2013 IEEE/RSJ International Conference on
Intelligent Robots and Systems (IROS)
November 3-7, 2013. Tokyo, Japan

Multi-Robot Caravanning

Jory Denny, Andrew Giese, Aditya Mahadevan, Arnaud Marfaing,
Rachel Glockenmeier, Colton Revia, Samuel Rodriguez, and Nancy M. Amato

***Abstract*— We study *multi-robot caravanning*, which is loosely defined as the problem of a heterogeneous team of robots visiting specific areas of an environment (waypoints) as a group. After formally defining this problem, we propose a novel solution that requires minimal communication and scales with the number of waypoints and robots. Our approach restricts explicit communication and coordination to occur only when robots reach waypoints, and relies on implicit coordination when moving between a given pair of waypoints. At the heart of our algorithm is the use of leader election to efficiently exploit the unique environmental knowledge available to each robot in order to plan paths for the group, which makes it general enough to work with robots that have heterogeneous representations of the environment.**

We implement our approach both in simulation and on a physical platform, and characterize the performance of the approach under various scenarios. We demonstrate that our approach can successfully be used to combine the planning capabilities of different agents.

I. Introduction

Multi-robot coordination, especially among heterogeneous robots, is becoming commonplace in robotics applications including swarming, flocking, task cooperation, and more. In scenarios such as collaborative surveillance [16], robot soccer [25], and search and rescue [21], heterogeneity presents an advantage because it allows robots with different capabilities to cooperate in manners that homogeneous robot groups cannot. However, communication, coordination [26], and robust task execution [7] among such groups present challenges such as determining what information needs to be combined and how to do so. In this paper, we explore these benefits and challenges in the context of *multi-robot caravanning*, the problem of directing a team of robots to cooperatively visit a sequence of areas of interest (*waypoints*) in an environment and in a manner that ensures that the robots stay together at all times.

The problem of multi-robot caravanning is inspired by the historical role of caravans – collections of travellers journeying together across potentially hostile territory – in human commerce and societal development. For humans, travelling in groups offers benefits such as the distribution of payload among individuals, the sharing of resources such as food and water, and more efficient management of work such as cooking or herding. In addition, it offers safety in numbers against adversarial threats, and allows individuals to better cope with harsh climates or rough terrain. We explore the benefits of caravanning in a robotics context, considering what a team of robots can gain by travelling together as a caravan to complete shared tasks.

Caravanning arises in scenarios where robots must move from task to task together, cooperating as a unit in order to complete each task. For instance, in a collaborative object transport task [13], a group of robots must cooperate to move a large object, such as a disaster victim [13], from one location to another. The combined effort of multiple identical robots is required to complete the task, i.e., a single robot is insufficient. In other scenarios, groups of heterogeneous robots cooperate by combining capabilities to complete tasks that could not be completed by multiple instances of just one type of robot. An example of this is demonstrated in a highway maintenance task [12], where a group of simple robots serves as safety markers, while a more sophisticated leader with global knowledge is responsible for guiding the other robots to their positions.

After we formally define the problem, we propose a novel approach to the multi-robot caravanning problem that efficiently exploits the individual knowledge of the robots to benefit the group. The cornerstone of our approach is the use of leader election in conjunction with leader following. The former exploits the differing environmental information of the robots to decide which robot should become the leader, and the latter specifies how robots should follow a leader in order to move from one waypoint to the next. Our solution requires limited communication and sensing ability, and works in scenarios where robots have different representations of the environment.

We make the following contributions:

- A formal definition of and a scalable solution to the multi-robot caravanning problem that requires minimal explicit communication and sensing ability.
- A novel application of leader election to exploit heterogeneity in representation, which is applicable to robots whose representations are incomplete and/or generated in a distributed manner.
- An implementation of the proposed approach both in simulation and on physical robots, and an empirical characterization of its performance.

This research supported in part by NSF awards CNS-0551685, CCF-0833199, CCF-0830753, IIS-0917266, IIS-0916053, EFRI-1240483, RI-1217991, by NIH NCI R25 CA090301-11, by THECB NHARP award 000512-0097-2009, by Chevron, IBM, Intel, Oracle/Sun and by Award KUS-C1-016-04, made by King Abdullah University of Science and Technology (KAUST).

J. Denny, A. Giese, A. Mahadevan, A. Marfaing, R. Glockenmeier, C. Revia, S. Rodriguez, and N. M. Amato are with the Department of Computer Science and Engineering, Texas A&M University, College Station, TX, 77843, USA {jdenny, awg4619, maditya, amarfaing, rglock, crevia, sor8687, amato} at cse.tamu.edu.

