The 2009 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 11-15, 2009 St. Louis, USA

Online Complete Coverage Path Planning for Mobile Robots Based on Linked Spiral Paths Using Constrained Inverse Distance Transform

Young-Ho Choi, Tae-Kyeong Lee, Sang-Hoon Baek, and Se-Young Oh, *Senior Member, IEEE*

Abstract— **This paper presents a sensor-based online coverage path planning algorithm guaranteeing a complete coverage of unstructured planar environments by a mobile robot. The proposed complete coverage algorithm abstracts the environment as a union of robot-sized cells and then uses a spiral filling rule. It can be largely classified as an approximate cellular decomposition approach as defined by Choset. In this paper, we first propose a special map coordinate assignment scheme based on active wall-finding using the history of sensor readings, which can drastically reduce the number of turns on the generated coverage path. Next, we develop an efficient path planner to link the simple spiral paths using the constrained inverse distance transform that we introduced the first time. This planner selects the next target cell which is at the minimal path length away from the current cell among the remaining non-contiguous uncovered cells while at the same time, finding the path to this target to save both the memory and time which are important concern in embedded robotics. Experiments on both simulated and real cleaning robots demonstrate the practical efficiency and robustness of the proposed algorithm.**

I. Introduction

MOST of the existing robot path planning research has addressed finding a path from a start location to a goal location. It usually requires minimization of the path length, journey time, or energy consumption. However, some applications such as floor cleaning [1], lawn mowing [2], mine hunting [3], harvesting [4], etc. require other kinds of path planning which are capable of finding an optimal path which ensures the complete coverage of an environment. This problem is called coverage path planning whose goal is to find paths maximizing coverage as well as minimizing some cost functions such as the time-to-completion.

Some of the early approaches to coverage path planning have been based on the behavior based paradigm including both heuristic and randomized components [5], [6]. These are amenable to low cost real world applications because they do not require expensive sensors nor do they consume valuable computational resources for mapping and localization. As such, they not only cannot guarantee a complete coverage, but their performance greatly varies as a function of the particular algorithm and the surrounding environment. Recently, complete coverage path planning algorithms have been proposed generating a path that completely covers the free space, most of which either implicitly or explicitly adopt cellular decomposition to achieve a complete coverage. A cellular decomposition breaks down the target region into cells such that coverage in each cell is "simple".

Choset [7] categorizes the cellular decomposition into three types: approximate, semi-approximate, and exact. Approximate decomposition uses a fine-grid based representation of the free space. Here, the cells are all of the same size and shape so that the union of the cells only approximates the target region [8]. The semi-approximate cellular decomposition relies on a partial discretization of the space where the cells are fixed in width but their tops and bottoms can have any shape [9]. The exact cellular decomposition divides the target environment into a set of non-intersecting cells, whose union fills the target environment and each cell is typically covered using simple back-and-forth motions [10].

Our approach to complete coverage falls under the approximate cellular decomposition category utilizing the spiral filling rules. Similar approaches include the Backtracking Spiral Algorithm (BSA) [11] and the Spiral-STC [12] algorithm. The BSA is based on the execution of spiral filling paths and assures completeness using a backtracking mechanism to link a collection of the simple spiral paths. It is robust to the robot's initial orientation due to the use of a spiral filling path instead of a zig-zag path which has been more popular in the previous algorithms. BSA can be implemented using a small set of rules, which allows it to work with low sensorial and computational requirements. However, it requires a great deal of memory space for the stack that stores the backtracking points as well as additional computational resources to manage this stack structure. This algorithm, especially when implemented in embedded systems, may cause a serious concern in a large target environment. On the other hand, Spiral-STC incrementally subdivides the planar work area into disjoint tool-size cells, while following a spanning tree path of the resulting grid. It generates a coverage path which has a comparatively less overlap than the other approximate cellular decomposition methods. However, this algorithm

Y. Choi was with the Electronic and Electrical Engineering Department, Pohang University of Science and Technology (POSTECH), Pohang, Kyungbuk, Korea, He is now with the Pohang Institute of Intelligent Robotics (PIRO), Pohang, Kyungbuk, Korea (corresponding author to provide phone: +82-54-279-0445; fax: +82-54-279-0429; e-mail: rockboy@postech.ac.kr) .

