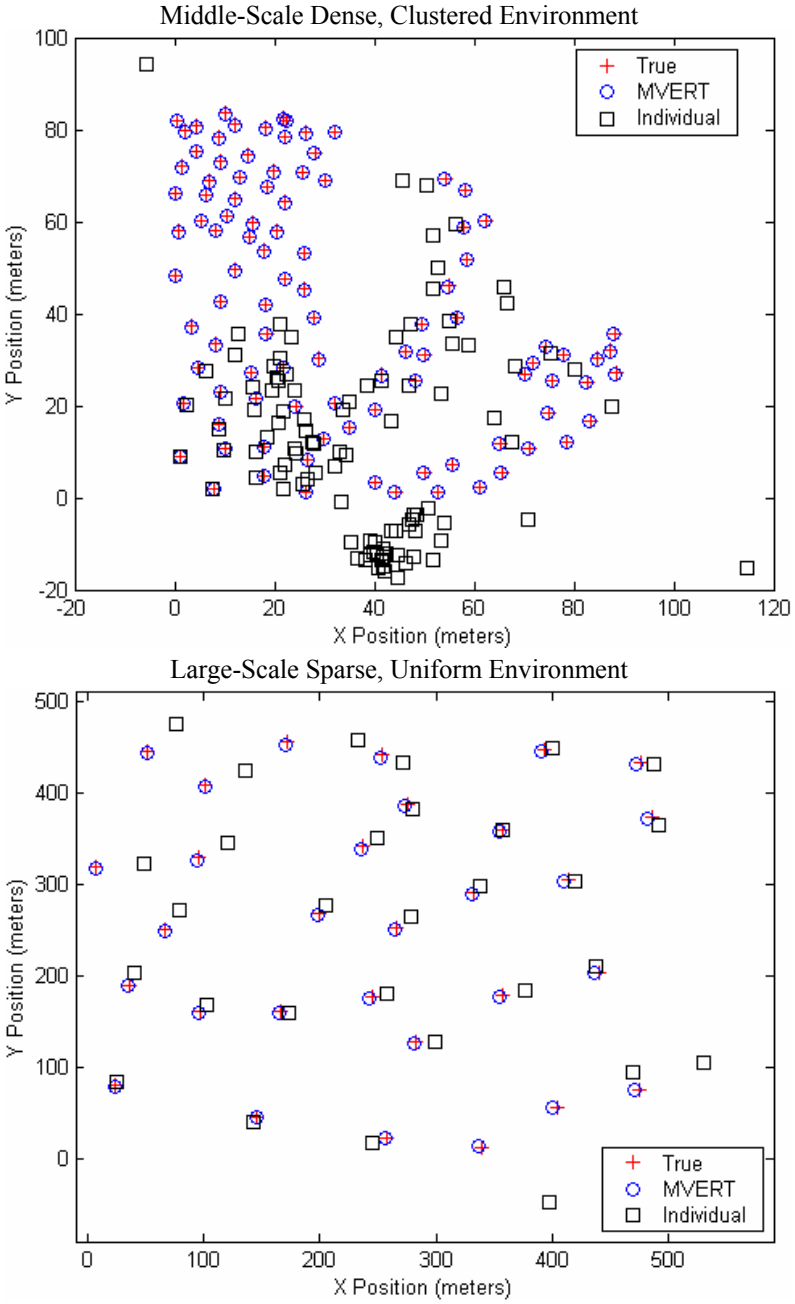

Figure 74. Comparison of Map Results: Large-Scale Mapping Simulation
Top: Example maps for the middle-scale environment with dense, clustered targets. *Bottom:* Example maps for the large-scale environment with sparse, uniformly distributed targets. Note in both cases, map accuracy is similar in landmarks that are close to the robot start locations (square and circles properly overlap the +'s). As robots move further away, however, the discrepancy increases for individual action selection (squares do not overlap +'s) while MVERT continues to maintain a high level of accuracy. With individual action selection, despite covering the space, inaccurate robot positions lead to mapping upper left landmarks to other locations (see central bottom cluster). It is also evident in that several landmarks are never observed by individual action selection.

Table 26. Localization Error (E_5): Large-Scale Mapping

Environment	E_5	Approach	Robots			
			4	9	16	25
Clustered, Dense	Mean	MVERT	0.197±0.760m	0.164±0.042m	0.155±0.049m	0.151±0.055m
		Individual	46.6±40.6m	262±662m	801±22.5m	1397±47.8
		Ratio	241	1596	5171	9222
	Maximum	MVERT	0.319 m	0.295 m	0.264 m	0.258 m
		Individual	365 m	848 m	4149 m	8188 m
		Ratio	1146	28737	157277	319870
Uniform, Sparse	Mean	MVERT	2.78±1.52m	2.78±1.24m	2.86±1.21m	2.79±1.19
		Individual	403±716m	1322±6125m	8114±51397m	$2x10^6$±2 $x10^7$m
		Ratio	145	476	2838	$8x10^5$
	Maximum	MVERT	4.66 m	4.62 m	5.27 m	4.80
		Individual	2256 m	40928 m	4.5 $x10^5$ m	$2.8x10^8$ m
		Ratio	484	8846	86141	5.8 $x10^7$

Results for localization error are shown for the final step of the experiment. Heading results are not displayed for clarity, but present the same pattern of results as for position. Again, the large uncertainties in the individual action selection result from some robots that become very lost during some of the trials; the large variances are due to the fact that most robots do not become completely lost (in some trials no robots become lost), keeping their uncertainties lower.

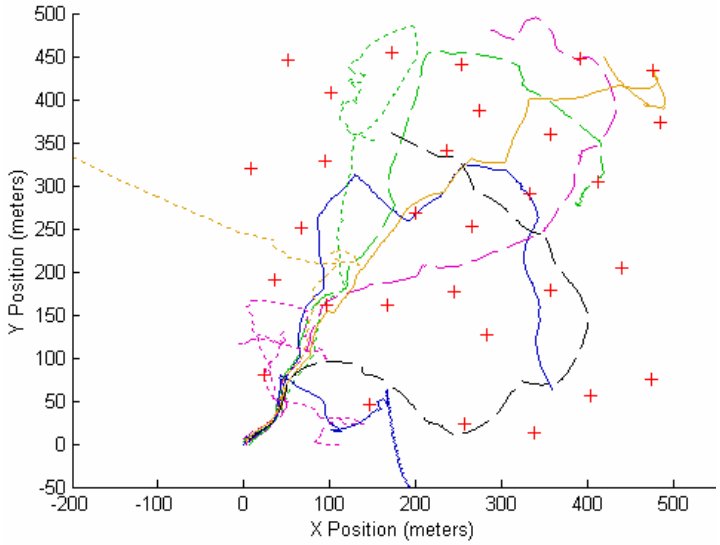

Figure 75. Individual Action Selection Lost Robots: Large-Scale Mapping Simulation
An example of a case in which several robots become lost during execution. Two robots wander away from the mapping area (dotted left, solid bottom). Targets observed after become lost become poorly located within the map. Once leaving the exploration area, robots no longer contribute (positively or negatively) to the map, effectively reducing the team size.

In each case, the resulting value at each step (E_2) and the maximum uncertainty (E_3) are improved using MVERT relative to individual action selection, when the number of visible targets is considered. The amount of improvement increases with team size, except in the case of final map error, though the map of

the same quality is achieved much faster with the larger team. Larger teams can share observations further into the exploration due to their larger numbers. Occasionally, the value of the individual action selection's map will be greater than that generated by MVERT; as in the prior sections, this occurs when MVERT observes more targets than the individual action selection, and thus has more targets are contributing to the total uncertainty area. Additionally, the time to desired map quality (E_4) is much improved using MVERT. With smaller teams (four robots), individual action selection fails to search the space, and leaves landmarks undiscovered. Final map error (E_1) is also improved using MVERT. The map error is always better using MVERT than using individual action selection, indicating that the resulting map is always of a total higher value when comparing the same number of observed targets. In MVERT, the joint map assists in two ways: first, robots becoming lost can rely on a joint map to relocalize; second, if their observations lead to large discrepancies with the observations of other (well-localized) robots, the SLAM update will assign a low weight to the lost robot's measurement. In individual action selection, the lack of a joint map to counter these bad measurements allows uncertainty to grow. Lastly, localization error (E_5) is vastly improved using MVERT. The large errors in pose in many of the individual action selection cases indicate robots are occasionally completely lost, moving outside the mapping area, and unable to contribute to the generation of the joint map. Apparent better localization in the large environment is due to the fact that uncertainty per step is the same but the step size is larger; thus, over long range, total uncertainty is less.

In both large environments, MVERT adequately distributes the robots in the environment, and results in sharing complementary information for as long as fields of view can overlap. In the non-uniform middle-scale environment, robots map all clusters of targets, but more robots focus on the larger clusters of targets. As robots move further away, robots specialize in individual groups of targets, and the space is covered without explicit planning. Using individual action selection, robots typically tend to follow similar paths, only branching away for collision avoidance or due to small differences in the local map being optimized. The larger cluster of targets are more attractive, often leading all the robots to map these clusters, leaving others unmapped; this occurs even as team size increases (as shown with twenty-five robots, though occasional noise and collision avoidance results in robots covering the space better (as with nine robots). Each robot maximizing its own value also results in robots remaining close to central targets, leaving more distant ones more uncertain. In the uniform large-scale environment, with a smaller number of landmarks available at a time, poorly localized by few measurements early in the experiment, allows some robots to become lost and follow trajectories outside of the exploration area. This effectively reduces the team size and prevents these robots from productively contributing to the map. If lost robots leave the exploration area quickly and observe no targets, the map is not corrupted. However, in some cases, the robot may remain lost within the exploration area, adding erroneous observations to the map.

MVERT does suffer from some problems due to local optima when there are gaps in the environment larger than the robot's sensing area. For example, in the large-scale mapping scenario, one or more robots move to investigate the lower left target, which leaves them more than one step away from any other landmarks. Unable to improve estimates on other landmarks, they remain close to the one landmark. These local optima do not necessarily result in a failure, however, due to the fact that other robots, relying on these teammates to explore the lone landmark, move away and continue to explore other areas. A failure only can arise if each robot reached such a local optimum before the space was explored. This can be avoided by introducing additional value for exploring unknown areas, or adjusting the value function to increase for moving toward uncertain targets even if they are not going to be visible for several steps. This type of exploration will be discussed and implemented in Chapter 10.

8.5 Comparison to Mapping with Coverage Patterns (M4)

Coverage patterns are a typical approach for single- and multi-robot mapping. Comparison with this approach addresses question 7 (comparison to other approaches). For coverage pattern mapping, the exploration area is divided into sectors of equal size, each of which is assigned to a robot for mapping. As robots move toward their designated area and observe targets, they contribute to the joint map. For these experiments, robots fully communicate and build the map jointly as they explore to compare action selection approaches only. Each experiment is run five times. The experiments for series M4 are:

I. 4, 9, 16, and 25 robots mapping the middle-scale clustered simulation environment;

J. 4, 9, 16, and 25 robots mapping the large-scale uniform simulation environment.

Coverage patterns were assigned to provide each robot an equal area to map. The total map area is divided into square cells for mapping. Robots with areas further from the initial position take longer to achieve completion of their pattern. Robots drive directly to the closest corner of the region assigned for mapping, diverting only for collision avoidance with teammates and targets. Coverage patterns are defined by waypoints at the boundary of each row of the pattern. Robots choose the move that reduces the distance to the current waypoint the most while not moving too close to targets and teammates for collision avoidance. Robots also divert from the designated coverage pattern if necessary to avoid collisions with targets or teammates. Evaluation is based on completeness and accuracy of the map.

In the case of middle-scale environment, the coverage pattern rows are separated by a distance of one sensor range. In the case of the large-scale environment, the coverage pattern rows are separated by a distance of one-third of a sensor range. In both cases, this allows targets to be observed in multiple passes (at least two). This spacing is maintained also between adjacent map areas. The closer relative spacing in the large-scale mapping environment is intended to ensure targets can be observed from closer-range to compensate for the noisier sensor. Rows are interrupted at map boundaries by approximately one step size; thus the number of measurements taken along each row is approximately constant across team sizes.

Example trajectories for mapping with coverage patterns are shown in Figure 76 and Figure 77.

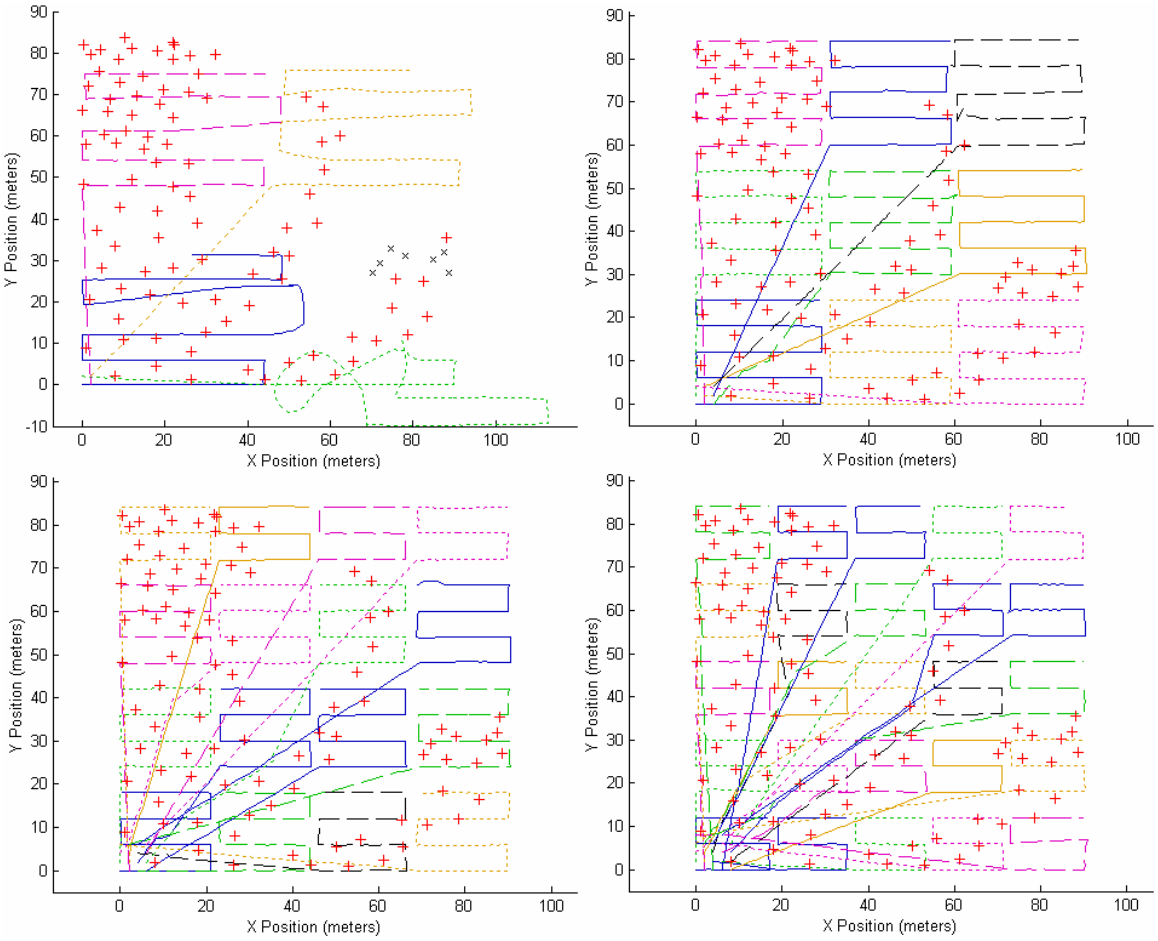

Figure 76. Experiment I, M4: Large-Scale Coverage Pattern Clustered Environment Mapping
Examples of coverage patterns for the middle-scale clustered environment. Robots covering areas with fewer landmarks become more lost (such as the two right robots in the four-robot case). As the number of robots increases, the area assigned to each is reduced; this leads to better performance, as there is more overlapping area where robot regions meet and shorter paths keep robots from becoming most lost. Additionally, at the start of exploration while many robots are in the same region, landmarks in this area become well located, providing an accurate start to each robot as it moved into new territory.

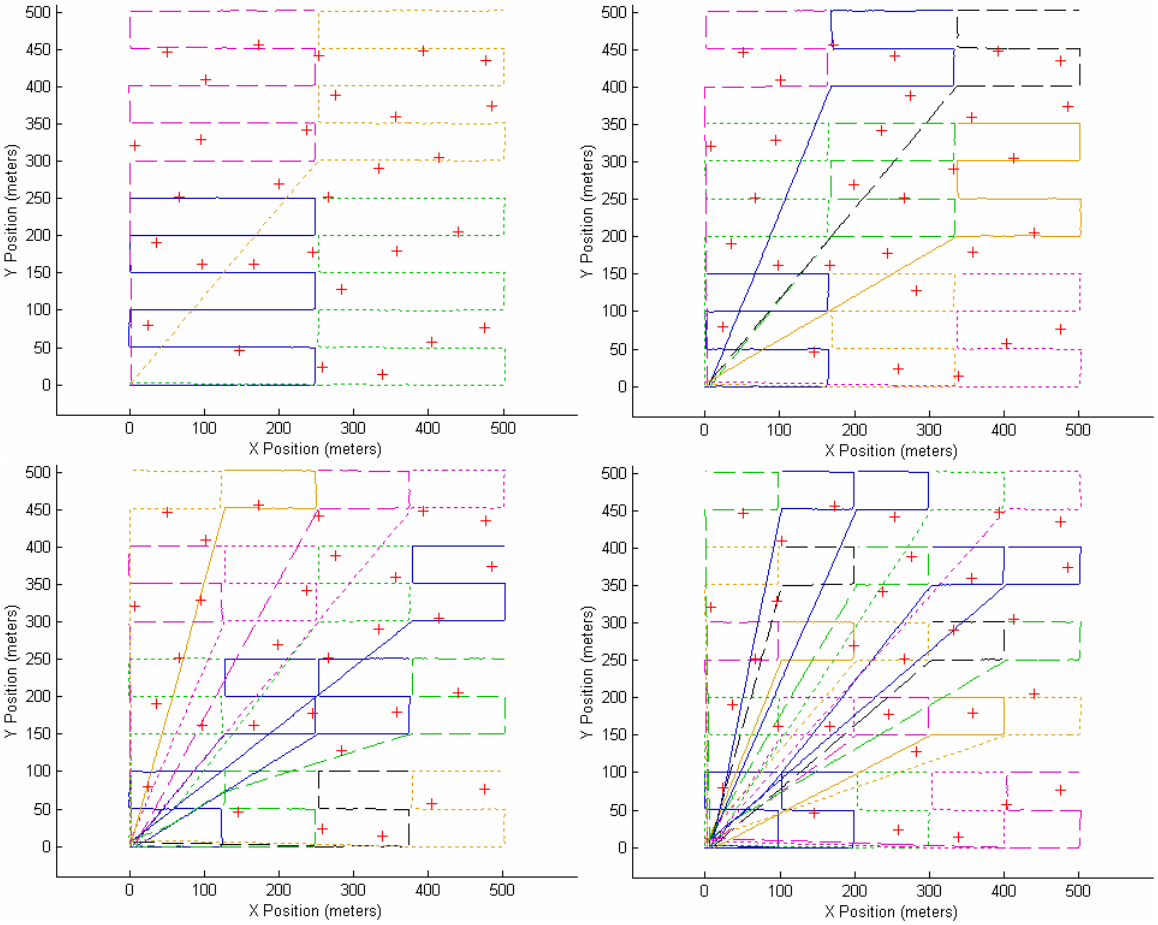

Figure 77. Experiment J, M4: Large-Scale Coverage Pattern Uniform Environment Mapping
Examples of coverage patterns for the middle-scale uniform environment with four, nine, sixteen and twenty-five robots.

Quantitative results comparing coverage patterns with MVERT are shown in Table 27 and Table 28. Standard deviations and 95% confidence intervals for the five runs are shown for mean errors in the tables.

Table 27. Map Error (E_1): Clustered Environment Mapping MVERT and Coverage Patterns

Error Type	Approach	Robots			
		4 (Step 155)	9 (Step 106)	16 (Step 94)	25 (Step 85)
Mean Error	MVERT	0.139±0.068 m (±0.0003)	0.132±0.074 m (±0.0003)	0.113±0.072 m (±0.0003)	0.123±0.067 m (±0.0003)
	Coverage	5.31±5.4 m (±0.288)	0.103±0.078 m (±0.0003)	0.070±0.040 m (±0.0002)	1.51±4.3 m (±0.017)
Maximum Error	MVERT	0.435 m	0.507 m	0.782 m	0.659 m
	Coverage	1039 m	0.584 m	0.584 m	46.7 m

Results reflect error present at the first step at which both approaches have located all the targets. Traversing large areas without landmarks leaves robots susceptible to becoming lost and producing low-quality maps. In clustered environments, robots following coverage patterns traverse larger areas without landmarks and thus lead to high map errors, particularly in those regions covered by robots that cross large open areas. MVERT provides a large advantage in this type of environment, as robots remain in the parts

of the environment populated by landmarks. This allows more measurements on landmarks and keeps robots well localized, improving the quality of observations. In sparse environments, both approaches are prone to getting lost. However, the larger number of turns executed by MVERT robots compared to the relatively straight paths of coverage patterns causes MVERT robots to get more lost due to the high motion noise in turns. In this case, MVERT produces reasonable maps, but of higher error than coverage patterns.