II. Related Work

A. Motion Planning

Motion planning is the problem of finding a valid path through an environment from a start location to a goal location. Many methods have been proposed to solve this problem [15], [17]. More recently, sampling-based motion planning has shown great success. Probabilistic roadmaps (PRMs) [15] are one class of sampling-based motion planners. In these methods, an approximation of the space is constructed using a graph (the roadmap) whose nodes are randomly sampled configurations of the planning space, and whose edges represent valid simple pathways between the various points. After the map is constructed, it can be queried for paths. Numerous variations on the basic technique address improving sample quality with heuristics (such as generating nodes close to obstacles [2] or in regions of high clearance [29]), quality [14], handling non-holonomic constraints [17], accounting for uncertainty [1] and so on.

A robot's *representation* of the environment is an *approximation* of its configuration space that incorporates the observations and information available to it and determines the actions it can take. We say two robots are *representation heterogeneous* if their representations constitute different approximations of the environment. An important consequence of random sampling in the construction of Probabilistic roadmap (PRM) representations is that two roadmaps representing the same environment are usually topologically different and return homotopically different paths. The use of different sampling methods, local planners, and construction strategies [19], [27] introduces even more variability. The heterogeneity that naturally arises in constructing PRMs makes them an excellent focus for considering representational heterogeneity in this work.

B. Coordination of Multiple Agents

Probabilistic roadmap methods have been extended to define a collection of robots as a single configuration [24]. This generalizes the planning space to include all robots. Although powerful if agents share global information, this technique is not robust to failure because any change in the available set of robots causes a change in the dimensionality of the planning space, necessitating the complete reconstruction of the roadmap. It also suffers from the curse of dimensionality — as the number of robots increases, the planning space becomes infeasibly large.

The problem of sensing-heterogeneous robots inspecting every point of a cluttered environment's boundary is considered in [28]. A graph is used to represent regions in the environment, where a node represents a visibility region of an obstacle and an edge represents the overlap between two visibility regions. The regions themselves are determined by the robots' sensing capabilities. In contrast to our scenario, this graph representation is shared and complete. Moreover, agents are not required to stay together; rather, the only requirement is that all points on the boundary must be visited.

One important class of approaches to multi-agent movement is flocking [22]. In most flocking models, there is an attractive force to the center of the group (*cohesion*) combined with a separation force (avoiding inter-agent collisions) and an alignment force to allow the group to show cohesive, coordinated movement. This method has been successfully applied to roadmaps in a centralized manner for simulation [4] and in a centralized robotic setup for heuristic approaches to pursuit-evasion techniques [23].

C. Leader Following

A leader-follower behavior [7], [8] is a case of coordinated movement in which an agent leads a group of agents, and one or more followers attempt to follow the leader agent. This technique has applications in formation control [9], multi-robot planning [3], and cooperative task execution [12]. Generally, a follower's goal is to stay within a given distance of the leader and adjust its relative angle such that the leader remains within the follower's field of view. We utilize this technique as a coordination mechanism in the caravanning problem.

The current state of the art in cooperative movement [7], [12] assumes that all robots have global knowledge or that robots with global knowledge are designated as leaders beforehand. For instance, in [12], the leader is assumed to have global knowledge and precision positioning, e.g., using GPS, while followers use only simple sensors and perform simple computation to follow the leader. In this approach, it is impossible to recover from a failure of the leader.

In contrast, our approach requires none of the robots to have global knowledge, and any robot could be elected the leader (provided it has sufficient environmental information to find a path between a given pair of waypoints). Thus, our approach advances the state of the art by generalizing to scenarios in which *all* robots have incomplete or overlapping information. Moreover, while prior approaches typically address the problem of *how* to follow the leader effectively, we focus on *who* should become the leader based on the robots' representations.

D. Leader Election

Leader election is the task of selecting a coordinator from a group of entities. Such a task is often seen in the distributed computing community, however it has many applications in distributed robotics as well. In our example, at each waypoint a leader should be selected to lead the group to the next waypoint. Although many algorithms have been proposed to perform election, we use a variation of the *Bully Algorithm* [10] for its simplicity, limited communication, and asynchronous nature.

Leader election is applied to flocking in [6]. The approach combines a distributed leader election algorithm with a flocking behavior in which followers move according to the leader's actions. The focus of the paper is achieving leader election with no explicit communication. In contrast, we assume communication is still allowed, and the purpose of leader election using path metrics is to serve as a cheap substitute to fully broadcasting paths or representations.

Moreover, prior approaches that employ leader election typically use randomization or relative positioning to elect a leader. For instance, in [6], the robot with the smallest angle to its two closest neighbors becomes the leader. In contrast, we elect a leader based on a path metric. In this case, the robot with the best path to the next waypoint is elected the leader. Paths can be ranked by shortest distance, smoothness, or highest clearance, etc. This is the mechanism by which robots exploit redundancy in information in a manner that is efficient and which allows them to use the best available information.

