Applied Mechanics and Materials Vols. 513-517 (2014) pp 4227-4230 *Online: 2014-02-06*
© (2014) Trans Tech Publications, Switzerland
doi:10.4028/www.scientific.net/AMM.513-517.4227

Micromouse Solve Maze based on Flood-fill Algorithm

Zhang Haoming[1, a], PEH lian Soon[2,b] , Wang Yinghai [3,c]

[1] Department of Electrical and Information Engineering, Tongling University, Anhui Province,China

[2]SHINING SUNSHINES GLOBAL PTE LTD, Singapore

[3] School of Electrical and Information Engineering, SIPIVT, Jiangsu Province, China

[a]zhm20060616@163.com, [b] bai_ls@yahoo.com, [c] yin287362@163.com

Keywords: micromouse, maze, flood-fill algorithm

Abstract. Micromouse championship is an international robotics competition is an event where small robot micromouse solves a 16x16 maze. Use center rule + straight -right rule the micro-mouse can quickly explore the unknown maze and find their way from a predetermined starting cell to the central area of the maze, then back to the starting cell. Map out the maze and use flood-fill algorithm to find an optimal route from the starting cell to the center cell, the micromouse will run that route in the shortest possible time.

Introduction

Micromouse championship is an international robotics competition is an event where small robot Micromouse solves a 16x16 maze. It began in late 1970s, although there is some indication of events in 1950. Events are held worldwide, and are most popular in the UK, US, Japan, Singapore, India and South Korea[1,2].

The Micromouse is completely autonomous robots that must find their way from a predetermined starting position to the central area of the maze unaided. The Micromouse will need to keep track of where it is, discover walls as it explores, map out the maze and detect when it has reached the goal. Having reached the goal, the Micromouse will typically perform additional searches of the maze until it has found an optimal route from the start to the center. Once the optimal route has been found, the Micromouse will run that route in the shortest possible time[3,4,5].

Micromouse Rules

A micromouse shall be self-contained. It shall not use an energy source employing a combustion process.

The basic function of a micromouse is to travel from the start square to the destination square. This is called a run. The time it takes is called the run time. Traveling from the destination square back to the start square is not considered a run. The total time from the first activation of the micromouse until the start of each run is also measured. This is called the maze time. If a mouse requires manual assistance at any time during the contest it is considered touched. By using these three parameters the scoring of the contest is designed to reward speed, efficiency of maze solving, and self-reliance of the micromouse.

The scoring of a micromouse shall be done by computing a handicapped time for each run. This shall be calculated by adding the time for each run to 1/30 of the maze time associated with that run and subtracting a 2 second bonus if the micromouse has not been touched yet (For example assume a micromouse, after being on the maze for 4 minutes without being touched, starts a run which takes 20 seconds; the run will have a handicapped time of: 20 + (4 * 60/ 30) - 2 = 26 seconds). The run with the fastest handicapped time for each micromouse shall be the official time of that micromouse.

Algorithm of Micro-mouse solve maze

Mark the maze In most micromouse competitions in the world today, the maze consists of 16 by 16 cells as shown in figure 1.

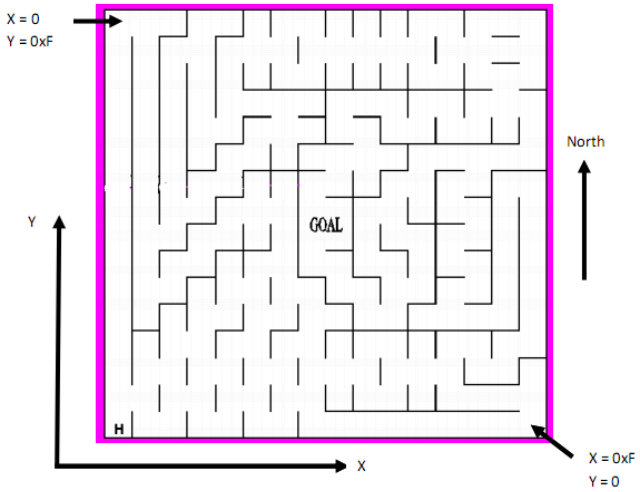

Fig.1 Maze information

Micromouse uses X and Y coordinate to keep track of its position information in the maze.