Table 28. Map Error (E_1): Uniform Environment Mapping MVERT and Coverage Patterns

Environment	Approach	Robots			
		4 (Step 336)	9 (Step 314)	16 (Step 267)	25 (Step 261)
Mean Error	MVERT	2.48±0.99 m (±0.01)	2.32±0.99 m (±0.01)	2.35±0.90 m (±0.01)	2.28±0.89 m (±0.01)
	Coverage	1.60±1.24 m (±0.02)	1.48±0.59 m (±0.01)	1.64±0.62 m (±0.01)	1.73±0.64 m (±0.01)
Maximum Error	MVERT	5.48 m	9.10 m	14.4 m	15.4 m
	Coverage	8.71 m	4.20 m	4.75 m	5.86 m

Completeness with 95% confidence interval over the five runs is shown in Figure 78 and Figure 79.

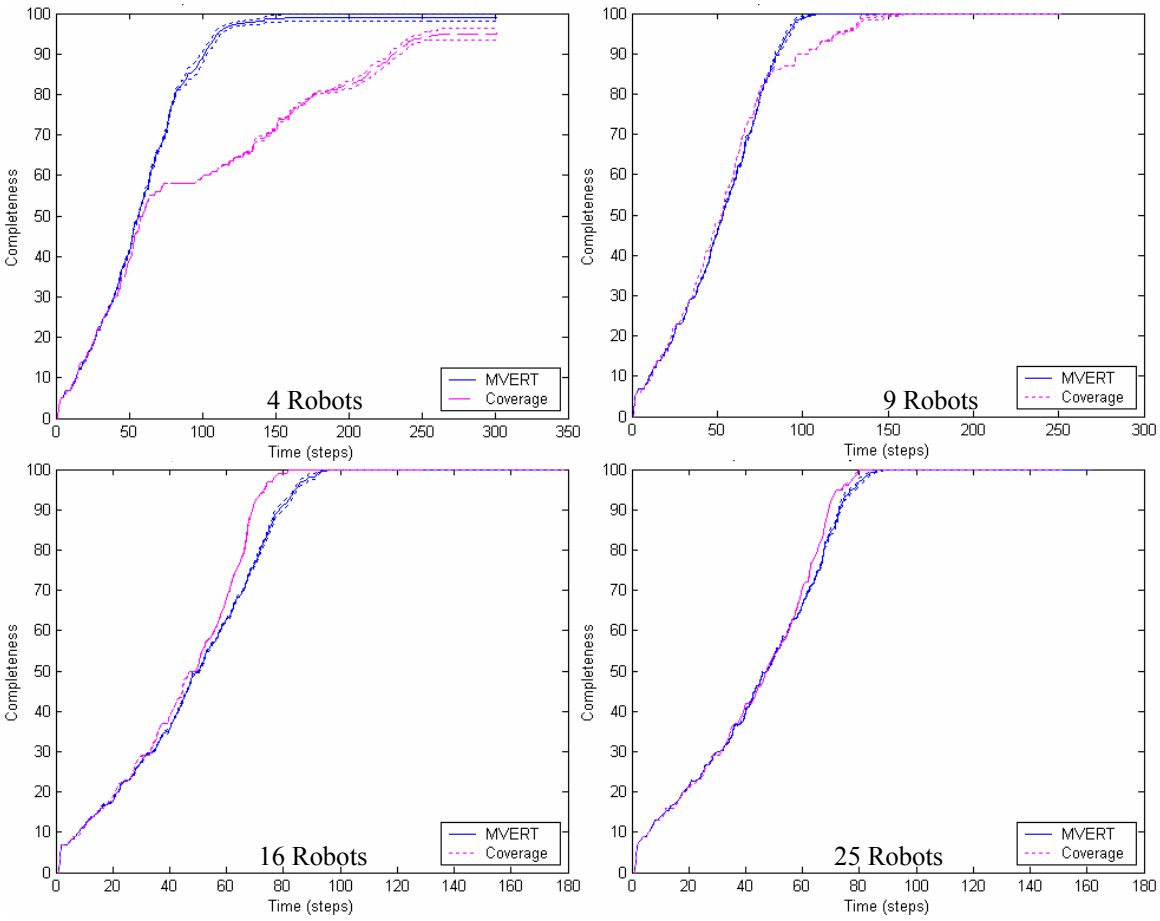

Figure 78. Completeness: Middle-Scale Mapping Clustered Environment Simulation

Mean completeness for robots mapping the middle-scale environment. MVERT provides a large advantage for smaller teams, allowing them to more quickly move to the farther away landmarks, while in small teams robots following coverage patterns must traverse much of their pattern before finding these landmarks. With larger teams, robots observe close landmarks well, giving coverage patterns a well-localized start. Additionally, this reduces the amount of empty space robots must traverse.

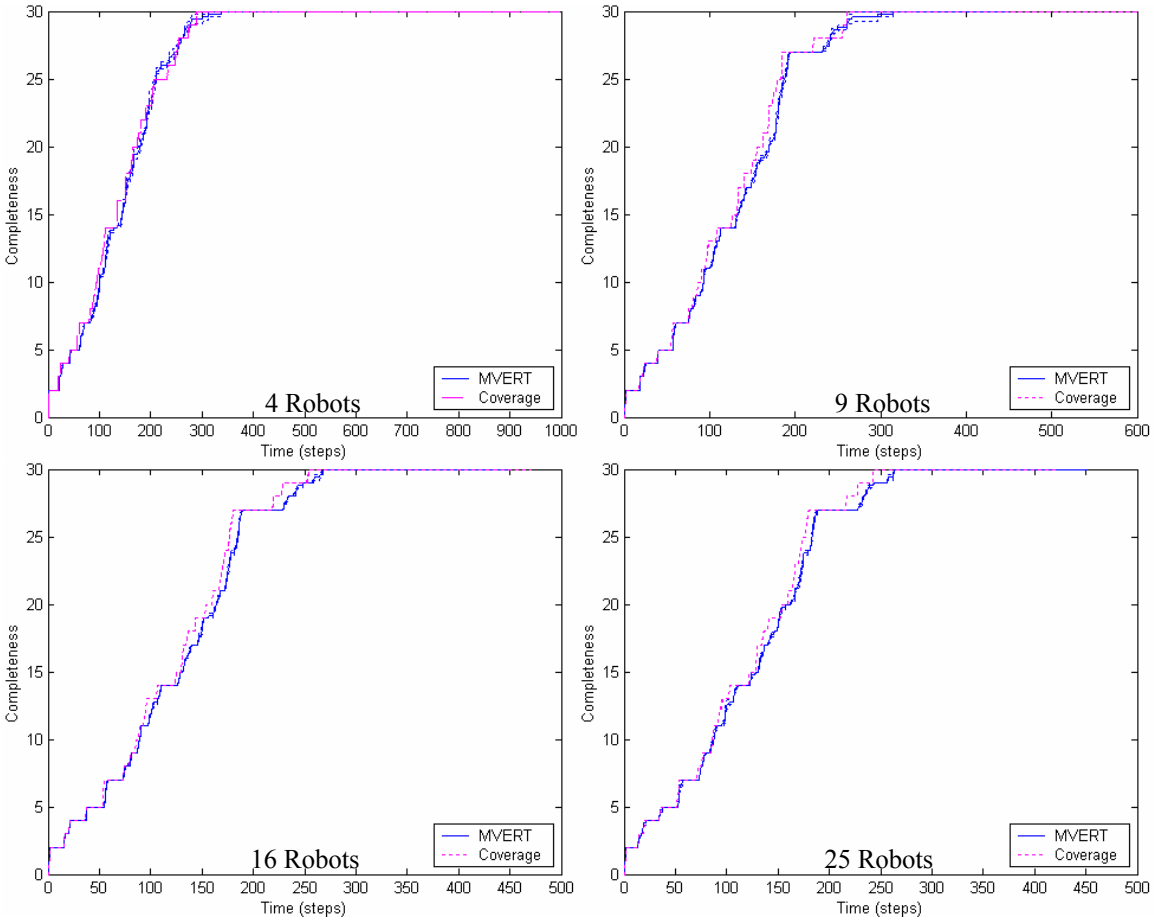

Figure 79. Completeness: Large-Scale Uniform Environment Mapping
Mean completeness for robots mapping the large-scale uniform environment. Performance using MVERT and coverage patterns are quite close. The enforced dispersion of coverage patterns and small map areas for large teams allows the coverage patterns to produce maps that are often slightly more complete than MVERT robots at any given time.

The differences in completeness for the clustered environment with four and nine robots are statistically significant, as the means differ by more than the 95% confidence intervals and by more than three standard deviations. The small differences favoring coverage patterns over MVERT in larger teams and in the large-scale mapping environment are only marginally statistically significant.

Coverage patterns are guaranteed to cover the space if the inter-row spacing is less than or equal to the sensor range and the boundaries of the coverage patterns define the desired exploration area. Driving along straight lines does provide a variety of views on targets as the robots pass them, allowing for some triangulation with measurements if needed due to an asymmetrical sensor.

In the case of a non-uniform environment in which clusters of landmarks are present, MVERT may perform much better than coverage patterns. MVERT focuses more robots on areas with greater numbers of landmarks, taking more observations and thus reducing total map uncertainty, while coverage patterns focus robots in a uniform manner. Thus, MVERT may reach desired levels of completeness much more quickly than coverage patterns. While both approaches may be susceptible to robots getting lost if they are mapping an area alone, particularly when the landmarks are sparse, MVERT is less likely to direct robots into areas that are void of landmarks. Coverage patterns require robots to pass through areas even if they do not seem have landmarks, and if little overlap exists between submaps, large uncertainties in sensing and motion can quickly lead to large map errors and lost robot. Coverage patterns are better able to provide

good localization in sparse patches when landmarks overlap with adjacent robot territories, allowing for better locations on these landmarks. As the team size grows, the relative performance of coverage patterns increases as robots travel directly through more of the space to reach smaller (faster) mapping areas.

In the case of a sparse, uniform environment, MVERT and coverage patterns perform similarly. Coverage patterns obtain a desired level of completeness slightly more quickly than MVERT because robots that are mapping far areas move directly to those areas rather than wandering slightly to maximize local observation value. Coverage patterns also obtain slightly higher map quality in terms of final area in this type of environment because the robots assigned to each area stay within that area, taking more observations, while robots using MVERT move on to new areas and do not tend to move back to regions previously covered.

The apparently better performance in coverage patterns of uniform environment exploration compared to clustered environments is partially due to differences in the motion model. The Aibo can be calibrated to any single step size with great reliability. Thus, while both environments have the same uncertainty per step, the larger step size provides robots in the large environment less uncertainty over long ranges.

In answering question 7 (regarding comparison to other approaches), MVERT provides a great advantage over coverage patterns in non-uniform environments with smaller teams that can make better use of resources than simply dividing up the search area. MVERT provides quite similar performance to coverage patterns in uniform environments and with very large teams. Thus MVERT can produce better results than coverage patterns in environments of this particular structure. Lastly, coverage patterns do not provide flexibility for taking advantage of unexpected opportunities, or for prioritizing targets or tasks.

8.6 Summary and Discussion

In this chapter, MVERT is applied to mapping unknown environments with motion and sensing noise. In both large- and small-scale mapping environments, MVERT produces low-error, low-uncertainty maps of the targets. In comparison to the individual action selection approach, MVERT provides a higher chance of completely covering the space due to the automatic dispersion of robots selecting actions that maximize joint value. Individual action selection leads all robots to concentrate only on the most interesting areas, leaving other areas unexplored.

In comparison with coverage patterns, a mix of results is observed. Coverage patterns can guarantee coverage, provided robots do not become lost, while MVERT can only guarantee coverage if target spacing is less than the sensor range. In cases where all areas are of equal interest (such as environments with uniformly distributed targets), coverage patterns may perform slightly better, taking more total measurements in each area and moving directly to far areas to complete the mission slightly faster. This is most evident with large teams. However, in environments with non-uniform interest, MVERT allows robots to focus on target clusters, taking more measurements on these important regions, while coverage patterns may allow robots to remain in uninteresting areas, making no contribution. MVERT also provides flexibility for adding opportunistic tasks and combining multiple tasks, as demonstrated in Chapter 10. Thus, in completely unknown (an expected uniform) environments that require only mapping, coverage patterns may be preferable, while in non-uniform environments or multi-task mission, MVERT can provide advantages.

Chapter 9 Dynamic Target Tracking Mission

The application of MVERT to the Dynamic Target Tracking Mission is designed to address the following research questions:

9. Can MVERT be applied to effective tracking of moving targets?
10. Can tracking moving targets be accomplished with a simple value function for uncertainty?
11. How does noise affect MVERT tracking performance?

9.1 Experimental Summary

In the Dynamic Target Tracking Mission it is assumed that robots are operating in an environment with known landmarks for localization. Within this environment, dynamic objects (which can be uniquely identified) move; the motions of these objects are not known a priori by the observing robot team. Dynamic target tracking is evaluated both in simulation and on the physical multi-robot system. Experiments are designed to investigate tracking behavior in isolation as well as in more realistic noisy situations. The experimental series for the Dynamic Target Tracking Mission are:

D1. MVERT Target Tracking in Simulation without Noise;

D2. MVERT Target Tracking in Simulation with Noise;

D3. MVERT Target Tracking in a Physical System.

Experiments are run in three environments:

- large simulation: up to 6 robots and 4 targets in a 20 by 20 meter area,
- small simulation: up to 4 robots and up to 4 targets in a 5 by 5 meter area,
- physical: up to 3 robots and up to 3 targets in a 5 by 3 meter area.

Two types of dynamic targets are investigated:

- random : targets move a fixed number of steps and then turn randomly,
- directed: targets follow a predefined path.

In each of these experiments, the value function for target location, V_{TL}, Eq. 2 (Section 3.3.1), is used. Dynamics are not predicted, but accommodated by increasing the uncertainty in the target locations at each step by 0.3 meters (approximately one target step). The first environment is applied to series D1, and the robots have perfect motion and perfect sensing in order to evaluate behavior of MVERT in isolation. The second environment is applied to series D2. For this series, robots localize on 6 fixed, known landmarks using Probabilistic Constraint-Based Localization (Chapter 6). Robot motions and robot sensing are noisy, with models based on the Aibo, as shown in 5.4. The third environment, for series D3, uses the same landmark configuration and motion and sensing models for the physical robot system in series D2.

9.2 Target Tracking Simulation without Noise (D1)

The simulations without noise are designed to investigate the basic, desired tracking behavior induced by MVERT and to address research questions 9 and 10. The targets are allowed to move over long distances in order to demonstrate long-term behavior. Additionally, three different speeds of targets (relative to the speed of the robots) are investigated to determine the impact on tracking behavior. D1 experiments are:

A. 2, 3, 4, and 6 robots observing four targets moving at one-third robot speed,

B. 2, 3, 4, and 6 robots observing four targets moving at one-sixth robot speed,

C. 2, 3, 4, and 6 robots observing four targets moving at 0.80 times robot speed,

D. 2, 3, 4, and 6 robots observing four targets moving at 0.95 times robot speed,

E. Experiment A with individual action selection.

In this set of experiments, the targets follow a directed trajectory at the specified speed. These experiments were completed using the sensor model presented in 5.4, with standard deviations of 0.1r and 0.5º for range and bearing, respectively. Range is infinite and angular field of view is 360º. Step size is 0.25 meters and candidate move resolution is 5º.

Example tracking trajectories are shown in Figure 80 – Figure 82.

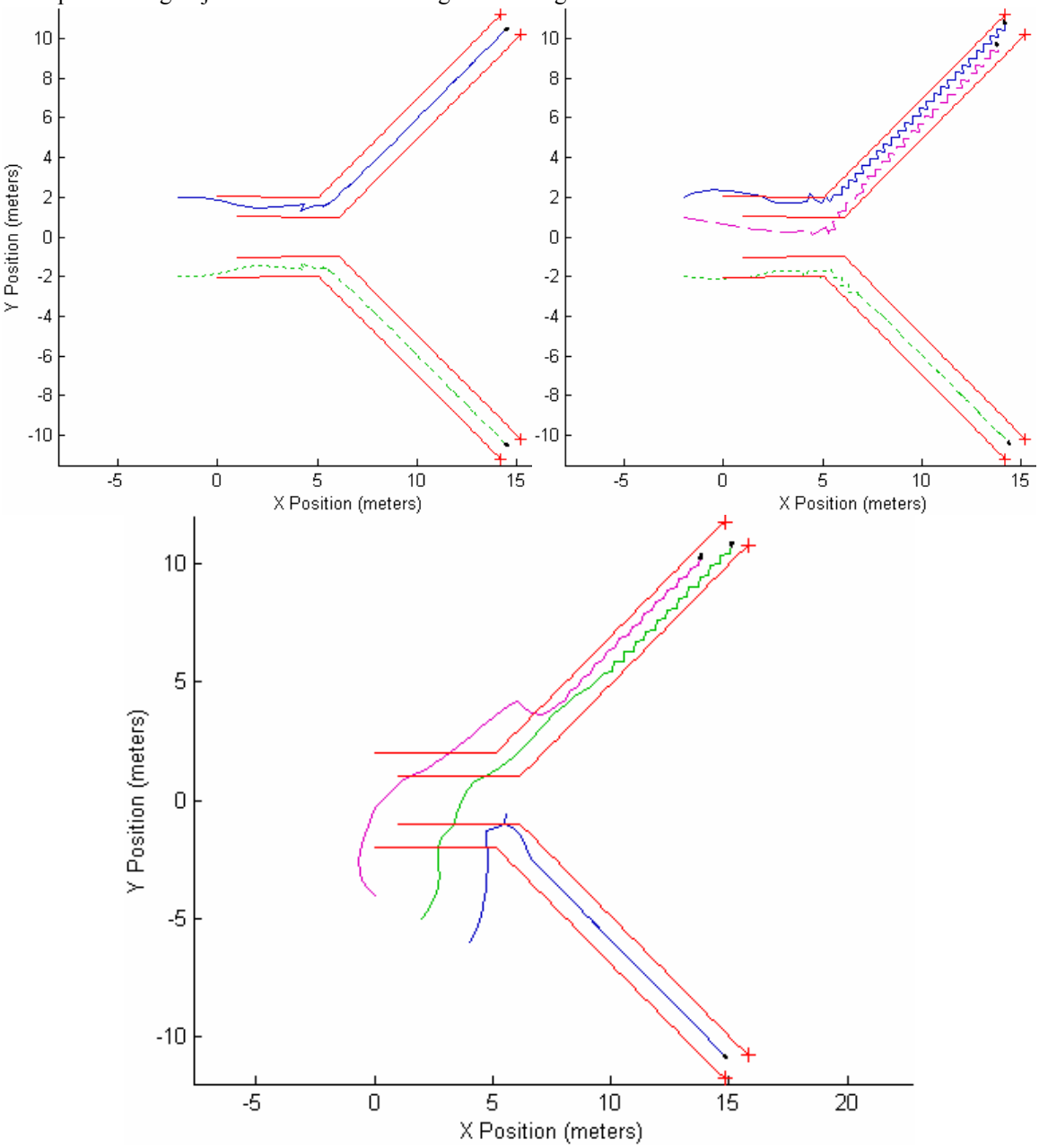

Figure 80. Experiment A, D1: Target Tracking Trajectories
Robots observe from multiple angles while targets are grouped, then distribute as much as possible among targets as needed. Robots following two targets remain balanced between the two. Robots oscillate behind single targets obtaining varied views to optimize value.

The targets follow piecewise linear trajectories. For 5 meters, they move linearly in a fixed formation. After 5 meters, targets split into two divergent groups, each maintaining their part of the fixed formation. While targets are grouped, robots maximize value by arcing around and taking complementary views. Robots then distribute among the two groups of targets. In the two-robot case, the robots move directly between the two nearest targets. In the four-robot case, the robots arc more, placing one robot on either side of each target group. In the three robot case, there is one occurrence of each of the previous types of approaches. In the six-robot case, robots arc less since multiple views are available from teammates.