III. The Caravanning Problem and Algorithm

A. Problem Definition

In this section, we define the multi-robot caravanning problem, as well as related concepts such as the environmental representation and path.

Definition 1. *A path is a sequence of valid configurations connecting a given start and goal configuration.*

Definition 2. *A representation is a data structure, or collection of data structures, that an individual robot can query to obtain a path from a start position to a goal position such that if the robot follows the path it will be guaranteed to arrive at the goal within a finite expected time.*

The representation is assumed to include a source of observations that the robot can use to verify that it has arrived at its goal and has not collided with an obstacle.

Definition 3. *A representation is incomplete if there exists a start and goal in the environment for which it is unable to return a valid path for the robot to transition from the start to the goal, even though such a path exists.*

Definition 4. *Two robots are representation heterogeneous if there exists a start and goal pair for which their respective representations return different paths.*

Note that by this definition, two robots are also representation heterogeneous if one representation returns a path but the other fails.

Definition 5. *A waypoint is a coordinate in the robot's configuration space.*

A group of robots may share a set of waypoints that represent, for instance, task locations or locations that must be inspected. We assume each waypoint is reachable, i.e., there exists a path to it from every portion of the environment. However, the robot's environmental representation may be incomplete for some or all of the waypoints.

Definition 6. *A caravan is a group of robots that operate while meeting a visibility or cohesion constraint that applies to the group.*

The constraints may require, for example, that all robots in the group stay within a predefined distance of one another or to the group's centroid. In our implementation, each robot must be able to see at least one other robot, and the graph of visibility between robots must not be disjoint. Robots that have failed are not considered part of the group.

We are now ready to define the multi-robot caravanning problem:

Definition 7. *Given a group of n representation heterogeneous robots $r = \langle r_1, r_2, \ldots, r_n \rangle$, and a set of waypoints $W = \langle w_1, w_2, \ldots, w_m \rangle$, the multi-robot caravanning (MRC) problem is to generate a valid path for each r_i to visit all the waypoints in W such that the robots visit each waypoint as a caravan.*

Informally, the MRC problem is the problem of planning for a group of agents to visit a sequence of locations (waypoints) in the environment as a group.

B. Approach

We propose a novel solution to the MRC problem. Our solution divides the MRC problem into stages. At each stage, a leader is elected and a leader following approach is used to move robots from one waypoint to the next. The novelty of our approach lies in the application of leader election to decide which robot should become the leader. For every pair of waypoints in the sequence, the robot with the "best" path according to some metric (for instance, lowest path length or highest path clearance) becomes the leader. The other robots follow the leader until the next waypoint, where the process is repeated.

Prior approaches that perform leader following tend to differentiate between leaders and followers offline, based on heterogeneity in capabilities [7]. For instance, followers have just enough sensing and communication ability to localize themselves with respect to the leader so that they can follow it, while the leaders have more sophisticated global knowledge [12]. Moreover, prior approaches that perform leader election either elect a random robot as the leader, or rely on robot IDs (e.g., selecting the robot with the lowest or highest ID) or relative positions [6].

In contrast, we perform leader election both dynamically and in a problem-specific manner. Doing so has several benefits:

- The use of a path metric in performing leader election allows us to handle scenarios in which robots have different, even incomplete, representations of the environment. This scenario arises frequently in problems that involve generating or storing the representation in a distributed manner.
- Limited communication is required since robots communicate solely at waypoints and never communicate their representations or paths to one another; only the path metric is communicated.
- Our solution can exploit overlap between representations. If a robot with the best path has already failed or been lost, the one with the next best path will be elected.

C. Algorithm

Algorithm 1 Agent Algorithm Overview

Input: Waypoints $W = \langle w_1, w_2, \ldots, w_m \rangle$, Roadmap R

```
for all w_k ∈ W do
    p = R.FindPath(w_k, w_{k+1})
    result = ElectLeader(p)
    if result == "leader" then
        SwitchLeader()
        Traverse p while localizing
        Call for leader election
    else
        repeat
            FollowLeader()
        until Leader election call
    end if
end for
```

The overall algorithm is shown in Algorithm 1 (failure conditions omitted). Each stage can be explained in terms of three steps: leader election, leader switching, and leader following.

In the Leader Election step, one robot that knows a path between the current and next waypoint is chosen as the leader and will be responsible for traversing its path. In the Leader Switching step, the newly elected leader travels to a designated position near the current waypoint, from which it will begin to traverse its path. In the Leader Following step, all other robots follow the leader by maintaining a constant position and orientation relative to it. We now explain these steps in detail.