T. Lee, S. Baek and S. Oh are with the Electronic and Electrical Engineering Department, Pohang University of Science and Technology (POSTECH), Pohang, Kyungbuk, Korea (e-mail: devilee@postech.ac.kr; tkeb100@postech.ac.kr; syoh@postech.ac.kr).

cannot handle partially occupied cells. All these algorithms based on approximate cellular decomposition suffer from the common problem that the algorithm efficiency in view of time-to-completion is significantly affected by the initial orientation of the robot because the number of turns can drastically increase according to the robot's initial orientation relative to the wall.

This paper has two main contributions. First, we propose a simple map coordinate assignment scheme based on the history of sensor readings to improve the time-to-completion by reducing the number of turns on the generated path. Next, we introduce the constrained inverse distance transform (CIDT) to link the pieces of the simple spiral paths for complete coverage, which can find the non-contiguous uncovered cell with the shortest path length away from the robot's current footprint as well as simultaneously find the path leading to it.

The rest of this paper is organized as follows. Section II briefly reviews the problems of existing cellular decomposition methods and spiral filling rules. Section III then describes our approach to complete coverage to solve the problems. Section IV presents the experimental results with comparison to a conventional approach.

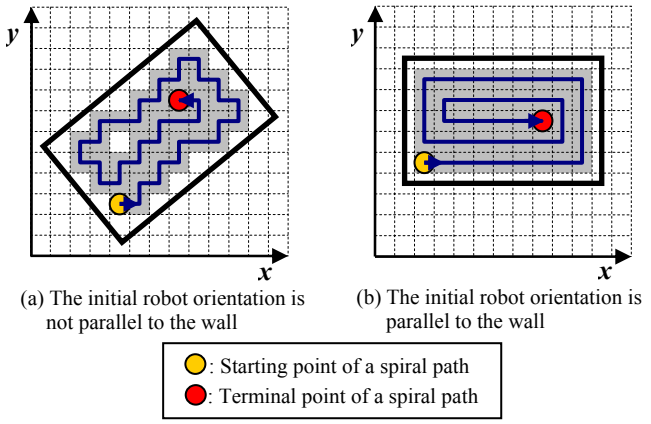
II. Coverage Problem Definition

As mentioned in Section I, efficiency of the generated path from the algorithm based on approximate cellular decomposition is affected by the specific scheme of the map coordinate assignment. This is primarily due to the constrained mobility where the robot is only allowed to move to one of the 4 neighborhood cells. This problem, illustrated in Fig. 1, leads to the fact that the number of turns on the generated path is significantly affected by the map coordinate assignment when using the spiral filling rules. The following reactive rules are typically used for the execution of the simple spiral path based on the right hand rule:

IF (*No obstacle in the right*) Turn right
ELSE IF (*No obstacle in the front*) Move forward
ELSE IF (*No obstacle in the left*) Turn left
OTHERWISE Terminate spiral point detection;

Here, the previously covered cells as well as the occupied cells are considered as obstacles.

Furthermore, approximate cellular decomposition suffers from the problem of having to fill partially occupied cells due to the use of coarse resolution in surface representation as depicted in Fig. 2. These partially occupied cells cannot be covered with any simple spiral filling rule because they only apply to the completely free cells. To handle this, Gonzalez [11] suggested the wall-following procedure as soon as an obstacle is found. This is a very reasonable solution because all the partially occupied cells are located nearby the obstacles including the walls. We also follow this strategy in our coverage algorithm.



(a) The initial robot orientation is not parallel to the wall

(b) The initial robot orientation is parallel to the wall

Fig. 1. Variation of the number of required turns according to different map coordinate assignments.

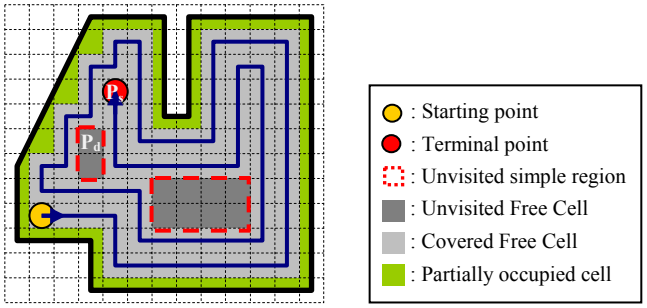


Fig. 2. Coverage map after finishing a simple spiral path.

