Article

The International Journal of Robotics Research
2014, Vol. 33(4) 519–533

DOI: 10.1177/0278364913515309
ijr.sagepub.com

Coordination strategies for multi-robot exploration and mapping

Carlos Nieto-Granda[1], John G. Rogers III[2] and Henrik I. Christensen[1]

Abstract

Situational awareness in rescue operations can be provided by teams of autonomous mobile robots. Human operators are required to teleoperate the current generation of mobile robots for such applications; however, teleoperation is increasingly difficult as the number of robots is expanded. As the number of robots is increased, each robot may also interfere with one another and eventually decrease mapping performance. As presented here, through careful consideration of robot team coordination and exploration strategy, large numbers of mobile robots can be allocated to accomplish the mapping task more quickly and accurately. We present both the coordination and exploration strategies and present results from experiments in simulation as well as with up to nine mobile platforms.

Keywords

Robot team coordination, multi-robot mapping, multi-robot exploration

1. Introduction

Mobile robots are already widely used by first responders both in civilian and military operations. Today, these robot missions are usually performed by a person through teleoperation. Such a mode of operation challenges the operator as the cognitive load is significant, as was presented in Zheng et al. (2011). There is consequently a desire to introduce some degree of autonomy to reduce the burden on the operator. For design of fully autonomous systems, there is a need to supply the system with a complete map of the environment or to endow the system with methods for automated mapping and exploration. Significant progress has been made on mapping and exploration with single robot systems; a thorough overview can be found in Bailey and Durrant-Whyte (2006) and Durrant-Whyte and Bailey (2006).

Moving from single robot systems to multi-robot teams poses a number of additional challenges. First of all the operator is posed with an added complexity in terms of controlling multiple entities at the same time, which is known to be a challenge (Zheng et al., 2011). In addition, integration of maps generated by multiple robots into a coherent representation is also a challenge. Finally, there is a need to consider how the team members can cooperate to accelerate the exploration of a previously unseen environment. There has been some progress reported on multi-robot mapping as presented in Fox et al. (2006). A number of methods for exploration of spaces have also been presented; see Parker (2008) for a recent summary of related research.

In this paper we consider multi-robot exploration and mapping for larger teams, i.e. multi-robot systems with up to nine team members. The main contribution for this work is an evaluation of different strategies for coordinating the efforts of a robot team during an exploration mission in an unknown environment. Three strategies are presented; these strategies will be referred to as *Reserves*, *Divide and Conquer*, and *Buddy System*. These strategies differ in how *proactive* extra members of the team are when they would otherwise not be needed for the exploration task, i.e. when all potential paths or frontiers are allocated to other team members. The first strategy, *Reserves*, is the least proactive. In this strategy, extra robots will wait in the starting area until they are needed. The second strategy, *Divide and Conquer*, is the most proactive. In this strategy, robots travel in as large a group as possible and split in half when new navigation goals (branches, junctions in corridors) are uncovered. The final strategy, *Buddy System*, represents a compromise between the other two strategies. In *Buddy System*, robots travel in teams of two until new navigation goals are detected and the team will split to follow both paths.

[1]Robotics and Intelligent Machines, Georgia Institute of Technology, Atlanta, USA
[2]United States Army Research Laboratory, USA

Corresponding author:
John G. Rogers III, United States Army Research Laboratory, 2800 Powder Mill Rd, Adelphi, MD 20783, USA.
Email: john.g.rogers59.civ@mail.mil

Mobile robot simultaneous localization and mapping (SLAM) has been thoroughly addressed in the literature; see Bailey and Durrant-Whyte (2006) and Durrant-Whyte and Bailey (2006) for a detailed review of the history and state-of-the-art in SLAM research. The specific techniques used in this paper are based upon the Square Root SAM algorithm presented in Dellaert (2005) and Dellaert and Kaess (2006), which uses methods of linear algebra for solving least-squares systems to compute the map and robot trajectory based on a set of measurements.

Multi-robot mapping based upon the *M-space* representation of Folkesson and Christensen (2004) was presented in Benedettelli et al. (2012). The authors demonstrated how this representation can be used to merge maps across pairs of robots when they rendezvous and establish their relative pose. Our approach to multi-robot mapping is to form the global map on a remote server based upon an initial estimate of the robot's relative starting poses. The robots in Benedettelli et al. (2012) are moved through a predefined trajectory in the environment; whereas, our robots are directed by an autonomous central authority to explore the unknown environment with various collaboration strategies.

Multi-robot mapping and exploration was addressed by Fox et al. (2006) and Vincent et al. (2008). These papers build a map using up to three robots with a decision-theoretic planner that trades off robot rendezvous operations with frontier exploration. These robots rendezvous to determine their relative pose transforms to provide constraints to recover the final map. In contrast, our approach does not require this rendezvous step because landmarks are globally data associated between each robot using a centralized map coordinator. The exploration strategy used is similar to our strategy called *Reserves*; however, we will not use a rendezvous step and do not require a decision-theoretic planner.

In Olson et al. (2012) the authors describe a system which controls a team of up to 14 mobile robots in an urban reconnaissance mission. In this system, a planning algorithm allocates robots to explore navigation goals in an unknown environment and build a map. The exploration strategy used in this paper is also similar to the *Reserve* strategy described in this paper, in that idle robots would remain in the starting area until they are needed. Robots are coordinated in team collaboration behavior to accomplish other goals of the MAGIC 2010 competition, but they do not move together in formations to maximize availability of additional robots as in our *Divide and Conquer* strategy.

In Hollinger et al. (2010), the authors prove performance characteristics on a multi-robot collaboration strategy to perform adversary search. By representing the topological configuration of a map as a graph, the robots can guarantee that the adversarial search will prevent recontamination of previously cleared nodes with an arbitrarily sized team. In Joyeux et al. (2009), the authors describe a distributed system for managing robot plans for performing high-level tasks. This architecture prevents conflicts between robot plans and can handle communication failures. These papers both present strategies and architectures for collaboration between robot agents to perform tasks.

In Simmons et al. (2000) and Burgard et al. (2005), the authors present a coordination algorithm for multi-robot exploration and mapping which tries to maximize information gain through greedy assignment of robots to exploration frontiers. Since the allocation of one robot to an exploration frontier will diminish the predicted information gain to that frontier when allocating other robots, the team will tend to spread out. The authors recognize an important property that their system exhibits of minimizing interference between platforms; one of the key differences between the coordination strategies which we will evaluate in this paper is this property of platform interference. The authors in Simmons et al. (2000) describe a collaboration strategy which is most similar to the *Reserves* strategy, in that idle robots remain until new exploration goals with sufficient predicted information gain require their attention. The authors of Simmons et al. (2000) allude to potential future work where idle robots move to near the exploration frontier to try to be ready for new exploration goals. This proposed future work would be similar to some of the collaboration strategies described in this paper.

We have asserted that each of these previous works employs something similar to the *Reserves* strategy. This is because idle robots do not move proactively to be available for exploration frontiers which are uncovered by active members of the team. In outdoor environments or with moderate team sizes, the entire team might be allocated to points along a broad frontier throughout the exploration process. In indoor environments and with larger team sizes, more robots will be idle and may benefit from more proactive collaboration strategies.

The paper is organized as follows. A presentation of our technical approach is given in Section 2. This section includes details of the implementation and development of our mapping framework, the exploration controller, and the various alternative coordination strategies which will be evaluated in this paper. Our approach has been implemented and evaluated in several experiments. A series of simulated robot experiments are described in Section 3. These simulation experiments allow us to evaluate the collaboration strategies with large groups of robots which would be difficult to manage in real-world experiments due to complexity and expense. The second series of experiments is performed with teams of robots in a variety of buildings. These experiments are presented in Section 4 to establish that these components and strategies are able to deal with real-world issues of sensor and actuator uncertainty. The main technical points and results are summarized in Section 5. Finally, future work will be discussed in Section 6.

