UNITECH 2016 GABROVO

INTERNATIONAL SCIENTIFIC CONFERENCE
18 – 19 November 2016, GABROVO

MAZE GENERATION FOR REAL MOBILE ROBOT APPLICATION

Aydın GÜLLÜ
Trakya University

Assoc. Prof. Dr. Hilmi KUŞÇU
Trakya University

Abstract

In this study, it has developed a software that generates random mazes. Important point in the maze is at least one solution. In this context the DFS-Based Software has been developed for the maze algorithm generating. The software will be used for the testing of real mobile robot maze solving skills. Software provides facility to generate mazes which is line, corridors, one solution and more solution. Maze size can be adjusted as desired row or column. The maze created in the software by physically established, the ability of the maze solving of a real mobile robot will be measured. By developed software, finding the shortest path testing of the mobile robot can be done for maze which has multiple solutions.

Keywords: Maze Generator, Mobile Robot, DFS

INTRODUCTION

The path planning is an important point for autonomous mobile robots. The robot choose their own direction to go. If it goes the wrong direction, robot must find the right direction returning back.[1-4] Maze is the best environment for testing this situation. Test maze different from each other provides variety results.[5-7]

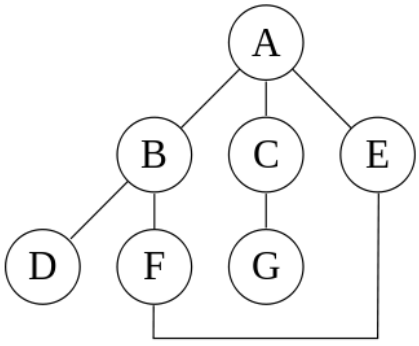

Fig. 1. *Graph type data representation*

In this study, it has developed a software that generates random mazes in any size for testing mobile robots. Depth-first search (DFS) algorithm is used to generate maze[8, 9]. Software Generates maze having at least one solution. Also all the way on the maze are connected with each other. Optionally, more than one solution in the maze can be generated. In this way, the shortest problems can be analyzed.

DEPTH-FIRST SEARCH (DFS) ALGORITHM

DFS is a tree and graph search algorithm. Graph type data representation is shown in figure 1. When running this search algorithm, it will be go the end node. Then, the end of the previous node is moved to the end again. It is continued this process until the all tree structure analysis. When DFS algorithm applied to the graph in the figure 1, Output of the algorithm happen "A-B-D-F-E-C-G". The algorithm makes a depth-first search.[9]

Route where one node to another, is selected random in the application of this algorithm. Identified each path and node are recorded as cells. Cells constitute the maze. Cells are converted to a maze by means of visual development software.

The formed maze must be at least one solution, because it is structurally depth search algorithm. Also, all paths are connected with each other because the tree structure. Multiple solving mazes can be produced, in the maze creation phase.

```
procedure DFS-iterative(G,v):
2       let S be a stack
3       S.push(v)
4       while S is not empty
```

```
5           v = S.pop()
6           if v is not labeled as
discovered:
7               label v as discovered
8               for all edges from v
to w in G.adjacentEdges(v) do
9                   S.push(w)
```

Fig. 2. *DFS Pseudocode*

First, maze boundaries is determined. It is moved a selected starting point. Direction is determined randomly. It is continued the maze limit. It is returned to a previous node when it is reached limit. Then again, it is moved toward undefined point. This process continues until all of the maze be generated. Pseudocode is shown in figure 2[10].

DEVELOPED SOFTWARE

Maze generator was developed with C# programming language. The maze generated by C# which is a visual language have been drawn. It could be specify the characteristics of the maze where you want to create via designed menus by user. When pressing the button generates the maze is created.