The coordinates for the starting home position is (0, 0), that is, X=0 and Y=0. There are 4 goal cells in the maze. The goal cells are (7, 7), (7, 8), (8, 7), (8, 8). Entering into any of the 4 goal cells is considered to have reached the goal.

Can change the starting cell and the centre cell by change home position and centre position parameters: HomeXY, GoalXY1, GoalXY2, GoalXY3, GoalXY4. Normally, these parameters set as follows:

```
#define HomeXY      0      #define GoalXY1      0x77
#define GoalXY2     0x78   #define GoalXY3      0x87
#define GoalXY4     0x88
```

The micromouse keeps track of its current position and direction using the variables: CurrentXY ,CurrentDir.

There are currently 4 directions for the micromouse: North, South, East and West. The starting direction of the micromouse at the Home position is considered North. Normally ,value of directions set as follows:

```
#defineNorth    0    #define East      2
#define South   4    #defineWest       6
```

The micromouse will explore the unknown maze and save the information into EastWall[256] and NorthWall[256]. EastWall[0xXY] refers to the EastWall value at coordinate (X,Y). NorthWall[0xXY] refers to the NorthWall value at coordinate (X,Y).

There are 3 possible values for EastWall[0xXY] and NorthWall[0xXY]. These values are UnknownWall, HaveWall or NoWall. Normally ,values of wall information set as follows:

```
//Wall Information
#define UnknownWall     0xFF    #define HaveWall      1
#define NoWall          0
```

Explore maze strategy Micro-mouse walks in any unit, the running direction may be at most three (front, left, right), if there are two or more possible direction can reach, following rules may use to decide which direction to go :

1: right hand rule: priority direction of forward is the right, then the straight direction, the left direction.

2: left-hand rule: priority direction of forward is the left, then the straight direction, the right direction.

3: straight -left rule: priority direction of forward is the straight line, then the left direction and right direction.
4: straight -right rule: priority direction of forward is the straight line, then the right direction and the left direction.
5: Random rule: take a random direction as the forward direction.
6: center rule: In case of intersection, priority direction point to the center of the maze.
In this paper use center rule + straight -right rule to explore the unknown maze, that is, In case of intersection priority direction points to the center of the maze, But when the direction toward the center of the road is not, use straight -right rule, A straight line as the priority forward direction, then the right direction, the left direction.
From the strategy, a micromouse can explore the unknown maze from the starting cell to the centre cell and memory the information of those cells that it have been reached. with the explored maze information, algorithm can find a way for the micromouse to dash in shortest time.

The flood-fill algorithm The simplest method available to a micromouse is some variation on the flood-fill or Bellman algorithm. The idea is to start at the goal and fill the maze with values which represent the distance from each cell to the goal. When the flooding reaches the starting cell then you can stop and follow the values downhill to the goal.

How to cross the maze using the flood information

1) Set up at start & pointing North
2)Get flood no from current cell .
3)Look for lowest flood no in adjacent cells you can get to.
4)If several equal, choose the one in same direction as going now and save as the next cell to go to
5)Make this one the new current cell and loop round from step 2 until until at end cell (middle square if going to centre)

When to finish searching

After you get to the centre or back to the start, if you run
through the flood derived route and you have visited every cell in it, there is no point in exploring any more.

solve maze with flood-fill algorithm

Use above mentioned method, maze solve as shown in follows steps.

1) As shown in figure2, first mark any cells that are adjacent to and reachable from the goal as frontier cells. Flag then added to a queue of cells to be processed.

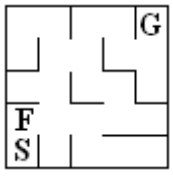

Fig.2 Flag added to maze

2) As shown in figure3, while there are cells in the frontier queue, process each one. The contents of each is replaced with the distance to the goal, either stored in the frontier queue or calculated from the neighbors as you go. The replaced cell then has its neighbours marked and placed in the frontier queue (as shown in figure4）.

