Proceedings of the 18th World Congress
The International Federation of Automatic Control
Milano (Italy) August 28 - September 2, 2011

Complete Coverage D* Algorithm for Path Planning of a Floor-Cleaning Mobile Robot

Marija Dakulović * **Sanja Horvatić** * **Ivan Petrović** *

* *Department of Control and Computer Engineering, Faculty of Electrical Engineering and Computing, University of Zagreb, Croatia, (e-mail: {marija.dakulovic, sanja.horvatic, ivan.petrovic}@fer.hr).*

Abstract: Inspired by the path transform (PT) algorithm of Zelinsky *et al.* the novel algorithm of complete coverage called complete coverage D* (CCD*) algorithm is developed, based on the D* search of the two-dimensional occupancy grid map of the environment. Unlike the original PT algorithm the CCD* algorithm takes the robot's dimension into account, with emphasis on safety of motion and reductions of path length and search time. Additionally, the proposed CCD* algorithm has ability to produce new complete coverage path as the environment changes. The algorithms were tested on a Pioneer 3DX mobile robot equipped with a laser range finder.

Keywords: mobile robot, path planning, obstacle avoidance, robotic coverage.

1. INTRODUCTION

The motion planning problem of a floor-cleaning mobile robot comes down to planning a path from a start position to a final position in an environment so that the robot cleans all the reachable positions while following the planned path. The task of finding the path that visits all nodes in the graph is called the complete coverage path planning, LaValle (2006). Finding an optimal path that visits every node in the graph exactly once is NP-hard problem known as the traveling salesman problem. Therefore, approximate or even heuristic solutions are used for the complete coverage path planning task. Gabriely and Rimon (2003) consider partially or completely unknown environments in the complete coverage path planning problem (i.e. exploration task). On the other hand, De Carvalho et al. (1997) rely on the *a priori* knowledge of a 2D map of the environment and cope with unknown obstacles detected by range sensor. A common approach is decomposing the environment into subregions (cell-decomposition, Latombe (1991)), selecting a sequence of those subregions, and then generating a path that covers each subregion in turn. Most methods like ones proposed by Huang (2001) or Choset (2000) assume convex polygonal environments and perform exact cell decomposition, which can be very time consuming in changing environments. De Carvalho et al. (1997) proposed method based on templates – the robot combines a number of "basic motion macros". Zelinsky et al. (1993) proposed method based on the approximative cell decomposition (i.e. grid maps), which is less time consuming, but suppose that the mobile robot has the dimensions of exactly one cell within the grid map. With such a crude representation the details of the environment are lost and a sequence of free cells leading through a narrow passage could also be lost although the mobile robot could pass through it.

⋆ This research has been supported by the Ministry of Science, Education and Sports of the Republic of Croatia under grant No. 036 – 0363078 – 3018.

This paper presents a new complete coverage path planning algorithm, called the complete coverage D* (CCD*) algorithm, based on the high resolution occupancy grid map. The CCD* algorithm uses the path transform (PT) methodology introduced by Zelinsky et al. (1993). The novelty of the CCD* algorithm is using D* search in order to have fast replanning capability in changing environments and taking the robot's dimension into account, with an emphasis on safety of motion and reduction of path length and search time.

The rest of the paper is organized as follows. Section 2 briefly reviews the D* and PT algorithms. Section 3 presents the proposed CCD* algorithm. In Section 4 test results are given.

2. RELATED ALGORITHMS

Complete coverage path planning algorithm proposed in this paper is based on the D* algorithm by Stentz (1994), and the PT algorithm by Zelinsky et al. (1993), and therefore these two algorithms are briefly surveyed in this section. Since both algorithms as well as our proposed algorithm use occupancy grid map of the environment, we firstly present used map.

2.1 Used occupancy grid map and robot representation

An occupancy grid map is created by approximate cell decomposition of the environment. The whole environment is divided into squared cells of equal size e_{cell}, which are abstractly represented as the set of M elements $\mathcal{M} = \{1, \ldots, M\}$ with corresponding Cartesian coordinates of cell centers $c_i \in \mathbb{R}^2$, $i \in \mathcal{M}$. Each cell contains occupancy information of the part of the environment that it covers.

