Reducing the Number of Mobile Sensors for Coverage Tasks

Yongguo Mei, Yung-Hsiang Lu, Y. Charlie Hu, and C. S. George Lee
School of Electrical and Computer Engineering, Purdue University
{ymei, yunglu, ychu, csglee}@purdue.edu

***Abstract*— Mobile robots provide new opportunities for research in environment sensing. By combining sensors and robots, we can develop applications to enhance security, search and rescue, and detect hazardous materials. One important problem is to use fewer mobile sensors to cover an area. This paper focuses on two problems in robot sensor deployment: speed management and detouring distance under energy and timing constraints. We determine the robots' traveling speed based on the remaining energy and remaining time before deadline. We use probabilistic models for environments with obstacles and propose an empirical analysis to estimate the extra traveling distance due to detouring. We compute the number of robots (fleet size) needed for cover an area and our approach reduces the fleet size. Compared with two simple heuristics, our method uses 21% fewer robots.**

I. Introduction

Sensors can be used in many applications, such as detecting chemicals for safety or animal migration for wildlife studies. Mobile robots can be combined with appropriate sensors and greatly expand the potential applications. For example, mobile robots may scan an area to detect land mines. Mobile robots may also be deployed to detect survivors after an earthquake. The robots' task are often constrained by time and energy capacity (usually rechargeable batteries). For example, detecting survivors should be accomplished within 24 hours after a disaster; otherwise, the chance of survival diminishes quickly. The timing constraint creates a deadline for the robots. Energy capacity is another constraint. It can be difficult or impossible to recharge the robots' batteries after deployment. Mobile sensors have to be deployed into the working field before performing their tasks. This paper studies the robot deployment under both timing and energy constraints.

A typical scenario for deployment consists of three steps: (a) The robots are transported to the working field by a carrier from a storage site, such as an emergency response center. (b) The robots are unloaded from the carrier in one or multiple locations. (c) The robots move to their individual initial locations and start executing their tasks. This paper considers the coverage task— the robots have to cover an area using the sensors installed on the robots. Coverage is a basic problem in many complex tasks, including lawn mowing, carpet cleaning, and land mine detection. Deployment is considered in our previous study for open fields without obstacles while all robots travel at the same speed [5]. This paper generalizes our previous study by considering different speeds for robots unloaded at different times and by adding obstacles into the environment. In this paper, we use scan-lines as the basic coverage strategy, as explained in [5]. A coverage task may require other techniques, for example, to cover different areas with a particular order. This involves a widely studied topic called path planning. In an environment with obstacles, the robots have to detect and avoid obstacles using, for example, radar or vision. This paper focuses on estimating the number of needed robots and does not consider path planning or obstacle detection.

If a robot has sufficient time to accomplish the coverage task, the robot should travel at the most energy-efficient speed, measured by the covered area per unit amount of energy. In contrast, if the deadline is approaching soon, the robot should travel faster even though the energy efficiency is lower. In this paper, we explain that each robot should use a constant speed but different robots may adopt different speeds. Two probabilistic models are developed to describe obstacles in the environment. The first models scattered obstacles and the second models clustered obstacles. We derive an empirical rule to calculate the additional distance each robot needs to travel for detouring around the obstacles. The empirical rule is validated through simulations. Our approach is compared with two other deployment strategies. One strategy unloads all robots at the same location. The second strategy unloads the same number of robots in multiple locations. Our method uses 21% fewer robots than these two strategies.

[1]This work was supported in part by the National Science Foundation IIS-0329061.

II. Related Work

Mobile robots can be used in many applications. Kurabayashi et al. [3] present an algorithm to coordinate multiple robots sweeping an area with movable obstacles. Rybski et al. [8] use small robots for reconnaissance and surveillance; they use large robots to transport small robots into the fields. Simmons et al. [9] study the control and coordination of multiple robots. They use deployment to validate their coordination method. The robots start from the same location and move to their target locations. Poduri et al. [6] study the deployment strategies for mobile sensor networks using artificial potential fields until desired connectivity among sensors is reached. Barili et al. [1] propose speed control to avoid dramatic speed changes to save energy. Their work does not consider timing and energy constraints. Taylor et al. [10] model the environment by boundary graphs and present their vision-based path planning algorithms to explore the area. However, they do not estimate the detouring distance.

