Simulation of Autonomous Navigation Mobile Robot System

Lecturer. Dr. Mofeed Turky Rashid
Electrical Engineering Department / University of Basrah, Basrah / Iraq.
mofid76@yahoo.com

Asst. Lecturer. Huda Ameer Zaki
Computer Science Department / Shatt Al-arab College University, Basrah / Iraq.
hazz79@yahoo.com

Asst. Lecturer. Rana Jassim Mohammed
Computer Science Department / Education of Pure Science College
Basrah University, Basrah / Iraq.
rana2005ra@yahoo.com

Abstract:

The goal of real-time mobile robots is to travel in the shortest path between two points in a navigation environment. Autonomous navigation allows robots to planning this path without the need for human intervention. In this paper, Autonomous navigation mobile robot system has been simulated by MATLAB which a camera is used to snapshotting robot environment represented by maze while image processing techniques are used for extracting maze parameters. These parameters are used to planning the shortest path between start point and target based on Depth-first Search (DFS) and Breadth-first Search (BFS) searching algorithms.

Keywords: Autonomous mobile robot, DFS, BFS, image processing, maze.

محاكاة نظام روبوت متحرك ذاتي الملاحة

م.د. مفيد تركي رشيد
قسم الهندسة الكهربائية

م.م هدى أمير زكي
قسم علوم الحاسبات \ كلية شط العرب الجامعة

م.م رنا جاسم محمد
قسم علوم الحاسبات \ كلية التربية للعلوم الصرفة \ جامعة البصرة

الخلاصة :

الهدف من روبوتات الزمن الحقيقي هو سلوك اقصر مسار للانتقال بين نقطتين في البيئة التي يتواجد فيها الروبوت. الابحار الذاتي يتيح للروبوت تخطيط مسارات للوصول للهدف دون الحاجة إلى تدخل الإنسان. في هذا البحث تم محاكات نظام إبحار ذاتي لروبوت بواسطة برنامج MATLAB في بيئة متمثلة بالمتاهة حيث استخدمت كاميرا لتصوير البيئة التي يتواجد فيها الروبوت والمتمثلة بمتاهة بينما استخدمت تقنيات معالجة الصور لاستخلاص بارامترات المتاهة. هذه البارامترات استخدمت لتخطيط اقصر مسار من نقطة البداية إلى الهدف بالاعتماد على خوارزميات Depth-first search و Breadth-first search .

I. Introduction

Autonomous mobile robots without human intervention are required in a lot of applications such as environment navigation, scientific, military, etc. In order for these robots to get its designated destination, they must be highly intelligent and must make their own decisions. When an autonomous mobile robot moves from one point to another representing the target in its environment, it is necessary to plan a better path and at the same time avoid any obstacles [1][2]. Thus carefully planning the path is essential in autonomous mobile robots which its main purpose is to find the best collision-free path from the starting point to the end or target point and that's according to some standards like distance, time and energy. Time and distance are the mostly used[3].

There are two types of algorithms used to plan a path , they are the global path planning (off-line) and the local path planning (on-line) and that's based on the environment's information at hand. Global path planning for robots in the environments which have all the information about the location of fixed obstacles and paths for the moving obstacles, the robot calculates the best path only once then follow this path to the target point. In case there was no or little information about the environment, the mobile robot attempts itself by using sensors to figure out the surroundings, and this is called the local path planning (on-line) [4][5]. In this type of planning, the robot needs to be able to build a complete map of the environment, which is a repetitive operation of moving to the target point, updating the map, and planning the next move. The mobile robot generally has one camera or more or high frequencies sensors which help detecting any obstacle in the robot's path [6].

The problem of path planning of the mobile robot has become the center of many researchers since the mid 1970's. Many suggestions were made to handle this problem, each one of these suggestions has its strong and weak points. It is essential to choose the proper method to get both quality and efficiency in searching [7]. According to this, the desired path must not make the robot waste time in unnecessary moves or be stuck in an obstacle. Moreover, the desired path must avoid obstacles [8].

Various techniques applied for navigation of an intelligent mobile robot described in [9]. They showed that the heuristic approaches (Fuzzy logic (FL)), Artificial Neural Networks (ANN), Neuro-Fuzzy, Genetic Algorithm (GA), Particle Swam Optimization (PSO), Ant Colony Optimization (ACO) and Artificial Immune System) gave asuitable and effective results for mobile robot navigation (target reaching and obstacle-avoidance).