2. Technical approach

This section will describe the various software components and algorithms which we have developed to perform

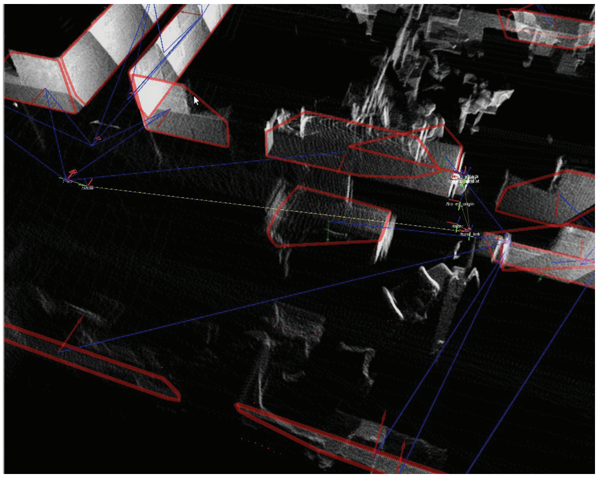


Fig. 1. Example map generated by *Omni-Mapper*. Red polygons are the convex hulls of 3D planes. White point clouds are sensor data rendered from the optimized trajectory for clarity.

autonomous multi robot mapping and exploration. In Section 2.1, we present the mapping library developed in our lab called *Omni-Mapper*. This mapping library makes use of landmark feature measurements, which are given by software modules described in Section 2.2. The robots use a frontier-based exploration strategy which is described in Section 2.3. The coordination strategies used by the team of robots is described in Section 2.4.

2.1. Mapping system

Our mapping system is called *Omni-Mapper*; it is based upon the *GTsam* library developed at Georgia Tech. *GTsam* is a library which uses a graph representation the sparse structure of feature-based mapping to optimize a set of nonlinear measurements between robot poses and landmarks. This library is based upon the Square Root SAM technique from Dellaert and Kaess (2006). *Omni-Mapper* is a *front-end* which interfaces with sensors to take in new measurements and to perform *data-association*, to determine which mapped feature each measurement corresponds to. In addition, *Omni-Mapper* extends the nonlinear factor graph representation of *GTsam* with the *M-space* formulation of Folkesson and Christensen (2004). *Omni-Mapper* is implemented as a framework of plugins which interface with each sensor and can handle multiple types of landmarks at the same time. In our prior work, we have used *Omni-Mapper* to build maps using a variety of types of landmarks such as 2D walls (Rogers et al., 2010), doors (Rogers, Cunningham et al., 2011), 3D walls (planes) (Trevor et al., 2010), and objects (Rogers, Trevor et al., 2011). 3D plane mapping with *Omni-Mapper* can be seen in Figure 1.

In Rogers et al. (2012), we described an extension to the *Omni-Mapper* system for multi-robot mapping. Each robot in the team runs an instance of *Omni-Mapper* with some set of mapping plugins. Each time one of these robots incorporates a new measurement into its map from any of its sensing modalities, it sends the relevant feature measurement information to a remote master map coordinator.

The relevant information which is sent from the robot to the remote master map coordinator consists of as little data as possible to minimize communication overhead. This information is collected into a *map-chunk* message. The extracted feature measurement is one component which is sent; this extracted feature is typically much simpler than the raw sensor data which it is based upon. A plane measurement consists of four numbers for a plane equation together with the convex hull of points on the observed portion of this plane. This plane was originally made of around 10,000 points and can now be transmitted to the master mapper with only around 20 points worth of data. Planes are extracted from Kinect depth images via a RANSAC algorithm which is described briefly in Section 2.2. Additionally, the robot sends its best *odometric* estimate of its pose; this value is used to compare to the previous value to give an initial estimate of the robot's motion to the global mapper. An *odometric* estimate is used because it is a smooth value and will not jump due to loop closures on the individual robot. For small motions, odometry is a good estimate of the robot's relative pose to the previous measurement taken. Finally, the individual robot's posterior estimate of where it thinks it is in its own *local map frame* is sent to the master map coordinator. The master uses this estimate to produce a correction to establish the relationship between the *global map frame* and each of the *local map frames*. Since the master mapper knows the transformation between the global map frame and each robot's local map frame, it translates all motion commands into the robots local frames of reference before they are sent to the robots.

Map-chunk messages are sent from each robot to the central map coordinator whenever a new measurement is incorporated into an individual robot's local map. The experiments presented in this paper are performed with a wireless network consisting of a single portable infrastructure router. In the future, a dynamic mesh topology network structure would be more appropriate for a multi-robot exploration and mapping as operating environments progress to large scale; however, in the buildings tested for these experiments, a single wireless node maintains connectivity well.

Map-chunk messages contain sequence information so that dropped messages can be detected and re-sent. When an unexpected message with a sequence number m which is greater than the expected sequence number n is received at the master map coordinator, it sends a request back to the robot which sent the out-of-sequence message for all messages in the range $[n, m]$. If the robot does not receive this message, it will continue and send message $m + 1$ to the master, which will then send another request for all messages from $[n, m + 1]$. In this way, if a robot ever returns to communications range, all relevant messages will be recovered in the map coordinator. Even without this resending mechanism, the master map coordinator would be fault-tolerant to missing data, since new messages contain odometry relative information so it can be placed relative to the previous message. The range of missing data for which the master map coordinator would be fault-tolerant is dependent on the accuracy of the robot's odometry to place the new information relative to what has been seen previously. In the current centralized server architecture, if a robot has a network failure and cannot communicate with the server, it will cease operation when it arrives at its currently assigned exploration frontier, due to the fact that it will be unable to receive new goals. If the central server does not receive information from the robot with lost communication within one minute, then that robot's exploration goal will be reallocated to another team.

The master map coordinator is also a modified version of the *Omni-Mapper* system. The system has been modified to track and maintain a trajectory for each robot in the team instead of only tracking a single robot's trajectory. A plugin module was built which takes a *map-chunk* message from the members of the robot team. The master map coordinator data associates measurements from each of these *map-chunk* messages into a single global set of landmarks. This implementation is done in such a way so as to allow for a potentially unlimited number of team members to build a global map.

Currently, the map coordinator must be given an estimate of the initial formation of the robot team. Landmark measurements are fused in the map coordinator into a single set of landmarks; an estimate of the initial formation is used to determine which measurements are of the same landmarks in the environment.

A more typical modern SLAM approach would be to build a *pose-graph* (Grisetti et al., 2007). This type of representation would use 2D or 3D *iterative closest point* (ICP) to determine the rigid body transformation between two sensor views of the environment. This type of system would not require the representation of landmarks and feature-based measurements; therefore it could admit a significantly simpler implementation. This approach is effective on single-robot mapping; however, it has some drawbacks which prevent its use in our multi-robot mapping application. Performing rigid alignment of sensor data across multiple platforms would require transmission of some large point clouds. We are assuming that our robots are going to be deployed in an environment where communication is limited due to large-scale mesh topology or poor channel characteristics. Our assumption is that a multi-robot mapping system must minimize the amount of data which is transmitted since bandwidth would be limited. Also, these sensor alignment operations are computationally intensive especially for real-time implementation. Bandwidth usage is minimized by summarizing complex sensor information into a relevant feature measurement before transmitting it to the master map coordinator.

2.2. Sensor measurements

Individual robots measure 3D planar wall structures using the Microsoft Kinect sensor. A RANSAC (Fischler and Bolles, 1981) based algorithm is used in the point-cloud-library (PCL) (Rusu and Cousins, 2011) to extract planar wall structures. More details on the implementation of the wall extraction algorithm can be found in Trevor et al. (2010) and Trevor et al. (2012).

