ISSN 2277-2685
IJESR/April. 2020/ Vol-10/Issue-2/1-10

Exploration Robot Teams Sharing Data

S Sharmila Bhanuy[1], M Prabhakar [2] Y Venkateswara Raju[3], S Jabeen [4]

[1,2,3,4] Asst , Professor, Department of ECE, K. S. R. M College of Engineering(A), Kadapa

Abstract

In this article's abstract, we describe the challenges that arise when a large number of robots attempt to explore and map an unknown region. Hill climbing is used in this mapping technique to locate maps that are most consistent with collected sensor data and odometry. It's a way to maximize your chances of success online. The robots' movements are precisely orchestrated by the program. The goal is to maximize value by minimizing wasted effort as robots learn the same things again. The bulk of the math for both the exploratory and mapping approaches is dispersed. Extensive testing in simulations and real-world trials has been done on the approaches. The results show how our multirobot exploration strategy increases robustness and performance.

Introduction

An essential problem in mobile robots is mapping the surrounding area. In most cases, effective investigation tactics are required for this to be the case. In order to accurately map uncharted regions, robots need guidance on where to focus their efforts and how to best disperse themselves throughout the landscape. Previous mapping efforts mostly focused on using a single robot. However, there are benefits to using a fleet of robots for mapping. The most blatant is that a team of robots can often do a job in less time than a single robot. However, owing to interrobot interference, this may not always be the case [6, 8]. This emphasizes the need of maintaining adequate distances between the robots as a primary goal of the exploration tactics. Having numerous robots has the potential to create more precise maps by combining overlapping data. When the robots have varying sensing and/or localization capabilities, this may assist compensate for the resulting uncertainty and mistake [7]. In this study, we offer methods for directing a team of diverse robots while they perform the job of mapping and exploring a large interior space. We take into account two coordination issues: (1) combining the data collected by the robots' sensors into a single global map, and (2) planning the optimal paths for the robots to take while gathering that data. While an ideal solution to the latter is unachievable, we describe a greedy technique that achieves respectable results in practice. Our fundamental strategy for solving both coordination issues is the same: Make the robots do most of the work and then execute some global calculations over the data to combine their outputs asynchronously. For instance, robots generate a unified neighborhood map by independently processing laser data. After that, a centralized mapper module combines all of the regional maps into one coherent world map. By matching laser scans, local mappers can minimize localization error and hence reduce uncertainty in the data. By merging data from the robots in an iterative process, the central mapper is able to further refine the map (reducing localization error). This presupposes both that the robots have access to high-bandwidth communication and that they are aware of their poses in relation to one another. Our method for organizing investigation similarly blends decentralized processing with worldwide judgment making.

Robots create what are called "bids," which detail the benefits and expenses they anticipate receiving from visiting certain areas. To optimize total usefulness, while minimizing overlap in coverage by the robots, a central executive accepts the bids and allocates jobs. Both scenarios benefit from decentralized computing performed by the individual robots and centralized modules that efficiently aggregate and coordinate data. Following a discussion of relevant prior work, Sections 3 and 4 detail our strategies for producing and navigating multi-robot maps, respectively. In Section 5, we see a real-world example of how three robots worked together to map a large interior space. Quantitative simulation findings demonstrating the impact of our exploratory tactics on task performance are also analyzed. Finally, we talk about some promising future avenues for solving multi-robot exploration and mapping difficulties.

Related Work

There have been a number of ways to mapping and exploration using single robot systems [3, 4, 9, 17, 18], but not many using multiple robots. Several academics have looked at the issue of deploying several robots to lessen the need for localization Error in exploration protected by Copyright 2000, American Association for Artificial Intelligence [10, 13]. For example, the environment in Realities. al. [13] is laid out in horizontal stripes. One robot at a time explores each strip, while the others wait at fixed locations to gather information about the moving robot's location. While this will result in a more precise map, it will slow down the exploring process. On the contrary, in order to maintain their visibility, the robots must stay in close proximity to one another. The graze job, in which the goal is to thoroughly cover an unfamiliar area, was one of the tasks studied by Balch and Arkin [1], who looked at the effects of communication in multi-robot systems on various tasks. The robots explore their surroundings in a quite haphazard fashion. Although we have not conducted head-to-head comparisons of the two approaches, we find that their performance outcomes are qualitatively comparable to what we experience. Advanced methods for using many robots for an exploratory mission are described in [15, 21, 22]. For heterogeneous robots, Singh and Fujimura [15] suggest an online, decentralized strategy. If a robot finds a doorway to a previously uncharted place, but it is too big to fit through it, it will choose another robot that can complete the exploration mission. The candidate robot is selected by balancing the number of locations to be investigated against the robot's size and the direct distance from the robot to the target area. Yamauchi devised a method wherein the robots cooperatively construct a shared map (an occupancy grid) [21, 22]. The concept of a frontier, defined as a place close to an uncharted region of the natural world, is introduced.

