A Shortest Path Planning Algorithm for Mobile Robots Using a Modified Visibility Graph Method

Duk-Young Lee[1], Kyung-Chul Koh[2] and Hyungsuck Cho[1]

[1]*Department of Mechanical Engineering*
Korea Advanced Institute of Science and Technology
{ducky2, hscho}@lca.kaist.ac.kr
[2]*Department of Mechanical & Control Engineering*
Sumoon University
kckoh@sunmoon.ac.kr

Abstract

This paper presents a global path planning algorithm based on a visibility graph method, and applies additionally various constraints for constructing the reduced visibility graph. The modification algorithm for generating the rounded path is applied to the globally shortest path of the visibility graph using the robot size constraint in order to avoid the obstacle. In order to check the visibility in given 3D map data, 3D CAD data with VRML format is projected to the 2D plane of the mobile robot, and the projected map is converted into an image for easy map analysis. The image processing are applied to this grid map for extracting the obstacles and the free space. Generally, the tree size of visibility graph is proportional to the factorial of the number of the corner points. In order to reduce the tree size and search the shortest path efficiently, the various constraints are proposed. After short paths that crosses the corner points of obstacles lists up, the shortest path among these paths is selected and it is modified to the combination of the line path and the arc path for the mobile robot to avoid the obstacles and follow the rounded path in the environment. The proposed path planning algorithm is applied to the mobile robot LCAR-III.

1. Introduction

Path planning is to find a collision-free path which safely guide the robots to reach target point in working space with unpredictable obstacles. Particularly, the path planning of autonomous mobile robots moving in structured environments assumes the working space as 2-dimensional plane and considers obstacles as polygon or circular model. The specific path should be designed by considering the total length of path, curvature variation, and the time to complete the path. Since the last decade, there have been many advanced research works on path planning of mobile robots which can be classified into artificial potential approach[1-2], image processing approach[3-4] and graph search method[5-11]. The artificial potential method produces attraction potential field to objective point and repulsive potential field against obstacles. The merit of this method lies in simplicity of the algorithm while it is difficult to predict the path from the potential field. It also has the possibility of falling into local minima problem. The image processing approaches include the quadtree method[3] and the skeleton method[4] which are effective for dynamic environments with moving obstacles and target. However, they cannot use a priori information about the environment described by CAD data. Finally, the graph search methods include the configuration method[5], MAXLINK[6], generalized Voronoi diagram graph methods[7], and visibility graph methods[8-11]. These methods can easily predict the planned path and use the CAD information about the working environments, while they require rather long computation time searching for appropriate paths among all the possible ways to target from departure point.

In this study, we developed a more effective path planning method, which combines the image processing method with the graph search method. In particular, our work is mainly focused on reducing its algorithm processing time for real time application. Various constraint algorithms are proposed in implementation of the modified VG method. The proposed path planning algorithm is really applied to a mobile robot, LCAR-III, which has been developed for this research. The experimental results show the practical usefulness of the proposed algorithm.

2. The Proposed Path Planning

This work presents a method to find the shortest path for a mobile robot from a starting position to a goal position using a given map data. The shortest path passes through the corner points of obstacles and the visibility

graph method began from this idea by Lozano-Perez, 1979[8]. In the conventional approaches using the visibility graph method, the computational time is very long because of the graph size. In order to find the shortest path with shortest possible time, we propose a modified visibility graph method with additional constraints, which remarkably reduces the computational time. The corner points and the line features of the obstacles are extracted from the given 3D map data, and then the proposed method checks the visibility between arbitrary two points in the grid map. Additionally, some constraint conditions are applied to the visibility check routine in order to reduce each visible connection between the corner points. Therefore, the size of the visibility graph and the computational time can be efficiently reduced. When the mobile robot follows these planned paths, it may approach obstacles and contact with them. Therefore, the size of the mobile robot must be considered to generate the final path, and the kinematic condition of the robot must be considered to make smooth turn of the moving direction. In this work, we propose a path planning method which searches for the globally shortest path of the visibility graph and generates the rounded path consisting of line or arc segments preventing the robot from colliding with obstacles.

