Technical Report

Department of Computer Science
and Engineering
University of Minnesota
4-192 EECS Building
200 Union Street SE
Minneapolis, MN 55455-0159 USA

TR 05-006

Self-deployment algorithms for mobile sensors networks

Monica Lapoint, Maria Gini, and Nikos Papanikolopoulos

March 31, 2005

Self-deployment algorithms for mobile sensors networks

Monica Anderson LaPoint, Maria Gini, and Nikolaos Papanikolopoulos
Department of Computer Science and Engineering, University of Minnesota

Abstract

Large and sophisticated networks of sensors are being developed and deployed in a variety of environments, from military surveillance, to environmental monitoring, and smart spaces. Adding mobility to sensors, by placing them on robots, can make the sensors more useful but at the cost of adding new challenges.

We focus on the problem of automated deployment of the sensors, in particular on how to disperse a group of robots in an unknown environment so as to cover the environment as much as possible while staying within communications range. We assume there is no central control, the robots operate independently, and the only communications among them are to ensure they are within communication range. Whenever a robot loses communication, it moves to attempt to regain communication.

We evaluate different algorithms based on the percentage of the environment that the group of robots succeeds in observing, the percentage of time the robots are within communications range, and how long it takes for them to reacquire communication.

1 Introduction

Mobile robots capabilities are improving dramatically while their costs are decreasing. Robot teams hold great promise in providing help for several difficult tasks associated with emergency response. For example, teams of robots can perform urban surveillance after a hurricane, execute effectively remote operations in the case of hazardous spills, participate in decontamination and decommissioning efforts immediately after a nuclear disaster, execute search and rescue operations, and find survivors in collapsed structures.

The primary motivation for this work comes from the Scout project [11]. The scouts are small, two wheeled robots with limited processing capability, that can deployed either by being hauled or launched into the environment by a larger robot. Because of their small size they can get into tight areas and be used in large numbers. Scouts are being considered for search and rescue applications, locating the source of a biological or chemical release, decontamination and decommissioning efforts, and monitoring highly sensitive areas.

Scouts are often teleoperated, but their ability to do autonomous operations has been demonstrated in a task of hiding and watching for motion [11]. There are different models of Scouts. The one shown in Figure 1, has more computing power (two 16MHz processors), better communication hardware (Bluetooth), more sensors (an accelerometer, a camera, and light and heat sensors) than the older models, while keeping the small form factor (approximately 150 mm in length) that makes it versatile.

The Scouts are smaller and more effective that most other existing robots in their same size range. The Scouts are comparable in size to the CotsBots [2], but have larger wheels and can move faster over rougher terrain. Communications among Scouts is done using Bluetooth instead of the RFM radio more common for small robots. Other smaller robots, such as the Khepera, are able to cover only limited distances and require perfectly clean surfaces.

One of the major issues when using very small robots is their extremely limited ability to estimate their own location and their limited communications range. In addition, their computing power and suite of sensors are limited. Because of these limitations, we believe that the robots should be controlled not as individuals but as a swarm [3]. In a swarm, each robot uses simple local rules to decide its own actions, without needing any command from a central controller or from any other robot. This makes the overall system highly scalable and robust to failures. On the other side, since robots in a swarm interact with