D. Leader Election

Algorithm 2 ElectLeader

```
Broadcast ID and path metric
Receive M as a map of IDs to path metrics
bestID = arg max_{id∈M} M[id]
if bestID ≡ myID then
    Broadcast end of leader election
    return leader
else
    return follower
end if
```

In the first step, one robot is selected (the leader) that will be assigned the responsibility for executing a plan from waypoint w_k to waypoint w_{k+1}. This is achieved using a slightly modified version (Algorithm 2) of the Bully algorithm [10] for leader election. First, each robot queries its environmental representation for a path between waypoints w_k and w_{k+1}. It then broadcasts a *path metric* based on the result of its query to the other robots, together with its ID. If no robot finds a valid path (i.e., all robots broadcast an invalid metric), the algorithm terminates and returns failure. If exactly one robot finds a path, it is elected the leader by default. If two or more robots find a path, the one with the better path metric is elected leader. In case of a tie, the robot with the lower ID is chosen. The path metric is any scalar value that summarizes the quality of the candidate path. In this work, we choose to use path length as the metric; the shortest path is the most desirable. Other possible metrics include path clearance, path smoothness, etc.

E. Leader Switching

If an agent decides that it has been selected to be the leader, it will need to move to a designated leader position. In our implementation, it creates a Rapidly-exploring Random Tree (RRT) [18] from its current position to a position along the path between the current waypoint w_k and the next. Next, the robot traverses the path provided by the RRT and turns to face the waypoint. As explained in Section III-F, this step is necessary to update the formation that the robots assume in the leader following step.

The leader switching process is outlined in Algorithm 3. All other robots remain stationary until the leader has successfully reached its designated position or notified them of failure. The leader initially creates the RRT plan without taking into account any of the other robots' positions. However, as each robot is seen for the first time, the leader updates its list of obstacles to include the new agent. The leader re-evaluates its RRT before it continues to move along it; if any point is in collision because of a change to the list of obstacles, a new RRT is created from the leader's current position. At the end of the leader switch, the robots that were added to the obstacle list are removed. If the leader is unable to find a path, it notifies the other robots that it has failed and the algorithm terminates with failure.

Algorithm 3 SwitchLeader

Input: Waypoint w

```
repeat
    g = CreateGoal(w)
    s = GetCurrentPosition()
    P = GetRRTPath(s, g)
    for all p ∈ P do
        V = GetNewlyVisibleRobots()
        for all v ∈ V do
            O = O ∪ AddTempObstacle(v)
        end for
        if IsInCollision(p, O) then
            break
        else
            MoveToPoint(p)
        end if
    end for
until AtGoal(g)
RemoveTempObstacles(O)
```

F. Leader Following

In the leader following step, the leader executes the plan by following its queried path. All other robots move relative to the leader while attempting to maintain visibility to it or one another. At the start of this step, we maintain the invariant that the *visibility graph* of active robots (i.e., all robots that have not failed) is connected and at least one robot is at or near the current waypoint. The nodes of the visibility graph are robots, and there is an edge between every pair of robots that can observe one another.

A number of different flocking or formation techniques could be employed at this stage. We employ a simple leader-following approach in which robots form a chain that is headed by the leader. Each robot tracks the one in front of it (its target) and attempts to visit each position its target does. This approach has several advantages over other flocking techniques. Firstly, each robot attempts to follow a path along the leader's roadmap, which is known to be valid at least for the leader. This also means robots need not employ any kind of obstacle sensor. Moreover, this technique is scalable to a large number of robots since each robot's movements depend only on its observations of the robot in front of it (and are therefore independent of the number of robots in the chain). This step ends when the next waypoint is reached.

IV. Implementation

A. Robot Platform

The robot platform we use is an Asus Eee PC netbook equipped with an on-board webcam and wireless networking capability, mounted on an iRobot Create (Figure 1) that we control through the Player robot interface [11].

The camera has a maximum resolution of 640x480 pixels. The Creates are two-wheel differential drive unicycle robots with a maximum speed of about $0.5m/s$ and a minimum speed of about $0.1m/s$, below which their motion is highly unreliable. Internal robot odometry information is highly inaccurate, especially when rotating, and at particular speeds. To compensate for this, we require accurate observations of environmental features.

Fig. 1. iRobot Create with mounted Eee PC netbook for webcam use. Markers placed around the robot are used by neighboring robots to determine relative position and orientation

B. Localization and Robot Detection

We rely on frequent localization using visual markers. For robust marker creation and detection, we utilize the ArUco marker detection library from the University of Córdoba [20]. Markers are placed along the walls in the environment at roughly regular intervals. Each marker has a known unique ID, and known absolute position and orientation in the environment. Robots localize themselves by calculating their relative pose to the markers and transforming it into global coordinates based on the markers' known positions and orientations. Markers are also used by robots to detect other robots' poses. For this purpose, each Create's perimeter is covered with markers whose relative positions to its centroid are known.

