MULTI-ROBOT COVERAGE WITH DYNAMIC COVERAGE INFORMATION COMPRESSION

Zachary Wilson, Taylor Whipple and Prithviraj Dasgupta
Computer Science Department, University of Nebraska, NE 68182, Omaha, U.S.A.

Keywords: Multi-robot coverage, Exploration, Information compression.

Abstract: We consider the problem of distributed terrain or area coverage of an initially unknown environment using a set of mobile robots. We describe a distributed algorithm that is able to solve the distributed coverage problem without having each robot exchange its complete coverage map with other robots. The central part of our technique is a compression algorithm used by a robot to approximate the regions that have been previously covered and a fitness function that calculates the degree of accuracy of the approximated coverage information. The operation of our coverage algorithm is evaluated through experiments on simulated as well as physical Corobot robots. We have quantified the extent of overhead introduced by our coverage algorithm to prevent robots from performing repeated coverage. Overall, our results show that the robots are able to cover the environment within different environment settings while significantly reducing the amount of coverage information communicated between different robots.

1 INTRODUCTION

Over the past few years, multi-robot area or terrain coverage has emerged as an important research direction in multi-robot systems (Stachniss et al., 2008). Using multiple, autonomous robots to perform the coverage task in a distributed manner offers several advantages that improve the system's performance - the system can scale efficiently with an increase in the number of robots which in turn improves the speedup of the coverage task, the communication overhead in the system can be reduced as compared to a centralized system, and the system is robust against failures of individual robots. However, a major challenge in implementing a multi-robot system is to design efficient coordination behaviors among the different robots. Two such challenges in the context of multi-robot coverage are: the recording and integrating the coverage histories of all robots and making autonomous decisions about future coverage regions for each robot based on these histories, and reducing the energy and time expended in the coverage process by avoiding re-covering regions that have already been covered. Several researchers have developed multi-robot systems that describe different techniques to perform area coverage while addressing these issues. These techniques range from dispersion-based mechanisms where robots do not record or exchange individual coverage information with each other (Batalin and Sukhatme, 2002; Howard et al., 2002), to approaches in which the environment is modeled as a graph and the coverage problem is treated as constructing the least-cost spanning tree (Hazon and Kaminka, 2008; Rutishauser et al., 2009). In contrast to these approaches, our paper targets the middle ground lying between pure dispersion-based approaches, and, approaches that build a complete map or representation of the environment on each robot. In this paper, we investigate a complementary yet important issue in multi-robot coverage - reducing the amount of coverage information exchanged between robots in the system while ensuring that coverage performance is not adversely affected. We describe a coverage algorithm that interleaves coverage information exchange between robots with dynamic compression of the coverage information to cover the environment with limited communication between the robots. We have verified our coverage algorithm via experiments on both simulated and physical Corobot robots while showing that a system of multiple robots using our coverage algorithm is capable of covering different types of environments with very limited communication overhead.

Wilson Z., Whipple T. and Dasgupta P..
MULTI-ROBOT COVERAGE WITH DYNAMIC COVERAGE INFORMATION COMPRESSION.
DOI: 10.5220/0003538202360241
In *Proceedings of the 8th International Conference on Informatics in Control, Automation and Robotics* (ICINCO-2011), pages 236-241
ISBN: 978-989-8425-75-1

2 MULTI-ROBOT COVERAGE WITH COMPRESSED COVERAGE MAPS

We consider a scenario where R mobile robots are deployed into an initially unknown two-dimensional environment. Each robot is localized with respect to the environment using an on-board localization device. The objective of the multi-robot distributed coverage problem is to explore the environment while ensuring that the area collectively covered by the robots is maximized and the overlap between the regions covered by different robots is minimized. The basic idea in our approach is to enable each robot to repeatedly select a portion of the environment to cover by defining a bounded polygon. The robot then records the boundary coordinates of this polygon and uses a deterministic coverage pattern to cover the inside of the polygon. Before proceeding to define another polygonal block the robot has to inform other robots about the region within the block that it has just covered, so that other robots do not repeat coverage of this region. Communicating this information as is could incur significant communication overhead of communicating many coordinate points, as well as significant computational overhead of merging overlapping polygons. To address this problem, we describe a coverage algorithm where each robot compresses its coverage information using a polygonal approximation algorithm prior to communication. This results in each robot getting an approximate, yet fairly accurate map of the regions that are known to be covered, so that it can avoid repeated coverage of previously explored regions and navigate towards previously unexplored regions.