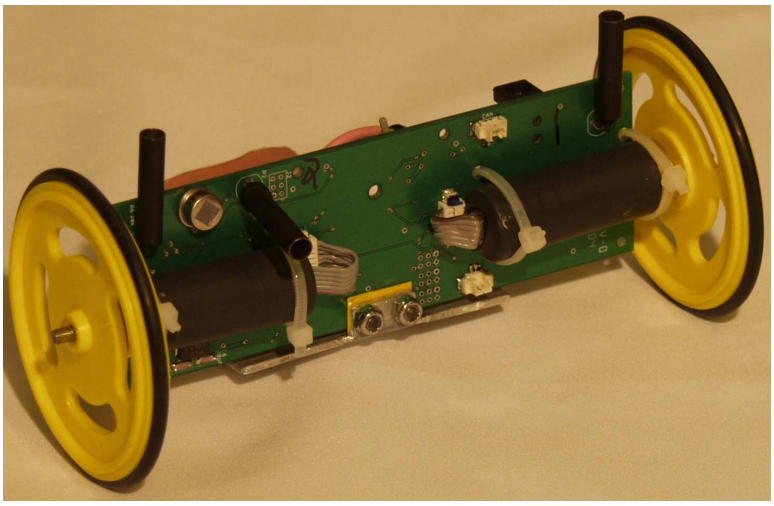

Figure 1: A Scout robot. At 150mm in length it is quite small, yet its large wheels allow it to move fast over rough surfaces.

each other in ways often unexpected, unexpected behaviors tend to emerge, which makes it hard to provide guarantees of performance.

In this study, we are interested in determining the tradeoffs between performance, hardware complexity, and control strategies. The task we chose is dispersion in an unknown environment while keeping connectivity with at least one other robot. The performance criteria we selected are the percentage of the environment explored by the end of the experiment, the percentage of the time the robots are within communications range, and the time needed to reacquire connectivity. We are specially interested in finding what motions methods will work best for exploration and for reestablishing communications between robots when it was lost. The robots use explicit communication via Bluetooth.

2 Related Work

Coverage of terrain during motion is important in many application domains, such as floor cleaning, lawn mowing, demining, harvesting, etc. In such applications usually one needs to cover the terrain only once. Wagner et al. [13] formalize the terrain covering problem and propose two algorithms, one called mark and cover, the second called probabilistic coverage, both for single and multiple robots. They show how several cooperating robots can obtain faster coverage. Algorithms inspired by insect behaviors, such as ants, are becoming popular both for terrain coverage [7], where robots leave trails and cover the terrain repeatedly, An algorithm for complete coverage of free space with a team of robots is presented in [10], where it is assumed that robots have odometry to track their own position as they construct a cell decomposition of the free space.

Exploration by multiple robots has been studied extensively. For instance, in [12] an algorithm is presented to build a global map, select goal locations for exploration, and use explicit communication to prevent multiple robots to go after the same goal. We are interested in a combination of exploration and deployment, where robots have to explore but also to maintain coverage of the areas explored and maintain communications with other robots. The applications we are considering are urban search and rescue, where it is important to form a network to allow for transmission of information and monitoring.

Various algorithms have been proposed for dispersion to ensure maximum coverage. For instance, in [1] two dispersion methods are proposed. The methods use beacons on the robots and visual communication, which require direct line of sight, and a simple environment with only convex obstacles. A simple diffusion algorithm has recently been demonstrated [2] with small robots that use their buzzers and microphones to disperse.

Howard et al. [5] address the problem of incremental deployment, where robots are deployed one-at-a-time into an unknown environment, and each robot uses information gathered by previously deployed robots to determine its deployment location. They assume every robot is equipped with an ideal localization sensor. We do not assume the robots know where they are, since for small robots localization is very hard. New technology, such as the Cricket [9], could provide position information, but it requires placing beacons in the environment, not having walls between the beacons, and has limits on the accuracy of the position information. These requirements rule out using the Cricket in the applications we are envisaging.

The problem of coverage has been extended to include the constraint that each node in the network must have a minimum number of neighbors. The dispersion method presented in [8] is based on artificial potential fields, and assumes that each node is equipped with a sensor that allows it to determine the range and bearing of its neighbors. Obstacles, such as walls, are not modeled. In our work, we disperse the robots in an environment which includes walls and obstacles. We require that each robot remains within communications range of at least one other robot. We are interested in working in very large environments where requiring a higher degree of connectivity might require to use too many robots. We will explore the effect of increasing the degree of connectivity in the future.

