International Journal of Control, Automation, and Systems (2013) 11(2):369-376 ISSN:1598-6446 eISSN:2005-4092
DOI 10.1007/s12555-011-0184-5 http://www.springer.com/12555

Time-Efficient and Complete Coverage Path Planning Based on Flow Networks for Multi-Robots

Adiyabaatar Janchiv, Dugarjav Batsaikhan, ByungSoo Kim, Won Gu Lee, and Soon-Geul Lee*

Abstract: Complete coverage path planning (CCPP), specifically, the efficiency and completeness of coverage of robots, is one of the major problems in autonomous mobile robotics. This study proposes a path planning technique to solve global time optimization. Conventional algorithms related to template-based coverage can minimize the time required to cover particular cells. The minimal turning path is mostly based on the shape and size of the cell. Conventional algorithms can determine the optimum time path inside a cell; however, these algorithms cannot ensure that the total time determined for the coverage path is the global optimum. This study presents an algorithm that can convert a CCPP problem into a flow network by exact cell decomposition. The total time cost to reach the edge of a flow network is the sum of the time to cover the current cell and the time to shift in adjacent cells. The time cost determines a minimum-cost path from the start node to the final node through the flow network, which is capable of visiting each node exactly once through the network search algorithm. Search results show that the time-efficient coverage can obtain the global optimum. Simulation and experimental results demonstrate that the proposed algorithm operates in a time-efficient manner.

Keywords: Cellular decomposition, cleaning robot, complete coverage path planning, multi-robot, time efficiency.

1. INTRODUCTION

The complete coverage path planning (CCCP) algorithm determines the path that a robot must follow to pass through every space in a given workspace. The CCPP algorithm has gained considerable attention in recent decades because of its wide application in robotics, including functions such as cleaning, mining, inspection, and exploration, among others.

Completeness and time efficiency of coverage are key factors to consider to improve the performance of the algorithm. With completeness taken into consideration, various approaches to coverage algorithms have been proposed. Cellular decomposition is one of the most robust approaches to the completeness problem. Choset [1] introduced boustrophedon decomposition, a coverage algorithm based on exact cellular decomposition. Boustrophedon decomposition is an enhancement of trapezoidal decomposition. In the aforementioned study, the narrow cells were merged into one cell to reduce the excessive motions because the trapezoidal decomposition requires excessive redundant back-and-forth motions to cover a number of narrow cells between IN and OUT events to achieve completeness. Wong and McDonald [2,3] proposed a cell decomposition algorithm based on the topological structure of a map representation.

In the early stage of coverage development, research focus was directed on the completeness of coverage [1-3]. Such interest has currently been expanded to include both time efficiency and completeness of coverage [4-6]. Existing methods have been combined to address these issues. For instance, several studies [5,6] combined template-based path planning with heuristic coverage planning. The use of predefined templates for a given small region ensures completeness of coverage, whereas a heuristic coverage algorithm can help optimize energy and time according to the shape of a given region. This type of algorithm only determines the optimal time path inside a particular cell despite its role in time optimization, which indicates that the solution to time efficiency is only a local and not a global optimum. Other studies [4,7] used cell decomposition with a grid map representation, a technique similar to the boustrophedon approach, to achieve completeness. These studies also adopted template-based path planning to achieve efficiency with two kinds of template motions. In the present study, back-and-forth as well as spiral motions have been selected depending on the current cell. Surve *et al*. [8] minimized the coverage time by using the

Manuscript received November 9, 2011; revised May 2, 2012 and October 7, 2012; accepted January 2, 2013. Recommended by Editorial Board member Fuchun Sun under the direction of Editor Hyouk Ryeol Choi.

This research was partially supported by the Korea Evaluation Institute of Industrial Technology (No. 10035544) and the Implementation of Technologies for Identification, Behavior, and Location of Human based on Sensor Network Fusion Program (Grant Number: 10041629) of the Ministry of Knowledge Economy (MKE), Korea.

Adiyabaatar Janchiv, Dugarjav Batsaikhan, and ByungSoo Kim are with the Department of Mechanical Engineering, Kyung Hee University, Yongin, Korea (e-mails: janchiv@wagnerasia.com, djbsn_1986@yahoo.com, bskim4284@khu.ac.kr).

Won Gu Lee and Soon-Geul Lee are with the Industrial Liaison Research Center, Kyung Hee University, Yongin, Korea (e-mails: {termylee, sglee}@khu.ac.kr).

* Corresponding author.

curvature path instead of the back-and-forth motion. However, this path is incompatible with small environments with obstacles. Multi-robot systems can more rapidly complete and further execute a task compared with single robots [9]. Therefore, multi-robots can reduce the time required to complete a task over an entire area.