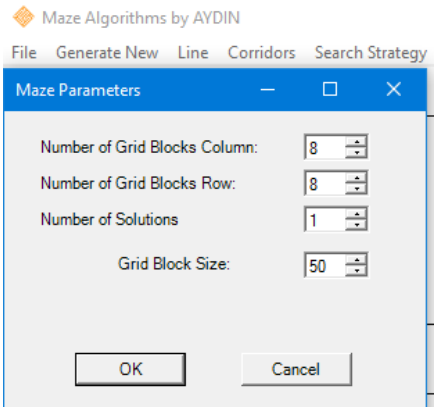

Fig. 3. *Maze Parameter Setting Screen*

The user can identify maze parameters. These parameters are maze width (columns), maze height (rows) and the number of solutions. Entering parameters is carried out under the file menu. Menu is shown in figure3. Maze structure is in the form of a matrix. Each element of this matrix is a cell. Cells consist of walls. If there is a connection between two adjacent cells, there is no wall in this section.

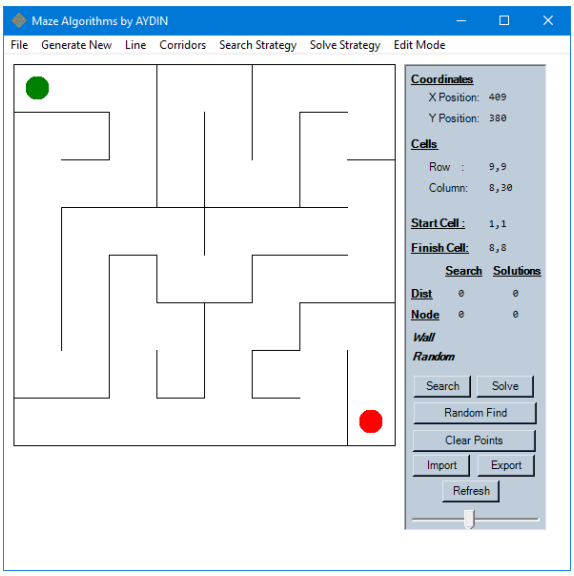

Fig. 4. *Generated Corridor Maze (8x8)*

When the maze generate button is pressed, cells are connected together randomly from first cells (0,0). This process continues until it reaches a dead end. From here back to the previous node and the process is repeated. When ways to go in the node is finished the process is completed.

The software interface is shown in figure 4. Corridors type maze were produced in this figure. Same maze convertible into line with the button in the menu. Figure 5 shows the line type maze. Applying the generated maze. There is editing mode on the software. In this way, editing can be performed on the maze.

Generated maze can be saved as a txt to be opened later. Export button is clicked through the interface to save. Import button is clicked to open the file back.

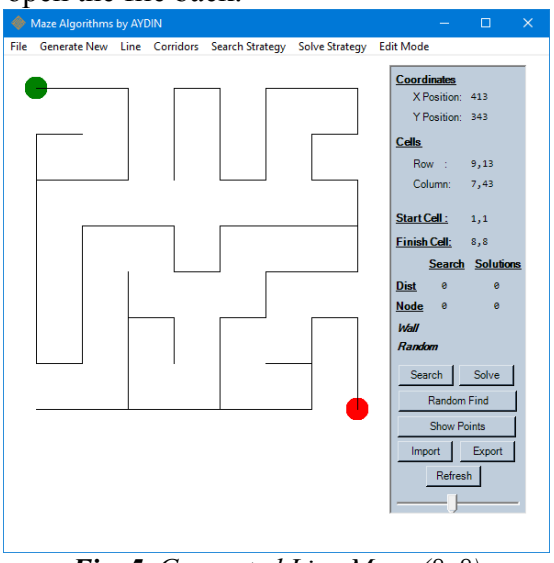

Fig. 5. *Generated Line Maze (8x8)*

APPLYING THE GENERATED MAZE

Random maze with developed software is produced. That the maze can be a real maze with chipboard size 25x12x8 mm and sigma profiles. It is shown in figure 6. At the same time, chipboard pieces arrayed on the ground, the line maze produced with the help of electrical tape, too (figure 7).

