SICE Annual Conference 2011
September 13-18, 2011, Waseda University, Tokyo, Japan

Path Planning Method Design for Mobile Robots

Chia-Jun Yu, Yi-Hong Chen, and Ching-Chang Wong

Department of Electrical Engineering, Tamkang University, New Taipei City, Taiwan
(Tel : +886-2-2621-5656 ext 2733; E-mail: wong@ee.tku.edu.tw)

Abstract: In this paper, a path planning method for autonomous mobile robots in a known indoor environment is proposed. A traditional A* algorithm modified by a weighted cost function is proposed. The factors of distance and safety are considered simultaneously in the cost function so that the path planning can let the mobile robot reach its goal safely and quickly. Some simulation results are presented to illustrate the proposed method has a good path planning for mobile robots. The proposed method has also been implemented and tested on a real mobile robot. The experiment results illustrate that the proposed method can let the autonomous mobile robot have a safe path-planning in a known environment.

Keywords: Path Planning, A* algorithm, autonomous robot, mobile robot.

1. INTRODUCTION

Path planning is one important task in the field of mobile robot design. In order to let an autonomous robot can navigate reliably in an indoor environment, the robot must know where it and plan an appropriate path by itself. In general, the basic problem in the path planning of mobile robots is how to navigate the robot from point A to point B without colliding into any walls or obstacles, in hopefully close to the shortest amount of the time. There are various path planning methods based on an environment map for mobile robots. For example, Dijkstra's algorithm [1] is a kind of most short-path search method, which it guarantees to find a shortest path. A* algorithm [2-4] is like Dijkstra's algorithm, and it can use a heuristic to guide itself. Other studies on A* can be found in [5-6]. Fu and Xue [5] proposed one kind of A* algorithm based on restricted searching area to compute the fastest path. Chabini and Lan [6] extends the A* methodology to shortest path problems in dynamic networks, in which arc travel times are time dependent. Furthermore, also has some different path planning method, M. Nakamiya et al. [7] proposed a path planning method for mobile sensor nodes using a cost map. Chan et al. [8] proposed a disk-based algorithm for finding shortest paths in a large network system. Hashemzadeh [9] proposed a new route finding method for car navigation systems meanwhile, have utilized learning power and high speed of neural networks. Jan et al. [10] proposed a higher geometry maze routing algorithm that has fewer constraints and can be applied to the previous aspects in the path-planning problem. However, there are some problems. The most important one is that the shortest path may let the mobile robot very close to obstacles, which is not safe. Therefore, a modified A* algorithm is proposed so that the safe path is selected to avoid mobile robot too close to obstacles.

The rest of this paper is organized as follows: In Section 2, a path planning method based on a modified A* algorithm is proposed In Section 3, some simulation results of the proposed path planning method and some compared results are presented. Furthermore, one experiment carried out on a real robot is presented. Finally, some conclusions are summarized in Section 4.

2. PATH PLANNING

2.1 A* algorithm

To apply the traditional A* algorithm to the path planning of mobile robots, the cost evaluation function is given by

$$f(\mathbf{n}) = g(\mathbf{n}) + h(\mathbf{n}) \tag{1}$$

where $\mathbf{n} = (x_n, y_n)$ is the current expanding node, and $f(\mathbf{n})$ is the estimation of the minimum cost among all paths from the start node S to the goal node G constrained to go through node $\mathbf{n}$. $g(\mathbf{n})$ is the actual movement cost of the current path from the start node $\boldsymbol{s}_0 = (x_0, y_0)$ to the node $\mathbf{n}$ denoted by

$$g(\mathbf{n}) = \begin{cases} \mathrm{dist}(\mathrm{prev}(\mathbf{n}),\ \mathbf{n}), & \textit{if}\ \mathrm{prev}(\mathbf{n}) = \boldsymbol{s}_0 \\ g(\mathrm{prev}(\mathbf{n})) + \mathrm{dist}(\mathrm{prev}(\mathbf{n}),\ \mathbf{n}), & \textit{else} \end{cases} \tag{2}$$

where $\mathrm{prev}(\mathbf{n}) = \boldsymbol{s}_k$, and $\boldsymbol{s}_k$ is the current node of the k step, and

$$\mathrm{dist}(\mathrm{prev}(\mathbf{n}),\ \mathbf{n}) = \mathrm{dist}(\boldsymbol{s}_k,\ \mathbf{n}) = \sqrt{(x_k - x_n)^2 + (y_k - y_n)^2} \tag{3}$$

