Dynamic-Programming Approach to Path-Planning of Autonomous Mobile Robots

Hee-Sang Yoon and Tae-Hyoung Park

Dept. of Control and Instrumentation Eng., Chungbuk National University, Cheongju, Chungbuk 361-763, Korea.
(Tel : 82-43-261-3240; Fax : 82-43-271-3240)
(taehpark@cbnu.ac.kr)

Abstract: We propose a new path planning method for autonomous mobile robots. To maximize the utility of mobile robots, the collision-free shortest path should be generated by on-line computation. In this paper, we develop an effective and practical method to generate a good solution by lower computation time. The initial path is obtained from skeleton graph by Dijkstra's algorithm. Then the path is improved by changing the graph and path dynamically. We apply the dynamic programming algorithm into the stage of improvement. Simulation results are presented to verify the performance of the proposed method.

Keywords: Robot path planning, Autonomous mobile robots, Dynamic programming, Path optimization, Local search.

I. INTRODUCTION

Path planning of autonomous mobile robots is to find a path from a start point to a goal point inside of a given map. Fig.1 shows an example of the path planning problem. To increase the productivity of mobile robots, the total moving distance should be minimized considering obstacles. For on-line path planning in the shop floor, the computation time required to generate the path should also be minimized. [1]

Several approaches including potential field [2], and graph search [3,4] have been developed to solve the path-planning problem. The graph-search methods are most popular since the shortest path algorithms such as A* algorithm or Dijkstra's algorithm can be easily applied, which makes a good performance for complex maps. To apply the shortest path algorithm, a graph which consists of vertices and arcs should be crated on given map. Fig.2 shows graphs created for path planning of mobile robots. The vertices of visibility graph (V-graph) and skeleton graph are all corner points of obstacles and cross points of skeleton lines, respectively. The computational complexity of the shortest path algorithms depends on the number of vertices (or arcs), so the number of vertices is desirable to be minimized. The V-graph usually generates the shorter path, but needs more calculation time due to its bulky number of vertices and arcs.[1]

Traditional graph-search methods usually solve the problem through two hierarchical and independent stages: the graph stage and the path stage. The graph stage creates a graph by locating vertices at appropriate points of map, and the path stage generates a path by use of the shortest-path algorithm. Since the path is a sequence of vertices, the vertex location is important in minimizing distance of the path. Hence, it is desirable to decide the point location and the path simultaneously to improve the performance.

In this paper, we propose a new method determining the vertex location and the path simultaneously. We generate the path through two stages: path-generation stage and path-improvement stage. The path-generation stage creates the initial vertex location and initial path

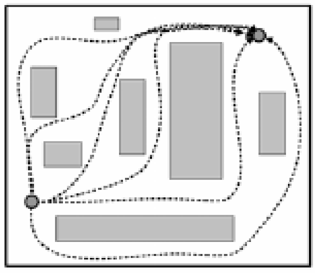


Fig.1. A path-planning problem.

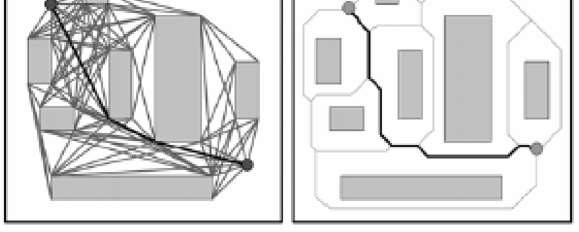


Fig.2. Graphs for path planning .
(a) visibility graph (b) skeleton graph

by the thinning algorithm and Dijkstra's algorithm, respectively. The path-improvement stage modifies the initial vertex location and the initial path to reduce the overall distance. We apply dynamic programming technique into this stage. Finally, the performance of the proposed method is verified by computer simulation.

II. PROBLEM DEFINITION

We define path-planning problem of autonomous mobile robots mathematically. Define $s \in R^2$ and $g \in R^2$ be the start and goal point of a mobile robot, respectively. Each obstacle located in the plane is represented by rectangle area. Let O_i be the area of i th obstacle as

$$O_i = \{(x, y) \mid l_i \le x \le r_i, b_i \le y \le t_i\}, i = 1, \cdots, N \quad (1)$$

, where l_i, r_i, b_i and t_i are of left, right, bottom and top values of rectangle. To consider the interference of robot and obstacle, we assume that each area of obstacle in (1) includes the area of robot. Then, the robot can be considered as a point moving in the plane.