This study proposes a time-efficient CCCP algorithm that can achieve both completeness and time efficiency. The algorithm converts the CCPP problem into a flow network problem by exact cellular decomposition. Therefore, the total time cost to reach the edge of network equals the total time required to cover the present cell and the time spent to shift between adjacent cells. The algorithm then determines a minimum-cost path from the start node to the final node through the flow network, which visits each node exactly once by using the network search algorithm. The search result provides the global optimum time-efficient covering order over the cells, thus generating a set of predefined template paths for each cell.

2. TIME-EFFICIENT AND COMPLETE COVERAGE PATH PLANNING ALGORITHM

The coverage algorithm used in this study primarily aims to determine the global optimum time required to cover an entire environment. A robot covers each cell with one of the 12 templates consisting of several back-and-forth motions.

2.1. Cell decomposition algorithm

One of the most robust approaches to the completeness problem involves the divide-and-conquer strategy, which is a generalization of cellular decomposition. Cell decomposition methods are commonly classified into exact and approximate methods by map representation. Trapezoidal, boustrophedon, and Morse decompositions are typical techniques related to exact cellular decomposition. This study uses boustrophedon decomposition for coverage. The environment $\varepsilon = \mathbb{R}^2$ is assumed to contain an obstacle region $o \subset \varepsilon$. If both ε and o can be of any polygonal shape; the free space $\varepsilon_f \subseteq \varepsilon$, is defined as

$$\varepsilon_f = \varepsilon - o, \tag{1}$$

which is the free section of the environment.

The purpose underlying the exact cellular decomposition is to partition the free space of the environment into disjoint sets called cells. These cells form the nodes of a flow network, which is a non-directed connectivity graph G. Two nodes are connected by an edge if and only if the corresponding cells are adjacent.

The boustrophedon decomposition assumes that a vertical line referred to as slice and is denoted by a thick red line in Fig. 1 sweeps from left to right through a bounded environment with arbitrarily shaped obstacles. Cells are formed via a sequence of open and close events, which occurs when the slice meets the first or the last

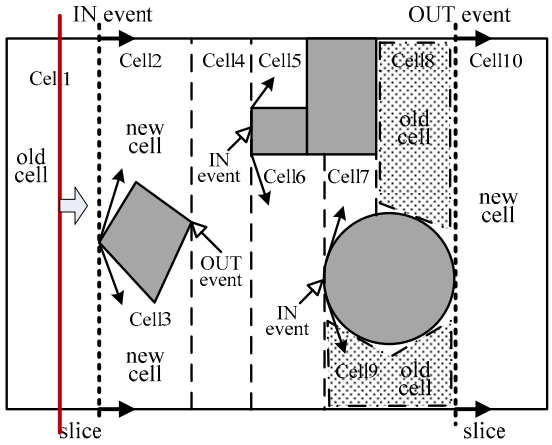


Fig. 1. IN and OUT events.

point of an obstacle. The open and close events are referred to as IN and OUT, respectively. When the slice meets the first obstacle, denoted by the leftmost thick dotted line in Fig. 1, the current cell, Cell1, is closed; two new cells, Cell2 and Cell3, are opened at an IN event. An OUT event is the opposite case of an IN event. Denoted by the rightmost thick dotted line in Fig. 1, Cell8 and Cell9 are closed, and a new cell, Cell10, is opened at the OUT event. The IN event can be viewed as a cell breaking up into two cells, whereas the OUT event consists of two cells merging into one [1].

When arbitrarily shaped obstacles are present in the workspace, each decomposed cell is either rectangular such as Cell1, or a figure with two parallel sides such as Cell8 in Fig. 1. Despite its non-straight line(s) on more than one side, such as Cell8 or Cell9, the cell can be approximated as a trapezoid. The trapezoid is represented by a shaded shape similar to that in Cell8, or as a combined shape with more than a trapezoidal area similar to that in Cell9.

The input to the algorithm is a type of information on the environment; this information consists of obstacle vertices and the size of the environment. Fig. 2 shows the output of the algorithm, which includes the cell size and the flow network that indicates the adjacency of the cells. The flow network is implemented using modern adjacency lists.

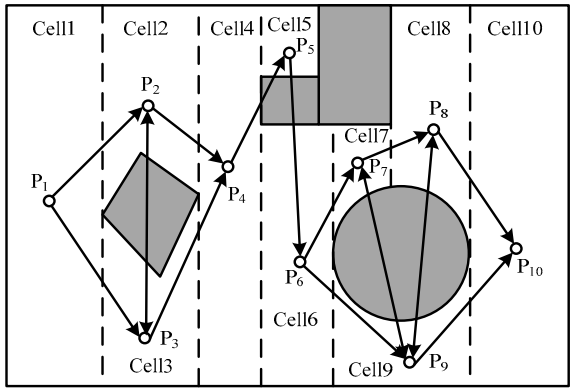


Fig. 2. Output of cellular decomposition and flow network.