Once the targets split into two groups, robots maximize value by choosing to follow one or two targets, as needed. In the two-target case, each robot is responsible for two targets, and stays between the two to

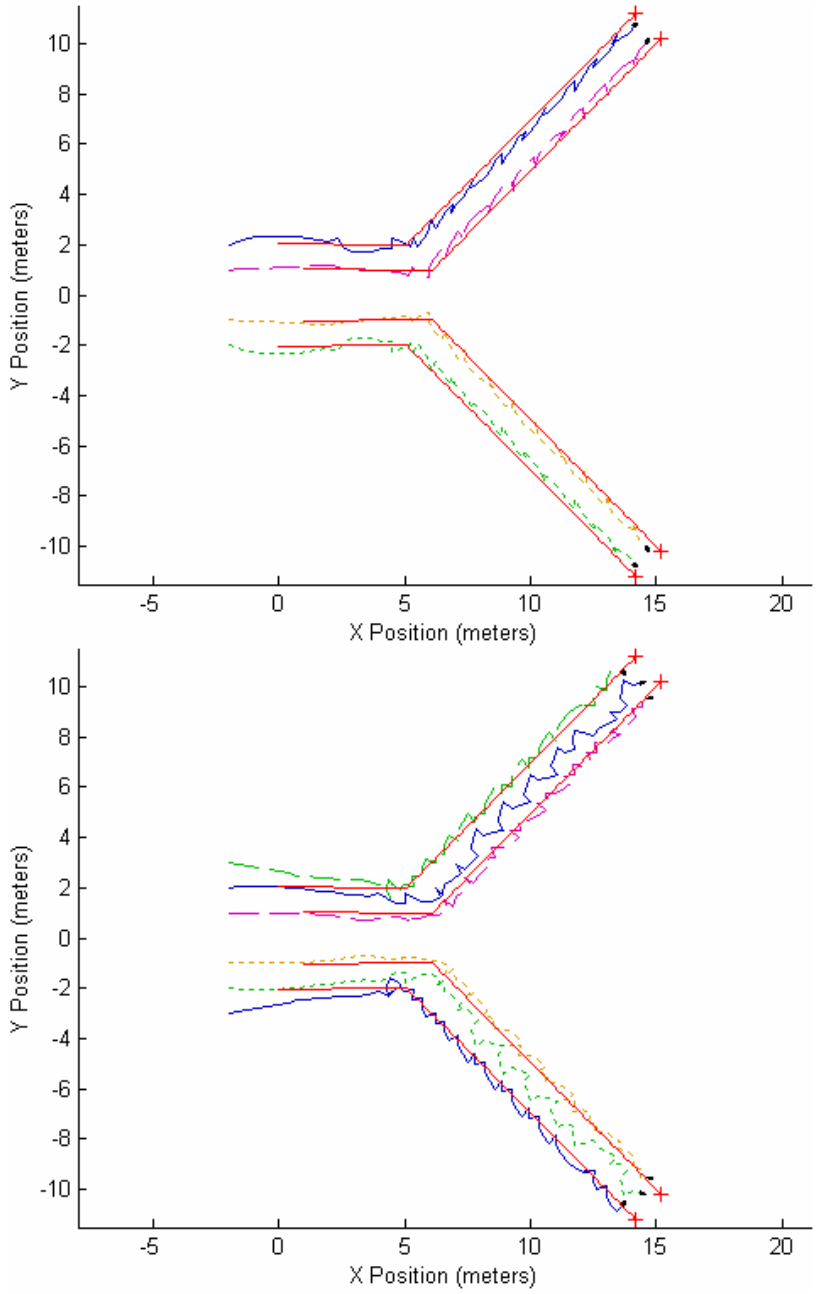

Figure 80 (Cont). Experiment A, D1: Target Tracking Trajectories
Once targets diverge, robots divide into two groups to follow. Robots oscillate behind single targets obtaining varied views to optimize value, while the third robot moves between the two targets to add maximum value.

obtain the closest (best quality) measurements on each target at each step. In the four-robot case, each robot specializes on one target, while also sharing observations on other targets. When robots specialize on a single target, they oscillate behind the target to obtain the multiple viewpoints that will maximize value. The oscillation seems biased toward remaining between the two targets, so that good additional measurements can be obtained on the other target. In the six-robot case, four of the robots specialize in individual targets, oscillating behind them to obtain multiple points of view, reducing uncertainty along multiple axes. The oscillation is balanced around the target to avoid and complement the third robot. The third robot in each group balances between the two targets, moving between them to obtain multiple points of view and close-up measurements on both targets alternately.

If the speed of the targets is below that of the robot, robots following two targets (in the two-robot and three-robot cases) remain balanced between the targets, and prevent over-taking the targets by occasionally backtracking. Robots following a single target oscillate more tightly to prevent moving away from the targets, and only occasionally backtrack. This is illustrated by the three-robot example in Figure 81.

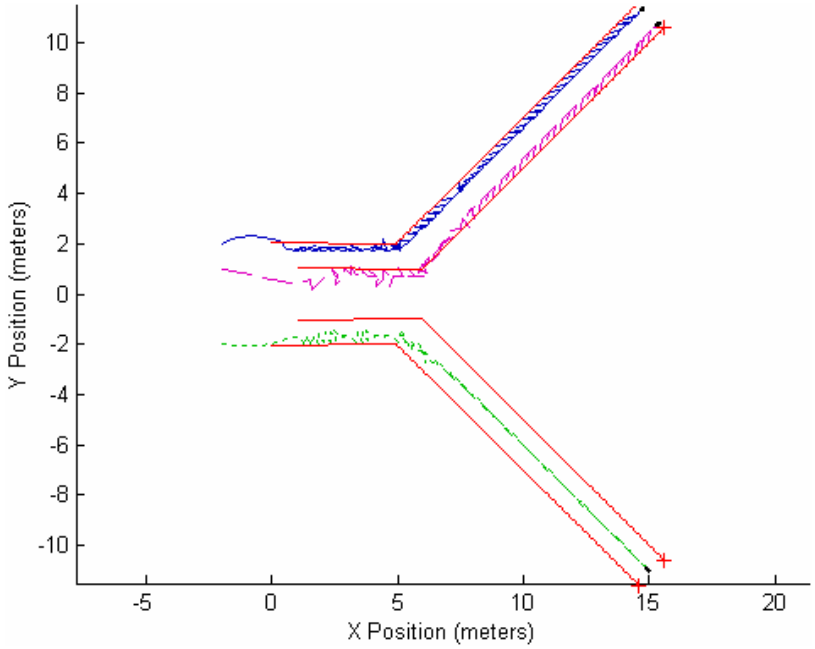

Figure 81. Experiment B, D1: Tracking Slow Targets

To prevent overtaking slow targets, robots oscillate more tightly and occasionally backtrack. Single robots following multiple targets oscillate less to remain close to both targets, but do backtrack.

If the targets move faster, the robots maximize value by remaining closer to the targets. As the relative speed of targets increases, oscillating for multiple views becomes less valuable than remaining close to the targets. In the example six-robot case, illustrated in Figure 82, the third robot now remains to one side of the targets rather than falling behind by moving laterally to be between targets.

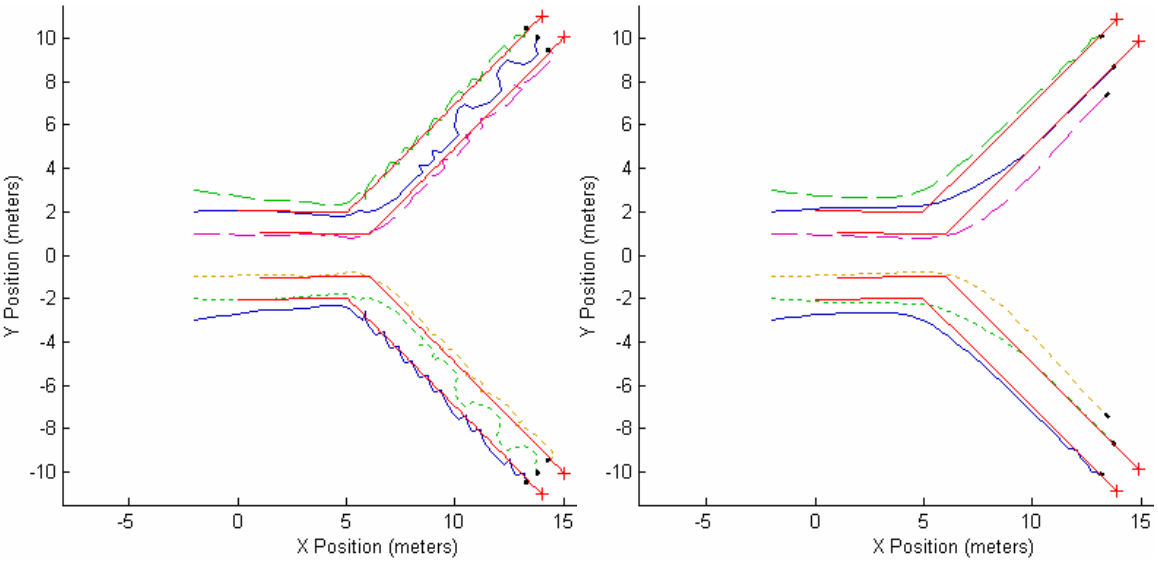

Figure 82. Experiments C and D, D1: Tracking Fast Targets

Left: Targets at 0.80 times robot speed. *Right:* Targets at 0.95 times robot speed. As target speed increases, robots cannot afford to oscillate for multiple views or they will fall behind, reducing observation quality. Eventually oscillation is eliminated entirely.

These experiments are used in answering questions 9 and 10 regarding tracking ability. MVERT can use a simple value function for reducing target uncertainty in order to maintain observation of multiple moving targets. Robots take advantage of teammates by obtaining multiple points of view to maximize value (minimize uncertainty) when possible, and are automatically distributed among targets when targets diverge. In contrast, if robots use individual action selection, each robot attempts to simultaneously minimize the uncertainty on all targets only from its own measurements. Thus, robots tend to remain centrally located rather than closely following targets. An example is shown in for the three-robot case. Jagged motions, similar to the oscillation demonstrated by MVERT to obtain multiple points of view, are in this case due to robots attempting to move into the same central path, which necessitates collision avoidance. In this case, robots can maintain observation of all targets; however, if the robots were to travel in more opposite directions, individual action selection (remaining central) would fail to follow any targets. Final value is -0.25 compared to -0.0033 for MVERT.

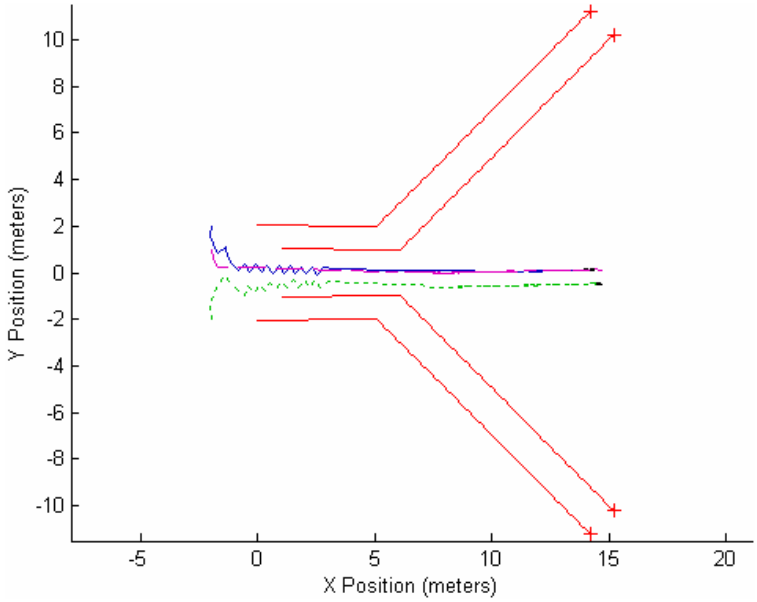

Figure 83. Experiment E, D1: Tracking With Individual Action Selection
Robots must attempt to maintain observation of all targets simultaneously, and thus remain centrally located rather than closely following any target.

If more accurate tracking of targets is required, a Kalman-Bucy filter could be implemented for each target, using observations from all robots to update at each step. This filter can be used not only to improve the tracked target location, but can be used to improve action selection by accounting for future target motion in a more accurate manner than growing uncertainty at each step.

9.3 Target Tracking Simulation with Sensor Noise (D2)

This series of experiments is designed to illustrate how MVERT may be applied to tracking multiple moving objects using robots with noisy motions and sensing, addressing question 11. Environments are known in that there are landmarks at known locations for localization. The Experiments for series D2 are:

F. 2, 3, and 4 robots observing two targets moving on three directed trajectory sets,
G. 2, 3, and 4 robots observing three targets moving on a directed trajectory set,
H. 3, 4, and 6 robots observing four targets moving on a directed trajectory set,
I. Experiment F without noise,
J. Experiment G without noise,
K. 2, 3, and 4 robots observing two targets moving on a random trajectory set,
L. 2, 3, and 4 robots observing three targets moving on a random trajectory set,
M. 3, 4, and 6 robots observing four targets moving on a random trajectory set.

These experiments are performed using the Sony quadruped sensor and motion models as described in 5.4, including limited vision range and angular field of view. 'Step size is 0.25 meters and candidate move resolution is 5º. Experiments I and J are added for qualitative comparison to ideal behavior as illustrated in 9.2 without noise.

Examples of the trajectories for tracking targets on directed trajectories are shown in Figure 84 -Figure 87.

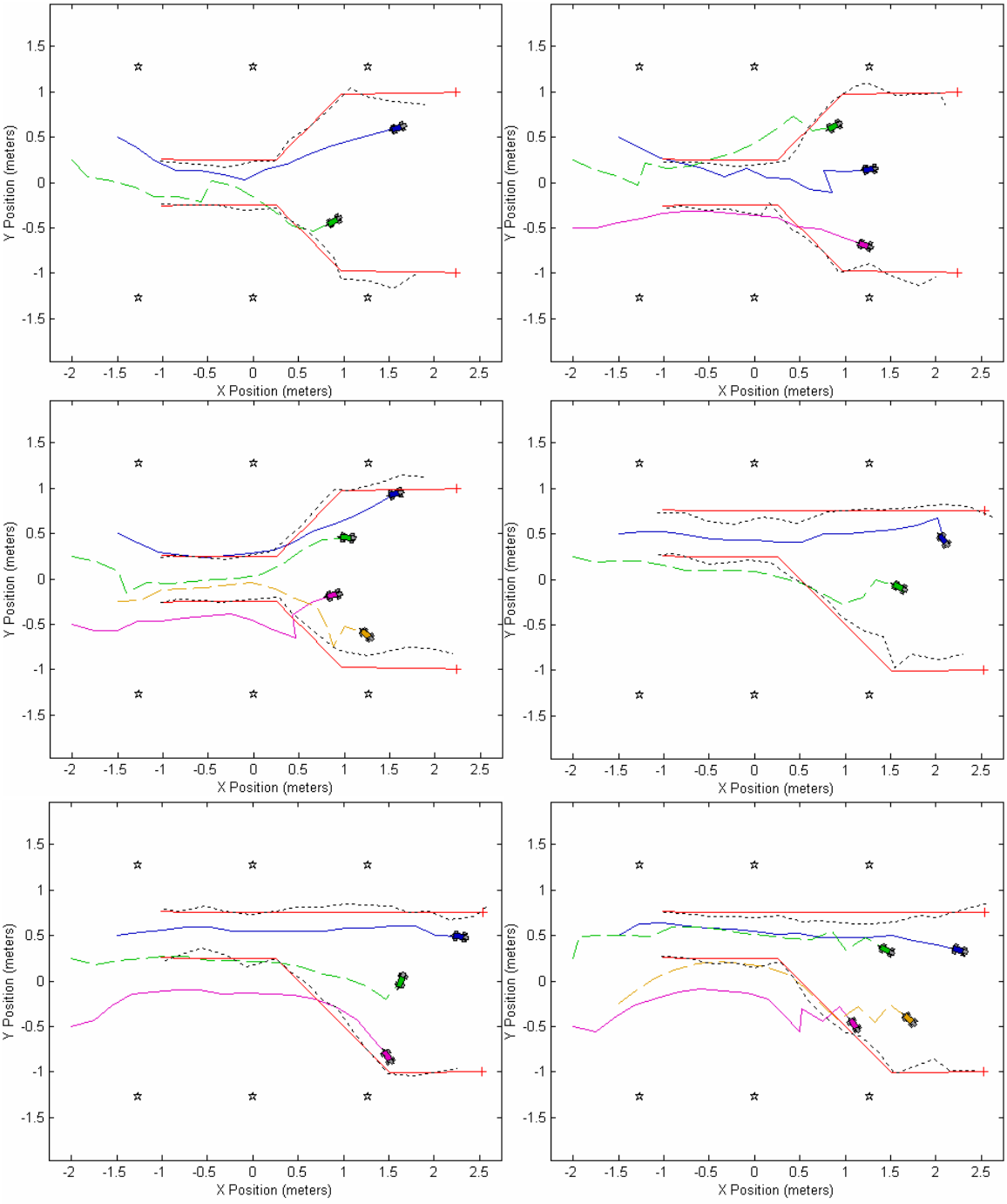

Figure 84. Experiment F, D2: Tracking Two Targets

Top, Center Left: Configuration 1. *Center Right, Bottom:* Configuration 2. Robots tend to distribute themselves evenly among targets, but position themselves to observe the other target as closely as possible and contribute most to the joint knowledge. Actual target trajectories are shown in solid red; estimated trajectories are shown in dotted black. Landmarks are shown as stars.

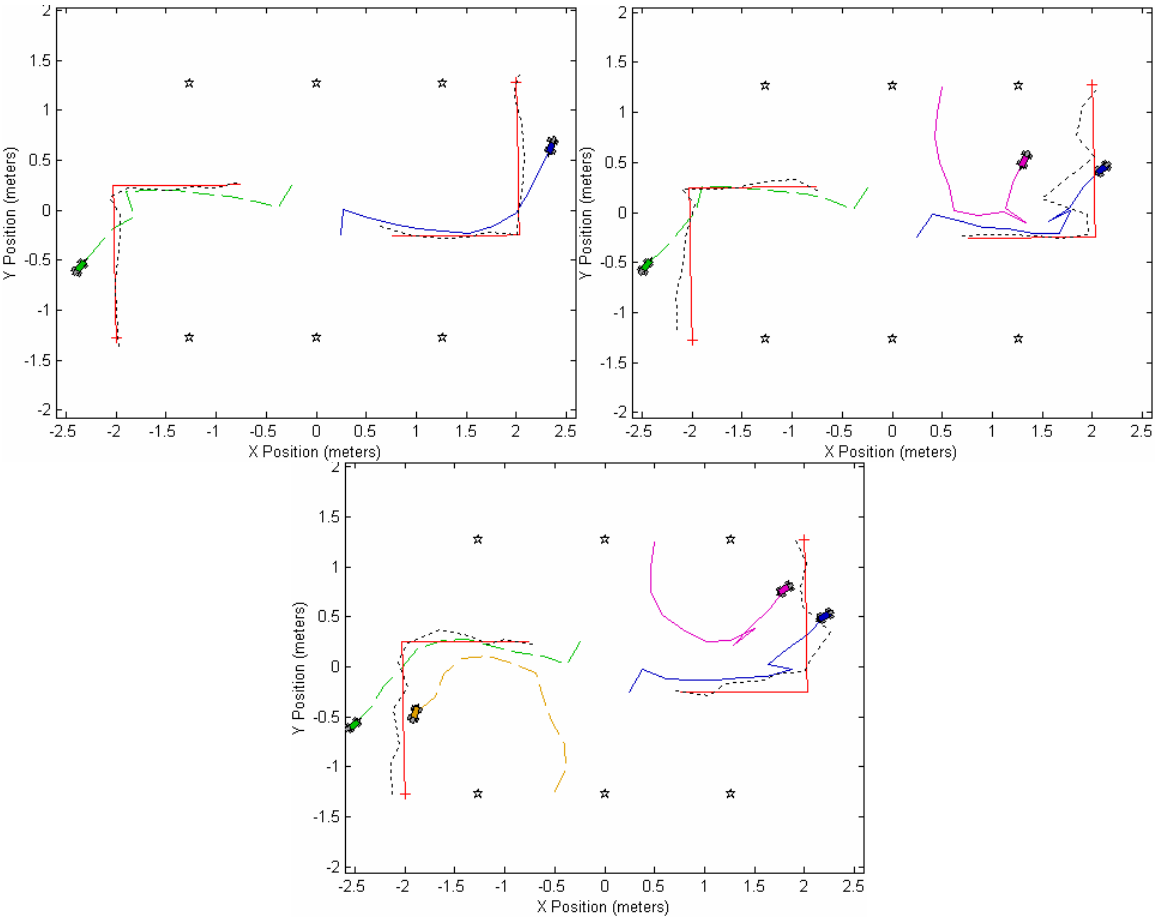

Figure 84 (Cont). Experiment F, D2: Tracking Two Targets
Configuration 3. Robots turn around to follow the closer targets, despite not seeing them initially.

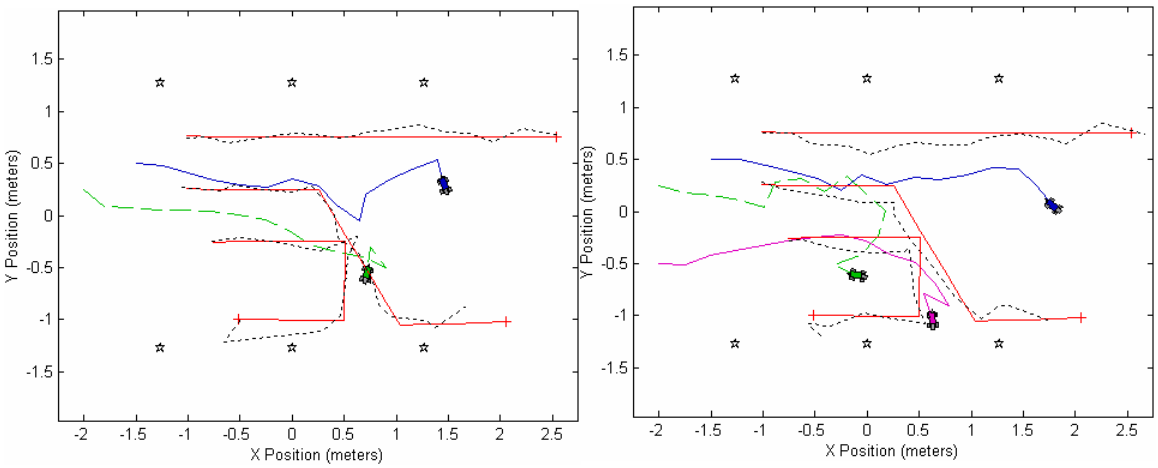

Figure 85. Experiment G, D2: Tracking Three Targets
Again, robots tend to distribute themselves evenly among targets, but position themselves to observe the other target as closely as possible and contribute most to the joint knowledge (as with the middle robot, left, which remains between the two diverging targets).