Our previous study [5] presents a method to deploy multiple robots into a field for exploration purposes. The method is called Space Partition Area Coverage Algorithm (SPACA). The robots are transported by a carrier and unloaded at different locations. Three types of overhead are considered: unloading time, dispersing time and area overlap. Unloading time and dispersing time refer to the time for unloading the robots from the carrier and robots' dispersing from the unloading location to their working locations. The area overlap refers to the areas covered by two robots. An efficient deployment is presented to reduce the overhead and the fleet size, namely, to increase the average area covered by one robot. The robots are unloaded at different locations; each location corresponds to one group of robots. The method assigns each robot a rectangle to cover and keeps the minimum covered area similar in different groups. All robots travel at the same speed. The fleet size is calculated using the following steps. First, we assign a small number as the first group size and compute the area covered by this group. Second, we determine the size and covered area for the next group so that the minimum covered areas of the robots in different groups have similar sizes. This process continues by gradually adding more groups and more robots into each group until the whole area is covered. Since the unloading and dispersing time is considered, the process may fail to find a solution due to a tight timing constraint.

This paper extends our previous study in two ways: (1) Robots may travel at different speeds. A robot that is unloaded later may travel faster even though the energy efficiency is lower. (2) The area contains obstacles and the robots have to detour around the obstacles. This generalization makes our method applicable to more complex and realistic environments. The method presented in this paper is called SPACA+.

III. Problem Description

The area to be covered is two-dimensional with static obstacles. The robots have no map so they have to discover and respond to obstacles.

A. Traveling Speed

We assume that the sensor on each robot has a finite and constant sensing range. Thus, a robot can cover a larger area if the robot can travel across a longer distance. A power model of a mobile robot has been built in our previous work [4]. The power model $p(v)$ is a convex function of robot traveling speed v. We use v_m to represent the maximum speed of the robot. The most energy-efficient speed, v_o, is defined as the speed at which the robot has the minimum ratio of power and speed, $v_o = \underset{v}{\operatorname{argmin}}\{\frac{p(v)}{v}\}$, $0 < v_o < v_m$. Moving at speed v_o, a robot can travel the longest distance with the same amount of energy. However, if we consider the timing constraints, this may no longer be true. We use the symbol τ to denote the deadline, and $\mathcal{E}$ for the remaining energy. All robots have the same deadline. The speed management problem is to determine the speed at which each individual robot can travel the longest under the timing and the energy constraints.

B. Obstacles

The robots have no map of the area. However, we assume the density of obstacles is available. The obstacle density d_o is the ratio of area occupied by obstacles over the total area; it can be estimated by sampling a small portion of the total area. This paper explains the difference between open areas and areas with probabilistic obstacles. We estimate the additional distance each robot has to travel for detouring around obstacles. We use r_d as the ratio; in an open area, the ratio is one. Intuitively, r_d rises as d_o increases. When d_o is too high, however, a large portion of the area becomes inaccessible and r_d may decrease.

IV. Speed Management

Speed management determines a robot's speed as a function of time. The purpose of speed management is to increase the traveled distance under timing and energy constraints. Let $v(t)$ be a robot's speed at time t; the traveled distance is $\int_0^\tau v(t)dt$. Speed management finds $v(t)$ to maximize $\int_0^\tau v(t)dt$. The solution has to satisfy

three constraints: (1) $0 \leq t \leq \tau$, (2) $\int_0^\tau p(v(t)) \leq \mathcal{E}$, and (3) $0 < v(t) \leq v_m$. To determine $v(t)$, we consider the properties of power, $p(v)$: it is a monotonically increasing function of the speed v. Usually, it is a convex function $\frac{d^2p(v)}{dv^2} > 0$, as described in [2] [4].

