International Journal of Automation and Computing 7(3), August 2010, 310-316
DOI: 10.1007/s11633-010-0508-6

Path Planning Approach in Unknown Environment

Ting-Kai Wang Quan Dang Pei-Yuan Pan
Faculty of Computing, London Metropolitan University, London N7 8DB, UK

Abstract: This paper presents a new algorithm of path planning for mobile robots, which utilises the characteristics of the obstacle border and fuzzy logical reasoning. The environment topology or working space is described by the time-variable grid method that can be further described by the moving obstacles and the variation of path safety. Based on the algorithm, a new path planning approach for mobile robots in an unknown environment has been developed. The path planning approach can let a mobile robot find a safe path from the current position to the goal based on a sensor system. The two types of machine learning: advancing learning and exploitation learning or trial learning are explored, and both are applied to the learning of mobile robot path planning algorithm. Comparison with A^* path planning approach and various simulation results are given to demonstrate the efficiency of the algorithm. This path planning approach can also be applied to computer games.

Keywords: Path planning, fuzzy reasoning, unknown environment, mobile robot, learning algorithm.

1 Introduction

Path planning has been a focus of research in the field of mobile robot[1, 2] and interactive virtual environment, such as computer games[3−5] in recent years. Depending on whether the environment model is known or not, the path planning is usually classified into two categories. The first is called path planning based on the environment model as the mobile robot knows all information about the environment or working space; the second is called path planning based on sensors as the mobile robot does not know (at most partially know) the information of environment or working space.

There have been some approaches of the first kind of path planning, such as the map-based navigation approach[6, 7], environment database navigation approach[8], etc. For the second kind of path planning, currently, there are various guidance methods, including the virtual force field method[9], the edge detection method[10], the wall following method[11], the fuzzy reasoning, the neural network approach, and the genetic approach[12−16], etc. There are some path planning approaches between the first and the second, such as A^* search algorithm[17]. In this paper, we consider the second kind of path planning.

We present an algorithm for path planning that utilises the characteristics of the obstacle border and applies fuzzy logical reasoning. It addresses some of the features and weaknesses of the second path planning approaches. The environment topology or working space is described by the time-variable grid method that can be further described by the moving obstacles and the variation of path safety. The mobile robot will then find a safe path from the current position to the goal according to the algorithm. The learning algorithm for mobile robot path planning is also discussed in this paper. The remainder of the paper is organised as follows. Section 2 introduces the time-variable grid model and convex properties of an obstacle. The path planning algorithm is presented in Section 3. Section 4 explores the improvement of path planning, i.e., path planning learning method. The path planning examples, which illustrate the algorithm and comparison of our algorithm with A^* path planning algorithm are given in Section 5. Section 6 comes to the conclusion and discussions on the advantage and disadvantage of the algorithm.

Manuscript received January 1, 2010; revised March 31, 2010

2 Time-variable grid model and convex properties of an obstacle

2.1 Time-variable grid model

We use a grid model to describe the mobile robot working space (or environment). The grid model gives a view of the mobile robot environment as a two-dimensional plane that is divided into m equal parts in both x and y axes, resulting in m^2 grids. If there is an obstacle in a grid, the grid can be assigned a specified symbol or a specified colour, representing that the cost for the mobile robot to pass through the gird is infinite. A region without any obstacle is denoted as blank, indicating that the cost for the mobile robot to pass through it is one (or zero). According to the requirement on safety, time, etc., the cost of the mobile robot passing through the gird will be assigned a finite value in the range $[0, \infty]$. Because some obstacles can move, the value of the grid is changing with time, i.e., with the change of the position of an obstacle with time, the cost of the grid also changes with time. The safety cost of grids in the neighbourhood should also change with time. This is, generally speaking, the mobile robot can pass by the moving obstacle from its front, side, or back. However, the degree of danger from the front is greater than that from the side or from back, and it increases with increase of the mobile robot speed.

Let us take into account the factors of safety, time, etc., and describe them by a synthetic cost that is fuzzily represented by "too large", "large", "middle", "small", and "very small". The cost of the grid with an obstacle is denoted as "too large". The cost of the grid in front of the moving obstacle is denoted as "large". The cost of the grid in other sides of the moving obstacle is denoted as "middle". The cost of the grid neighbourhood of a static obstacle is denoted as "small". The costs for the remaining grids are

denoted as "very small". Therefore, an ideal route of the mobile robot should be through the regions designated as "very small".

