Mazes – Classification, Algorithms for Finding an Exit

M. Todorova[1], N. Nikolov [2]

Abstract - **The paper presents a classification of various types of mazes according to three basic criteria. The report analyzes, explores, and researches an algorithm for finding an exit by a line-following robot in a maze. The paper outlines the basic algorithms for controlling an object and suggests possible modifications for achieving better results.**

Keywords - **maze, robot, reflective sensor, exit finding algorithm, dimension classes, tessellation classes, routing classes**

I. Introduction

An algorithm for finding a route in a maze is a method aiming the execution of predefined tasks for passing from an initial state through several consecutive intermediate states to a final state. The transition between the separate states from the start to the finish does not have to be predetermined. There are existing probability algorithms allowing for a random choice [1].

Maze navigation is one of the main tasks in robotics. Finding the exit of the maze and reaching it as quickly as possible is the main goal. The information provided by the external sensors is a prerequisite for accomplishing the task successfully.

The paper introduces some algorithms for steering a robot moving in a maze.

II. Maze classification

Mazes can be organized along three classification criteria:

- Dimension;
- Tessellation;
- Routing;

The dimension classes are shown in Fig. 1.

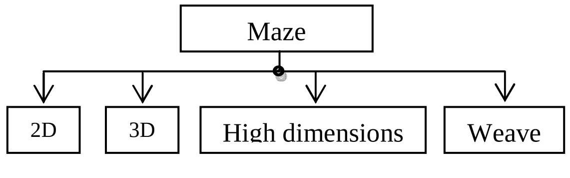

Fig. 1. Dimension Classes

- 2D – most mazes belong to that class. They do not have any overlapping passages;
- 3D – mazes with multiple levels. Passages can go in six directions – left, right, up, down, forward and backward.
- Higher dimensions - 4D and higher dimension mazes;

The tessellation classes according to the geometry of the individual cells that compose the maze are nine. The classes are shown in Fig. 2.

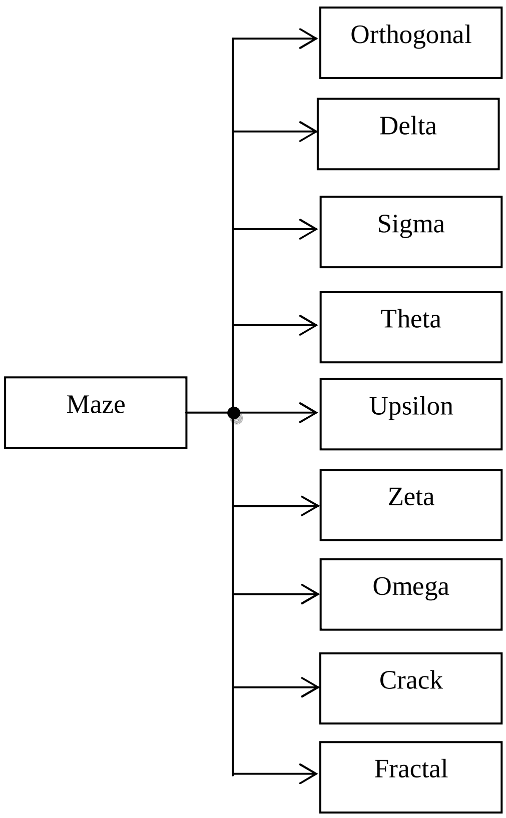

Fig. 2. Tessellation Classes

- Orthogonal – a standard rectangular grid;
- Delta – the cells are triangular. Each cell can have three neighbouring cells;
- Sigma – each cell is a hexagon. Each cell can have up to six neighbouring cells;
- Theta –The start (exit) of the maze is in the centre and the finish (entry) is on the outermost concentric circle. Each cell can have four or more neighbouring cells;

[1]Maya P. Todorova, Technical University – Varna, Computer Sciences and Engineering Department, 9010 Varna, Bulgaria, E-mail: mayasvilen@abv.bg

[2]Nedyalko N. Nikolov, Technical University – Varna, Computer Sciences and Engineering Department, 9010 Varna, Bulgaria, E-mail: ned.nikolov@tu-varna.bg.