We use weighted occupancy grid map introduced in our previous work Seder and Petrović (2007). A weighted occupancy function $o(i) \in \{1, 2, \ldots, M_c + 1, \infty\}$, $i \in \mathcal{M}$ is used for representing the set of all obstacles in the

environment noted as $\mathcal{O} = \{i \in \mathcal{M} \mid o(i) = \infty\}$ and unoccupied environment is represented by the set of unoccupied cells noted as $\mathcal{N} = \mathcal{M} \setminus \mathcal{O}$. Occupancy function is defined as follows:

$$o(i) = \begin{cases} \max\{1, (M_c + 2 - \min_{j \in \mathcal{O}} \|c_i - c_j\|_\infty)\} & \text{if } i \notin \mathcal{O} \\ \infty \text{ if } i \in \mathcal{O} \end{cases} \quad (1)$$

Described procedure generates the so-called safety cost mask around obstacles with smooth decrease of occupancy values from the obstacles towards the free space. The size of the safety cost mask is defined by the predefined integer number of cells M_c. The weighed occupancy grid map with $M_c = 4$ cells wide safety cost mask of the section of the experimental environment is shown in Fig. 1.

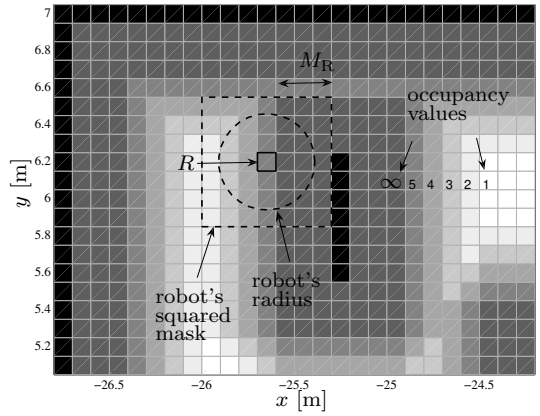

Fig. 1. The occupancy grid map with the safety cost mask ($M_c = 4$) and the robot squared mask ($M_R = 3$).

In this paper we assume that the cleaning tool mounted on the robot is a square of size that is integer number of cell sizes and wider than the robot itself. In that case the robot is represented by a squared mask in the grid map, within which the robot's real circular shape can be drawn. To allow the robot to be located within any unoccupied cell, all obstacles in the grid map are enlarged for the integer number of cells M_R, i.e. the robot is described by a squared mask of size $2 \cdot M_R + 1$ cells. Real shape of the robot and its squared mask in the occupancy grid map are depicted in Fig. 1. The robot's position is considered to be the cell R. The real obstacle placement is shown by black color.

Weighted undirected graph $\mathcal{G}(\mathcal{N}, \mathcal{E}, \mathcal{W})$ is created from the occupancy grid map in such a way that all unoccupied cells $\mathcal{N}$ represent the set of nodes in the graph. Further, we say that two nodes $i, j \in \mathcal{N}$ in the graph are *neighbors* if $\|c_i - c_j\|_\infty = e_{cell}$. The set of edges is defined as $\mathcal{E} = \{\{i, j\} \mid i, j \in \mathcal{N}, i \text{ and } j \text{ are neighbors}\}$. The set of edge weights $\mathcal{W} = \{w_{i,j} \mid i, j \in \mathcal{N}, i \text{ and } j \text{ are neighbors}\}$ is defined as the cost of transition between neighbors as

$$w_{i,j} := \|c_i - c_j\| \cdot \max\{o(i), o(j)\}. \quad (2)$$

A path in the graph $\mathcal{G}(\mathcal{N}, \mathcal{E}, \mathcal{W})$ is defined as a succession of nodes. For example a path $\mathcal{P}$ of length L is defined by $\mathcal{P} = (n_1, n_2, \ldots, n_L)$ if $\{n_i, n_{i+1}\} \in \mathcal{E}, i = 1, \ldots, L-1$. The cost of the path $\mathcal{P}$ is defined as the sum of weights of edges along the path, i.e., $c(\mathcal{P}) := \sum_{i=1}^{L-1} w_{n_i, n_{i+1}}$. The *optimal cost* of the path from the start node S to the goal node G, noted by $g^*(S, G)$, is the minimal cost among all possible paths from S to G in the graph with implicit assumption that $g^*(S, G) = \infty$ if no path exists. The *optimal path* from the start node to the goal node is the path (or more than one path) that has optimal cost $g^*(S, G)$. In the following, the shorthand $g(n) \equiv g^*(n, G)$ is used.

2.2 The D* algorithm

Stentz (1994) proposed a well known graph search algorithm capable of fast replanning in changing environments. It is also known as dynamic version of the A* algorithm without the heuristic function, Hart et al. (1968).

