A Path Planning Method for Micro Robot Soccer Game[1]

T. H. Lee
Centre for Multimedia Signal Processing, Department of Electronic and Information Engineering, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong
thlee@eie.polyu.edu.hk

H. K. Lam
Centre for Multimedia Signal Processing, Department of Electronic and Information Engineering, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong
hkl@eie.polyu.edu.hk

F. H. F. Leung
Centre for Multimedia Signal Processing, Department of Electronic and Information Engineering, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong
enfrank@polyu.edu.hk

P. K. S. Tam
Department of Electronic and Information Engineering, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong
enptam@hkpucc.polyu.edu.hk

Abstract – This paper describes the design and development of a path planning method for Micro Robot Soccer Tournament (MIROSOT). Every object in the stadium is given a working layer which is decomposed into a fixed number of small areas. We propose to assign values to different small areas of the layer according to the object's movement. The final path is obtained by combining the small areas of the layers and getting the path with the smallest values. Since the host computer uses only a simple equation to calculate the value of each small area, this algorithm requires only a low computing power, and the speed of path generation is very fast. A simulation program has been developed to test the algorithm's performance.

I. Introduction

Path planning is one important topic in the problem of robot navigation. In this paper, a path-planning method is proposed which is to be applied to wheeled mobile robots (WMRs) for soccer games in a 2-dimensional area. In a robot soccer game environment, a video camera captures the stadium image and the host computer extracts the locations of the home soccer robots, the opponent soccer robots and the ball. The target position of each home robot is determined based the game strategy. The detailed operation of the robot soccer game is shown in [1]. Once a target position is given, our proposed method will plan the optimal path for the soccer robot to move to the target position without colliding with other robots.

Many methods of path planning have been reported. The simplest method is to generate a straight-line path connecting the source and destination points [2]. This method will fail if obstacles are present in the generated path. Many researchers develop methods for various applications. The roadmap method [3] involves pre-processing of the known environment, making it suitable for navigation in a static environment. Cell decomposition [4] method is based on the idea of a product operation defined on the cells in a decomposition of a two-dimensional free space. It also works on a static environment, and the complexity of the path-planning depends on the decomposition method, the number of cells and the algorithm of sorting the cells. Potential field method [5] involves defining a potential function and solving for the gradient of this function. It will take a long computational time although it can be applied in a dynamic environment to give an optimally planned path. The method proposed in this paper can work in a dynamic environment while a low computation power is demanded thanks for its simplicity. This paper is organized as follows. The working principles are described in Section II. The path-planning algorithm with examples for explanations are given in Section III. Simulation results verifying the algorithm and displaying the small area values of the stadium are given in Section IV. A conclusion will be given in Section V.

[1] The work described in this paper was substantially supported by a Research Grant of The Hong Kong Polytechnic University (project number G-V764).

II. Working Principle

The stadium of the micro robot soccer tournament (MiroSot) is divided into a number of small areas, and values will be assigned to each of them. Every object (a home robot, an opponent robot, a target point or the ball) in the stadium will have its individual layer of assigned values in the small areas. These layers include: Source layer – for the home robot to be controlled in the next command; Obstacle layers – for the opponent robots and the home robots not under the control of the next command; Target layer – for the destination position, which may be a ball position or a special position in the stadium. If the target is not the ball, a ball layer should be added. The assigned values are based on the object movement, object position and the operation mode of the game strategy; all values are governed by an equation to be discussed in the next section. A program with repeated calculation procedures will be used to fill up all values in each layer. After the values in the small areas of all layers are assigned, the layers will be combined such that each small area of the combined layer will have a resultant value. The planned path starts from the source position and connects the small areas with the minimum resultant values until it reaches the target position. It should be noted that on assigning values to the small areas, the moving direction of the object will be considered. For the opponent robots and the ball, the trick is to consider the history of their positions. For obstacle layers, the smallest value will be assigned to the small area opposite to the area pointed by the moving direction. In this way, the generated path will avoid the obstacles.

III. Path Planning Algorithm

The proposed path-planning algorithm can be summarized as follows:

Step 1. Divide the working area (the stadium) into small areas. The number of areas will affect the

smoothness of the path. More areas will give a smoother path but the computer will need more computational power to realize the real-time operation. Since the robots take different positions, the working layer of each robot will have its own characteristics. In MiroSot, six robots and one ball are present in the stadium. Considering also the target point, we may have eight layers (seven layers if the ball is the target). The number and the form of small areas in each layer are the same.