Also there are many algorithms such as potential field methods which are widely used for mobile robot path planning for both static and dynamic environments [10]. Visibility graph methods which constructs a graph of vertices of polygons representing obstacles [11]. It means that two vertices are connected in the graph if they are mutually visible [12] and grid methods [13]. In the grid methods, where grids are used to form the map of the environment, the main problem is how to determine the size of grids, the smaller grids, and the more precise representation of the environment. However, using smaller grids will result in exponential increase in memory space and search range [14].

In this paper, design and simulation of mobile robot path planning system based on DFS and BFS searching algorithms is achieved which are a camera will be used for snapshotting the environment of robot. The snapshot picture of the mobile robot environment will be analyzed by image processing techniques in order to find all paths of environment, and then the short distance will be found, it is full in range between starting point and target point inside robot environment. Autonomous mobile robot will use the shortest path for arriving to target.

II. The comprehensive hardware structure of autonomous mobile robot system

Comprehensive hardware structure is described by block diagram shown in **(Figure.1)**. The Camera is used for capturing the top view picture of mobile robot environment this picture is sent to personal computer in order to extract all paths of environment. Image processing has been used to analyze the picture. All information extracted from the picture has been sent to autonomous mobile robot. In autonomous mobile robot system, shortest path between start point and target has been evaluated by searching algorithm which DFS and BFS has been used for this task .

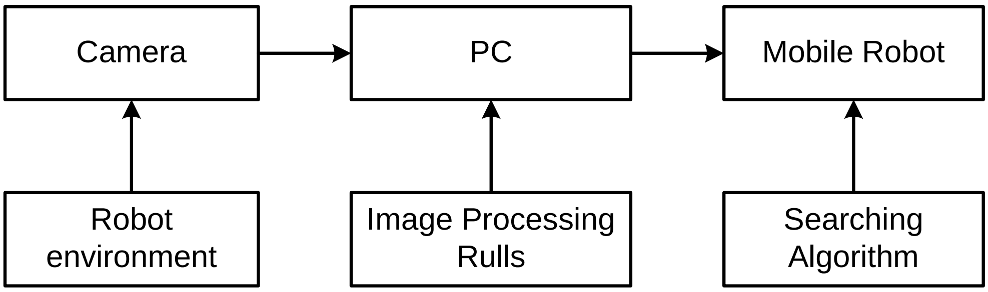


Fig .(1): Block diagram of hardware structure of system.

III. The environment picture analysis and path planning

In this part, the environment picture has been analyzed by image processing (feature extraction) in order to extract all environment paths. Environment picture is converted from RGB image to grayscale image. A grayscale image is a data matrix whose values represent intensities within range (0-255). The equation converting RGB values to grayscale values by forming a weighted sum of the R, G, and B components is given by [15]:

$$pixel\,value = (0.2989 * R + 0.5870 * G + 0.1140 * B) \qquad (1)$$

where R is red color intensity, G is green color intensity and B is blue color intensity.

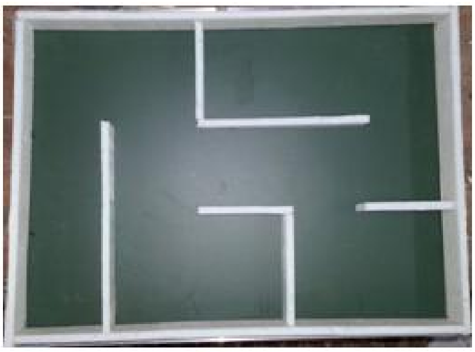
Now a grayscale image is converted to a binary image. In a binary image, each pixel assumes one of only two discrete values: 1 or 0. A binary image is stored as a logical array represents black or white color

$$pixel\ value = \begin{cases} 1\ if\ grayscale\ pixel\ value\ \geq luminance\ threshold \\ 0\ if\ grayscale\ pixel\ value\ < luminance\ threshold \end{cases} \quad (2)$$

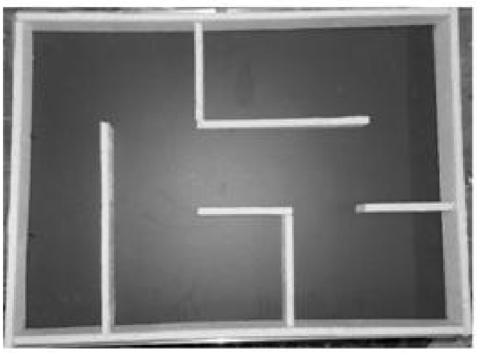
The selected value of luminance threshold depends on the distributed light of image which represented by brightness and contrast of image.