This method clusters neighboring cells into boundary zones. The robots each go separately to the next border region's centroid; they may exchange maps, but they don't work together in any other way. Therefore, the robots can wind up traversing the same area and might even collide with one another. In contrast, our method often keeps the robots isolated, which may drastically reduce the amount of time required to complete the mapping process. This paper builds upon our previous work [2] in numerous significant ways. First, the method shown here effectively splits up the computing. This allows the robots to build bids based on their own capabilities (sensor range, trip expenses, etc.), and it allows for the robots' "bids" to be computed in parallel, which helps scaling to greater numbers of robots. Second, unlike previous approaches, the present technique takes into consideration both the state of knowledge about the map and the robots' specific talents when calculating the projected information gain. This paves the way for more nuanced forms of coordination, such as the robots staying close to one another even though the map indicates that they are separated by a complete wall.

Coordinated Mapping

At the core of our approach is a distributed algorithm for concurrent mapping and localization in real-time [19]. The approach makes two major assumptions: First, it assumes the world is reasonably static, and so it cannot handle

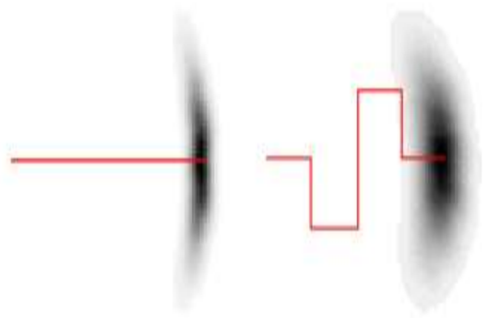
Figure 1: Probabilistic Motion Model Robot

Each diagram is read from left to right, following the solid line. The probabilistic distribution of the robot's posterior position is shown in grey. Darker areas are more likely to be highly inhabited or undergoing radical transformations (such as the disappearance of walls). Additionally, it presumes that the robots are initially visible to one another and are given information about their relative positions (within roughly 1 meter and 20 degrees of direction). The first criterion is used as given in the majority of papers discussing simultaneous localization and mapping [3, 12, 16, 18]. Since the challenge of team-based mapping without beginning pose knowledge is so challenging, it is a good thing that the second assumption holds for many real-world applications. Our method takes a hierarchical, modular approach to decomposing the mapping problem: Each robot has its own map of the immediate area, updating it as necessary to account for odometry errors. The local maps are sent to a centralized module, which then synthesizes them into a world map. The processing needs of the modules are dynamically adjusted in real time to best use the available hardware.

The method's brilliance lies in the fact that it utilizes essentially the same software for both local and global operations. To begin, we provide each robot a set of sensor and odometry readings (laser range scans in this example). From there, it gradually builds three things: maximum likelihood estimates of its own position, the map's (location of surrounding objects), and its "true" location, with the latter accounting for the fact that certain errors cannot be identified when building a map [20]. To further understand the method, let's pretend a robot has built a rough map. It currently seeks to improve the map by adding sensor and odometry data. Our technique optimizes a mixture likelihood function that models (1) the noise in mobility (odometry) and (2) the noise in perception in order to find the robot's most probable location. The motion model is shown in Figure 1. It represents P (s | s', a), the likelihood of being in posture s if the robot was in s' and performed action a (which may have been forward motion or a turn). Assuming that robot motion is noisy along its translational and rotational components, this distribution is derived by solving the (obvious) kinematic equations. The probability function for sensor readings is shown in Figure 2, which represents the perceptual model. The concept here is that sensors won't pick up anything when they previously picked up nothing but empty space. Figure 2's shaded area represents

