Boundary Following in Unknown Polygonal Environment Based on Fast Marching Method

Chia Hsun Chiang
Institute of Information Science, Academia Sinica
fetishist@gmail.com

Jing- Sin Liu
Institute of Information Science, Academia Sinica
liu@iis.sinica.edu.tw

***Abstract*-The fast marching method (FMM) is a powerful tool in the problem of shortest path planning. However it is limited to be used on a completely known terrain. In this paper, given only start and goal in an unknown polygonal environment consisting only polygonal obstacles, we would utilize the technique of boundary following to develop a boundary following FMM (BFFMM) that could be applied well to an unknown terrain. Comparative simulation results in different environments with single or multiple convex and/or concave polygons show that the efficiency of BFFMM outperforms FMM.**

I. INTRODUCTION

With the progress of the robot technology, more and more tasks could be processed by the robot automatically. One important part of the robotic applications is the autonomous navigation in terrains from a start position to a goal position by following a Euclidean distance shortest path. The field of robot path planning and navigation has been extensively studied by a lot of researchers for decades, and developments and evolution of plenty of various works, for example roadmap, cell decomposition, potential field methods, and constrained optimization techniques have been delivered in known static environments. In an environment in which the robot does not have a priori knowledge of the environment, sensor-based reactive navigation is widely used for robot navigation, where vision, sonar, laser range finder, ultrasonic sensors or their fusion are used. This method is suitable for local obstacle avoidance, since the goal may be out of the range of the sensors. [7] formulated the path planning problem in uncertain environment as an adaptive optimal control of Markov decision process. A survey of earlier work on path planning in unknown environment is given in [3].Among them, the algorithms of Bug family are well-known robot navigation approach in an unknown 2D environment. Early work on this class of algorithm could be traced back to Shannon, Sutherland, Lumelsky, and so on [3]. Nowadays a variety of Bug-like algorithm has being studied and proposed, such as Bug1, Bug2, Alg1, Alg2, DistBug, LeaveBug etc.. Knowing only the target position that is specified by distance and direction relative to known start position of the robot, Bug family algorithms operate switching between two modes: (i) to move toward the goal and (ii) move along the borders of obstacles. In the first, the robot moves toward the target along the straight line until it reaches the target point or encounters the obstacle. If the robot encounters the obstacle, it follows its boundary until some leaving conditions are satisfied. Then the robot continues moving toward the target. The robot either reaches the target position or reports unreachable when there is no way connecting the start and the goal. The comparisons of the path length generated by some members in Bug family could be found in [1].

Fast marching method (FMM) has been applied to path planning problems recently [4], [8]-[11]. FMM, which could find the shortest path in a given completely known map [5], is employed as the path planner. As shown in [5] and other literature (e.g. [11]), FMM can generate a good curvature-control, generally smooth and short path that is a numerically consistent approximation to the true shortest continuous path as the resolution of map is finer and finer. A disadvantage of FMM is that its area of search is large. The wider area of search, the more time of computation.

This paper will propose a simple method of practical value for mobile robots to explore an initially unknown terrain to build its own map, which is a subset of global map (i.e. a partial map) through local sensor system, and to perform global path planning on this partial map with fast marching method (FMM). This paper will focus on boundary following [6], where a mobile robot moves along the borders of obstructed obstacles it encounter. The currently implemented version of navigation is inspired from the procedure of Bug algorithm. The way of exploring the unknown environment is through following the boundary of polygonal obstacles, so we call the new method as boundary following FMM (BFFMM). Our simulation shows that the efficiency of FMM is improved greatly at the cost of only a little increase of the path length.