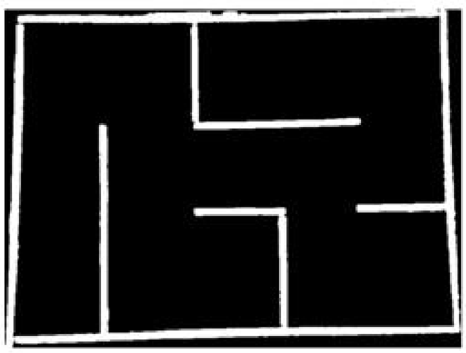
Mathematical morphology has been applied to picture in order to remove all unwanted noise and to perform sharpness and smoothness in environment paths. Morphology is a broad set of image processing operations that process images based on shapes. Morphological operations apply a structuring element to an input image, creating an output image of the same size. In a morphological operation, the value of each pixel in the output image is based on a comparison of the corresponding pixel in the input image with its neighbors. By choosing the size and shape of the neighborhood, a morphological operation can be constructed that is sensitive to specific shapes in the input image.

Mathematical morphology extracts information about the geometrical structure of an image object by transforming it through its interaction with another object, called the structuring element, which is of simpler shape and size than the original image object. Information about size, spatial distribution, shape, connectivity, convexity, smoothness, and orientation can be obtained by transforming the image object using different structuring elements [16].



Original image sample.





Grayscale image sample.Binary image sample.

Fig .(2): Sample of grayscale and binary image

The labeling functions perform connected-component labeling, which is a method for identifying objects in a binary image. These functions return a matrix, called label matrix. A label matrix is an image, the same size as the input image, in which the objects in the input image are distinguished by different integer values in the output matrix. Objects in a binary image are pixels 1 which are considered as being the foreground. When viewing a binary image, the foreground pixels appear white. Pixels 0 are considered to be the background. When viewing a binary image, the background pixels appear black.

The following example, illustrates how labeling equation can identify the objects in a binary image. In the input matrix, the pixels are set to the value one (1) which represents objects in the image. The pixels set to zero (0) represent the background. In the output label matrix, labeling equation finds every object in the input image and numbers them, in the order it finds them.

0	0	0	0	0	0	0	0
0	1	1	0	0	1	1	1
0	1	1	0	0	0	1	1
0	1	1	0	0	0	0	0
0	0	0	1	1	0	0	0
0	0	0	1	1	0	0	0
0	0	0	1	1	0	0	0
0	0	0	0	0	0	0	0

Input Image

0	0	0	0	0	0	0	0
0	1	1	0	0	3	3	3
0	1	1	0	0	0	3	3
0	1	1	0	0	0	0	0
0	0	0	2	2	0	0	0
0	0	0	2	2	0	0	0
0	0	0	2	2	0	0	0
0	0	0	0	0	0	0	0

Output Image

Image processing was performed in MATLAB. Environment picture represented as an m x n x 3 matrix; i.e., 3 color layers (red, green, blue) of m x n matrices, where m and n are the horizontal and vertical pixels. The center of mass of each object has been calculated. The overall picture has been divided into grids in which to evaluate parameters required for path planning. The size of grids has been selected depending on the size of objects in pictures (foreground) and the area of background as shown in **Figure.(3)**.

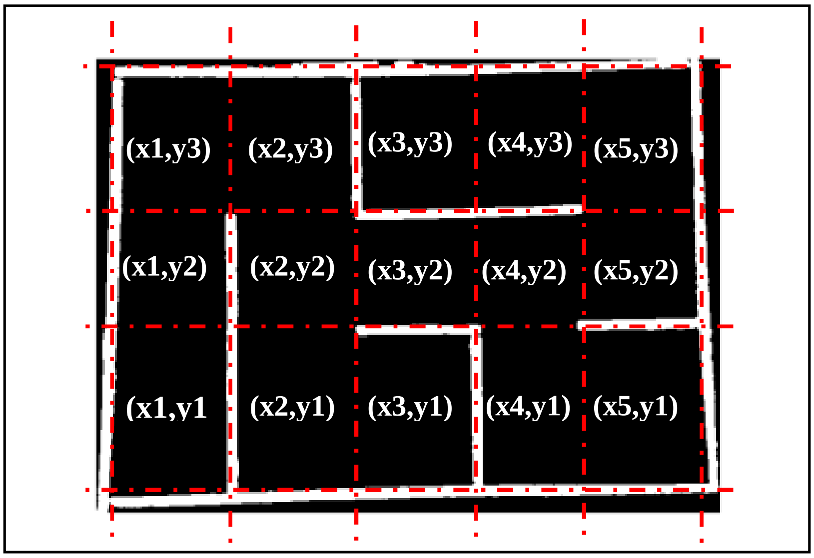


Fig .(3): Labeling robot environment.