For every searched node n, the D* algorithm computes the cost value $g(n)$ of the optimal path from the node n to the goal node G and the value of the key function $k(n)$ for the replanning process, which stores the old values $g(n)$ before changes of weights in the graph $\mathcal{G}(\mathcal{N}, \mathcal{E}, \mathcal{W})$ happened. The algorithm stores the *backpointers* for every searched node n, which point to the parent node with the smallest cost g. A backpointer is noted by the function $b(\cdot)$, where $b(n) = m$ means that the node n has the smallest cost because it follows the node m. The backpointers ensure that optimal path from any searched node n to the goal node G can be extracted according to the function $b(\cdot)$.

The execution of the D* algorithm can be divided into *initial planning* and *replanning* phases. Initial planning is performed if the robot is standstill at the start position and replanning is performed if the robot detects nodes with changed occupancy values during its motion. The number of expanded nodes is kept to a minimum and consequently the time of execution (for more details see Stentz (1994)).

2.3 The PT algorithm

The PT algorithm is based on the wavefront algorithm which produces the cost values from every node to the goal node G. The cost values are calculated by propagation of a wave front of values through all free nodes starting from the specified goal node G. For example, the goal node G gets value 1, its neighbor nodes get value 2, the neighbors of its neighbor nodes that do not have already assigned values get value 3, etc., see Fig. 2. Notice that diagonal and straight transitions are treated equally. The wavefront algorithm can be equalized to the D* algorithm invoked only initially without the replanning capability. The complete coverage path is determined by following the largest values of costs starting from the specified start node S, see Fig. 2. Notice that the final point of the complete coverage path is the goal node G from which the wave front is propagated. The PT algorithm uses modification of the

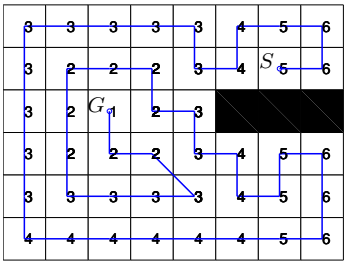

Fig. 2. The complete coverage path from S to G by the PT algorithm.

wavefront algorithm. The wave front which is propagated is a weighted sum of the distance from the goal together with a measure of the discomfort of moving too close to obstacles. This can be compared to searching the paths in weighted occupancy grid map with the safety cost mask.

3. THE COMPLETE COVERAGE D* ALGORITHM

The proposed algorithm extends the PT algorithm in order to take into account all the free cells which the robot's mask covers in the grid map with its dimensions. Additional improvement is made by using D* instead of the wavefront algorithm to ensure fast replanning capability in changing environments. Hereafter we first describe how the CCD* algorithm plans the initial complete coverage path when the robot is standstill at the start position, and then how it replans the path when the robot detects obstacles on its journey to the final position of the complete coverage path. The D* algorithm is invoked not from the chosen goal position but from the robot's position (the goal node is replaced by the start node S in initial planning or the node R in the case of replanning). By this the final position of the complete coverage path is different at each algorithm execution during the robot's moving. The pseudocode of the complete coverage path calculation is given by Alg. 1, and the pseudocode which invokes the CCD* algorithm during the robot's motion is given by Alg. 2.

3.1 Initial complete coverage path planning

Initially, the D* algorithm does the exhaustive search of the graph $\mathcal{G}(\mathcal{N}, \mathcal{E}, \mathcal{W})$ from the start node S. For every node n in the graph $\mathcal{G}(\mathcal{N}, \mathcal{E}, \mathcal{W})$, D* calculates cost $g(n)$ to the start node as well as $k(n)$ and the backpointer $b(n)$ needed for path replanning. In order to avoid visiting already visited nodes the binary function $\textbf{visited}(n) = \{0, 1\}$ is used and its values are stored for each node. In order to avoid overlapping of the part of the complete coverage paths the binary function $\textbf{overlapped}(n) = \{0, 1\}$ is used and its values are stored for each node. Initially, all nodes are set to be non-visited and non-overlapped. The current node C is set to the start node S. The path of the complete coverage is produced by following the increase of costs g around the current node C starting from the start node S. The surrounding candidate nodes $\mathcal{N}_C$ are those non-visited and non-overlapped nodes that are reachable ($g(n) < \infty$) and distanced from the current node C for the robot square size ($2 \cdot M_{\mathrm{R}} + 1$ cells) in four straight directions through the grid. The next node M in the complete coverage path is the one with the smallest cost g among surrounding candidate nodes. The connection between two nodes in the complete coverage path is stored as auxiliary path $\mathcal{P}_{aux}$ composed of neighbor nodes in the grid map. All nodes distanced from the auxiliary path $\mathcal{P}_{aux}$ within the robot mask M_{R} are set to visited, and within the two robot masks ($2 \cdot M_{\mathrm{R}}$) are set to overlapped. If no surrounding candidate node exists then extra D* search is executed from the current node C. This D* search is noted as D*' since it must not change values of g, k and b calculated by the first D* search. It is used as initial planner and values g' and b' are used instead of the g, k and b values. The searching by D*' stops if (a) the first non-visited node (node M) is found or (b) the visited node M is found that is one cell near the enlarged obstacle and exists non-visited node n distanced from the node M less than or equal to the robot mask M_{R}. The shortest path from C to M is computed by following the backpointers b' from M to C. The node M is set to be the new current node C for the next algorithm iteration. The algorithm stops when there are no non-visited reachable nodes or non-reachable nodes distanced for the robot mask M_{R} from the reachable nodes.