Hsiang et al. [6] propose methods for dispersing robots from fixed locations to cover the entire environment. They assume a continuous stream of robots would be entering the environment through predetermined locations. The goal of the robots is to position themselves such that the entire accessible area is covered. Through the use of deterministic robot motions and infinite supply of robots the information available is sufficient to guarantee that the robot will make the correct choice for its motion. We do not assume that there are enough robots to cover the entire space and to guarantee that every robot can remain within sensor range of the other robots. We require only that the robots stay within communications range, which is larger that sensor range and can go through walls.

3 Proposed Method

Our approach to the problem of robot dispersion and exploration in an unknown environment takes into account the physical limitations of the scout, the ultimate target of this research. Although the robots have two-way communication and an advanced sensor suite, the processing capability is limited.

We assume the robots only have local knowledge, i.e. they are not under global control (no central source knows the state of all of the robots), and do not have any knowledge of the environment other than what they can detect with their sensors. This assumption is useful for redundancy purposes since central control introduces a single point of failure.

In addition, we prefer that the robots do not need to know how many other robots are operating in the same environment, where those robots are located, and where those robots have been. This serves two purposes. First, it allows for some flexibility since more robots could be introduced after the initial deployment and failures are not catastrophic. Second, it does not put a heavy burden on sensors to either recognize other robots or calculate odometry accurately.

This paper presents results on a number of simulation experiments used to validate our initial hypotheses regarding the effectiveness of behaviors to disperse, cover, and maintain communications in an unknown environment. Simulations allow us to run a large number of tests with different behaviors and test the most promising ones on the actual robots.

Player/Stage [4] was used as the simulation environment. The pioneer robot was used since the physical form factor was less of an issue in a 2D simulation. The main sensors were a sonar array that provided distances to obstacles and some immediate environmental information. We use two of the worlds from Player/Stage, the hospital (see Figure 2) and the home (see Figure 3). Other researchers have used the same environments and this facilitates comparing results.

A new Stage device was created to simulate the communications using an onboard Bluetooth device. The discovery process, simulated using the WiFiInterface, gives each robot the names of other robots within approximately 8 meters. Degradation experienced due to intervening walls was implemented using a discounting scheme that deducted for any walls between robots. In addition, messages can be sent between robots if they are in range.

Three dispersement behaviors were the most effective for our purposes. The first behavior was a RANDOM WALK behavior. Given no obstacles, the robot moves on a slightly curved path by turning a random amount between 10° and -10° per time step. When the robot detects an obstacle, it stops, turns by a larger random amount (in the range 120° to 240°), and then resumes moving while turning a small random amount.

The second behavior was a FIND OPENINGS behavior that uses sensors to locate openings such as doorways or halls. Although, many sensors that measure range can be used, in the simulation, we used a sonar array that is similar to the sonar array found on the pioneer. The ranges in front of the robot are compared and the robot turns in the direction of the two adjacent ranges with the highest change in value.

The third behavior is a comparison behavior, AVOID ROBOTS. It simulates a robot that is able to recognize another robot that is nearby and move away from it. The simulation is implemented by putting a target on each robot that any robot can recognize. Physically, this could be implemented with varying level of effectiveness with vision or infrared sensors. Range sensors cannot be used since robots cannot distinguish between another robot and an obstacle.

In addition, two behaviors were implemented that react to a robot moving outside the range of all other robots. The RANDOM RANGE control algorithm moves the robot randomly until communications is reestablished. The BACKTRACK RANGE control algorithm attempts to move the robot back to previous positions. In rationale is that at a previous time step, the robot was in range so moving back to that position should reestablish communications, assuming the other robots did not move too far.

4 Simulations and Results

Six major sets of simulations were run that consisted of the three main behaviors with each of the range control algorithms. Within each set, the number of robots deployed varied between 10 and 50. The environments in Figures 2 and 3 varied in size and in complexity of rooms and openings.

