Analysis and Classification of Robot Control Algorithms

M. Todorova[1]

***Abstract* – The paper introduces a classification of the main categories of object movement algorithms in a maze. The paper also presents a brief description and a comparative analysis of the basic types of algorithms. A movement algorithm of a linear robot on a preset trajectory is outlined. The results are analysed and the necessary corrections are made in order to improve the control of the robot.**

***Keywords* – robot, robot control algorithms, random mouse algorithm, Pledge algorithm, recursive algorithm, shortest path algorithm.**

I. INTRODUCTION

The automated control systems as whole are widely implemented in various industries, medicine, households, etc.The main goal of robotics is designing and putting into effect a mobile robot movement and control algorithm. An algorithm is a finite sequence of actions carried out in succession in order to solve a given problem. The problem might be related to calculations or data processing [1]. For the programming of algorithms, information from the environment is needed. The robot cannot be analysed and evaluated separately from its environment and the tasks to be fulfilled. The robot as a device, the environment and the tasks of the robot are entwined.The relationship is shown in Fig 1. [2].

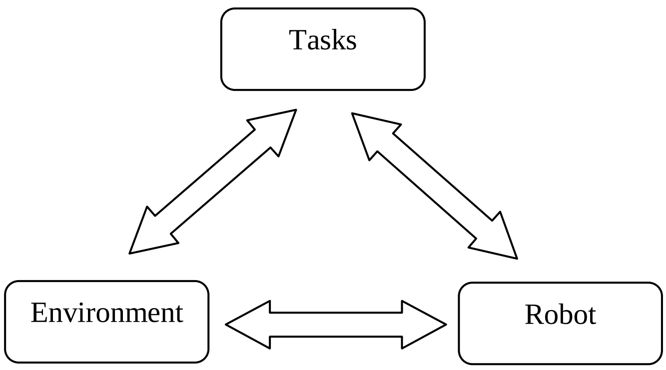

Fig 1. Tasks, Robot and Environment

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

II. CLASSIFICATION OF ROBOT CONTROL ALGORITHMS

One of the main problems in robotics is the route tracing task. The existing maze solving algorithms can be classified into two basic categories.

The first category – algorithms used when information about the labyrinth is not available;

The second category – algorithms used when an overview of the labyrinth is available;

The algorithms for robot movement in a maze are shown in Fig 2.

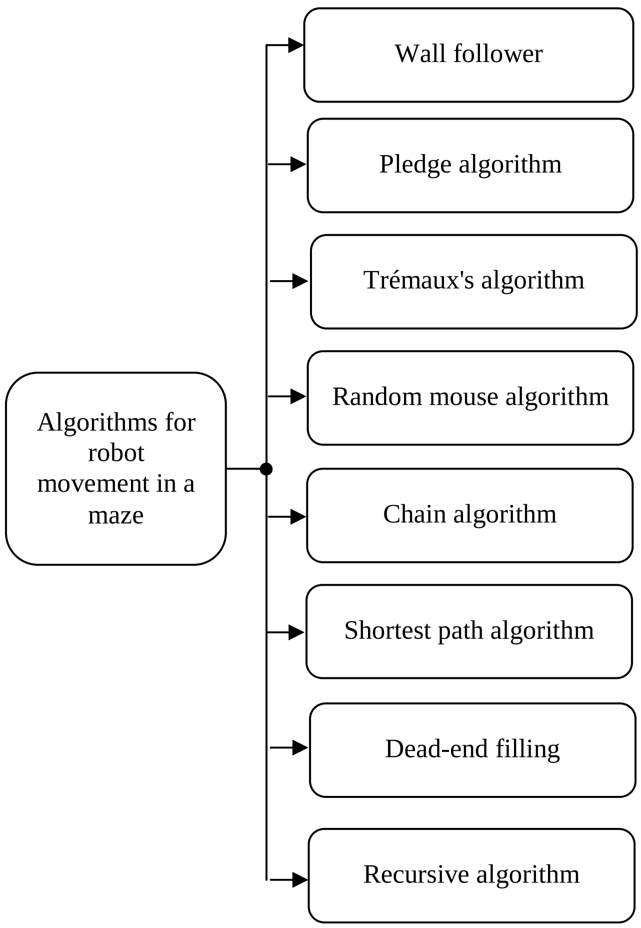

Fig 2. Depending on the means for accomplishing the task

- „Wall follower” – an algorithm also known as the “left or right hand rule”. This algorithm guarantees finding an exit from the maze if such exists. If an exit does not exist a mobile object following the algorithm will get back to the beginning of the labyrinth. One of the main algorithms used in robotics.

