Coalition game-based distributed coverage of unknown environments by robot swarms

(Short Paper)

Ke Cheng, Prithviraj Dasgupta
Computer Science Department
University of Nebraska, Omaha, NE 68182, USA.
E-mail: {kcheng,pdasgupta}@mail.unomaha.edu

ABSTRACT

We consider the problem of distributed exploration or coverage of an unknown environment by a swarm of mobile mini-robots that have limited memory, computation and communication capabilities. We describe a novel mechanism of distributed coverage of an unknown environment by swarmed robots that can dynamically merge and split into structured teams or exchange team members to improve the efficiency of solving the coverage problem. Our mechanism combines the technique of swarm-based flocking with coalition games to enable robots dynamically select utility maximizing teams that move in formation.

Categories and Subject Descriptors

I.2.9 [**Artificial Intelligence**]: Robotics—*Autonomous Vehicles*; I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Multi-agent Systems*

Keywords: Swarming, flocking, coalition game, distributed coverage.

1. INTRODUCTION

Over the past few years, techniques from emergent computing such as swarming have been employed increasingly as an attractive mechanism to design large teams of mobile mini-robots. In swarmed systems, complex behaviors are manifest at the global level by encoding simple behavior patterns on the individual swarm units. Swarmed robot systems are particularly attractive for building large robot teams because they are inherently distributed and robust against failures of a significant number of the swarm units(robots), and, relatively inexpensive to build because of the simple behavior requirements on each robot. However, the simplicity and low cost of each mini-robot makes each robot constrained in the amount of memory, resources (e.g., sensors), computation and communication capabilities it has available onboard, and, limits the complexity and efficiency of problems that can be solved using swarmed multi-robot systems. Our work in this paper is based on the insight that distributed coverage of an unknown environment using swarmed robots can be improved if the robots are capable of forming small structured teams and dynamically adapting the structural formation and membership in each team based on the performance of the robots and the operational constraints in the environment. Previous research in robot team formation has been largely inspired by Reynolds' model of distributed flocking behavior observed in herds or flocks of animals[6]. In this model, each robot in a robot team adapts its motion and position based on the current position and heading of other team members such as a team leader or an immediate neighbor(s). However, the flocking model does not provide mechanisms to dynamically adapt the membership of robot teams based on team performance. To address the issue of dynamic team formation, we use a mechanism based on coalition games in multi-agent systems. Specifically, we make two contributions in this paper to address swarmed multi-robot coverage of unknown environments: (i) We describe a robot team formation mechanism based on flocking behavior that allows multiple robot teams to adapt their formation in the presence of obstacles and narrow passages. (ii) We describe a coalition game based mechanism that enables robots having inefficient behavior in a team to dynamically select and join teams to improve the system's performance.

Cite as: Coalition game-based distributed coverage of unknown environments by robot swarms (Short Paper), K. Cheng and P. Dasgupta, *Proc. of 7th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2008)*, Padgham, Parkes, Müller and Parsons (eds.), May, 12-16., 2008, Estoril, Portugal, pp.1191-1194.

2. RELATED WORK

Most research on formation of robot-teams using distributed techniques has been inspired by Reynolds' seminal work on the mobility of flocks[6]. Following this model, [11] describe mechanisms for robot-team motion while maintaining specific formations where individual robots determine their motion strategies from the movement of a team leader or neighbor(s). In contrast to these approaches, [4] describes techniques for robot team formation without using global knowledge such as robot locations, or the positions/headings of other robots, while using little communication between robots. However, in most of these approaches, achieving performance efficiency is not a principal objective, and consequently, none of these flocking-based approaches enable robots in a team to dynamically select and change teams to improve the system performance. Complementary to the flocking-based model, [10] uses a contract-net protocol to realize multi-robot coalitions, [9] uses robot team formation inspired by molecular motion in physics, [5] uses coordination in agent teams inspired by *SharedPlans*. Multi-agent coalition formation has been an active research topic within multi-agent systems[7, 8]. However, almost all these these techniques are computationally expensive and require time of the order of days to months to execute on relatively small