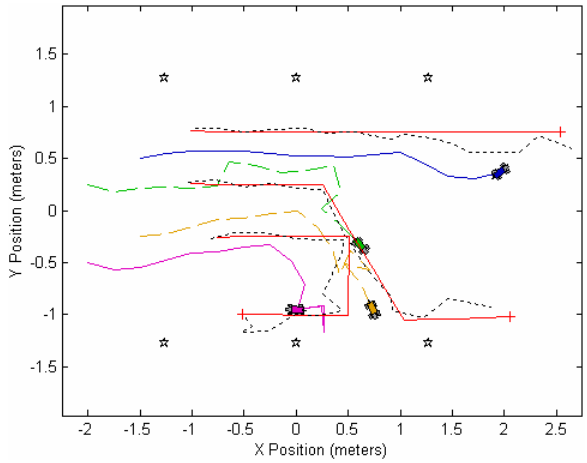

Figure 85 (cont): Experiment G, D2: Tracking Three Targets
Robots distribute among targets and maximize utility of overlapping observations.

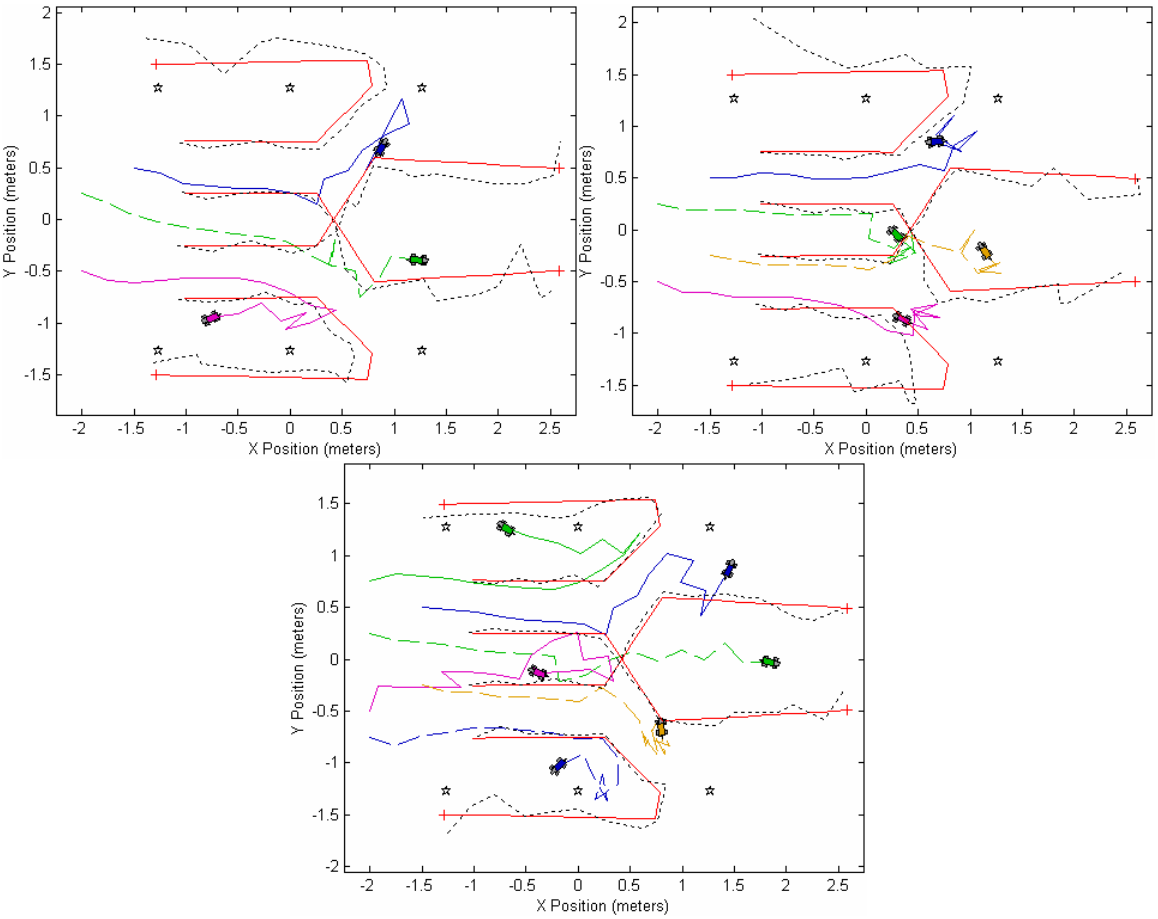

Figure 86. Experiment H, D2: Tracking Four Targets
Again, robots tend to distribute themselves evenly among targets, but position themselves to observe the other target as closely as possible and contribute most to the joint knowledge (as with the middle robot, left, which remains between the two diverging targets). For three robots, two (bottom) follow individual targets, while the third tries to remain where it can see the other two. For four robots, all robots seem to try to remain where they can observe multiple targets to improve estimates.

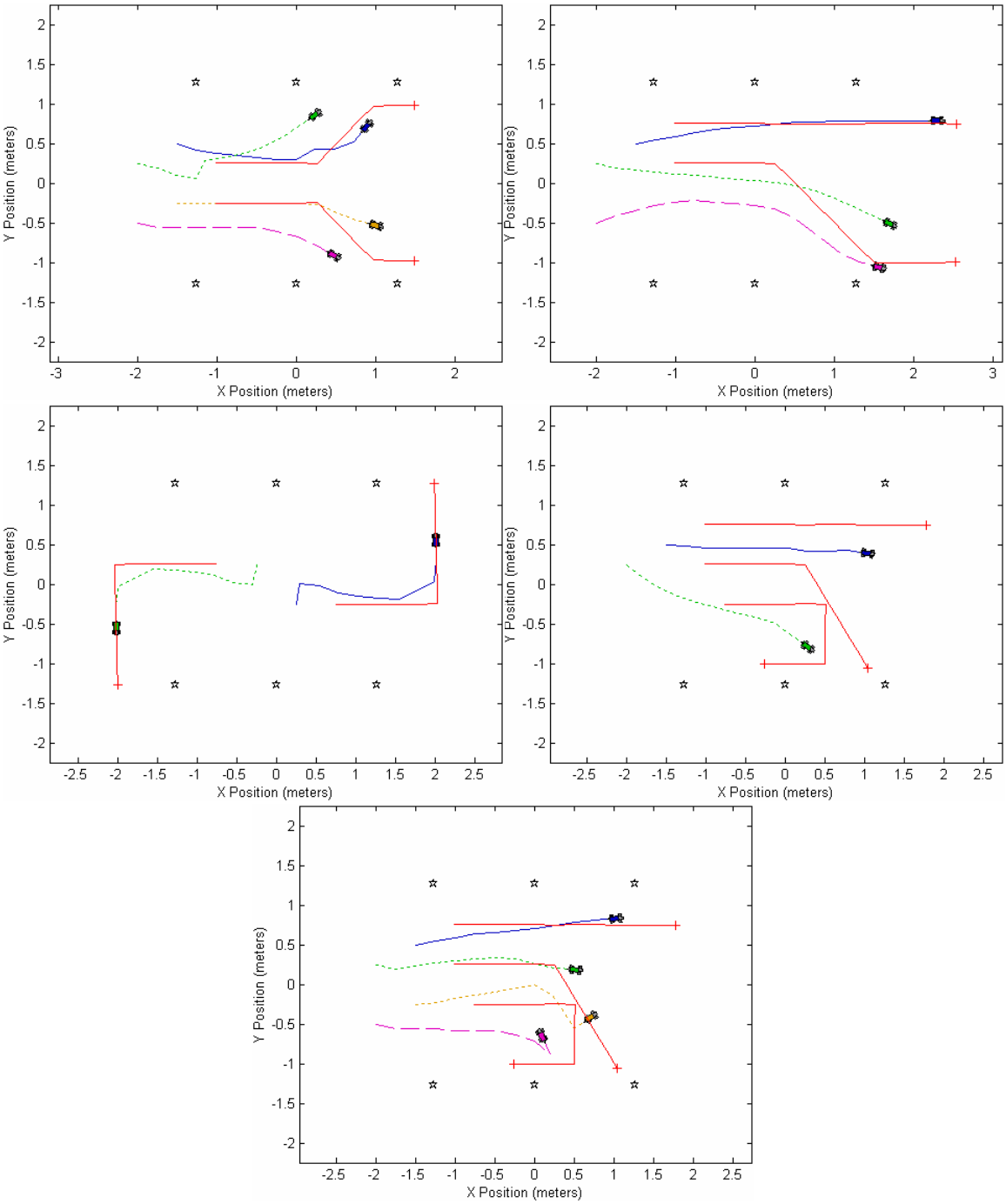

Figure 87. Experiments I and J, D2: Tracking Targets without Sensor Noise
Without noise, the distribution of robots is more symmetrical and specific targets are clearly followed, and the trajectories are smoother. Points at which robots are positioning themselves between targets to optimize multiple observations are also more clearly evident, as in the bottom figure, where the bottom three robots are oriented to observe multiple targets.

In the dynamic tracking experiments in this section, robots have limited sensing, both in range and in field of view. This alters the tracking behavior slightly from the case of unlimited sensing in the previous section. In order to share information, robots will divert from exactly following individual targets in order to be able to also observe adjacent targets and improve the joint information about target locations. This is

evident in many examples. For example, in the three-target configuration, robots tend to position themselves between targets, with an orientation that enables seeing multiple targets (see the central robot in the three-robot case, Figure 85). Other examples of robots changing heading to observe additional targets can be specifically seen in the four-target configuration, such as in the bottom right robot in the four-robot case of Figure 84, which has several turns in its path allow it to see the target above as well as the one it is following. The smaller bias in uncertainty in range versus bearing mostly eliminates oscillation for obtaining multiple points of view in favor of moving closer to targets and orienting to see multiple targets.

The target trajectories produced are noisy, since each step is based on only the previous belief (grown in uncertainty), and a few instantaneous measurements from the robots currently viewing the target. These few measurements are also noisy due to the high level of motion and sensing noise. As mentioned previously, the quality of the actual tracks could be improved by using a Kalman-Bucy filter instead of a simple Gaussian multiplication update. In some cases the simplistic update caused robots to lose sight of targets and continue to track only the others. This particularly occurred in the three-robot case with the target that turns back; in some cases the robot observing this target did not turn quickly enough to catch the change in direction and lost the target. This could often be avoided by using a tracking filter, but could also be aided by the addition of a value for searching for a target if it is lost. Thus, to answer questions 9 and 10 regarding tracking ability, MVERT can track moving targets in the presence of noise. To answer question 11, sensing and motion noise makes trajectories less smooth. The balancing behavior is observed in the noiseless trajectories as well, though the division of targets among robots is occasionally more defined.

The resulting trajectories following random targets are shown in Figure 88 and Figure 89.

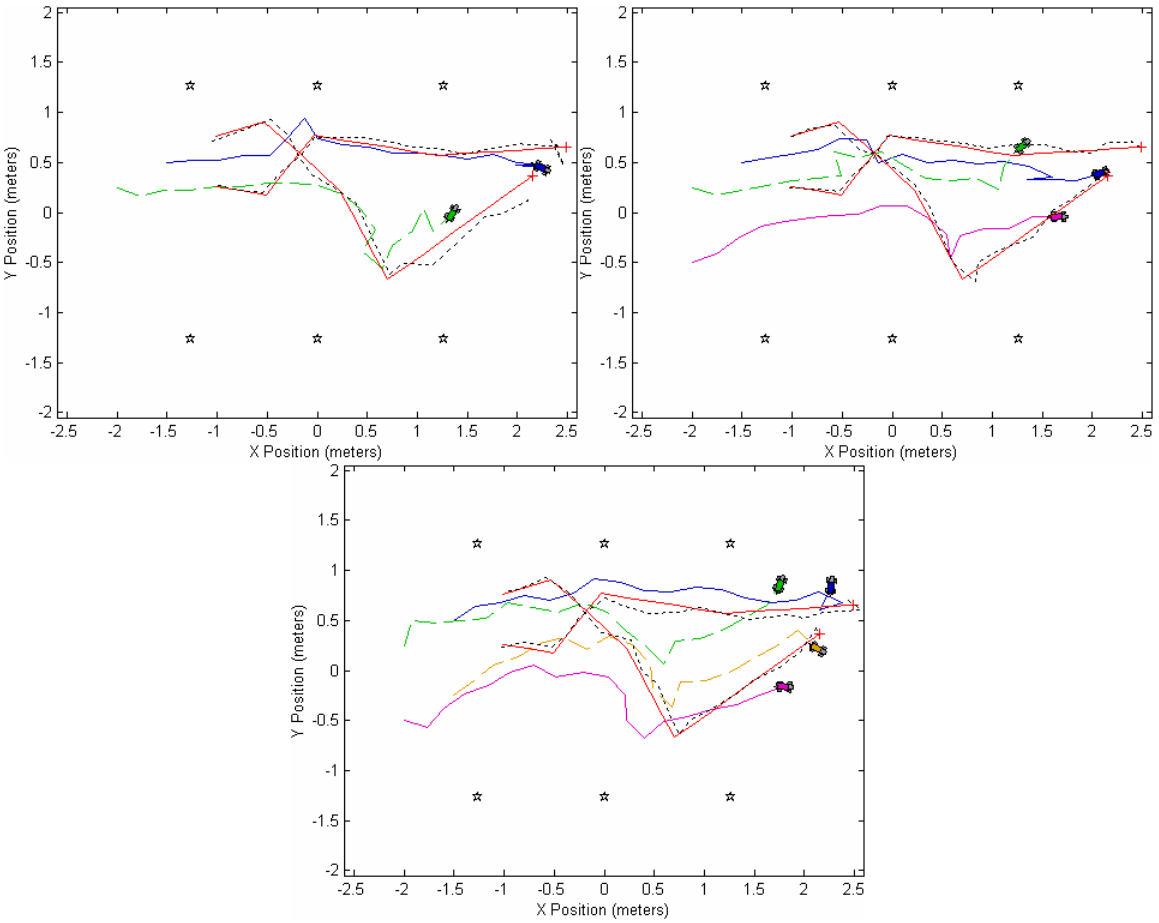

Figure 88. Experiment K, D2: Tracking Two Random Targets

As in the case with the more directed targets, robots distribute among targets. They change heading occasionally in order to ensure observing other targets to maximize value. Actual target trajectories are shown as solid red. Estimated trajectories are shown as dotted black.

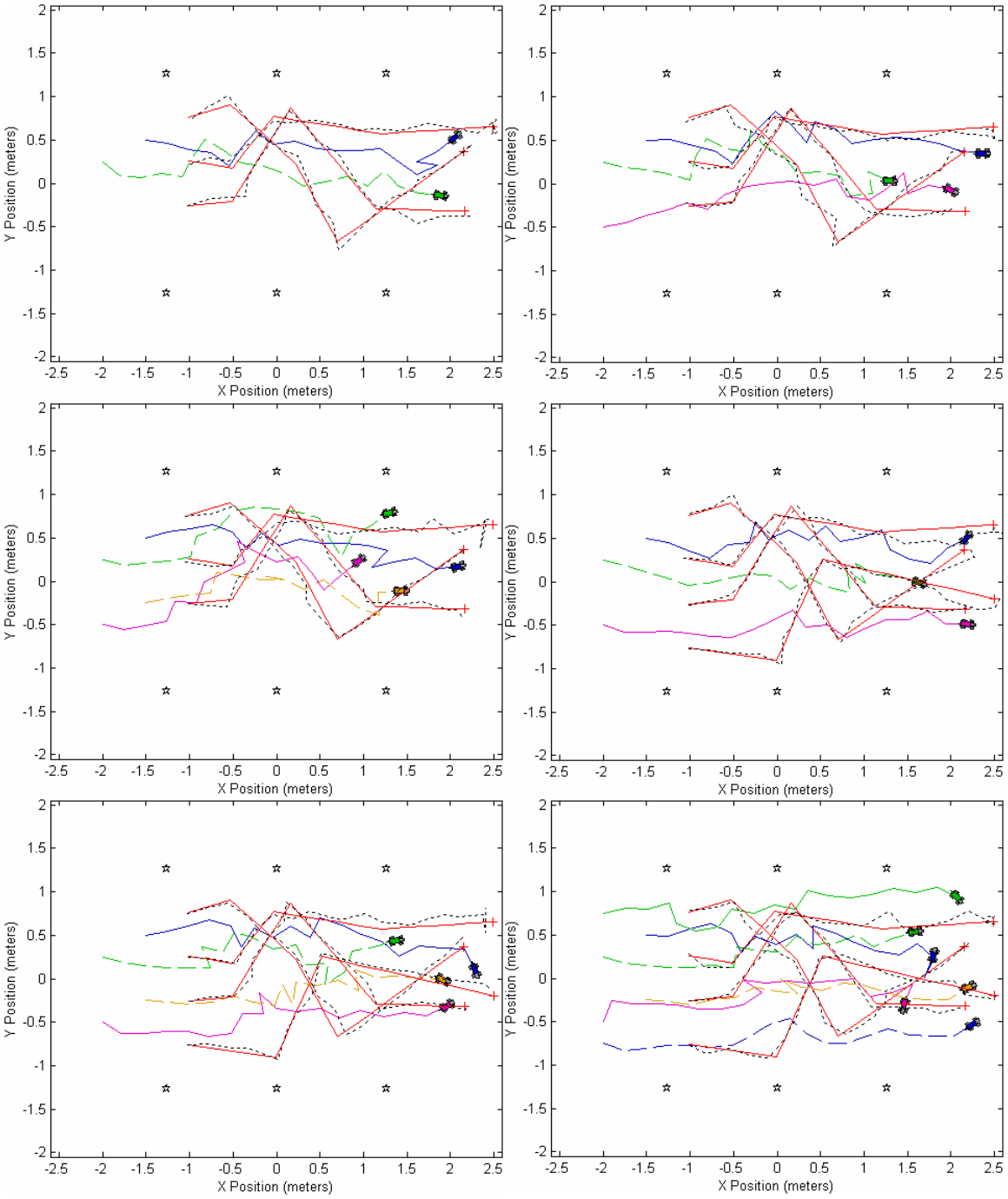

Figure 89. Experiments L and M, D2: Tracking Three and Four Random Targets
Top, Center Left: Tracking three targets. *Center Right, Bottom:* Tracking four targets. The same pattern emerges with larger numbers of targets. Robots specialize in a target as much as possible. Additionally, robots turn to observe additional targets to assist in lowering their uncertainty. With fewer robots than targets, robots balance more between targets when they cannot specialize.

Results with randomly moving targets support the same conclusions as those with directed targets. Robots specialize on a target when robots equal or outnumber targets. While following targets, they turn to observe additional targets to maximize value by additionally reducing their uncertainty. With fewer robots than targets, robots do more balancing between targets to reduce uncertainty by taking overlapping observations.

9.4 Dynamic Target Tracking in a Physical System (D3)

As in the simulation experiments, the physical system experiments are designed to evaluate how well MVERT can be applied to dynamic target tracking and to assess any benefits derived from predicting teammate contributions, addressing questions 9-11 regarding tracking behavior ability with and without noise. Additionally, these experiments illustrate how the MVERT system may be applied on a physical multi-robot system. The experiments for series D3 are:

N. 2 and 3 robots observing two targets in three and one directed trajectory sets, respectively,

O. 2 and 3 robots observing three targets in a directed trajectory set.

The trajectories of targets in the physical system experiments is a subset of those directed trajectories in the simulation environment. Again, the Aibo sensor and motion models are used (5.4) with a step size of 0.25 meters and a candidate move resolution of 5º.

Results from applying tracking to the physical multi-robot system are shown in Figure 90 and Figure 91. Actual target trajectories are shown as solid (red) lines. Estimated target trajectories are shown as dotted (black) lines.