- „Pledge algorithm” – improved version of the wall follower algorithm.The algorithm solves the maze if it is a two-dimensional (2D) maze and the exits are situated on the external walls and the robot starts moving from any point inside the maze. Using this algorithm the robot does not need to store information about routes.The basic steps for accomplishing the algorithm are two: choosing a direction and following the chosen direction until possible. When a wall is reached it must be followed until the chosen direction becomes possible again.When implementing the algorithm, a count must be kept of the turns made +1 left turn, -1 right turn. When the total count of the turns equals zero the route tracing is aborted. The farthest wall of the current section is searched for and a transition is made to the next section. Thus, the external wall where the exit of the maze lies can be reached. Implementation of the algorithm is presented in Fig 3.

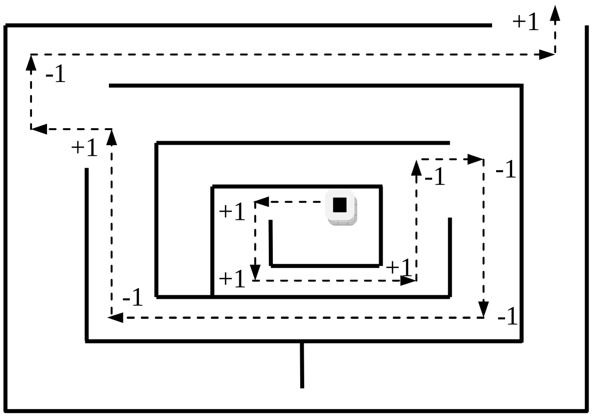

Fig. 3. Movement in a maze

- „Trémaux's algorithm” – the starting point of the algorithm is inside the maze.When moving down a passage, the path is marked.When a position in which a further movement is not possible the route is traced back following the marked path. When reaching a junction the continuation is chosen randomly. If the passage is marked it is considered a dead end. When following a marked passage an unmarked passage is chosen on the next junction. If all passages at the junction are marked, an old passage is followed and marked once again. Each passage has three possible states: unmarked (unvisited), marked once (visited only once), marked twice. If solution exists, the passages marked once will lead directly from the start to the exit of the maze. If the maze has no exit, all passages will be marked twice.
- „Random mouse algorithm“ – the algorithm includes forward movement until reaching a junction. When a junction is reached, a random direction is chosen. The basic rule of the algorithm is that the U-turns (180 degree turns) are forbidden. They are acceptable only when all other possibilities are exhausted. The algorithm is slow, ineffective and does not guarantee finding a solution. Tracing the back route from the exit is almost impossible to be implemented. The advantage of the algorithm is that additional memory is not required.
- „Dead end filling“ – the algorithm includes scanning the maze and filling each and every dead end.A dead end is a passage without an exit. The sections constituting parts of dead ends are also filled. The algorithm is characterized by rapidness and does not require additional memory.
- „Chain algorithm“ – the algorithm treats the maze as a number of smaller mazes and solves each of them separately. The algorithm specifies the start and desired end location. The algorithm is similar to the Pledge algorithm. Firstly, a straight line is drawn from the start to the end (even going through walls). Secondly, the line is followed. When a wall is reached the movement goes round the wall. This is achieved by the wall follower algorithm along the reached wall. Upon reaching the guiding line, if the new location is closer to the exit, the line is followed again. If the starting location is reached, the maze does not have a solution.
- „Recursive backtracking“ – the algorithm requires additional memory and is suitable for various types of mazes. When applying this maze solving algorithm the visited cells are marked with a special marker so that repeated entries from other directions are avoided. The algorithm always finds a solution if such exists. The solution found is usually not the shortest one.
- „Shortest path algorithm“ – the algorithm is a quick one. It requires additional memory.It is suitable for various types of mazes [3].