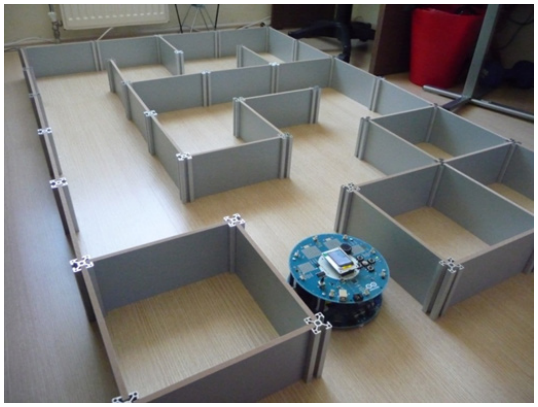

Fig. 6. *Real Corridor Maze*

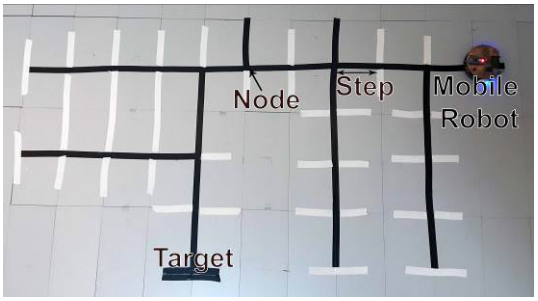

Fig. 7. *Real Line Maze*

CONCLUSION

In this study, a software that generates random mazes is developed. DFS algorithm is used to generate the maze. The generated maze is intended for use in real mobile robot movement test. The maze produced by the DFS algorithm were drawn with C # platform. Maze is shown in two ways that are corridors maze and lines maze. Maze dimensions and number of solutions can be set by users. It can be edited maze with developed software. Also maze can be saved and opened back.

The generated maze by this study was applied to the real environment. Whereby a real mobile robot's behavior can be examined. In addition, the shortest path problem of robots can be solved with generated many solving maze.

REFERENCE

[1]. Garrido, S., et al., *General Path Planning Methodology for Leader-Follower Robot Formations.* INTERNATIONAL JOURNAL OF ADVANCED ROBOTIC SYSTEMS, 2013. **10**.

[2]. Rivera, G.G.A., *Path planning for general mazes*, in *Electrical Engneering*. 2012, MISSOURI UNIVERSITY OF SCIENCE AND TECHNOLOGY. p. 45.

[3]. Achour, N. and M. Chaalal. *Mobile Robots Path Planning using Genetic Algorithms*. in *ICAS 2011, The Seventh International Conference on Autonomic and Autonomous Systems*. 2011.

[4]. Crous, C., *Autonomous robot path planning*. 2009, Stellenbosch: University of Stellenbosch.

[5]. Kozlova, A., J.A. Brown, and E. Reading. *Examination of representational expression in maze generation algorithms*. in *2015 IEEE Conference on Computational Intelligence and Games (CIG)*. 2015. IEEE.

[6]. Mae, J., et al., *Modified Line-Maze Algorithm for Mobile Robot Navigation.* Procedia Engineering, 2012. **50**: p. 740-747.

[7]. Mishra, S. and P. Bande, *Maze Solving Algorithms for Micro Mouse*, in *4th International Conference on Signal Image Technology & Internet Based Systems*, N. Vincent Oria, USA, Editor. 2008, IEEE: Bali, Indonesia. p. 86-93.

[8]. Kwek, S. *On a simple depth-first search strategy for exploring unknown graphs*. in *Workshop on Algorithms and Data Structures*. 1997. Springer.

[9]. Tarjan, R., *Depth-first search and linear graph algorithms.* SIAM journal on computing, 1972. **1**(2): p. 146-160.

[10]. Wikipedia. *Depth-first search*. 2016 [cited 2016 5.10.2016]; Available from: https://en.wikipedia.org/wiki/Depth-first_search.

Supports

This study is supported by the scientific projects unit of Trakya University. (Tubap 2014-04)