2.1 Map Analysis

Fig. 1 shows the process to transform the 3D map data to the 2D grid map. Fig. 2 shows an example of the result of this process. At first, the 2D map data is composed of the lines and the circles, which are obtained from projecting 3D map data as shown in Fig. 2(a) to the motion plane of the mobile robot. The circle features are divided to a line set for reducing the complex of the map. The lines represent the boundaries of the obstacles. And the corner features are extracted from this line set, which are drawn as shown in Fig. 2(b): The corner feature is important to calculate intermediate path points for the path planning. In order to check the visibility between two adjacent corner points, the map should classify the pixels occupied by obstacles and the pixels representing the free space as shown Fig. 2(c). The visibility means that there is no obstacle between two adjacent points. The grid map is used to check the visibility between two adjacent corner points: The visibility means that there is no obstacle between two adjacent points. The map is composed of the pixels occupied by obstacles and the pixels representing the free space as shown Fig. 2(c). Thus, the visibility between two corner points can be determined from checking whether there are occupied pixels on the straight line connecting the two points or not. Let us assume that the starting position of the mobile robot is known. The mobile robot will be located at free space. We can separate the obstacles with the free space.

The free space is filled as expanding the seed point at the starting position by using image filling algorithm. The other points become the obstacle in the grid map. Therefore, the grid map is divided the free space points and the occupied points as shown in Fig. 2(c).

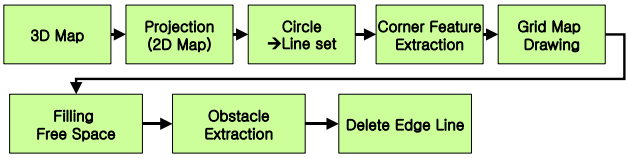
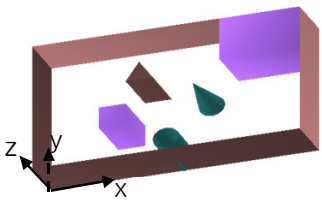
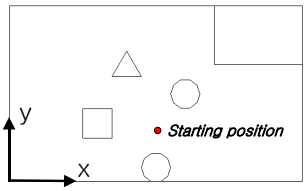


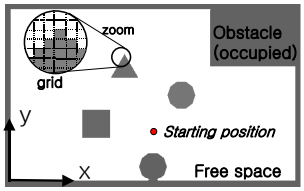
Fig. 1 Map analysis sequence



(a) 3D map



(b) 2D line map

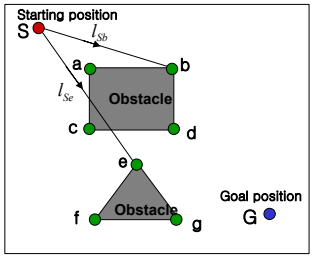


(c) 2D occupied grid map

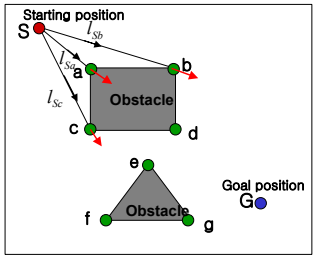
Fig. 2 Map Analysis

2.2 Short Path Planning

The visibility graph is the list of path points which can be connected by straight line in free space without crossing the obstacles. In the example of Fig. 3(a), line S-b is visible, but line S-e is not visible.



(a) Visibility



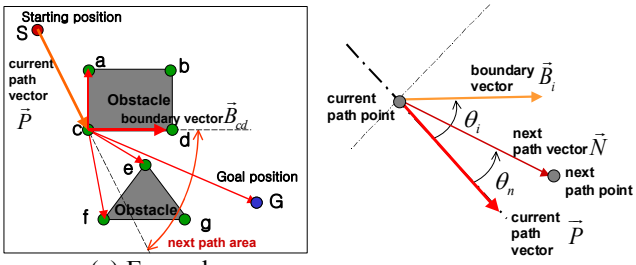
(b) Tangential link

Fig. 3 Visibility check

The previous research[9] introduced the concept of tangential link which effectively reduces the number of visible lines. The tangential link denotes whether the extended path across the visible point is passable or not. From this definition, the line S-c and S-b of Fig. 2(b), are tangential link, but the line S-a is not tangential link.

In this work, the additional constraints are proposed to

reduce the size of the visibility graph. One of those is called the "area constraint". The area constraint limits the possible area of the next path point during making up the path list.

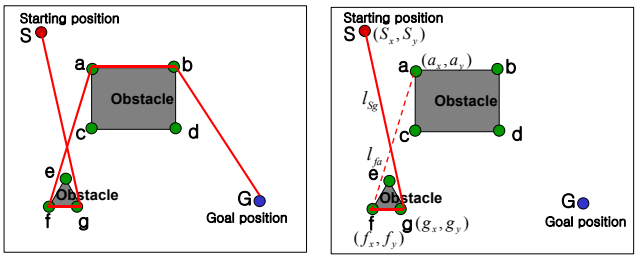


(a) Example (b) Vector

Fig. 4 Area Constraint

The area constraint considers next path points in the space between the current path vector and the boundary vector as the candidates of next visible path points. The current path vector denotes the vector from the previous path point to current path point, and the next path vectors are the vectors from the current path point to the next path points.