III. ROBOT MOVEMENT

The robot is a machine that can follow a path or can move in a maze. The path can be invisible or visible. Example for a visible path is black line on a white surface [4]. The tasks related to robot control and robot movement fall into four categories:

- Robot movement along a preset trajectory. In solving this problem no sensors are needed. The main goal is setting the parameters controlling the left, right and U – turns as well as the velocity in line with the clock rate of the processor and the reduction coefficient;
- Straight line movement;
- Movement in a maze using one or more distance sensors;
- Robot movement along a black line using digital sensors;

A robot movement along a preset trajectories one of the most important tasks.

Having completed it we will have all the necessary values of the parameters controlling the robot. The values of these parameter are used for accomplishing the tasks “Movement in a maze using one or more distance sensors” and “Robot movement along a black line using reflexive sensors”. We have at our disposal two independent motion systems which allow for three different ways of making a turn.

- Firstly – the first wheel is blocked the other wheel moves forward;

- Secondly - the first wheel is blocked the other wheel moves backward;
- Thirdly – simultaneous movement of both wheels in different directions.

When the robot makes a turn using the first or the second method not only its orientation towards its current situation is altered but the situation itself is also changed. When the robot implements the wall follower algorithm the change in the distance from the surrounding walls is a major drawback. That's why the third method has to be applied. Using it, a rotation towards the spatial geometric centre of the mobile object is made.

In order to make a U-turn the function left or right turn has to be carried out twice in a row. For forward movement equal velocities have to be set for both engines. As a result the robot will move forward following a straight line. This is possible only in a perfect environment. In real situation there are various disturbances such as:

- using two different engines reflecting on the seed of the two wheels – major flaw resulting in a robot moving off the straight line;
- the surface is not perfectly smooth – the discrepancy in the resistance leads to diversion from the direction of movement.

These disturbances lead to breaks from the perfect trajectory both in the straight sections and in the turns.This calls for corrective actions in the control algorithm in order the system to achieve a perfect trajectory. The nature of the actions depends on the given task. Deviations from the perfect trajectory are given in Fig.5.

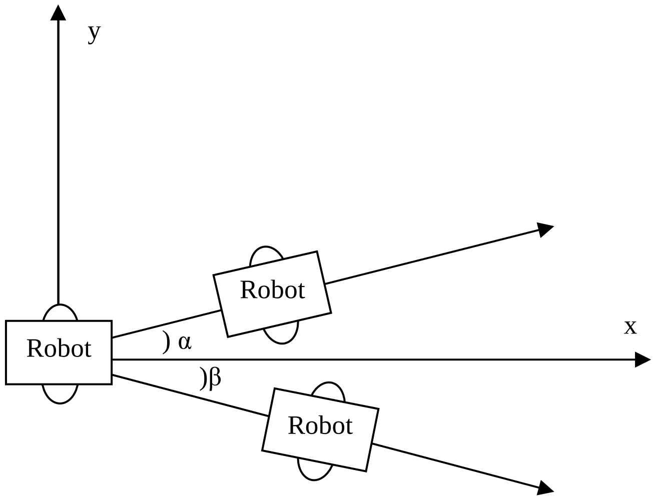

Fig. 5. Types of deviations

IV. Conclusions

The choice of a mobile robot control and movement algorithm depends on the problems to be solved, the hardware of the mobile device, i.e. the type and number of the sensors used for obtaining information about the robot location in the environment, and the disturbances arising from the random characteristics of the incoming influences. In order to avoid disturbances when implementing the algorithm of robot movement along a preset trajectory, it is necessary to carry out tests on every surface the robot might come along. The values of the coefficients of each wheel are determined thus avoiding diversions from the straight line of movement. Another way of overcoming disturbances is real – time corrections of movement by reading information from the sensors.

References

[1]. http://bg.wikipedia.org/wiki/Алгоритъм
[2]. доц. д-р. Димитър Димитров, "Системи с интелигентно поведение - Записки", 2007
[3]. Дипломна работа, дипломант И. Иванов, научерн ръководител доц. д-р инж. Недялко Николов
[4]. Priyank Patil, Department of Information Technology, K. J. Somaiya College of Engineering Mumbai, India, "Line following Robot
[5]. Александров, П. С., „Проблемы Гильберта", Москва: s.n., 1969..
[6]. Harris, Tom. "How Robots Work", http://science.howstuffworks.com/robot.htm.
[7]. Юревич И. Е., „Основы робототехники", Санкт-Петербург, 2010