This paper is organized as follows. Section II describes the method to explore an unknown environment aiming to build a robot's own partial map connecting start and goal, and

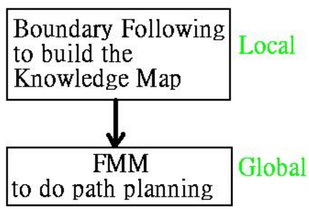


Fig. 1. The scheme of BFFMM combining goal-directed exploration to build a robot's own map and FMM path planning on the obtained partial map

introduces FMM for path planning. In section III, we present two polygonal environments to test our approach. Finally, some discussions will be given in section IV.

II. METHODS OF GOAL-DIRECTED EXPLORATION AND NAVIGATION

We make the following assumptions. The robot moves in a 2D polygonal environment, arbitrarily populated with a set of static polygonal obstacles. The robot is modeled as a point (by enlarging the obstacles to account for the robot size). The robot does not have a priori knowledge of the environment, except the location of START and GOAL. However, the robot is equipped with a sensory system capable of exploring the environment by measuring the distance and direction to an obstacle within the visibility range of the sensors. More specifically, the robot is equipped with a compass to detect the direction connecting the robot and the goal. It can record the coordinate of the hit point on which the robot first considers that it bumps the obstacle when the distance between the robot and the obstacle is below some threshold value. The aim is to navigate the robot in the environment to reach the GOAL.

In this Section, we will introduce two building blocks of our scheme (See Fig. 1) for navigating the point robot in the unknown environment to the goal. The first thing for the robot is to build its own map via an exploration strategy, which is a partial map of the entire terrain. Given an unknown terrain, only the visible parts of the obstacles are represented by the sensory readings. A local obstacle avoidance system is used to avoid collision when an obstacle is encountered. Meanwhile, the robot would gather the map information around its neighborhood and record these data on a knowledge map. By the accumulation of the local data, the terrain would be partly built to obtain a partial map connecting start and goal as a global approximation of the entire terrain. Since our objective is navigation to a goal, this partial map is enough for path planning by fast marching method (FMM) to generate a shortest path from start to goal on it.

A. *Method of Navigation for Building Partial Map*

The following is the basic steps of our approach:

● *Goal Chasing Mode*

1.) From the start position, the robot move directly towards

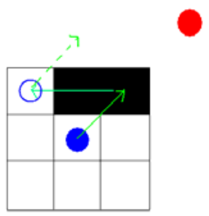


Fig. 2(a). Illustration of the method to implement boundary following. Where the blue dot represents the robot; the red dot represents the goal The representative point of each grid is the centre of grid.

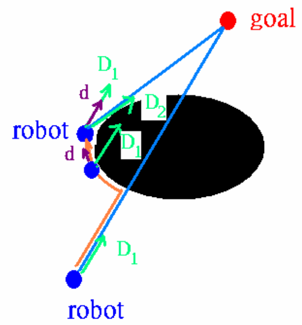


Fig. 2(b). Illustration of the direction refreshing

the goal G by following the line connecting START-GOAL, until either:

a) The goal is reached. Algorithm stops.
b) An obstacle is encountered. Record the hit point H and the direction to the goal D. Go to step 2, assuming a local obstacle avoidance by boundary following is enabled.

● *Boundary Following Mode*

2.) Perform counter-clockwise circumnavigation and step forward in the direction of the newly found free space direction d.
3.) Repeat the step 2 until the direction d equal to G. Go to step 1.
4.) If re-meet the hit point H for the first time, then change the circumnavigation direction from counter-clockwise to clockwise in step 2. Perform step 2-3 again.
5.) If re-meet the hit point H for the second time, then report the target is unreachable. Algorithm stop.

The number of directions the robot can traverse in a grid map is 8(see Fig. 2(a) for illustration). We use the sign of the difference between two 2-dimentional coordinates of the grids to assign one of the eight directions. As exploring, the robot would survey the eight neighboring grids around the grid that the point robot currently occupied. In boundary following mode, the robot records the direction D between the entering point and the goal, and in each step measures the location of the goal relative to the robot at that time, and search for a movable direction to go along. The direction of search could be clockwise or counter-clockwise. Because we have divided whole orientation into eight parts, the relative direction d between the goal and the robot would fall into one of these eight parts. While the d parallel to the D, the robot exits the boundary following mode and enters a new mode. By this procedure, the built partial map is less time-consuming to construct, requires less amounts of memory storage, and is well-suited for use with boundary following scheme.