Step 2. The value assignment to the areas in each layer is considered. For the home robot to be controlled, there is a current position (I) and a target position (E) in a source layer such that the moving direction is the direction of the shortest line from I to E. For other home robots, the moving direction is also known. For an opponent robot or the ball, the moving direction is obtained by considering the current position and the previous position stored. The source layer is considered first by calculating the lengths of the distances from the current position to all other small areas in the layer. The assigned value is obtained using the following equation:

$$R_{(x,y)} = \text{round}(l_{(x,y)} + \alpha \times |\theta_{(x,y)}|), \quad (1)$$

where $R_{(x,y)}$ is the value in the small area of coordinates (x,y), $l_{(x,y)}$ is the length between the current position and (x,y), α is a constant used in all layers, $\theta_{(x,y)}$ is the angle between the line of the shortest distance and the line of $l_{(x,y)}$, round(·) rounds up its argument to the nearest interger. From (1), it can be seen that the value in each small area is assigned based on the necessary linear displacement from the current position to (x, y), and the angle between the desired orientation and the orientation towards (x, y). The value of $\theta_{(x,y)}$ is normalized to lie within [0, 1]. Under a chosen value of α, $R_{(x,y)}$ is rounded up to an integer. The criterion of choosing the value of α is to ensure the integer values assigned the small areas are sufficient to distinguish the relative possibility a small area is on the planned path. Every layer should use the same value of α. For instance, as shown in Fig. 1, if we choose α=4, $R_{(x,y)}$ can be calculated as follows:

$R_{(2,2)}$: $\theta_{(2,2)}$= 169°, $l_{(2,2)}$= 1.414,
$R_{(2,2)}$= 1.414+ (4×(|169 |/180)= 5.2≈ 5
$R_{(2,3)}$: $\theta_{(2,3)}$= 124°, $l_{(2,3)}$= 1,
$R_{(2,3)}$= 1+ (4×(|124 |/180)= 3.75≈ 4
$R_{(2,4)}$: $\theta_{(2,4)}$= 79°, $l_{(2,4)}$= 1.414,
$R_{(2,4)}$= 1.414+ (4×(|79 |/180)= 3.2≈ 3
$R_{(3,4)}$: $\theta_{(3,4)}$= 34°, $l_{(3,4)}$= 1,
$R_{(3,4)}$= 1+ (4×(|34 |/180)= 1.76≈ 2
$R_{(4,4)}$: $\theta_{(4,4)}$= −11°, $l_{(4,4)}$= 1.414,
$R_{(4,4)}$= 1.414+ (4×(|−11 |/180)= 1.66≈ 2
$R_{(4,3)}$: $\theta_{(4,3)}$= −56°, $l_{(4,3)}$= 1,
$R_{(4,3)}$ 1+ (4×(|−56 |/180)= 2.24≈ 2
$R_{(4,2)}$: $\theta_{(4,2)}$= −101°, $l_{(4,2)}$= 1.414,
$R_{(4,2)}$= 1.414+ (4×(|−101 |/180)= 3.7≈ 4
$R_{(3,2)}$: $\theta_{(3,2)}$= −146°, $l_{(3,2)}$= 1,
$R_{(3,2)}$= 1+ (4×(|−146 |/180)= 4.24≈ 4

Fig. 1 shows a home robot layer with current position at (I) and the values assigned to the 42 small areas.

When the ball is not the target of movement. We have a target layer. The angle $\theta_{(x,y)}$ is formed between the line from (E) to (I) and the line of $l_{(x,y)}$. Fig. 2 shows the area values in the target layer. Since now the ball is not the target, the target should be a stationary position. In order to generate the path, the initial and target positions must be given. The path-planning algorithm is to generate the optimal path between these two positions with obstacle avoidance.

If there is no obstacle in the working place, the two layers (source and target layers) are combined by taking the minimum values in the small areas of the same (x, y) coordinates, and these minimum values are stored into the target layer. Finally the planned path will start from the current position (I) and pass through small areas with the lowest values, until it reaches the target position. Fig. 3 shows the planned path and the values of different small areas. It should be noted that the optimal path is not necessarily unique.

Step 3. If an obstacle (O) is present, an obstacle layer is generated. The value assignment for each small area is still based on (1). Since the basic aim is to avoid colliding with this obstacle, $\theta_{(x,y)}$ is formed between the line starting from the obstacle position in the opposite direction of its movement and the line of $l_{(x,y)}$ measured from the obstacle to (x, y).