2.2. Search algorithm

The robot can search for the global optimum path in time by decomposing the environment into cells and using the flow network. This study primarily aims to implement this approach. The goal of the search algorithm is to determine the minimum cost path from the start node to the final node by visiting every node exactly once in the given flow network. Simply stated, the solution to the problem is the path that contains every node in the flow network. Thus, the path must begin at the start node and end at the final node.

2.2.1 Comparison study with different network search problems

Several problems in combinatorial optimization are similar to the one addressed in this study. Problems that involve combinatorial optimization include the minimum spanning tree (MST) problem, the traveling salesman problem (TSP), and the shortest path problem (SPP). The solution to the MST problem includes every node in one graph; however, the solution is a tree instead of a path. The solution to the SPP contains the minimum cost path from the initial node to the final node but may not include every node. The solution to the TSP includes all nodes in a graph; this solution is a path but ends with the initial node. All algorithms for the TSP assume that a graph is completely connected. However, the flow network is not completely connected, and only adjacent nodes are joined by an edge. Thus, the existing algorithms cannot be used directly. The proposed search algorithm performs two functions: identification of all possible spanning paths from the initial node to the final node and determination of the minimum cost-spanning path among all possible paths.

2.2.2 Templates

As mentioned in Section 2.1, decomposed cells are either rectangular or trapezoidal. A cell consists of two parallel sides regardless of the shape and can be quantitatively equivalent to a rectangle with the same as the target cell. Fig. 3 shows the equivalent rectangles of Cell8 and Cell9. The equivalent rectangle has the same path topology and time cost as those of the original cell; that is, a case with arbitrarily shaped obstacles can be handled by using a case with rectangular obstacles. The non-rectangular cells are replaced with corresponding rectangular cells. Henceforth, the case in which all obstacles are rectangles will be described.

The robot covers each cell with only back-and-forth motions if all decomposed cells are rectangles. The main path inside a cell is the path along the long axis of the cell because such movement along the main path reduces the number of turns, thus improving time efficiency. A path connecting adjacent main paths is referred to as an interval, as shown in Fig. 4. The interval path of a cell starts at one corner, which is called the start point, and moves back and forth along the main paths and ends at another corner, which is called the end point. The number of such 2 permutations of the four corners of a rectangular cell is denoted by ${}_4P_2 = 12$. Fig. 4 shows 12

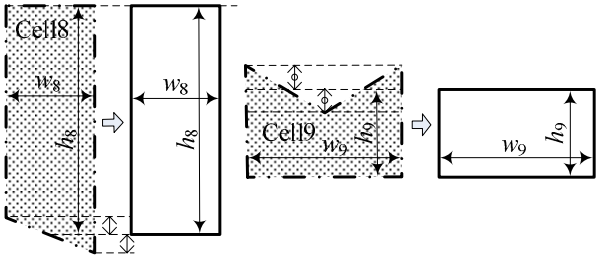


Fig. 3. Equivalent rectangles.

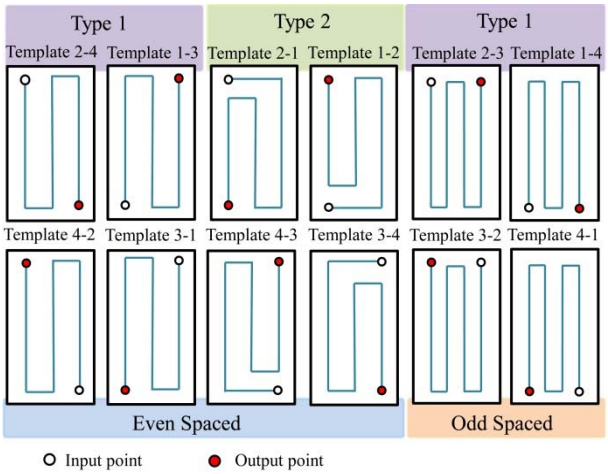


Fig. 4. Twelve templates used in the study.

cases classified as templates depending on the positions of their start and end points, where the first and second numbers are the corner numbers of the start and end points, respectively. The corner number is equal to the number at the left bottom and increases clockwise by one.

We denote a minimum number of main paths and intervals for each template. Any sized cell can be covered with the 12 templates provided, and no alternative template remains. The templates are classified into odd-spaced and even-spaced templates according to the number of intervals used. If the start and end points of the template are placed along the horizontal axis, then the template is odd-spaced. Otherwise, the template is even-spaced, which can be further decomposed into Types 1 and 2, depending on the time computation method used.

2.2.3 Computation of time cost of the templates

The total time required to cover the current cell is a function of cell size, template type, and robot size and speed. The template of the current cell is determined by the location of the previous and consequent cells. The cell size is determined by cell decomposition. The size of the robot and its speed are known from specification. Given the above information, the time cost for a particular template is computed by the following steps:

First, the width (w_c) and the height (h_c) of the configuration space are calculated from the size of the cell and the diameter of the robot (D_R). The configuration space is a set of all robot configurations in which the robot does not overlap an obstacle, as shown in Fig. 5.