The moving path of robot between the start and goal point is defined as

$$p = < v_1, v_2, \cdots, v_M > \quad (2)$$

, where $v_1 = s$, $v_M = s$, and $v_k \in R^2$ is the k -th via points of path. Define the arc (v_k, v_{k-1}) as the straight-line between v_k and v_{k+1}, where the following collision-free constraints should be satisfied.

$$(v_k, v_{k+1}) \not\subset O_i, \ \forall_{i \in \{i, \cdots, N\}}, \ k = 1, \cdots, M-1 \quad (3)$$

Let $d(v_k, v_{k+1})$ be the Euclidean distance between v_k and v_{k+1}, then total distance of path is

$$\sum_{k=1}^{M-1} d(v_k, v_{k+1}) \quad (4)$$

Now, the path-planning problem of autonomous mobile robots is to find the optimal path of (2) minimizing the total distance (4) subject to the constraints in (3).

III. PATH-PLANNING ALGORITHM

The local optimization method [7] is very practical approach to solve the complex optimization problem. The method usually generates an initial solution heuristically, and then finds near-optimal solution by iteratively improvement of the initial solution.

We apply the local optimization method to solve the path-planning problem of autonomous mobile robots. The overall path-planning algorithms are divided into two hierarchical stages: path generation and path improvement. The path-generation stage finds an initial path, and the path-improvement stage generates final path by improving the initial path iteratively.

1. Path Generation

To find an initial path of robot, we create a skeleton graph from a given map. The thinning algorithm [5] is a simple algorithm widely used for skeletonization of binary images. We apply the thinning algorithm to create the initial graph. Fig.3 shows steps of thinning algorithm, which generates a skeleton graph by thinning input images iteratively. The vertices and arcs of graph are defined from the final skeleton lines.

Now we generate an initial path on the skeleton graph created. The problem is a typical shortest-path problem [1] that finds a path minimizing overall distance from start vertex to final vertex. The Dijkstra's algorithm [6] is a well known method to solve the shortest-path problem. The optimal path is obtained in $O(V \log_2 V + E)$, where V and E are numbers of vertices and arcs, respectively. Fig.4 shows an example

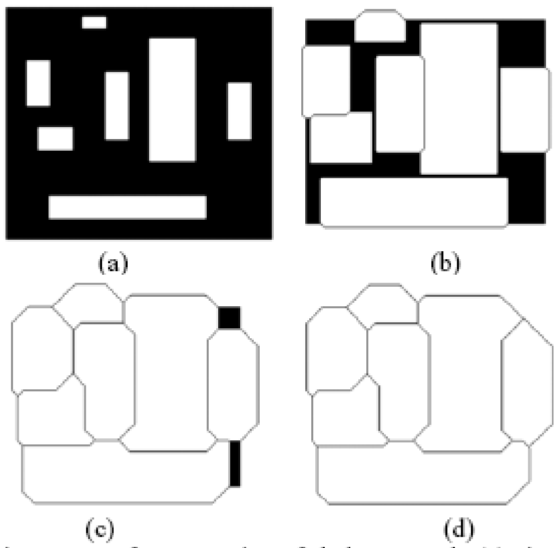


Fig. 3. Steps for generation of skeleton graph. (a) given map (b) step of 10 iterations (c) step of 20 iteration (d) final graph

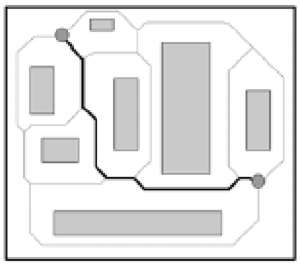
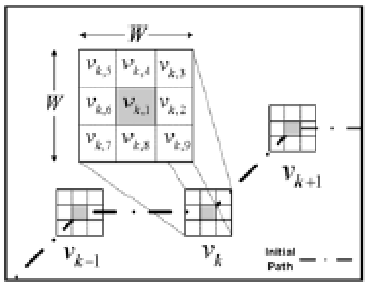


Fig. 4. Initial path.