Fig. 4 shows the obstacle layer with assigned values in the 42 small areas. This layer's values will be compared with the values obtained in step 2, and the minimum values will be taken. This step is repeated until all obstacles are considered.

Step 4. If the target is not the ball, a ball layer should be added which is regarded as an additional obstacle layer. If the target is the ball, the target layer is replaced by a ball layer, and the value assignment is still based on (1). Since the basic aim is to catch the ball, $\theta_{(x,y)}$ is formed between the line starting from the ball position in the same direction of its movement and the line of $l_{(x,y)}$ measured from the ball to (x,y). The values in the small areas of this layer will be compared with other layers' values obtained in step 2 & 3 and the minimum values will be taken.

Step 5 To plan the path based on the minimum grid values, a mask with 3×3 small areas should be used. Starting with a mask centered at the initial position (I), the 8 small area values surrounding it are checked and the minimum of them will become the next small area the robot moves to. This small area will also be the center of the mask for finding the next move. This procedure will repeat until the mask center is the target position (E). If more than one small area in a mask have the same minimum value, the small area nearest to the target position will be selected. An example for planning the final path is shown in Fig. 5.

It should be noted that when the ball is the target, we are tackling a moving target. Besides, other obstacles are moving. It is assumed that all these movements are not fast so that they can be handled with a sampling period of 33 ms, which is the sampling period of the camera. In other words, the values assigned to the small areas of all layers and the planned path will be updated every 33 ms until the target is reached.

IV. Simulation

A Matlab program is developed to test the proposed path-planning algorithm. The stadium size is 150cm × 130cm. It is divided into 15 × 13 = 195 small areas. Every small area is characterized by the coordinates (x,y,θ), where x,y depict the position and θ depicts the moving direction. The home robot locations are H1: (15, 25, 0.785), H2: (55, 95, 0), H3: (115, 25, 3.927), and the obstacles locations are O1: (45, 35, 3.927), O2: (85, 65, 3.142), O3: (115, 75, 0). The ball location is E: (145, 95, 3.927). The source is the home robot at H1. The target is the ball at E. The planned path is shown in Fig. 6. The 3D plot is shown in Fig. 7, in which the small areas' values are given as the z-axis values, while the positions of the home robots, the opponent robots and the ball are shown as vertical lines.

V. Conclusion

This paper presents a method to plan the path for wheeled mobile robots. The proposed method involves decomposing the stadium into small areas and assigning values to them. We fix the size of the small areas and assign each object a layer in order to reduce the complexity. This method is fast when compared with the potential field method. It is because only a simple equation is needed to calculate the values of the small areas. The number of small areas in the stadium will affect the processing time, the values assigned to each small area and the memory size required in the computer program.

References

[1] H. K. Kyung, K. W. Ko, J. G. Kim, S. H. Lee, H. S. Cho, *"The development of a Micro Robot System for Robot Soccer Game,"* in Proc. IEEE International Conference on Robotic and Automation, *1997. Vol.1. pp.644-649.*

[2] T. H. Lee, H. F. Leung, P. K S. Tam, "A simple path planning algorithm for a robot soccer game," *in Proc. International Symposium on Signal Processing and Intelligent System (ISSPIS 99)*, 1999. pp.607-611.

[3] L. E. Kavraki, M. N. Kolountzakis and J.Latombe, "Analysis of probabilistic roadmap for path planning," *IEEE J. Robotics and automation*, vol. 14 no.1, pp. 166-171 , Feb., 1998.

[4] K. Kedem and M. Sharir, Tech. Rep. 253: An efficient motion planning algorithm for a convex polygonal object in 2-dimensional polygonal space, *Courant Inst. Math. Sci.*, New York Univ., New York, NY, 1986.

[5] Y. K. Hwang, N. Ahuja, "A potential field approach to path planning," *IEEE Transactions on Robotics and Automation*, Vol. 8, No. 1, pp. 23-32, Feb., 1992.

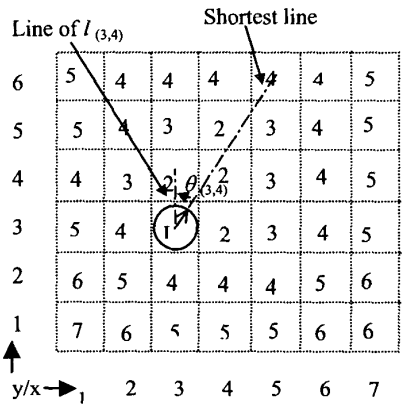


Fig. 1. Starting position, heading direction and small areas in a source layer

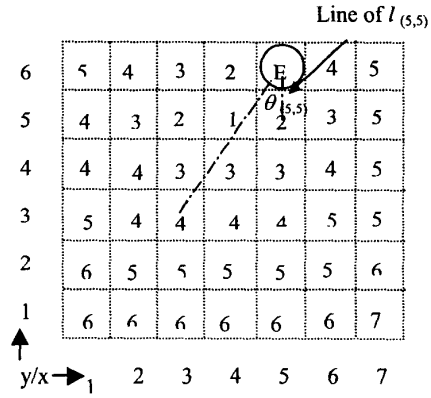


Fig. 2. Target layer.