The movement of each robot's camera relative to both wall markers and markers on other robots introduces camera blur, leading to intermittent failures in observations. To mitigate this, every movement of the robot is accompanied by a brief period in which the robot is stopped, but still making observations. Typically, each robot will stop for $0.1s$ after every $0.15m$ of movement. This temporarily minimizes camera blur, allowing the robot to make observations of both wall markers and other stopped robots.

C. State Estimation

An Extended Kalman Filter (EKF) is used to estimate the robot's state, accounting for uncertainty in both movement and sensor observations. Motion uncertainty is caused by uneven tiles on the floor of the environment, slippage of the wheels, and variations in the length of time controls are applied. Observation uncertainty results from variations in the intrinsic parameters of the netbook cameras, latency between the movement of the robot and the detection of the next image, and intermittent failures in observation due to motion blur of the camera.

The intermittent failures in observation described in IV-B lead to an increase in the covariance of the state estimate maintained by the EKF. To correct for this, when the covariance is sufficiently high, an information gathering phase is executed. The leader stops, rotates until it can detect a marker a wall marker, and updates its state estimate using the EKF until the error covariance is acceptably low before continuing towards its previous goal.

D. Communication

A number of steps in our proposed algorithm involve communication, such as the broadcast of path metrics during leader election. Our algorithm involves a distributed communication model, i.e., each robot can communicate with any other robot independently. However, at present the messages are *routed* through a central server to which any robot can connect or disconnect. For simplicity, we do not consider communication failures in this work.

E. Simulation

In addition to physical robots, we have implemented our proposed approach in simulation. The particular challenges we faced with the physical robots informed the design constraints of our simulation. Significant elements of our approach such as the leader election method and the flocking technique were chosen partly because they were compatible with the idiosyncrasies and limitations of the hardware and environment available to us. This made our simulation much better at predicting how changes to our algorithm would affect the behavior of the physical robots.

V. EXPERIMENTS

In this section, we describe the experiments we use to assess the feasibility of our approach. We run a variety experiments, some in simulation and some on physical robots, to study different aspects of our solution.

A. Experimental Setup

The environment used for testing is the fourth floor of the Bright (HRBB) Building of Texas A&M's campus in College Station, TX. A floorplan can be seen in Figure 2. The floor spans 40m of hallways 2m wide on average. In the simulation experiments, a 2D model of this environment is used. In all experiments, robots are required to visit a set of waypoints in a prescribed order (all robots start at or near a waypoint). In order to visit a waypoint, a robot must be able to query its roadmap for a path to the waypoint from the previous waypoint. The experiments test the ability of the robots to cooperate using the caravanning approach in order to visit every waypoint. A robot is successful if it can visit all the waypoints and does not collide with obstacles. The waypoints are known beforehand and common to all the robots, but do not influence the construction of each robot's roadmap.

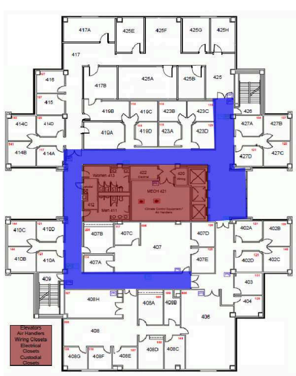

Fig. 2. HRBB Fourth Floor Floorplan with hallways highlighted blue

B. Success rate metric

The success rate is a measure of the percentage of waypoints a chain of robots manages to successfully visit. There are numerous possibilities for failure during the course of the experiment:

- A given Leader Following robot may lose track of its target.
- A robot may collide with other robots or obstacles in the environment during the Leader Following stage due to deviations caused by uncertainty. Both the leader and the followers are vulnerable to this.
- A newly elected leader may collide with other robots during the Leader Switching step.
- A newly elected leader may fail to find a path to the head of the chain.
- All robots may fail to find a valid path during the Leader Election.
- A robot may stall indefinitely because its target (the robot it is trying to follow) has failed. This may be in part due to the prior failure of a robot that knows a valid path.

It is apparent that these failures are not independent of one another. During the Leader Following step, the occurrence of one failure of a particular robot in the chain may have consequences for robots behind it. During the Leader Election step, the prior failure of a robot with a valid path may have consequences for the whole group. To account for these nuances, we define success rate as the *number of waypoints visited* normalized by the *maximum total number of waypoints that can be visited*. Under these terms, in a fully successful experiment (success rate of 1), the number of waypoints visited is $n \times m$, where n is the number of robots and m is the number of waypoints.