Each robot should use a constant speed; the speed is determined by the energy capacity of the battery and the remaining time before the deadline. If a robot is unloaded early — with sufficient time before the deadline — the robot should adopt the most energy-efficient speed v_o. If a robot is unloaded late with only a short period of time before the deadline, the robot should move faster than v_o.

Lemma 1: If the $\mathcal{E} \leq \tau p(v_o)$, the robot can travel the longest distance at the most energy-efficient speed v_o.

Lemma 2: If $\mathcal{E} \geq \tau p(v_m)$, the robot can travel the longest distance at the maximum speed v_m.

Lemma 3: If $\tau p(v_o) \leq \mathcal{E} \leq \tau p(v_m)$, there exists a speed v_1 ($v_o \leq v_1 \leq v_m$) such that the robot can travel the longest and exhaust the energy at the deadline. The value of v_1 satisfies $p(v_1) = \frac{\mathcal{E}}{\tau}$.

The first two lemmas are intuitive, and the third lemma is a result of the convexity of the function $p(v)$. The proofs are omitted due to the limitation of space. The following theorem concludes our speed management strategy.

Theorem 1: With the initial energy $\mathcal{E}$ and the deadline τ, a robot can maximize the traveling distance at a constant speed, determined by the following rules:

- v_o, if $\mathcal{E} \leq \tau p(v_o)$.
- v_m, if $\mathcal{E} \geq \tau p(v_m)$.
- v_1 such that $p(v_1) = \frac{\mathcal{E}}{\tau}$, if $\tau p(v_o) \leq \mathcal{E} \leq \tau p(v_m)$.

V. Environments with Obstacles

In this section, we present two probabilistic models for obstacles. An empirical rule is developed to calculate the distance ratio r_d for different obstacle density d_o. We use simulations to validate the rule.

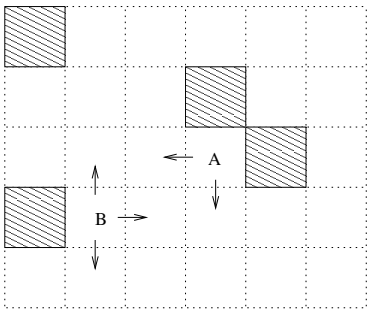


Fig. 1. Grid lines, cells and neighbor cells.

A. Obstacle Model

To model the environments, we make the following assumptions: (1) The area is divided into equal-size square cells. (2) Each cell is either open or occupied by an obstacle. (3) Obstacles do not move. (4) A robot can move from one cell to its four neighbors: upper, down, left, and right. (5) The density of the obstacles is known in advance but their exact locations are unavailable. Figure 1 shows an area with 5 row $\times$ 6 column cells. The four cells with shading are obstacles; the obstacle density is $\frac{4}{30} \approx 13.3\%$. A robot at cell 'A' can move only to its left and down cells.

We develop two environment models: random model and clustering model. In the random model, the probability of one cell occupied by an obstacle is equal to d_o and is independent of it neighbors. In the clustering model, the obstacle probability depends on its upper and left neighbors. We use four values p_{ul}, p_u, p_l and p_n to represent the obstacle probability of a cell in four different situations: (1) When both its upper and left neighbors are obstacles. (2) Only the upper cell is an obstacle. (3) Only the left cell is an obstacle. (4) Neither neighbor is an obstacle. Since an environment generated by the clustering model has a obstacle density of d_o, the following equation must hold: $d_o^2 p_{ul} + d_o(1-d_o)(p_l + p_u) + (1-d_o)^2 p_n = d_o$. We further assume that $p_u = p_l = k_1 p_{ul}$ and $p_n = k_2 p_{ul}$, where k_1 and k_2 are two constants. The value of p_{ul} can be calculated as $p_{ul} = \frac{d_o}{d_o^2 + 2d_o(1-d_o)k_1 + (1-d_o)^2 k_2}$, and the four probabilities can be uniquely determined. The values of k_1 and k_2 model the likelihood that a cell is occupied if its neighbors are occupied. When $k_1 < 1$ and $k_2 < 1$, p_{ul} is larger so a cell is more likely occupied if its upper and left neighbors are occupied. When $k_1 = k_2 = 1$, $p_{ul} = p_u = p_l = p_n = d_o$ and the clustering model degenerates to the random model. Figure 2 shows four areas generated with $d_o = 8\%$. In Figure 2 (b)-(d), the values of (k_1, k_2) are $(0.6, 0.1)$, $(0.8, 0.2)$, and $(0.7, 0.3)$, respectively.