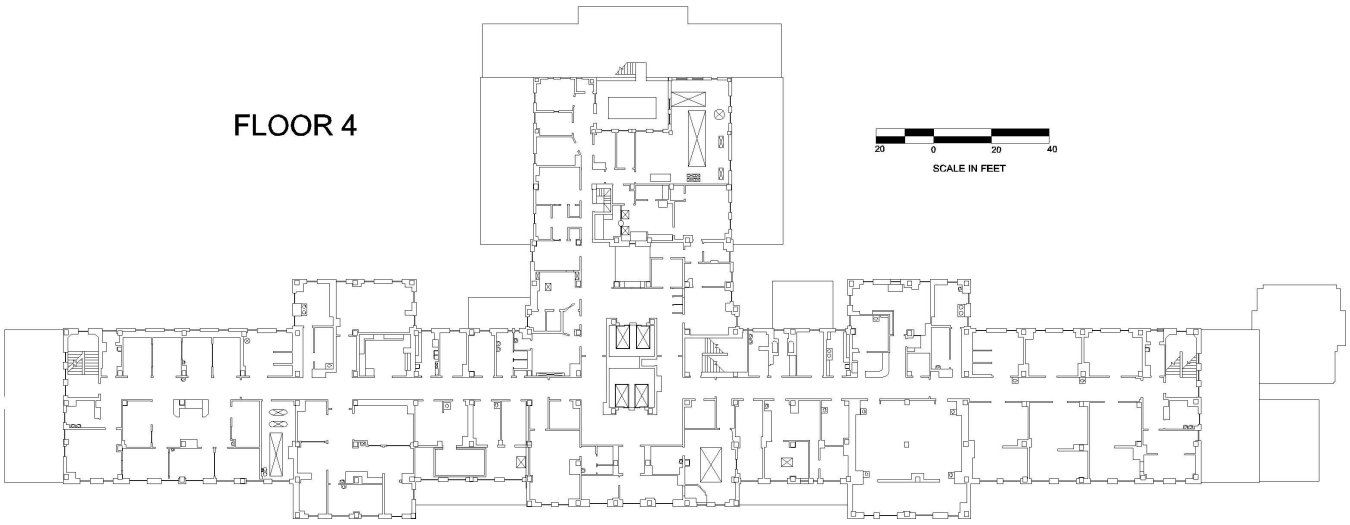

Figure 2: Complex environment in which the robots navigate.

For each simulation, we measured the percentage of area covered by the robot sensors at any time instant as well as the total for the simulation. Results are reported in Tables 1 and 2.

Figures 4 and 5 show the results of using various numbers of robots in the same environment for the same amount of time. We also tracked the amount of time that a robot was out of range as well as the time to reacquire communications in Table 3.

The effectiveness of the dispersion and exploration behaviors varied with the deployment environment. AVOID ROBOTS and FIND OPENINGS were both very successful in the house environment (see Table 1)

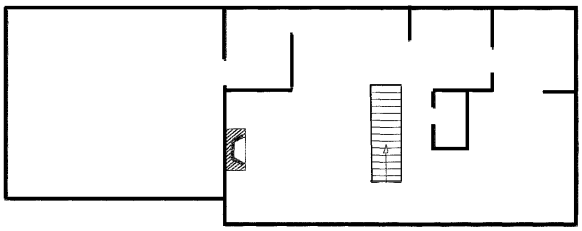

Figure 3: Simple home layout provides a less challenging task.