Planar landmark measurements are made using the Kinect on each robot. This measurement is useful for localization and mapping; however, it is not ideally suited for exploration or navigation due to the presence of non-planar clutter and other robots in the scene. To permit safe navigation, a *laser scan* is synthesized from the Kinect data by looking at a strip of the depth image corresponding to the robot's height. The closest range within this strip at each bearing, corresponding to a vertical stack of pixels, is collapsed into a 2D measurement analogous to a laser scan. The narrow field-of-view of the Kinect sensor on each robot caused a number of problems in our system. First, it was very easy for robots to not be able to see each other and accidentally collide, by approaching each other just outside the field-of-view of their sensors. Second, the frontier-based exploration algorithm needs to be able to see in a wide field-of-view; otherwise, it will focus the robot exploration on regions right next to each robot instead of further out into the environment. To alleviate these problems, we incorporated a controller which subsumes the control of each robot after it has moved a certain distance, between 1 and 2 m. This controller causes the robot to execute a slow 360° rotation-in-place. During this rotation maneuver, the sensor information from the Kinect will be populated into a 360° laser scan type measurement. This 360° laser scan measurement is sent to the local mapper. When the local planner gets one of these measurements, it immediately creates a trajectory element and records the scan. This scan is used to build an occupancy grid map which is used by the local robot's motion planner for obstacle avoidance and navigation. The local mapper then sends this 360° laser scan message in a *map-chunk* message to the master map controller where it is combined with data from other robots to build an occupancy grid which is analyzed to find exploration frontiers.

The *map-chunk* message with the 360° laser scan is received in the master map coordinator where it is used to build a global cost-map together with the laser scan messages from the other robots. This global cost-map is used to select goals for the robots in the frontier-based exploration controller, which is described in Section 2.3. It should be noted that sending the 360° laser view does amount to sending more than just feature measurements to the master map coordinator. This message does occur at a lower rate than feature measurements. Plane feature measurements could be used to generate the occupancy grid representation used for exploration; however, at this time the 360° laser

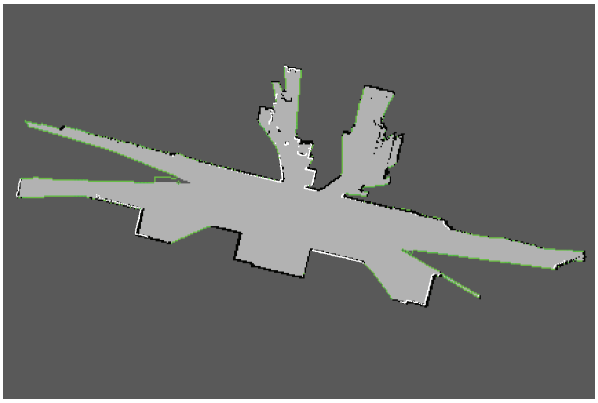


Fig. 2. An example cost-map representing the exploration frontier. Green cells represent the exploration frontier between *Unknown* and *Clear* grid cells. Black cells have the *Obstacle* label.

scan was needed to get enough detail in the many types of environments used in this paper.

2.3. Exploration strategy

For each of the coordination strategies which are described in Section 2.4, the master node selects navigation goals in a frontier-based exploration strategy first described in Yamauchi (1998), and similar to the one used in Vincent et al. (2008). The frontier-based explorations strategy uses a cost-map representation where grid cells can have one of three labels: *Clear*, *Obstacle*, and *Unknown*. The entire cost-map is initialized with the *Unknown* label. Sensor data is added to the cost-map from the 360° laser scan messages described in the last section by setting the grid cell in which each laser beam ends as *Obstacle* and clearing all cells between this new *Obstacle* and the robot's current location to the *Clear* label. An exploration frontier is defined as a grid cell which is marked as *Clear*, but it is adjacent to at least one cell which is still marked *Unknown*. An example of the exploration frontier is shown in Figure 2. This represents the state of the exploration frontier after the robots have performed some initial exploration.

The navigation goals are defined as clusters of cells which meet the criteria for being on the exploration frontier. The coordination strategies, which are described in the next section, choose which size teams are formed to explore each of these navigation goals. We have chosen a greedy approach in each of these coordination strategies where the (robot team asset, navigation goal) pairs with the shortest distance are selected first instead of a traveling salesman type strategy where all future exploration goals are assigned optimally from the initial position. We have chosen this greedy strategy because the act of exploration will uncover additional exploration goals which are not available when the robots start moving. Re-planning will be needed as the robots uncover more of the exploration frontier.

2.4. Coordination strategy

Autonomous multi-robot mapping systems contain two unique design aspects: how local map data is integrated into a global map, and how assets are allocated to explore the environment. Assets can be allocated by either an uncoordinated strategy where each robot behaves as if it were mapping on its own. This uncoordinated strategy is clearly suboptimal because robots will explore the same regions and will not cover the area much more quickly than a single robot. The alternative that we explore in this paper is a coordinated strategy where robots work together to explore the environment.

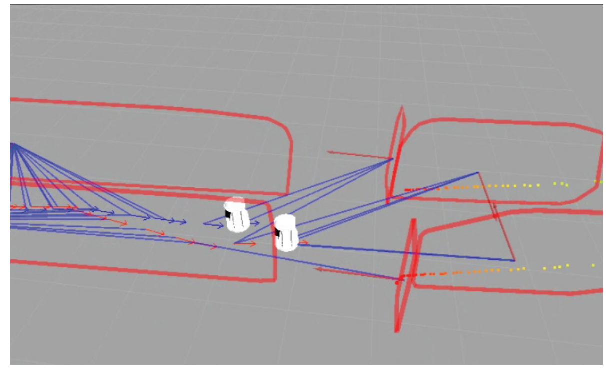
The coordination strategy used between robot agents as well as the number of robots are the independent variables in the experiments performed in this paper. The coordination strategy refers to the proportion of robots which are dispatched to each exploration goal. On one extreme, a single robot can be sent to explore a new goal; at the other extreme all available robots can be sent to a new goal. Larger robot teams sent to a new exploration goal will improve availability of new agents at the location of new exploration goals that are discovered. The larger group has spare robots which can be quickly allocated to explore new goals, such as those discovered when the team moves past a corridor intersection or T-junction. If the group of robots allocated to a navigation goal is too large, then the robots can interfere with each other due to local reactive control of multiple agents with respect to dynamic obstacles and limited space in corridors. The strategies selected for testing trade off *availability* (robots are close and able to explore branching structure quickly) with *non-interference* (robots do not get in each other's way).

The first coordination algorithm is called *Reserve*. In this algorithm, all unallocated robots remain in the starting location until new exploration goals are uncovered. When a branching point is detected by an active robot, the closest reserve robot will be recruited into active status to explore the other path. This strategy has low availability because all of the reserve robots remain far away at the entrance; however, it has minimal interference because the exploring robots will usually be further away from other robots.

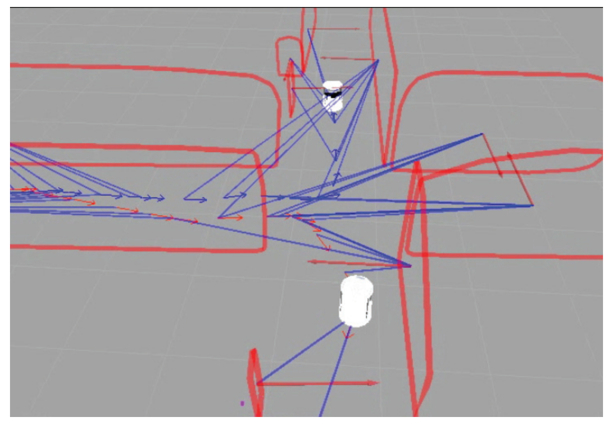
The second coordination algorithm is *Divide and Conquer*. In this strategy, the entire robot group follows the leader until a branching point is detected. The group splits in half, with the first $\lceil \frac{n}{2} \rceil$ robots following the original leader, robot $\lceil \frac{n}{2} \rceil + 1$ is selected as the leader of the second group, and robots $\lceil \frac{n}{2} \rceil + 2$ through n are now members of its squad. Once there are n squads with one robot, no further divide operations can be made and new exploration goals will only be allocated once a robot has reached a dead-end or looped back into a previously explored area. When a robot or group encounters a dead-end, it will join the closest group that is currently actively exploring. This algorithm maximizes availability, but potentially causes significant interference between robots.

The third coordination algorithm is called the *Buddy System*. In this strategy, robots are recruited from the reserve pool in teams of two. When a branching point is detected by a full team of two robots, the team will split into two and proceed along both paths. When these single robots detect additional split points, new teams of two robots will be allocated out of the reserve pool and they will explore this new goal and divide when another branching point is reached. When a robot which has already split from its "buddy" reaches a dead-end without any new known navigation frontiers, it will join another lone robot who is still actively exploring. This strategy uses small teams of robots which are able to maneuver around one another without too much interference, while maintaining good availability to respond quickly to explore new frontiers.