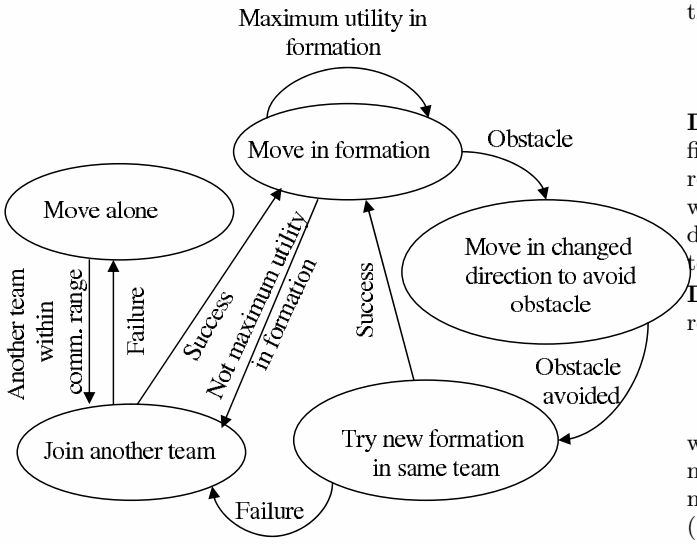

Figure 1: State transition diagram for a robot moving in formation.

number (e.g., 10-50) of agents. Therefore, they are not readily amenable for use within the memory and computational constraints on mobile mini-robots, and, the near-real time motion requirements for each robot in our system.

3. TEAM-BASED, SWARMED MULTI-ROBOT COVERAGE

The motivation behind our team-based coverage algorithm is to improve the coverage of the environment in terms of time and redundancy by using robot teams organized in a coverage- maximizing formation instead of using single robots. First, we define some important parameters that are used to implement our team-based coverage algorithm.

Definition 2.1 *Team:* A robot team in our system comprises a group of robots with a contiguous sensor swathe between themselves. Let $t \subset R$ be a subset of robots. t is considered a team if every robot $r \in t$ has the same heading and every robot maintains a *distance balance* criterion of maintaining a separation equal to the diameter of a robot's sensor cone (d_s), measured along the direction orthogonal to the team's motion, between itself and at least another robot in the same team t. The distance balance criterion of a team ensures that the combined region covered by the sensors of robots in the same team remains connected, while the same heading of each team member ensures that this region does not get disconnected over time due to the team's movement.

Definition 2.2 *Robot Redundancy(d_r):* A robot r's coverage is redundant in a team if the region covered by its sensor has already been covered either partially or completely by another robot belonging to the same team. d_r denotes the fraction of the area covered by r's sensor that has already been covered by another robot's sensor in the same team. In this paper, we consider $d_r \in \{0, 1\}$ to denote non-redundant($d_r = 0$) or completely redundant($d_r = 1$) coverage.

Definition 2.3 *Team Redundancy(D_t):* The coverage redundancy of a team t is given by the ratio between the number of robots in team t with redundant coverage and the total number of robots in the team.

$$D_t = \frac{\mid r \mid}{\mid t \mid} \mid r \in t \wedge d_r = 1$$

Definition 2.4 *Team Utility(U_t)*: Team t's utility is defined as $U_t = 1 - D_t$. When no robot in the team does redundant coverage($D_t = 0$) $U_t = 1$. On the other hand, when all robots in the team, except one, are performing redundant coverage by following immediately behind another team member($D_t = \frac{1}{|t|}$) $U_t = 1 - \frac{1}{|t|}$.

Definition 2.5 *Robot Utility($u_{r,t}$)*: The utility derived by robot r from participating in team t is given by:

$$\mathbf{u_{r,t}} = \begin{cases} (1 - d_r) + \frac{U_t}{T_{max} - |t| + 1}, & \text{if } \mid t \mid \leq T_{max}, \\ 0 & \text{if } \mid t \mid > T_{max} \end{cases}$$

where T_{max} is the maximum allowable size for a team. The maximum possible value for a robot's utility is $u_{r,t} = 2$ when none of the robots in the team perform redundant coverage ($d_r = 0$ and $U_t = 1$) and the team has the maximum possible size ($\mid t \mid = T_{max}$). When a robot's team size is less than the maximum, it receives a utility $u_{r,t} \in [1, 2)$. We call this *tolerable* utility for the robot. However, when a robot performs redundant coverage in a team, its utility $u_{r,t} \in [0, 1)$. We call this *inadmissible* utility for the robot. The robot utility function described above achieves two objectives towards improving the system's performance. First, it discourages redundant coverage by a robot by assigning lower, inadmissible utilities to redundant coverage. Secondly, the utility function ensures that robots derive a higher utility from participating in larger teams by assigning lower utilities to robots forming smaller teams, or worse, covering the region individually. Therefore, it prevents the team-based coverage from degrading into individual coverage.