The state transition diagram for a robot in our system is shown in Figure 1. Each robot has four states of operation - block clearing, block define, dispersing and history-compressing. To begin coverage, each robot selects an unexplored location - a location that, according to the robot's current coverage information in its coverage history, has not yet been covered by itself or by another robot. After selecting an unexplored location, the robot starts defining the boundary of the block it plans to cover.

Defining and Covering Blocks. The first step for a robot r performing coverage is to select a block or sub-region to cover. To achieve this, a robot first defines the block virtually as a regular polygon with n vertices and each edge of length d, with the start and end vertices of the polygon rooted at the robot's current location. The robot then uses the block defining behavior to trace the edges of this virtual polygon and

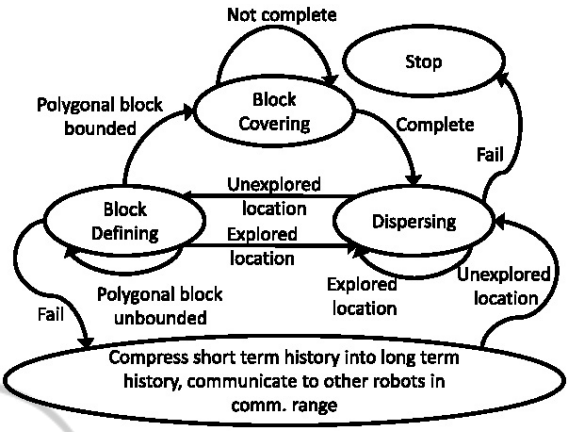


Figure 1: State transition diagram for a robot's behaviors.

determines if any edges or vertices of the block are occupied by obstacles or if any portion of the block is unreachable. On reaching each vertex of the block the robot is defining, a robot checks to see if any other robot within its communication range has reported information that its current location has already been explored. In this case, the robot aborts block-defining, discards the set of recorded vertices, and disperses to an unexplored location to start defining another block. On the other hand, if the robot completes defining the bounded polygon block without encountering a previously explored location, it changes its state to block covering and starts covering the inside of the block whose boundary it just defined; a robot uses the online, single-robot spanning tree coverage (STC) algorithm (Gabriely and Rimon, 2001) to cover a block whose boundary it has defined.

After a robot has finished covering a single block, it selects the next block to cover. At the block selection level, a robot prefers to cover regions that are adjacent to its most recently covered block, if such a region is available for coverage. Such a local block selection rule ensures that the coverage performed by a single robot in a navigable, previously uncovered region remains contiguous and 'holes' of uncovered regions that need to be covered later are not left in between covered blocks. Contiguous block selection is realized by a robot by selecting the start point for the next block as the vertex along the edge of the recently covered block, that is farthest from the start point of the covered block, as illustrated in the bottom part of Figure 3. The vertices of contiguous blocks covered by a robot are appended to its short term coverage history.

Map Compression and Map Merging. As robots cover multiple polygonal blocks, the size of the short term coverage history grows linearly for each robot. However, communicating the short term history be-

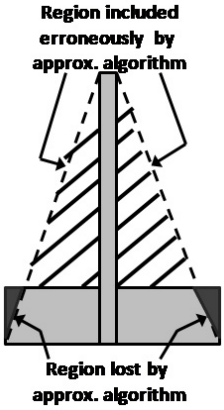


Figure 2: Effect of vertex compression of two overlapping rectangles with a combined polygon with $m = 4$. The gray area (light and dark) is the *desiredArea*, the stroked area is the *gainedArea* and the dark gray area is the *lostArea*.