An example 3D map built by two robots as they approach a branch point can be seen in Figure 3(a). At this point, the robot team splits and each team member takes a separate path, as seen in Figure 3(b). The map shown is built concurrently with local maps built on each robot. The global map is used to establish a global frame of reference for robot collaboration message coordinates.



(a) Two robots approach the intersection.



(b) Two robots split and move past the intersection

Fig. 3. An illustration of the *Divide and Conquer* exploration strategy with a team of two robots. As the robots approach an intersection, the team splits to explore two of the new paths. The third remaining path will be explored when one of the robots hits a dead end. Red polygons represent the convex hulls of 3D planes in the merged master map. Blue lines represent measurements between robot poses and 3D planes. The robot team is illustrated by two small white and black TurtleBot robots.

2.5. *Implementation*

We use the Robot Operating System (ROS) from Quigley et al. (2009). ROS provides interprocess communication as well as coordination of sensor data with pose information. Our robot algorithms are implemented as a distributed set of programs which run in the ROS system. In addition, we make use of several implementations of common mobile robot software components which are provided in the ROS distribution. Motion planning is provided by the *move base* framework which uses *TrajectoryPlannerROS* for long-range motion planning and local obstacle avoidance. Low level platform control is handled by the iRobot Create driver for the TurtleBot in ROS. The TurtleBot IMU and odometry are fused in the *robot pose EKF* to estimate platform motion.

3. Simulation experiments

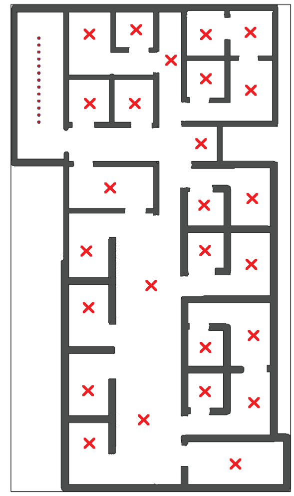
We first evaluated our collaboration strategies in simulated environments with the *Stage* simulator (Gerkey et al., 2003) in which the robot can explore and navigate.[1] We then performed live robot experiments in multiple environments including an office building and a training facility. Results from live robot experiments will be presented in Section 4.

Three different scenarios were chosen for these simulation experiments. In each of these scenarios, the robots were introduced in a single entrance from an initial formation to simulate a breach entrance into a hostile environment. The first scenario is shown in Figure 4(a); this scenario is designed to be typical of a home or office environment. There is a combination of open space and smaller rooms. Many rooms are connected via main corridors, but some rooms are connected to each other through other side passages. Since this environment has a lot of maneuvering room and early opportunities for exploration in multiple directions, we expect that exploration will be accelerated as robot team size is increased. Availability of additional robots will also be important in this environment, since new exploration frontiers will be uncovered quickly as exploration proceeds; therefore, we expect the *Divide and Conquer* strategy to perform well.

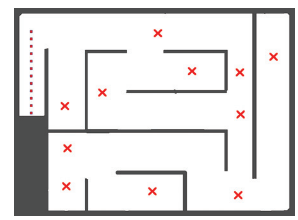
The second scenario, which is modeled after a simple maze, is shown in Figure 4(b). Again, the robot team is introduced from an initial formation at an entrance point to the maze. Since this is a relatively simple maze, it has a low branching factor and will not benefit large robot team sizes. Availability of additional robots will be helpful, but due to the limited number of points where additional robots are needed for exploring branching structure, *Divide and*

Conquer should enjoy less of an advantage over the other strategies.

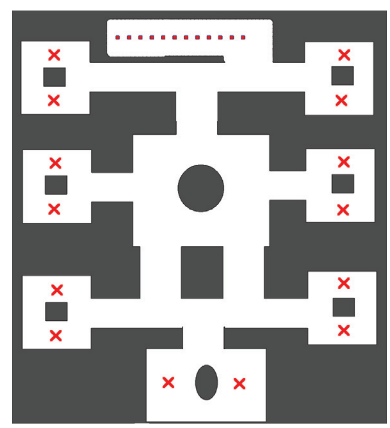
The third scenario is shown in Figure 4(c). It consists of a series of moderately-sized rooms with large obstacles connected by long corridors. This structure is meant to represent an underground base where the rooms are placed down long corridors to provide separation from the entrance and protection from assault. In this environment, the robots will have to travel longer distances before branching points, so the availability of additional robots in *Divide and Conquer*, as well as *Buddy System* to a lesser extent, should be helpful for exploring the map quickly. Large team sizes may not be that helpful in this environment, since teams should be able to re-form and continue the exploration after vising the rooms while other team members are still proceeding down the next section of corridor.



(a) Simulated environment *S1*: Rooms world. This map is meant to represent a home or office environment.



(b) Simulated environment *S2*: Maze world. This is a maze type structure with a relatively low branching factor.



(c) Simulated environment *S3*: Lair world. This environment is meant to simulate an underground bunker with high branching factor and many corridors.

Fig. 4. Simulated maps used to test coordination strategies on various types of environments within the *Stage* simulator. Robots are shown as a line of red dots in their initial starting area which represents a breach entrance. Navigation keypoints are illustrated with red “X” marks.

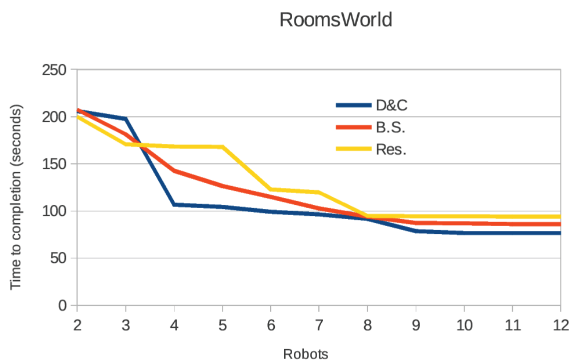
3.1. Simulation results

Simulation test runs were launched automatically from a script. Three computers were used for running simulations; to keep machine performance from affecting timing results, all runs for each scenario were run on one machine. A checking program was implemented which monitors the robots and compares their progress through the exploration procedure by referring to a set of ground-truth *navigation keypoints*. When a robot passes within a 1.5 m threshold of a navigation keypoint, this event is recorded by the checking program. When all navigation keypoints have been visited by the robot team, the exploration task is complete and the checking program records results and allows the script to start the next test run.

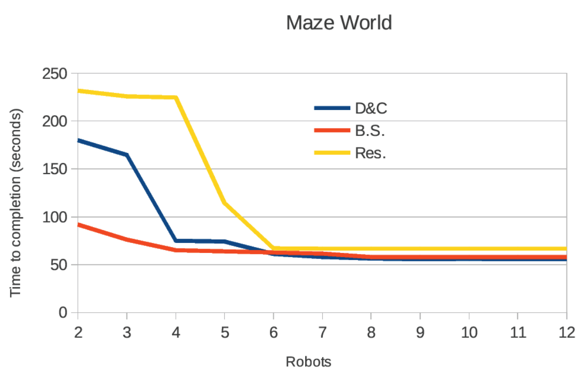
This system of navigation keypoints is used to get a reasonable measure of when the robot team has completed the exploration and mapping task. Navigation keypoints are selected such that if all of them are uncovered, then the environment is fully explored, and vice-versa.

Total exploration time results for these three simulation scenarios are presented in Figure 5. Each of these graphs presents the time taken to fully explore the environment with a team size from 2 to 12 robots and each of the three strategies *Divide and Conquer*, *Buddy System*, and *Reserves*. Since there is no collaboration with a single robot, there is not any difference between the performance of each strategy. For this reason, exploration timing for a team with a single robot was omitted.