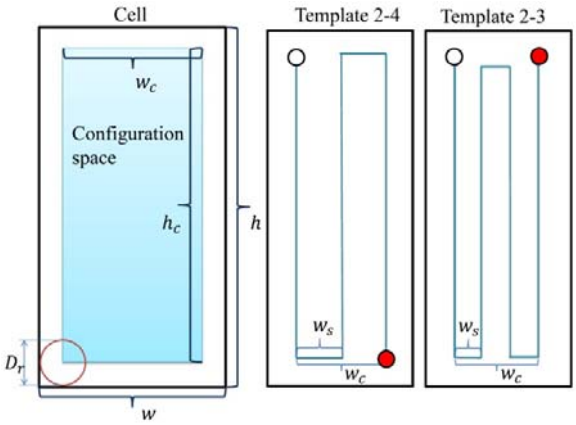


Fig. 5. Configuration space and even/odd templates.

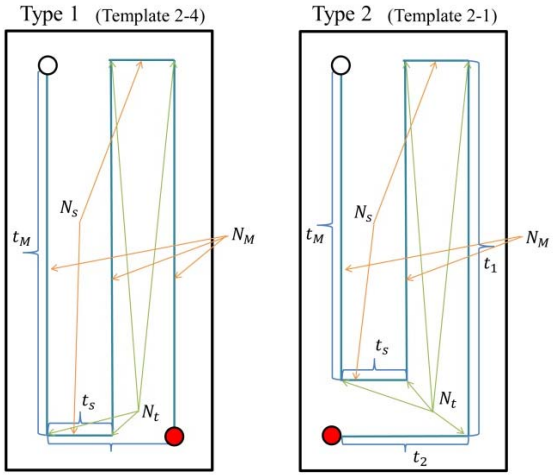


Fig. 6. Definition of the number of main paths, intervals, and turns.

$$h_c = h - D_R, \tag{2}$$

$$w_c = w - D_R, \tag{3}$$

where h and w represent the height and the width of the cell, respectively.

Second, the length of the interval (w_s) is computed while maintaining completeness and without increasing the redundancy of coverage. The interval depends on the size of the configuration space and the type of template. As depicted in Fig. 5, the odd and even templates have different lengths of interval.

Finally, the time cost of the template is computed as follows:

$$t_{24} = N_M * t_M + N_s * t_s + N_t * t_t \text{ (Type1)}, \tag{4}$$

$$t_{23} = N_M * t_M + N_s * t_s + N_t * t_t + t1 + t2 \text{ (Type2)}, \tag{5}$$

where t_{24} and t_{23} are the total times required to cover Templates 2-4 and 2-3, respectively; t_t represents the time required for a turn, and t_M is the time required to cover a main path; t_s denotes the time required to cover an interval, which can be expressed as (6). Fig. 6 shows that the number of intervals (N_s), the number of main paths (N_M), and the number of turns (N_t) in the current template are defined by (7) and (8).

$$t_s, t_N, t_1, t_2 = \begin{cases} \Sigma_{i=1}^{n} t_i & for\ d > 40\text{cm} \\ \Sigma_{i=1}^{n-1} t_i & for\ 30\text{cm} < d < 40\text{cm} \\ \Sigma_{i=1}^{n-2} t_i & for\ 15\text{cm} < d < 30\text{cm} \\ \Sigma_{i=1}^{n-3} t_i & for\ 10\text{cm} < d < 15\text{cm} \\ \Sigma_{i=1}^{n-4} t_i & for\ 5\text{cm} < d < 10\text{cm} \\ \Sigma_{i=1}^{n-5} t_i & for\ 0\text{cm} < d < 5\text{cm}, \end{cases} \tag{6}$$

where n denotes the number of different speeds (n = 6), $t_i = d / V_i$, and d is distance between the current position of the robot and the turning point.

$$N_s = w_c / w_s, \quad N_M = N_H + 1, \quad N_t = 2 * N_H \text{ (type1)}, \tag{7}$$

$$N_s = w_c / w_s, \quad N_M = N_H, \quad N_t = 2 * N_H \text{ (type2)}. \tag{8}$$

2.2.4 Computation of time cost between two adjacent cells

To calculate the time cost between adjacent cells, each node of the flow network must carry the information of the corresponding cell. This information includes the geometry, start point, and end point of the cell. After cell decomposition, the nodes in the flow network retain only the information on the size of the corresponding cells, except for the start node. However, the start node stores the information on the size of the corresponding cells, as well as the information at the start point. The start point of the start node is the point closest to the location of the initial location of the robot to one of the four cell corners.