The effect of the above assignment operation is to make an obstacle look bigger or grow, as shown in Fig. 1. We use the term "fuzzy configuration obstacle" to describe the grown obstacle. While the original obstacle is transformed into a fuzzy configuration obstacle, the mobile robot is treated as a grid or a point. If two configuration obstacles intersect, their union is treated as a single configuration obstacle[18].

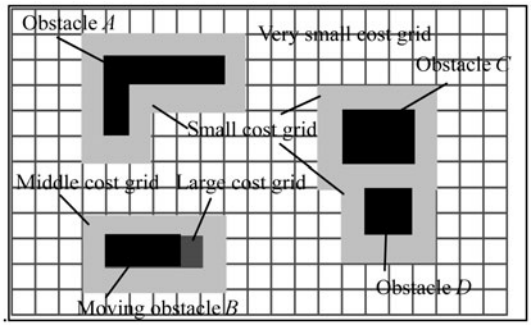


Fig. 1 The obstacle growing

2.2 Convex properties of an obstacle

An obstacle possesses topological properties. Here, our concern is the convex and concave properties of the boundary of the obstacle region. There is a rigorous definition of convex properties in mathematics[19, 20]:

Definition 1 (Convex domain). Let S be a subset of M (M is a whole space), and u and v be points in M. S is convex if, for any u, $v \in S$, $tu + (1-t)v \in S$ for all t such that $0 \leqslant t \leqslant 1$.

Definition 2 (Convex point). Let A be a point boundary of S. Point A is a convex point if there are two points B and C nearby (neighbourhood) point A on boundary S, making the subset BAC a convex domain.

It is necessary to define a point (or grid) to be a convex point (grid). In the Cartesian space, we can judge it by the derivative and the change rate of derivative on this point. With respect to the grid model describing the working space, it is not suitable to use the concept of derivative. However, similar to the convex point definition, we can define convex grid or convex boundary of an obstacle as follows:

Definition 3. If the boundary of an obstacle (or a grid in the boundary of an obstacle) is convex, the grid is one of the following:

1) The grid whose three parallel (in X and Y axes) and two opposite-angle neighbour grids are not occupied by an obstacle, as A shown in Fig. 2.

2) The grid whose two parallel (in X and Y axes) and three opposite-angle neighbour grids are not occupied by an obstacle, as B shown in Fig. 2.

3) The grid whose two parallel (in X and Y axes) and two opposite-angle neighbour grids are not occupied by an obstacle, as C shown in Fig. 2.

4) The grid whose four parallel (in X and Y axes) and four opposite-angle neighbour grids are not occupied by an obstacle, as D shown in Fig. 2.

In summary, if the gird has at least two parallel (in X and Y axes) and two opposite-angle neighbour grids that are not occupied by an obstacle, it is convex.

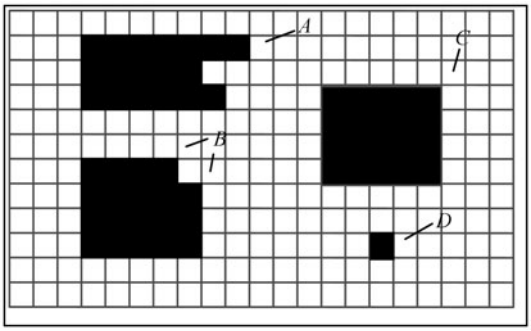


Fig. 2 Four kinds of convex points in grid model

3 Path planning algorithm

Suppose that there are a number of obstacles in the mobile robot working space, and the circumference of each obstacle is limited. We also assume that the mobile robot does not have any priori knowledge about the working space. The cost of the grid neighbourhood of mobile robot can only be found out by its sensor system (which should be adequate enough to stop the vehicle if necessary). The mobile robot should also be able to locate the positions of itself and the goal, and be able to move in any direction.