B. Distance Ratio

Figure 3 shows an example of scanning an area of thirty cells with $d_o = \frac{4}{30} \approx 10.33\%$. This example illustrates how we count the distance to cover an area with obstacles. Since the top left cell is an obstacle cell, the robot starts from cell 'A'. The robot moves toward the right, until it reaches the end of the first row. Then, it moves down to the right end of the second row and heads left. At cell 'B', it encounters an obstacle cell. It observes that the next unvisited free (i.e. unoccupied) cell is cell 'C'. From 'B' to 'C', there are two shortest paths,

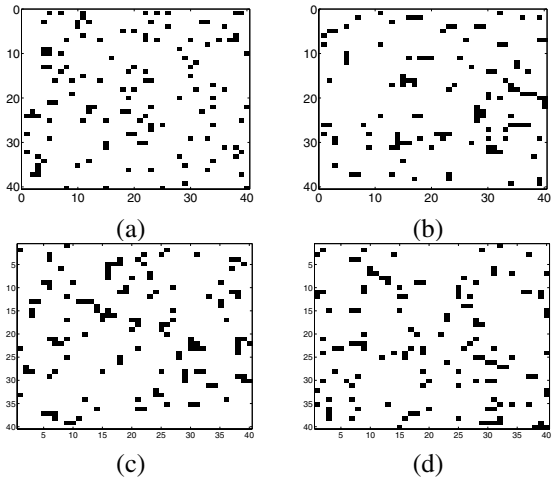


Fig. 2. Four examples of environments filled with probabilistic obstacles of density 8%: (a) generated by the random model; (b), (c), and (d) generated by the clustering model with different (k_1, k_2).

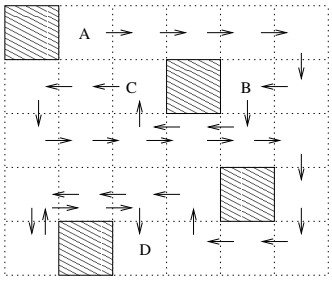


Fig. 3. Scanline covering an environment with obstacles

one above and the other below the obstacle. Suppose the robot chooses the lower one, and reaches cell 'C'. It continues moving toward the left until it finishes the second row. The robot then moves down to the left side of the third row and heads right. The middle three cells of the third row are visited by the robot twice. At the fourth row, the robot encounters an obstacle, and chooses the lower shortest path. At the last row, the robot encounters an obstacle cell again, and it chooses the only shortest path from the upper to reach the next accessible cell 'D'. The left three cells of the fourth row are visited twice. The right three cells of the last row have already been visited, so the robot stops at cell 'D'. The arrows in the figure show the whole path from 'A' to 'D'. There are totally 4 obstacle cells, and the robot visits 6 cells twice to avoid the obstacles. The length of this path is $5 \times 6 + 6 - 4 = 32$ steps. Without the obstacles, the robot scans the whole area using $5 \times 6 = 30$ steps. The distance ratio is $r_d = \frac{32}{30} \approx 1.067$.

To derive the relationship between r_d and d_o, we analyze two special cases as examples shown in Figure

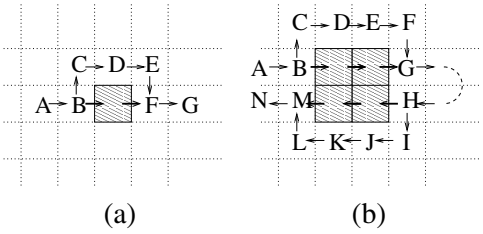


Fig. 4. Obstacles and traveling steps.