The algorithm has been used to draw the maze (mobile robot environment) is as follow:

1. The mobile robot environment has been divided into grids.
2. The dimensions of girds are evaluated by the x and y coordinates of each object in image as shown by dotted lines in **Figure .(3)**.
3. Redrawing the maze which for each wall there are parameters (wall coordinates, wall dimensions, and direction). The place of each wall depending on object coordinates in image.
4. The size of matrix of the maze is3x5. The nodes will be fixed depending on locations of neighbor's walls.

IV. Evaluation shortest path algorithm

In this part, two methods have been introduced that can be used to search, and it will discuss how effective they are in different situations. DFS and BFS are the best-known and widest-used search methods [17]. Also looking at a number of properties of search methods, including optimality and completeness, which can be used to determine how useful a search method will be used for solving a particular problem.

Problem solving is an important aspect of Artificial Intelligence. A problem can be considered to consist of a goal and a set of actions that can be taken to lead to the goal. At any given time, the state of the search space has been considered to represent where have been reached as a result of the actions that have been applied so far.

Searching is a method that can be used by Processing Unit (Microprocessor, Micro controller, PLD, FPGA, etc.) to examine a problem space like this in order to find a goal. Often, required found the goal as quickly as possible or without using too many resources. A problem space can also be considered to be a search space because in order to solve the problem, the space will be searched for a goal state.

A. DFS algorithm

A commonly used search algorithm is DFS. It is so called because it follows each path to its greatest depth before moving on to the next path. The principle behind the DFS approach is illustrated in **Figure. (4)**. Assuming that we start from the left side and work toward the right, DFS involves working all the way down the left-most path in the tree until a leaf node is reached. If this is a goal state, the search is complete, and success is reported.If the leaf node does not represent a goal state, search backtracks up to the next highest node that has an unexplored path. This process continues until either all the nodes have been examined, in which case the search has failed, or until a goal state has been reached, in which case the search has succeeded. In **Figure. (4)**, search stops at node Jwhich is the goal node. As a result, nodes F, K, and L are never examined [17].

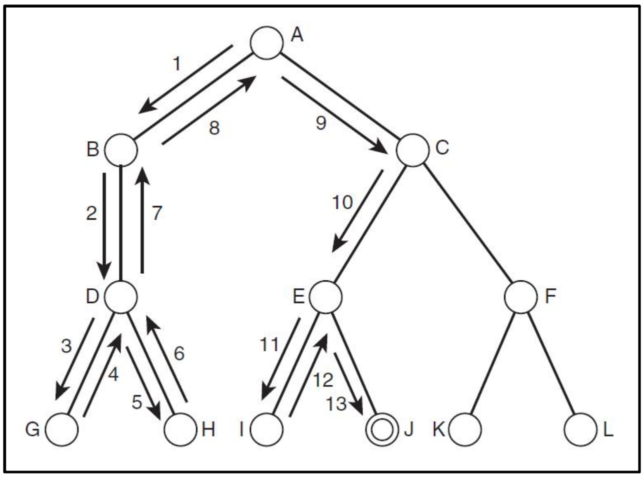


Fig .(4): Illustrating DFS algorithm.

B. BFS Algorithm

An alternative to DFS is BFS. As its name suggests،this approach involves traversing a tree by breadth rather than by depth. As can be seen from **Figure .(5)**, the breadth-first algorithm starts by examining all nodes one level down from the root node. If a goal state is reached here, success is reported. Otherwise, search continues by expanding paths from all the nodes in the current level down to the next level. In this way, search continues examining nodes in a particular level, reporting success when a goal node is found, and reporting failure if all nodes have been examined and no goal node has been found.