The route of a mobile robot can be expressed as a set of direct lines towards the goal and a set of curved lines around obstacles. The intersection point of the i-th direct line and the boundary of the obstacle are called the near-collision point, denoted as C_i. The direct line route is ended at this point, and the i-th curved route starts. The end point of the i-th curved line is called the end-collision point, denoted as L_i. The curved route ends at this point and then the $i+1$ direct line starts. The mobile robot moves around the obstacle in either a clockwise or anti-clockwise direction. This is determined by the topological properties of an obstacle relative to the mobile robot heading, and the position of the goal relative to the position of the mobile robot. The topological properties of the obstacle relative to the mobile robot heading can be classified into three kinds: horizontal, vertical, and non-horizontal and non-vertical, as shown in Fig. 3. Determination of the direction in which the mobile robot will move around the obstacle is given by Tables 1–3.

Table 1 Horizontal obstacle and heading

Conditions	Action
$x_g - x < 0,\ \ y_g - y > 0$	Clockwise
$x_g - x > 0,\ \ y_g - y > 0$	Anti-clockwise
$x_g - x < 0,\ \ y_g - y < 0$	Anti-clockwise
$x_g - x > 0,\ \ y_g - y < 0$	Clockwise

Table 2 Vertical obstacle and heading

Conditions	Action
$x_g - x < 0,\ \ y_g - y > 0$	Anti-clockwise
$x_g - x > 0,\ \ y_g - y > 0$	Clockwise
$x_g - x < 0,\ \ y_g - y < 0$	Clockwise
$x_g - x > 0,\ \ y_g - y < 0$	Anti-clockwise

Table 3 Non-horizontal and non-vertical obstacle relative to mobile robot heading

Conditions		Action
$x_g > x$, $y_g > y$ or	$\|x_g - x\| > \|y_g - y\|$	Anti-clockwise
$x_g < x$, $y_g < y$	$\|x_g - x\| > \|y_g - y\|$	Clockwise
$x_g > x$, $y_g < y$ or	$\|x_g - x\| > \|y_g - y\|$	Clockwise
$x_g < x$, $y_g > y$	$\|x_g - x\| > \|y_g - y\|$	Anti-clockwise

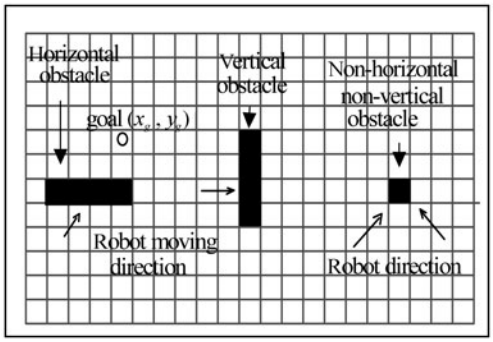


Fig. 3 Topological properties of obstacle for moving

Here, x_g and y_g are the horizontal and vertical position coordinates of the goal, x and y are the horizontal and vertical coordinates of the mobile robot′s current position, and $|x_g - x|$ is the absolute value of $x_g - x$.

In order to execute the algorithm, the mobile robot needs to remember the following:

1) All previous close-collision points, which is represented by a 2-dimensional array.

2) All previous out-collision points, which is represented by a 2-dimensional array.

3) All previous moving around directions at the close-collision points. Here, 0 is used to represent clockwise and 1 to represent anti-clockwise direction.

4) Mobile robot trajectory from the start position to the goal. These are required for mobile robot learning.

If the start position is S and the goal is G, then the path planning algorithm is described as follows, with initialisation of all memory set to zero.

Step 1. The mobile robot moves from S to G in a direct line route until one of the following conditions is satisfied:

1) if the mobile robot arrives at G;

2) if the mobile robot meets an obstacle.

Step 2. If the obstacle is moving, the mobile robot waits for the obstacle to pass, then go to Step 1; otherwise, go to Step 3.

Step 3. If the obstacle is static, this point is treated as the close-collision point and is put into memory. The moving around direction should be determined and recorded. Then, the mobile robot moves along the boundary of the obstacle in the determined direction:

1) If the mobile robot arrives at the goal G, end.

2) If the current position has been recorded before, examine the possibility of reaching the goal G and set a flag. If the flag is 0, then go to Step 3 3). If the flag is 1, since it is impossible to reach the goal G, end.

3) If the current position has been recorded before, then move in the direction opposite to the previous direction and set the flag as 1.

4) If the mobile robot position satisfies the following conditions, the current position is the end-collision point, and the mobile robot reverts to a direct line movement (go to Step 1). The current position is a convex grid (point) and has not been recorded before as a point on the mobile robot trajectory between the current position of the mobile robot and goal.