Algorithm 1 Coverage(S)

```
1:  C ← S   //Set the current node to S
2:  P_aux ← C
3:  P ← ∅
4:  while 1
5:      ∀n ∈ P_aux, m ∈ N, ‖c_n − c_m‖ < M_R · e_cell
            visited(m) = 1
6:      ∀n ∈ P_aux, m ∈ N, ‖c_n − c_m‖ < 2M_R · e_cell
            overlapped(m) = 1
7:      N_C ← {n ∈ N | ‖c_n − c_C‖_∞ = (2M_R + 1) · e_cell
            and overlapped(n) = 0 and g(n) < ∞}
8:      if N_C ≠ ∅
9:          find M ∈ N_C with minimal g
10:     else
11:         D*'(C) and stop at visited(M) = 0
                or ‖c_M − c_o‖_∞ = e_cell, o ∈ O and ∃n,
                visited(n) = 0, ‖c_M − c_n‖ < M_R · e_cell
12:         if no such node M exists
13:             return P
14:         end
15:     end
16:     P_aux ← P_aux(C, M)
17:     C ← M   //Set the new current node
18:     P ← P ∪ P_aux
19: end
```

3.2 Complete coverage path replanning

If nodes in vicinity of the moving robot change their occupancies (function **changed-nodes**(R) in Alg. 2), the path replanning process is initiated in order to find the new complete coverage path from the robot's current position (node R). The CCD* algorithm replans the path by execution of the D* to compute the new cost values g and then by execution of the complete coverage path calculation given by Alg. 1 (function **Coverage**(R) in Alg. 2). The complete coverage path calculation depends on the cost values g calculated by the D* search and on the nodes already visited by the robot while following the complete coverage path before the change in the environment. To track which cells are visited by the robot while following the complete coverage path, functions $\textbf{visited}_R(n) = \{0, 1\}$ and $\textbf{overlapped}_R(n) = \{0, 1\}$ are used. Before each execution of the complete coverage path calculation values of functions **visited**(n) and **overlapped**(n) are rewritten by the new ones $\textbf{visited}_R(n)$ and $\textbf{overlapped}_R(n)$, respectively. Since D* stores information about previous calculations the minimal number of nodes are examined. The complete coverage path $\mathcal{P}$ is followed by the robot until the final position is reached or is calculated again if a new change in the environment is detected. The function **path-following** is based on the Dynamic window algorithm described in our previous work Seder et al. (2005),

Algorithm 2 move-robot-CCD*(S)

$\forall n \in \mathcal{N}$, $k(n) \leftarrow \infty$, $g(n) \leftarrow \infty$, $b(n) \leftarrow n$,
  visited$(n) = 0$, **overlapped**$(n) = 0$,
  visited$_R(n) = 0$, **overlapped**$_R(n) = 0$
$R \leftarrow S$ *//Initialization*
while 1
  if $R = S$ or **changed-nodes**(R)
    D*(S) *//(Re)compute* $g(n)$, $\forall n \in \mathcal{N}$
    $\mathcal{P} \leftarrow$**Coverage**(R)
  end
  $R \leftarrow$**path-following**$(\mathcal{P})$
  for $\forall n \in \mathcal{N}$
    if $\|c_n - c_R\| < M_\mathrm{R} \cdot e_\mathrm{cell}$
      visited$_R(n) = 1$
    if $\|c_n - c_R\| < 2M_\mathrm{R} \cdot e_\mathrm{cell}$
      overlapped$_R(n) = 1$
    visited$(n) =$ **visited**$_R(n)$
    overlapped$(n) =$ **overlapped**$_R(n)$
  end
end

which takes into account robot's kinematic and dynamic constraints.