However, in this paper, the grip map is used and the distance relation between two nodes is described in Fig.1, where d denotes the direct distance of two nodes. $h(\mathbf{n})$ is the heuristic estimation of the minimum cost of a path from the current expanding node $\mathbf{n}$ to the goal node $\mathrm{G} = (x_G, y_G)$ denoted by

$$h(\mathbf{n}) = d \times \sqrt{(x_n - x_G)^2 + (y_n - y_G)^2} \tag{4}$$

Based on A* algorithm, the optimum path $\boldsymbol{p}_0\text{-}\boldsymbol{p}_1\text{-}\boldsymbol{p}_2\text{-}\ldots\text{-}\boldsymbol{p}_{n-1}\text{-}\boldsymbol{p}_n\text{-}\ldots\text{-}\boldsymbol{p}_{N-1}\text{-}\boldsymbol{p}_N$ is determined by tracing back trough prev(G) from the goal node G firstly and back to the start node S finally, where $N \le K$, $\boldsymbol{p}_0$ =G and $\boldsymbol{p}_N$ =S.

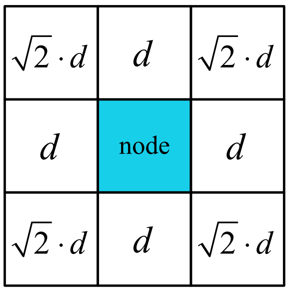


Fig.1 Distance relation between two nodes

2.2 Distance and safety cost

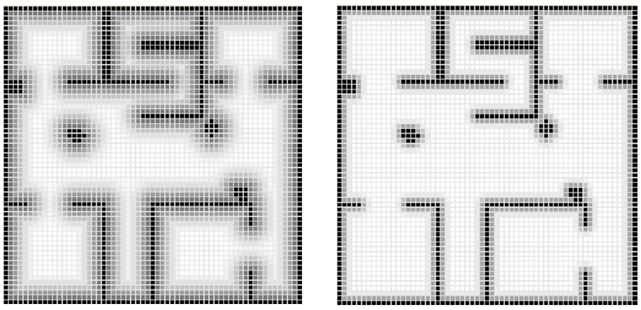
The traditional A* algorithm in the path planning is chiefly focusing on calculating the shortest path. However, the shortest path becomes less practical and helpful in the complicated environment with many obstacles. The concern of this paper is how to arrive at the destination in the most safe and fastest way. Therefore, $g(\mathbf{n})$ (the actual cost of the current path from the start node $\boldsymbol{s}_0$ to the current expanding node $\mathbf{n}$) is modified by

$$g_s(\mathbf{n}) = \mathrm{g}(\mathbf{n}) + \mathrm{safe}(\mathbf{n}) \tag{5}$$

where

$$\mathrm{safe}(\mathbf{n}) = \begin{cases} \alpha \cdot \exp(-\beta \cdot (\dfrac{D_{\mathrm{wall}}(\mathbf{n})}{D_{\max}})^2), & \textit{if } D_{\mathrm{wall}}(\mathbf{n}) \le D_{\max} \\ 0, & \textit{else} \end{cases} \tag{6}$$

where α denotes the maximum value of safety cost, β denotes the rate of change, $D_{\max}$ is a maximum safe distance between the robot and obstacles, and $D_{\mathrm{wall}}(\mathbf{n})$ is the shortest distance between the robot and obstacles.



(a) $\alpha = 150$, $\beta = 2.5$, and $D_{\max} = 50$

(b) $\alpha = 150$, $\beta = 4.5$, and $D_{\max} = 30$

Fig. 2 Relationship map of different parameters in considering the security factor. The darker part indicates the robot in the position is very closer to the obstacles, and a higher cost is needed to go through in the safety requirement.

2.3 Optimized path

Finding a most safe and fastest path for mobile robots becomes more valuable for research in the path planning of a complicated environment with many obstacles. In general, the safe and shortest path can be find by the modified A* algorithm described by Eq. (1), but sometimes this path is not very smooth. There may be two or more different paths are determined under the same moving cost. For example in Fig. 3, we can find that the solid line path has three tuning points, but the dotted line path needs only one turning point. Less turning points are good for mobile robots. Thus Equation (2) is modified by

$$g_w(\mathbf{n}) = w \cdot g(\mathbf{n}) + \mathrm{safe}(\mathbf{n}) \tag{7}$$

where w is an inertia parameter. When the selected direction of the current expanding node is the same that of the parent node, $w = 1$. On the contrary, $w = 1.3$. The inertia parameter will let the path planning tend to select the original moving direction.