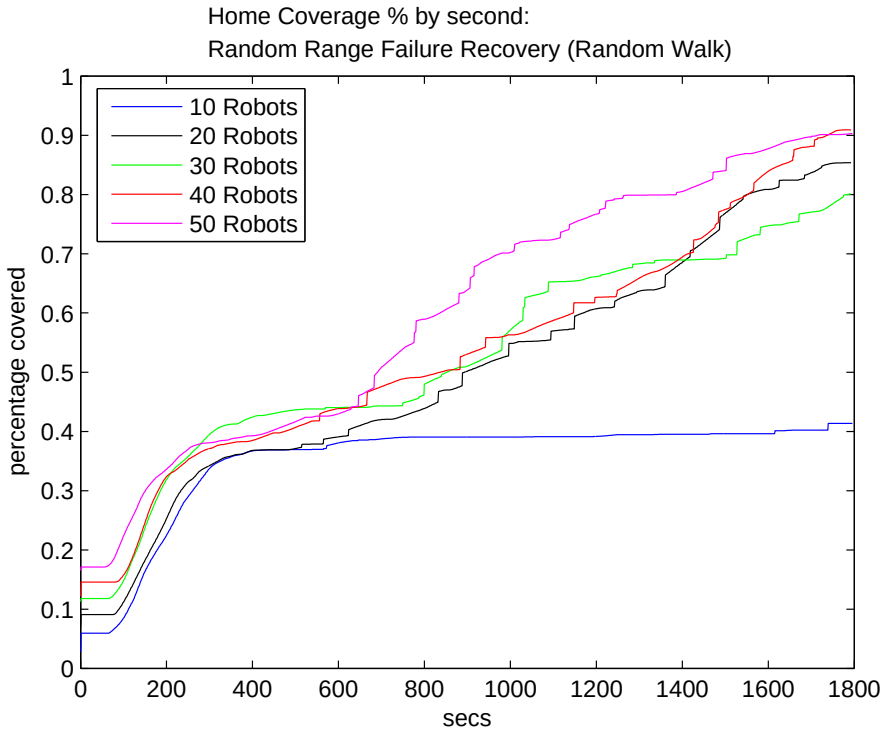

Figure 4: Percentage of home coverage with different numbers of robots as time progresses using RANDOM WALK and RANDOM RANGE.

Behavior	*10 Robots-Total*	*50 Robots-Total*	*10 Robots-Instant*	*50 Robots-Instant*
Random Walk[1]	41.37%	90.28%	16.91%	46.82%
Random Walk[2]	61.19%	78.52%	18.87%	41.98%
Avoid Robots[1]	71.13%	91.57%	18.9%	45.97%
Avoid Robots[2]	60.91%	85.18%	19.46%	50.28%
Find Openings[1]	94.32%	98.74%	20.29%	50.33%
Find Openings[2]	92.96%	98.08%	16.93%	50.51%

Table 1: Results of home simulations after a running time of 30 minutes.[1] employs the RANDOM RANGE control. [2] employs BACKTRACK RANGE control.

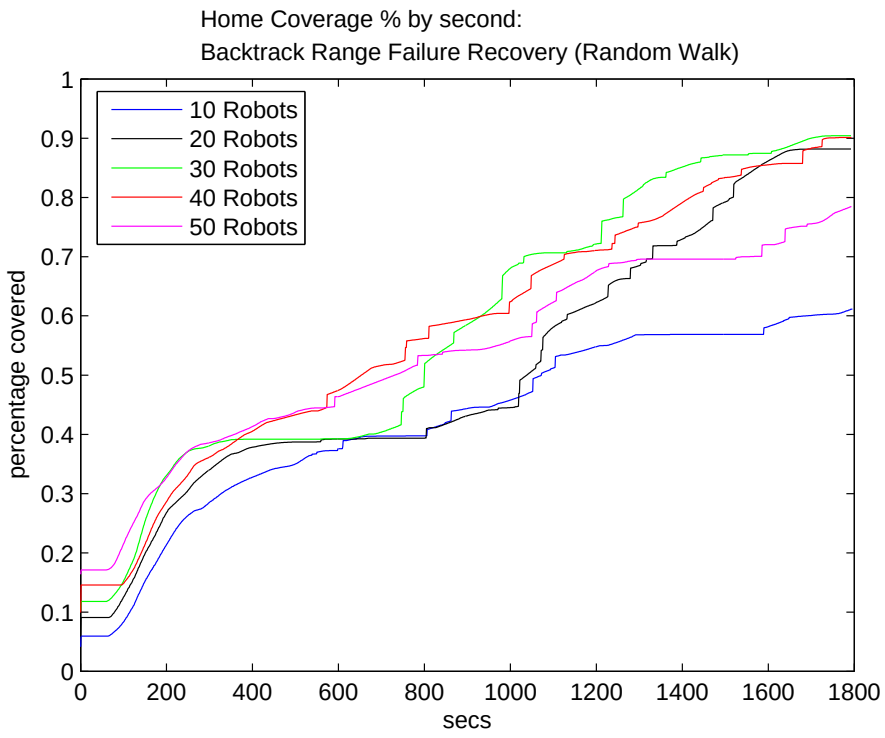

Figure 5: Percentage of home coverage with different numbers of robots as time progresses using RANDOM WALK and BACKTRACK RANGE.