In most of the complete coverage algorithms, they use the concept of driving the robot towards unvisited areas when there is no unvisited neighboring area around the robot, regardless of the filling approach. How to link simple regions is the key to a complete coverage algorithm using simple spiral filling rules as described in Fig. 2. It exemplifies that two simple regions are left unvisited after execution of the first simple spiral path. To link simple spiral paths, Gonzalez [11] proposed a backtracking mechanism (BTM) which consists of three steps: (A) Backtracking points (BP) to which two or more alternative possible paths exist are stored into a BP stack, while carrying out the spiral filling path (B) A valid BP which has at least one free unvisited neighboring cell is retrieved from the BP stack when the execution of a simple spiral-filling path is finished and one of these unvisited cells is selected as an entry point to an unfilled associated region (C) The shortest return path is generated from a terminal spiral point P_s to a selected entry point P_d using a region growing procedure starting in P_d and ending when P_s is reached. To our knowledge, this mechanism appears to be the best way to link simple regions in the spiral kind of coverage path planning algorithm and its feasibility was demonstrated in several simulated environments. However, it still has some drawbacks as follows:

1) It must check the criterion for a BP every time before applying spiral filling rules.
2) A point can be pushed into the BP stack, multiple times. Consequently, there are redundancies in the BP stack so

that a BP can yield the entry point of more than one uncovered region.

3) It requires a supplementary routine to discard an invalid BP when the free cells nearby a BP have vanished in order to avoid useless robot motions.

In this paper, we propose a more efficient method to link simple regions, which reduces the memory usage as well as the computational cost needed to manage BPs with the BP stack. The proposed constraint inverse distance transform (CIDT) yields a unified solution which selects the next target cell which is at the minimal path length away from the current cell among the remaining non-contiguous uncovered cells while at the same time, finding the path to this target. The details of CIDT are described in the following section.

III. The Proposed Complete Coverage Algorithm – LSP-CIDT

The proposed online complete coverage algorithm is based on linking spiral paths through a constrained inverse distance transform (LSP-CIDT). It basically covers a simple region using a spiral filling path and then links these simple regions using a constrained inverse distance transform (CIDT). Here, a simple region denotes an area that can be filled by executing a single spiral path. To this end, the environment is incrementally modeled by a coarse-grain occupancy grid and each cell is set to one of the five types: unexplored, covered, partially covered, occupied, and candidate. Here, we define a candidate cell as the one which is covered by the range sensors of a robot but has not been visited by a robot on a map and then regard it as a cell of interest in the CIDT. The overall process of the proposed LSP-CIDT coverage algorithm is shown in Fig. 3. Initially, all the cells are set to an unexplored state and a wall following procedure is taken with an active map coordinate assignment process. After wall following is finished, the spiral filling rules described in Section II are applied. If an obstacle is detected during spiral filling, contour following is carried out to sweep around it. When a simple spiral path is finished, CIDT is executed to select the next target cell which is at the minimal path length away from the current cell among the remaining non-contiguous uncovered cells while at the same time, finding the path to this target. If a candidate cell is found, the robot moves to that cell through the generated path and then starts a new spiral path in the new region; otherwise, the overall process terminates.

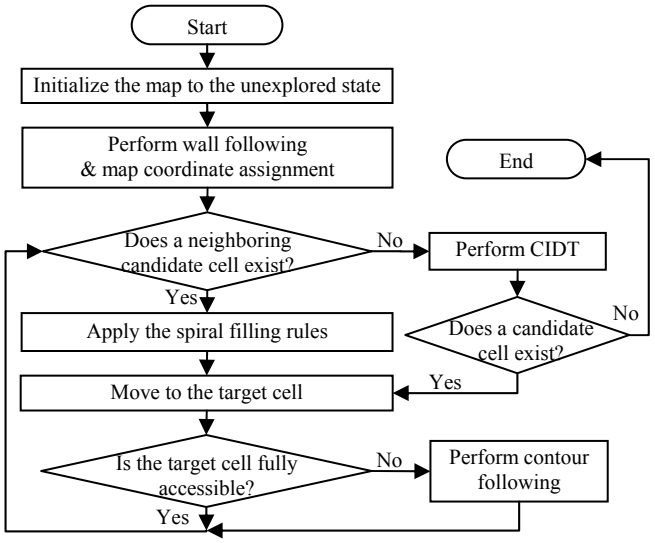


Fig. 3. Overall process of the proposed LSP-CIDT coverage algorithm.