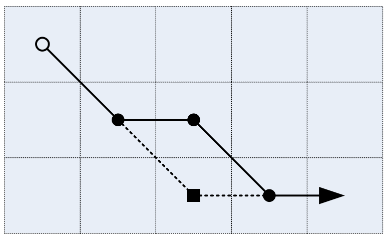


Fig. 3 Description of two different paths with the same path length.

The procedure of the modified A* algorithm to find an optimal path could be described as follows:

Step 1: Let $n = 1$, the start node S be $\mathrm{S} = \boldsymbol{s}_0 = (x_0, y_0)$, the goal node G be $\mathrm{G} = \boldsymbol{s}_G = (x_G, y_G)$, the set of obstacle nodes be $\mathrm{OBST} = \{\mathbf{o}_1, \mathbf{o}_2, ..., \mathbf{o}_m\}$, the close set be $\mathrm{CLOSE} = \{\boldsymbol{s}_0\}$, and the open set be $\mathrm{OPEN} = \{(x_0 - 1, y_0 + 1)$, $(x_0, y_0 + 1)$, $(x_0 + 1, y_0 + 1)$, $(x_0 - 1, y_0)$, $(x_0 + 1, y_0)$,$(x_0 - 1, y_0 - 1)$, $(x_0, y_0 - 1)$, $(x_0 + 1, y_0 - 1)\}$ in the initial.

Step 2: If $\mathrm{G} \in \mathrm{OPEN}$, then the goal node has been found and go to Step 11. Otherwise let $\mathrm{prev}(\mathbf{c}_j) = \boldsymbol{s}_0$ be its parent node of each node in the open set and determine the values of w, $g_w(\mathbf{c}_j)$, $h(\mathbf{c}_j)$, and $f(\mathbf{c}_j)$, where $\mathbf{c}_j \in \mathrm{OPEN}$.

Step 3: Let $\boldsymbol{s}_k = \mathbf{c}_{j^*}$ be the current node, where $\mathbf{c}_{j^*} = \arg \min\limits_{\mathbf{c}_j \in \mathrm{OPEN}} f(\mathbf{c}_j)$.

Step 4: Let the set of adjacent nodes of the current node $\boldsymbol{s}_k$ be $\mathrm{ADJ}(\boldsymbol{s}_k) = \{(x_k - 1, y_k + 1)$, $(x_k, y_k + 1)$, $(x_k + 1, y_k + 1)$, $(x_k - 1, y_k)$, $(x_k + 1, y_k)$, $(x_k - 1, y_k - 1)$, $(x_k, y_k - 1)$, $(x_k + 1, y_k - 1)\} = \{\mathbf{n}_1, \mathbf{n}_2, ..., \mathbf{n}_8\}$.

Step 5: If $\mathrm{G} \in \mathrm{ADJ}(\boldsymbol{s}_k)$, then the goal node has been found and go to Step 11. Otherwise let $q = 1$.

Step 6: If $\mathbf{n}_q \in \mathrm{CLOSE} \cup \mathrm{OBST}$, then go to Step 9.

Step 7: If $\mathbf{n}_q \in \mathrm{OPEN}$, and $g_w(\mathbf{n}_q) > \overline{g_w(\mathbf{n}_q)}$, $\overline{g_w(\mathbf{n}_q)}$

is new movement cost by using new current node s_k, then change the values and its parent node by $\text{prev}(\mathbf{n}_q)=s_k$, and $g_w(\mathbf{n}_q)=\overline{g_w(\mathbf{n}_q)}$. Go to Step 9.

Step 8: Let $\text{OPEN}=\text{OPEN}\cup\{\mathbf{n}_q\}$, $\text{prev}(\mathbf{n}_q)=s_k$ be the parent node of $\mathbf{n}_q$, and determine the values of w, $g_w(\mathbf{n}_q)$, $h(\mathbf{n}_q)$, and $f(\mathbf{n}_q)$.

Step 9: Let $q=q+1$. If $q\le 8$, then go to Step 6.