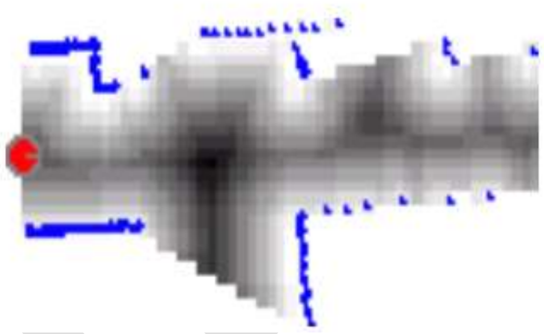

Figure 2: Likelihood Function Generated

The robot is the circle on the left. There are 180 dots in front of the robot, representing the scan. It is less likely that an item may be detected in a dark area. white for areas that are obscured.

the blank area of the displayed scan, where detection probability decreases as greyscale intensity increases. Therefore, the chance of a scan that is properly aligned is much greater than that of a scan that is not. This probability function is distinguishable, which is an important feature. Furthermore, Newton's technique allows for very fast search in the robot's relative pose space (e.g., 1,000 iterations per second). The robot's odometry reading is used as a starting point for our estimation process, and then we apply gradient descent to locate the closest maximum in probability space. This approach swiftly converges to the correct alignment and has a high degree of dependability since maps are constructed sequentially and short-term mistakes are not substantial. The map is the result of compiling all scans with their refined scan coordinates. In order for our mobility planner and exploration module to function, the scan map is efficiently translated into an occupancy grid map [5]. Here, we take on the challenge of mapping using a squadron of moving robots. Using the aforementioned process, each robot constructs its own internal map of the immediate area. Due to a lack of direct communication between the robots, their individual coordinate systems are out of sync with one another. In addition, even if the coordinate systems were properly aligned, the maps normally would not match well owing to residual inaccuracies in the local maps. The data from all of the robots is combined in real time by the central mapper module. With the revised scan coordinates in hand, each robot reports a fraction of its scans (say, every 10th) to the centralized mapper. Therefore, the robots' maps are not used directly. Instead, they are used covertly to generate scan sequences with very low (relative) location errors. The centralized mapper then employs the aforementioned gradient descent approach to reduce the discrepancy between the robots' scans. Since we assume that the robots' starting locations are roughly known, our local search method can locate them properly in relation to one another. Small discrepancies are weeded out when new scans are added and mapped in the same way into the global coordinate system. The

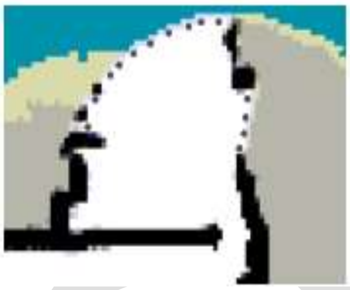

Figure 3: Occupancy Map Used for Exploration “Obstacle” cells are black, “clear” cells are white, “unknown” cells are grey. Frontier cells are marked by small circles.

resulting map integrates every robot’sscan into a single, consistent map with relatively little computation. We have tested our procedure for up to 5 robots, and have no doubt that the same architecture easily scales to 10, or more, robots.

Coordinated Exploration

The objective in coordinating the exploration of multiple robots is to maximize expected information gain (map knowledge) over time. While the optimal solution is computationally intractable, we have developed a relatively low-cost technique that provides good results, in practice. To start, each robot constructs a “bid” consisting of the estimated utilities for it to travel to various locations. The bids are sent to a central executive, which assigns tasks to each robot based on all the bids received, taking into account potential overlaps in coverage. Thus, while a robot may prefer to visit one location, the executive might assign it a different location if another robot is expected to gain much the same information. Robots submit new bids when their maps are updated, which can cause them to be retacked. Exploration ends when there is no useful information to be gained.

Constructing Bids

A robot constructs a new bid each time it receives a map update from the central mapper. It categorizes map cells into three different types (Figure 3) — “obstacle” (probability of occupancy above a given threshold po), “clear” (probability below a threshold pc) and “unknown” (either never been sensed, or probability is between po and pc). A bid is a list of the estimated costs and information gains for visiting various frontier cells. We define a frontier cell as any “clear” cell adjacent to at least one “unknown” cell (Figure 3). For efficiency, we further stipulate that each frontier cell must be at least some minimum distance from all other frontier cells. For instance, even though our grid has 15 cm resolution, we require frontier cells to be at least 30 cm (approximate radius of the robots) from each other. To estimate the cost of visiting a frontier cell, we compute the optimal path (shortest distance, assuming deterministic motion) from the robot’s current position. All costs are