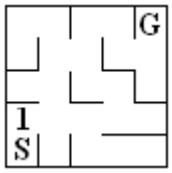

Fig.3 Distance added to maze

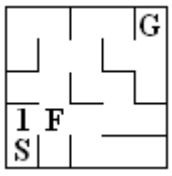

Fig.4 Next flag added to maze

4) As shown in figure5, the frontier moves toward the start and the visited cells fill with numbers.

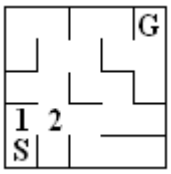

Fig.5 Another distance added to maze

5) As shown in figure6, the next sequence shows the rest of the maze being flooded.

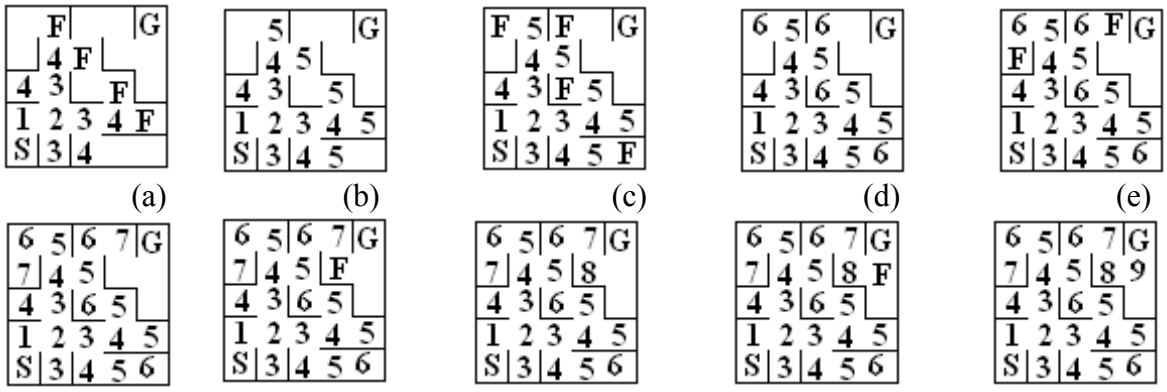

Fig.6 Sequence maze of flooded

From all above figures, can clearly see how dead ends are handled and what happens when there is more than one way through. From flood-fill algorithm, can find the shortest path of maze as shown in figure7.

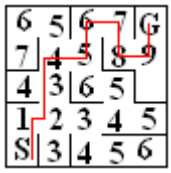

Fig.7 The shortest path based on flood-fill

Conclusion

The strategy of center rule + straight-right rule can make micro-mouse quickly explore the unknown mazeUse the saved information of explored maze can make a map of the maze, and use flood-fill algorithm according to the map can more faster than other algorithms to find an optimal route from the starting cell to the center cell.

References

[1] Su,Juing-Huei; Lee, Chyi-Shyong; Huang, Hsin-Hsiung, etc. A micromouse kit for teaching autonomous mobile robots[J], International Journal of Electrical Engineering Education, 2011, Vol.48 (2):188-201

[2] Mishra Swati, Bande Pankaj. Maze Solving Algorithms for Micro Mouse[C]. Proceedings of the 2008 IEEE International Conference on Signal Image Technology and Internet Based Systems.2008,11.

[3] L. Wyard-Scott, Q.-H.M. Meng，A potential maze solving algorithm for a micromouse robot[C]，In proceeding of: Communications, Computers, and Signal Processing, 1995：175-180.

[4] Kiblcr S G,Ilauer A E, Giesscl D S, etc. IEEE Micromousc for mechatronics research and education[C],Mechalronics (1CM), 2011 IEEE International Conference: 887-892.

[5] Po-Jen Cheng , Chin-Hsing Cheng , Chun-Cheng Tung. Design of DC motor's torque control using DSP[J]. Journal of Information and Optimization Sciences, 2009, Vol.30 (6):1197-1207

Applied Science, Materials Science and Information Technologies in Industry
10.4028/www.scientific.net/AMM.513-517

Micromouse Solve Maze Based on Flood-Fill Algorithm
10.4028/www.scientific.net/AMM.513-517.4227