Step 10: Let $\text{CLOSE}=\text{CLOSE}\cup\{s_k\}$, $\text{OPEN}=\text{OPEN}-\{s_k\}$, $k=k+1$, and go to Step 3.

Step 11: Let $K=k$ and determine the optimum path from the goal node G firstly and back to the start node S finally by

```
$p_0 = \text{G}, n = 0$
while $p_n$ is S
    $p_{n+1} = \text{prev}(p_n)$
    $n = n+1$
end
$N = n$
```

Some results of an illustrated example of the proposed modified A* algorithm are shown in Figs. 4~8, where S=(2,4) is a start node, G=(6,2) is a goal node, and the explanation of notation and values in the block of node is described in Fig. 9. As shown in Fig.8, there are seven searching steps to find the optimal path S=(2,4)-(3,3)-(4,2)-(5,2)-(6,2)=G. They are described in detail as follows:

In the first step, s_0=(2,4), CLOSE={s_0=(2,4)}, OPEN={(1,5), (2,5), (3,5), (1,4), (3,4), (1,3), (2,3), (3,3)}, and $\text{G}\notin\text{OPEN}$, so the score and its parent node of each node in OPEN are determined and shown in Fig.4, where node (3,3) in OPEN has the lowest cost value.

In the second step, node (3,3) is considered as the current node (i.e. s_1=(3,3)). We find that ADJ((3,3))={(2,4), (3,4), (4,4), (2,3), (4,3), (2,2), (3,2), (4,2)}, where node (2,4) is in the close set, and nodes (3,4) and (2,3) are in the open set, and the values of w, $g_w(.)$, $h(.)$, $f(.)$ and its parent node of each node are determined and shown in Fig.5. Now, s_1=(3,3), CLOSE={ s_0=(2,4), s_1=(3,3)}, and OPEN={(1,5), (2,5), (3,5), (1,4), (3,4), (4,4), (1,3), (2,3), (3,3), (4,3), (2,2), (3,2), (4,2)}. We find that node (3,4) in OPEN has the lowest cost value.

In the third step, node (3,4) is considered as the current node (i.e. s_2=(3,4)). Similarly, we can find that ADJ((3,4))={(2,5), (3,5), (4,5), (2,4), (4,4), (2,3), (3,3), (4,3)}, nodes (2,4) and (3,3) are in the close set, and nodes (2,5), (3,5), (4,4), (2,3), and (4,3) are in the open set, and the values of w, $g_w(.)$, $h(.)$, $f(.)$ and its parent node of each node are shown in Fig.6. Note that the values of node (4,4) and its parent node is modified. Now, CLOSE={(2,4), (3,3), (3,4)}, and OPEN={(1,5), (2,5), (3,5), (4,5), (1,4), (4,4), (1,3), (2,3), (4,3), (2,2), (3,2), (4,2)}. Note that nodes (4,2) and (4,4) in OPEN have the same lowest cost value.

In the fourth step, node (4,2) is considered as the current node (i.e. s_3=(4,2)) because node (4,2) than node (4,4) was found earlier. Similarly, we can find that ADJ((4,2))={(3,3), (4,3), (5,3), (3,2), (5,2), (3,1), (4,1), (5,1)}, (3,3) is in the close set, and nodes (4,3) and (3,2) are in the open set, and the values of w, $g_w(.)$, $h(.)$, $f(.)$ and its parent node of each node are shown in Fig.7. Now, CLOSE={(2,4), (3,3), (3,4), (4,2)}, and OPEN={(1,5), (2,5), (3,5), (4,5), (1,4), (4,4), (1,3), (2,3), (4,3), (5,3), (2,2), (3,2), (5,2), (3,1), (4,1), (5,1) }.We find that node (5,2) in OPEN has the lowest cost value

Similarly in the fifth and sixth steps, nodes (4,4) and (4,3) are respectively considered as the current node (i.e. s_4=(4,4) and s_5=(4,3))). In the seventh step, node (5,2) is considered as the current node (i.e. s_6=(5,2))). We can find that node (6,2) in ADJ((5,2))={(4,3), (5,3), (6,3), (4,2), (6,2), (4,1), (5,1), (6,1)} is the goal node, so the optimal path shown in Fig. 8 is obtained by backing through their parent nodes.

The path determined by the traditional A* algorithm is shown in Fig. 10, we can find that the path determined by the proposed method is good for the mobile robot than the traditional A* algorithm.