3.3 An illustrative example

Hereafter we illustrate on a simple example how the CCD* algorithm plans the complete coverage path in weighted occupancy grid maps. Initially, the D* algorithm does the exhaustive search from the node S, which is shown in Fig. 3. Cost values g are calculated for every reachable node and shown by corresponding colors. White spaces around the obstacles (black) are due to obstacle enlargement to account for the robot's dimensions ($g(n) = \infty$). In this white space the robot can not be located as a point (i.e. by the center of its squared mask), but can reach some of this space by the border of its mask.

The path of the complete coverage is produced by following the increase of costs g around the current node C starting from the start node S. The current node C is set to the start node S and all nodes distanced from the node C

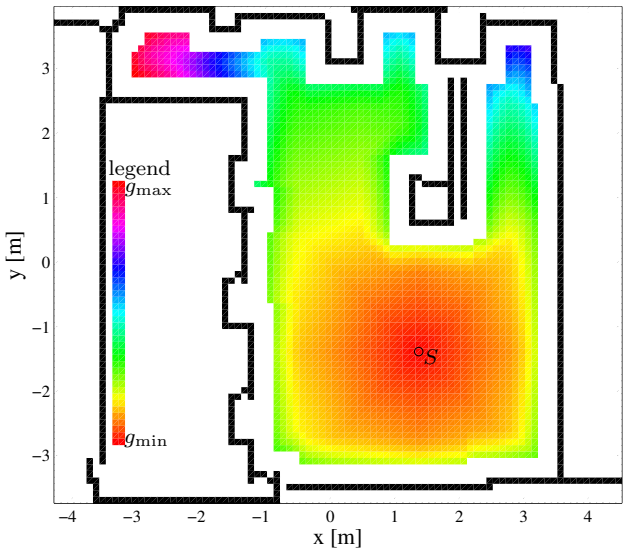

Fig. 3. Initial searching by D*: increasing g-values starting from the minimal value at the start node S ($g(S) = 0$).

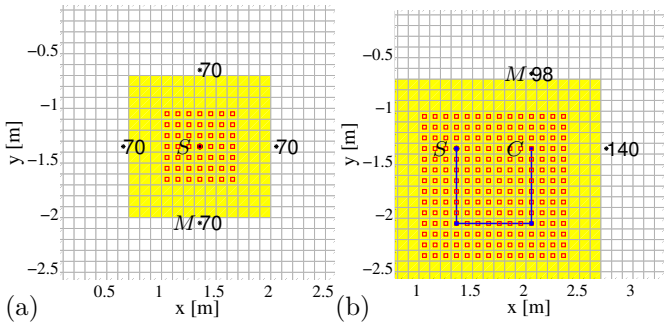

Fig. 4. Surrounding candidate nodes in (a) the first and (b) the fourth step of the complete coverage execution.

within the robot mask M_R are set to visited, and within the two robot masks ($2 \cdot M_\mathrm{R}$) are set to overlapped. The first step of the algorithm execution is shown if Fig. 4.(a). Visited nodes are noted by small squares inside of the grid cell, and overlapped nodes are noted by color filled grid cells. There are four surrounding candidate nodes noted by asterisk and with corresponding g values, i.e. four non-visited and non-overlapped nodes distanced from the current node C for the robot square size ($2 \cdot M_\mathrm{R} + 1$ cells) in four straight directions with cost $g(n) < \infty$. The next node M in the complete coverage path is the one with the smallest cost g among surrounding candidate nodes. Since all surrounding candidate nodes have the same cost value ($g = 70$, using $e_\mathrm{cell} = 10$, $\sqrt{2}e_\mathrm{cell} \approx 14$), then the node M is chosen arbitrarily. The node M is set to be the new current node C for the next (second) algorithm iteration and the same procedure is performed again. The fourth step of the algorithm execution is shown if Fig. 4.(b), where the complete coverage path has already four points and an area of four robot's masks has been visited. There are only two surrounding candidate nodes since other two possible candidate nodes are already overlapped and visited. The node M is the node with the smallest g value ($g = 98$). This procedure is repeated until there are no surrounding candidate nodes.