A. Active Map Coordinate Assignment

The proposed LSP-CIDT algorithm is primarily focused on indoor environments (e.g. home, office, etc.) which have a rectangular shape consisting of four walls. In this case, we can drastically reduce the number of required turns by aligning the *x*-axis of the map coordinate frame to one of the walls of the target environment as shown in Fig. 1. To this end, we develop a special map coordinate assignment scheme based upon sensor readings as explained in Fig. 4. Here, we assume that the robot starts from close to the wall and then approach and follow this wall to start our coverage algorithm. This procedure is called "wall approach-then-follow". When the robot's orientation is stably aligned to the wall as shown in the Fig. 4 (c), we then use this robot's heading as the *x*-coordinate of the map reference frame throughout our algorithm. To check stable alignment in "wall approach-then-follow", we monitor an empirical parameter ***L***, defined as a traveling length over which the robot moves without significant change of its heading. Experimental results show that this is a simple but an effective way to reduce the number of turns in rectilinear environments.

B. Wall and Contour Following

As mentioned in Section II, we adopt the wall and contour following procedures to cover the partially occupied cells nearby the walls and obstacles. The wall following procedure is always invoked at the initial stage of LSP-CIDT whereas the contour following procedure is called whenever the robot meets an obstacle during the execution of a spiral filling path. Each procedure terminates when the robot returns to the cell from which the procedure started from. These procedures are conducted in a reactive manner while spiral filling is performed within the cell. Consequently, while spiral filling guarantees a complete coverage of a cell since the robot moves via the centers of the cells, the wall and contour following procedures do not guarantee a complete coverage as shown in Fig. 5. It shows that some cells are not completely covered even when the center of robot goes through the inside of these cells when the robot follows the wall using a right hand rule. To handle this problem, these cells are first marked as partially covered and thus considered as cells of future interest in applying both the spiral filling rules and CIDT. These cells are finally revisited for complete coverage. In this way, we can readily eliminate the possibility of any uncovered regions at the end.

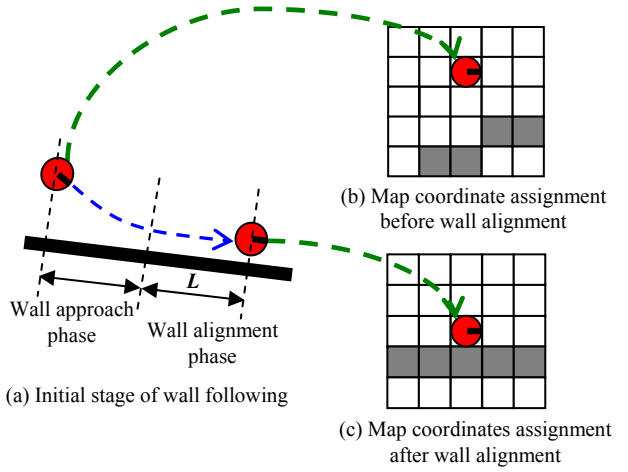


Fig. 4. An active map coordinate assignment scheme.

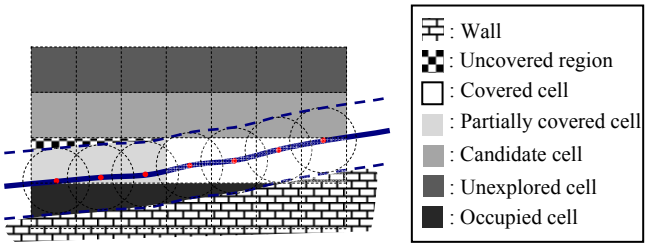


Fig. 5. Emergence of partially covered cells during wall and contour following.

