# Path Planning Algorithm Using the Particle Swarm Optimization and the Improved Dijkstra Algorithm

Hwan Il Kang[1], Byunghee Lee[1], Kabil Kim [2]
[1]*Department of Information engineering,*
*Myongji University, Yongin, Gyeonggi Province, South Korea,*
[2]*Department of Electrical engineering,*
*Myongji University, Yongin, Gyeonggi Province, South Korea,*
*hwan@mju.ac.kr, lbh0329@naver.com, kkl@mju.ac.kr*

## Abstract

*In this paper, we develop the path planning algorithm using the improved Dijkstra algorithm and the particle swarm optimization. To get the optimal path, at first we construct the MAKLINK on the world environment and then make a graph associated with the MAKLINK. From the graph, we obtain the Dijkstra path between the starting point and the destination point. From the optimal path, we search the improved Dijkstra path using the graph. Finally, applying the particle swarm optimization to the improved Dijkstra path, we obtain the optimal path for the mobile robot. It turns out that the proposed method has better performance than the result [1].*

## 1. Introduction

The generation of the collision-free path may be an essential part for the navigation of the mobile robot. The assumption is that we may be able to know all the information of the robot's known environment. Under theses conditions, there are some methods such as the visibility graph, the artificial potential field and the grid methods. The efficiency of the visibility graph may be low. The artificial potential field method [3] has a simpler structure and may be used in real-time collision-free path. But the method may lose the global optimization property and may be trapped in the local minimum path. The grid method [4] introduces a new approach to modelling robot workspace in two dimensions. The scheme proves to be attractive because it requires less memory space and is execution-efficient. It is very useful for real time applications. The main problem is to how to determine the size of grids.

We may apply the evolutionary algorithm [2], which includes the genetic algorithm, the ant colony algorithm, and the particle swarm optimization, to the collision-free path. We use the particle swarm optimization as an optimization tool. Section 2 presents how to generate the Marklink graph. Section 3 explains the Dijkstra algorithm. In Section 4, we explain the particle swarm optimization. Section 5 presents the improved Dijkstra algorithm. Section 6 presents the experiment. In Section 7, conclusions are followed.

## 2. Graph Generation

To obtain the MAKLINK graph, we are in a position to present the algorithm. Step 1. find the convex set which includes all the obstacles and the boundary of the convex set has the edges of some obstacles. At each vertex of the convex set, we generate the MAKLINK lines. The lines are perpendicular to the whole boundary and choose the shorter one among the perpendicular ones. The line should not intersect the convex set. Step 2. find out all other free MAKLINK lines according to the definition of free MAKLINK line [1]. The boundary of each free convex area to be constructed will have at least two links as apart of its total number of edges. The resulting graph is shown in Fig. 1.

The generation of the graph from Fig. 1 is depicted in Fig. 2. All the vertices come from each midpoint generated from each free link in Fig. 1.