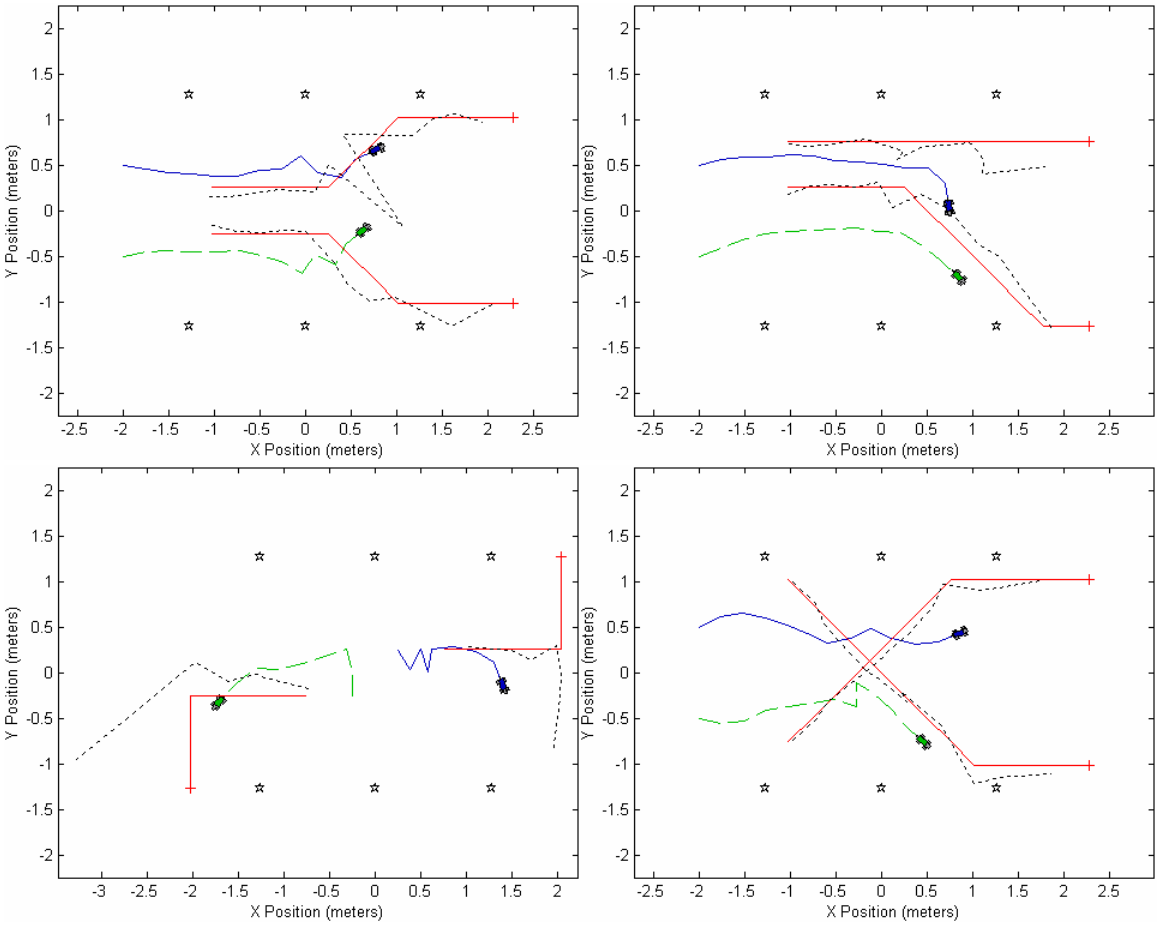

Figure 90. Experiments N, D3: Two Physical Robots Tracking Two Targets

Four configurations of two targets tracked by two robots. Actual target paths are shown as solid red; estimated are shown as dotted black. The two robots automatically are distributed among targets by choosing moves to maximize value. Robots are automatically set to switch targets if the proximity of targets changes, as in lower right. Occasional high levels of sensor noise lead to outliers, as shown in the left two figures.

The results of tracking with the physical multi-robot system are quite similar to those demonstrated in simulation. The high level of motion and sensing noise leads to occasionally outlying points along the trajectory, as in the middle target in the case of three robots following three targets (Figure 91, bottom right). However, these are typically corrected in the next time step. As in the simulations, robots can occasionally loose sight of a target since they are not predicting and have limited visual fields of view. This again is illustrated by the bottom target in the three-robot, three-target case (Figure 91, bottom right); the robots do not see the target turn and fail to continue tracking it. In the three-robot case, the team loses the target while in the two-robot case the team does not. This is due to the fact that the third robot, spread out from its teammates, is much closer to the line on which the target is moving, making the lateral field of view smaller than for the farther-away robot.

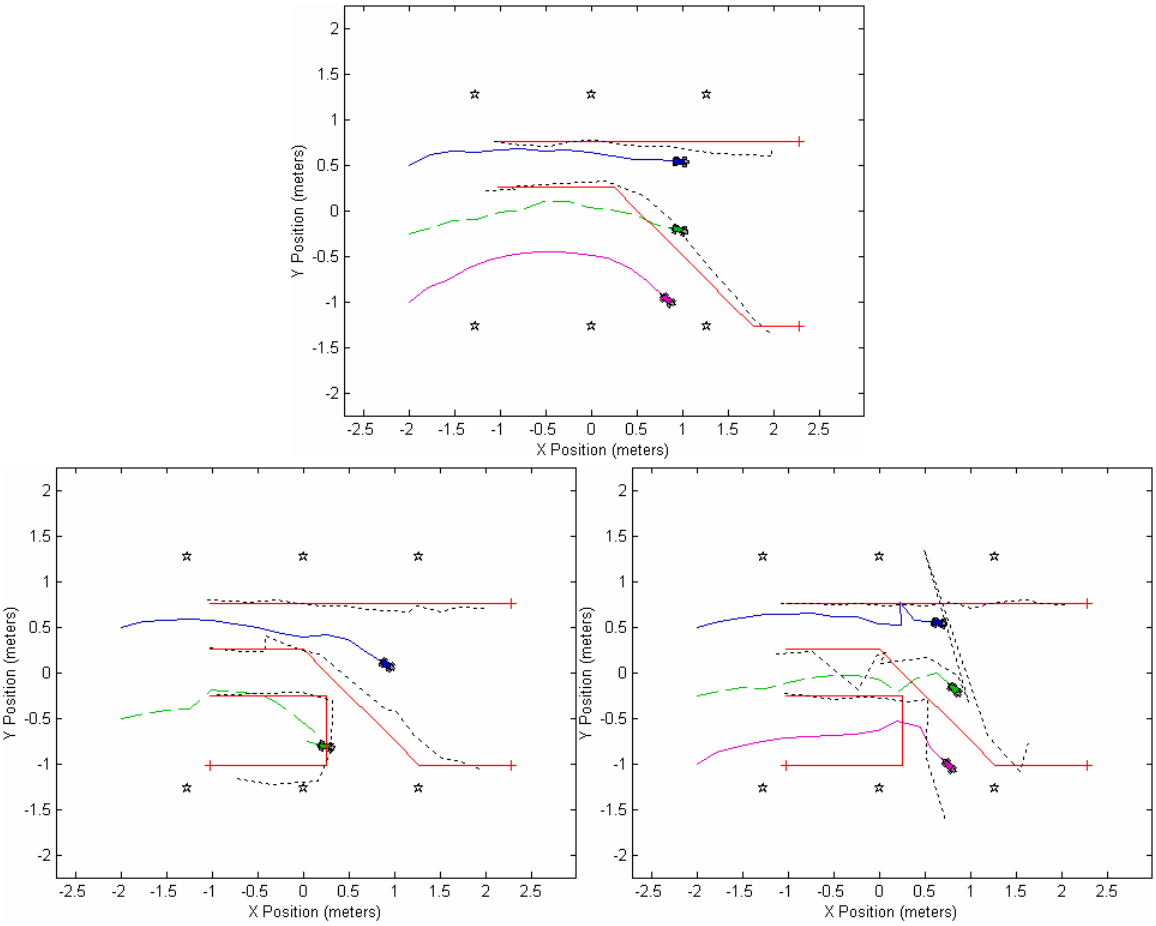

Figure 91. Experiments N and O, D3: Physical Robots Tracking Two and Three Targets
Top: Three robots tracking two targets. *Bottom Left:* Two robots tracking three targets. *Bottom Right:* Three robots tracking three targets. Robots automatically are distributed among targets. High levels of noise can lead to outliers (bottom right, middle target) and limited sensing can lead to robots losing track of targets (bottom right, bottom target)

Based on these experimental results, it can be concluded that MVERT can be applied to successfully tracking targets in a real multi-robot system. However, high levels of sensing and motion noise, such as that present in the Aibo, can lead to occasional large measurement errors which can produce large tracking errors in the tracked path of the targets with the simple update applied in this implementation. This noisy performance could likely be improved with the addition of a tracking filter (such as a Kalman-Bucy filter). The simple method of updating positions and limited sensing can also lead to targets being lost. This failure in performance could be improved by instantiating a search strategy to reacquire lost targets.

A snapshot of a dynamic target tracking experiment with two robots and three dynamic targets is shown in Figure 92.

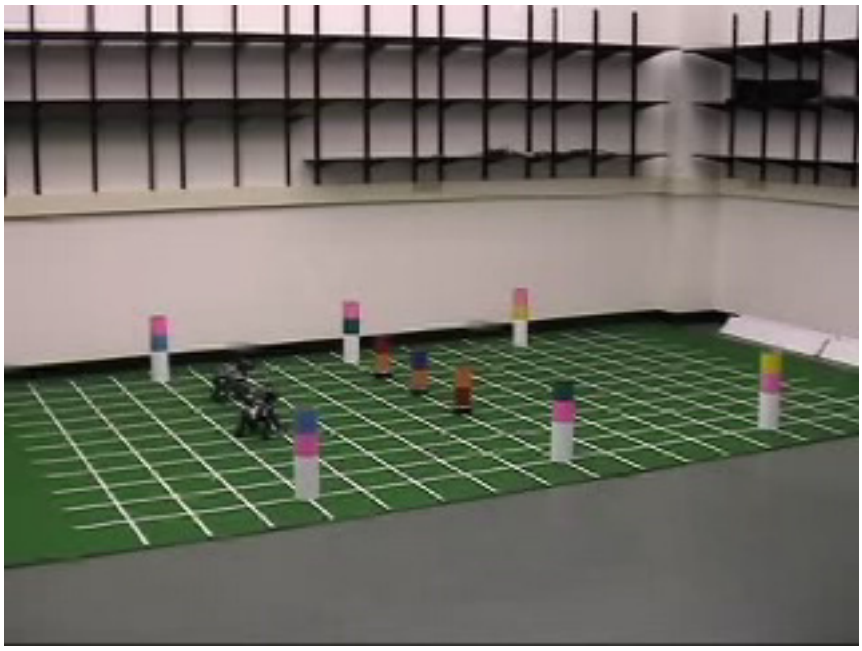

Figure 92. Experiment N, D3: Two Physical Robots Tracking Two Targets
A snapshot from an experiment with two robots tracking three dynamic targets. At this time, the targets are moving together, and thus the robots follow together, moving to remain close to the targets and, when possible, obtain complementary views.

9.5 Summary and Discussion

Using a simple value function for reducing uncertainty with no anticipation of future target motion, MVERT can provide some basic target tracking functionality. Robots position themselves at each step to attempt to observe all the targets and to obtain overlapping observations when possible. When targets outnumber robots, robots must balance observations, particularly when targets diverge. If targets diverge, splitting into more subgroups than robots, robots will maximize value by rearranging subteams if they can still see targets in danger of being lost and contribute to observing them. The one-step look-ahead with no prediction, however, does leave MVERT open to losing targets and would benefit from a tracking algorithm (such as an EKF).

Chapter 10 Planetary Exploration Mission

The purpose of experiments in the Planetary Exploration Mission is to evaluate MVERT performance in the context of a more complex mission scenario. The following research questions are specifically addressed:

12. Can MVERT be applied to selecting actions for robots with multiple competing mission tasks?
13. Can MVERT be applied to selecting actions for heterogeneous teams?
14. Can MBERT behavior be adjusted to reflect mission goals by changing value weights?

The Planetary Exploration Mission combines multiple mission goals (tasks) and is designed to reflect mission needs expressed for future exploration of Mars (as illustrated in Figure 1). The tasks considered include complete exploration a specified region, mapping other types of targets in the region (which can be used as landmarks for localization), performing sampling/analysis tasks at pre-defined targets, and maintaining line of sight (such as for maintaining communications). These tasks are completed in an unknown environment. Application of MVERT to this type of task is intended to evaluate MVERT's ability to select contextually appropriate actions in a more complex scenario. The goal of conducting the planetary exploration experiments is to qualitatively demonstrate that MVERT can be applied in these more complex multi-task missions. Thus, detailed quantitative analysis of results is not performed.

To combine the needs of completing multiple tasks in order to select an action, each robot uses MVERT to compute an overall value to each candidate action. This overall value is a weighted sum of the individual values for each task. The overall value is computed with respect to expected teammate contributions in all of the individual tasks. Thus, the select action is the one that makes the most progress toward all of the mission goals, with prioritization. Details of these value functions are presented in 3.3.

10.1 Experimental Summary

The planetary exploration task combines multiple mission goals in a search scenario. Two experimental series were run for this task:

P1. MVERT Small-Scale Planetary Exploration Task;

P2. MVERT Large-Scale Planetary Exploration Task.

The experiments are run in a two simulated environments:

- small: 2 robots, 6 landmarks, and 3 target sampling tasks in a 3 by 4 meter area,
- large: 6 robots, 55 landmarks, 20 target sampling tasks, and 5 obstacles in a 300 by 300 meter area.

In the small environment, only mapping and target sampling tasks are combined and simplifications are made in terms of noise and update model. This subset of the planetary exploration mission goals is examined in detail to investigate tradeoffs between tasks. In the large environment, mapping, target sampling, communication, and exploration tasks are considered. Obstacles can occlude observation and communication. Sensing and motions are noisy. Occlusion for both is determined using Eq. 11, Section 3.3.3. In both cases, the candidate move resolution is 5º.

10.2 Small-Scale MVERT Planetary Exploration (P1)

In the small-scale environment, landmark target location uncertainty is modeled, but noise is not added to the measurements or motions. This provides a baseline for MVERT with respect to research question 12. The value functions used are the target location value (Eq. 2, Section 3.3.1) and the target sampling value (Eq. 5 and Eq. 6, Section 3.3.2). Additionally, affects due to changing the weights of values when they are combined into an overall value are explored to study question 14, regarding tuning behavior with these weights. κ_{VS}, the weight on target sampling value, varies from 1.0 to 0.0000001 while κ_{VTL}, the weight on target location (mapping), remains fixed at 1.0. Robots have infinite vision range and a 360º angular field of view. Updates on landmark target position estimates are Gaussian multiplication. The sensor model is bearing-biased, with σ_r=0.1r and σ_ϕ=0.5º. The step size is 0.254 meters. The buffer (distance within which robots must be to sample) for target sampling is 0.3 meters. The experiments run for this series are:

A. 2 robots exploring the small environment with 6 sets of relative task value weights.

Trajectories produced for the small-scale planetary exploration task are shown in Figure 93. Quantitative results are summarized in Table 29. For this table, two mission goals are set: to complete the sampling of all sample targets and to reduce maximum map uncertainty (E_2) to less than 5.0×10^{-6} (arbitrarily chosen).

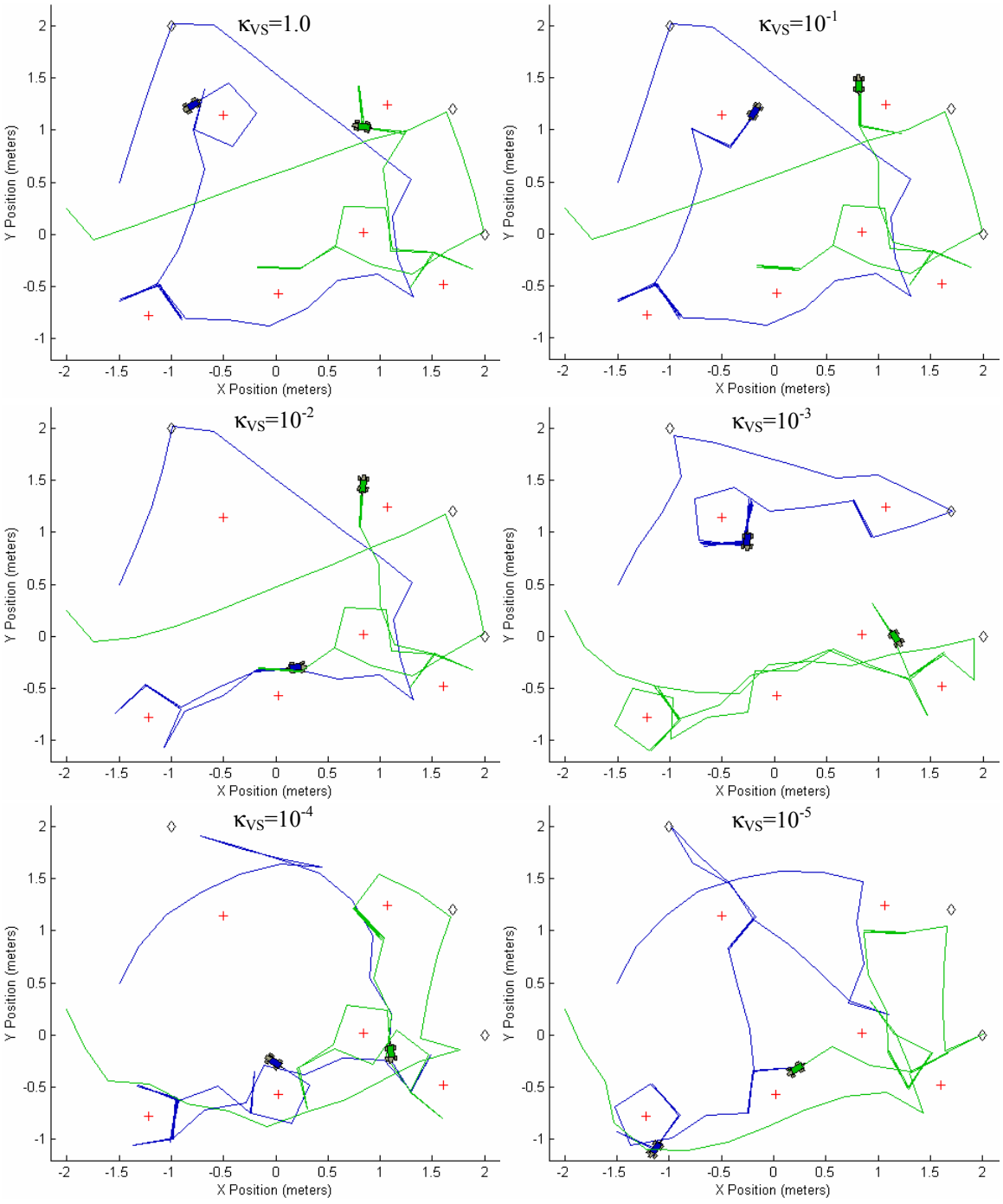

Figure 93. Experiment A, P1: Small-Scale Planetary Exploration

Top Left: $\kappa_{VS}=1.0$. Top *Right:* $\kappa_{VS}=10^{-1}$. *Middle Left:* $\kappa_{VS}=10^{-2}$. *Middle Right:* $\kappa_{VS}=10^{-3}$. *Bottom Left:* $\kappa_{VS}=10^{-4}$. *Bottom Right:* $\kappa_{VS}=10^{-5}$. ◊ represents a target sampling location. With a high weight on target sampling tasks (top), robots move quickly to these locations and only then start to map landmarks. With slightly more weight on mapping (middle), the nearly straight paths to targets bend somewhat to get closer measurements on targets. As weight for mapping increases (bottom), robots take more indirect paths to improve map before reaching targets.

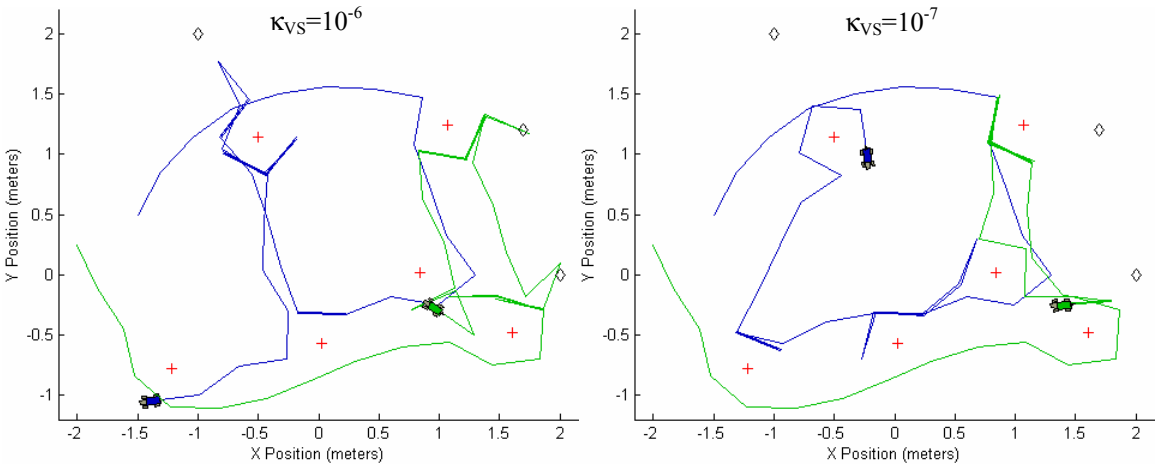

Figure 93 (Cont). Experiment A, P1: Small-Scale Planetary Exploration
Bottom Left: $\kappa_{VS}=10^{-6}$. *Bottom Right:* $\kappa_{VS}=10^{-7}$. ◊ represents a target sampling location. As weight on target sampling drops, robots reach sampling targets only at the end of the mission, after map uncertainty has been reduced (left). Ultimately, robots do not reach sampling targets, instead they continue to reduce map uncertainty (right).