By choosing the next node in the complete coverage path only among the surrounding candidate nodes, cells that are near real obstacles remain non-visited. If no surrounding candidate node exists around the current node C, the D*' search is performed to find the closest non-visited node. This step is shown in Fig. 5.(a). All possible candidate surrounding nodes noted by asterisk are overlapped or unreachable ($g = \infty$). By performing the D*' search from

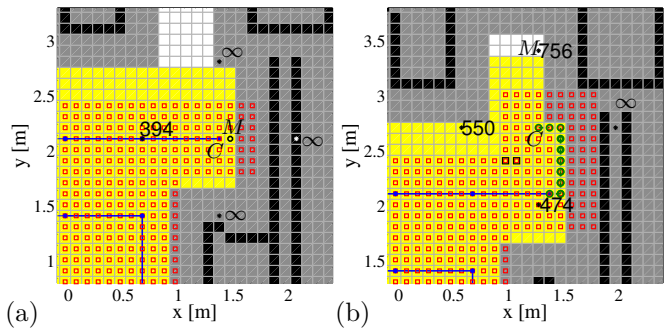

Fig. 5. (a) Determing the next node M by the D* search if no surrounding candidate node exists and (b) choosing again surrounding candidate node.

the current node C the new node M is found near the enlarged obstacle that is distanced less or equal to the M_{R} from the non-visited nodes near the border of the real obstacle. The node M in Fig. 5.(a) is one grid cell distanced from the node C. This procedure is repeated in further steps until at least one surrounding candidate node appears again (Fig. 5.(b)). The path between these two C nodes in Fig. 5.(a) and (b), noted by circle markers in Fig. 5.(b), "follows the wall" and the robot visits by its mask all nodes near the real obstacle edge. Notice that by this procedure some gird cells are redundantly visited (i.e. more than once), which are noted by the larger squares outside of the small squares near the node C if Fig. 5.(b).

Finally, when there are no more non-visited reachable nodes or non-reachable nodes that are distanced for the robot mask M_{R} from the reachable nodes the complete coverage planning is finished. The complete coverage path is shown in Fig. 6. The final node in the path G is the last node visited by the CCD* algorithm. Visited cells are noted by different colors according to the number of visits (cells are visited 1, 2 and 3 times). The points in the path are rare (distanced $2 \cdot M_{\mathrm{R}} + 1$ cells) in case of choosing the neighbor surrounding candidate node, and dense (distanced 1 cell) if D*' search is performed to find the closest non-visited node. White spaces could not be reached by the robot's mask and therefore have not been visited. The length of the complete coverage path is 647.5 cell sizes. From the total number of 3449 cells to be visited, 2300 cells are visited only once, 1110 cells are visited twice and 39 cells are visited three times.

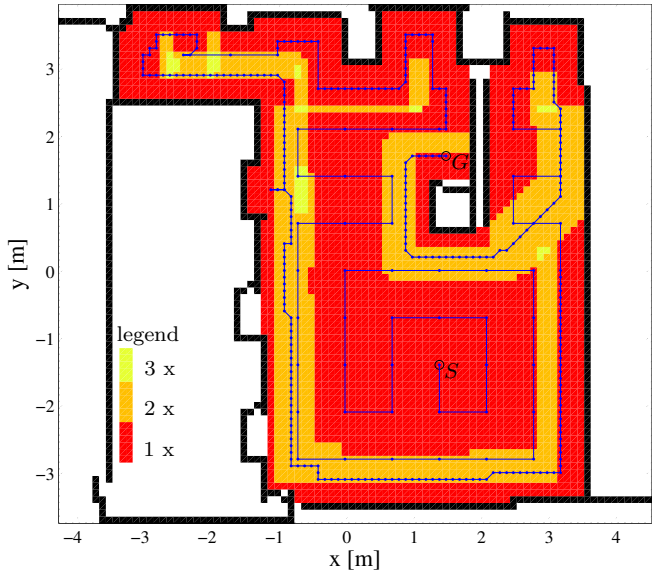

Fig. 6. Initial complete coverage path finishing at G and the numbers of visits of each cell.

4. TEST RESULTS

The proposed algorithm was implemented in *Player/Stage* (a free software tool, www.playerstage.sourceforge.net) and executed on a Pioneer 3DX mobile robot. The laser range finder mounted on the robot was used for environment perception.