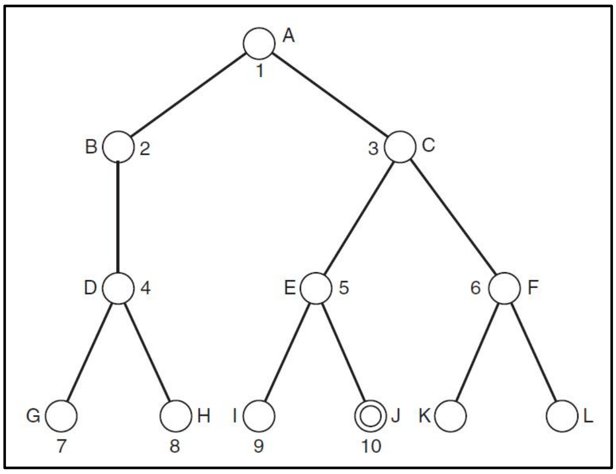


Fig .(5): Illustrating BFS algorithm. The numbers indicate the order in which the nodes are examined.

BFS is a far better method to use in situations where the tree may have very deep paths, and particularly where the goal node is in a shallower part of the tree. Unfortunately, it does not perform so well where the branching factor of the tree is extremely high.

BFS is a poor idea in trees where all paths lead to a goal node with similar length paths. In situations such as this, DFS would perform far better because it would identify a goal node when it reached the bottom of the first path it examined.The comparative advantages of DFS and BFS are tabulated in **Table (1)**

.

Table .(1): Comparison between depth-first and breadth-first search.

Scenario	DFS	BFS
Some paths are extremely long, or even infinite.	Performs badly	Performs well
All paths are of similar length	Performs well	Performs well
All paths are of similar length, and paths lead to a goal state	Performs well	Wasteful of time and memory
High branching factor	Performance depends on other factors	Performs poorly

There are several important properties that search methods shouldhave in order to be most useful.

1. Complexity

It is useful to know how efficient that method is, over time and space. The time complexity of a method is related to the length of time that the method would take to find a goal state. The space complexity is related to the amount of memory that the method needs to use.

2. Completeness

A search method is described as being complete if it is guaranteed to find a goal state if one exists. BFS is complete, but DFS is not because it may explore a path of infinite length and never find a goal node that exists on another path.

3. Optimality

A search method is optimal if it is guaranteed to find the best solution that exists. In other words, it will find the path to a goal state that involves taking the least number of steps.BFS is an optimal search method, but DFS is not. DFS returns the first solution it happens to find, which may be the worst solution that exists. Because BFS examines all nodes at a given depth before moving on to the next depth, if it finds a solution, there cannot be another solution before it in the search tree.

DFS and BFS are easy to implement, although DFS is somewhat easier. DFS is usually simpler to implement than BFS, and it usually requires less memory usage because it only needs to store information about the path it is currently exploring, whereas BFS needs to store information about all paths that reach the current depth. This is one of the main reasons that DFS is used so widely to solve everyday Processor Unit problems [17].

V. Modeling of mobile robot

In two-wheeled mobile robot control, many motion control design methods are proposed in order to move efficiently in a two-dimensional space [18]. One motion control problem of two-wheeled mobile robot is how to control the left-wheeled motor and right-wheeled motor, independently.

Let X and Y indicate the coordinates of a global reference frame, x_m and y_m, those of a reference system placed in the center of the robot P, R the radius wheel, L the distance of the wheel from P, (x_0, y_0) the coordinates of P in the global reference frame and θ the robot orientation in this reference frame, as shown in **Figure.(6).** Using these notations, the equations to derive the control signals v_R and v_L (i.e., the linear velocities of the left and right wheel, respectively) are reported in the following. For a differential drive robot،it holds:

$$\begin{aligned} \dot{x} &= v\cos(\theta) \\ \dot{y} &= v\sin(\theta) \\ \dot{\theta} &= \omega \end{aligned} \qquad (2)$$

Where $\dot{x}$ and $\dot{y}$ denote the velocity of the robot inthe direction of X and Y axes, respectively, $\dot{\theta}$ = ωthe angular velocity and v the linear velocity of the robot in the head direction. Hence, the robot motion is controlled by v and ω which are function of time t. These parameters are linked to v_R and v_L by:

$$\begin{aligned} v &= \frac{v_R + v_L}{2} \\ \omega &= \frac{v_R - v_L}{2L} \end{aligned} \qquad (3)$$

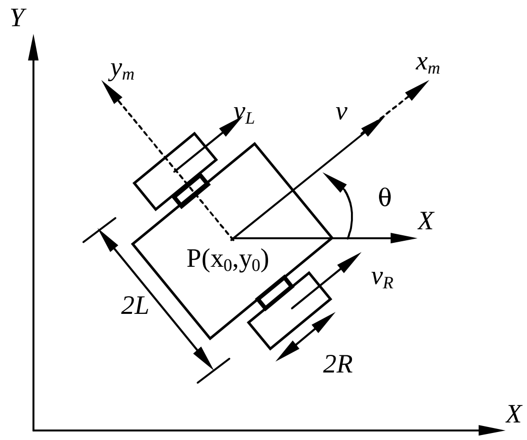


Fig .(6): Graph represents mobile robot model.