Table 29. Mission Performance in Small-Scale Planetary Exploration

Sampling Weight	Sampling Complete	Map $E_2 < 5x10^{-6}$	Mission Complete
10^{0}	18	30	30
10^{-1}	18	29	39
10^{-2}	18	29	39
10^{-3}	16	25	25
10^{-4}	19	27	27
10^{-5}	23	16	23
10^{-6}	35	16	35
10^{-7}	--	12	--

Table 29 indicates that the maximum performance in terms of minimum time to completing both mission goals occurs with a mixed weight of 10^{-5}. In contrast, minimum sampling time occurs at a weight of 10^{-3}. This is better performance than for a weight of 1 because the greedy sampling task allocation in this case falls into a less optimal sampling task allocation among the robots that leads to a delay in sampling the final target. The minimum mapping time occurs for a sampling weight of 10^{-7}.

The results of these experiments illustrate that mission performance can be tuned using the relative weights of the individual value functions, answering question 14. With almost all weight on target sampling tasks, robots move to improve the map only after all the target sampling tasks have been completed (Figure 93, left). Rather than distributing to make measurements on the landmarks, they move on direct paths toward the selected target sampling task location. Conversely, with almost all weight on mapping, robots do not choose to divert from the desired mapping trajectory to visit the sampling targets (Figure 93 Cont, bottom right). With weights in between, robots place different emphasis on visiting sampling targets versus mapping. With a strong bias toward target sampling tasks (Figure 93 Cont, top left), robots are pulled slightly toward landmark targets while approaching target sampling task locations. This results in slightly better measurements on landmark targets without much delay in sampling. Once all the target sampling tasks are complete, the robots specifically move to improve map quality. With slightly more emphasis on mapping (Figure 93 cont, top right), the robots disrupt their paths to the target sampling tasks more but still

focus on achieving the target sampling tasks. As the weight on mapping overwhelms the weight on target sampling tasks (Figure 93 cont, middle and bottom left), robots begin moving to produce a high-quality map. They divert toward target sampling tasks later and later in the process, at the point where value for improvements in uncertainty no longer override value for moving toward the target sampling task locations. Eventually (Figure 93, bottom right), map improvement value dominates target sampling task value for the duration of the experiment, and robots never complete the target sampling tasks. When mapping, robots exhibit the arcing behavior in order to get multiple points of view on each landmark target.

In answer to research question 12 regarding combining tasks, MVERT can produce trajectories that approximate the one-step optimal for balancing multiple mission goals. The value of each candidate move is computed as a weighted sum of simple, independent value functions rather than designing a more complex value function that accounts for all mission goals. Relative weights of the individual value functions must be tuned in order to achieve the desired type of behavior, with priorities set to reflect mission needs. Weights may be selected by hand tuning, as in these experiments, or could potentially be optimized through a learning process that rewards desired behavior and penalizes undesirable behavior.

10.3 Large-Scale MVERT Planetary Exploration (P2)

These experiments address questions 12 and 13, regarding multiple tasks and team heterogeneity, respectively. For the large-scale environment, motions and sensing are noisy. Position updates use SLAM for landmark targets and robots. The step size is 2.0 meters. The experiment series conducted is:

B. 9 robots exploring the large environment.

The relative weights used for this series were selected empirically to provide a mixed behavior between mapping and target sampling tasks. The weights were tuned by iteratively (~10 trials) changing weights until desired performance was obtained. In this case, desired performance was subjectively defined to primarily favor target sampling (or exploration) but to allow small diversions for mapping and maintaining line of sight. All robots use the same value functions and same weights in these experiments. Total value for candidate moves is computed as a weighted sum of individual values, and Eq. 1 (Section 3.3) becomes:

$$V = V_{TL} + 0.00003\,V_S + 0.000003\,V_E + 0.001\,V_N \qquad \textbf{Eq. 79}$$

Robots are assigned either a target sampling task or an exploration point or neither at any given time. Exploration points are defined as target sampling task locations with a lower priority, and seeking these exploration points encourages robots to completely explore the space. The value for exploration is weighted differently than for target sampling, as shown above. The robots for this mission are heterogeneous with respect to their sensing and motion accuracy. Nearly half of the robots have the ability to perform target sampling tasks, but have higher noise for vision and for motions. The remainder of the team cannot perform target sampling tasks, but can perform exploration and have less noise in vision and in motions. The robots have perfect communication, for simplicity, but attempt to maintain line of sight to demonstrate how this could be implemented to maintain communications when possible. The buffer for approaching target sampling targets is 1.5 meters. The robot capabilities are summarized in Table 30 by robot number. The experiment was run five times in order to evaluate the variability of the results.

Table 30. Large-Scale Planetary Exploration Robot Model Parameters

Robot	Sample	Sensing Model				Walking Model			Turning Model		
		Range	Angle	σ_r	σ_ϕ	σ_x	σ_y	σ_θ	σ_x	σ_y	σ_θ
1-4	Yes	150 m	360º	0.06r m	0.03 rad	0.02m	0.004m	0.04 rad	0.002m	0.002m	0.04 rad
5-9	No	150 m	360º	0.02r m	0.005rad	0.01m	0.002m	0.02 rad	0.001m	0.001m	0.02 rad

The exploration points are chosen along the far perimeter of the environment to encourage exploration of the fringe and are spaced to ensure that all area between them can be observed, based on visual range. In this way, if robots visit each exploration point, their sensor footprints will be adequate to observe the entire environment. The exploration points are: (50, 250), (150, 250), (250, 250), (250, 150), and (250, 50).

Trajectories for the five runs of the planetary exploration mission are shown in Figure 94.

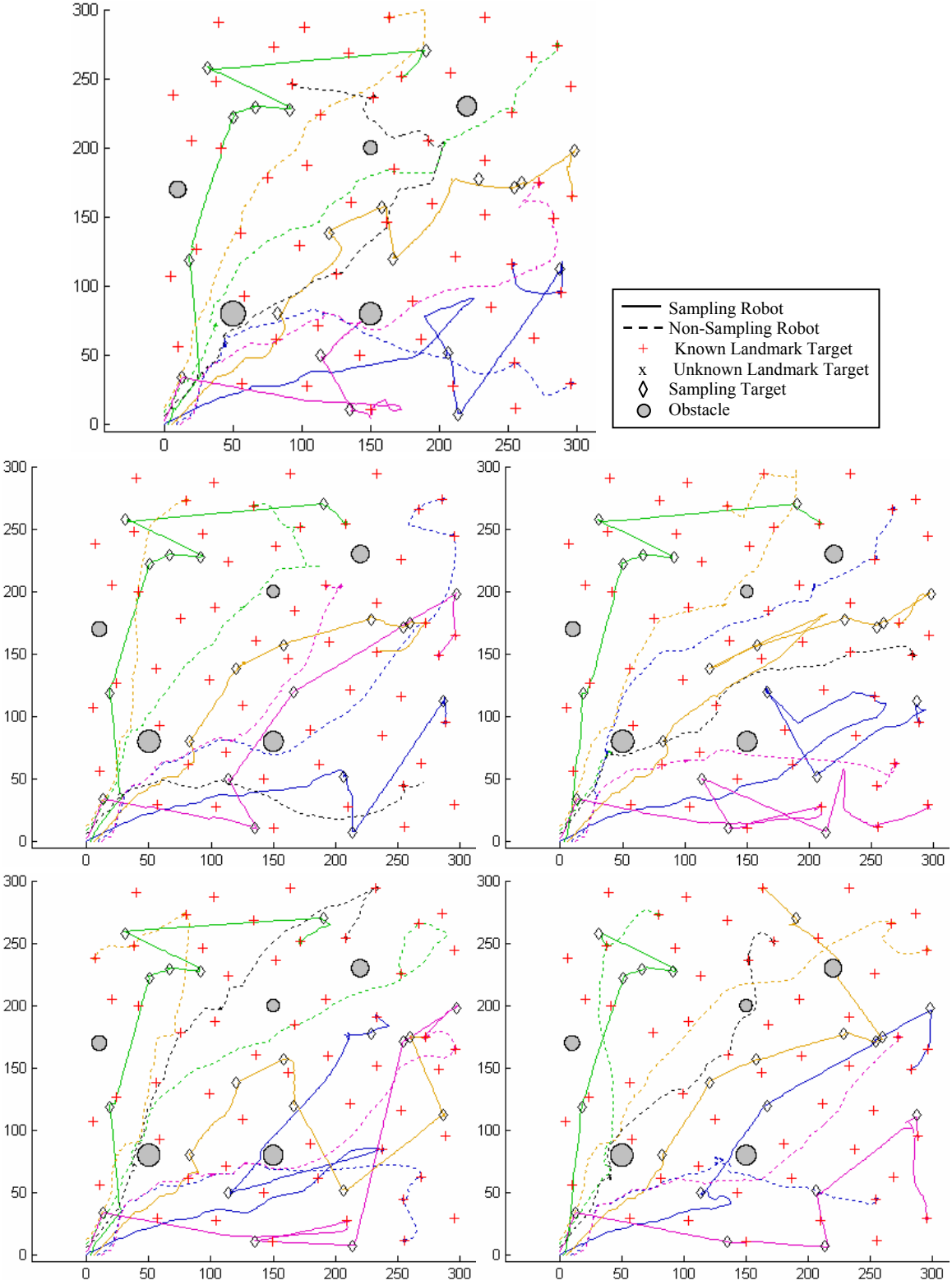

Figure 94. Experiment B, P2: Large-Scale Planetary Exploration

Sampling robots primarily go to target sampling locations (◊) directly, with small mapping diversions in poorly known regions. Non-sampling robots divert more to map targets, but move generally toward exploration points. Units are meters.

Quantitative results for the large-scale planetary exploration are summarized in Table 31.

Table 31. Large-Scale Planetary Exploration Results

Trial	All Targets Seen (E_4, steps)	Target Sampling Complete (E_4, steps)	Mean Landmark Target Error (E_1, meters)
1	130	310	0.520
2	294	320	4.64
3	130	312	1.70
4	149	463	2.09
5	155	352	1.48
Mean	171.1±69.3 (95% conf: ±27.2)	351.4±64.6 (95% conf: ±25.3)	2.08±1.54 (95% conf: ±0.60)

The high weight placed on target sampling causes the sampling-equipped robots to move almost directly to each target sampling location in sequence. It is observed that there are some small diversions in these paths to improve areas of the map that are not being investigated by teammates. One example is the case illustrated in Figure 95. Once all the target sampling tasks have been completed, these robots focus more on mapping, and start following trajectories similar to those demonstrated in the large-scale mapping experiments (8.4). These arcing paths moving toward landmark targets can be observed, for example, at the top of each figure, after the robots performing target sampling complete their last target sampling tasks.

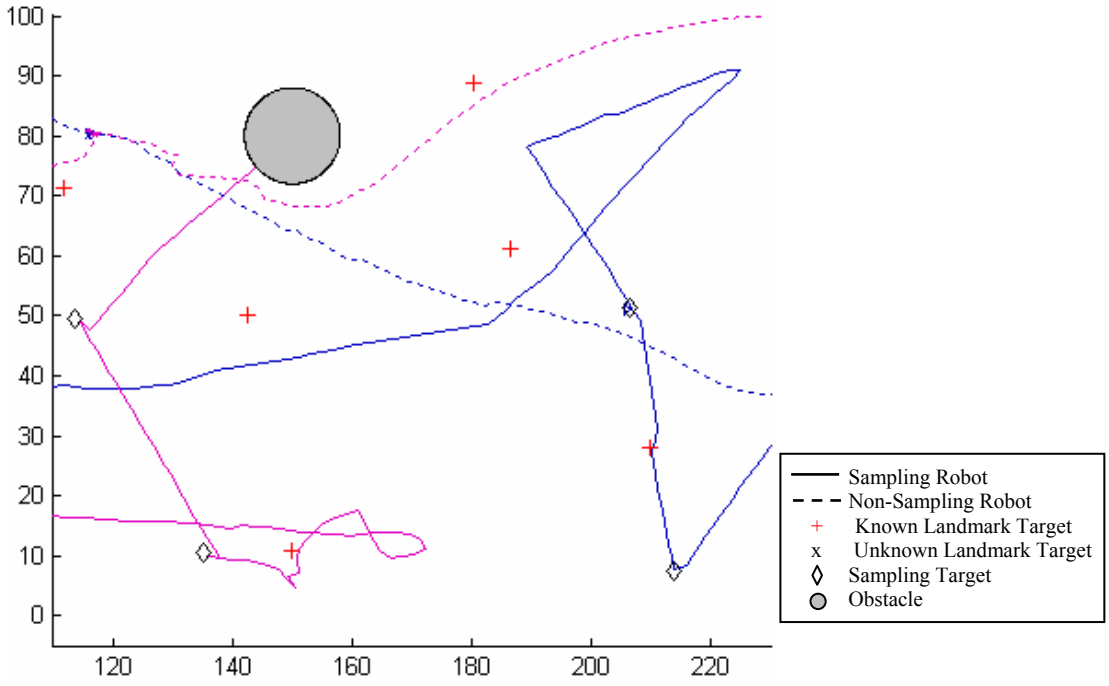

Figure 95. Experiment B, P2: Large-Scale Planetary Exploration Close-up
A close-up of robot trajectories (top, Figure 94). The robots with trajectories take slightly curved paths to better observe targets, and in fact move past sample targets to map very poorly known areas before returning to sample the target. At the top left a dotted (blue) robot remains stationary near the obstacle, to maintain connectivity until other robots move past the obstacle. The solid pink robot eventually gets stuck in a local optimum near the obstacle. Units are meters.

Robots occasionally pause in their paths to maintain line of sight connectivity of the whole team. This is shown in the example of Figure 95, where a robot (dotted, blue) remains to the left of the obstacle at (150, 70). It can also be clearly observed in the center of Figure 96, where two robots (dotted, blue and green) remain to the lower left of the obstacle at (50, 80) to maintain line of sight around the large obstacle.

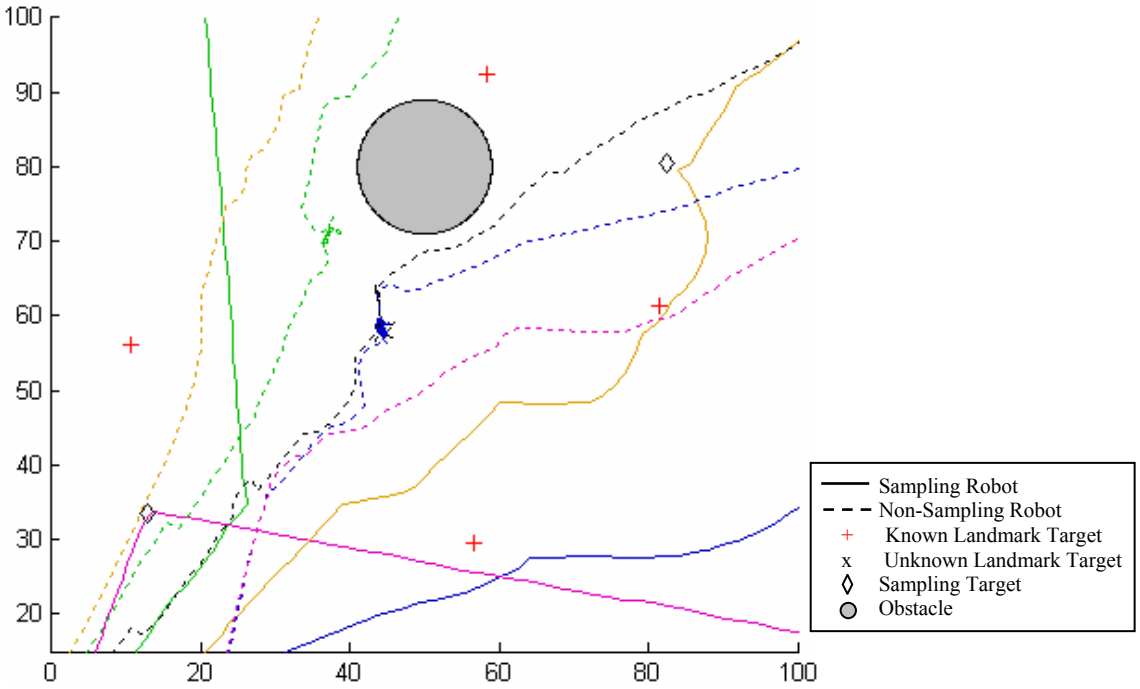

Figure 96. Experiment B, P2: Large-Scale Planetary Exploration Close-up
A close-up of robot trajectories (top, Figure 94). Some of he robots with solid trajectories take slightly curved paths to better observe targets while others move more directly to targets. Below and to the left of the obstacle, two robots remain stationary to maintain network connectivity until other robots move past the obstacle. They remain longer than needed, maintaining connectivity with each other, due to a local optimum. Units are meters.

In contrast to the higher ratio of weights for target sampling versus target location (these robots move directly toward sampling targets), the weight on exploration value relative to the weight on target location (mapping) allows robots engaged in exploration to divert more to improve the map quality while exploring. This is evident, for example, in the mapping robots in Figure 96. However, they are still strongly drawn to the exploration points to ensure that the region is explored thoroughly and quickly. Once the exploration points have been reached, these robots focus only on mapping, and the trajectories divert even more to improve map quality where it is needed.

The team finds all the landmark targets, completing the exploration of the environment, in an average of 171.6 steps. The robots complete all the target sampling tasks in an average of 351.4 steps. In some cases, the robots, due to pose error, do not actually approach the target sampling tasks within the desired range. This in practice would be easily remedied by allowing the robot to approach the identified target sampling tasks using observational feedback once it was within close proximity. This step is omitted from the control system for simplicity.

10.4 Summary and Discussion

In summary, it can be concluded from these experiments that MVERT can efficiently select actions for a team of robots carrying out a complex, multi-task mission. Selected actions maximize the team progress toward completion of all mission goals, given an overall value function to maximize. These actions are

appropriate in that they are biased toward accomplishing high-priority tasks more quickly than low-priority tasks, where priority is set using the relative weights of the individual task value functions.

By selecting actions that maximize total value across all tasks, MVERT can produce trajectories that provide maximal progress toward mission goals, relative to the priorities provided and the candidate actions. This can improve performance by selecting actions that progress toward multiple tasks rather than addressing them sequentially. Team behavior can be adjusted to reflect desired performance by adjusting the weights on each individual task. These weights may be different for different team members, and may be changed during the mission to reflect changing needs by humans or an AI planner.

In these types of missions, MVERT may also be a useful tool for mission design. Running in simulation with different team compositions could provide insight into the types of teams best suited to the particular mission in terms of the area, expected terrain, and expected number and density of targets of interest.

MVERT provides a greedy one-step solution, which may provide occasionally undesirable results. This may be addressed by designing more precise value functions or by integrating MVERT with a planner that can interact with and monitor execution. In this way, planning is made more efficient by allowing planning to neglect low-level actions, while execution is made more efficient by eliminating local optima.

Chapter 11 Discussion and Conclusion

11.1 MVERT Architecture

The MVERT Architecture is a hybrid behavior-based/one-step plan framework for *multi-robot move action selection* at the execution level. MVERT is specifically designed for application to heterogeneous multi-robot teams carrying out complex missions in complex environments. The goal of developing MVERT is to provide a framework for selecting reasonable actions for a team by approximating optimal team action selection. The optimal action is defined as the team action that makes the most progress toward mission goals. Progress toward completing mission goals is represented by mathematical value functions that map the current state and potential robot move action to a numerical value representing mission completeness. In applying these value functions, MVERT incorporates the poses of teammates into the current state. Approximations of the expected next-step teammate contributions are made by applying teammate pose, sensing models, and capability to the value functions. The approximate expected teammate contribution serves as a baseline with which expected contributions of candidate actions are computed in order to select an action that improves the overall team performance.

MVERT is designed to investigate the following research question:

How can a heterogeneous team of robots maximize team progress in a multi-task mission in a scalable, efficient manner at execution?

Specifically, MVERT is applied to team execution of missions to address specific subsidiary questions:

- *How can robot teams effectively integrate multiple mission tasks?*
- *How should robot teams be distributed dynamically to best improve the uncertainty of maps and target locations?*
- *How should dynamic elements of the environment be treated?*
- *How can performance of multi-robot systems be evaluated in this context?*

This work in particular addresses the following thesis statement:

Applying value functions in action selection can provide efficient team execution of multiple parallel tasks.

Actions are selected to maximize team progress toward the mission goals, making it a greedy search. Mathematical value functions represent progress toward each individual mission goal, making progress toward these goals easy to compute. Overall mission progress is represented by a weighted sum of each individual value function. The weights selected for each individual value function affect the overall behavior of the team by placing greater priority on completing some tasks than on others. These weights are therefore selected based on the desired team performance; in this case weights are selected experimentally.