C. Constrained Inverse Distance Transform

Constrained Inverse Distance Transform (CIDT) is a key component of our coverage algorithm to link simple spiral paths. In conventional path planning approaches, Distance Transform was used to find the shortest path from the starting point to the goal point, given these points. BSA also follows this strategy to find the shortest path from the current cell to the entry cell of an unvisited region, found by searching the recently visited valid backtracking point which has at least one free unvisited neighboring cell from the BP stack. However, in this subsection, we develop a unified approach to find the candidate cell which is at the shortest path away from the current cell as well as finding the shortest path between them when only the current cell position as a starting point is known.

In Distance Transform, region growing starts from the goal point and ends at the starting point after which the shortest path from the starting to the goal point is found using gradient descent. However, we newly introduce the Inverse Distance Transform (IDT) where region growing starts from a known current cell (starting point) and terminates when meeting an unknown candidate cell (goal point) at the shortest path away from the current cell. Inverse Distance Transform generates the inverse path from the goal to the starting point, hence the name. To implement IDT, we adopted the distance transform algorithm proposed in [13]. In the original algorithm, all the non-occupied cells among the 8 neighborhood of the cell being evaluated are first put into a first-in-first-out linked list buffer (FIFO), and then the next cell to be evaluated, in terms

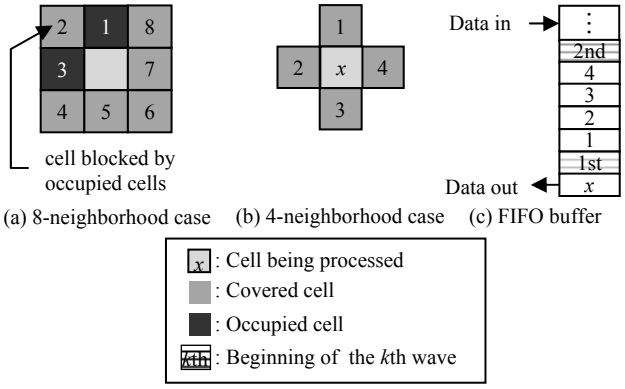


Fig. 6. Evaluation of the distance propagation cost in Inverse Distance Transform.

of the distance propagation cost, is retrieved from the FIFO. In contrast, we only consider a 4 neighborhood of the cell being processed as shown in Figs. 6(b) and (c) because an incorrect path, which the robot cannot follow due to a blocked cell, may still be generated even though the distance wave can propagate as shown in Fig 6 (a). We also store the running count of the distance wave when the transition to the next distance wave occurs as shown in Fig. 6 (c). When this wave count is retrieved from the stack, it also indicates the start of cost evaluation of a further distance wave, when only the increased wave count is stored in the stack without evaluation of the distance propagation cost.

Intuitively, it is easy to realize that the first found candidate cell, while carrying out IDT, is the goal point which lies on the shortest path from the current cell. However, this is not always guaranteed because distance wave propagates with an arbitrary shape in general cluttered environments. Even in the simple environment, this may happen because the distance wave propagates with the shape of a square. To illustrate this point, we show a propagation of distance waves in the simple environment with no obstacles in Fig. 7. As shown, we cannot assure that the first found candidate cell is the one which is at the shortest path until we evaluate the distance propagation cost of all the candidate cells existing in the circle including the first found candidate cell. To ensure this, we add a terminal condition to constrain the propagation of distance waves in IDT by comparing the minimum distance cost of the currently evaluated distance wave and the distance cost of the nearest candidate found so far. The terminating condition of IDT is checked every time when a transition to the next distance wave occurs, at which point we can obtain the minimum distance cost of the current distance wave. The condition is described as follows:

IF $D^c_{\min} \geq D_{\min}$ Terminate IDT ,
OTHERWISE Continue IDT

where $D_{\min}$ and $D^c_{\min}$ stand for the distance propagation cost of the nearest candidate found so far and the minimum distance propagation cost of the currently evaluated distance wave, respectively. Here, $D_{\min}$ is updated every time a new candidate cell with a lower distance propagation cost is found.