$v_L=R\omega_L$ and $v_R=R\omega_R$ are the linear velocities of the left-wheel and right-wheel, respectively.R is the radius of the wheel and ω_L and ω_R are angle velocities of the left-wheel and right-wheel, respectively. From (2) and (3), the relationship between the trajectory parameters expressed in terms of $\dot{x}$, $\dot{y}$ and $\dot{\theta}$and the control signals v_L and v_R can be derived:

$$\begin{aligned} v_L &= \cos(\theta)\,\dot{x} + \sin(\theta)\,\dot{y} + L\dot{\theta} \\ v_R &= \cos(\theta)\,\dot{x} + \sin(\theta)\,\dot{y} - L\dot{\theta} \end{aligned} \quad (4)$$

Equation (4) is used to derive the reference control signals for the wheel motors for several test trajectories used in experiments.

MATLAB has been used to simulate the motion of autonomous mobile robot which is graphically drawn by MATLAB as shown in **Figure.(7)**; the mobile robot model described above has been applied in order to achieve the motion emulation of mobile robot inside the environment.

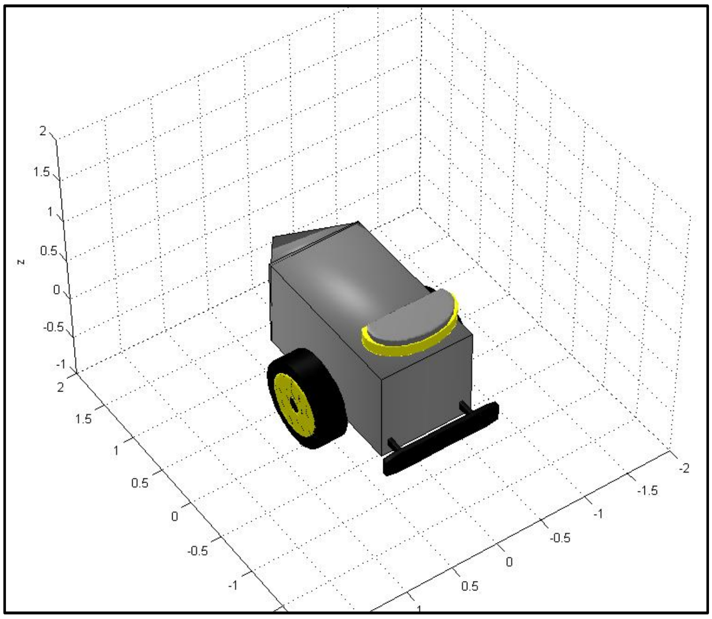


Fig .(7) : Autonomous mobile robot in MATLAB.

VI. Simulation results

The first step of experiment has been achieved by camera where a snapshot picture of autonomous mobile robot environment has been captured. This picture is send to PC in order to extract all environment paths from the picture by image processing techniques. The paths parameters have been sent to autonomous mobile robot in order to find the target by using DFS or BFS searching methods and finally shortest path from starting to end point has been selected which the mobile used to arrive to target.

MATLAB software has been used to perform image processing also to apply searching algorithms. The results represented by graphical view which the environment represented by maze. The maze is drawing by using parameters extracted from environment picture. Autonomous mobile robot drawn by MATLAB and mobile robot model described in section (V) has been achieved in order to perform mobile robot motion control.

In **Figure.(8)**, the maze of mobile robot environment drawn by MATLAB is shown. Also in **Figure.(9)** the tree diagram of all possible environment paths have been drawn, these paths evaluated by DFS and BFS search algorithm can be used for connection between any two nodes in robot environment. The boxes of red color in the diagram represented the shortest path between start and end point.