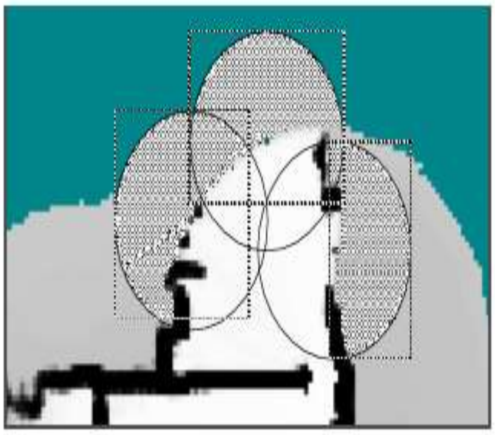

Figure 4: Expected Information Gain Information gain regions for several representative frontier cells. Circles indicate sensor range. Cross-hatched areas are information gain regions. Dotted lines are the rectangular approximations.

computed simultaneously, using a simple flood-fill algorithm [11] that employs an efficient implementation of a priority queue to propagate minimum path costs through the map. To further decrease computation, we consider only “clear” cells, stopping propagation whenever an “obstacle” or “unknown” cell is reached. Estimating information gain is more difficult. In fact, the actual information gain is impossible to predict, since it very much depends on the structure of the environment. Our previous work [2] assumed information gain to be constant for each frontier cell, which tended to make the robots spend too much time exploring nearby areas that were nearly known already. Here we use the current map to provide a more informed estimate. Specifically, we assume that the robot has some nominal sensor range and count the number of “unknown” cells that fall within that radius of the frontier cell, subject to the restriction that the resulting information gain region forms a connected set (Figure 4). For efficiency, we again use a flood-fill algorithm, this time ending propagation when either a “clear” or “obstacle” cell is encountered, or when the distance to the frontier cell is greater than the sensor range. In addition to counting the number of cells, we record the minimum and maximum extent, forming a rectangle that approximates the information gain region (Figure 4).

This enclosing rectangle is used by the executive to estimate potential overlaps in coverage. While there are definite improvements that can be made in estimating information gain (a simple one would be to bias the count by the occupancy probability of the cells, giving less weight to cells that are already partially known), we have found our metric works quite well in practice. In particular, while the metric usually acts to keep the robots well separated, it still allows them to be spatially near one another if there are known obstacles separating them. Thus, we have seen cases where two robots are tasked to explore adjacent rooms — one goes into the first room and perceives the walls, which get added as obstacles in the map. This separates the room from the adjacent room, informationwise, which allows the executive to send the other robot into the second room. This would not be possible with methods that merely try to maintain a given distance between robots.

Assigning Tasks

Each robot asynchronously constructs its bids and sends them to a central executive. The executive tries to maximize the total expected utility of the robots by assigning them tasks, based on their bids. A simple greedy algorithm is used to keep the computation real time. The executive first finds the bid location with the highest net utility (information gain minus cost) and assigns that task to the robot that made the bid. 1 It then discounts the bids of the remaining robots based on the current assigned tasks (see below) and chooses the highest remaining net utility. This continues until either all robots have been assigned tasks or no task remains whose (discounted) expected information gain is above a minimum threshold. Key to this algorithm is the discounting. Without it, the robots would act in an uncoordinated manner, being assigned tasks that they, independently, estimate as best. Our previous work discounted the utility of a location as a function of its distance to the other assigned tasks [2]. Here, we explicitly use the estimated information gain for discounting. Specifically, we estimate the percentage of overlap between the information gain regions by how much the approximating rectangles overlap (Figure 4) and decrease the expected information gain by that percentage:

$$d_j = Area\left(IGR_j \cap \left(\bigcup_{i \in R} IGR_i\right)\right) / Area(IGR_j)$$

$$u_j = (1 - d_j) \times i_j - c_j$$

Here, IGRj is the rectangular approximation of the information gain region for some frontier cell j, the IGRi are the rectangular information gain regions for the assigned robots R, and i j and cj are the expected information gain and path cost of going to cell j. This method is both efficient to compute and a fairly accurate approximation. In one set of experiments, it was within 15% of the true overlap (obtained by counting the actual number of overlapping cells), while being hundreds of times more efficient. The executive is implemented using the Task Description Language (TDL), an extension of C++ that includes syntactical support for hierarchical task decomposition, task sequencing, execution monitoring, and exception handling [14]. When the executive receives a bid, it waits a short while in case other bids arrive. It then assigns tasks to all robots that are either currently unassigned or have submitted new bids (leaving the currently active ones to continue).