Unlike the case in Bug2 algorithm, our method does not keep a fixed start-goal line. As we can see from figure 2(b), the initial start-goal condition only gives us the initial direction to guide our robot to move forward. Later the robot would refresh the robot-goal direction and adjust its orientation to advance. To gather more information about the boundaries of the environment, after the robot toured from the start to goal, we permute the start and goal and run the algorithm again so as to cover more area traversed by the robot.

B. Fast Marching Method

Isotropic Eikonal equation is the following first order PDE

$$\|\nabla u(x)\| = \tau(x), x \in \Omega$$
$$u(x) = q(x), x \in \partial\Omega \qquad (1)$$

where τ is the speed (or cost) function that is a function of local position, Ω is the domain, $\partial\Omega$ its boundary. As implicated in the robotic navigation, the formulation can be interpreted as isotropic front propagation or isotropic min-time optimal trajectory problems. In the control-theoretic context, the characteristic lines of equation (1) can be interpreted as the optimal trajectories.

The key feature of Eikonal equation (1) is that their characteristic lines coincide with the gradient lines of the viscosity solution $u(x)$; this allows the construction of single-pass Fast Marching Method (FMM) that solves the Eikonal equation in a stable and consistent manner. Under 4-connectivity 2D grid condition, using the first order finite difference upwind scheme to approximate the Eikonal equation (1) in continuous domain yields a quadratic equation of u to estimate actual (geodesic) distances in discrete domain:

$$\left(\max\left\{u-U_{i-1,j}, u-U_{i+1,j}, 0\right\}\right)^2 + \left(\max\left\{u-U_{i,j-1}, u-U_{i,j+1}, 0\right\}\right)^2 \quad (2)$$
$$= \tau_{i,j}^2$$

where we assume the Cartesian grid with unit grid spacing; $U_{x,y}, \tau_{x,y}$ denotes the distance and speed at grid coordinate (x, y), respectively. If the discriminant of quadratic equation (2) is larger than zero, we solve the larger u solution of the quadratic equation; else we set the value of u with the following equation instead:

$$u = \tau_{ij} + \min\{U_{i-1,j}, U_{i+1,j}, U_{i,j-1}, U_{i,j+1}\} \qquad (3)$$

In the following, we present the algorithm to incarnate the front propagation of FMM.

FMM Algorithm:

Definition. The nodes in the grid map are classified into three categories

1 *Alive (or Known)* is the set of all grid points at which the distance value u has been reached and will not be changed;

(a) *Trial* (or *Near*, *Narrow band*) is the set of next grid points to be examined/readjusted and for which an estimate U of u has been computed using equation (2) or (3) only from *Alive* points. U may be changed later;

2 *Far* is the set of all other grid points for which U is not yet computed;

Initialize

4 Alive points: Let Alive be the set of starting grid point p_0. Set

$$U(p_0) = u(p_0) = 0$$

5 Narrow Band points: Let Narrow Band be the set of all grid points neighboring to p_0 with initial values $U(p) = \tau(p)$;

6 Far Away points: Let Far Away be the set of the rest of the grid points; set their value to ∞;

Marching Forwards Loop

1 Let $p = (i_{\min}, j_{\min})$ be the Trial point with the smallest distance U;

2 Move it from the *Trial* set to the *Alive* set (i.e. $U_{i_{\min}, j_{\min}} = u_{i_{\min}, j_{\min}}$ is frozen);

3 For each nearest neighbor (i, j) (4-connectivity in 2D) of $(i_{\min}, j_{\min})$:

(b) If (i, j) is Far Away, add it to the Trial set and compute a first estimate U of u using equation (2) or (3);