C. Region Decomposition Scenario

In this experiment, which we perform in simulation, we demonstrate the motivation for considering representation heterogeneity in multi-agent coordination, as well as qualitatively characterizing the behavior of robots while caravanning. For R robots, we decompose the environment into M regions and run our caravanning algorithm. Each agent is assigned to one region, and the roadmap it constructs must fall entirely within that region.

The decompositions are made such that no robot's roadmap can successfully return a path between *every* pair of waypoints in the sequence, i.e., there is at least one pair of waypoints between which a given agent's roadmap will fail to return a path. We impose this restriction to force the robots to cooperate by caravanning. However, we allow some overlap between regions such that every consecutive pair of waypoints falls completely within at least one region, i.e., for every pair of waypoints, at least one roadmap could be constructed in a finite time that can be queried for a valid path between them.

We run a scenario with 12 robots, with the environment decomposed into 4 overlapping regions. Each region is assigned to 3 robots. There are 7 waypoints to visit. We are interested in the rate of success when the robots are caravanning, i.e., the average proportion of waypoints the robots successfully visit. We run this scenario for different roadmap sizes (ranging from 10 to 90 nodes) with different seeds, taking the average of the success rates.

The results are shown in Figure 3 (with error bars representing standard deviation). They show that agents can caravan effectively as long as each agent has a roadmap with sufficiently good quality as given by the number of nodes. The stabilization of the success rate indicates that other factors limit the success rate once the size of the roadmap is large enough.

D. Competing Roadmap Scenario

In this experiment, which we perform in simulation, we consider a scenario in which robots have different roadmaps of the same environment. We investigate how the properties

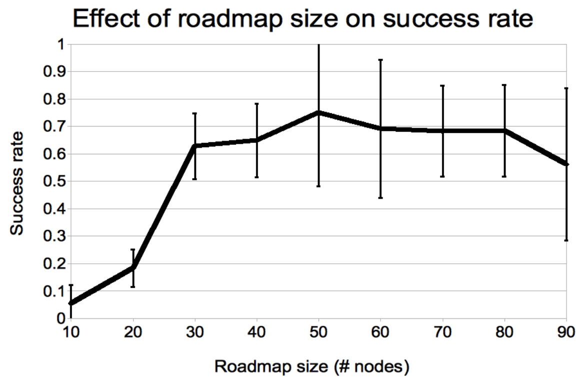

Fig. 3. Effect of roadmap size (number of nodes) on success rate. Error bars represent standard deviation.

of each agent's roadmap affect the success rate (the average percentage of waypoints the agents successfully visit).

We consider 5 scenarios. In the first 4, all agents generate roadmaps using the same sampling method (MAPRM [29], Uniform [15], OBPRM [2], and then Gaussian [5]). In the last scenario, the group of agents uses a mixture of roadmaps: each sampling method is used by 3 agents. In all cases, each robot is assigned a unique seed so that no two robots generate the same roadmap. For each scenario, we take the average of 10 runs. We compare the success rates of the scenarios in Figure 4.

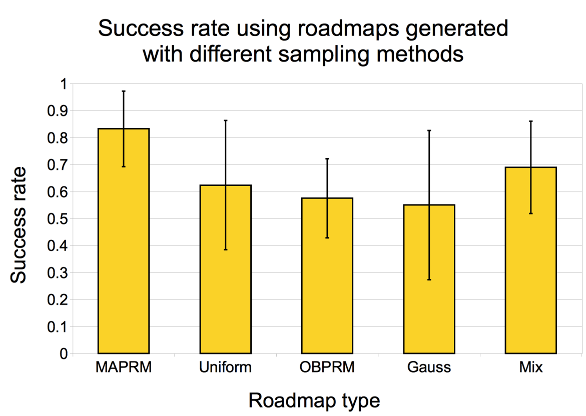

Fig. 4. Success rates for roadmaps generated using different sampling methods. Error bars represent standard deviation.

Our results demonstrate firstly that the success rate depends significantly on the type of roadmap used. Particularly, using MAPRM sampling leads to by far the highest success rate. MAPRM sampling yields roadmaps with higher clearance, which reduces both the chance of collision and sharpness of turns that might lead to missed observations of targets. Secondly, there is a significant benefit to using a variety of roadmaps. The "Mix" scenario has a higher success rate than any of the other scenarios except MAPRM. Some of this benefit probably arises from the ability of a robot using MAPRM to share its paths with other robots.

E. Physical Robots

We validate our simulation results with experiments on physical robots. The particular scenario we consider is to visit 5 waypoints along an L-shaped subsection of the HRBB environment using 3 robots. We conduct 10 runs with the physical robots, noting the number of waypoints reached by each robot, as well as its cause of failure (if any).