During action selection, each robot first approximates the expected contributions of teammates using the value functions and known models of the teammates' sensing, capabilities, and current poses. This single average expected contribution by teammates serves as the basis for selecting the action. Each robot then considers a set of candidate actions (in this case, locations one step away) and evaluates the value of each candidate action. The action that results in the highest-valued pose, given the expected teammate contribution, is selected and executed. Flexibility in computational complexity is achieved at this level by adjusting the number of candidate actions that are considered (in this case, selecting the angular resolution at which points on the one-step radius circle are sampled). As an execution-level approach, the computational complexity is made more scalable for large environments and teams by maximizing expected progress only for the next step, making it a type of greedy search.

The MVERT architecture can be applied to the execution of tasks that can be represented, at least approximately, by a mathematical function. Many types of tasks that are not usually considered in this way may in fact be appropriate. In this work, the mission tasks investigated are mapping, dynamic target tracking, and complex exploration (including mapping, exploration, maintaining line of sight, and time-consuming target sampling tasks). Mapping here is represented as a task to reduce the uncertainty on target locations and is represented by the reduction in the area of the ellipse representing one standard deviation

of the Gaussian distribution. Target tracking, also a target location task, uses the same mathematical representation for value. Exploration and target sampling require moving toward a desired location and then remaining at that location for a specified period of time (to complete whatever task is required). The first part is represented essentially as a potential field with the desired location as an attractor. Once there, a binary value is assigned, high value for remaining to consume and high negative value for moving away. Lastly, for maintaining line of sight, value is related to the number of teammates that are visible, directly or indirectly.

11.2 Performance

This work shows that the MVERT Architecture produces successful trajectories for multi-robot teams to perform mapping, tracking, and exploration missions. These trajectories maximize use of the robot's sensors in concert with its teammates, automatically selecting moves that triangulate for asymmetrical sensors which require multiple views and selecting moves that directly approach targets for symmetrical sensors. Trajectories take advantage of teammate contributions, and automatically, by optimizing joint value, distribute robots to simultaneously observe different areas and from different locations. By minimizing uncertainty, a team of robots can successfully map and track targets. MVERT also allows for an easy combination of multiple tasks by simple weighting of value functions, and successfully completes complex missions, such as planetary exploration.

As demonstrated in 7.7, the reduction in map uncertainty by using MVERT is only slightly less than one-step optimal. The loss due to the approximations made by MVERT is relatively small and the trajectories produced are qualitatively very similar. Additionally, computational complexity is reduced from exponential in team size (for one-step optimal) to linear in team size. This improvement allows the practical implementation of MVERT for much larger teams, and much larger environments. Given that the one-step optimal took several hours to compute an action for a team size of 3 for only the single task of target location, applying the optimal for a more complex mission, such as the 9-robot planetary exploration task investigated here (Chapter 10), would be impossible for real-time action selection.

In comparison to individual action selection, as demonstrated in 7.6 and 8.2-8.4, MVERT provides a significant improvement in performance. The improvement relative to individual action selection increases with team size. By including approximate expected teammate contributions in evaluating the value of each move, MVERT selects moves that distribute robots in the environment and produces observations the complement each other, in terms of uncertainty reduction. Individual action selection causes robots to follow similar paths, which leads to frequent failure to explore the space, failure to maximize value (minimize uncertainty), and more frequent inter-robot interference. Additionally, by using only their own observations, robots using individual action selection have lower-quality maps during execution, making observations less accurate and making localization more noisy, leaving robots more prone to getting lost. While maintaining the same ability as the multi-robot MVERT to integrate multiple tasks, the multi-task mission would suffer from the same robot clustering plaguing the mapping task.

In comparison to the traditional multi-robot mapping approach using coverage patterns in subregions, the MVERT results are mixed (8.5): in some environments, coverage patterns perform better while in others, MVERT provides better performance. When using coverage patterns, robots move very directly to the assigned mapping areas, while MVERT robots spend more time exploring and mapping and do not reach the farther parts of the environment as quickly. In this way, particularly in large teams, robots using coverage patterns may cover the space slightly more quickly than MVERT. Additionally, robots remain within their assigned areas observing while MVERT robots move on to explore more areas. This means that in environments with uniformly distributed targets, more overall measurements may be made on each target, leading to a map with lower uncertainty and higher accuracy. However, in environments with very non-uniform target distributions, coverage pattern performance may suffer for several reasons. First, some robots will be assigned to explore unproductive areas, effectively reducing team size, while all MVERT robots will follow the targets. Additionally, since each area is assigned only one robot for coverage, with the greater concentration of robots in the target-rich areas with MVERT, overall more measurements are taken on these targets using MVERT than using coverage. A last advantage of MVERT over coverage patterns is its flexibility. Coverage patterns do not account for some tasks or targets having priority over others, requiring a sequential and methodical coverage of the area. Coverage patterns and their rigidity also do not allow diversions for unexpected opportunities. Other approaches, not investigated here, also suffer

from this same type of rigidity. For example, the market-based approach [30] determines the optimal robot-target location assignment, but cannot alone account for selecting paths to these targets to improve the map because they rely on accurate models of cost for reaching the targets.

As shown in Chapter 9, MVERT can produce appropriate trajectories for following multiple dynamic targets, even using a simple value function for uncertainty reduction and without any target motion prediction. Robots maximize value by attempting to simultaneously observe all targets. This leads to automatic distribution of robots among targets when they diverge in the environment.

As demonstrated by the experiments in Chapter 10, MVERT can produce successful trajectories for completing multiple parallel mission tasks. In Section 10.2, the ability to tune the team's behavior to match desired performance is demonstrated. This tuning is achieved by changing the relative weights of the different tasks, providing higher priority to more important tasks. The ability to simultaneously complete multiple mission tasks with a heterogeneous team is illustrated in Section 10.3. Here, contextually appropriate actions are selected by each team member and the team is able to successfully complete the task using only the one-step approach. Robots balance the different tasks, placing highest importance on those highest priority tasks. Robots automatically shift roles depending on the state, such as when robots temporarily remain near an obstacle to serve as a communications relay.

As an approach that selects actions for only one step, MVERT is highly adaptive and flexible. Potentially, as robots join or leave the team, decisions in the next step will consider only the current team composition in selecting an action. This approach is also partially self-correcting. If poor information leads to a poor decision at one step, additional information obtained in that step will improve the decision made at the next step. Loss of communication or poor information may degrade performance (as in any robotic system), but MVERT will continue to make action decisions to maximize progress at each step, allowing it to degrade gracefully without necessarily failing. In the worst case, robots reduce to making decisions based only on their own information and maximize their own value.

The MVERT approach, being an approach that anticipates only one step, suffers from some of the typical limitations of behavior-based and reactive control. First, robots may become stuck in local optima. For example, robots may remain very close to a single target rather than moving to another target that is well known. This is a local optimum in the sense that the small improvement by moving slightly closer to a far target is overwhelmed by the large loss of improvement by moving farther from a close target. This may be handled by providing an upper limit on uncertainty value, making no further improvement possible on the close target and allowing the robot to take advantage of the small improvement possible on the far target. Similarly, there may be oscillations around these local optima; this can also be handled using traditional approaches, such as adding noise or a timeout.

One-step approaches, such as MVERT, are inherently suboptimal, trading optimality for computational efficiency. Since MVERT approximates teammate contributions by ignoring future movement, robots do not fully account for the correct teammate contribution and may choose actions that are slightly suboptimal. However, as previously mentioned, the effects of this are small. MVERT is also suboptimal in coverage. Looking ahead only one step, and choosing the next action greedily, robots may fail to cover the space. This is particularly possible if the spacing between targets is large compared to the sensing footprint of the robots. This, as demonstrated in Chapter 10, can be remedied by adding an additional value function to enforce exploration, similar to the concept behind coverage patterns. In the case of dynamic target tracking, the simple one-step approach also has limitations. If targets change direction quickly, they can be lost by robots with limited sensing footprints. This may be improved by adding tracking (such as with an EKBF) and a search strategy for reacquiring lost targets.

Another performance problem with the MVERT approach is that since it relies on models to predict mission progress, high levels of noise or other mismatches between model and reality may lead to poor behavior. This became evident in some of the physical robot experiments, where the Gaussian approximations to the sensor and motion models are not as well matched as in the simulations. The relative performance of MVERT on the physical robots drops below that demonstrated in simulation. This can only be improved by matching models as well as possible.

A final limitation of MVERT in its current implementation is that the value functions and weights on value functions for multiple tasks must be hand designed and hand-tuned. Designing these functions and weights

may be tedious and time consuming. For some tasks, such as those demonstrated here, designing the value functions may be tractable, with performance experimentally validated. Similarly, the relative weights of individual value functions can be empirically tuned to match performance with desired behavior. The process is simplified by separating out individual tasks. However, with many tasks or with more complex tasks, designing good value functions and determining these weights may be impractical. This is also true for modeling the robot sensing and motions.

11.3 Contributions

The contributions of this work include:

- An architecture for selecting actions for a heterogeneous team at the execution level during a complex mission, including:
  - A method for iterative, near-optimal distribution of a robot team for mapping multiple distributed static and dynamic objects;
  - A method for selecting actions to maximize mission progress in a multi-task mission.
- Validation of the approach in simulation for target location, mapping, target tracking, and a multi-task complex mission with heterogeneous team.
- An implementation of this framework on a real multi-robot system for mapping and target tracking.
- Metrics proposed for evaluating the performance of such multi-robot systems.
- Suggested simple value functions for some common tasks.

This work demonstrates that by combining the needs of multiple mission tasks and considering expected teammate contributions, simple behaviors can lead to complex, "intelligent" team performance.

11.4 Future Work

Several research directions are recommended to expand and improve this work. These directions are suggested by the experimental results and by the needs for implementing physical robot teams. The general areas for future work are: integration with more complex reasoning, applying learning for accuracy and adaptability, investigating scalability, expanding models beyond the two-dimensional representation, and applying more relevant models of communications.

First, local optima and other deviations from optimal performance due to the one-step approach may be significant for some tasks. In addition to the strategies mentioned above for dealing with these issues within the execution level itself, MVERT may be integrated as the execution level of a tiered architecture or more complex control system that includes some type of planning or reasoning. For example, using the market-based approach to initially assign robots to targets will prevent the potential for robots having to traverse long distances at the end of the mission (as in Figure 93 and Figure 94, Chapter 10). However, using MVERT to determine paths between targets, while requiring slightly higher cost to reach the targets, may benefit the overall mission by aiding in creating more accurate maps. This can prevent local optima, and make the action choices of MVERT more closely match the overall desired performance. Most importantly, by taking advantage of some one-step reasoning about teammates and overall performance, the space over which the task allocation or planning must be executed might be significantly reduced, improving computational efficiency. Integration with a planner could also be used to adjust the weights of the individual value functions according to varying mission needs; these adjustments could be dependent on time or state, or even follow complex non-linear functions if required.

Mismatches in models and value functions compared to true robot performance and mission progress have been demonstrated to have a large detrimental effect on MVERT performance. Such mismatches may be due to design flaws or to changing robot or environmental conditions. This type of limitation may be improved by applying learning techniques. Offline learning may be able to make robot models more accurately match truth and may be able to make value functions and value function weights provide behavior closer to desired performance. Additionally, online learning may be able to track and account for changes in robot performance and changes in the environment that affect mission progress.

This work has only thusfar investigated fixed team sizes of up to twenty-five robots. Further investigation of larger teams and larger environments could determine upper limits of team size for real-time computation. Larger teams could be made more scalable by reducing the number of teammates considered to those in the immediate area, or those that may have a direct impact on the next selected action. Additionally, robustness to the addition and removal of teammates can be directly investigated to verify the epected ability of MVERT to handle such changes.

This current implementation of this work has assumed an essentially two-dimensional terrain. More complex terrains could be handled by using a 2.5 dimensional representation, or an elevation map representation. In this way, an additional cost could be added to the value of each action to represent the cost of moving to that location. This cost might be due to power consumption and modeled, for example, as a function of slope. Slopes that are too steep might eliminate some potential actions. In itself, consideration of terrain might add little complexity to the computation. However, three-dimensional predictions of line of sight for target observation and communications could add significantly to the computation time, and this trade-off must be considered.

Lastly, this work assumes full communication to share all information required to create a common world view. In practicality, this is typically not available. MVERT may benefit from more accurate models of real communication performance for a particular robot team in the specific operating environment. Additionally, in the absence, observation of teammates may allow for estimating and complementing teammate contributions even without direct information. Investigations into the effects of reduced or absent communications would also benefit design of practical systems using MVERT.

11.5 Summary and Discussion

MVERT is a hybrid behavior-based/one-step reasoning multi-robot execution architecture that incorporates one-step reasoning about teammate contributions and action results to select actions. Actions are selected based on their expected contribution to multiple mission goals, as measured by a weighted sum of individual value functions representing task performance. This hybrid approach produces contextually appropriate action selection and produces successful trajectories for completing multi-task missions. MVERT provides improved team performance (compared to individual action selection) and improved flexibility while reducing computational complexity (relative to one-step optimal or optimal trajectory planning). Limitations due to one-step planning may be reduced by applying high level reasoning while allowing MVERT to optimize over the short term. MVERT has been applied in simulation and on physical robot teams to mapping, tracking, and multi-task exploration missions and produced successful trajectories for these missions.

List of References

[1] W. Anderson and R. Duffin. *Journal of Mathematical Analysis Applications*, 26, 1969.

[2] R. Arkin. "Motor Schema-Based Mobile Robot Navigation." *International Journal of Robotics Research*, 8(4), 1989.

[3] Y. Arai, T. Fujii, H. Asama, H. Kaetsu, and I. Endo. "Realization of Autonomous Navigation in Multirobot Environment." *Proceedings of the 1998 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 1998.

[4] T. Balch. "TeamBots." Web address: http://www.cs.cmu.edu/~trb/TeamBots, 2000.

[5] T. Balch and R. Arkin. "Communication in Reactive Multiagent Robotic Systems." *Autonomous Robots*, 1, 1994.

[6] T. Balch. "Hierarchic Social Entropy: An Information Theoretic Measure of Robot Team Diversity." *Autonomous Robots*, 8(3), 2000.

[7] S. Borthwick and H. Durrant-Whyte. "Simultaneous Localisation and Map Building for Autonomous Guided Vehicles." *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 1994.

[8] A. Briggs. "Efficient Geometric Algorithms for Robot Sensing and Control." PhD Thesis, Cornell University, 1995.

[9] R. Brooks. "A Robust Layered Control System for a Mobile Robot." *IEEE Journal of Robotics and Automation*, 2(1), 1986.

[10] R. G. Brown and B.R. Donald. "Mobile robot self-localization without explicit landmarks." *Algorithmica*, 26(3/4), 2000.

[11] J Bruce, T. Balch, and M. Veloso. "Fast and Inexpensive Color Image Segmentation for Interactive Robots." *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2000*, 2000.

[12] B. Brumitt and A. Stentz. "Dynamic Mission Planning for Multiple Mobile Robots." *Proceedings of the IEEE International Conference on Robotics and Automation 1996*, 1996.

[13] J. Brusey, A. Jennings, M. Makies, C. Keen, A. Kendall, L. Padgham, and D. Singh. "RMIT Raiders." In: **RoboCup-99: Robot Soccer World Cup III**, Veloso, Pagello, and Kitano (Eds.), Springer-Verlag, 2000.

[14] R. S. Bucy and J. P. D. Joseph. **Filtering for Stochastic Processes with Applications to Guidance**. John Wiley Press, New York, 1968.

[15] W. Burgard, D. Fox, H. Jans, C. Matenar, and S. Thrun. "Sonar-Based Mapping of Large-Scale Mobile Robot Environments Using EM." *Proceedings of the 1999 International Conference on Machine Learning*, 1999.

[16] W. Burgard, D. Fox, M. Moors, R. Simmons, and S. Thrun. "Collaborative Multi-Robot Exploration." *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA) 2000*, 2000.

[17] A. Cai, T. Fukuda, and F. Arai. "Information Sharing among Multiple Robots for Cooperation in Cellular Robotic System." *Proceedings of the 1997 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 1997.

[18] A. Cai, T. Fukuda, F. Arai, and H. Ishihara. "Cooperative Path Planning and Navigation Based on Distributed Sensing." *Proceedings of the 1996 IEEE International Conference on Robotics and Automation (ICRA) 1996*, 1996.

[19] N. Carver, V. Lesser, and Q. Long. "Distributed Sensor Interpretation: Modeling Agent Interpretations in DESRUN." SMPSCI Technical Report 93075, University of Massachusetts, 1993.

[20] J. A. Castellanos, J. M. M. Montiel, J. Neira, and J. D. Tardós. "The SPmap: A Probabilistic Framework for Simultaneous Localization and Map Building." *IEEE Transactions on Robotics and Automation*, 15(5), 1999.

[21] P. Costa, A. Moreira, A. Sousa, P. Marques, P. Costa, and A. Matos. "5dpo-2000 Team Description." In: **RoboCup-99: Robot Soccer World Cup III**, Veloso, Pagello, and Kitano (Eds.), Springer-Verlag, 2000.

[22] M. C. Deans. "Robust and Efficient Simultaneous Localization and Mapping from Bearings Only Sensing." Thesis Proposal, Carnegie Mellon University, 2000.

[23] M. Deans and M. Hebert. "Experimental Comparison of Techniques for Localization and Mapping using a Bearings Only Sensor." *Proceedings of the ISER '00 Seventh International Symposium on Experimental Robotics*, 2000.

[24] G. Dedeoglu and G. S. Sukhatme. "Landmark-based Matching Algorithm for Cooperative Mapping by Autonomous Robots." *Proceedings of the 5th International Symposium on Distributed Autonomous Robotic Systems*, 2000.

[25] F. Dellaert, D. Fox, W. Burgard, and S. Thrun. "Monte Carlo Localization for Mobile Robots." *Proceedings of the 1999 IEEE International Conference on Robotics and Automation (ICRA) 1999*, 1999.

[26] F. Dellaert, D. Fox, W. Burgard, and S. Thrun. "Using the CONDENSATION Algorithm for Robust, Vision-based Mobile Robot Localization." *Proceedings of the 1999 IEEE International Conference on Computer Vision and Pattern Recognition*, 1999.

[27] F. Dellaert, S. M. Seitz, C. Thorpe, and S. Thrun. "EM, MCMC, and Chain Flipping for Structure from Motion with Unknown Correspondence." *Machine Learning*, 2000.

[28] F. Dellaert and A. Stroupe. "Linear 2D Localization and Mapping for Single and Multiple Robot Scenarios." *Proceedings of the 2002 IEEE International Conference on Robotics and Automation (ICRA)*, 2002.

[29] M. B. Dias, D. Goldberg, and A. Stentz. "Market-Based Multirobot Coordination for Complex Space Applications." *The 7th International Symposium on Artificial Intelligence, Robotics and Automation in Space*, 2003.

[30] M. B. Dias and A. Stentz. "A Market Approach to Multirobot Coordination." Technical Report CMU-RI-TR-01-26, Robotics Institute, Carnegie Mellon University, 2001.

[31] M. Dietl, J.-S. Gutmann, and B. Nebel. "Cooperative Sensing in Dynamic Environments." *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2001.

[32] G. Dissanayake, H. Durant-Whyte, and T. Bailey. "A Computationally Efficient Solution to the Simultaneous Localisation and Map Building (SLAM) Problem." *Proceedings of the 2000 IEEE International Conference on Robotics and Automation (ICRA), 2000*, 2000.

[33] M. W. M. G. Dissanayake, P. Newman, S. Clark, H. F. Durrant-Whyte, and M. Csorba. "A Solution to the Simultaneous Localization and Map Building (SLAM) Problem." *IEEE Transactions on Robotics and Automation*, 17(3), 2001.

[34] E. J. P. Earon, T. D. Barfoot, and G. M. T. D'Eleuterio. "Development of a Multiagent Robotic System with Application to Space Exploration." *2001 IEEE/ASME International Conference on Advanced Intelligent Mechatronics Proceedings*, 2001.

[35] J. Fenwick, P. Newman, and J. Leonard. "Cooperative Concurrent Mapping and Localization." *Proceedings IEEE International Conference on Robotics and Automation (ICRA) 2002*, 2002.

[36] D. Fox, W. Burgard, H. Kruppa, and S. Thrun. "A Probabilistic Approach to Collaborative Multi-Robot Localization." *Autonomous Robots on Heterogeneous Multi-Robot Systems, Special Issue*, 8, (3), 2000.

[37] D. Fox, W. Burgard, and S. Thrun. "Markov Localization for Mobile Robots in Dynamic Environments." *Journal of Artificial Intelligence Research*, 11, 1999.

[38] A. Garuilli and A. Vicino. "Set Membership Localization of Mobile Robots via Angle Measurements." *IEEE Transactions on Robotics and Automation*, 17(4), 2001.

[39] D. Gennery. "Traversability Analysis and Path Planning for a Planetary Rover." *Autonomous Robots*, 1999.

[40] B. Gerkey and M. Matarić. "Sold!: Auction Methods for Multirobot Coordination." *IEEE Transactions on Robotics and Automation*, 18(5), 2002.

[41] P. Goel, S. I. Roumeliotis, and G. S. Sukhatme. "Robust Localization Using Relative and Absolute Position Estimates." *Proceedings of the 1999 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 1999.

[42] H. González-Baños, L Guibas, J.-C. Latombe, S. LaValle, D. Lin, R. Motwani, and C. Tomasi. "Motion Planning with Visibility Constraints: Building Autonomous Observers." *Proceedings of the 8th International Symposium on Robotics Research*, 1997.