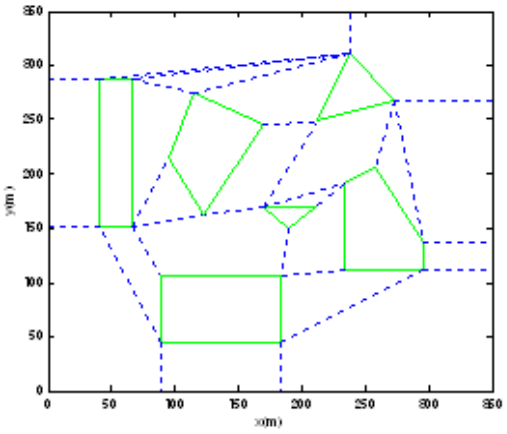


**Fig. 1. All free links including all the obstacles**

978-0-7695-3490-9/08 $25.00 © 2008 IEEE
DOI 10.1109/PACIIA.2008.376

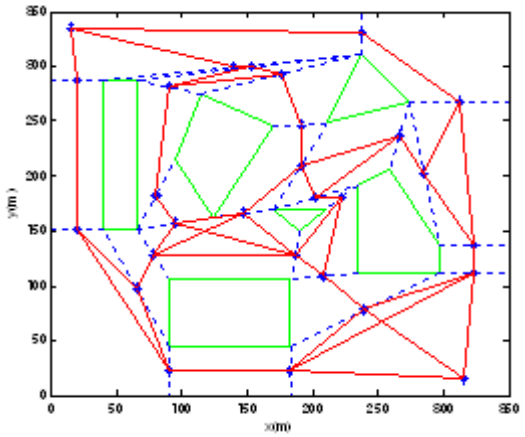

**Fig. 2. The graph generated from the all the free links**

## 3. Dijkstra Algorithm

From the graph, we obtain the shortest path from the starting point and the destination point. Before we describe the Dijkstra algorithm, we need some definitions [2].

$D(v)$ : Cost of the least-cost path from the source node to destination $v$ as of this iteration of the algorithm.

$p(v)$ : previous node ( neighbor of $v$ ) along the current least-cost path from the source to $v$ . $N'$ : subset of nodes; $v$ is in $N'$ if the least cost from path the source to $v$ is definitely known. The Dijkstra algorithm is described as follows:

Initialization :

$N' = \{u\}$

For all nodes $v$

If $v$ is a neighbor of $u$

Then $D(v) = c(u,v)$

Else $D(v) = \infty$

Loop

Find w not in $N'$ such that $D(w)$ is a minimum

Add $w$ to $D(v)$

Update $D(v)$ for each neighbor $v$ of w and not in $N'$:

$$D(v) = \min( D(v), D(w) + c(w,v) )$$

Until $N' = N$ .

Applying Dijkstra algorithm to the Fig. 2, we obtain the shortest path as shown in Fig. 3.

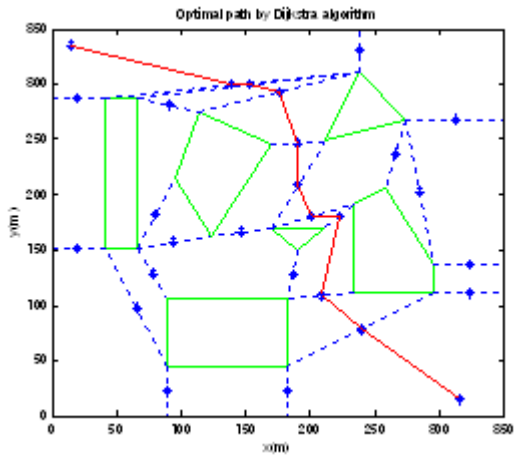


**Fig. 3 The shortest path using the Dijkstra algorithm**

## 4. Particle Swarm Optimization

Particle swarm optimization is an evolutionary computation technique developed by Dr. Eberhart and Dr. Kennedy in 1995, inspired by social behavior of bird flocking or fish schooling. Shi and Eberhart [6] develop a new version of the Particle swarm optimization including the weight adaptation. Compared to GA, the advantages of PSO are that PSO is easier to implement and there are fewer parameters to be adjusted. PSO has been successfully applied in many areas: such as function optimization, artificial neural network training, fuzzy system control and other areas where GA can not be used. In this paper after we generate the improved Dijkstra algorithm, we apply the PSO and get the optimal path.

## 5. Improved Dijketra Algorithm

Now, we are in a position to improve the Dijkstra shortest path. By using the triangular inquality, we try to reduce the length of path from the Dijkstra path. For each vertex, if connecting the other vertex, there would be possibility for reducing the length of the path. If so, we modify the Dijkstra path to a new path. We call it a improved Dijkstra path. See the paper [5].

The optimal collision-free path finding process is as follows:

Step 1: We find the MAKLINK graph from the world environment including all the obstacles. Make a graph including the starting point and the destination point.