4 Improvement of mobile robot′s path planning

The ability to learn is an important sign of a mobile robot′s intelligence. Learning is an information processing process based on memory and knowledge. Because we have supposed that the mobile robot does not have any priori knowledge about the working space, it is difficult to ensure that the path planned is the best. However, the mobile robot can improve the path planning next time based on a combination of the previous paths and partial knowledge of the working space. The process of improvement is thus a learning process for mobile robot path planning.

Generally speaking, there are two basic types of machine learning. One is called here advancing learning, as this type of learning is a process of progress made step-by-step based on the knowledge that the mobile robot has acquired. This is analogous to a person working carefully (persistently) to improve a production. Most of the learning approach used today can be catalogued into this type. In this paper, the advancing learning of the mobile robot path planning is composed of two steps, as shown below:

1) If the flag at Step 3 3) in the above path planning algorithm is previously recorded as 1, this means that there is an unnecessary sub-path in the previous path planning. It should be cut out. This can be realised by taking the last moving around direction in the previous path planning as the moving around direction at the same point (grid) for next path planning.

2) Taking all out-collision points (grids) in the previous path planning as sub-goals, the mobile robot first plans the path to the sub-goal 1, then sub-goal 2, $\cdots$, and, finally, the real goal.

For this improvement for the mobile robot path planning, the original path planning algorithm does not need to change its path planning process. It only requires to add some logical reasoning, and this can be easily achieved. The path planning will certainly be better than the original one after this type of learning.

We refer to another type of learning as exploitation learning or trial learning. Because it takes different actions from the original rules, the improvement on the path planning is made by knowing more about the working space. This is analogous to an explorer in an unknown world. Although it is not sure if this path planning will be better than the previous one, it gives the mobile robot a chance to gain some information that is not known before. It is therefore

possible to improve path planning next time. Here, we give a simple approach to the exploitation learning.

To change the path in which the mobile robot moved around the obstacle in the previous path planning, Tables 4–6 will be used for the next path planning instead of Tables 1–3 used previously.

Table 4 Horizontal obstacle and heading

Conditions	Action
$x_g - x < 0,\ \ y_g - y > 0$	Anti-clockwise
$x_g - x > 0,\ \ y_g - y > 0$	Clockwise
$x_g - x < 0,\ \ y_g - y < 0$	Clockwise
$x_g - x > 0,\ \ y_g - y < 0$	Anti-clockwise

Table 5 Vertical obstacle and heading

Conditions	Action
$x_g - x < 0,\ \ y_g - y > 0$	Clockwise
$x_g - x > 0,\ \ y_g - y > 0$	Anti-clockwise
$x_g - x < 0,\ \ y_g - y < 0$	Anti-clockwise
$\mathrm{x}_g - x > 0,\ \ y_g - y < 0$	Clockwise

Table 6 Non-horizontal and non-vertical obstacle relative to mobile robot heading

Conditions		Action
$x_g > x$, $y_g > y$ or	$\lvert x_g - x\rvert > \lvert y_g - y\rvert$	Clockwise
$x_g < x, y_g < y$	$\lvert x_g - x\rvert > \lvert y_g - y\rvert$	Anti-clockwise
$x_g > x$, $y_g < y$ or	$\lvert x_g - x\rvert > \lvert y_g - y\rvert$	Anti-clockwise
$x_g < x$, $y_g > y$	$\lvert x_g - x\rvert > \lvert y_g - y\rvert$	Clockwise

Another approach is to use these two types of learning methods alternatively or incooperatively to create a better result for path planning. As the mobile robot does not know all the information about the working space, it is still difficult to ensure if the planned path is the best, but the path will certainly be much better than the previous one.

5 Simulation and comparison

In this section, we give examples to illustrate the new path planning algorithm. The working space is of 64×64 grids, and all obstacles in the working space are described by fuzzy-configuration obstacles such that the mobile robot can be viewed as a point in a grid. The start point and the goal point for the mobile robot are S and G, respectively.