Behavior	*10 Robots-Total*	*50 Robots-Total*	*10 Robots-Instant*	*50 Robots-Instant*
Random Walk[1]	19.77%	36.41%	4.68%	6.33%
Random Walk[2]	17.05%	18.05%	4.74%	8.12%
Avoid Robots[1]	20.61%	55.32%	4.79%	13.03%
Avoid Robots[2]	24.14%	55.75%	4.55%	17.49%
Find Openings[1]	2.7%	5.8%	2.24%	5.03%
Find Openings[2]	2.84%	4.47%	2.07%	4.23%

Table 2: Results of hospital simulations after a running time of 1 hour. [1] employs RANDOM RANGE control. [2] employs BACKTRACK RANGE control.

as measured by total coverage. FIND OPENINGS needed less time to explore the environment than the other behaviors. However FIND OPENINGS did not do as well in the larger environment of the hospital (see table 2). The FIND OPENINGS had a tendency to move the robot into the edge of doorways which caused a stall. Although all the behaviors had the same stall reaction, it seemed that stalled robots were unable to free themselves when stuck in doorways. The hospital environment had more doorways and they were narrower.

The RANDOM WALK tended to have more variable results. In Figures 4 and 5, it is apparent that the number of robots had less effect on the performance of the algorithm than the length of time. In addition, Figure 6 shows that RANDOM WALK makes advances in bursts whereas the AVOID ROBOTS tends to make steady progress (see Figure 7). Although the range recovery actions were different, the progress made by the various AVOID ROBOTS simulations were closer together than the progress made by RANDOM WALK.

Dispersion success, unlike exploration, is measured by the total area covered at any instant in time

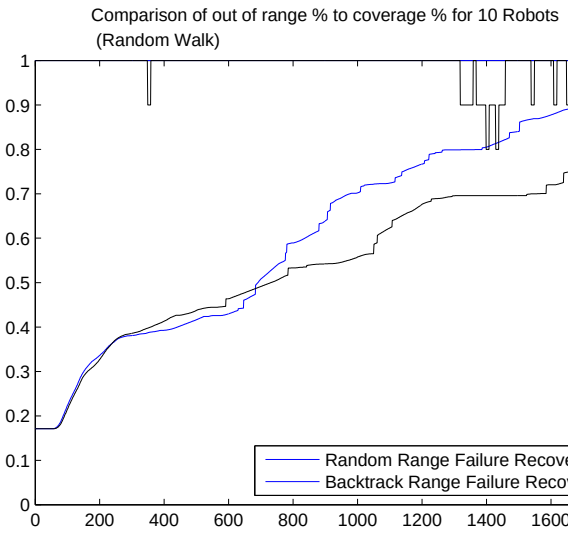

Figure 6: Comparison of performance of methods to get back into communications range for 10 robots using Random Walk.

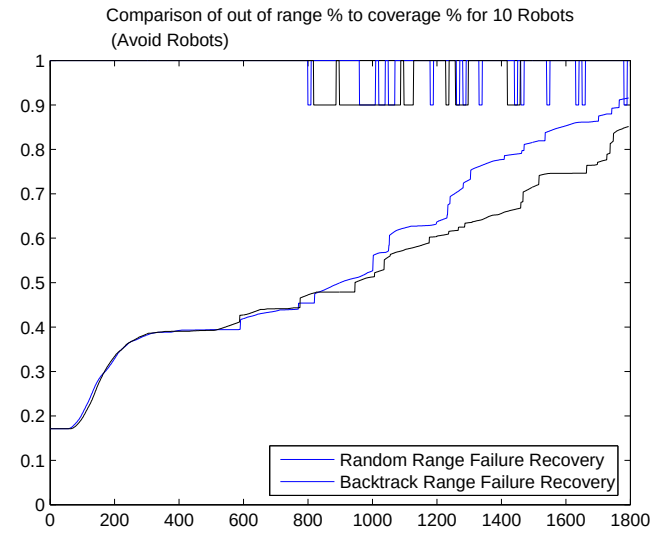

Figure 7: Comparison of performance of methods to get back into communications range for 10 robots using Avoid Robots.