The cost between adjacent cells is computed in two steps. As illustrated in Fig. 7, the start and end points of the current cell are determined based on the previous and subsequent cells, respectively. The start point of the initial node is determined from the initial location of the robot. Therefore, within each step of the search algorithm, an end point of the current cell and a start point of the next cell are determined by searching the closest couple points between one of the four corners of the current cell and one of the four corners of the next cell.

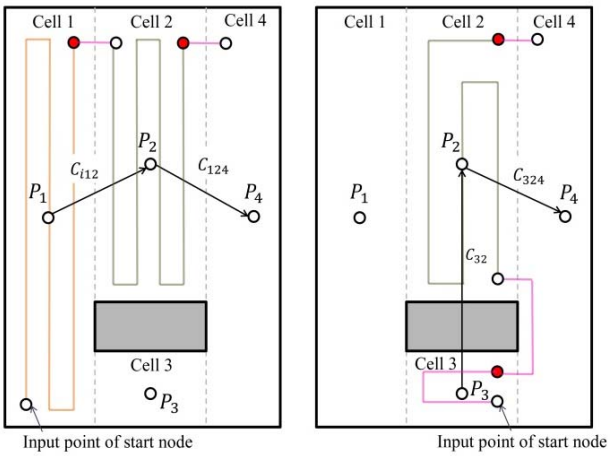


Fig. 7. Two different time costs to an edge.

A corresponding template is chosen based on the set of the start and end point of the current cell. The cost between adjacent cells is the summation of the time required to cover the current cell and the time spent to shift between the adjacent cells. As shown in Fig. 7, the time cost between Cell2 and Cell4 may vary depending on the previous cell.

$$^{1}C_{24} = t_{23} + s_1, \tag{9}$$

$$^{3}C_{24} = t_{43} + s_1, \tag{10}$$

where $^{1}C_{24}$ is the time cost covering Cell2 and Cell4 when the previous cell is Cell1; $^{3}C_{24}$ is the time cost covering Cell2 and Cell4 when the previous cell is Cell3; t_{23} and t_{43} denote the time costs for Templates 2–3 and 4–3, respectively; and s_1 is the time required to shift between adjacent cells.

3. SIMULATION

Simulation using a single robot is conducted in a LabView environment to verify the influence of the proposed algorithm on the time efficiency of the generated path. The robot is assumed to be operated in an enclosed rectangular workspace, and information on the workspace is obtained by the same method used in a previous study [1]. Map building and localization are completed by wall following with a laser range finder (LRF) and StarGazer. During wall following, LRF also gathers information on obstacles. With the combination of the LRF data reconstruction and the StarGazer position data, the contours of the obstacles are obtained by line extraction. The workspace measures 5 m × 5 m and has two rectangular obstacles. The speed control of the robot is the same in simulation and experiment. The robot can move at six different speeds (V_i), and its normal operating speed equals the maximum speed of 22.6 cm/sec. The robot must decelerate when it approaches a turning point; the robot begins to decelerate before it reaches 40 cm from a turning point and gradually slows down with constant distance. Furthermore, the turning time (t_t) of the robot is assumed as 5 sec in the simulation.

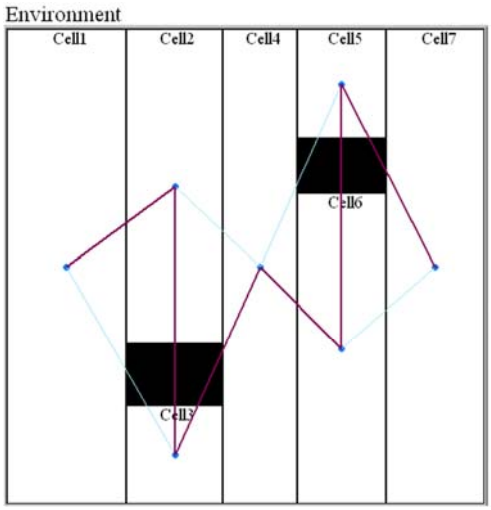


Fig. 8. Minimum cost path through a flow network.

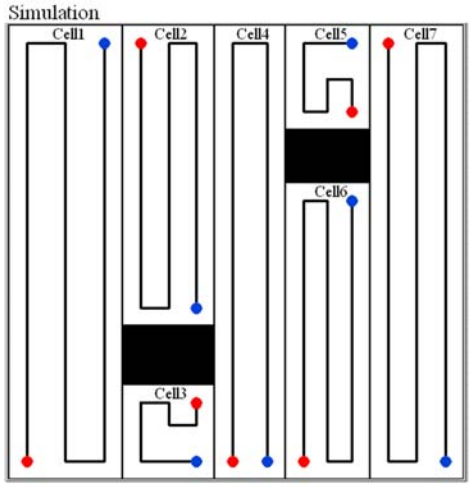


Fig. 9. Generated coverage path.

Table 1. Simulation result.