[43] R. Grabowski, L E. Navarro-Serment, C. J. J. Pareidis, P. K. Khosla. "Heterogeneous Teams of Modular Robots for Mapping and Exploration." *Autonomous Robots Special Issue on Heterogeneous Multi-Robot Systems*, 2000.

[44] D. Graydon and K. Hanson, (Editors). **Mountaineering: The Freedom of the Hills**, 6th edition, Mountaineers, Seattle, 1997.

[45] F. C. A. Groen, J. Roodhard, and J. Vunderink. "A Distributed Dynamic World Model for Robot Soccer." In: **RoboCup-2000: Robot Soccer World Cup IV**, Stone, Balch, and Kraetzschmar (Eds.), Springer-Verlag, 2001.

[46] J. E. Guivant and E. M. Nebot. "Optimization of the Simultaneous Localization and Map-Building Algorithm for Real-Time Implementation." *IEEE Transactions on Robotics and Automation*, 17(3), 2001.

[47] J.-S. Gutmann, W. Burgard, D. Fox and K. Konolige. "An Experimental Comparison of Localization Methods." *Proceedings IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS-98)*, 1998.

[48] J.-S. Gutmann, W. Hatzack, I. Herrmann, B. Nebel, F. Rittinger, A. Topor, and T. Weigel. "Reliable self-localization, multirobot sensor integration, accurate path-planning and basic soccer skills: playing an effective game of robotic soccer." *Proceedings of the Ninth International Conference on Advanced Robotics*, 1999.

[49] J.-S. Gutmann, T. Weigel, and B. Nebel. "A Fast, Accurate, and Robust Method for Self-Localization in Polygonal Environments Using Laser-Range-Finders." *Advanced Robotics Journal*, 14(8), 2001.

[50] K. Han and M. Veloso. "Physical Model Based Multi-objects Tracking and Prediction in RoboSoccer." *Working Notes of the AAAI 1997 Fall Symposium on Model-directed Autonomous Systems*, 1997.

[51] R. Hanek and T. Schmitt. "Vision-Based Localization and Data Fusion in a System of Cooperating Mobile Robots." *Proceedings of the 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS),* 2000.

[52] R. Hanek, T. Schmitt, M. Klupsch, and S. Buck. "From Multiple Images to a Consistent View." In: **RoboCup-2000: Robot Soccer World Cup IV**, Stone, Balch, and Kraetzschmar (Eds.), Springer-Verlag, 2001.

[53] D. Hershberger, R. Burrdige, D Kortenkamp, and R. Simmons. "Distributed Visual Servoing with a Roving Eye." *Proceedings of the Conference on Intelligent Robots and Systems (IROS) 2000*, 2000.

[54] B. Horling, R. Vincent, R. Mailler, J. Shen, R. Becker, K. Rawlins, and V. Lesser. "Distributed Sensor Network for Real Time Tracking." *Proceedings of the 5th International Conference on Autonomous Agents*, 2001.

[55] A. Howard and L. Kitchen. "Cooperative Localisation and Mapping." *Proceedings of the 1999 International Conference on Field and Service Robots*, 1999.

[56] A. Howard, M. Matarić, and G. Sukhatme. "Mobile Sensor Network Deployment using Potential Fields: A Distributed Scalable Solution to the Area Coverage Problem." *Distributed Autonomous Robotic Systems 5: Proceedings of the 6th International Conference on Distributed Autonomous Robotic Systems (DARS02)*, 2002.

[57] A. Howard, M. Matarić, and G. Sukhatme. "An Incremental Deployment Algorithm for Mobile Robot Teams." *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) 2002*, 2002.

[58] A. Howard, M. Matarić, and G. Sukhatme. "Localization for Mobile Robot Teams Using Maximum Likelihood Estimation." *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) 2002*, 2002.

[59] A. Howard, M. Matarić, and G. Sukhatme. "Cooperative Relative Localization for Mobile Robot Teams: An Ego-Centric Approach." In: **Multi-Robot Systems: From Swarms to Intelligent Automata, Vol. II**, Shultz, Parker, and Schneider (Eds.), Klewer, 2003.

[60] J. A. Janét, D. S. Schudel, M. W. White, A. G. England, R. C. Luo, and W. E. Snyder. "Global Self-Localization for Actual Mobile Robots: Generating and Sharing Topographical Knowledge Using the Region-Feature Neural Network." *Proceedings of the 1996 IEEE/SICE/RSJ International Conference on Multisensor Fusion and Integration for Intelligent Systems*, 1996.

[61] M. Jenkin and G. Dudek. "The Paparazzi Problem." *Proceedings of the 2000 IEEE/RSJ International Conference on Robots and Systems (IROS)*, 2000.

[62] C. Jennings, D. Murray, and J. J. Little. "Cooperative Robot Localization with Vision-based Mapping." *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA) 1999*, 1999.

[63] N. Jennings. "Coordination Through Joint Intentions in Industrial Multi-Agent Systems." *AI Magazine* 14(4), 1999.

[64] P. Jensfelt and H. I. Christensen. "Pose Tracking Using Laser Scanning and Minimalistic Environmental Models." *IEEE Transactions on Robotics and Automation*, 17(2), 2001.

[65] R. E. Kalman. "New Methods and Results in Linear Filtering and Prediction Theory." *ASME Journal of Basic Engineering*, Series D, 83, 1961.

[66] K. Kato, H. Ishiguro, and M. Bath. "Identification and Localization of Robots in a Multi-Robot System Environment." *Proceedings of the IEEE/RSJ/GI International Conference on Intelligent Robots and Systems*, 1999.

[67] K. Konolige. "Markov Localization Using Correlation." *Proceedings of the 1999 International Joint Conference on Artificial Intelligence (IJCAI)*, 1999.

[68] K. Konolige, G. Didier, and K. Nicewarner. "A Multi-Agent System for Multi-Robot Mapping and Exploration." In: **Multi-Robot Systems: From Swarms to Intelligent Automata,** Shultz and Parker (Eds.), Klewer, 2002.

[69] R. Kurazume, S. Hirose, S. Nagata, and N. Sashida. "Study on Cooperative Positioning System." *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), 1996*, 1996.

[70] T. D. Larsen, K. L. Hansen, N. A. Andersen, and O. Ravn. "Design of Kalman filters for mobile robots; evaluation of the kinematic and odometric approach." *Proceedings of the 1999 IEEE International Conference on Control Applications*, 2, 1999.

[71] S. LaValle, H. González-Baños, C. Becker, and J.-C. Latombe. "Motion Strategies for Maintaining Visibility of a Moving Target." *Proceedings of the International Conference on Robotics and Automation (ICRA) 1997*, 1997.

[72] S. LaValle, D. Lin, L. Guibas, J.-C. Latombe, and R. Motwani. "Finding and Unpredictable Target in a Workspace with Obstacles." *Proceedings of the International Conference on Robotics and Automation (ICRA) 1997*, 1997.

[73] S. Lenser and M. Veloso. "Sensor Resetting Localization for Poorly Modeled Mobile Robots." *Proceedings International Conference on Robotics and Automation (ICRA) 2000*, 2000.

[74] J. Leonard and H. Durrant-Whyte. "Mobile Robot Localization by Tracking Geometric Beacons." *IEEE Transactions on Robotics and Automation*, 7(3), 1991.

[75] J. J. Leonard and H. F. Durrant-Whyte. "Dynamic Map Building for an Autonomous Mobile Robot." *International Journal of Robotics Research*, 11(4), 1992.

[76] F. Lu and E. Milios. "Robot Pose Estimation in Unknown Environments by Matching 2D Range Scans." *Journal of Intelligent Robotic Systems*, 18, 1997.

[77] M. Matarić. "Reinforcement Learning in the Multi-Robot Domain." *Autonomous Robots*, 4(1), 1997.

[78] M. Matarić, M. Nilsson, and K. Simsarian. "Cooperative Multi-Robot Box-Pushing." *Proceedings of the 1995 IEEE/RSJ International Conference on Robots and Systems (IROS)*, 1995.

[79] M. Matarić and G. Sukhatme. "Task-allocation and Coordination of Multiple Robots for Planetary Exploration." *Proceedings of the 10th International Conference on Advanced Robotics 2001*, 2001.

[80] P. S. Maybeck. "The Kalman Filter: An Introduction to Concepts." In: **Autonomous Robot Vehicles**, Cox and Wilfong (Eds.), Springer, 1990.

[81] R. Mázl, M. Kulich, and L. Přeučil. "Range Scan-Based Localization Methods for Mobile Robots in Complex Environments." *IEEE Intelligent Transportation Systems Conference Proceedings*, 2001.

[82] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. "FastSLAM: A Factored Solution to the Simultaneous Localization and Mapping Problem." *Proceedings 18th National Conference on Artificial Intelligence*, 2002.

[83] M. Montemerlo, S. Thrun, and W. Whittaker. "Conditional Particle filters for Simultaneous Mobile Robot Localization and People-Tracking." *Proceedings 2002 IEEE International Conference on Robotics and Automation (ICRA)*, 2002.

[84] S. Moorehead. "Autonomous Surface Exploration for Mobile Robots." Thesis Dissertation, CMU-RI-TR-01-30, Carnegie Mellon University, 2001.

[85] H. P. Moravec. "Sensor Fusion in Certainty Grids for Mobile Robots." *AI Magazine*, Summer, 1988.

[86] H. P. Moravec. "Robust Navigation by Probabilistic Volumetric Sensing." Web address: http://www.ri.cmu.edu/~hpm/project.archive/robot.papers/2000/ARPA.MARS.reports.00/Report.0001.html.

[87] H. Moravec and A. Elfes. "High Resolution Maps from Wide Angle Sonar." *Proceedings of the 1985 IEEE International Conference on Robotics and Automation*, 1985.

[88] L. Moreno, J. M. Armingol, A. de la Escalera, and M. A. Salichs. "Global integration of ultrasonic sensors information in mobile robot localization." *Proceedings of the Ninth International Conference on Advanced Robotics*, 1999.

[89] P. Moutarlier and R. Chatila. "Stochastic Multisensory Data Fusion for Mobile Robot Location and Environment Modeling." *Proceedings of the 5th International Symposium on Robotics Research*, 1989.

[90] T. Nakamura, A. Ebina, M. Imai, T. Ogasawara, and H. Ishiguro. "Real-time Estimating Spatial Configuration between Multiple Robots by Triangle Enumeration Constraints." *Proceedings of the 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2000.

[91] L. E. Navarro-Serment, C. J. J. Pareidis, and P. K. Khosla. "A Beacon System for the Localization of Distributed Robotic Teams." *Proceedings of the 1999 International Conference on Field and Service Robots*, 1999.

[92] A. Nuechter, H. Surmann, and J. Hertzberg. "Planning Robot Motion for 3D Digitalization of Indoor Environments." *Proceedings of the 11th International Conference on Advanced Robotics (ICAR 03), Coimbra, Portugal*. 2003

[93] E. Østergaard, M. Matarić, and G. Sukhatme. "Distributed Multi-Robot Task Allocation for Emergency Handling." *Proceedings of IEEE/RSJ International Conference on Robots and Systems (IROS) 2001*, 2001.

[94] L. Parker. "ALLIANCE: an architecture for fault tolerant multirobot cooperation." *IEEE Transactions on Robotics and Automation*, 14(2), 1998.

[95] L. Parker. "Distributed Control of Multi-Robot Teams: Cooperative Baton-Passing Task." *Proceedings of the 4th International Conference on Information Systems Analysis and Synthesis (ISAS '98)*, 3, 1998.

[96] L. Parker. "Current State of the Art in Distributed Autonomous Mobile Robots." *Proceedings of the 4th International Symposium on Distributed Autonomous Robotic Systems (DARS)*, 2000.

[97] L. Parker. "Distributed Algorithms for Multi-Robot Observation of Multiple Moving Targets." *Autonomous Robots*, 12(3), 2002.

[98] L. Parker. "The Effect of Heterogeneity in Teams of 100+ Mobile Robots." In: **Multi-Robot Systems Volume II: From Swarms to Intelligent Automata**, Schultz, Parker, and Schneider (Eds.), Kluwer, 2003.

[99] L. E. Parker, and B. A. Emmons. "Cooperative Multi-Robot Observation of Multiple Moving Targets." *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), 1997*, 1997.

[100] L. Parker, K. Fregene, Y. Guo, and R. Madhavan. "Distributed Heterogeneous Sensing for Outdoor Multi-Robot Localization, Mapping, and Path Planning." In: **Multi-Robot Systems: From Swarms to Intelligent Automata,** Shultz and Parker (Eds.), Klewer, 2002.

[101] I. E. Paromtchik and H. Asama. "Optical Guidance System for Multiple Mobile Robots." *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), 2001*, 2001.

[102] G. A. S. Pereira. "Distributed Sensing for Cooperative Robotics." *Anais da V Semana de Pós-Graduação em Ciência da Compatção*, 2001.

[103] G. Petryk and M. Buehler. "Robust estimation of pre-contact object trajectories." *Robot Control 1997*, 2, 1997.

[104] M. Piaggio and R. Zaccaria. "Environment Exploration and Navigation by Multiple Robots." *IEEE International Conference on Knowledge Based Intelligent Electronic Systems 1997*, 1997.

[105] R. Pito. "A solution to the next best view problem for automated surface acquisition." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21, 1999.

[106] W. Press, S. Teukolsky, W. Vetterling, and B. Flannery. **Numerical Recipes in C: The Art of Scientific Computing**, second edition. Cambridge University Press, 1992.

[107] Probotics. "Cye," Web address: http://www.probotics.com/, 2001.

[108] I. M. Rekleitis, G. Dudek, and E. E. Milios. "Multi-Robot Collaboration for Robust Exploration." *Proceedings of the IEEE International Conference on Robotics and Automation,(ICRA) 2000*, 2000.

[109] D. Rembold, U. Zimmermann, T. Langle, and H. Worn. "Detection and handling of moving objects." *Proceedings of the 24th Annual Conference of the IEEE Independent Electronic Society, IECON '98*, 3, 1998.

[110] S. I. Roumeliotis and G. A. Bekey. "Distributed Multi-Robot Localization." *Proceedings of the 4th International Symposium on Distributed Autonomous Robotic Systems (DARS),* 2000.
[111] S. I. Roumeliotis and G. A. Bekey. "Synergetic Localization for Groups of Mobile Robots." *Proceedings of the IEEE Conference on Decision and Control 2000*, 2000.
[112] J. Z. Sasiadek and P. Hartana. "Sensor Data Fusion Using Kalman Filter." *Proceedings of the Third International Conference on Information Fusion*, 2, 2000.
[113] C. Schlegel and T. Kämpke. "Filter Design for Simultaneous Localization and Map Building (SLAM)." *Proceedings 2002 IEEE International Conference on Robotics and Automation (ICRA)*, 2002.
[114] M. Schneider-Fontán and M. Matarić. "Territorial Multi-Robot Task Division." *IEEE Transactions on Robotics and Automation*, 14(5), 1998.
[115] A. Schultz, W. Adams, B. Yamauchi, and M. Jones. "Unifying Exploration, Localization, Navigation, and Planning through a Common Representation." *Proceedings 1999 IEEE International Conference on Robotics and Automation (ICRA),* 1999.
[116] I. Shimshoni. "On Mobile Robot Localization from Landmark Bearings." *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA) 2001*, 2001.
[117] R. Simmons, D. Apfelbaum, W. Burgard, D. Fox, M. Moors, S. Thrun, and H. Younes. "Coordination for Multi-Robot Exploration and Mapping." *Proceedings of the National Conference on Artificial Intelligence 2000*, 2000.
[118] R. Simmons, S. Singh, D. Hershberger, J. Ramos, and T. Smith. "Coordination of Heterogeneous Robots for Large-Scale Assembly." *Proceedings of the International Symposium on Experimental Robotics 2000*, 2000.
[119] S. Singh, R. Simmons, T. Smith, A. Stentz, V. Verma, A. Yahja, and K. Schwehr. "Recent Progress in Local and Global Traversability for Planetary Rovers." *Proceedings of the IEEE Conference on Robotics and Automation (ICRA) 2000*, 2000.
[120] C. M. Smith, H. J. S. Feder, and J. J. Leonard. "Multiple Target Tracking with Navigation Uncertainty." *Proceedings of the 37th IEEE Conference on Decision and Control*, 1998.
[121] R. C. Smith and P. Cheeseman. "On the Representation and Estimation of Spatial Uncertainty." *International Journal of Robotics Research*, 5(4), 1986.
[122] J. Spletzer, A.K. Das, R. Fierro, C. J. Taylor, V. Kumar, and J. P. Ostrowski. "Cooperative Localization and Control for Multi-Robot Manipulation." *Proceedings of the 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2001.
[123] J. Spletzer and C. Taylor. "Sensor Planning and Control in a Dynamic Environment." *Proceedings of the IEEE Conference on Robotics and Automation (ICRA) 2002*, 2002.
[124] A. Stroupe. "Mission-Driven Collaborative Observation and Localization." Thesis Proposal, Carnegie Mellon University, 2001.
[125] A. Stroupe and T. Balch. "Mission Relevant Collaborative Observation and Localization." In: **Multi-Robot Systems: From Swarms to Intelligent Automata,** Shultz and Parker (Eds.), Klewer, 2002.
[126] A. W. Stroupe, M. C. Martin, and T. Balch. "Distributed Sensor Fusion for Object Position Estimation by Multi-Robot Systems." *Proceedings of the IEEE Conference on Robotics and Automation (ICRA) 2001*, 2000.
[127] A. Stroupe, K. Sikorski, and T. Balch. "Constraint-Based Landmark Localization." In: **RoboCup 2002: Robot Soccer World Cup VI**, Kaminka, Lima, and Rojas (Eds.), Springer-Verlag 2002.
[128] A. Stroupe and T. Balch. "Collaborative Probabilistic Constraint-Based Landmark Localization." *Proceedings IEEE Intelligent Robot Systems (IROS) 2002*, 2002.
[129] A. Stroupe and T. Balch. "Value-Based Observation with Robot Teams (VBORT) Using Probabilistic Techniques." *Proceedings International Conference on Advanced Robotics (ICAR) 2003*, 2003.
[130] S. Sukkarieh, E. Nettleton, B. Grocholsky, and H. Durrant-Whyte. "Information Fusion and Control for Multiple UAVS." In: **Multi-Robot Systems: From Swarms to Intelligent Automata, Vol. II**, Shultz, Parker, and Schneider (Eds.), Klewer, 2003.
[131] M. Tambe. "Recursive Agent and Agent-group Tracking in a Real-time Dynamic Environment." *Proceedings of the International Conference on Multi-Agent Systems 1995*, 1995.
[132] C. Y. Tang, Y. P. Hung, S. W. Shih, and Z. Chen. "A feature-based tracker for multiple object tracking." *Proceedings of the National Science Council, Republic of China*, Part A, 23(1), 1999.

[133] A. Tews and G. Wyeth. "Thinking as One: Coordination of Multiple Mobile Robots by Shared Representations." *Proceedings of the 2000 IEEE/RSJ International Conference on Intelligent Robot Systems (IROS)*, 2000.

[134] S. Thayer, B. Digney, M. B. Dias, A. Stentz, B. Nabbe, and M. Hebert. "Distributed Robotic Mapping of Extreme Environments." *Proceedings of SPIE: Mobile Robots XV and Telemanipulator and Telepresence Technologies VII*, 4195, 2000.

[135] S. Thrun. "A Probabilistic Online Mapping Algorithm for Teams of Mobile Robots." *International Journal of Robotics Research*, 20(5), 2001.

[136] K. Tumer, A. Agogino, and D. Wolpert. "Learning Sequences of Actions in Collectives of Autonomous Agents." *Proceedings of AAMAS*, 2002.

[137] D. Vail and M. Veloso. "Dynamic Multi-Robot Coordination." In: **Multi-Robot Systems: From Swarms to Intelligent Automata, Vol. II**, Shultz, Parker, and Schneider (Eds.), Klewer, 2003.

[138] M. Veloso, S. Lenser, D. Vail, M. Roth, A. Stroupe, and S. Chernova. "CMPack-02: CMU's Legged Robot Soccer Team." https://www.openr.org/page1_2003/index.html/. 2003.

[139] H. Wang, C. S. Chua, and C. T. Sim. "Real-time Object Tracking from Corners." *Robotica*, 16(1), 1999.

[140] S. Wang and M. Tan. "Multi-robot Cooperation and Data Fusion in Map Building." *Proceedings of the 3rd World Congress on Intelligent Control and Automation*, 2000.

[141] G. Weiß, C. Wetzler, and E. von Puttkamer. "Keeping Track of Position and Orientation of Moving Indoor Systems by Correlation of Range-Finder Scans." *Proceedings of the 1994 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 1994.

[142] D. Wolpert, K. Wheeler, and K. Tumer. "Collective Intelligence for Control of Distributed Systems." Technical Report NASA-ARC-IC-99-44, NASA, 1999.

[143] R. M. Zlot, A. Stentz, M. B. Dias, and S. Thayer. "Multi-Robot Exploration Controlled By A Market Economy," *Proceedings of IEEE International Conference on Robotics and Automation (ICRA) 2002*, 2002.