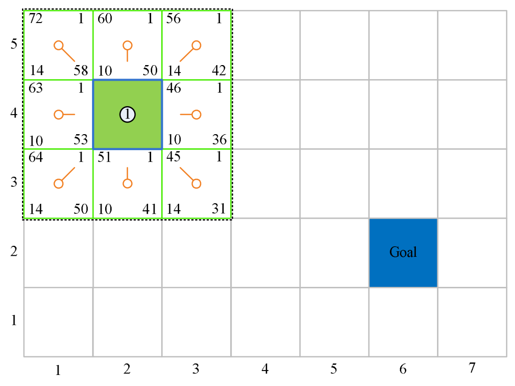


Fig. 4 Example explanation of the proposed modified A* algorithm, where (2,4) is a start node and (3,3) has a lowest score.

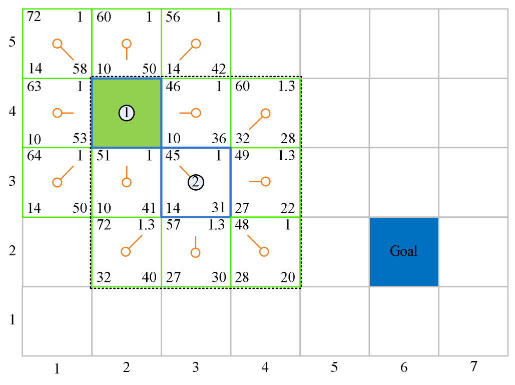


Fig. 5 Example explanation of the proposed modified A* algorithm, where (3,3) is a current node and only the w value of (4,2) is 1, the w value of the other expanding nodes are 1.3.

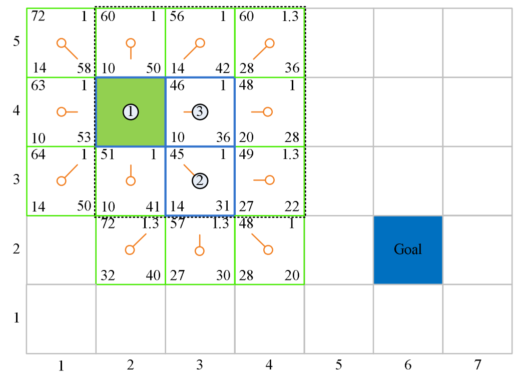


Fig. 6 Example explanation of the proposed modified A* algorithm, where (3,4) is a current node and the parent node of (4,4) can be modified to reduce the value of $g_w(\mathbf{n})$.

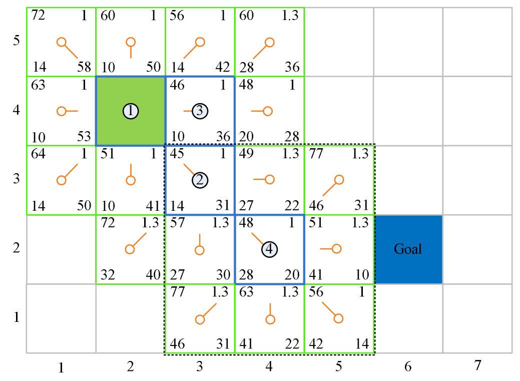


Fig. 7 Example explanation of the proposed modified A* algorithm, where (4,4) is a current node.

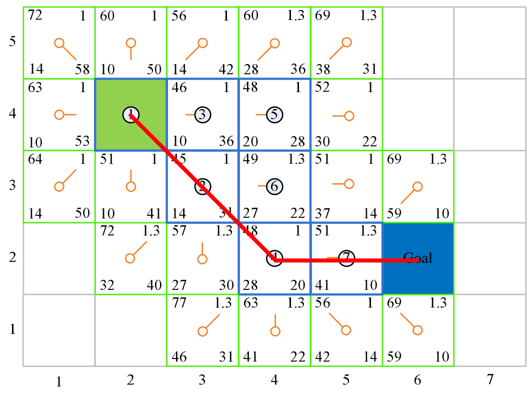


Fig. 8 Example explanation of the proposed modified A* algorithm: The optimal path is determined.