(c) If (i, j) is Trial, update the distance $U_{i,j}$ using equation (2) or (3);

Note that only the Alive points are considered to solve Eikonal equation with upwind finite difference scheme. We examine neighbors of the point being examined and then select the suitable Alive ones. The movement of the front, opposite to the gradient, will point outward from the selected Alive point. As shown in the algorithm, FMM sweeps the front ahead in an upwind fashion by considering a set of points in narrow band around the existing front and march this narrow band forward, freezing the values of existing points and bringing new ones into the narrow band structure. The central idea of narrow band is to build an adaptive mesh around the propagation interface and perform computation only on these grid points. We make a tube containing all the points neighboring to the interface curve within some distance by calculating the signed distance function, points inside the frontier are with negative value and outside ones positive, and using that to select the points. The key is in the selection of which grid point in the narrow band to update with the help of heap operator. Using a min-heap structure for the Trial list, the algorithm computational complexity is $O(N \log N)$ where N is the number of grid points. An optimal path is generated by backtrack the points from goal to start via gradient descent by using the computed u values. The bend of the generated path would be affected by the gradient computation mask and resolution of the map.

III. SIMULATION RESULTS

Three environments shown in Figure 3 are used to test our approach. Figure 3(a) depicts the map originally created by Sankar, Figure 3(b) is the map with a single obstacle of zigzag boundary, and Fig. 3(c) is a rectangle with cavity. All maps are uniformly discretized into an evenly distributed 2D rectangular grid of nodes with binary values. The size of each map is 100×100 . We assume that the mobile robot, modeled as a point, can navigate along each of 8 directions in rectangular grid environment for exploration, and can move along arbitrary direction for path planning.

All the simulations are run with MATLAB on a PC with Intel Core 2 Duo 1.8 GHz microprocessor. For a quantitative comparison of BFFMM and FMM, we define two measures of performance. One is the percentage of area of search region of BFFMM in the exploration stage to build the map over the area searched by FMM. The other is the path length generated by BFFMM as a percentage of the shortest path generated by FMM between start and goal. The simulation results are shown in Fig. 4, 5, where the blue spot represents the start point and the red one the goal. The cyan and the magenta part in the figures is the trace of the navigation path. The yellow zone is the region that is searched by FMM, and the red line and green line are the final paths planned, respectively, by BFFMM and FMM. From the examples shown in the Figure 4-5, it is seen that the proposed method could effectively reduce the search area of FMM. By goal-directed exploration, the robot which does not have its own map initially could hopefully establish a partial map, which is a channel connecting the start and end position and the path generated by BFFMM is bounded inside this channel.

In order to display the performance of our approach in a whole, we randomly select 10 pairs of start and goal location on each of the two maps, and run the simulation with BFFMM and original FMM. In the result of that the algorithm of navigation is primitive, sometimes the start-goal pair generated may make the robot build a knowledge map with start and goal being unconnected, and the path planning could not be done to reach the goal. In addition, it may occur that the randomly generated start-goal pair would locate on a straight line without obstacle impeding them. So we threw off improper data and keep suitable one. Figure 6 and figure 7 are the statistic results