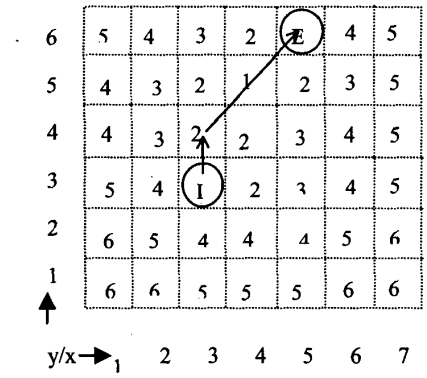


Fig. 3. Planned path by following the minimum area values

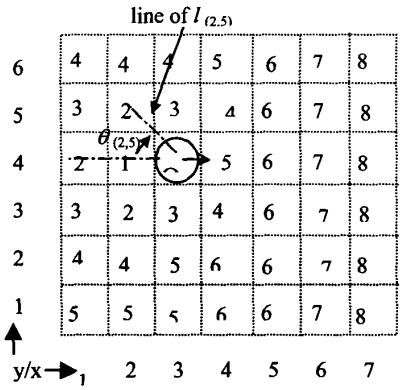
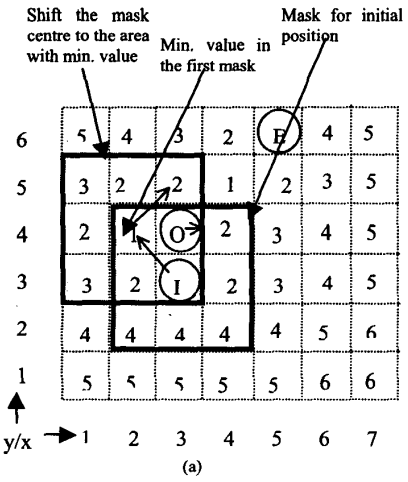
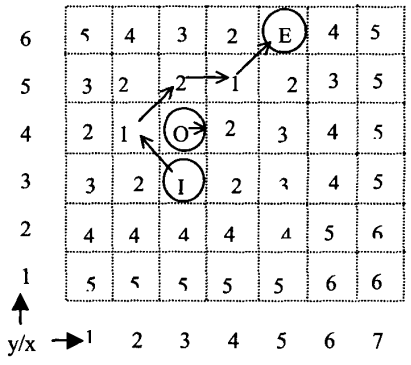


Fig. 4. The area values of an obstacle layer.



(a)



(b)

Fig. 5. Planning the path: (a) The first two steps to plan the path; (b) Selecting the minimum value of the combined map to form the path.

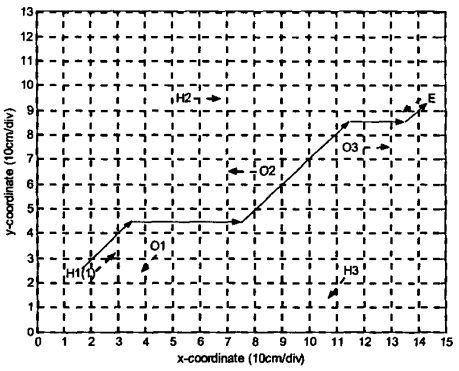


Fig. 6. The simulation result showing the reference path.

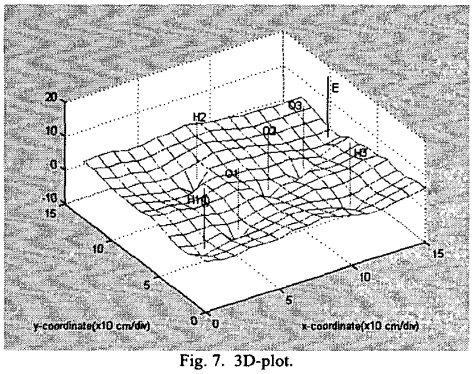


Fig. 7. 3D-plot.