The boundary vector is defined as one of the boundaries including the current path point in the obstacle, of which direction is closer to the current path vector. In Fig. 4, if the current path point is point c, $\vec{P}$ is the current path vector and $\vec{B}_{cd}$ is the boundary vector to have the smallest angle with current path vector of the boundary vectors including the current path point. Points d, e, G can be the next path points, but point f and a can not be the next path point. Point f can be directly connected from point S and the point c is not needed as a via point to reach point f from starting with point S. In this case, point S becomes the previous path point for the point f. In Fig. 4(b), if θ_n is smaller than θ_i, the next path point satisfies the area constraint and adds to the visibility graph. These angles are calculated from inner product between two vectors.



(a) Cross path (b) Cross check

Fig. 5 Cross constraint

In the visibility graph, both tangential link and area constraint are simultaneously applied to generate the reduced visibility graph. However, the path may still cross the already passed space in complex environments. Thus, the secondary additional constraint called "cross constraint" is also added to avoid this situation. In the following visibility checking step considering the next path point, the cross constraint checks whether the next path vector crosses the previous paths from the starting position to the current path point.

In the example of Fig. 5, path S-g-f satisfies the area constraint and f-a does so, but path S-g and path f-a cross each other. The path in Fig 5(a) can be simply replaced by the path S-b-G. Therefore, when the current path point is point f and the next path point is point a, the cross constraint checks whether the line l_{fg} crosses the previous paths, lines l_{Sg} and l_{fg} before the point a is added to the visibility graph. By the cross constraint, therefore, point a is excluded from the visibility graph.

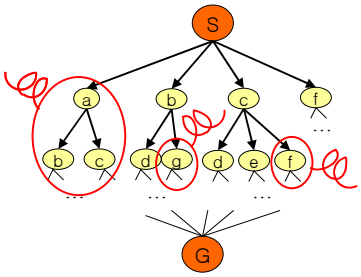


Fig. 6 Eliminating the unsatisfied path

As shown in Fig. 6, the visibility graph has a shape like tree structure. Its top level is the starting position, S, each node on the sub-levels denotes the intermediate path points and the last one becomes the goal position, G. If the traditional visibility graph method is applied to constructing the path, this tree will be a very large structure particularly in the case of numerous obstacles which cause exponential increase of nodes in the tree. The proposed method employing additional two constraints can dramatically remove the long paths and make the visibility graph have more efficient structure from the point of view of computational time. The figure 6 shows that elimination of the nodes in upper level can automatically decrease numerous candidate nodes in lower levels belonging to the eliminated node.

2.3 Constraint effect

Fig. 7 shows the shortest path planning results at each given map. Table 1 compares the number of the obtained path list for each given map according to the constraints. These results show the effectiveness of using the area constraint and the cross constraint.

In the proposed method, paths crossing via points from the initial position to the goal position are candidates for being the shortest path in the given free space. One belong to these path lists will be the globally shortest

path. The shortest path can be selected by comparing the total length of paths from initial position to final goal position.

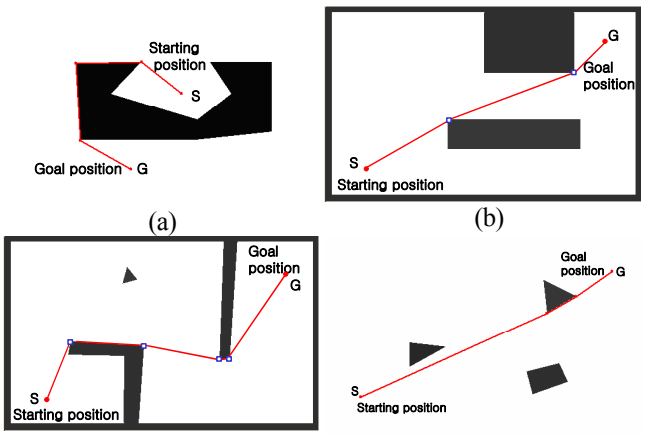


Fig. 7 Shortest path planning

Table. 1 The path number at the map shown as Fig. 7

Environment / Planning method	(a)	(b)	(c)	(d)
Tangential Link	20	20	32	19779
*Tangential+Area Constraint	2	2	3	34
*Tangential+Area+Cross Constraint	2	2	2	12

* Proposed method.