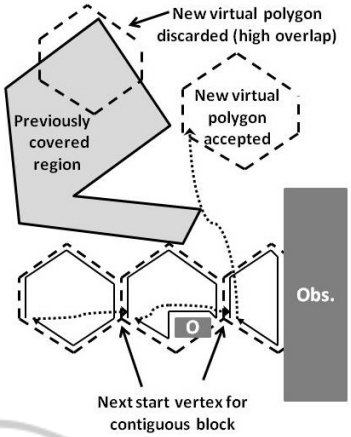


Figure 3: Path of a robot showing the definition and coverage of contiguous blocks. When contiguous block definition fails due to the obstacle on the right-end of the diagram, the robot iteratively selects a new virtual polygon whose percentage overlap with previously covered regions falls below a certain threshold.

tween all pairs of robots in the system is expensive if boundary vertices are uncompressed. Additionally, making calculations for determining covered regions with a large number of vertices on every robot involves considerable overhead. To prevent the rapid growth of the coverage information, a robot compresses its covered boundary vertices using a polygonal approximation algorithm called *compressPolygon* for a line curve called the $min-\varepsilon$ algorithm (Perez and Vidal, 1994), and combines the compression with the coverage information it has received from other robots. It then recompresses the combined region. This algorithm ensures that, after compression, the region is bounded by only a fixed number of vertices denoted by m. However, using an approximation algorithm for compressing a set of polygons introduces an error in the compressed region, as illustrated in Figure 2. To mitigate the effect of the erroneous region introduced by the compression algorithm, we have used a fitness function that tries to balance the loss in desirable region (denoted by *lostArea*) and the gain in undesirable region (denoted by *gainedArea*) as compared to the actual area of the combined, but uncompressed polygons (denoted by *desiredArea*), as given below:

$$fitness = w \times \frac{desiredArea - gainedArea}{desiredArea + gainedArea} + (1-w) \times \frac{desiredArea - lostArea}{desiredArea}. \quad (1)$$

The first fraction in Equation 1 denotes the error introduced by the amount of unnecessary area included in the approximated polygon while the second fraction denotes the error introduced by the loss in actual area of the uncompressed, combined polygons due to the approximation. The weight w determines the preference between these two errors, and, depending on the application domain, can be used to tolerate or reject repeated coverage. The value of the fitness function is used to determine whether the approximated polygon will be accepted for storage in the robot's long term history. If the fitness function value lies above a threshold of f_{Thr}, the approximated polygon is accepted. On the other hand, if this value lies below the threshold, the approximated polygon is rejected and the original polygons are stored in the long term history as disjoint polygons without combining or compressing them.

To combine the polygonal covered regions received from other robots' long term history, a robot first determines if the polygon representing its short term history overlaps with the polygons in the long term history. If there is no overlap, the robot stores these two polygons as disjoint polygons in its long term history; otherwise, it calculates the convex hull of the combined polygons, uses the polygon vertex-compression algorithm and fitness function value described above to store the the polygon with a fixed number of vertices within its long term history. Immediately after updating the long term history, a robot communicates its long term history to other robots within its communication range.

Dispersing and Selecting Unexplored Locations. A robot needs to select a new location to start coverage from when it is initially placed in the environment and after it has completed covering a series of contiguous blocks. The new start location must not overlap with any of the regions previously covered by the robot itself or by other robots. Also, the contiguous block selection by a robot fails when the edge of the next