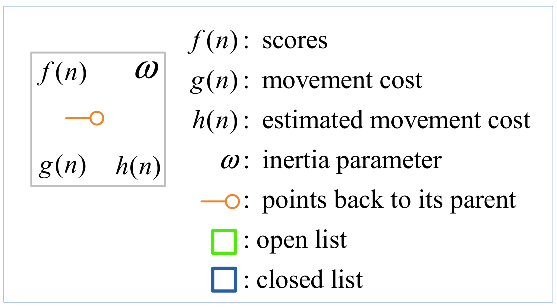


Fig. 9 Mean explanation of notation and values in the block of node.

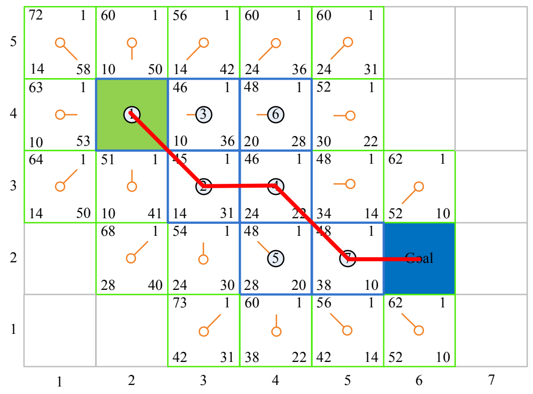


Fig. 10 Description of the optimal path determined by a traditional A* algorithm.

3. EXPERIMENTAL RESULTS

In the proposed method, different path will be chosen according to the selected parameters. Simulation results of six different parameter sets are sown in Fig.11 and their performance of path length, turning number, and execution time (searching steps) are summary in Table 1. In this paper, we choose the parameter set: $\alpha = 150$, $\beta = 2.5$, $D_{\max} = 40$, and $w = 1.3$. In Fig.12, three different paths are determined by a traditional A* algorithm, a modified A* algorithm with w=1, and the proposed method. The performance of path length, turning number, and execution time are summary in Table 2. In Fig. 12 (a), we can see that the shortest path lets the mobile robot very close to obstacles. This path will let the mobile robot collide obstacles easily in the real environment. In Fig. 12 (b) and (c), we can see that this problem has been solved by the proposed A* algorithm. The mainly improved is according to the relationship between obstacles and distance to add safety costs for the cost function of A* algorithm so that it has enough information to evaluate a safe path for the mobile robot. Furthermore, a smooth path can be determined when the inertia is considered in the evaluation function.

Table 1 Performance of path planning under different parameters.

Parameters			Path length (cm)	Turning number	Execution time (ms) or Searching steps
α	β	$D_{\max}$			
150	2.5	10	774	5	9.5 (1108)
150	2.5	20	798	5	9.5 (1082)
150	2.5	40	859	7	8.4 (977)
150	2.5	60	921	12	12 (1511)
150	1	40	895	7	8.2 (946)
150	5	40	829	7	9.2 (1064)

The proposed approach has also been implemented and tested on our mobile robot system. As shown in Fig. 13, the mobile robot is equipped with a laser range finder and two drive wheels which have encoders. A laser range finder, Hokuyo UTM-30LX, is used for the area scanning. Furthermore, the proposed safe path planning method had been used to attend the "2010 SKS Security Robot Competition" and won the Champion. The actual field of experiment and a simulation result are shown in Fig. 14, in which the size of the competition field is 6×6 meters and there are five different kind of rooms. It illustrates that the proposed method can let the autonomous mobile robot have a good path planning in a known environment.

Table 2 Performance comparison of three path planning methods.

Type	Path length (cm)	Turning number	Execution time (ms)
Traditional A*	776	14	9
Modified A* (w=1)	865	17	8.2
Proposed method	859	7	8.4