4. In Figure 4(a), a robot detours around an isolated obstacle in the sequence shown by the letters and the arrows. It takes two more steps (moving from one cell to a neighbor cell is counted as one step) to detour around this obstacle. In Figure 4(b), a robot detours around four obstacles clustering as a square. The letters and arrows show the traveling sequence. At cell 'G', the robot travels toward the right until reaching the boundary, turns down to the next row, and then travels left to 'H'. The steps from 'G' to 'H' are omitted in the figure with the dash curve. It takes four extra steps to detour around the four-cell obstacle. Suppose an area contains a total of N cells. If there is no obstacle, it takes N steps to cover this area. One obstacle cell increases d_o by $\frac{1}{N}$, and one extra step increases r_d also by $\frac{1}{N}$. If all the obstacles are isolated, similar to the one in Figure 4(a), an obstacle cell will incur two extra steps. Therefore, we can model the relationship as $r_d = 1 + 2 \times d_o$. This analysis can be applied to the clustering obstacles. If all the obstacles are clustered as in Figure 4(b), then the relationship is $r_d = 1 + d_o$. We model the relationship between r_d and d_o by using the following equation with a positive constant α:

$$r_d = 1 + \alpha \times d_o \tag{1}$$

To validate this model, we perform extensive simulation of different maps with different values of d_o. The simulation first randomly generates an area using either model, and then simulates the scanline-covering process. Two sets of simulations have been performed, and both assume that every robot knows its current location. The first set assumes that the robot has the map of the obstacles. The robot scans the area without considering timing or energy constraints. The robot starts from the top left and moves right, following the scanlines until all accessible free cells have been visited. If the robot encounters obstacles, it chooses the next unvisited accessible free cell along the scanlines, and then chooses a shortest path from the current cell to that unvisited free cell. The robot remembers the visited cells. Because the robot has the map and can determine the next unvisited

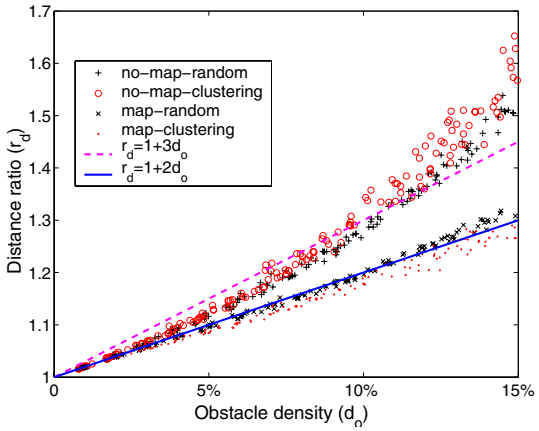


Fig. 5. Distance ratio verse obstacle density

accessible cell, this simulation provides a lower bound for r_d.

The second set of simulations is more realistic because the robot has no map. The robot starts from the top left and moves right, following the scanlines until covering the last row of cells. When the robot encounters obstacles, it detours around obstacles until the next free cell along the scanline is reached. There are two detouring directions: clockwise and counter-clockwise. When the robot moves toward the right, it detours in a counter-clockwise direction first to uncovered cells. Since the robot has no map, it does not know whether this detouring strategy is viable. If this strategy fails, the robot detours around the obstacles in a clockwise direction. When the robot moves toward the left, it tries the clockwise direction first. This simulation adopts a simple yet effective covering algorithm. Because the robot has no map, the robot travels a longer distance than the result in the first set of simulations.

We run simulations with obstacle densities ranging from 1% to 15%. Figure 5 shows the simulation results. The top two sets of data ($+$ and $\circ$) are from the second set of simulations. They are called "no-map-random" and "no-map-clustering" for the two probabilistic models. The other two sets of data ($\times$ and $\cdot$) are from the first set of simulations. They are called "map-random" and "map-clustering". Two lines enclose the range between $r_d = 1 + 2 \times d_o$ and $r_d = 1 + 3 \times d_o$. Figure 5 shows that, with maps, the robot can travel shorter distances to cover the same area. The results from our models are consistent. This shows that at a low d_o the two models have negligible difference. When the density is less than 10%, the relationship $r_d = 1 + 3 \times d_o$ is valid for the no-map simulations. With maps and $d_o < 14\%$, the ralationship can be closely approximated by $r_d = 1+2\times d_o$. From our stimulation results, the value of α in Equation (1) is between 2 and 3 for $d_o \leq 0.1$.