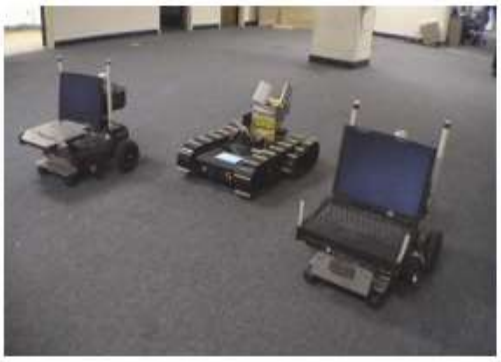

Figure 5: Robin, Marian and LittleJohn

Besides assigning tasks, the executive monitors task execution, interfaces with a remote GUI, and interleaves exploration with other tasks. In particular, at any time the user (through the GUI) can request that one of the robots visit a particular location (e.g., to take a closer look). The executive terminates that robot's current task, reassigns other robots to cover for its loss, assigns and monitors the new task, and then integrates the robot back into the exploration pool when it is finished. One important addition to the task assignment algorithm is the use of hysteresis. If a frontier cell for a robot falls within the information gain region of the robot's currently assigned task, then its expected information gain is divided by the hysteresis ratio (a constant between 0 and 1, usually 0.85). Lower values for the hysteresis ratio will make the executive less disposed to switching tasks. The basic problem is that, because the robot is continuously sensing the environment, the information gain metric can change drastically after only small motions. For instance, by the time a robot maneuvers to position itself in front of a doorway, it typically has seen a large portion of the room. Without hysteresis, entering and completely exploring the room would not have as much utility as going somewhere else. While not the ultimate solution, hysteresis handles the problem fairly well.

Experiments

The multi-robot exploration and mapping system has been tested extensively using a team of three heterogeneous robots — two Pioneer AT robots from RWI and an Urbie robot from IS Robotics (Figure 5). All three robots are equipped with Sick laser scanners that have a 180-degree field of view. The ATs have a 300 MHz on-board laptop running Linux, and all three robots communicate, via Breezecom radio links, with off-board Linux workstations that run the rest of the system, including the mapping, planning, and executive modules. The most extensive testing was in October, 1999 in an empty hospital building at Fort Sam Houston, San Antonio TX, as part of DARPA's Tactical Mobile Robot (TMR) project. During a five day period, we made repeated runs with the robots, mapping large areas of one floor of the building.

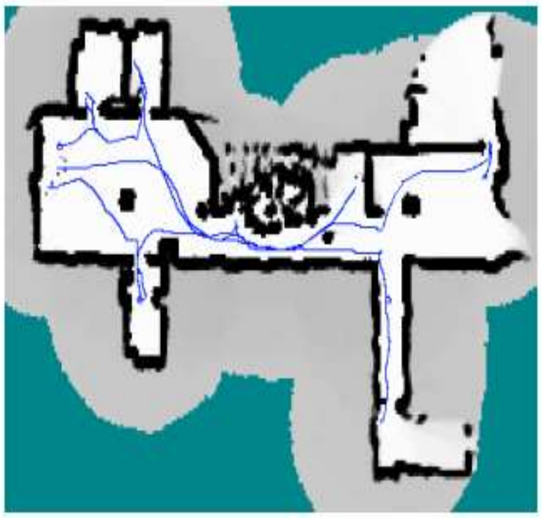

Figure 6: Map Created by Three Robots (62 x 43 m) Robots start at left. The three solid lines indicate the robots' paths through the environment.