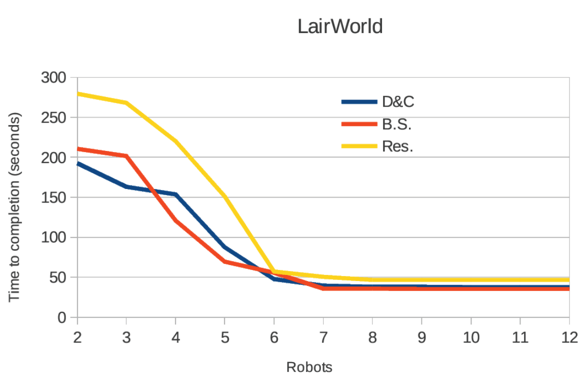
For the *Rooms World* simulation scenario shown in Figure 4(a), the results can be seen in Figure 5(a). *Rooms World* is meant to represent a typical domestic or office environment with a mixture of larger and smaller spaces separated by both corridors as well as some direct connections. For all three strategies, the time taken to fully explore the map is reduced as the team size is increased initially, but then plateaus above nine robots. We believe that this is due to the fact that this is still a relatively simple scenario and large teams of robots are not necessary to proceed through the map. In scenarios *Divide and Conquer* and *Buddy System*, we noticed that robot teams often stayed together as they proceeded around structures, such as in the upper-right and lower-right corners of Figure 4(a). These rooms often did not present more than one exploration frontier at a time, and therefore additional robots were not very helpful in pursuing alternate frontiers. *Divide and Conquer* seems to maintain its performance at larger team sizes better than *Reserves*. This indicates that in a typical domestic or office environment, the additional availability of additional robots in a *Divide and Conquer* strategy is helpful for exploration. *Divide and Conquer* outperforms other strategies for almost all robot team compositions. In this scenario, using more than not robots is not helpful in terms of exploration time;



(a) Results from environment *S1*: Rooms world



(b) Results from environment *S2*: Maze world



(c) Results from environment *S3*: Lair world

Fig. 5. Times taken to fully explore each simulation environment with varying team size, by strategy. A map is fully explored when every location marked with a red "X" from Figure 4 has a robot approach within a 1 m radius.

however, this is the largest environment and the other two test scenarios will plateau with a smaller team size.

The results for the *Maze World* scenario illustrated in Figure 4(b) are shown in Figure 5(b). In this environment, additional robots do not have much of an advantage over smaller team sizes beyond six robots. This is likely due to the fact that though this is a maze environment, it does not exhibit significant branching factor; additional robots are not needed beyond the number of exploration frontiers which are open at a given time. As the robots explore this maze, the number of exploration frontiers which are open at a time is around this number, so it makes sense that additional robots beyond four would not be very helpful, as is seen with the more proactive strategies *Divide and Conquer* and *Buddy System*. *Divide and Conquer* and *Buddy System* appear to have an advantage in exploration time over *Reserves* with smaller team sizes. This map is relatively small, so robots are able to move to new frontiers quickly, and having additional robots available only saves this small amount of additional time. Due to the small branching factor in this scenario, high availability collaboration strategies like *Divide and Conquer* allocate resources more efficiently when they are scarce with smaller team sizes, but this advantage disappears as team sizes grow.

The results for the *Lair World* simulation environment from Figure 4(c) are shown in Figure 5(c). This environment is comprised of a series of corridors which attach rooms to a central room. In this simulation, additional robots seem to not improve performance beyond 7 robots. *Buddy System* outperforms other strategies, which indicates that each robot encounters a single branch point by the time that a new team is available to re-form the teams. This result is also apparent in the results for *Maze World* shown in Figure 5(b).

This series of simulations has demonstrated that the performance of the collaboration strategies which we have presented is dependent on the topology and geometry of the environment. Strategies such as *Divide and Conquer* keep additional robots available to explore branching structure; however, these strategies also generate more interference between robots. *Divide and Conquer*, and, to a lesser extent, *Buddy System* require robots to navigate close to each other. With such close proximity of other robots, navigation is more cautious to avoid collision. In smaller environments, the risk of collision is higher and larger teams of robots interfere with each other more. Also, the utility of these types of strategies is dependent on the degree of branching structure exhibited by the environment during exploration.

4. Live robot experiments

The setting for the multi-robot mapping task for this series of experiments consists of a team of robots being introduced into a single entrance in an unknown environment. Each robot is an inexpensive Willow Garage *TurtleBot*; a team of nine of these robots is shown in Figure 6. The *TurtleBot* was chosen for this application due to its low cost and the ease of integrating large numbers of robots through ROS. The *TurtleBot* platform is based on the iRobot *Create* base. The robots make measurements of planes with a Kinect sensor, and use an onboard IMU together with odometry to estimate ego-motion.

We evaluated the performance of various robot coordination strategies in the multi-robot exploration and mapping task. An example scenario for the *Divide and Conquer* cooperative mapping strategy can be seen in the panorama

Fig. 6. Our nine TurtleBots used in these experiments.

image in Figure 7. Live-robot experiments were performed in two settings: an office environment, and a training facility consisting of several buildings simulating a small village. At the time when the office experiments were performed, only the *Divide and Conquer* and *Reserves* strategies were available; the *Buddy System* strategy was not tested here. The experiments in the office environment are presented in Section 4.1. All three strategies were tested in many of the buildings in the simulated village training facility, these experiments are presented in Section 4.2.

4.1. *Office environment*

In the first series of live robot experiments, we evaluated the first two strategies which were developed, the *Reserves* and *Divide and Conquer*. This series of experiments was designed to evaluate the performance of these two cooperative exploration strategies. A total of six runs were performed for each cooperation strategy, team size, and starting location. For each experiment run, the *TurtleBot* team explored the environment from a wedge-shaped starting configuration, which can be seen in Figure 6. These experiments were performed in an office environment. In order to measure the exploration and mapping performance in each location, we chose specific starting locations which are labeled *Base1* and *Base2* in Figure 8. These initial configurations were chosen to represent a breaching action where mobile robots would be introduced into a hostile environment.

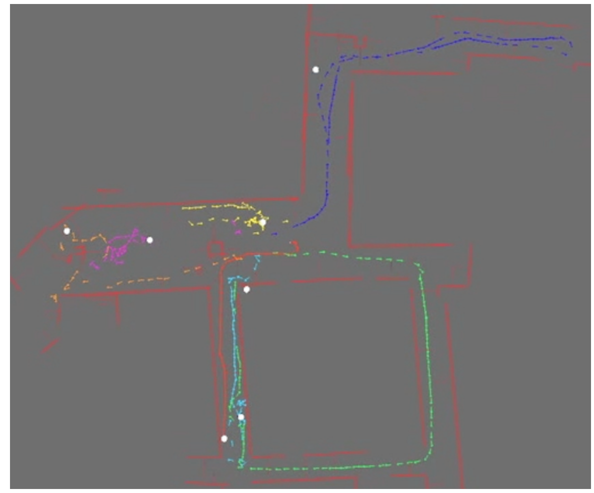
The first series of experiments demonstrate team performance based upon coverage in a mapping task on an unknown office environment. Robot team sizes were varied from two to nine robots. A map built with seven TurtleBots using the *Reserve* strategy is seen in Figure 9(a). An image showing the same final global map from a side view demonstrates the 3D plane features in Figure 9(b).

Each of the collaboration strategy and robot team size experiments were performed from two starting locations. These starting locations are labeled *Base1* and *Base2* in Figure 8. A series of interesting locations was determined in advance by examining the building floor-plan; these points of interest are also marked in Figure 8. Each experiment run gets a score based on how many of these points of interest are visited and mapped before a time limit is reached. This score represents the effectiveness of that algorithm and team size at exploring the entirety of an unknown map.