3. Rounded Path Generation

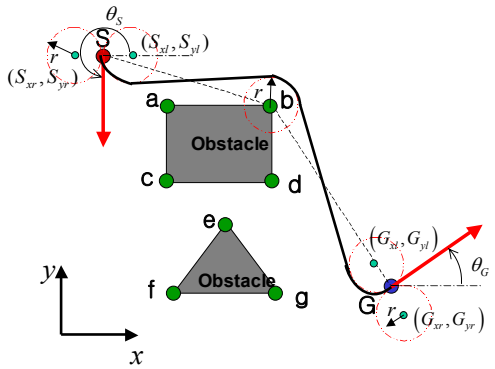


Fig. 8 Rounded Path Generation

The straight line paths simply connecting via points are not inappropriate for non-holonomic mobile robots with differential wheels. To cope with this, we will modify the shortest line path passing the corner point of the obstacles to be rounded path which consists of the line paths and the arc paths.

Fig. 8 shows a simple example of the rounded path. In starting position, S, and goal positions, G, the pose of the mobile robot is represented by its position in x-y coordinates and the angle of direction. The angle of direction is defined by angle between heading direction of the robot and x-coordinate. Its sign becomes positive with direction of counterclockwise. As shown in Fig. 8(b), virtual circles enclose each corner point of obstacles with radius of r. The radius r is determined by considering robot size and safety factor. At starting and goal positions, there are two virtual circles tangential to each direction, respectively. From two circles at each position, the only one circle is selected by considering tangential line to the circle at the next or previous visible corner points, respectively. Each center of two circles is located on the points distanced by r in perpendicular to the robot direction. The position of the virtual circles at the starting position and the goal position can be calculated by:

- At starting position (S)

Left virtual circle: $S_{lx} = S_x - r\sin\theta_S$, $S_{ly} = S_y + r\cos\theta_S$

Right virtual circle: $S_{rx} = S_x + r\sin\theta_S$, $S_{ry} = S_y - r\cos\theta_S$

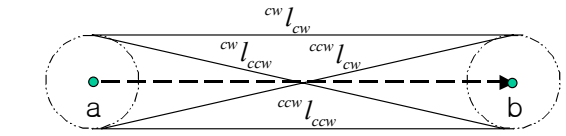
- At goal position (G)

Left virtual circle: $G_{lx} = G_x - r\sin\theta_G$, $G_{ly} = G_y + r\cos\theta_G$

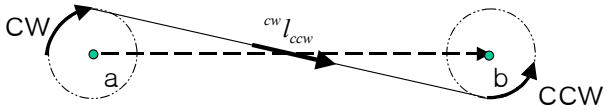
Right virtual circle: $G_{rx} = G_x + r\sin\theta_G$, $G_{ry} = G_y - r\cos\theta_G$ (1)

where (S_x, S_y) is the x-y coordinate components of the starting position while (G_x, G_y) denotes those of the goal position. θ_S and θ_G are the viewing angles at the starting position and the goal position, respectively. Subscripts x and y represent the x and y coordinates of each point, and subscripts r and l describe the right and the left positions of each virtual circle center with respect to the direction of the mobile robot.

Fig. 9(a) shows the four tangential lines between the virtual circle, a and b with respect to the shortest path from point a to point b. In general, there are four tangential lines between two circles. The rotational direction of the arc path at each virtual circle is divided into two cases, the clockwise direction (cw) and the counterclockwise direction (ccw) as shown in Fig. 9(b). In the line notation l, the superscript describes the rotational direction of the previous path point a, and the subscript describes the rotational direction at the next path point b. One of four tangential lines is selected as the shortest rounded path connecting two points, a and b, if the resultant path does not pass any zone occupied by obstacles on the grid map. Therefore, the path consists of the connected lines and arc paths, which starts with arc path at the starting position, connects to intermediate tangential lines and the arc paths around the corners of obstacles, and finally arrives at the goal position with its tangential circles.