Our results are displayed in Table I. Of the 10 runs, 5 were complete successes, with the whole group reaching the final waypoint (runs 3, 6-9). In addition, in two of the runs (1, 10), at least one robot reached the final waypoint. The majority of failures were due to one of two causes. The first was a follower losing sight of its target while moving between waypoints. These were generally due to sharp changes in the path traversed by the target. The second cause of failure was collision among the robots during the leader switching stage, particularly in runs 4 and 5. One reason for this is that due to observation uncertainty, the robot that is performing the leader switch obtains wrong estimates of other robots' positions. Hence the resulting RRT produces a path that is too close to another robot, leading to collision.

F. Discussion

Our experiments suggest that utilizing data heterogeneity in cooperative systems such as caravanning yields solutions to a larger breadth of problems. By decomposing the environment into nearly-disjoint regions, we can allow for efficient distributed mapping and exploration of the environment. By generating different roadmaps of varying topology in the same region to capture distinct map properties, we can improve overall path quality and robustness.

Even robot platforms with only an internal representation of the environment, ability to localize, and simplistic method of detecting other agents are able to caravan successfully.

VI. Conclusion

In this paper, we present the *multi-robot caravanning problem* in which a group of agents cooperate to traverse an ordered sequence of points in an environment as a group. We propose a solution that requires minimal communication and works even with robots that have different and/or incomplete knowledge about their environment. Our approach relies on a novel use of leader election that allows us to handle failures in individual robots. We demonstrate our solution both in simulation and on a physical platform. The results show that the approach can compensate for incompleteness in representations and exploit redundancy.

In the future, we plan to study the scalability of our solution with a larger number of robots, as well as examine how the ordering of robots affects our ability to exploit redundancy in environmental representations.

References

[1] A. Agha-mohammadi, S. Chakravorty, and N. M. Amato. FIRM: Feedback controller-based information-state roadmap - a framework for motion planning under uncertainty. In *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pages 4284–4291, 2011.

TABLE I

RESULTS OF 10 PHYSICAL ROBOT TRIALS, WITH FAILURE EVENT (IF ANY) AND LAST WAYPOINT REACHED

Run #	Robot1	Robot2	Robot3
1	**Reached final waypoint** (5)	Lost sight of target (3)	Lost sight of target (2)
2	Collided with wall (3)	Failed because target (robot 1) failed (3)	Lost sight of target (2)
3	**Reached final waypoint** (5)	**Reached final waypoint** (5)	**Reached final waypoint** (5)
4	Collided with other robots while switching (3)	Collided with leader while switching (3)	Collided with leader while switching 3)
5	Collided with other robots while switching (4)	Collided with leader while switching (4)	Collided with leader while switching (4)
6	**Reached final waypoint** (5)	**Reached final waypoint** (5)	**Reached final waypoint** (5)
7	**Reached final waypoint** (5)	**Reached final waypoint** (5)	**Reached final waypoint** (5)
8	**Reached final waypoint** (5)	**Reached final waypoint** (5)	**Reached final waypoint** (5)
9	**Reached final waypoint** (5)	**Reached final waypoint** (5)	**Reached final waypoint** (5)
10	**Reached final waypoint** (5)	Lost sight of target (3)	Target failed (3)

[2] N. M. Amato, O. B. Bayazit, L. K. Dale, C. Jones, and D. Vallejo. Obprm: an obstacle-based prm for 3d workspaces. In *Proceedings of the third workshop on the algorithmic foundations of robotics on Robotics : the algorithmic perspective: the algorithmic perspective*, WAFR '98, pages 155–168, Natick, MA, USA, 1998. A. K. Peters, Ltd.

[3] N. Basilico, N. Gatti, and F. Amigoni. Leader-follower strategies for robotic patrolling in environments with arbitrary topologies. In *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems*, volume 1 of *AAMAS '09*, pages 57–64, Richland, SC, 2009. International Foundation for Autonomous Agents and Multiagent Systems.

[4] O. B. Bayazit, J.-M. Lien, and N. M. Amato. Roadmap-based flocking for complex environments. In *Proc. Pacific Graphics*, pages 104–113, Oct 2002.

[5] V. Boor, M. H. Overmars, and A. F. van der Stappen. The Gaussian sampling strategy for probabilistic roadmap planners. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, volume 2, pages 1018–1023, May 1999.

[6] D. Canepa and M. G. Potop-Butucaru. Stabilizing flocking via leader election in robot networks. In *Proceedings of the 9h international conference on Stabilization, safety, and security of distributed systems*, SSS'07, pages 52–66, Berlin, Heidelberg, 2007. Springer-Verlag.