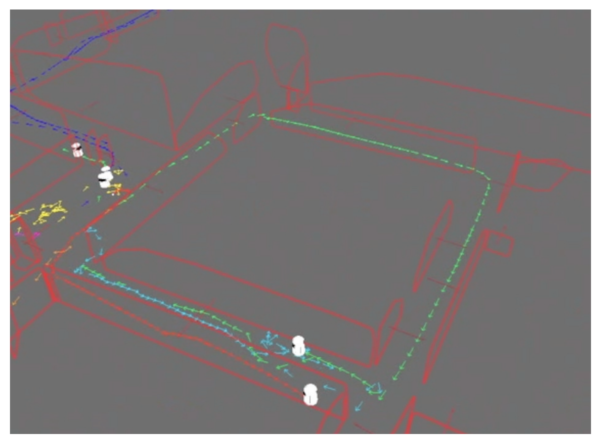
In the first experiment series from *Base1* in Figure 8, both strategies achieve reduced exploration coverage per robot as the team size is increased, as can be seen in the graphs in Figure 10. In this starting location, there is limited space to maneuver, so both strategies generate significant interference between robots trying to move to their goals. In several instances, pairs of robots even crashed into each other due to the limited field-of-view of their sensors. We believe that the *Divide and Conquer* strategy results in Figure 10(b) indicate that the team was slightly more effective than the *Reserves* strategy in Figure 10(a). At the largest

Fig. 7. An example scenario for the experiments described in this paper. Three teams of two robots are exploring the branching hallway structure in an office environment. In this illustration, the robots are using the *Divide and Conquer* cooperative mapping strategy.

Fig. 8. Our office environment where the experiments were performed. The areas labeled *Base1* and *Base2* are the initial position of the robots. Red lines indicate artificial barricades to restrict the initial exploration of the robot teams to simulate a breach entrance into a hostile environment. Blue squares indicate the location of navigation keypoints used for evaulating test runs, as described in the text.



(a) A map built by seven robots in an experiment using the *Reserve* cooperative mapping strategy.



(b) The same map shown from a different angle to demonstrate 3D plane features which are used for map landmarks.

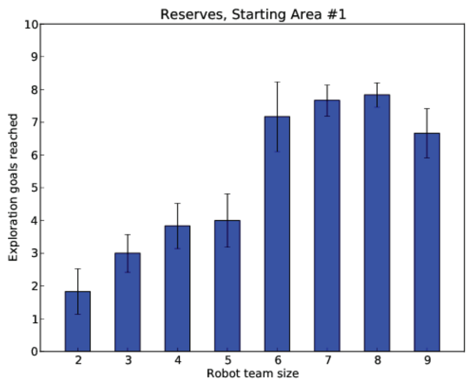
Fig. 9. Global map gathered by a team of seven mobile robots using the Reserve strategy. Figure 9(a) shows a top-down view after a loop closure is detected at the large loop. Figure 9(b) is the same map viewed from an angle to show the extent of the 3D walls.

team size of nine robots, the *Divide and Conquer* strategy usually visited one additional point-of-interest more than the *Reserves* strategy. Additional qualitative impressions are that the *Divide and Conquer* strategy explored the points-of-interest that it reached more quickly than with the *Reserves* strategy. For both strategies, the best team size appears to be eight robots in this starting location, with a sharp increase at six robots.

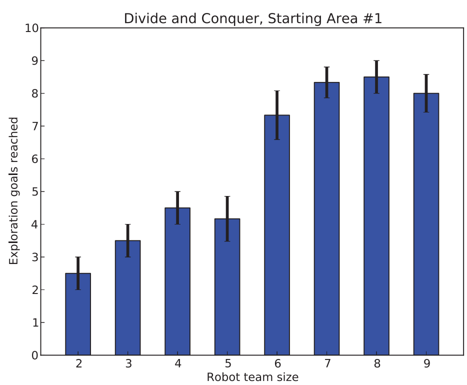
In the second set of the first series of experiments, the robot teams were placed in the starting area labeled *Base2* in Figure 8. As in the first experiment, the per-robot performance of both strategies decreased as the number of robots were increased. This series of experiments demonstrates a marked improvement of the *Divide and Conquer* strategy over the *Reserves* strategy, as can be seen in Figure 11. The *Divide and Conquer* strategy causes more robots to be making observations of exploration frontiers due to the fact that groups contain more than one robot. These additional observations of the frontier allow the *Divide and Conquer* strategy to find exploration frontiers faster than the *Reserves* strategy, and therefore explore more points-of-interest. The second experiment started from an area where there is more room to maneuver. This allowed the *Divide and Conquer* strategy to have less interference since the entire team moved together out of the starting area into the larger area before any divide operations were performed. The *Reserves* strategy still had to initially maneuver from the cramped starting location. As in the first experiment, the *Divide and Conquer* strategy qualitatively explored the environment faster than the *Reserves* strategy. The *Divide and Conquer* strategy behaved similarly with the first experiment, with increased performance with additional robots up to the limits of our study, while the *Reserves* strategy exhibited reduced performance due to interference with team sizes above six.

4.2. "Village" training facility

In the second series of live robot experiments, we evaluated all three collaboration strategies *Reserves*, *Divide and Conquer*, and the new strategy *Buddy System* in various buildings in a training facility designed to simulate an urban environment or village. Due to the remoteness of this training facility, we only brought five robots for testing. The robot team size is varied from three to five robots. We

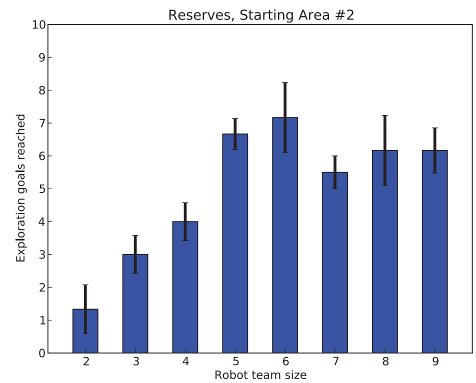


(a) Reserves

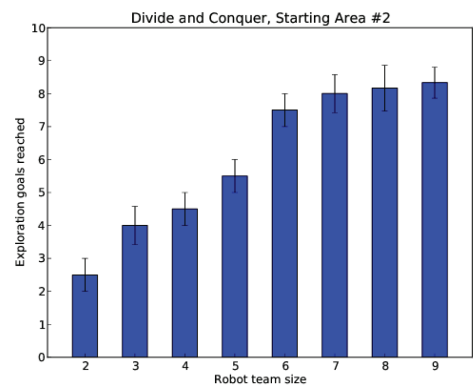


(b) Divide and Conquer

Fig. 10. Results from the first starting area.



(a) *Reserves*



(b) *Divide and Conquer*

Fig. 11. Results from the second starting area.

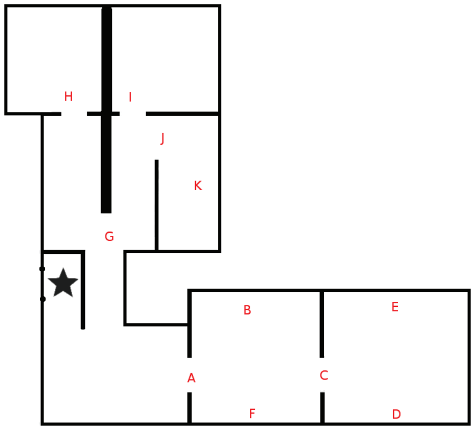
attempted to run each test three times, though some runs were not completed due to scheduling constraints.

The procedure used for the experiments at the training facility differs somewhat from the experiments at the office building. In these experiments, we evaluated robot exploration performance using the same metric as in the simulations from Section 3. In the experiments in the training facility, we allowed the teams to take as long as needed to explore every keypoint. The amount of time elapsed to fully explore each building is the metric used for comparing strategies in these experiments.

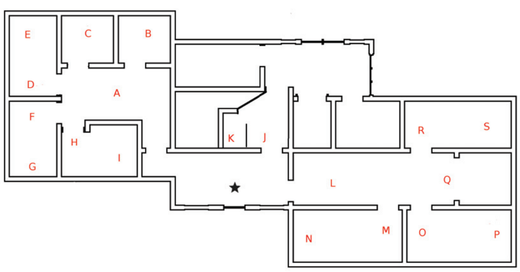
Architectural floor-plans from the buildings explored in this experiment are shown in Figure 12. The set of keypoint locations were manually labeled by room separation. Some of these locations are difficult to reach given the physical capabilities of the robots. In Figure 13, the robot needs to navigate through a narrow hallway in order to explore the rest of the floor. In Figure 12(b), the robot needs to reach the goals labeled *P* and *O* in order to complete the exploration of the room.