(a) Four cases



(b) Rotational direction (CW-CCW case)

Fig. 9 Tangential lines to the virtual circle

4. Navigation Experiment

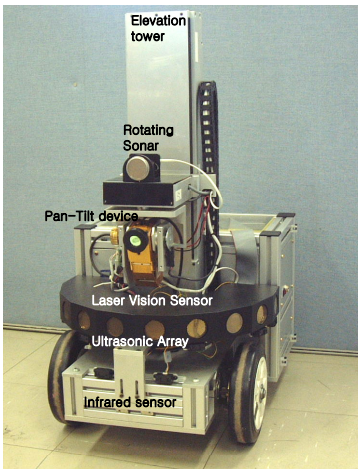


Fig. 10 Mobile Robot (LCAR-III)

To investigate the proposed path planning algorithm in real environment, a series of experiments were conducted for a mobile robot LCAR-III shown in Fig. 10. The robot consists of the equipments for the navigation and the 3D environmental measurement. The navigation part is two driven wheels, one supplementary rear wheel, ten ultrasonic sensors and eight infrared sensors. The ultrasonic sensor and infrared sensor are used to detect and avoid the obstacles during navigation. The environment measurement part consists of a laser vision sensor and a driving part for motion of this vision sensor. The laser vision sensor includes a laser structure light and a CCD camera. This sensor can measure the 3D information of the environment using the triangular method. To scan the sensor, the driving part of the laser vision sensor system can generate the pan-tilt motion and elevation.

In order to implement the proposed global path planning results, we applied a proportional path tracking controller as given by:

$$\begin{aligned} v_x &= K_x e_x \\ \omega &= K_y e_y + K_\theta e_\theta \end{aligned} \qquad (2)$$

where v_x and ω are the linear and the rotational velocity commands, respectively. e_x, e_y and e_θ are the x direction error, the y direction error and the rotational angle error with respect to the current position and orientation. In the above, K_x, K_y and K_θ are the controller gains of each error. Once the linear and the rotational velocity of the mobile robot is obtained from the equation (2), each wheel velocity is calculated from the kinematic relation of the mobile robot. The kinematic relation of the two-wheel driven LCAR-III is written by

$$\begin{aligned} \omega_L &= \frac{\left(v_x - \omega(D_w/2)\right)}{R_w} \\ \omega_R &= \frac{\left(v_x + \omega(D_w/2)\right)}{R_w} \end{aligned} \qquad (3)$$

where ω_L and ω_R denote left and right wheel velocity, respectively, D_w and R_w are the distance between two wheels and the wheel radius, respectively.

Fig. 11 shows the experimental results for a given map. Fig. 11(a) shows a given environment map that describes the navigation condition including the starting position, S, and the goal position, G.. The grid map size is 500 x 300 pixels and the real map size is 5m x 3m. The starting position is about (0.6, 0.7)m. For the localization at the starting position, the mobile robot measured the distance with two walls using the laser vision sensor system and the mobile robot turns to 0 degree. The measured starting position is used for the map analysis. The goal position is (4.2, 2.2)m and its orientation is 45 degree. The stable factor r is 0.4m. The proposed method generates 66 short paths, which consist of the line paths passing the corner points, and then shortest one of those is selected as the candidate path. The rounded path to be consisted of piecewise line and arc are generated from this shortest line path as shown in Fig. 11(a). The mobile robot navigates through the given environment using the rounded path as shown in Fig. 11(b)~(e) according to the controller gains, and can detect and avoid the obstacles using the ultrasonic sensors. The pose of the mobile robot during the navigation is calculated by the integration of the encoder. The dashed line denotes the desired path and the dot line denotes the real tracking path of the mobile robot in the result figures. Fig. 11(b) and (c) show the effect of the controller K_x. The real tracking path has the delay with desired path due to small K_x. Fig. 11(b) and (e) show the effect of the controller gain K_θ. K_θ is important to track the desired path at the arc path because the moving direction of the mobile robot are changing. Fig. 11(b) and (d) show the effect of the controller gain K_y. Fine-tuned K_y can reduce the error of lateral direction. Fig. 12 shows the scenes of this navigation experiment at this given environment.

5. Conclusion

In order to construct more effective visibility graph, we obtained the grid map and the corner point list of the obstacles from the 3D VRML map data. Additionally, the tangential link, the area constraint and the cross constraint are applied, so that the size of the visibility graph is drastically reduced. Therefore, the computational time and memory requirement can be decreased. In next step, the rounded path for non holonomic robots is generated based on globally shortest path selected from these path lists, which describe the shortest path in each free space. The rounded path is composed of the line and the arc path. The proposed method makes the visibility graph applicable to real time navigation system. In the rounded path generation, the robot size has influence on the radius of the virtual circle and also, the resultant rounded path. If the robot size is considered as the constraint at the generation process of the visibility graph, the visibility graph will be reduced more effectively.