Figure 6 shows one typical run that produced a 62 by 43 meter map in about eight minutes. During these runs, we tended to see similar qualitative behavior — one robot would head down the initial corridor, while the other two would explore rooms on opposite sides of the corridor. When the two finished the initial set of rooms, they would move down the corridor to explore openings that the third robot had discovered, but passed by. We also performed tests where we would teleoperate one of the robots while having the other two autonomously explore the areas not visited by the first. Some interesting behaviors were observed that are attributable to the coordination algorithm. For one, if three robots start in the middle of a narrow corridor, two tend to head down the corridor in opposite directions, while the third just waits until one of the others spots a doorway. This is because, initially, there are just two distinct frontiers, and assigning one robot to each leaves the third with no expected information gain. Another behavior, noted earlier, is that the robots sometimes explore adjacent offices that, while spatially close, are disconnected in terms of information gain. Finally, in one instance, we noticed a robot having trouble getting near an office it was tasked to explore. A second robot was tasked to explore further down the corridor. However, at some point the executive swapped tasks, since the second robot had fortuitously gotten closer to the office than the first. Such flexibility in dynamically coordinating tasks gives our system the ability to efficiently explore in a wide variety of situations. To augment the robot tests, we ran experiments in simulation to compare the effects of different numbers of robots in different types of environments. The simulator realistically models the environment and a robot's interaction with it, so that the programs used on real robots can be used with the simulator without modification.

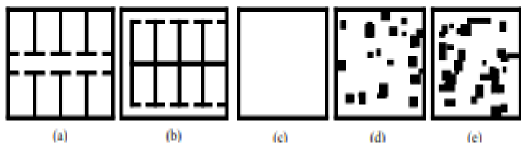

Figure 7: Simulation Environments

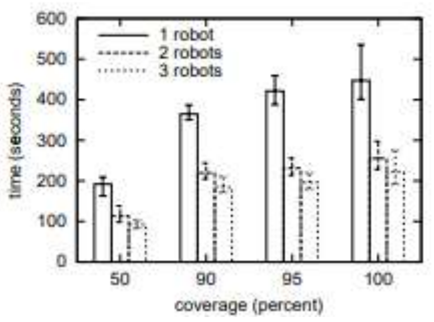

Figure 8: Results from Single-Corridor Environment

Previously, we demonstrated the performance increase that obtains using coordinated versus uncoordinated robots [2]. In the current experiments, we varied the number of robots from one to three, and used five different environments. In the two office-like environments (Figure 7, A and B, 25m x 20m) and the obstacle-free environment (C, 20m x 20m), we ran ten simulations for each number of robots. For the other two environments (D and E, 20m x 20m), we used ten different randomly generated maps, and ran one simulation with each. While our primary performance metric is the time needed to completely explore the environment, it is also of interest to see how the coverage evolves over time. It might be the case that most of an environment is mapped quickly, while it takes a long time to cover a few last spots at the end. For this reason, we report the time it takes to cover 50, 90, 95 and 100 percent of the environment. Figure 8 presents the results for the single-corridor office environment (A). It shows that two robots perform significantly better than one, while there is not much gain in having three robots instead of two. The results are similar for two parallel corridors (B). There, two robots can go in separate directions at the beginning, each exploring one part without any overlap. While a third robot can assist initially in the exploration of one of the corridors, once done it must travel a long way in order to help explore the other corridor. In many cases, the other robots do not arrive in time to help out. In contrast, in the random environments, there is a smaller gain when going from one to two robots and a larger gain when going from two to three robots (Figure 9), compared with the results from the office-like environments. The apparent reason is that the obstacles in the environment help in spreading the robots out.

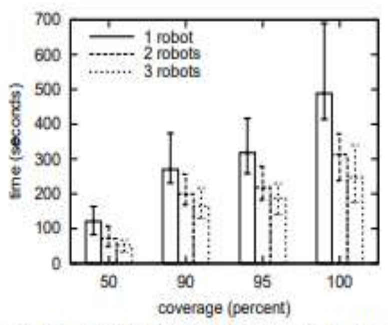

Figure 9: Results from 15% Random Obstacles

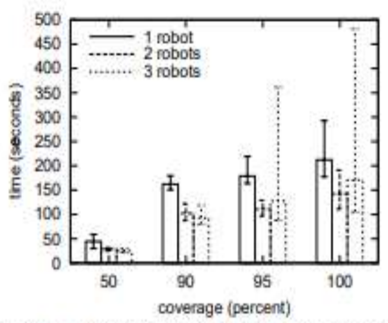

Figure 10: Results from Obstacle-Free Environment

Surprisingly, in the obstacle-free environment, three robots actually take longer to complete the task, on average, than two (Figure 10). This seems to be because they end up interfering with one another [6, 8]. In contrast, in this environment multiple robots have demonstrable positive effects on map accuracy. With few features, the robots get little help in localizing. This has the greatest impact when there is only one robot exploring — in fact, in 30% of the trials the robot failed to complete the mapping task successfully (i.e., the resultant map was qualitatively wrong). With multiple robots, however, the added sensor information helps significantly: For two robots, only one failure was observed (10%) and with three robots, no failures were observed.