These experiments were performed with teams of three, four and five robots. Figure 14 shows the initial configuration of one of the experiments. This initial configuration needs to be set in order to fit the robots close to a door. This is meant to simulate an exploration task for a rescue mission which starts by introducing the robots through a doorway into the building. As shown in Figure 15, the robots explore and navigate in an environment which exhibits difficult lighting conditions; however, this is not a problem for the Microsoft Kinect sensor.

Our results are summarized in Tables 1 and 2 and show the time that each group of robot takes to explore the buildings. The longest trial lasted up to 42 min, which occurred when evaluating the *Reserves* strategy in Building A. In comparison with the simulation experiments, the implementation of real robots is subject to hardware failures and a hostile environment with debris and small potholes. We attempted to run these experiments in a third building at this training site, but the TurtleBots kept getting stuck on concrete seams and other ground defects. The ground was smoother in Buildings A and B; the robots were able to move freely. These two buildings present a better view of how these strategies and team sizes perform on this task, given that a real-world robot would be designed to handle the terrain upon which it is expected to operate. The live robot runs take significantly more time to complete than the simulation runs for two reasons: the top speed of the robots is limited to avoid collision, and the 360° rotation maneuver (see Section 2.2) is not used in the simulations.



(a) Building *A*.



(b) Building *B*.

Fig. 12. Floor maps, keypoint locations, and starting areas for the experiments performed in the "Village" training facility. Keypoint locations are denoted by capital letters and the starting area is denoted with a star.

Fig. 13. An example of the complexity of the environment where the robots need to navigate through the hallway in order to continue to explore the floor.

Fig. 14. An example of the initial configuration of the robots in Building *A*. This configuration simulates the robots entering from the door.

Fig. 15. Here, the robot team can be seen exploring Building *B*.

Table 1. Time in seconds for each exploration strategy on Building *A*. Each experiment was run three times, median values are shown in bold. The *Reserves* strategy was only run once with three robots on this building.

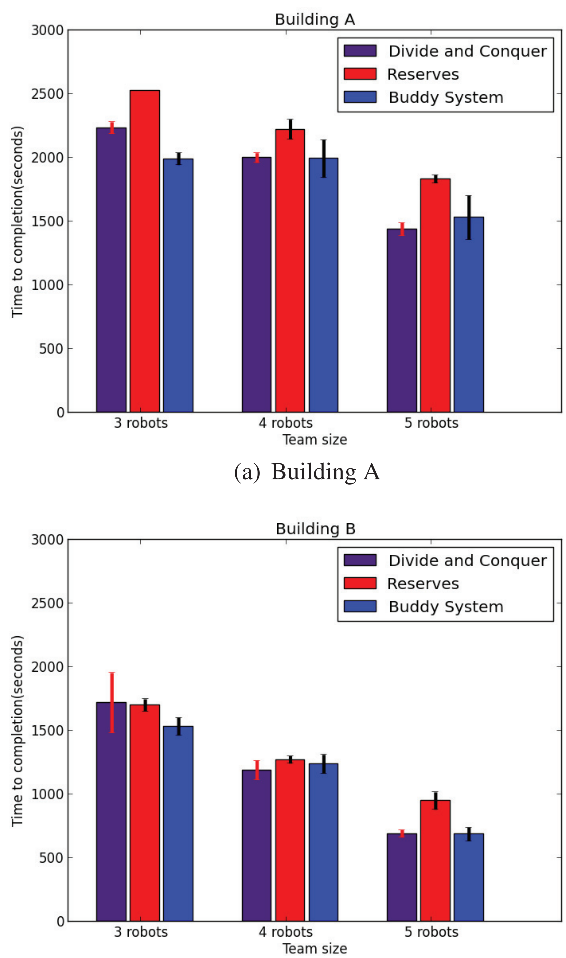
Strat.	Three robots	Four robots	Five robots
D&C	2175, **2237**, 2287	1956, **1989**, 2048	1397, **1407**, 1397
Res	2526	2114, **2250**, 2299	1809, **1810**, 1877
BS	1930, **1992**, 2047	1810, **1989**, 2175	1328, **1510**, 1751

Unfortunately, there are some missing runs in Table 1. This is due to scheduling and time constraints for performing these experiments in this test facility. This test facility is distant from our home location and we only had a finite window of time in which to perform these experiments. As a result, some test runs were not completed.

We analyze each strategy in terms of the amount of time taken for the robots to finish the exploration task given team size and collaboration strategy. As the results indicate in Figure 16, the strategies *Divide and Conquer* and *Buddy System* always performed better than the *Reserves* strategy. The performance also always increases as the team size is increased in Buildings *A* and *B*. These results are consistent

Table 2. Time in seconds for each exploration strategy on Building *B*. Each strategy was run three times, median values are shown in bold.

Strat.	Three robots	Four robots	Five robots
D&C	1415, **1753**, 1987	1106, **1170**, 1290	660, **680**, 729
Res	1639, **1705**, 1755	1231, **1280**, 1302	876, **927**, 1040
BS	1459, **1509**, 1627	1147, **1236**, 1328	615, **696**, 746



(a) Building A

(b) Building B

Fig. 16. Comparison of the strategies and team sizes the buildings at the test facility.

with the office building results presented in the first series of experiments. Those experiments showed decreased performance improvement only once the team size was increased beyond six robots, which is larger than the teams we were able to test in this experiment series. The team performance is not improved in Building *C* due to the poor conditions in that building, including debris and concrete seams which inhibited progress.

5. Discussion

We have presented experiments which evaluate three collaboration strategies which can be used by teams of mobile robots to map and explore an unknown environment. We have also evaluated the impact of the number of robots on coverage in the exploration and mapping task.

The first collaboration strategy, called *Reserves*, keeps a pool of unallocated robots at the starting location. A new robot is activated when there are more exploration frontiers than currently active robots. This strategy was intended to minimize the amount of interference between robot agents since robots would be far away from each other during exploration. The results from our experiments do not indicate that this strategy results in less interference than other strategies since performance decreases more when more robots are added in some environments. The *Reserves* strategy is significantly slower at exploring the environment than other strategies.

The second collaboration strategy, called *Divide and Conquer* has all available robots proceed in one large group. Once there are two exploration frontiers, at a corridor T-junction for example, the team will divide in half and each sub-team will follow one of the exploration frontiers. This process will be repeated with teams dividing in half each time they see branching structure in the environment. It was anticipated that this strategy would result in higher interference, since robots would be maneuvering close together; however, the increased availability of robots near new exploration frontiers offsets this phenomenon.

The third collaboration strategy, called *Buddy System*, gives some of the advantages of the *Divide and Conquer* strategy with limited interference between robots. This strategy performed equally well as *Divide and Conquer* in the second series of experiments. We believe that it might outperform *Divide and Conquer* when the robot team size is significantly increased.

The experiments indicate that a *Divide and Conquer* type collaboration strategy is more effective at exploring an unknown environment than a *Reserves* strategy. Many existing multi-robot exploration and mapping schemes use a collaboration strategy which is similar to the *Reserves* strategy and might benefit from a *Divide and Conquer* based approach.

The results shown in Section 4.2 express a dependency between the performance of the chosen collaboration strategy and robot team size with the topology and geometry of the test environment. The strategy *Divide and Conquer* is the most successful, because the robots are able to split and explore the map. In the environments tested in these experiments, there is sufficient maneuvering room, so the potential for interference between platforms is minimized, at least with moderate team sizes. The team performance in Building B suggests that the exploration strategies operating in large, open environments with many rooms generally give the robots enough space to navigate without interference between one another. Therefore, the times for reaching the goals are very similar to the results indicated in 16(b). In addition, it is clear that the performance of the strategies improve in terms of the number of goals reached with the *Divide and Conquer* strategy; however, a scenario with smaller and tighter maneuvering conditions might benefit from the traditional *Reserves* strategy to minimize robot interference. Environments with intermediate maneuvering constraints would favor a *Buddy System* strategy.