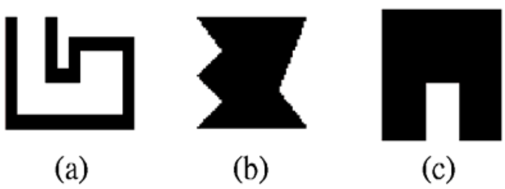
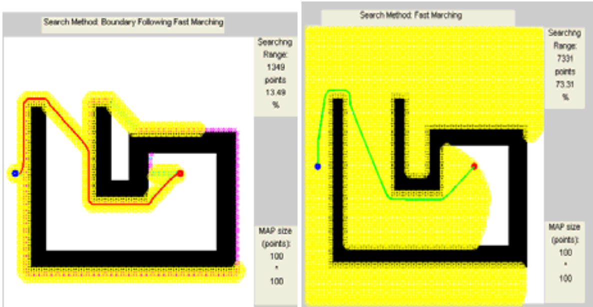
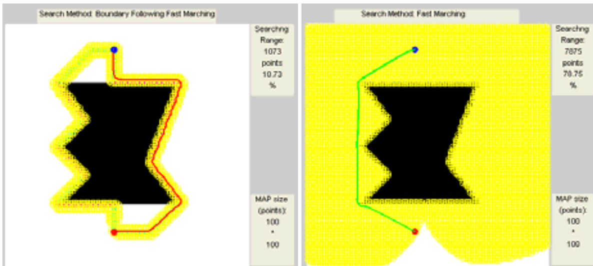


Fig. 3. The environment we use to do simulation. (a) The Sankar's terrain. (b) The zigzag terrain (c) The cavity terrain



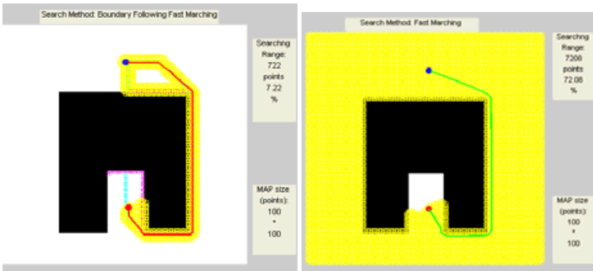
(a) (b)

Fig. 4. The comparison of search range in Sankar's terrain, where the yellow part represents the search range. (a) BFFMM and the generated path (b) FMM and the shortest path



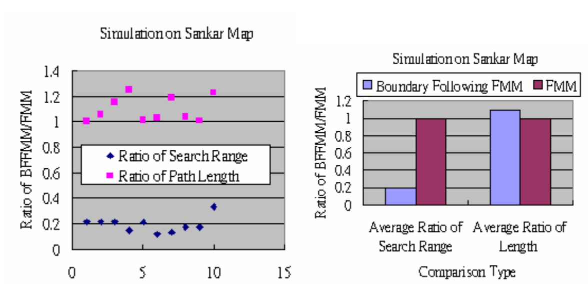
(a) (b)

Fig. 5. The comparison of search range in zigzag terrain, where the yellow part represents the search range. (a)BFFMM and the generated path (b)FMM and the shortest path



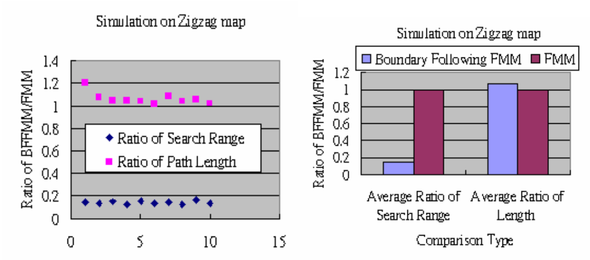
(a) (b)

Fig. 6. The comparison of search range in cavity terrain, where the yellow part represents the search range. (a) BFFMM and the generated path (b) FMM and the shortest path



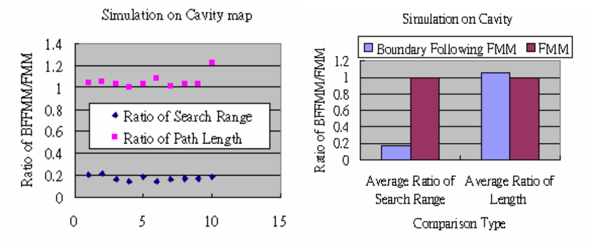
(a) (b)

Fig. 7. The comparison of search range and path length between BFFMM and FMM in Sankar's terrain in 10 runs of randomly generated feasible (start, goal) pair. We normalize the performance of FMM to one as standard reference. (a)Raw data of performance. (b)The average performance



(a) (b)

Fig. 8. The comparison of search range and path length between BFFMM and FMM in zigzag terrain in 10 runs of randomly generated feasible (start, goal) pair. We normalize the performance of FMM to one as standard reference. (a)Raw data . (b)The average performance.