W
W
$v_{k,5}$ $v_{k,4}$ $v_{k,3}$
$v_{k,6}$ $v_{k,1}$ $v_{k,2}$
$v_{k,7}$ $v_{k,8}$ $v_{k,9}$
v_{k+1}
v_{k-1}
v_k
Initial Path
Fig.5. Grid map for path improvement.

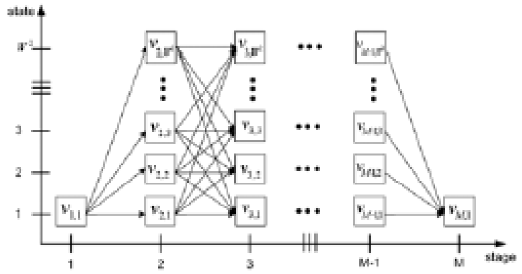


Fig. 6. Search plane for path-improvement stage.

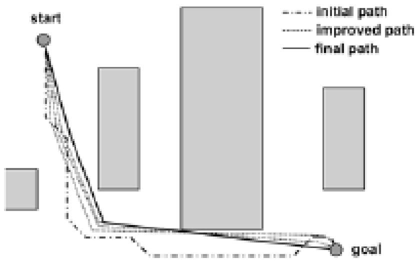


Fig.7. Example of path improvement.

of initial path generated.

2. Path Improvement

In this stage, we improve the initial path by modification of via points. The shortest path of mobile robots usually passes through around corner points of obstacles. However our initial path passes on the skeleton lines, so it needs to move via points toward corner points.

We apply the $W \times W$ grids to each via point located on the initial path as shown in Fig.5. Let $v_k (k = 2, \cdots, M-1)$ be the k-th via point of initial path, and $v_{k,1}, \cdots, v_{k,W^2}$ be center points of each grid. Then $v_{k,1}, \cdots, v_{k,W^2}$ are candidates for new via point which reduces overall distance of path. Since $W \times W$ grids are located for all via points, we need to select the best combination of new via points among $W \times W \times (M-2)$ combinations. Fig.6 shows a search plane for path-improvement stage.

We apply dynamic programming (DP) method to find the optimal grids in the search plane. DP is an optimization method that can generate the global solution by the principle of optimality. [8]

Denote $d^*_{k,a}$ as the minimum distance from start point to $v_{k,a}$, and $d_{(k,a)(k+1,\beta)}$ as distance between $v_{k,a}$ and $v_{k+1,\beta}$. The basic steps for path improvement are as follows:

Step1. Set stage counter k=1, and initialize $d^*_{1,1} = 0$.

Step2. Increment stage counter $(k = 2, \cdots, M-1)$, and calculate $d^*_{k,a}$ by

$$d^*_{k,\alpha} = \min_{\beta}(d^*_{k-1,\beta} + d^*_{(k-1,\beta)(k,\alpha)}) \tag{5}$$

Set the pointer $\pi_{k,\alpha}$ as β generating $d^*_{k,\alpha}$.

Step3. Set stage counter $k = M$. Compute $d^*_{M,1}$ by (5) and set pointer $\pi_{M,1}$.

Step4. Decrement the stage counter $(k = M, \cdots, 2)$, and follow the pointer. Set new via point v^*_k as the pointer of each stage. The new path p' is then

$$p' =< v_1, v^*_2, \cdots v^*_{M-1}, v_M > \tag{6}$$

Step5. Apply new grids for via points of new path, and repeat Step1-4 until no further improvement.

Fig. 7 shows an improvement procedure by DP algorithm. Via points are changed iteratively, which results in the reduction of overall distance of path.

IV. SIMULATION RESULTS

We implemented the proposed method by computer program by C++ language under MS-Windows XP, and executed it at Pentium IV (2.8GHz) personal computer.