Covering order	# of turns	Total time (sec)
1-2-3-4-6-5-7	41	602
1-2-3-4-5-6-7	46	650
1-3-2-4-6-5-7	54	682
1-3-2-4-5-6-7	50	665

Figs. 8 and 9 show the generated minimum cost path with the flow network as well as the coverage path from the simulation. The time-efficient order in which the cells are to be cleaned by the robot is presented as Path = {Cell1, Cell2, Cell3, Cell4, Cell6, Cell5, Cell7}. The optimum time-efficient path is compared with all other possible paths to show the improved time efficiency of the proposed algorithm. Table 1 shows the simulation result in which all possible paths are compared. The total number of turns and the time spent to complete the coverage task are also shown.

4. EXPERIMENT

Experiments with two robots are conducted with two different obstacle placements. To demonstrate the practical efficiency and robustness of the proposed algorithm, the experimental results are compared with the simulation results by using the two performance indexes.

4.1. Experimental setup

The size and shape of the workspace and obstacle locations of the experimental setup are similar to those in the simulation. The landmarks are attached beneath the ceiling of the room for localization, as shown in Fig. 10. Figs. 11 and 12 illustrate the two different experimental setups. The purpose of changing the obstacle location is to validate the robustness of the proposed algorithm for various environments. The lines attached on the floor do not contribute to the experiments and are merely used to estimate visually the sensor error of localization. The marked lines also provide convenience in observing the experiments with the naked eye.

Fig. 10. Passive landmarks beneath the ceiling.

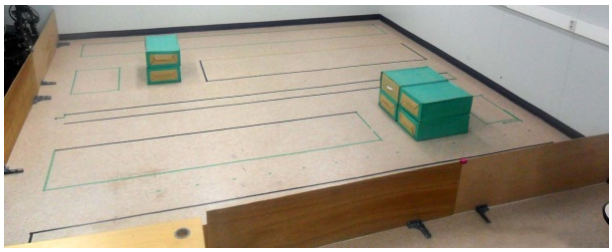


Fig. 11. The first layout of obstacles for experiment.

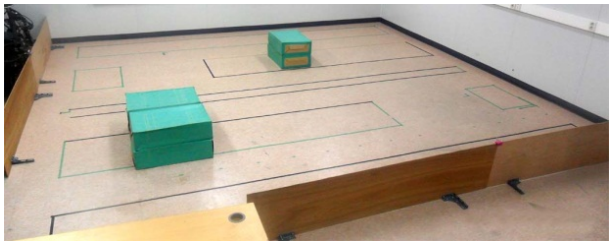


Fig. 12. The second layout of obstacles for experiment.

4.2. System setup

Two differential-driven mobile robots ("X-bot" from Yujin Robot Co., Ltd.) are used to accomplish the experiment on the proposed CCPP algorithm. The robots are equipped with two sensors, as illustrated in Fig. 13. One is a laser range sensor for recognizing indoor environmental components such as walls and static obstacles. This sensor is also used to avoid collision with obstacles and other robots.

Stargazer, a landmark-based indoor navigation sensor system, is used for localization to execute the coverage task, The robots determine their positions by detecting and recognizing landmarks beneath the ceiling (Fig. 10) with the onboard sensor unit. Stargazer provides robust data on its position and heading angle with high resolution for localization. All sensors and robot controllers are interfaced to the remote supervisory control computer with a wireless universal serial bus hub.

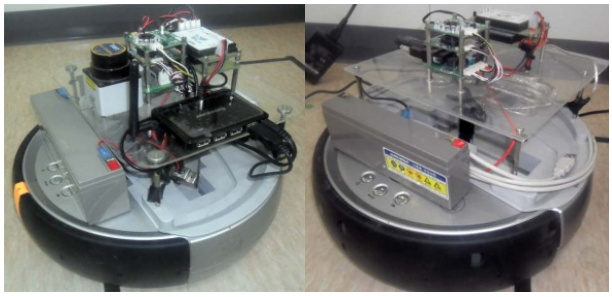


Fig. 13. Robots used in the experiment.

The main control code is written in the "VI" of LabView, and the experiment is processed online through wireless communication.

4.3. Experimental complexity

The complexity of the experiment stems primarily from the errors and jumps of the localization sensor. Some data jump and become lost when the sensor shifts from one landmark to another because the localization sensor uses landmarks to obtain its position data. The data corresponding to the current landmark jump suddenly to different values belonging to a new landmark. When the robot speed increases, the jump error tends to increase. Thus, the maximum speed of the robot is limited to 22.6 cm/sec in the experiment.