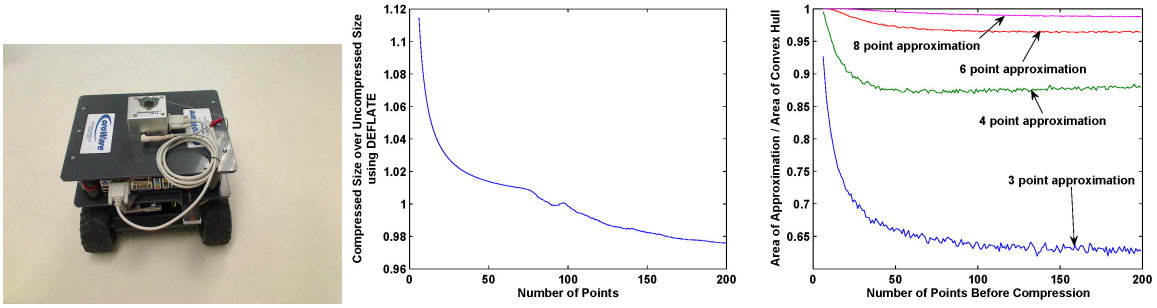
block it is planning to select is occluded by an obstacle or a wall, as shown in Figure 3. In this scenario too, the robot has to select a previously unexplored location to start defining a new block. Selecting a new location requires the robot to check its own short term coverage history, as well as its fused long term coverage history. The scenario in which a robot selects an unexplored location is shown in Figure 3. The robot first selects a virtual polygonal area p at random within the environment. It then calculates the overlap between this region p and the region it is aware of as being covered from its coverage history. If the percentage overlap is below a threshold that is initialized to $O_{Thr,init}$, the virtual polygon p is accepted as the polygon to start coverage from next and the robot selects one vertex of p to as the start vertex for using *singleBlockDefine* with this polygon. If the percentage overlap crosses the threshold for n_{TRIES} successive attempts, the robot reattempts to select another virtual polygon at a different location while increasing the threshold by a factor δ_{Thr}. If a polygon with overlap below $O_{Thr,max}$ cannot be found, the robot aborts trying to select an unexplored location and stops. If the *selectUnexploredLocation* algorithm successfully returns a valid location, the robot uses an A* search algorithm to plan its path to reach the location.

3 EXPERIMENTAL RESULTS

For verifying our coverage algorithm we have run several experiments using Corobot robots within a robot simulator called Webots. The Corobot robot is a four-wheeled platform. Two cross-beam IR sensors with a maximum range of 80 cm were placed on the front of the robot for obstacle avoidance while one IR sensor with a maximum range of 80 cm was placed on each side of the robot to enable wall-following. Each robot has wireless communications capabilities, and a localization module (Hagisonic Stargazer RS kit) that provides 2-D coordinates of the robot's position within the reference frame of the environment and has an accuracy within ± 2 cm of the robot's actual location. A photograph of the Corobot robot is shown in Figure 4(a). The default speed of the robot was set to 5.2 cm/sec. A robot defined a single block as a square ($n = 4$) with each side's length as $d = 1$ m. The distance at which a robot checks if it is still inside the virtual block while following a wall is set to 0.1 m (one half the length of a side of the robot). For the map compression algorithm, the fitness threshold value for discarding a compressed polygon is $f_{THr} = 0.0$ to achieve a fair balance between introducing undesirable region and discarding desirable regions. Finally, for the *selectUnexploredLocation* algorithm, the initial and maximum values of the overlap threshold for discarding a polygon are $O_{Thr,init} = 0.0$ (start optimistically with a target of finding a polygon with no overlap), and $O_{Thr,max} = 0.95$ (stop finding new locations for starting a block define when 95% of two polygons overlap in successive tries) respectively. The threshold increment, in case of successive n_{TRIES} failures, is $\delta_{Thr} = 0.05$, so that the *selectUnexploredLocation* algorithm does not take excessively long to reach the termination condition when successive attempts to find a new location end in failure.

Evaluating the min-ε Compression Algorithm. In our first set of experiments we measure the efficiency of the min- ε algorithm used for coverage information compression in our coverage algorithm. For the dataset used as input for this algorithm, we generated 2-D point-sets with set-sizes ranging from $4 - 200$. Each point set corresponds to a set of polygons within a 10×10 m^2 region. Figure 4(b) shows the effect of using the standard zip algorithm called deflate (Salomon, 2006) to compress different sizes of the point sets. We observe that the zip algorithm achieves very nominal compression, close to 2% for point set sizes > 150. This can be attributed to the fact that the data points in each point set are generated randomly and there is very little, if at all any, correlation between the points in each set. In contrast, our coverage information algorithm first computes a convex hull of the points in each point set, then performs min-ε compression on the hull, ensuring that the amount of data stored is constant. However, constant data size comes at the cost of the accuracy of the region enclosed by the polygons in the point sets. In Figure 4(c), we show the decrease in the fidelity of the region when we approximate the region in a convex hull using the min-ε algorithm. We see that when the compression ratio is higher (3 points per convex hull), the accuracy quickly decreases to about 65%. On the other hand, using more points in the min-ε compression results in higher accuracy of the region. Based on these results, for the rest of our experiments we have used 4 points (as parameter m) in the min-ε algorithm to approximate the region enclosed by the convex hull, which yields about 90% accuracy in the approximated region.