A high-level description of the controller for a robot in our team-based coverage system is shown in Figure 1. We assume that robots are initially deployed into the environment in teams of size T_{max}. In Figure 1, each robot that is part of a team moving in formation checks its utility at certain intervals to see if the utility it is obtaining from participating in the team can be improved. If its current utility is the maximum possible, it continues with its current formation in the current team. If a robot moving in formation within a team encounters an obstacle, it alters its direction to avoid the obstacle. However, while avoiding the obstacle, other team members who did not encounter the obstacle can continue moving in formation. Therefore, after a robot avoids an obstacle it tries to reenter at its position in the team formation before encountering the obstacle. If it is able to retain its earlier position in the team, the team continues in formation. Otherwise, the robot tries a new formation in the same team. However, this new formation might be redundant for the robot (e.g., it is directly behind another robot in the same team and covers the same area as its preceding robot). If the formation is redundant, its utility is not the maximum possible utility for the set of robots in the team. The robot that is redundant in the formation then tries to join another team, if it has any other teams within its communication range. If the robot succeeds in joining another team, it becomes a member of that team. Otherwise, it continues to move individually using the Manhattan distance heuristic described in [2], because the utility of a robot moving individually ($= 1 + \frac{1}{T_{max}}$) is higher than

its utility from redundant coverage ($= 0 + \frac{1-\frac{1}{|t|}}{T_{max}-|t|+1}$) in a team. When the individually moving robot comes within communication range of another team that it can join, it attempts to join that team and move in formation if that yields a higher utility than the utility from moving individually. In the following sections, we describe the different functionalities of the robot to implement this controller.

3.1 Team Formation

Robots in a team must have a distributed mechanism to maintain a team formation while moving as well as adapt the formation in response to environment changes such as obstacles, walls or passages encountered in the team's path. Our formation maintenence mechanism is based on Reynolds' flocking model and uses neighbor-referenced separation[1] to adjust robot positions to retain formation while moving. In our system, each robot in a team has to maintain the *distance balance* criterion with its immediate neighbors in the team at each time step. To achieve this, each robot r in a team first observes the locations of its two adjacent (nearest two) neighbours from the locations stored within its communication map c_r. If the distance balance criterion is satisfied, the robot continues with its current speed and heading. On the other hand, if the distance balance criterion is not satisfied between the robot and its neighbors, the robot must adjust its speed to maintain team formation. To realize this, first the robot calculates its desired location for the next time step that would enable it reattain the distance balance criterion w.r.t its neighbors. It then adjusts its speed and heading so that it either reaches this desired location, or reaches as close as possible to it, in the next time step.

Proposition: (*Distance Balance Criterion.)* The optimal formation in a team that ensures maximum coverage by its team members is obtained when adjacent pairs of robots in the team are separated by a distance that equals the diameter of their sensor cones, measured along the orthogonal to the team's heading.(Proof omitted for space)

3.2 Team Dispersion

To avoid redundant coverage between the regions covered by different teams, teams moving in formation should disperse away from each other. To achieve this, we can use a technique similar to the Manhattan distance heuristic used for individual swarm-based coverage (described in [2]) to make robot teams disperse away from each other. However, if every individual robot in a team tries to implement the Manhattan distance heuristic to move away from robots in other teams while simultaneously staying in formation with its own team members, it would require a very complex, perhaps infeasible procedure, that would be difficult to realize within the robot's memory and computational constraints. To implement team dispersion in our system we identify a special robot in each team called the team's *core* robot. The core robot is the only member in the team that does not use the neighbor-referenced separation for team formation, but instead, uses the Manhattan distance heuristic to disperse away from core robots in other teams. This mechanism manifests itself in every member of a team, except the core robot, following the core-robot using neighbor referenced separation, while the teams themselves disperse away from each other to avoid redundant coverage.