- Upsilon – the cells are octagons or squares. Each cell has either eight or four neighbours;
- Zeta – a rectangular grid with diagonal interlocks (passages);
- Omega – all mazes with a consistent non-orthogonal tessellation. Mazes with randomly shaped and sized cells;
- Crack – mazes without any consistent tessellation, but rather have walls or passages at random;
- Fractal – a maze composed of several smaller mazes.

The routing classification refers to the types of passages between separate elements within the defined geometry. The routing classes are shown in Fig. 3.

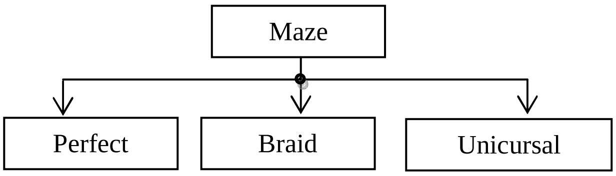

Fig. 3. Routing Classes

- Perfect – mazes without loops or closed circuits. The maze has one solution;
- Braid – mazes without dead ends. The maze consists of passages coiling around and running back into each other;
- Unicursal – mazes without any junctions. The maze consists of a single passage spanning from the start of the maze to the finish. When a U-turn is made the passage leads back to the beginning [3].

Almost all mazes can be described as a directed graph with a finite number of states and finite number of possible continuations of each state.

III. Algorithm for Finding an Exit in a Maze

A. „Black Line Follower”

The algorithm involves determining the exact orientation of the robot towards the outlines of the maze. This is achieved with the use of reflective sensors placed 5 -6 mm above the surface. The robot has five reflective sensors. The sensor reads a value of 1 when on a light surface and 0 when on a dark surface. The algorithm is applicable for mazes without closed circuits. There are eight possible situations that have to be taken into account and worked out – left turn, right turn, left or right turn, to the left or forward, to the right or forward, the four compass directions, dead end, end of the maze. The main execution stages are the following:

- following the line until a junction is reached;
- establishing the type of the junction;
- deciding a follow-up action depending on the type of the junction.

In order to achieve faster exit from the maze, the process of movement has to be split into two stages. The first stage is a “route reconnaissance” passing through the maze. During that stage the shortest route to the exit is established and stored into memory. The second stage involves following that route and reaching the exit in the shortest time possible.

The turns made during the initial passing are stored. Non–optimal turns have to be corrected. When a part of the maze requires a left or right turn the robot makes a turn without having to make a decision – there is no alternative way. These turns are not stored. The U-turn is also without an alternative but has to be stored because a dead end is reached. The shortest route does not include dead ends. That’s why the last turn has to be corrected in order to be avoided during the second passing. The movement of a robot in a maze is traced experimentally so that the shortest route is found. The passing is carried out following the “left hand” rule. The maze with its entry point and exit indicated is shown in Fig. 4.

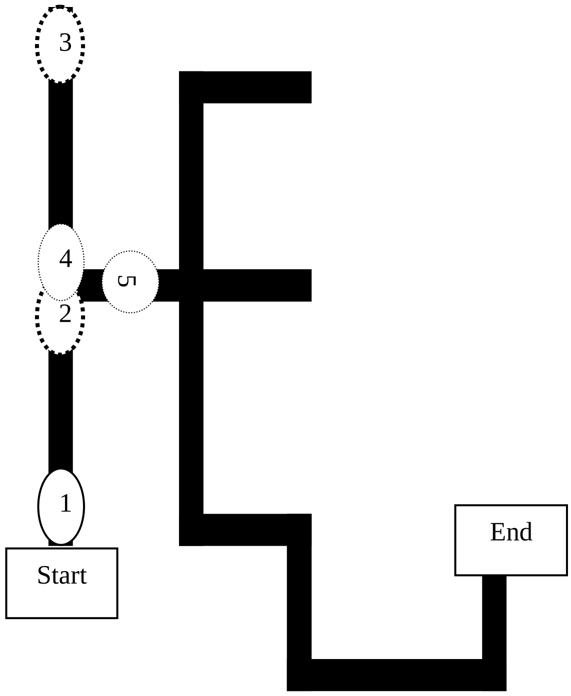

Fig. 4. Maze without Overlapping Passages

The robot is at the beginning of the maze (1 in Fig. 4) and starts its forward movement. The “decisions” memory is empty. The first junction is reached (2 in Fig. 4). The next direction is forward, the “left hand” rule is applied. The choice is stored into the memory and its record reads S. Moving forward, the sensors data change from 11011 to 11111. Therefore, a dead end is reached (3 in Fig.4) and U-turn has to be made. It is written into the memory and the robot goes on with its forward motion until the next junction (4 in Fig.4). Upon reaching the junction, a decision is made for a left turn and L is written into the memory. The memory record is SUL.