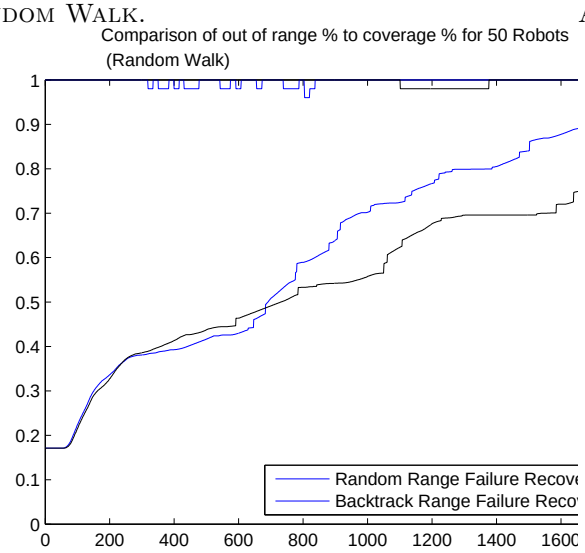

Figure 8: Comparison of performance of methods to get back into communications range for 50 robots using Random Walk.

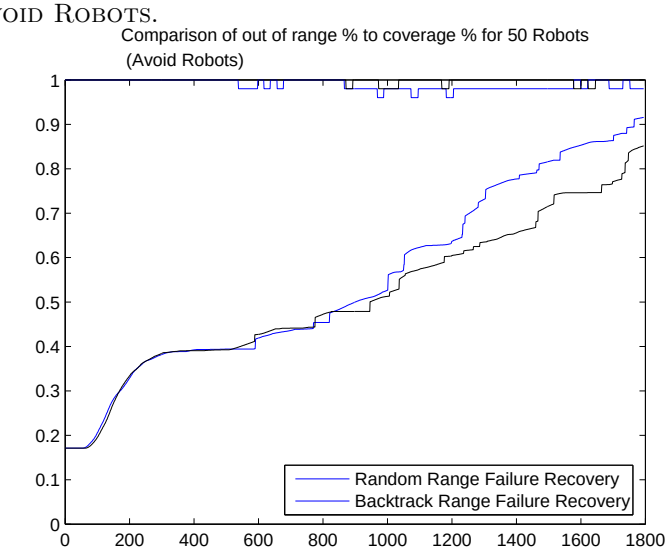

Figure 9: Comparison of performance of methods to get back into communications range for 50 robots using Avoid Robots.

given in Tables 1 and 2. This number is naturally bounded by the number of robots and the environmental constraints (i. e. doors, walls, etc). The maximum instant coverage was essentially the same for all behaviors in the house environment. However, the more complex environment of the hospital hindered the spread of the robots. Avoid Robots fared better here since it was designed to allow robots to repel each other. However in practice, as we have stated, the effective implementation of this behavior depending the sensors and processing available may not be easily done in physical robots.

The Random Range Control was no more effective than the Backtrack Range behavior (see Table 3). One reason may be that Backtrack Range had to rely solely on odometry to move to previous

Robots	*Mean-Random Recovery*	*Mean-Backtrack Recovery*
10	101.04	126.05
20	123.31	138.69
30	48.2	115.33
40	158.33	143.7
50	126.05	159.13

Table 3: Communications reestablishment time by number of robots and recovery action

positions. Since odometry is not reliable and moving to previous positions may be hampered by obstacles and errors, the robots may not have managed to move very far from the position where communications was lost.