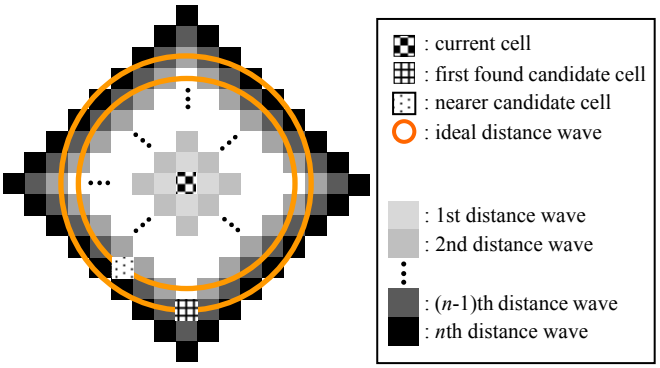
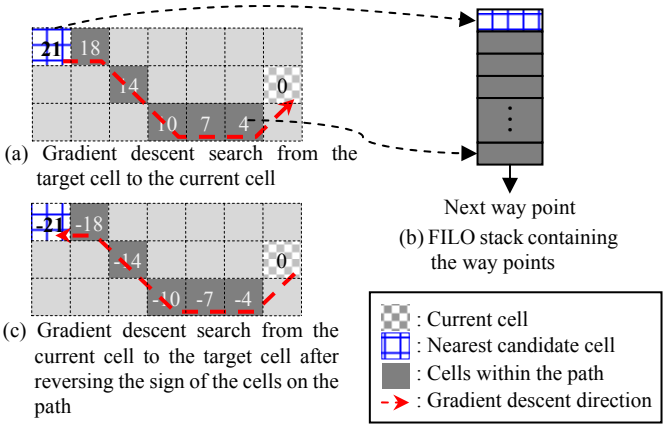


Fig. 7. Property of distance wavefront propagation in Inverse Distance Transform.



(a) Gradient descent search from the target cell to the current cell

(b) FILO stack containing the way points

(c) Gradient descent search from the current cell to the target cell after reversing the sign of the cells on the path

Fig. 8. The proposed method to find the path to the next target cell using the distance map obtained through CIDT.

This terminating condition is very reasonable because the minimum distance propagation cost is a monotone increasing function of the wave count, i.e., there cannot exist any cell which has a lower cost than $D_{\min}$ in the further wave than the wave meeting the terminal condition. Using these rules, we can efficiently constrain the search boundary of IDT to save memory and time. Therefore, we call the overall process of searching the nearest candidate cell Constrained Inverse Distance Transform (CIDT).

D. Path Generation based on CIDT

Up to this point, we found the nearest candidate point which is a non-contiguous uncovered cell from the robot. However, as mentioned previously, the path obtained from CIDT is an inverse path from the goal to the starting point. Therefore, we should rebuild the forward path between the current cell and the target cell from this inverse path. We can do this in two ways. We can store each way point, a 2D index of the cell found by a gradient decent search starting from the target cell, into a first -in-last-out (FILO) linked list buffer as shown in Figs. 8(a)(b). The robot can move to the target cell using the way points retrieved in reverse order from this buffer. Or we can simply put a minus sign to the distance value of each way point found in the previous sentences as shown in Fig. 8(c). Using this changed distance map, the robot can move to the target cell using an additional gradient descent search starting from the current cell. The first method requires less computation than the second because it needs to apply gradient descent search once while the second needs to apply it twice. However, the first method requires an additional memory to save the way points.

Fig. 9. Sensor configuration of our robot platform.

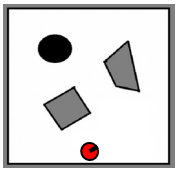


Fig. 10. Simulation environment with the initial robot pose marked.

IV. Experimental Results

We have conducted experiments in both the simulated and real environments to verify the workings of our approach. Fig. 9 shows the sensor configuration of the robot platform used as an experimental test bed. It is a differential-drive robot equipped with seven infrared sensors having a maximum range of 30 *cm*. In our experiments, the maximal velocity of the robot and the size of a cell are set to 25 *cm/s* and 30 *cm*, respectively.