The path planning algorithm is programmed in Matlab and Java on a PC. The working space is described by a 64×64 matrix E, the value of the element of the matrix E represents the cost with which the mobile robot passes through the grid; for example, if the i-th horizontal and j-th vertical grid is an obstacle, then the value of the element of matrix $E(i, j)$ is infinite (in a program, this is assigned a very large number, say, 10^{10}); if the n-th horizontal and m-th vertical grid is a free space, then the value of the element of matrix $E(n, m)$ is 0. The value of the element of matrix E that represents mobile robot start or goal grid is 1, and the value of the element of matrix E that expresses the mobile robot′s trajectory grid is changed to 1 (the values of these elements of the matrix were 0 before mobile robot passes). We can get the total cost of the mobile robot trajectory by summing up the elements of matrix E with value of 1, and draw the trajectory of the mobile robot in the working space, which comprises all elements in the matrix E with value 1.

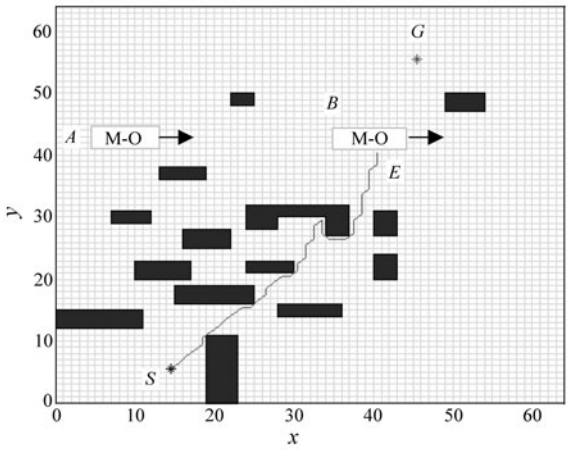
Example 1. A mobile robot is passing through a maze with a moving obstacle, as shown in Fig. 4, where M-O represents a moving obstacle, with the speed of 1 grid/s. The mobile robot speed is 2 grid/s. When the mobile robot reaches the position of E, the moving obstacle is at position B. According to the path planning algorithm, the mobile robot will wait at position E for 8 s until the moving obstacle passes by, as shown in Fig. 4 (a). Then, the mobile robot will continue to move towards the goal, as shown in Fig. 4 (b). Fig. 4 (c) gives the path planning result, where A is the position of the moving obstacle at the start time, and D is the position of the moving obstacle when the mobile robot reaches the goal.

Example 2. Fig. 5 gives another path planning example. Path (1) is the result of the basic path planning algorithm, path (2) is the result of using the advancing learning to improve mobile robot path planning, and path (3) in Fig. 6 is a path planning after exploitation learning, while path (4) in Fig. 6 is the result of path planning alternatively using these two types of learning methods.

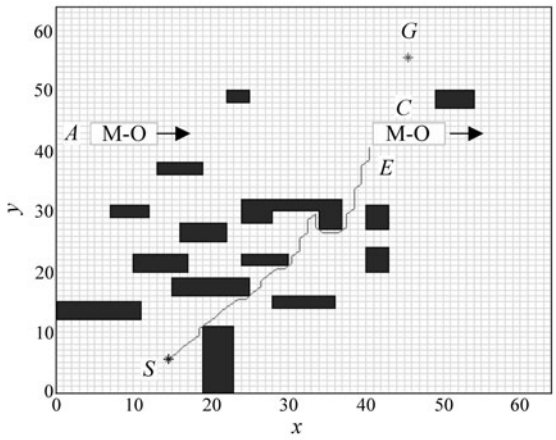
If the cost of the grid, which the mobile robot has passed is 1, then the total costs of the paths are shown as follows:

$$\begin{aligned}\text{Path}\,(1) &= 144\\ \text{Path}\,(2) &= 96\\ \text{Path}\,(3) &= 73\\ \text{Path}\,(4) &= 68.\end{aligned}$$

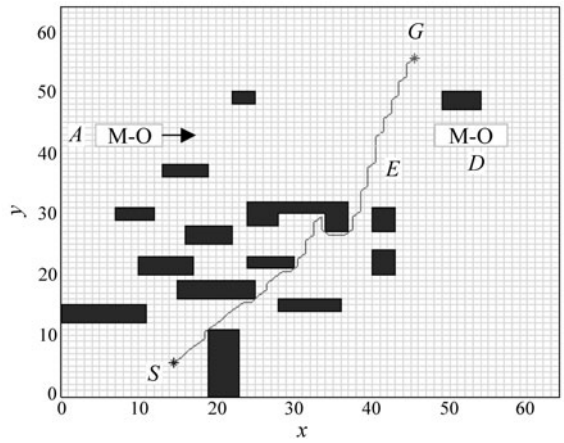
Clearly, path (4) is the best.