Step 2: Frm the MAKLINK graph, we apply the Dijkstra algorithm. We obtain the suboptimal path.

Step 3: Applying the improved Dijkstra algorithm, we obtain the another suboptimal path.

Step 4: We obtain the optimal path form the suboptimal path generated in Step 3. The final path is the optimal path.

## 6. Experiment

In Fig. 4, the three optimal paths are shown. The first path can be obtained by the Dijkstra algorithm, the second by the improved Dijkstra algorithm and the third by both the improved Dijkstra algorithm and the Particle swarm optimization.

As shown in Table 1, the result using the improved Dijkstra algorithm and the PSO has better performance than the other two cases such as the results using the Dijkstra algorithm and the improved Dijkstra algorithm. The previous result have shown that the optimal length turns out to be 439.372. Our best result turns out to has better performance than the result in the previous work [1].

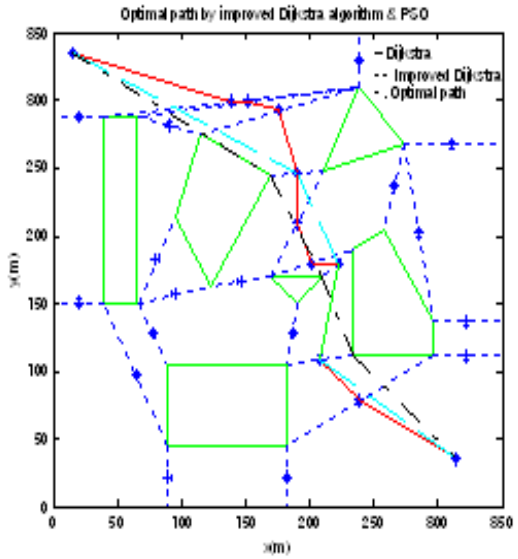


**Fig. 4. Optimal paths by Dijkstra algorithm, improved Dijkstra algorithm, and the improved Dijkstra algorithm and the PSO**

**Table 1. Test Results for the Experiment**

| | By Dijkstra algorithm | By improved DIjkstra algorithm | By improved algorithm & PSO |
|---|---|---|---|
| Length | 507.6910 | 472.5326 | 439.0011 |

## 7. Conclusions

In this paper, we developed the path planning algorithm using the improved Dijkstra algorithm and the particle swarm optimization. To get the optimal path, at first we construct the MAKLINK on the world environment and then make a graph associated with the MAKLINK. From the graph, we obtain the Dijkstra path

## References


[1] Tan Guan_Zheng, HE Huan and SLOMAN Aaron, "Ant Colony System Algorithm for Real-Time Globally Optimal Path Planning of Mobile Robots, Acta Automatica Sinica, vol. 33, no. 3, March 2007, pp. 279—285.

[2] Yuan-Qing Qin, De-Bao Sun, Ning Li and Yi-Gang Cen, " Path planning for Mobile Robot Using the Particle Swarm Optimization with Mutator Operator," Proceedings of the Third International Conference on Machine Learning and Cybernatics, Shaghai, August 2004, pp. 2473-2478.

[3] Yoram Keron, Johann Borenstein, "Potential field methods and their inherent limitations for mobile robot navigation," Conf. Robot Automation, California, pp. 1398-1404, April, 1991.

[4] Ma Zhao-qing, Yuan Zen-ren, "Real time navigation and obstacle avoidance based on grids method for fast mobile robot," Robot, No. 6, pp. 344-348, Nov., 1996.

[5] Habib M K, ASama H., "Efficient method to generate collision free paths for autonomous mobile robot based on new free space structuring approach," IEEE/RSJ International Workshop on Intelligent Robots and Systems IROS'91, Osaka, Japan, pp. 563—567, 1991.

[6] Shi Y. and Eberhart, R. C., Emperical study of particle swarm optimization," Proceedings of the 1999 Congress on Evolutionary Computation, Piscataway, NJ, pp. 1945—1950, 1999.