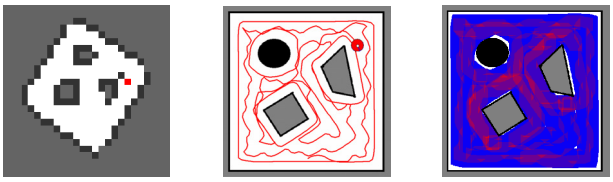
A. Simulation Result

To verify the influence of the proposed active map coordinate assignment scheme on efficiency of the generated path, we carried out two simulations with the same initial robot pose in the same environment depicted in Fig. 10.

In the first simulation, we align the *x*-axis of the map frame with the initial robot heading without active map coordinate assignment. As seen in Fig. 11, the generated path contains many "zig-zag" sections. However, the path with the proposed coordinate assignment has much fewer "zig-zag" sections as shown in Fig. 12. In Figs. 11 and 12, the color of the cells in the coverage map becomes redder as the robot sweeps the same cell repeatedly. Table 1 shows that the number of turns is drastically reduced using active map coordinate assignment, which leads to a shorter time-to-completion. However, the coverage itself does not differ significantly because the overall paths of both simulation results are almost the same.

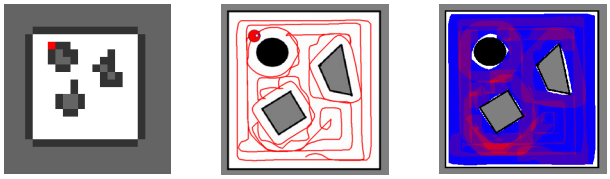
TABLE I
COMPARISON OF THE NUMBER OF TURNS IN THE COVERAGE PATH

Case	*Number of turns*	*Time-to-completion (minute)*	*Coverage (%)*
Without active map coordinate assignment	*88*	*8.80*	*91.65*
With active map coordinate assignment	*31*	*7.34*	*91.73*



(a) Resulting Map (b) Overall trajectory (c) Coverage map

Fig. 11. Simulation results without our active map coordinates assignment.



(a) Resulting Map (b) Overall trajectory (c) Coverage map

Fig. 12. Simulation results with our active map coordinate assignment.

B. *Results on Real Cleaning Robots*

To show the robustness of our algorithm, we conducted experiments in four different environments, all with the size of 4.2 *m* by 5 *m*, shown in the first column of Fig. 13. When applying the LSP-CIDT algorithm to a real environment, the accuracy of the robot pose estimate is a key factor affecting the overall performance of the algorithm. Therefore, we built our own odometry system combining the encoder and gyro sensor readings to limit the robot's position error within a tolerable error boundary defined as the half size of a cell. In the cleaning robot, the entire algorithm has been embedded into a DSP F2812 on board while a ceiling camera is used to extract the ground truth position of the robot. Fig. 13 shows the results. Here, the coverage is measured based on the ground truth of the robot position calculated from ceiling image. The averages of the time-to-completion and coverage are 8.86 minutes and 94.99 percents, respectively. These results support that our algorithm is also applicable to various cluttered indoor environments.

V. Conclusion

This paper presents a robust online complete coverage algorithm based on approximate cellular decomposition and spiral filling paths. This algorithm drastically reduces the number of turns on the generated path by devising a sensor-based map coordinate assignment scheme. In addition, we introduce a new transform called constrained inverse distance transform (CIDT) to link the simple spiral paths for complete coverage. CIDT enables a unified solution which can select the next target cell which is at a minimal path length away from the current cell among the remaining non-contiguous uncovered cells as well as find the path to this target, simultaneously. Finally, we demonstrate that our algorithm is applicable to embedded robotic systems with a robust performance in various complex environments for robots. However, since our algorithm has been tested in a single room so far, in the future, it also needs to be tested against multiple rooms.