(a) (b)

Fig. 9. The comparison of search range and path length between BFFMM and FMM in cavity terrain in 10 runs of randomly generated feasible (start, goal) pair. We normalize the performance of FMM to one as standard reference. (a)Raw data . (b)The average performance.

of the comparison on the area of search and the length of path produced by BFFMM and FMM. We set the performance of original FMM as 1, and use it as the reference to compare with the performance of improved one, BFFMM.

Assuming known map, FMM could generate a shorter path than the path generated by BFFMM, and the ratio of path lengths by BFFMM over FMM is no less than 1. Referring to Figure 6(b) and 7(b), the increase of path length is about 10 percent; however, the decrease of search area is over 70 percent. In general, BFFMM generates a boundary following path consists of more turns that may be not easily followed by the mobile robot at improved efficiency on computation.

The principle of boundary following exploration is to track the boundary of obstacles as possible. The influence of the boundary outside the current boundary is tested by the map with multi-obstacles as shown in figure 10. The experiment result is shown in figure 11-(a) and 11-(b). From figure 11-(a), goal-directed exploration takes a route of following the outer contour of the obstacles. This phenomenon is resulted from that the number of turning direction in exploration is only eight, and the robot only selects to circumvent the boundary of obstacles clockwise or counter-clockwise. This makes the robot could not find the possible shorter route in the inner free space among obstacles and the serrate profiles of the obstacles make the robot follow a path with longer length. Compared to BFFMM, the case of FMM in figure 11-(b) length is large and in general the ratio of path length between BFFMM and FMM

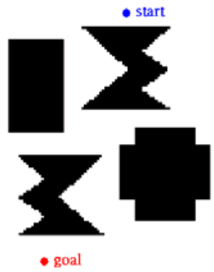
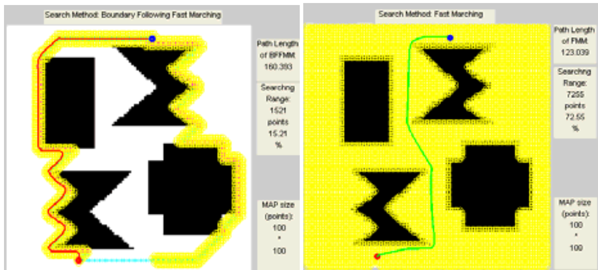


Fig. 10. The environment with multi-obstacles to test the effect of the boundary of the obstacles on the path produced.



(a) (b)

Fig. 11. The comparison between the paths produced by BFFMM and FMM. (a) The red line is the path produced by BFFMM. (b)The green line is the path produced by FMM.

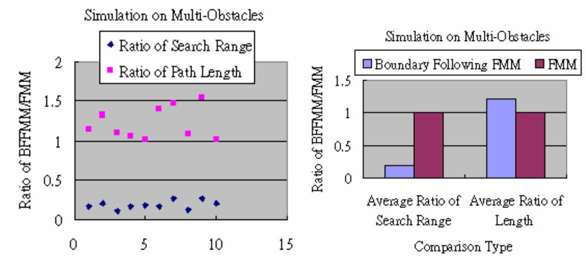


Fig. 12. The comparison of search range and path length between BFFMM and FMM in a map with many obstacles in 10 runs of randomly generated feasible (start, goal) pair. We normalize the performance of FMM to one as standard reference. (a)Raw data . (b)The average performance.

is larger than foregoing results. The average ratio of path length increases by about 10 percent. However the average ratio of search range still holds at the same level. This elucidates that our method is distinctly robust in keeping small search range under multi-obstacles environment.