In order to illustrate the functionality of the proposed algorithm, the results of a test are presented in Fig. 7, where the robot need to clean one room with completely known static obstacle configuration (walls, boxes, etc.) and with some unknown obstacles that are moving continuously through the environment (other robots, humans). The upper row represents robot's trajectory while following the complete coverage path and cells that have been actually visited by the robot. The lower row represents calculated complete coverage paths and cells that are visited by the complete coverage path. Different number of visits are noted by corresponding colors (1, 2, 3 times visited, etc.). A sequence of three simulation shots are arranged consecutive in columns (1), (2) and (3). In column (1), the robot follows the initial complete coverage path (top figure) and when reaches the position R detects unknown obstacles in front of it. The new complete coverage path is planned while avoiding already visited nodes by the robot and detected unknown obstacles. The new complete coverage path (shown in the lower figure) starts from the robot's position R and finishes at the calculated goal position G in the bottom right corner of the room. While the length of the initial complete coverage path is 64.75 m, the length of the recalculated complete coverage path is 72.86 m, which is longer because the path goes around the detected obstacles. In column (2), the robot follows the new complete coverage path from the lower figure of column (1). When it reaches the position R it detects changed obstacles positions. The unknown obstacles changes positions and make a barrier that split the environment into two parts – the one that the robot can reach and the other which can not. The new complete coverage path is planed avoiding already visited nodes starting from the robot's position R and finishing at the calculated goal position G near the middle of the left wall of the room. The length of the new complete coverage path is now 29.24 m since not all parts of the environment could be reached. In column (3), the robot follows the new complete coverage path from the lower figure of column (2). When it reaches the position R (almost at the goal position G in column (2)) it detects changed obstacles positions. The unknown obstacles moved into the visited area, and now all non-visited cells are included in the new path calculation. The length of the new complete coverage path is 30.84 m. The new complete coverage path goes from the position R to the calculated goal position G in the bottom left corner. The path goes back again through some parts of the environment since the robot did not ideally follow the planned path and some nodes that can be visited by the ideal path following remained non-visited. An interesting thing to notice is that at each replanning of the complete coverage path, nodes that were non-visited due to non-ideal path following are taken into account for the new complete coverage path calculation and they are visited by the new path.

The total robot's trajectory while following the planned and replanned complete coverage paths is shown in Fig. 8. The total length of the robot's trajectory is 73.52 m. The cells that have been actually visited by the robot are noted by corresponding colors according to the number of visits (from 1 to 5 times). From the total number of 3464 cells to be visited, 47% cells are visited only once, 39% cells are visited twice, 13% cells are visited three times and 1%