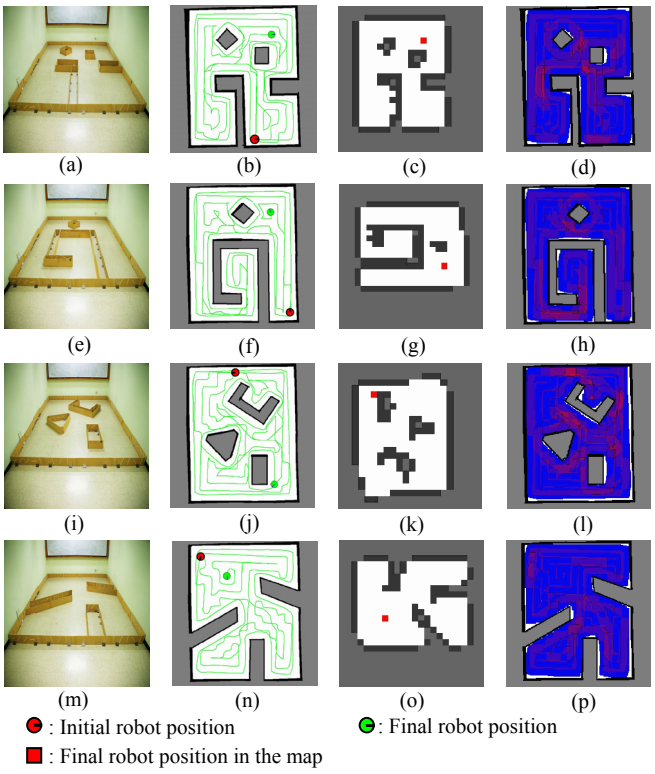


Fig. 13. Real-world experiment. The first column shows the four different environment configurations, the second, the overall trajectories, the third, the resulting maps, and the last shows the coverage.

References

[1] J. Colegrave and A. Branch, "A case study of autonomous household vacuum cleaner," in *AIAA/NASA CIRFFSS*, 1994.

[2] Y. Y. Huang, Z. L. Cao and E. L. Hall, "Region filling operations for mobile robot using computer graphics," in *Proc. IEEE Int. Conf. Robotics and Automation*, vol. 3, 1986, pp. 1607-1614.

[3] S. Land and H. Choset, "Coverage path planning for landmine location," in *3rd Int. Symp. Technology and the Mine Problem*, Monterey, CA, 1998.

[4] M. Ollis and A. Stentz, "First results in vision-based crop line tracking," in *Proc. IEEE Int. Conf. Robotics and Automation*, vol. 1, 1996, pp. 951-956.

[5] R. A. Brooks, "A robust layered control system for a mobile robot," *IEEE J. Robotics and Automation*, vol. 2, 1986, pp. 14-23.

[6] D. Gage, "Randomized search strategies with imperfect sensors," in *Proc. SPIE Mobile Robots* VIII, Boston, MA, 1993, pp. 270-279.

[7] H. Choset, "Coverage for robotics - A survey of recent results," in *Annals of Mathematics and Artificial Intelligence*, 2001, pp. 113-126.

[8] S. V. Spires and S.Y. Goldsmith, "Exhaustive geographic search with mobile robots along space-filling curves," in *Proc. of the 1st Int. Workshop in Collective Robotics*, CRW'98, Paris, France, July 1998, *Lecture Notes in Computer Science*, vol. 1456, 1998, pp. 1-12.

[9] S. Hert, S. Tiwari and V. Lumelsky, "A terrain-covering algorithm for an AUV," *Autonomous Robots,* vol. 3, 1996, pp. 91-119.

[10] H. Choset, E. Acar, A. Rizzi and J. Luntz, "Exact cellular decompositions in terms of critical points of Morse functions," in *Proc. IEEE Int. Conf. Robotics and Automation*, vol. 3, 2000, pp. 2270-2277.

[11] E. González, O. Álvarez, Y. Díaz, C. Parra and C. Bustacara, "BSA: A Complete Coverage Algorithm," in *Proc. IEEE Int. Conf. Robotics and Automation*, 2005, pp. 2040-2044.

[12] Y. Gabriely and E. Rimon, "Spiral-STC: An On-Line Coverage Algorithm of Grid Environments by a Mobile Robot," in *Proc. IEEE Int. Conf. Robotics and Automation*, vol. 1, 2002, pp. 954-960.

[13] Y. T. Chin, H. Wang, L. P. Tay, H. Wang and W. Y. C. Soh, "Vision Guided AGV Using Distance Transform," in *Proc. 32nd Int. Symp. Robotics*, April 2001, pp. 19-21.