[7] S. Carpin and L. Parker. Cooperative leader following in a distributed multi-robot system. In *Robotics and Automation, 2002. Proceedings. ICRA '02. IEEE International Conference on*, volume 3, pages 2994 –3001, 2002.

[8] S. Carpin and L. E. Parker. Cooperative motion coordination amidst dynamic obstacles. In *in Distributed Autonomous Robotic Systems*, pages 145–154, 2002.

[9] J. Desai, J. Ostrowski, and V. Kumar. Controlling formations of multiple mobile robots. In *Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on*, volume 4, pages 2864–2869, may 1998.

[10] H. Garcia-Molina. Elections in a distributed computing system. *Computers, IEEE Transactions on*, C-31(1):48–59, jan. 1982.

[11] B. P. Gerkey, R. T. Vaughan, and A. Howard. The player/stage project: Tools for multi-robot and distributed sensor systems. In *In Proceedings of the 11th International Conference on Advanced Robotics*, pages 317–323, 2003.

[12] J. Huang, S. Farritor, A. Qadi, and S. Goddard. Localization and follow-the-leader control of a heterogeneous group of mobile robots. *Mechatronics, IEEE/ASME Transactions on*, 11(2):205–215, april 2006.

[13] J. Jennings, G. Whelan, and W. Evans. Cooperative search and rescue with a team of mobile robots. In *Advanced Robotics, 1997. ICAR '97. Proceedings., 8th International Conference on*, pages 193–200, 1997.

[14] S. Karaman and E. Frazzoli. Sampling-based algorithms for optimal motion planning. *CoRR*, abs/1105.1186, 2011.

[15] L. E. Kavraki, P. Švestka, J. C. Latombe, and M. H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Trans. Robot. Automat.*, 12(4):566–580, August 1996.

[16] A. Kolling and S. Carpin. Multi-robot surveillance: An improved algorithm for the graph-clear problem. In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pages 2360–2365, may 2008.

[17] S. M. Lavalle. Rapidly-exploring random trees: A new tool for path planning. Technical report, 1998.

[18] S. M. LaValle and J. J. Kuffner. Randomized kinodynamic planning. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 473–479, 1999.

[19] T. McMahon, S. Jacobs, B. Boyd, L. Tapia, and N. Amato. Evaluation of the k-closest neighbor selection strategy for prm construction. Technical Report TR12-002, Texas A&M, College Station Tx., 2011.

[20] R. Munoz-Salinas. Aruco, 2012. http://sourceforge.net/projects/aruco/.

[21] R. R. Murphy, J. Casper, M. Micire, and J. Hyams. Mixed-initiative control of multiple heterogeneous robots for urban search and rescue, 2000.

[22] C. W. Reynolds. Flocks, herds, and schools: A distributed behavioral model. *Comput. Graph.*, 21(4):25–34, 1987. Proc. SIGGRAPH '87.

[23] S. Rodriguez, J. Denny, , J. Burgos, A. Mahadevan, K. Manavi, L. Murray, A. Kodochygov, T. Zourntos, and N. M. Amato. Toward realistic pursuit-evasion using a roadmap-based approach. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2011.

[24] G. Sanchez and J.-C. Latombe. Using a prm planner to compare centralized and decoupled planning for multi-robot systems. In *Robotics and Automation, 2002. Proceedings. ICRA '02. IEEE International Conference on*, volume 2, pages 2112–2119, 2002.

[25] M. T. J. Spaan and F. C. A. Groen. Team coordination among robotic soccer players. In *In Proceedings of RoboCup International Symposium*, 2002.

[26] A. Stranieri, E. Ferrante, A. E. Turgut, V. Trianni, C. Pinciroli, M. Birattari, and M. Dorigo. Self-organized flocking with an heterogeneous mobile robot swarm. In T. Lenaerts, M. Giacobini, H. Bersini, P. Bourgine, M. Dorigo, and R. Doursat, editors, *Advances in Artificial Life. Proceedings of the 11th European Conference on Artificial Life (ECAL 2011)*, pages 789–796. MIT Press, Cambridge, MA, 2011.

[27] L. Tapia, S. Thomas, B. Boyd, and N. M. Amato. An unsupervised adaptive strategy for constructing probabilistic roadmaps. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 4037–4044, May 2009.

[28] K. Williams and J. Burdick. Multi-robot boundary coverage with plan revision. In *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pages 1716 –1723, may 2006.

[29] S. A. Wilmarth, N. M. Amato, and P. F. Stiller. MAPRM: A probabilistic roadmap planner with sampling on the medial axis of the free space. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, volume 2, pages 1024–1031, 1999.