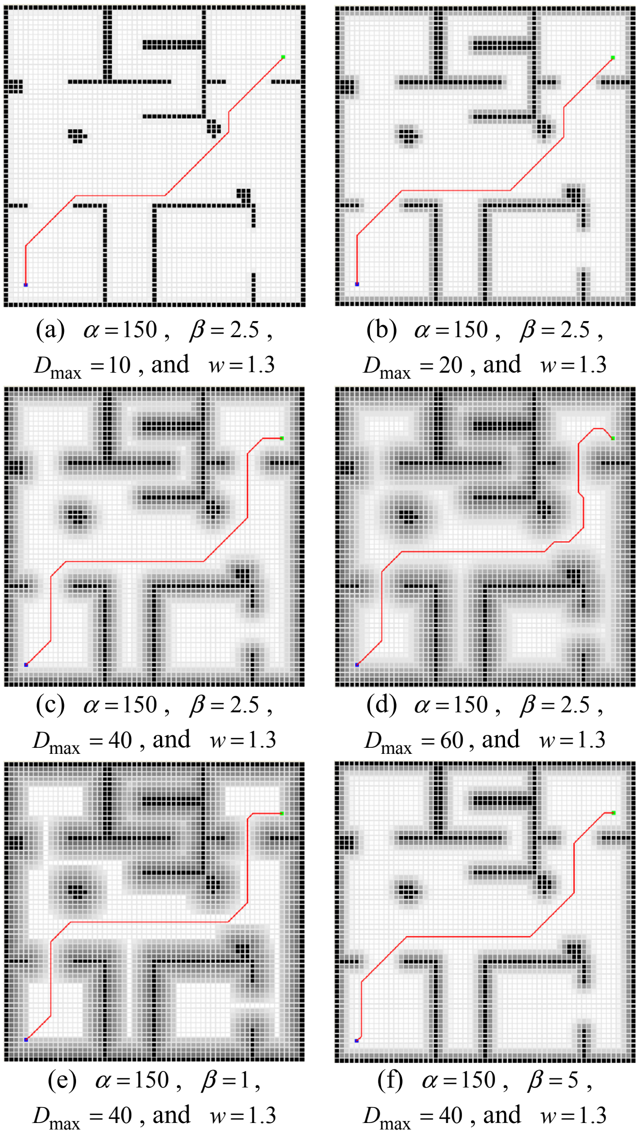
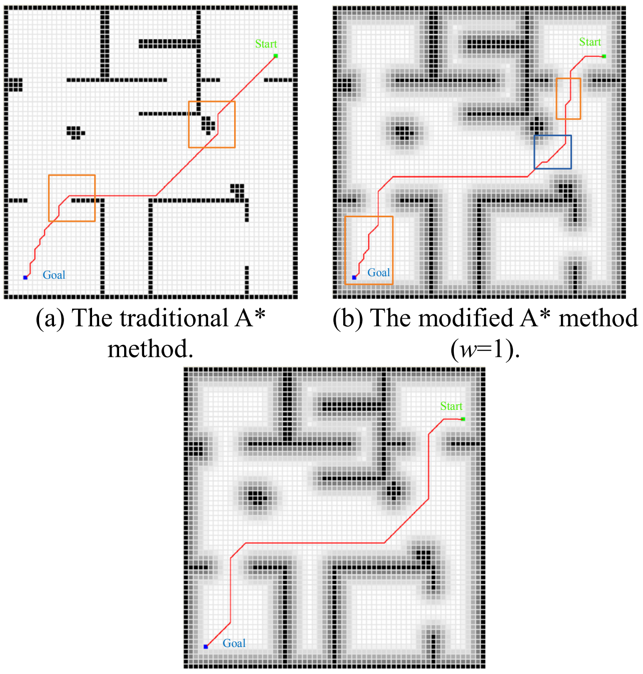

(a) $\alpha = 150$, $\beta = 2.5$, $D_{\max} = 10$, and $w = 1.3$
(b) $\alpha = 150$, $\beta = 2.5$, $D_{\max} = 20$, and $w = 1.3$
(c) $\alpha = 150$, $\beta = 2.5$, $D_{\max} = 40$, and $w = 1.3$
(d) $\alpha = 150$, $\beta = 2.5$, $D_{\max} = 60$, and $w = 1.3$
(e) $\alpha = 150$, $\beta = 1$, $D_{\max} = 40$, and $w = 1.3$
(f) $\alpha = 150$, $\beta = 5$, $D_{\max} = 40$, and $w = 1.3$

Fig.11 Path planning results of the proposed method based on different parameter sets.



(a) The traditional A* method.
(b) The modified A* method (w=1).
(c) The proposed path planning method.

Fig. 12 Results of the path planning by three path planning methods.