VI. Simulations

A. Simulation Setup

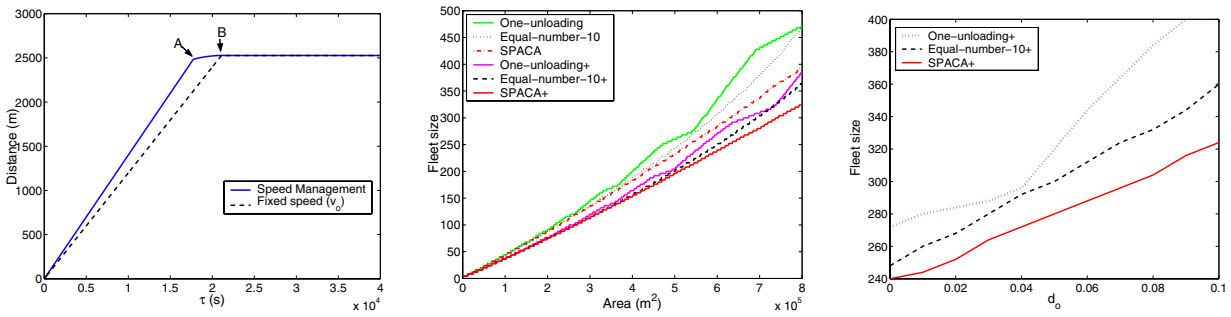
We use specifications and measurements of a commercial robot called PPRK for our case study. The robot is developed at Carnegie Mellon University [7]. It has three polyurethane omni-directional wheels driven by three MS492MH DC servo motors. The default energy capacity is $20736J$ for four AA batteries (1200 mAh and 1.2V). A data acquisition card is used to measure the voltage and current to calculate the robot's power at different speeds [4]. The power model used in this paper is $p(v) = 48.31v^2 - 3.37v + 0.69$. The maximum speed is $0.14m/s$, and the optimal speed is $0.12m/s$ with power consumption of 0.98W. The sensing distance is 0.8m.

We modify our previous deployment algorithm SPACA to incorporate the methods proposed in this paper. The new algorithm is named SPACA+. When a robot is unloaded, its speed is determined by the speed management method. Based on this speed, the distance it can travel under both the energy and timing constraints is computed. The original SPACA allows only open areas, where the distance ratio r_d is 1.0. We add the distance ratio r_d into the SPACA+, and the corresponding covering area to a certain traveling distance is smaller when r_d increases as the obstacle density d_o increases.

B. Speed Management

1) Traveling distance: The speed management scheme adjusts the traveling speed based upon the values of τ and $\mathcal{E}$. In Figure 6(a), τ varies from 0 to 3×10^4 seconds. The solid line shows the distance with speed management and the dash line shows the traveled distance at v_o. Before point 'A' (1.49×10^4 seconds), the timing constraint dominates and the robot travels at v_m. After point 'B' (2.11×10^4 seconds), the energy constraint dominates and the robot travels at v_o. Between the two points, the robot travels at speeds between v_o and v_m. At point 'A', the robot can travel $358m$, or 17%, longer than the fixed speed (v_o) method.

2) Robot Deployment: Figure 6(b) shows the number of robots of three different deployment strategies for $\tau = 4$ hours. The horizontal axis is the covered area in m^2. The three strategies are (1) SPACA, (2) one-unloading, and (3) equal-number-10. One-unloading unloads all robots in one place. Equal-number-10 unloads 10 robots each time. The top three curves are the three strategies when the robot travels at the constant speed v_o. The lower three curves use speed management by adding $+$ in the legend. We compute the fleet sizes at different



(a) (b) (c)

Fig. 6. (a) speed management and traveling distance (b) speed management and fleet size (c) fleet size and obstacle density

areas by increasing the area by 1600 m^2 each time. SPACA outperforms the other two deployment methods regardless whether speed management is applied. Compared with "one-unloading", our method SPACA+ uses 31% ($1 - \frac{324}{472} = 31\%$) fewer robots for covering $8 \times 10^5 m^2$.