Another finding in our experiments was the natural tradeoff between maintaining communications and exploring the area. The actions taken to reestablish communications slowed exploration progress, as shown by the plateaus in the coverage graphs at points were communications were lost by one or more robots.

5 Conclusions and Future Work

We have presented different methods for a group of robots to disperse in an unknown environment while remaining within communications range. We make minimal assumptions on the capabilities of the robots. In particular, we do not assume the robots know their own position or know where the other robots are or have been. Although random movements are largely effective, better behaviors can be created that can work more quickly depending on the sensors available.

Future work will include taking the behaviors we have presented here and implementing them on a team of robots to validate the time to cover an area in addition to the effectiveness of the range control algorithms.

An important question we have not addressed is how to deal with cases where the robots need to reach and maintain a steady state. This would be the case, for instance, when dispersing robots in a building and then watching for motions. Once the dispersion has been completed, the robots need to switch to a state where they do not move except that if a neighbor robot goes out of commission then they need to reposition themselves to guarantee coverage and communications.

References

[1] M. A. Batalin and G. S. Sukhatme. Spreading out: A local approach to multi-robot coverage. In *Proc. Int'l Symp. on Distributed Autonomous Robotic Systems"*, pages 373–382, 2002.

[2] S. Bergbreiter and K. Pister. Cotsbots: An off-the-shelf platform for distributed robotics. In *Proc. IEEE/RSJ Int'l Conf. on Intelligent Robots and Systems*, 2003.

[3] E. Bonabeau, M. Dorigo, and G. Theraulaz. *Swarm Intelligence: From Natural to Artificial Systems.* Oxford University Press, Oxford, England, 1999.

[4] B. P. Gerkey, R. T. Vaughan, K. Stöy, A. Howard, G. S. Sukhatme, and M. J. Matarić. Most valuable player: A robot device server for distributed control. In *Proc. IEEE/RSJ Int'l Conf. on Intelligent Robots and Systems*, pages 1226–1231, Oct. 2001.

[5] A. Howard, M. J. Mataric, and G. S. Sukhatme. An incremental self-deployment algorithm for mobile sensor networks. *Autonomous Robots*, 13(2):113–126, Sept. 2002.

[6] T.-R. Hsiang, E. Arkin, M. A. Bender, S. Fekete, and J. Mitchell. Algorithms for rapidly dispersing robot swarms in unknown environments. In *Proc. 5th Workshop on Algorithmic Foundations of Robotics (WAFR)*, 2002.

[7] S. Koenig, B. Szymanski, and Y. Liu. Efficient and inefficient ant coverage methods. *Annals of Mathematics and Artificial Intelligence*, 31:41–76, 2001.

[8] S. Poduri and G. S. Sukhatme. Constrained coverage for mobile sensor networks. In *Proc. of the IEEE Int'l Conf. on Robotics and Automation*, pages 165–172, 2004.

[9] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan. The cricket location-support system. In *Proc. 6th ACM MOBICOM*, Aug. 2000.

[10] I. Rekleitis, V. Lee-Shue, A. P. New, and H. Choset. Limited communication, multi-robot team based coverage. In *Proc. of the IEEE Int'l Conf. on Robotics and Automation*, Apr. 2004.

[11] P. E. Rybski, S. A. Stoeter, M. Gini, D. F. Hougen, and N. Papanikolopoulos. Performance of a distributed robotic system using shared communications channels. *IEEE Trans. on Robotics and Automation*, 22(5):713–727, Oct. 2002.

[12] R. Simmons, D. Apfelbaum, W. Burgard, D. Fox, M. Moors, S. Thrun, and H. Younes. Coordination for multi-robot exploration and mapping. In *Proc. Nat'l Conf. on Artificial Intelligence*, pages 852–858, 2000.

[13] I. A. Wagner, M. Lindenbaum, and A. M. Bruckstein. MAC vs PC – determinism and randomness as complementary approaches to robotic exploration of continuous unknown domains. *Int'l Journal of Robotics Research*, 19(1):12–31, 2000.