Webots Experiments. For our next series of experiments we verified the operation of our coverage algorithm. The arena for these experiments is 20×20 m^2 square region. We consider four different environments within this arena: (a) no obstacles (b) 10% of the arena's area occupied by obstacles, (c) 25% of its area occupied by obstacles, and, (d) walled office space with two narrow regions connected by a nar-



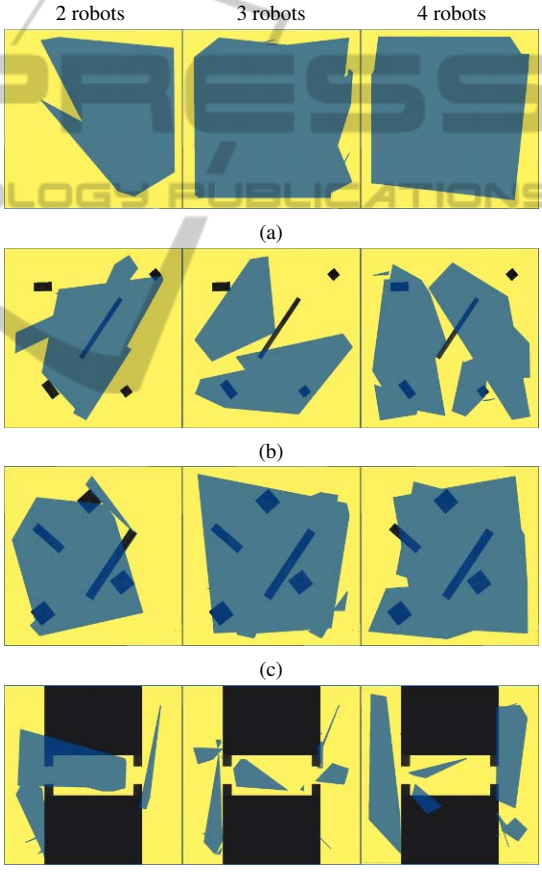
(a) (b) (c)

Figure 4: (a) Photograph of a Corobot fitted with a Stargazer localization device. (b) Compression ratio obtained using the DEFLATE zip library. (c) Ratio of the area of the approximated polygon calculated using the min-ε algorithm and the area of the convex hull for different sizes ($m = 3, 4, 6$ and 8 vertices) of the compressed set.

row channel. All experiments were run for a duration of 2 hours (real time), unless otherwise stated. In each environment, 2, 3 and 4 robots were used and the robots were started at random locations within each environment. All results are averaged over 10 simulation runs.

In our first set of experiments under this category, we observe the area covered by the robots in the different environment configurations. The coverage achieved by 2, 3 and 4 robots in the different environment settings after 2 hours of simulation are shown in Figures 5(a) - (d). We observe that across the different environments, as the number of robots increases the coverage performance improves. Overall, we see that within 2 hours with 4 robots we are able to get complete coverage in most of the environments, except the corridor environment; however, we also observe that when the environment becomes more complex, the robots have to decide on regions to cover more often and therefore end up with less coverage in the 2 hours. For the corridor environment complete coverage was obtained after 8 hours of simulation.

For our next set of experiments, we quantify the overhead of our coverage algorithm in terms of the distance traveled by the robots, as shown in Figures 6 (a) and (b). We observe that as we increase the number of robots the amount of useful region covered per robot reduces while the amount of overhead region increases, due to robots covering longer distances to select previously unexplored locations. Finally, we also observe that as the environment becomes complex with more obstacles, the overhead distance of the robots increases. This observation can be attributed to the fact that more obstacles in the environment result in robots encountering those obstacles and aborting their block definition more often, thereby leading to more overhead distance while traversing between covered regions. Overall, this set of experiments show that complex environments, as well as increasing the number of robots within the same environment result in more overhead distance being traveled by the