C. Obstacle Density and Fleet Size

Figure 6(c) compares the three deployment strategies to cover an area of $6 \times 10^5 m^2$ with $\tau = 4$ hours. The value of α is 3 in this simulation. When d_o increases, the fleet size increases. This is because r_d increases as d_o becomes larger. All deployment strategies use speed management. SPACA+ uses 21% ($1 - \frac{304}{384} = 21\%$) fewer robots than the one-unloading strategy for $d_o = 8\%$.

VII. Conclusion and Future Work

This paper presents a strategy to reduce the number of mobile sensors for coverage tasks. Speed management selects the appropriate speed for each robot to maximize the traveled distance under timing and energy constraints. Two probabilistic models are used for environments with obstacles. We present an empirical rule to estimate the additional distance the robots need to travel for detouring around obstacles. Simulation results show that our approach uses fewer robots than the two other heuristics.

Our work can be extended in several directions. First, our method can be integrated with multi-robot coordination, task allocation, obstacle avoidance and path planning. Our study complements existing studies by estimating the number of robots needed. Second, in some environments, obstacles are not randomly located. For example, a building's walls are connected except at the doors. In such an environment, the detouring distance can become significantly longer along a wall. Third, we do not consider communication among robots. When multiple robots are used for coverage, they may communicate their knowledge of the environment and gradually create a map. We also assume that each robot knows its current location and the area that has been covered. If such information is unavailable, a robot may visit a place more than once before recognizing that the place has been covered. Even though this paper makes several simplification assumptions, our work provides a critical initial step for coverage by estimating the number of robots needed.

References

[1] A. Barili, M. Ceresa, and C. Parisi. Energy-Saving Motion Control for An Autonomous Mobile Robot. In *International Symposium on Industrial Electronics*, pages 674–676, 1995.

[2] E.-C. Corporation. *DC Motors, Speed Controls, Servo Systems, An Engineering Handbook*. Hopkins, 1973.

[3] D. Kurabayashi, J. Ota, T. Arai, and E. Yoshida. Cooperative Sweeping by Multiple Mobile Robots. In *ICRA*, pages 1744–1749, 1996.

[4] Y. Mei, Y.-H. Lu, Y. C. Hu, and C. S. G. Lee. Energy-Efficient Motion Planning for Mobile Robots. In *International Conference on Robotics and Automation*, pages 4344–4349, 2004.

[5] Y. Mei, Y.-H. Lu, Y. C. Hu, and C. S. G. Lee. Deployment Strategy for Mobile Robots with Energy and Timing Constraints. In *IEEE International Conference on Robotics and Automation*, pages 2827–2832, 2005.

[6] S. Poduri and G. S. Sukhatme. Constrained Coverage for Mobile Sensor Networks. In *IEEE International Conference on Robotics and Automation*, pages 165–171, 2004.

[7] G. Reshko, M. T. Mason, and I. R. Nourbakhk. Rapid Prototyping of Small Robots. Technical report, CMU-RI-TR-02-11, Carnegie Mellon University, 2002.

[8] P. E. Rybski, N. P. Papanikolopoulos, S. A. Stoeter, D. G. Krantz, K. B. Yesin, M. Gini, R. Voyles, D. F. Hougen, B. Nelson, and M. D. Erickson. Enlisting Rangers and Scouts for Reconnaissance and Surveillance. *IEEE Robotics and Automation Magazine*, 7(4):14–24, December 2000.

[9] R. Simmons, D. Apfelbaum, D. Fox, R. P. Goldman, K. Z. Haigh, D. J. Muslineg, M. Pelican, and S. Thrun. Coordinated Deployment of Multiple, Heterogeneous Robots. In *IROS*, pages 2254–2260, 2000.

[10] C. J. Taylor and D. J. Kriegman. Vision-Based Motion Planning and Exploration Algorithms for Mobile Robots. *IEEE Transaction on Robotics and Automation*, 14(3):417–426, 1998.