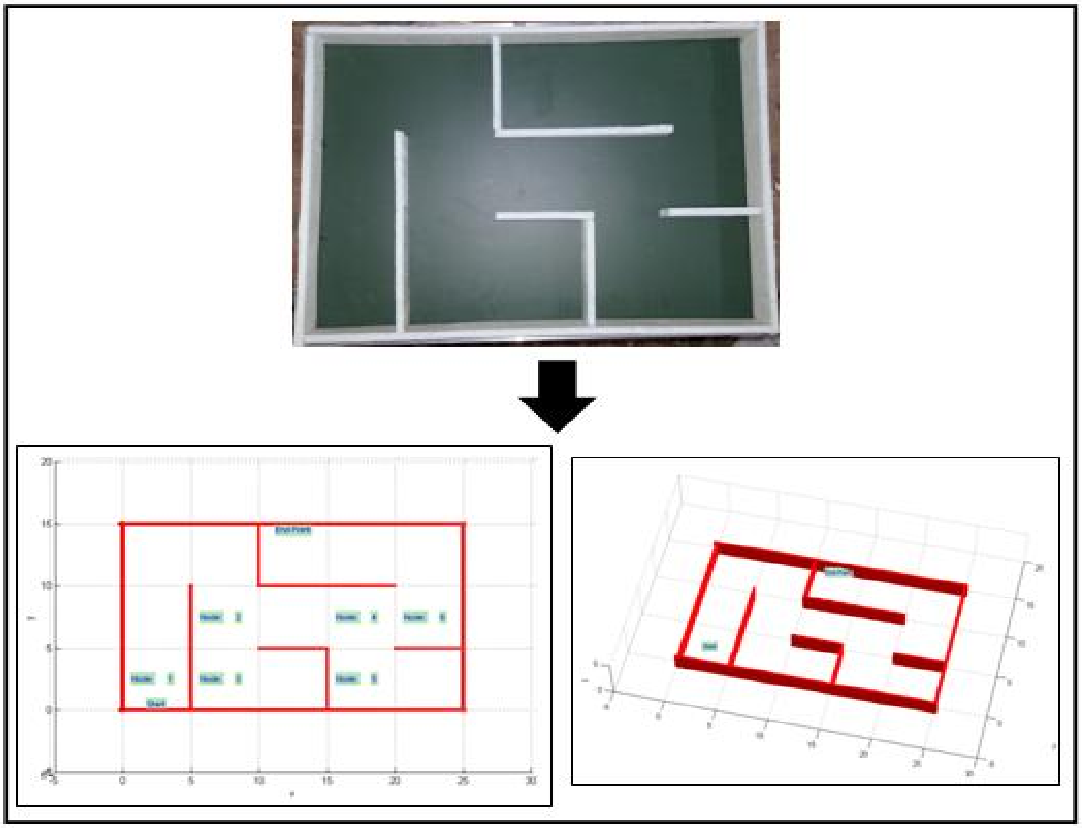


Fig .(8) : Mobile robot environment.

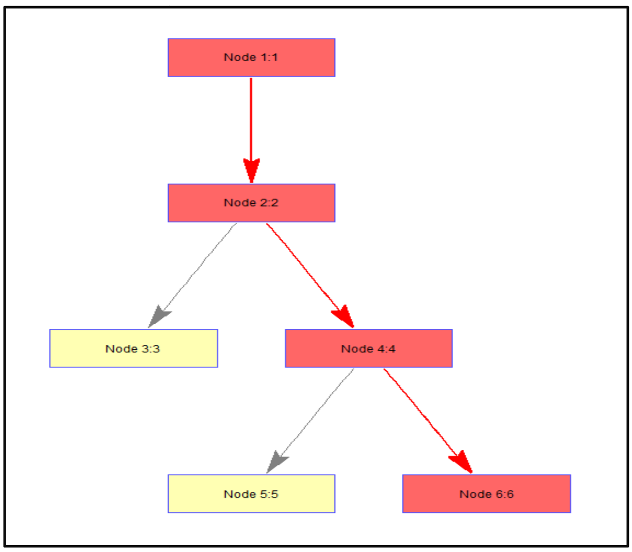


Fig .(9): Tree diagram of Mobile robot environment paths.

The trajectory of autonomous mobile robot is shown in **Figure.(10)** .Mobile robot starting from start to end point has taken the shortest path between two points.