4.4. Experimental results

The experimental results are obtained for different experimental setups and different numbers of robots (Figs. 14 and 15). The traced path of the cleaning robots is denoted as a thick curve in the figures. Fig. 14 shows the same suggested path in the simulation in Fig. 9. The path of the experiment is plotted with the acquired data from the localization sensor of the robots; thus, this path may vary from the trajectory of the real robot because of sensor errors and jumps. The first experiment is accomplished in the environment setup shown in Fig. 11; two cleaning robots are used to compare the experimental results with the simulation results. The first robot, Robot1, starts at a global coordinate (250, 250) and covers Cell1 to Cell4. The order in which the cells are to be cleaned by the first robot is described by $Path_1$ = {Cell1, Cell2, Cell3, Cell4}. The second robot, Robot2, begins at a coordinate (4750, 4750) and covers Cell5 to Cell7. The order of the path of the second robot is presented as $Path_2$ = {Cell7, Cell6, Cell5}. The second experiment is conducted in the environment setup illus-

Fig. 14. Experimental result for the first layout.

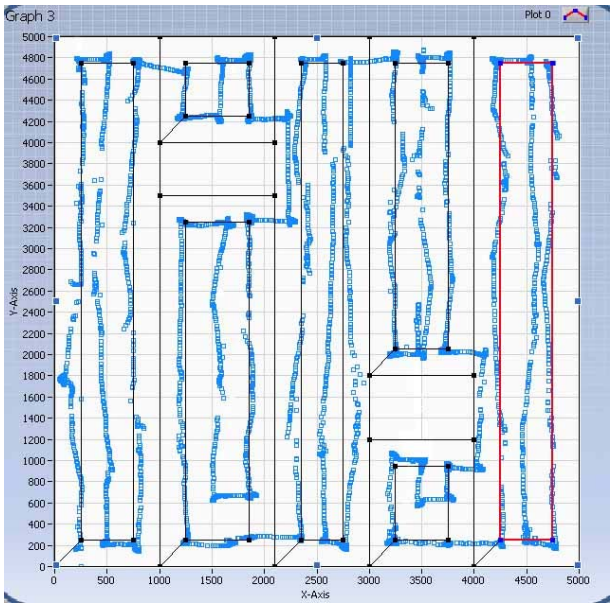


Fig. 15. Experimental result for the second layout.

Table 2. Comparison among other studies.

	# of turns	Total time (sec)
Results of the proposed algorithm	Multi robots	
Experimental	37	330
	Single robot	
Simulation	41	602
Experiment using the proposed algorithm	44	610
Experiment using the algorithm of [1]	52	756.6
Experimental results of [10]	35(67.6)	568.7(1097.9)
Experimental results of [11]	54(48)	556(498.4)

trated in Fig. 8. The traced path of a single cleaning robot for the second experiment setup is illustrated in Fig. 15, where the order in which the cells are to be cleaned by the robot is described by $Path_1$ = {Cell1, Cell2, Cell3, Cell4, Cell5, Cell6, Cell7}. Table 2 shows the total number of turns as well as the time spent to complete coverage for both the experiment and simulation. The experimental result of the proposed algorithm is compared with the experimental result of the algorithm used in each of [1,10], and [11]. The sizes of the workspaces in [10] and [11] are 3.7 m × 3.5 m and 4 m × 7 m, respectively. These dimensions vary considerably that direct comparison of the workspaces provides no relevant result. The numbers in the parentheses of Table 2 are approximately normalized values with respect to the experimental condition. The approximately normalized number of turns is obtained by dividing each result by the ratio of the area. The total time cannot be normalized because the maximum robot velocities in the previous studies are different and the speed of the robot changes during the experiment. However, a higher total time can be estimated given a larger normalized number of turns.. Thus, the approximate normalized total time is determined using the same calculations as that of the approximate normalized number of turns.

5. CONCLUSION

This study suggests a path planning method that plans the coverage path within a minimum time. The results show that the proposed path planning involves a smaller number of turns than do other existing methods when the cleaning robot path plan is in an environment with obstacles. A mobile robot requires a considerably long time to change its direction because the robot must first stop or decelerate. Path planning for an entire workspace is completed after planning the covering order of cells and deciding on the templates within each cell. These results indicate that the proposed CCPP is beneficial for practice as demonstrated by both simulation and experiment.

REFERENCES

[1] H. Choset, "Coverage of known spaces: the boustrophedon cellular decomposition," *Autonomous Robots*, vol. 9, no. 3, pp. 247-253, December 2000.

[2] S. C. Wong and B. A. MacDonald, "A topological coverage algorithm for mobile robots," *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 2, pp. 1685-1690, October 2003.

[3] S. C. Wong and B. A. MacDonald, "Complete coverage by mobile robots using slice decomposition based on natural land marks," *Proc. of the 8th Pacific Rim International Conference on Artificial intelligence*, vol. 3157, pp. 683-692, August 2004.

[4] J. W. Kang, S. J. Kim, and M. J. Chung, "Path planning for complete and efficient coverage operation of mobile robots," *Proc. of the IEEE International Conference on Mechatronics and Automation*, pp. 2126-2131, August 2007.