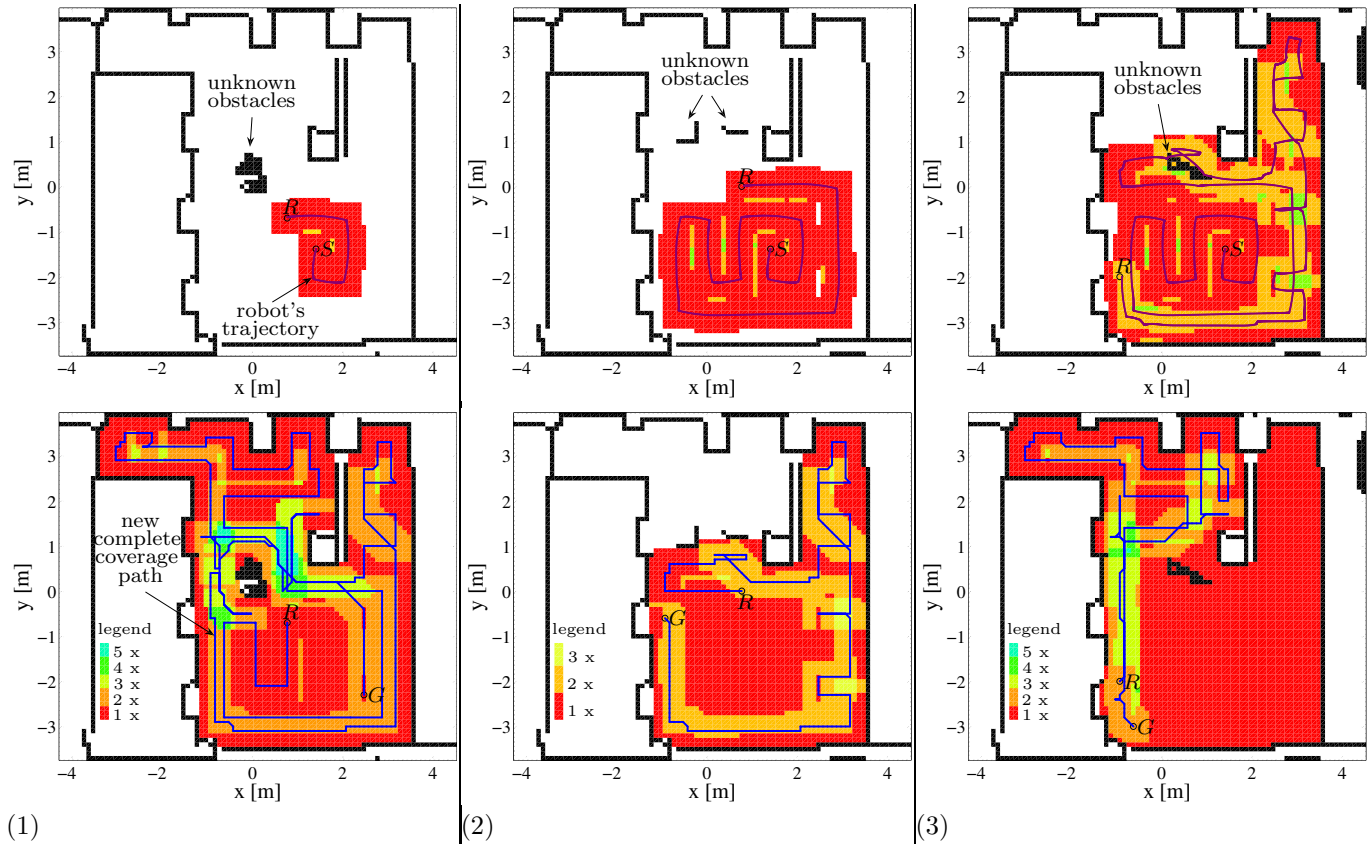

Fig. 7. The complete coverage of the room in the presence of the unknown moving obstacles.

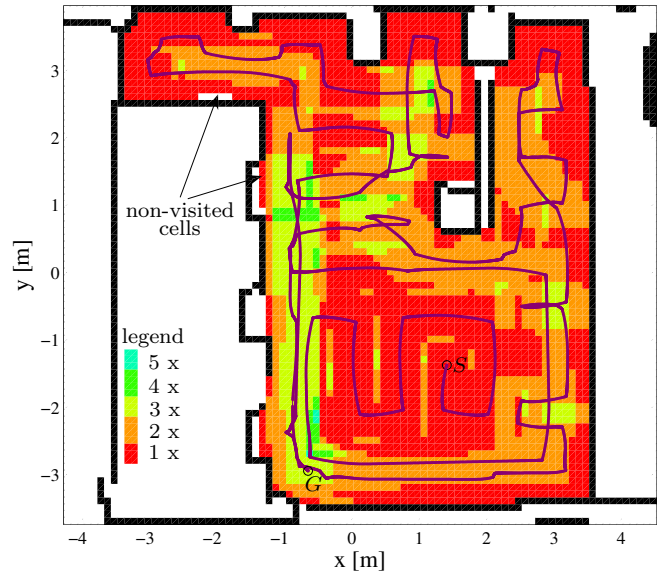

Fig. 8. The final complete coverage trajectory from S finished at G and numbers of visits of cells during the robot motion.

cells are visited four times. Less than 1% of cells have not been visited due to the non-ideal path following. However, by an extra recalculation of the complete coverage path, these cells will be visited.

REFERENCES

Choset, H. (2000). Coverage of known spaces: the boustrophedon cellular decomposition. *Autonomous Robots*, 9(3), 247–253.

De Carvalho, R., Vidal, H., Vieira, P., and Ribeiro, M. (1997). Complete coverage path planning and guidance for cleaning robots. In *IEEE ISIE'97*, 677–682.

Gabriely, Y. and Rimon, E. (2003). Competitive on-line coverage of grid environments by a mobile robot. *Computational Geometry*, 24(3), 197–224.

Hart, P., Nilsson, N., and Raphael, B. (1968). A formal basis for the heuristic determination of minimum cost paths. *IEEE Trans. Syst. Sci. Cybernetics*, 4(2), 100–107.

Huang, W. (2001). Optimal line-sweep-based decompositions for coverage algorithms. In *IEEE ICRA'01*, 27–32.

Latombe, J. (1991). *Robot Motion Planning*. Kluwer Academic Publishers, Dodrecht, Netherlands.

LaValle, S. (2006). *Planning algorithms*. Cambridge Univ. Pr.

Seder, M., Maček, K., and Petrović, I. (2005). An integrated approach to real-time mobile robot control in partially known indoor environments. In *IEEE IECON'05*, 1785–1790.

Seder, M. and Petrović, I. (2007). Dynamic window based approach to mobile robot motion control in the presence of moving obstacles. In *IEEE ICRA'07*, 1986–1991.

Stentz, A. (1994). Optimal and efficient path planning for partially-known environments. In *IEEE ICRA'94*, 3310–3317.

Zelinsky, A., Jarvis, R., Byrne, J., and Yuta, S. (1993). Planning paths of complete coverage of an unstructured environment by a mobile robot. In *Int. Conf. on Advanced Robotics (ICAR'93)*, 533–538.