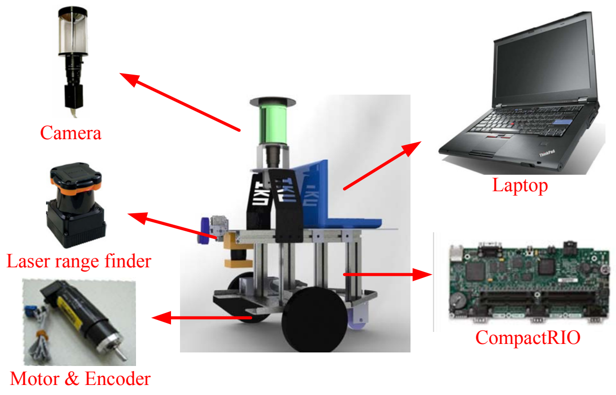


Fig.13 Robot system architecture.

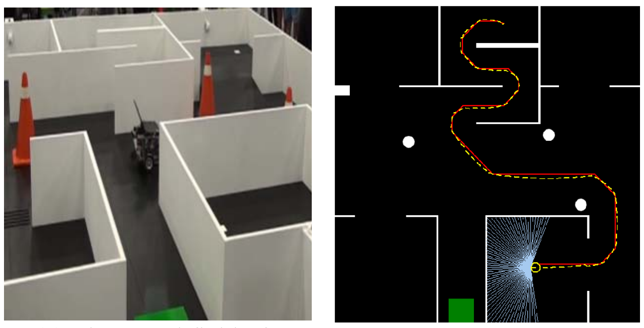

(a) The actual field of experiment
(b) Experiment result

Fig. 14 Experiment environment and the implementation result.

4. CONCLUSIONS

In this paper, a path planning method based on a modified A* algorithm is proposed to find a most safe and fastest path for mobile robots. A safe and a distance movement characteristic of the node are considered in the proposed method, thus it can determine an appropriate path that avoids the collision with obstacles and moves to the destination quickly. An example is presented to evaluate the proposed path planning method and the traditional A* algorithm in a simulation experiment. Although the A* algorithm can determine a shortest path, this may let the robot collide obstacles in an actual environment. From the results, we can see that a path determined by the proposed method is more smooth and effective than that determined by A* algorithm. Furthermore, the number of turning points of the proposed method is less than that of the traditional A* algorithm. It is very important for mobile robots to move smooth in an actual environment.

ACKNOWLEDGMENT

This research was supported in part by the National Science Council (NSC) of the Republic of China under contract NSC 99-2221-E-032 -027.

REFERENCES

[1] M. Barbehenn, “A note on the complexity of Dijkstra’s algorithm for Graphs with Weighted Vertices,” *IEEE Tran. on Computers*, vol.47, no.2, 1998.

[2] P. E. Hart, N. J. Nilsson, and B. Raphael, “A formal basis for the heuristic determination of minimum cost paths in graphs”, *IEEE Trans. of Systems Science and Cybernetics*. vol.SSC-4, no.2, pp.100-107, 1968.

[3] T. Goto, T. Kosaka, and H. Noborio, “On the heuristics of A* or a algorithm in ITS and robot path-planning,” *2003 IEEWRSJ Intl, Conference on Intelligent Robots and Systems*, Las Vegas, Nevada, 2003.

[4] Amit’s Game Programming Information, http://theory.stanford.edu/~amitp/GameProgramming/AStarComparison.html#S3

[5] M. Fu and B. Xue, “A path planning algorithm based on dynamic networks and restricted searching area,” *IEEE Int. Conf. on Automation and Logistics*, pp.1193-1197, 2007.

[6] I. Chabini and L. Shan, “Adaptations of the A* algorithm for the computation of fastest paths in deterministic discrete-time dynamic networks,” *IEEE Trans. on Intelligent Transportation Systems*, vol.3, no.1, pp.60-74, 2002.

[7] M. Nakamiya, Y. Kishino, T. Terada and S. Nishio, “A route planning method using cost map for mobile sensor nodes,” *Int. Symp. on Wireless Pervasive Computing*, pp.168-174, 2007.

[8] E. P. F. Chan and N. Zhang, “Finding shortest paths in large network systems,” *9th Int. Conference on Advances in Geographic Information Systems*, ACM Press New York, NY, USA, 2001.

[9] M. Hashemzadeh, “A fast and efficient route finding method for car navigation systems with neural networks,” *10th IEEE International Enterprise Distributed Object Computing Conf. (EDOC'06)*, pp.423-426, 2006.

[10] G.E. Jan, K. Y. Chang, and I. Parberry, “Optimal path planning for mobile robot navigation,“ *IEEE/ASME Transaction on Mechatronics*, vol. 13, no. 4, pp. 451-459, 2008.