[5] Y. Mao, L. Dou, J. Chen, H. Fang, H. Zhang, and H. Cao, "Combined complete coverage path planning for autonomous mobile robot in indoor environment," *Proc. of the 7th Asian Control Conference*, pp. 1468-1473, August 2009.

[6] S. H. Nam, I. S. Shin, J. J. Kim, and S. G. Lee, "Complete coverage path planning for multi-robots employing flow networks," *Proc. of the International Conference on Control, Automation and Systems*, pp. 2117-2120, October 2008.

[7] Y. H. Choi, T. K. Lee, S. H. Baek, and S. Y. Oh, "Online complete coverage path planning for mobile robot based on linked spiral paths using constrained inverse distance transform," *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5788-5793, October 2009.

[8] S. Surve, N. M. Singh, and B. K. Lande, "CPPA: A Fast Coverage Algorithm," *International Conference on Computational Intelligence and Multimedia Applications*, vol. 2, pp. 151-158, 2007.

[9] S. Hert and B. Richards, "Multiple robot motion planning = parallel processing + Geometry," *Sensor Based Intelligent Robots, LNCS 2238*, vol. 2238, pp. 195-215, October 2002.

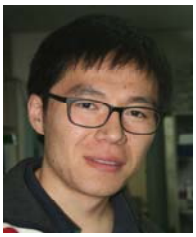
[10] G. H. Kim, H. Kim, B. S. Kim, and S. G. Lee,

"Path planning for an intelligent robot using flow networks," *Journal of Korea Institute of Intelligent Systems*, pp. 255-262, August 2011.

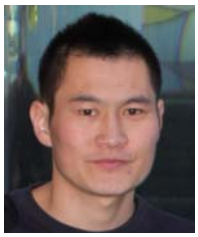
[11] S. H. Nam and S. B. Moon, "Minimal turning path planning for cleaning robots employing flow networks," *Journal of Control, Automation, and Systems Engineering*, vol. 11, no. 9, 2005.

[12] B. Park, J. Choi, and W. K. Chung, "Sampling-based retraction method for improving the quality of mobile robot path planning," *International Journal of Control, Automation, and Systems*, vol. 10, no. 5, pp. 982-991, October 2012.

[13] K. Yang, "Anytime synchronized-biased-greedy rapidly-exploring random tree path planning in two dimensional complex environments," *International Journal of Control, Automation, and Systems*, vol. 9, no. 4, pp. 750-758, August 2011.



Adiyabaatar Janchiv received his B.E. degree in Mechanical Engineering from Mongolian University of Science and Technology, Ulaanbaatar, Mongolia in 2009 and his M.S degree in Mechanical Engineering from Kyung Hee University, Yongin, Korea in 2012. He is currently working as sales engineer at Wagner Asia Equipment LLC in Mongolia. His research interest mobile robotics, automatic control and power supply network protection.



Dugarjav Batsaikhan received his B.E. degree in Mechanical Engineering from Mongolian University of Science and Technology, Ulaanbaatar, Mongolia in 2009. He is currently a Ph.D. student at the Department of Mechanical Engineering, Kyung Hee University. His research interests include mobile robotics and mechatronics.

ByungSoo Kim received his B.E. degree in Mechanical Design Engineering from Seoul National University of Science and Technology, Seoul, Korea, and his M.S. and Ph.D. degrees in Mechanical Engineering from Kyung Hee University, Yongin, Korea, in 1991, 1993, and 2011, respectively. He is currently as a post-doctoral researcher at the Department of Mechanical Engineering, Kyung Hee University. His research interests include humanoid robot, mechatronics, intelligent control, and mobile robot.

Won Gu Lee received his B.S. and M.S. degrees from the Department of Control & Mechanical Engineering at Pusan National University, and his Ph.D. degree from the School of Mechanical & Aerospace Engineering at Seoul National University, in 1996, 2000, and 2007, respectively. He got trained from the Harvard-MIT Health Sciences and Technology (HST) and Wyss Institute at Harvard University, U.S.A. as postdoctoral researcher. He is currently an Assistant Professor at the Department of Mechanical Engineering, Kyung Hee University. He is also an Adjunct Professor of Xi'an Jiaotong University, China. His research interests are in the area of developing a neural mechatronic system in association with human-on-a-chip in optofluidics.

Soon-Geul Lee received his B.E. degree in Mechanical Engineering from Seoul National University, Seoul, Korea, an M.S. degree in Production Engineering from KAIST, Seoul, Korea, and a Ph.D. degree in Mechanical Engineering from the University of Michigan, in 1983, 1985 and 1993 respectively. Since 1996, he has been with the Department of Mechanical Engineering of Kyung Hee University, Yongin, Korea, where he is currently a Professor. His research interests include robotics and automation, mechatronics, intelligent control, and biomechanics. He likewise served as the Director of the Korean Society of Precision Engineering (KSPE) (2005-2007) and for Institute of Control, Robotics, and Systems (ICROS) (2006).