All the BFFMM above are implemented with counter clockwise (CCW) search precedence. As a comparison, we also implement the BFFMM with clockwise (CW) search precedence. Refer to figure 13.From figure 13(b) and 13(c), the performance is a little better than figure 11, but this does not mean that the clockwise search precedence is better than counter clockwise search precedence. The environment setting such as the laying and shape of obstacles is an important limiting factor in the performance.

IV. DISCUSSIONS

Based on the performed simulations, we could conclude that the method we proposed extends the applicability of FMM from globally known static grid environments to unknown static grid environments. It is an effective way to reduce the computational cost of path planning by FMM with a little degradation of path quality. The path length and the area of search may be varied by the environment and the location of start-goal pair, in general the efficiency is improved by boundary following. Based on the partial map built in goal-directed exploration stage with limited sensor system of the robot, a boundary following path could be planned successfully by FMM in an initially unknown terrain. Notwithstanding the significant efficiency improvement, BFFMM cannot be guaranteed to succeed in all situations. If this partial map building procedure could not establish a knowledge map with start and goal being connected, then the FMM path planning would be unable to proceed in this built partial map. We could believe that the algorithm of navigation could succeed to connect the start and goal if all the obstacles are convex, while it may fail as non-convex obstacles exist. In the future, we would study on the reform of the algorithm of navigation.

ACKNOWLEDGMENT

This research was supported by National Science Council under contract NSC 96-2221-E-001-018-MY2.

REFERENCES

[1] James Ng, Thomas Braunl, “Performance Comparison of Bug Navigation Algorithm,” *Journal of Intelligent and Robotic Systems,* vol. 50, pp. 73-84, September 2007.

[2] Ricardo A. Langer, Leandro S. Coelho, and Gustavo H. C. Oliveira, “*K-Bug, A new bug approach of mobile robot path planning*”, *16th IEEE International Conference on Control Applications, Part of IEEE Multi-conference on Systems and Control, Singapore* , October 2007, pp.403-408.

[3] Nageswara S.V. Rao, Srikumar Kareti Weimin Shi, S. Sitharama Iyengar, “Robot Navigation in Unknown Terrains: Introductory Survey of Non-Heuristic Algorithms,” *Technical Report*, Oak Ridge National Laboratory, July 1999.

[4] J. A. Sethian, Level Set Methods, Evolving interfaces in geometry, fluid mechanics, computer vision, and materials science, Cambridge University Press, 1999.

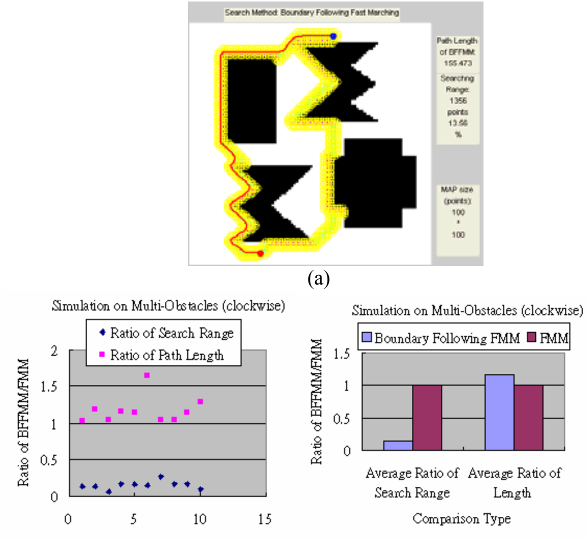
[5] Chia Hsun Chiang, Po Jui Chiang, Jerry Chien-Chih Fei, Jing Sin Liu, “A Comparative Study of Implementing Fast Marching Method and A* Search for Mobile Robot Path Planning in Grid Environment: Effect of Map Resolution,” *IEEE Workshop on Advanced Robotics and its Social Impacts,* December 2007, pp. 7-12.