Core Robot Selection. To select the core robot, every time the membership of a team changes (i.e.,every time a robot joins or leaves a team), every robot in the team uses a breadth-first-search on the locations stored within its communication map, starting from its current location upto a distance of $T_{max} \times d_s$, to determine the other members comprising the team. Each robot then determines the centroid of the points consisting of the locations of its team members including itself, and, selects the robot that is closest to this centroid as the team's core robot.

Checking for Redundant Coverage Within Teams. As mentioned in [2], each robot r maintains a local coverage map of radius μ_r centered at robot r's current location. The coverage map contains the locations within μ_r that were covered by the robot itself or neighboring robots over the last t_h time steps. For our team-based coverage, we set the radius $\mu_r = T_{max} \times d_s$, to ensure that the coverage map of a robot encompasses the furthest team member in its team when the team size is maximum. While moving in formation in a team, if the desired location calculated by a robot to maintain formation corresponds to a location that is already denoted as covered within its coverage map, the robot infers that it is performing redundant coverage within the team.

3.3 Dynamic Team Formation

When a robot derives inadmissible utility in a particular formation, it first checks to see if it can improve its utility to a *tolerable* value (between 1.0 and 2.0) by making a new formation within the same team. However, if the robot is unable to find a utility improving formation (e.g., the obstacle or passage that caused it to change formation is still there) in the same team it attempts to join another team.

Let r_c denote a robot that is currently participating in a team t_c and deriving inadmissible utility. As shown in Figure 1, r_c then attempts to change its membership by joining another team. To enable r_c select a suitable team, it should be able to calculate its expected utility from joining a team t_j that it is aware of. However, in our system, each robot does not keep track of the location and membership of other teams in the environment due to its memory and communication constraints. Therefore, when r_c attempts to join another team, it has to dynamically compute the location, heading and configuration in each team within its communication range, so that it can calculate an expected utility of joining another team and select the best possible, expected utility maximizing team. The algorithm used by a robot r_c to select another team maximizes its expected utility. Here, robot r_c uses information about the location and heading of other robots from its communication map. r_c first determines the teams within its communication range as connected components in a graph with edges less than d_s (sensor diameter) in length, using Kruskal's algorithm[3]. It then calculates the redundancy in the team, the location of the core robot of the team and the heading of the team from the locations and heading of the team members. Finally, it selects the team that maximizes the weighted product of the team's utility, its distance from the team's core and its alignment with the heading of the team. A challenging issue encountered by a robot wishing to change teams is that it might miscalculate the number of team members in the teams it expects to join because of some of its destination teams' members being outside its communication range. In such a scenario, we use a coalition game based framework, where the robot wishing to change its team considers the

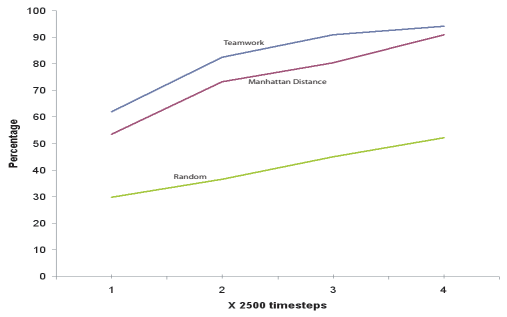

Figure 2: Percentage of area covered with 5 agents in a square environment for different coverage strategies.

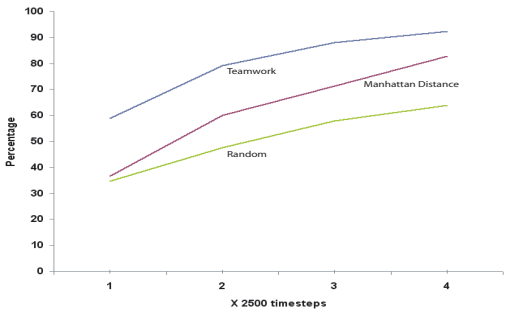

Figure 3: Percentage of area covered with 10 agents in a passage environment for different coverage strategies.

different teams it is aware of as possible coalitions it can join. It then builds a coalition tree and dynamically prunes the nodes of the coalition tree based on the utility of the coalition(node) as the different teams move closer or further away from it. Finally, the coalition tree is pruned to a single node corresponding to the team with the highest perceived utility for the team-changing robot to join.