(a) The mobile robot waiting to avoid a moving obstacle



(b) The mobile robot starting to move again



(c) Mobile robot reaches the goal

Fig. 4 Mobile robot route through a maze with moving obstacle (M-O)

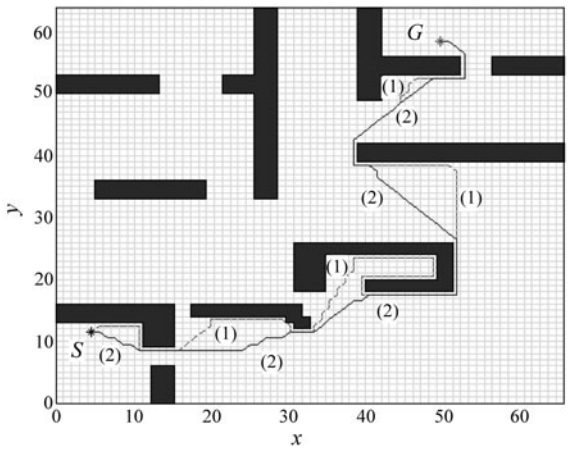


Fig. 5 Comparison of basic algorithm with advancing learning algorithm

From these examples, it is clear that the basic path planning algorithm can successfully plan a path from the start point to the goal. It is likely to find a better path when the learning process has been put into play.

Fig. 7 is a result of the classic A^* path planning. Obviously, the method could not find the path from the start point to the goal. The mission failed. After a long period of calculation, the modified A^* approach has found a path, as shown in Fig. 8. The total cost is 102. It is still worse than our path planning results paths (2)–(4).

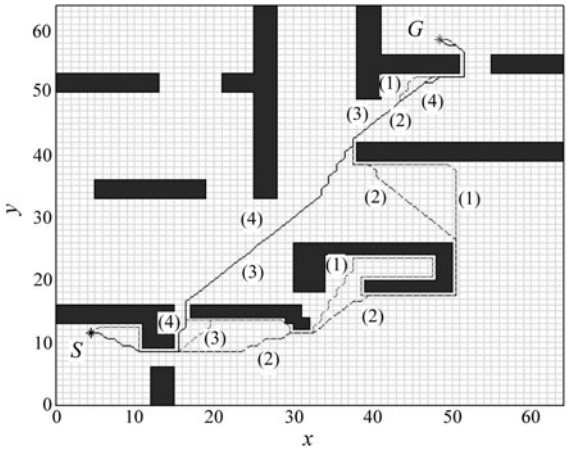


Fig. 6 Comparison of basic algorithm with advancing learning and exploitation learning

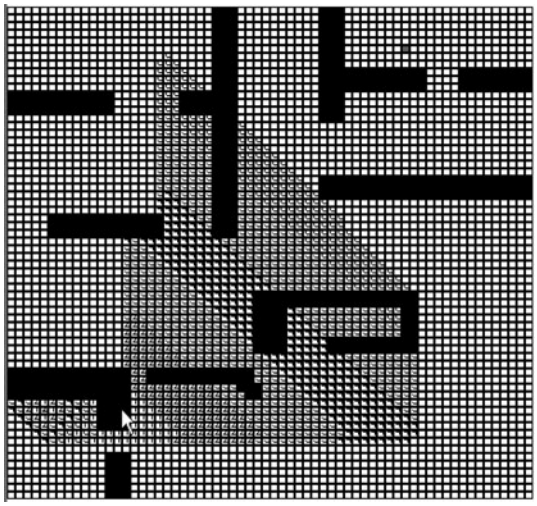


Fig. 7 Classic A^* path planning result

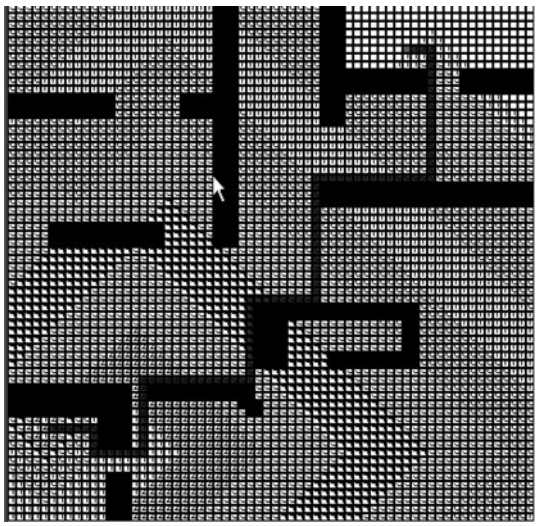


Fig. 8 Modified A^* path planning result

6 Conclusions

This paper presented an algorithm for path planning in an unknown environment based on the topological features of obstacles. Two types of path planning learning or improvement methods for mobile robot′s path planning were also explored. The simulation results have shown that the algorithm is efficient. Compared with other path planning algorithms in the unknown environment, this path planning algorithm has the following features:

1) Easy to program

According to the path planning algorithm, the route of an mobile robot consists of a set of direct lines towards the goal and a set of curved lines around obstacles, which are the simplest form of moving situation of the mobile robot. Furthermore, simple fuzzy reasoning based on several fuzzy rules is only required when the mobile robot is on near-collision and end-collision point; hence, the algorithm is simple and easy to implement.

2) Applicability to different environments

This path planning algorithm not only applies to a static environment (all obstacles are static in the working space), but also to an environment that has some moving obstacles.

3) Effective learning ability

Two types of machine learning are introduced, i.e., advancing learning and exploitation learning or trial learning. Combination of these two machine learning approaches greatly improves the mobile robot′s path planning capability.

4) Convergence

The convergence of this path planning algorithm has been proven.

References

[1] S. M. LaVall. *Planning Algorithms*, UK: Cambridge University Press, [Online], Available: http://msl.cs.uiuc.edu/planning, January 10, 2010.

[2] Q. J. Peng, X. M. Kang, T. T. Zhao. Effective virtual reality based building navigation using dynamic loading and path optimization. *International Journal of Automation and Computing*, vol. 6, no. 4, pp. 335–343 2009.

[3] L. Karamouzas, M. H. Overmars. Adding variation to path planning. *Computer Animation and Virtual Worlds*, vol. 19, no. 3–4, pp. 283–293, 2008,

[4] L. Karamouzas, R. Geraerts, M. Overmars. Indicative routes for path planning and crowd simulation. In *Proceedings of International Conference on the Foundation of Digital Games*, Orland, USA, pp. 113–120, 2009.

[5] D. Jung, H. Kim, J. Kim, K. Um, H. Cho. Efficient path finding in 3D games by using visibility tests with the A^* algorithm. In *Proceedings of Artificial Intelligence and Soft Computing*, Marbella, Spain, 2004.

[6] Y. Yagi, Y. Nishizawa, M. Yachida. Map-based navigation for a mobile robot with omnidirectional image sensor COPIS. *IEEE Transactions on Robotics and Automation*, vol. 11, no. 5, pp. 634–648, 1995.

[7] J. S. Gutmann, M. Fukuchi, M. Fujita. A floor and obstacle height map for 3D navigation of a humanoid robot. In *Proceedings of International Conference on Robotics and Automation*, Barcelona, Spain, pp. 1066–1071, 2005.

[8] T. Wang, Q. H. Mehdi, N. E. Gough. An integrated navigation system for AGV based on an environment database. *International Journal of Computers and Their Application*, vol. 6, no. 1, pp. 14–24, 1999.

[9] J. Borestein, Y. Koren. Real time obstacle avoidance for fast mobile robots. *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 19, no. 5, pp. 1179–1187, 1989.

[10] J. Borestein, Y. Koren. Obstacle avoidance with ultrasonic sensors. *IEEE Journal of Robotics and Automation*, vol. 4, no. 2, pp. 213–218, 1988.

[11] G. Bauzil, M. Briot, P. Ribes. A navigation subsystem using ultrasonic sensors for the robot Hilare. In *Proceedings of the 1st International Conference on Robot Vision and Sensory Controls*, Stratford-upon-Avon, UK, pp. 47–58, 1981.

[12] T. Kimura, T. Iokibe, H. Sasaki. Fuzzy path planning system for an autonomous vehicle. *Japanese Journal of Fuzzy Theory and System*, vol. 5, no. 4, pp. 626–636, 1993.

[13] C. D. Wu, Y. Zhang, M. X. Li, Y. Yue. A rough set GA-based hybrid method for robot path planning. *International Journal of Automation and Computing*, vol. 3, no. 1, pp. 29–34, 2006.