[6] Shuzhi Sam Ge, Xuecheng Lai, Abdullah Al Mamun, “Boundary Following and Globally Convergent Path Planning Using Instant Goals” , *IEEE Trans. on Systems, Man, and Cybernetics, Part B: Cybernetics,* Vol. 35, No. 2, April 2005.

[7] S. Chakravorty and J.L. Junkins, “A methodology for intelligent path planning,” *IEEE International Symposium on Intelligent Control*, pp.592-597, 2005.

[8] Santiago Garrido, Luis Moreno, Mohamed Abderrahim, Fernando Martin, “Path Planning for Mobile Robot Navigation using Voronoi Diagram and Fast Matching”, *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Beijing, China, October 9-15, 2006

[9] Santiago Garrido, Luis Moreno, Dolores Blanco, “Voronoi Diagram and Fast Marching applied to Path Planning”, *Proceedings of the 2006 IEEE International Conference on Robotics and Automation*, Oriando, Florida, May 2006



(b) (c)

Fig. 13. Performance comparison between FMM and BFFMM with clockwise search order in Multi-Obstacles environment.. (a) The red line is the path produced by BFFMM with clockwise search order. The start and goal is the same as in figure 11. (b) The raw data (c) The average performance .

[10] Santiago Garrido, Luis Moreno, Dolores Blanco, Fernando Martin, “Log of the inverse of the Distance Transform and Fast Marching applied to Path Planning”, *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Beijing, China, October 9-15, 2006

[11] Clement Petres, Yan Pailhas, Pedro Patron, Yvan Petillot, Jonathan Evans, David Lane, ”Path Planning for Autonomous Underwater Vehicles”, *IEEE Trans. Robotics*, vol. 23, no. 2, pp. 331-341, Apr. 2007

APPENDIX

Table I lists the set of the start and the goal pairs randomly generated in our simulation, where the number couples are the coordinates of the start and the goal. On the 100×100 grid map, $(1,1)$ is the lower left corner and $(100,100)$ is the upper right corner.

TABLE I
10 RANDOMLY GENERATED PAIRS OF START AND GOAL FOR SIMULATIONS

Sankar		Zigzag		Cavity		Multi-Obstacles (CCW)		Multi-Obstacles (CW)	
Start	Goal	Start	Goal	Start	Goal	Start	Goal	Start	Goal
(64,11)	(29,56)	(35,91)	(38,12)	(64,11)	(96,96)	(17,98)	(96,50)	(50,50)	(35,91)
(96,50)	(43,92)	(64,11)	(96,97)	(17,98)	(96,50)	(66,5)	(86,94)	(11,15)	(95,96)
(80,96)	(66,5)	(17,98)	(96,50)	(81,16)	(43,92)	(11,83)	(96,5)	(38,63)	(79,10)
(75,40)	(71,5)	(81,11)	(43,92)	(80,96)	(66,5)	(77,80)	(20,50)	(36,94)	(88,56)
(29,6)	(11,83)	(80,96)	(66,5)	(11,83)	(96,5)	(46,65)	(13,51)	(24,18)	(24,45)
(20,50)	(46,65)	(79,40)	(66,18)	(45,39)	(77,80)	(97,35)	(52,71)	(50,35)	(96,93)
(72,76)	(29,69)	(20,50)	(72,76)	(20,50)	(72,76)	(90,96)	(56,15)	(7,75)	(29,43)
(76,27)	(52,71)	(66,18)	(13,51)	(13,51)	(96,75)	(16,27)	(62,48)	(67,55)	(71,67)
(62,48)	(36,84)	(97,35)	(59,24)	(90,96)	(56,15)	(9,7)	(54,79)	(5,57)	(89,68)
(43,6)	(91,95)	(93,36)	(21,26)	(16,27)	(85,27)	(94,15)	(58,48)	(20,38)	(47,99)