Reaching a dead end means that the previous turn was wrong. A U-turn record in the memory means that corrections have to be made for avoiding reproduction of the wrong turn. An analysis of the direction choice just before and after the U-turn has to be made. In the case above these are the directions Forward/Straight (S) and Left (L). Therefore, in order the dead end at the first junction to be avoided, a right turn has to be made. The incorrect decision (SUL) is erased from the memory and changed into R. The memory record is R. After the corrections the robot continues moving through the maze. Determining the shortest route involves substituting ineffective decisions using the Substitution Table I.

TABLE I
INEFFECTIVE DECISIONS SUBSTITUTIONS

LUL	S
LUS	R
SUL	R

Following these basic rules as shown in Table 1, the robot will succeed in finding the shortest route in any maze without overlapping passages.

B. „Wall Follower”

The “Wall Follower” algorithm is described in Fig. 5.

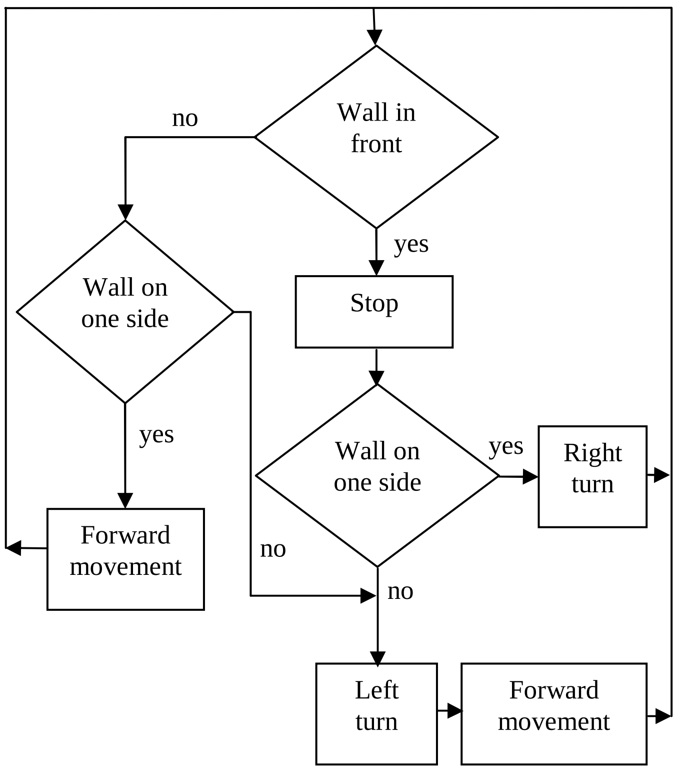

Fig. 5. “Wall Follower” Algorithm

Using that algorithm, the robot always finds the exit of the maze [5].

The algorithms can be realised with MPLab.

IV. CONCLUSION

Good knowledge of the various types of mazes helps for the correct choice of maze solving algorithm. The successful application of the algorithms for robot’s movement in a maze depends on:

- the type of the maze;
- the choice of the algorithm for passing through;
- using improvements in the maze movement algorithm;
- the information provided by the external sensors.

The main goal of the presented algorithm is reaching the exit of the maze in the shortest time possible.

REFERENCES

[1].http://en.wikipedia.org/wiki/Algorithm?&oldid=295366239

[2]. R. Dofr, R. Bishop, Modern Control System, Peason Education, 2008.

[3]. http://www.astrolog.org/labyrnth/algrithm.htm

[4]. Aleksandar, P. S. „*Проблемы Гильберта*“. Moskva : s.n., 1969.

[5]. Дипломна работа, дипломант И. Иванов, научерн ръководител доц. д-р инж. Недялко Николов

[6]. www.pololu.com