[14] I. J. Griffiths, Q. H. Mehdi, T. Wang, N. E. Gough. A genetic algorithm for path planning. In *Proceedings of IFAC Conference on Intelligent Components and Instruments for Control Applications*, France, pp. 531–536, 1997.

[15] T. R. Wan, H. Chen, R Earnshaw. Real-time path planning for navigation in unknown environment. In *Proceedings of Theory and Practice of Computer Graphics*, University of Birmingham, UK, pp. 138–145, 2003.

[16] T. Wang, Q. H. Mehdi, N. E. Gough. A human imitation approach to navigation and control of AGVs. *Chinese Journal of Advanced Software Research*, vol. 6, no. 2, pp. 137–143, 1999.

[17] N. D. Richards, M. Sharma, D. G. Ward. A hybrid A*/automaton approach to on-line path planning with

obstacle avoidance. In *Proceedings of AIAA 1st Intelligent Systems Technical Conference*, Chicago, Illinois, USA, pp. 20–22, 2004.

[18] T. Lozano-Pérez, M. A. Wesley. An algorithm for planning collision free paths among polyhedral obstacles. *Communications of the ACM*, vol. 22, no. 10, pp. 560–570, 1979.

[19] L. Råde, B. Westergren. *Mathematics Handbook for Science and Engineering*, Springer, 2004.

[20] N. Bourbaki. Topological vector spaces, Chapters 1–5, *Elements of Mathematics*, Springer, 1987.

Ting-Kai Wang received the first degree in automatic control from the Department of Mathematics of Zhongshan University, PRC in 1982, and the Ph. D. degree in computing from University of Wolverhampton, UK in 1998. He was a lecturer and associate professor at Department of Computer and Automation, Chongqing University for over ten years. He was also engaged as a postdoctoral research fellow in the School of Electronic Engineering, University of Surrey, UK in 1999 and the Centre for Virtual Environment, Information System Institute, University of Salford, UK in 2000. Currently, he is a senior lecturer in School of Computing, London Metropolitan University, UK. He won a Science and Technology Award from the State Education Commission of China in 1992, the Progress in Science and Technology Award from Chongqing Municipal Government in 1991, the Excellent Software Award from Sichuan Provincial Council of Science and Technology in 1991, the Achievements of Computer Development and Application Award from Chongqing Municipal Council of Science and Technology in 1989, and a number of the best paper awards from International Conferences. He is a fellow of the UK Higher Education Academy (FHEA).

His research interests include artificial intelligence and its application, virtual environments, navigation and control of mobile robots, system modelling and simulation, fuzzy systems, and control system.

E-mail: t.wang@londonmet.ac.uk

Quan Dang received the first degree in economic cybernetics from the State University of Management, Moscow, Russia in 1983, and the Ph. D. degree in computing from London South Bank University, UK in 1998. From 1984 to 1992, he was a researcher, then the head of Information Systems Research at the Institute of Information Technology, National Centre for Scientific Research of Vietnam, Hanoi, Vietnam. Currently, he is a senior lecturer at the Faculty of Computing, London Metropolitan University, UK. He received research scholarship and fellowship awards from London Metropolitan University, London South Bank University, and bodies of United Nations, the Vietnamese, French and British governments. He is a member of the British Computer Society (MBCS) and fellow of the UK Higher Education Academy (FHEA).

His research interests include system modelling and engineering, object technologies, and computer science education.

E-mail: q.dang@londonmet.ac.uk

Pei-Yuan Pan received the B. Sc. degree in computer science from Changsha Institute of Technology, PRC in 1982, and the Ph. D. degree in computing from Glasgow Caledonian University, UK in 1999. In 1991, he was promoted to an associate professor from lecturer at Changsha Institute of Technology. He worked as a senior visiting scholar at Brunel University, UK in 1993, post-doctoral research associate at Liverpool University in 1999 and lecturer at the University of Gloucestershire in 2001. Currently, he is a senior lecturer in the Faculty of Computing at London Metropolitan University, UK.

He received six national research awards: one State Award (second-class) of the Progress in Science and Technology, and five awards (three first-class and two second-class) from ministries of Chinese government, from 1988 to 1995.

His research interests include digital/global manufacturing, web applications, embedded systems, and application of artificial intelligence (AI) technologies.

E-mail: p.pan@londonmet.ac.uk (Corresponding author)