Conclusions

While we have extensive test results, we still need to quantify the effects of various design decisions, including the effects of hysteresis, the way information overlap is estimated, and the definition of expected information gain itself. We also intend to quantify the performance of the greedy method of task assignment, comparing it to more sophisticated algorithms such as A* or stochastic search. In both simulation and actual tests, robots are sometimes idle because their discounted utilities are below the minimum threshold. Instead of just staying where they are, they could position themselves strategically so as to minimize the expected distance they would have to travel once they are assigned a task. While for a single robot, a good idle location is one that minimizes the average path cost to all the frontier cells, the problem is much harder if there are several idle robots. Fundamentally, the approach described in this paper is limited in two respects. First, with respect to mapping, we currently assume that the robots begin in view of one another and are told their initial (approximate) relative location. More sophisticated techniques are needed for mapping and localization when the robots need to merge maps where the coordinate transform is initially unknown and the robots need to find out where they are relative to one another. Second, with respect to exploration, we currently assume it is sufficient to consider the utility of exploring a single point. The approach ignores both the fact that information is gained en route and that moving to a given area may facilitate, or possibly hinder, subsequent exploration. We are investigating more sophisticated algorithms that estimate information gain along paths, which we believe will improve overall performance significantly

References

[1] T. Balch and R.C. Arkin. "Communication in Reactive Multiagent Robotic Systems." Autonomous Robots 1, pp. 1-25, 1994.

[2] W. Burgard, D. Fox, M. Moors, R. Simmons and S. Thrun. "Collaborative Multi-Robot Exploration." In Proc. Intl. Conf. on Robotics and Automation, San Francisco CA, May 2000.

[3] H. Choset. Sensor Based Motion Planning: The Hierarchical Generalized Voronoi Graph. Ph.D. Thesis, California Institute of Technology, 1996.

[4] G. Dudek, M. Jenkin, E. Milios and D. Wilkes. "Robotic exploration as graph construction." IEEE Transactions on Robotics and Automation, 7:6, pp. 859-865, 1991.

[5] A. Elfes. Occupancy Grids: A Probabilistic Framework for Robot Perception and Navigation. Ph.D. Thesis, Department of Electrical and Computer Engineering, Carnegie Mellon University, 1989.

[6] M. Fontan and M. Mataric. "Territorial Multi-Robot Task Division." IEEE Transactions on Robotics and Automation, 14:5, 1998.

[7] D. Fox, W. Burgard, H. Kruppa and S. Thrun. "Collaborative Multi-Robot Localization." In Proc. 23rd German Conf. on Artificial Intelligence. Springer-Verlag, 1999.

[8] D. Goldberg and M.J. Mataric. "Interference as a Tool for Designing and Evaluating Multi-Robot Controllers." In Proc. AAAI-97, pp. 637-642, Providence, RI, July, 1997.

[9] B. Kuipers and Y.-T.Byun. "A Robot Exploration and Mapping Strategy Based on a Semantic Hierarchy of Spatial Representations." Journal of Robotics and Autonomous Systems, 8, pp. 47-63, 1991.

[10]R.Kurazume and N. Shigemi. "Cooperative Positioning with Multiple Robots." In Proc. IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS), 1994.

[11]J.C. Latombe. Robot Motion Planning. Kluwer Academic Publishers, 1991.

[12]J. Leonard, H. Durrant-Whyte and I. Cox. "Dynamic Map Building for an Autonomous Mobile Robot." International Journal of Robotics Research, 11:4, pp. 89-96, 1992.

[13]I. Rekleitis, G. Dudek, E. Milios. "Accurate Mapping of an Unknown World and Online Landmark Positioning." In Proc. of Vision Interface 1998, pp. 455-461, Nagoya Japan, 1997.

[14]R. Simmons and D. Apfelbaum. "A Task Description Language for Robot Control." In Proc. Conf. on Intelligent Robotics and Systems (IROS), Vancouver Canada, 1998.

[15]K. Singh and K. Fujimura. "Map Making by Cooperating Mobile Robots". In Proc. Intl. Conf. on Robotics and Automation, 1993