We arranged multiple obstacles (5 ~ 230) differently in 13,000(mm) x 10,000(mm) plane. The size of each grid for path improvement was set at 100(mm) x 100(mm), and number of grids for each vertex was set at 3 x 3.

We compared the proposed method with the path-planning methods by skeleton graph (S-graph) and the visibility graph (V-graph). The S-graph method is the

Table 1. Comparison of total moving distance

No. of obstacles	Total distance [mm]			Improvement ratio [%]	
	V-graph method	S-graph method	Proposed method	w.r.t V-graph method	w.r.t S-graph method
5	611	767	642	-5.07	16.30
10	912	977	890	2.14	8.90
20	1024	1236	1037	-1.27	16.10
35	732	963	767	-4.78	20.35
55	1058	1199	1056	0.19	11.93
80	915	1114	939	-2.62	15.71
110	1062	1209	1062	0	12.16
145	900	1032	912	-1.33	11.63
185	1106	1239	1110	-0.36	10.41
230	1155	1294	1165	-0.87	9.97
Average improvement ratio				-1.40	13.35

Table 2. Comparison of computational time.

No. of obstacles	Computation time [sec]		
	V-graph method	S-graph method	Proposed method
5	0.56	0.52	0.56
10	1.08	0.75	0.80
20	8.75	1.05	1.11
35	22.70	1.44	1.50
55	46.52	2.13	2.20
80	89.19	2.75	2.85
110	150.97	3.78	3.86
145	275.56	4.83	4.92
185	418.58	5.94	6.00
230	594.56	7.59	7.69

same with first stage of the proposed method. The V-graph method is known as the best method in generating the shortest path. We also implemented it for comparison.

Table 1 shows the total moving distance of robot for each method. The proposed method reduces the total distance about 13% relative to the S-graph method. It means that the second stage of proposed method has an enough effect on improving the path. The proposed method increase the moving distance a little relative to the V-graph method.

Table 2 shows the computation time required to generate path. The V-graph method needs much larger computation than S-graph. The proposed method needs almost same computation with the S-graph method. It means that the second stage of the proposed method requires very small additional computation. As a result, the proposed method can generate better path with faster computation.

V. CONCLUSION

We proposed a new path-planning method for autonomous mobile robots. The initial path was generated by traditional graph search method. And then the final path was obtained by changing via points through iterative dynamic programming. The comparative simulation results showed that the proposed method can improve the performance by fast computation. The method can be applied to the on-line path planning system of autonomous mobile robots, and improve its productivity and utility. In the future, we will take into account the dynamics of robots in our path planning system.

REFERENCES

[1] J. Latombe, Robot Motion Planning, Kluwer academic publishers, 1996.
[2] R. Brooks, "Solving the find path problems by good representation of free space," IEEE Trans. on Systems, Man and Cybernetics, vol. 13, no. 3, pp. 190-197, 1983.
[3] K. P. Valavanis, T. Hebert, R. Kolluru, N. Tsourveloudis, " Mobile robot navigation in 2-d dynamic environments using an electrostatic potential field," IEEE Trans. on Systems, Man and Cybernetics, vol. 30, no. 2, pp. 187-196, 2000.
[4] S. Simhon, G. Dudek, "A Global topological map formed by local metric maps," IEEE/RSJ Int. Conf. On Intelligent Robot and Systems, pp. 1708-1724, 1998.
[5] E. R. Davies, Machine Vision, 3rd Ed., Morgan Kaufmann publishers, 2004.
[6] T. H. Cormen, C. E. Leiserson, et al, Introduction to Algorithms, 2nd Ed., The MIT Press, 2001.
[7] Z. Michalewicz, D.B.Fogel, How to Solve It: Modern Heuristics, Springer-Verlag, 2000.
[8] T.H.Park, N.Kim, "A dynamic programming approach to PCB assembly optimization for surface mounters", Int. J. of Control, Automation, and Systems, vol.5, no.2, pp.192-199, 2007.

ACKNOWLEDGMENT

This work was supported by the Korea Research Foundation Grant funded by the Korean Government (MOEHRD) (The Regional Research Universities Program/Chungbuk BIT Research-Oriented University Consortium)