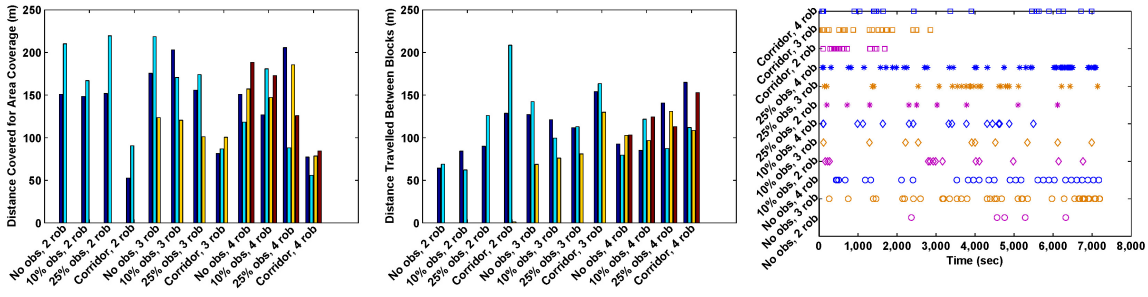
(a)

(b)

(c)

(d)

Figure 5: The amount of coverage achieved with 2 (left column), 3 (middle column) and 4 (right column) robots in the different environment settings after 2 real-time hours of simulation in Webots within a 20×20 m^2 arena (a) with no obstacles in it, (b) with 10% of its area occupied by obstacles, (c) with 25% of its area occupied by obstacles, (d) divided into corridors.



(a) (b) (c)

Figure 6: Distances traveled (m) by each robot (a) while covering the environment and (b) while not covering the environment (overhead) under the different robot and environment configurations. (c) Frequency of communication between robots in the different scenarios.

robots.

The final set of experiments under this category shows our main objective of this paper - the efficacy of the coverage information compression algorithm on the amount of communication between the robots. By using the min-ε algorithm that encodes any convex polygon with a fixed number ($m = 4$) of 2-D data points, we ensure that every communication between any pair of robots will be contain only 4 2-D data points. If k bytes are used to encode real numbers, each communication will be no more than $m \times 2 \times k$ bytes. The results of this experiment shown in Figure 6(c) illustrate that the frequency of coverage data transmission varies from infrequently to moderately frequently, depending on the environment. We performed all the experiments reported here on the physical Corobot robots as well and got commensurate results (omitted for space constraints).

4 CONCLUSIONS

In this paper we have described a distributed coverage algorithm for a multi-robot system that dynamically compresses the coverage information to reduce the communication overhead between the robots. Our ongoing work is focussed on dynamically adjusting the dimension of a block depending on the coverage performance of each robot and investigating lossless compression techniques for the coverage data to increase the accuracy of the fused covered regions.

REFERENCES

Batalin, M. and Sukhatme, G. (2002). Spreading out: A local approach to multi-robot coverage. In *in Proc. of 6th International Symposium on Distributed Autonomous Robotic Systems*, pages 373–382.

Gabriely, Y. and Rimon, E. (2001). Spanning-tree based coverage of continuous areas by a mobile robot. *Annals of Math and Artificial Intelligence*, 31:77–98.

Hazon, N. and Kaminka, G. (2008). On redundancy, efficiency, and robustness in coverage for multiple robots. *Robotics and Auto. Systems*, 56(12):1102–1114.

Howard, A., Mataric, M., and Sukhatme, G. (2002). Mobile sensor network deployment using potential fields: A distributed, scalable solution to the area coverage problem. In *in Proc. of 6th Int'l. Symp. on Distributed Autonomous Robotic Systems*, pages 299–308.

Perez, J. and Vidal, E. (1994). Optimum polygonal approximation of digitized curves. *Pattern recognition letters*, 15(8):743–750.

Rutishauser, S., Correll, N., and Martinoli, A. (2009). Collaborative coverage using a swarm of networked miniature robots. *Robotics and Auton Systems*, 57(5):517–525.

Salomon, D. (2006). *Data Compression: The Complete Reference*. Springer.

Stachniss, C., Mozos, O., and Burgard, W. (2008). Efficient exploration of unknown indoor environments using a team of mobile robots. *Annals of Mathematics and Artificial Intelligence*, 52(2-4):205–227.