Reference

[1] F. Miyazaki and S. Arimoto, "Sensory feedback for manipulators", J. of Robotic Systems, Vol.2, No.1, pp.53-72, 1985.

[2] O. Khatib, "Real time Obstacle Avoidance for Manipulators and Mobile Robots," Int. J. Robotics Research, Vol.5, No.1, 90-98, 1986.

[3] S. Kambhampati and L. S. Davis, "Multiresolution Path Planning for Mobile Robots," IEEE J. of Robotics and Automation, Vol. RA-2, No.3, 135-145,1986.

[4] H. R. Beom, "Path Planning for Mobile Robot Using Skeleton of Free Space", Proc. of IFAC-INCOM, 509-513, 1992.

[5] T. Lozano-Perez, "Spatial planning: A configuration space approach", IEEE Trans. Comput., vol. C-32, pp.108-120, 1983.

[6] M. K. Habib and H. Asama, "Efficient Method to Generate Collision Free Paths For Autonomous Mobile Robot Based on New Free Space Structuring Approach", Proc. of IEEE/RSJ Int. Workshop on Robots and Systems, 563-567, 1991.

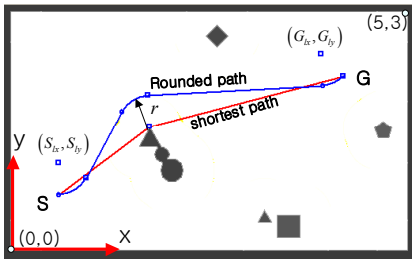
[7] R. Mahkovic and T. Slivnik, "Constructing the Generalized Local Voronoi Diagram from Laser Range Scanner Data", IEEE Trans. on SMC, Vol.30, No.6, 710-719, 2000.

[8] T. Lozano-Perez and M. A. Wesley, " An Algorithm for Planning Collision-Free Paths among Poyhedral Obstacles, " Commun. ACM, Vol.22, pp.560-570, 1979.

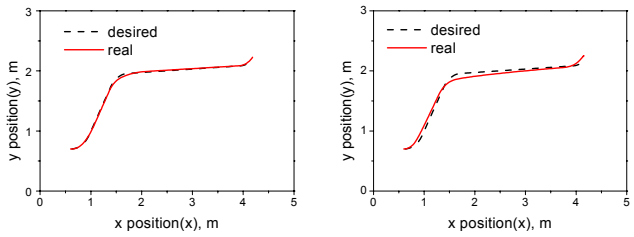
[9] H. Rohnert, "Shortest paths in the plane with convex polygonal obstacles," Information Processing letters, 23, 71-76, 1986.

[10] Y. H. Liu and S. Arimoto, "A Flexible Algorithm for Planning Local Shortest Path of Mobile Robots Based on Reachability Graph", Proc. of IEEE/RSJ Int. Workshop on Robots and Systems, 749-756, 1990.

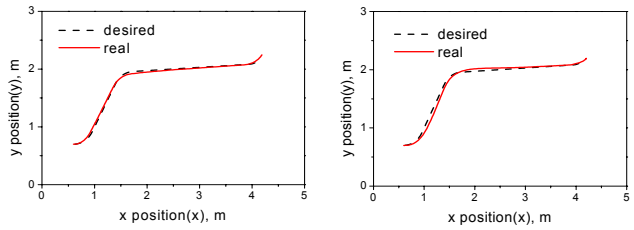
[11] K. Jiang and et al, "A Shortest Path Based Path Planning Algorithm for Nonholomic Mobile Robots", J. of Intelligent and Robotic Systems, 24, 347-366, 1999.



(a) Given environment map and Planned path

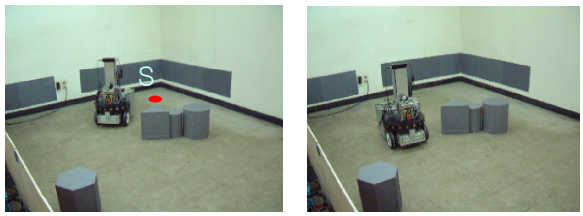


(b) $K_x = 15$, $K_y = 15$, $K_\theta = 20$ (c) $K_x = 10$, $K_y = 15$, $K_\theta = 20$



(d) $K_x = 15$, $K_y = 25$, $K_\theta = 20$ (e) $K_x = 15$, $K_y = 15$, $K_\theta = 10$

Fig. 11 Experimental results



(a) (b)

(c) (d)

Fig. 12 Navigation experiment