The primary result of this work is that idle robots should be proactive instead of waiting for new goals to be uncovered by active robots. Sometimes, this will cause them to get in the way of other robots, but it often will allow them to be in the "right place at the right time" to explore newly uncovered areas. These exploration techniques operate at the frontier; they cannot currently evaluate the entire trajectory to determine how moving to the goal will affect the future exploration goals, nor can they predict what will happen when they arrive at the frontier. Will exploring this goal uncover a dead-end, or a T-junction corridor where two robots are needed? Having a team member with you who is able to immediately proceed down the other path can help to explore unknown environments more quickly.

6. Future work

We have conceived of additional collaboration strategies which were not evaluated in the experiments for this paper. One of these additional strategies is called *Dynamic Reserves*; in this strategy, the pool of reserved robots moves to the centroid of the active robots. We believe that this strategy will minimize the amount of time needed to allocate a new reserved robot to a new navigation goal, mitigating some of the disadvantages of the *Reserves* strategy. The *Dynamic Reserves* strategy is similar to future work proposed in Simmons et al. (2000).

In the experiments performed in this paper, communications were always assumed to be possible between each robot and a central map coordinator, as in Olson et al. (2012). In more challenging and large-scale scenarios, idle robots might be more useful as *communication relays* to form a mesh network and maintain communication across a large team of robots.

The collaboration strategies presented in this paper are special cases of a more general strategy where the desired robot group size is a variable which can be adjusted based upon environment topology and robot maneuverability. The strategies we have presented explore the limits of this general strategy where if we have m robots, the desired group size is 1 for the *Reserves* strategy, 2 for the *Buddy System*, and m for the *Divide and Conquer* strategy. We have presented qualitative arguments about why certain environments might benefit from a specific strategy. It would be interesting to explore this general strategy in more detail and determine a mechanism for predicting which desired robot group size will work best for a given environment.

Acknowledgements

The authors would also like to thank our reviewers for all of their helpful comments and suggestions. This paper is an extension of the paper "Coordination strategies for multi-robot exploration and mapping" that was presented at the *International Symposium of Experimental Robotics (ISER) 2012*, with new experiments and strategies.

Funding

This work was supported by the Army Research Lab (ARL) MAST CTA project (contract number 329420) and the Boeing Corporation (project RE236).

Note

1. *Stage* is a 2D multiple-robot simulator from the Player project. See: http://playerstage.sourceforge.net.

   The first two authors contributed equally to this work.

References

Bailey T and Durrant-Whyte H (2006) Simultaneous localisation and mapping (SLAM): part 2. *IEEE Robotics and Automation Magazine* 13(3): 108–117.

Benedettelli D, Garulli A and Giannitrapani A (2012) Cooperative SLAM using M-Space representation of linear features. *Robotics and Autonomous Systems* 60: 1267–1278.

Burgard W, Moors M, Stachniss C, et al. (2005) Coordinated multi-robot exploration. *IEEE Transactions on Robotics* 21(3): 376–386.

Dellaert F (2005) Square root SAM: Simultaneous localization and mapping via square root information smoothing. In: *Proceedings of robotics: science and systems*, Cambridge, MA, 8–11 June 2005, pp. 177–184.

Dellaert F and Kaess M (2006) Square root SAM: Simultaneous localization and mapping via square root information smoothing. *International Journal of Robotics Research* 25(12): 1181–1203.

Durrant-Whyte H and Bailey T (2006) Simultaneous localisation and mapping (SLAM): part I. *IEEE Robotics and Automation Magazine* 13(2): 99–110.

Fischler MA and Bolles RC (1981) Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM* 24(6): 381–395.

Folkesson J and Christensen H (2004) Graphical SLAM—A self-correcting map. In: *Proceedings of IEEE International conference on robotics and automation (ICRA 2004)*, New Orleans, LA, 26 April–1 May 2004, Vol. 1, pp. 383–390.

Fox D, Ko J, Konolige K, et al. (2006) Distributed multirobot exploration and mapping. In: *Proceedings of the IEEE* special issue on multi-robot systems 94(7): 1325–1339.

Gerkey B, Vaughan R and Howard A (2003) The player/stage project: Tools for multi-robot and distributed sensor systems. In: *11th International conference on advanced robotics (ICAR 2003)*, Coimbra, Portugal, 30 June–2 July 2003, pp. 317–323.

Grisetti G, Grzonka S, Stachniss C, et al. (2007) Efficient estimation of accurate maximum likelihood maps in 3D. In: *Proceedings of IIEEE/RSJ International conference on Intelligent* *robots and systems (IROS 2007)*, San Diego, CA, 29 October–2 November 2007, pp. 3472–3478.

Hollinger G, Singh S and Kehagias A (2010) Improving the efficiency of clearing with multi-agent teams. *International Journal of Robotics Research* 29(8): 1088–1105.

Joyeux S, Alami R, Lacroix S, et al. (2009) A plan manager for multi-robot systems. *International Journal of Robotics Research* 28(2): 220–240.

Olson E, Strom J, Morton R, et al. (2012) Progress towards multi-robot reconnaissance and the MAGIC 2010 competition. *Journal of Field Robotics* 29(5): 762–792.

Parker L (2008) Multiple mobile robot systems. In: *Handbook of Robotics*. New York; Berlin: Springer Verlag, pp. 921–941.

Quigley M, Gerkey B, Conley K, et al. (2009) ROS: An open-source robot operating system. In: *Proceedings of ICRA workshop on open source software*, Kobe, Japan, 12–17 May 2009.

Rogers JG III, Trevor A, Nieto C, et al. (2010) Effects of sensory perception on mobile robot localization and mapping. In: *Proceedings of International symposium on experimental robotics (ISER)*, Delhi, India, 18–21 December 2010, pp. 433–446.

Rogers JG III, Cunningham A, Paluri M, et al. (2011) Cooperative mapping of indoor environments. In: *Proceedings of SPIE conference on defense, security and sensing (DSS 2011)*, Orlando, FL, USA, 25–29 April 2011 .

Rogers JG III, Trevor AJB, Nieto-Granda C, et al. (2011) Simultaneous localization and mapping with learned object recognition and semantic data association. In: *Proceedings of IEEE International conference on intelligent robots and systems (IROS)*, San Francisco, CA, 25–30 September 2011, pp. 1264–1270.

Rogers JG III, Stump E, Young S, et al. (2012) Autonomous 3D exploration and mapping with unmanned ground robots. In: *Proceedings of SPIE conference on defense, security and sensing (DSS 2014)*, Baltimore, MD, USA, 23–27 April 2012.

Rusu RB and Cousins S (2011) 3D is here: Point Cloud Library (PCL). In: *IEEE International conference on robotics and automation (ICRA 2011)*, Shanghai, China, 9–13 May 2011, pp. 1–4.

Simmons R, Apfelbaum D, Burgard W, et al. (2000) Coordination for multi-robot exploration and mapping. In: *Proceedings of the National conference on artificial intelligence*, Austin, TX, 30 July–3 August, 2000.

Trevor AJB, Rogers JG III, Nieto-Granda C, et al. (2010) Tables, counters, and shelves: Semantic mapping of surfaces in 3D. In: *Proceedings of IROS workshop on semantic mapping and autonomous knowledge acquisition*, Taipei, Taiwan, 18–22 October 2010.

Trevor AJB, Rogers JG III, Nieto-Granda C, et al. (2012) Planar surface SLAM with 3D and 2D sensors. In: *Proceedings of IEEE International conference on robotics and automation (ICRA 2012)*, Saint Paul, MN, 14–18 May 2012, pp. 3041–3048.

Vincent R, Fox D, Ko J, et al. (2008) Distributed multirobot exploration, mapping, and task allocation. *Annals of Mathematics and Artificial Intelligence* 52(2): 229–255.

Yamauchi B (1998) Frontier-based exploration using multiple robots. In: *Proceedings of the second International conference on autonomous agents*, Minneapolis, MN, 10–13 May, pp.47–53.

Zheng K, Glas D, Kanda T, et al. (2011) How many social robots can one operator control? In: *Proceedings of the sixth International conference on human–robot interaction*, Lausanne, Switzerland, 6–9 March 2011, pp. 379–386.