4. EXPERIMENTAL RESULTS

For our experiments we have used $5-10$ simulated robots within the Webots robot simulation platform. Each robot is simulated as the *DifferentialWheels* robot, whose speed and direction are controlled by changing the relative rotation speed between the two wheels. The standard speed of each wheel was set to 0.1 meter/time unit. Robots measure 0.1 meter $\times$0.1 meter. We consider two scenarios for the environment in our experiments. One is in the shape of a square measuring 10×10 meter^2; the other is in the shape of two squares which are of the same size as the former square and connnected by a 1×4 meter^2 passage. We have tested both scenarios with 5 and 10 robots that are able to dynamically form and split into smaller teams.

In our first set of experiments shown in Figure 2, we observe the coverage of a square shaped environment with different coverage strategies. In the random strategy, each robot moves in a random direction when it encounters a wall or another robot. In the Manhattan distance based strategy, a robot selects the next timestep's direction as the direction that maximizes the Manhattan distance between itself and others robots within its communication range. In the teamwork strategy, each robot moves in a team-like formation using the method introduced in this paper. As shown in Figure 2, the strategy that uses the random method gets the least coverage. The strategy with team formation performs a $5-10\%$ better than the Manhattan distance-based coverage strategy. In our second set of experiments, we analyzed the effect of these three strategies in an environment consisting of two square rooms connected by a narrow passage. All the robots started from one of the rooms. Once again, the random method gets the worst coverage, as shown in Figure3. The team-based strategy shows the best coverage time because the teams are able to split and merge into sub-teams and a reasonable number of robots are able to cross the passage as a team to reach into the other room. On the other hand, for the random walk and Manhattan distance-based strategies, most robots remain limited within the room from which they start and even if a few robots are able to cross into the other room through the passage, most robots still remain in the room they started in and are unable to distribute themselves evenly across the two rooms.

5. CONCLUSIONS AND FUTURE WORK

In this paper, we have described a novel team-based, swarmed strategy for dynamic coverage of an unknown environment that combines a flocking model with coalition games. We are currently testing our algorithms on e-puck mini-robots to study their behavior in covering unknown environments. We believe that addressing these issues will solve some of the challenges in coordination between mini-robots with constrained memory and communication capabilities for different problems, including area coverage.

6. REFERENCES

[1] T. Balch and R. Arkin, "Behavior-based formation control of multi-robot teams," IEEE Transactions on Robotics and Automation, vol. 14, no. 6, 1998, pp. 926-939.

[2] K. Cheng and P. Dasgupta, "Dynamic Area Coverage with Faulty Multi-agent Swarms," Proc. IEEE/WIC/ACM Intl. Conf. on Intelligent Agent Technology (IAT), 2007, Fresno, CA, pp. 17-23.

[3] T. Cormen, C. Leiserson, R. Rivest and C. Stein, "Introduction to Algorithms," MIT Press, 2003.

[4] J. Fredslund and M. Mataric, "A general algorithm for robot formations using local sensing and minimal communication," IEEE Trans. on Robotics and Automation, vol. 18, no. 5, 2002, pp. 837-846.

[5] K. Sycara, M. Paolucci, M. Velsen and J. Giampapa, "The RETSINA MAS Infrastructure," Autonomous Agents and Multi-Agent Systems vol. 7, no. 1-2, 2003, pp. 29-48.

[6] C. Reynolds, "Flocks, herds and schools: A distributed behavioral model," Computer Graphics, vol. 21, no. 4, 1987, pp. 25-34.

[7] T. Rahwan, S. Ramchurn, V. Dang and N. Jennings, "Near optimal anytime coalition structure generation, "Proc. IJCAI 2007, pp. 2365-2371.

[8] O. Shehory and S. Kraus, "Methods for task allocation via agent coalition formation," Artificial Intelligence, vol. 101, no.1-2, 1998, pp. 165-200.

[9] D. Spears, W. Kerr and W. Spears, "Physics-based robot swarms for coverage problems," International Journal on Intelligent Control and Systems, vol. 11, no. 3, 2006, pp.124-140.

[10] F. Tang and L. Parker, "Distributed Multi-Robot Coalitions through ASyMTRe-D," Proc. IEEE Intl. Conf. on Intelligent Robots and Systems (IROS), 2005.

[11] P. Wang, "Navigation strategies for multiple autonomous mobile robots moving in formation," IEEE/RSJ Intl. Workshop on IROS, Tsukuba, Japan, 1989, pp. 486-493.