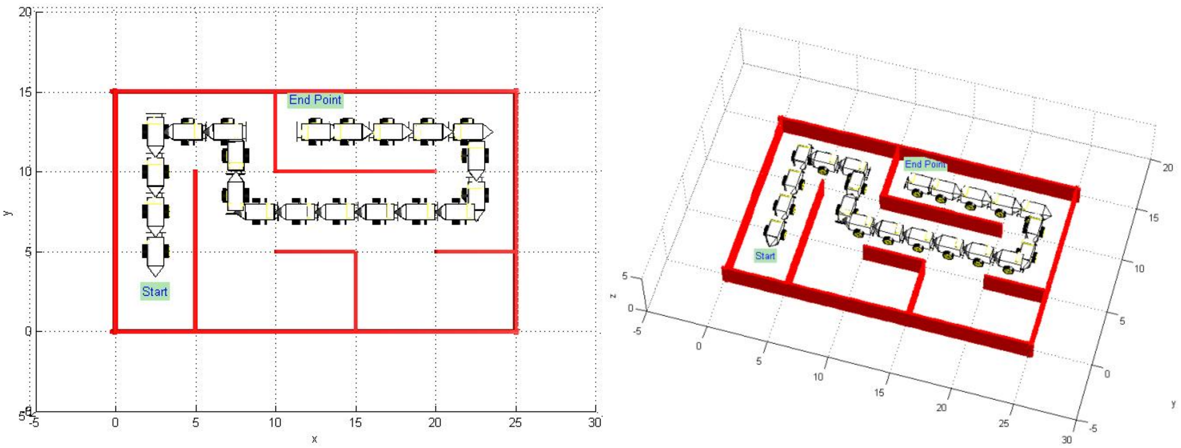


Fig .(10) : Autonomous mobile robot trajectory.

As shown in **Figure.(10)**, the mobile robot trajectory is similar for DFS and BFS searching methods because the shortest path between start and end point has been evaluated, but the difference between the two methods is how to evaluate the environment paths which DFS is less complexity than BFS while BFS is more reliable to find target than DFS.

VII. Conclusions

A maze is selected as example of robot environment which a camera is used to snapshot robot environment image. The parameters of the robot environment are extracted form snapshot picture by image processing techniques which these parameters are used to draw maze by MATLAB. Image processing technique affected directly on accuracy to find target which the image enhancement is used to modify the values of image brightness and contrast, also morphological algorithms is used to remove unwanted pieces in image.

Two searching methods have been used to evaluate all possible paths to target. The methods are DFS and BFS and the results of these methods are shown in Fig. 9. As can be seen they have the same results but there are differences in performance like complexity and completeness, optimally and time for best solution.

- In space DFS is very efficient compared with BFS because it only needs to store information about the path it is currently examining.
- In time BFS is not efficient compared with DFS because it can end up examining very deep branches of the tree.
- Some branches in tree infinite so DFS is not completeness.
- BFS is an optimal search method, but DFS is not.

References

1. **Mohammad Abdel Kareem Jaradat, Mohammad Al- Rousan ,Lara Quadan, "Reinforcement based Mobile Robot Navigation in Dynamic Environment" , Science direct Journal of Robotics and Computer Integrated Manufacturing,Vol.27,Issue1,Page no. 135-149, 2011.**
2. **Buniyamin N., Wan Ngah W.A.J., Sariff N., Mohammad Z., "A Simple Local Path Planning Algorithm for Autonomous Mobile Robots", International Journal Of Systems Applications, Engineering & Development, Volume 5, Issue2,Page no.151-159, 2011.**
3. **Raja P, Pugazhenthi S, "Path Planning for Mobile Robots in Dynamic Environments using particle swarm optimization", IEEE International Conference on Advances in Recent Technologies in Communication and Computing (ARTCom 2009), Kottayam, India, Pages no. 401-405, 2009.**
4. **Zhao Y.,Zu W.," Real-Time Obstacle Avoidance Method for Mobile Robots Based on a Modified Particle Swarm Optimization", International Joint Conference on Computational Sciences and Optimization, Pages no. 269-272, 2009.**
5. **Li P., Huang X., Wang M., "A New Hybrid Method for Mobile Robot Dynamic Local Path Planning in Unknown Environment", Journal of Computers, North America, Volume 5, Pages no. 773-781, 2010.**
6. **Velagic J., Lacevic B., and Osmic N., "Efficient path planning algorithm for mobile robot navigation with a local minima problem solving", In Proceedings of IEEE International Conference on Industrial Technology, IEEE, Mumbai, Pages no. 2325–2330, 2006.**
7. **Yun S. C., Ganapathy V., Chong L.O., "Improved genetic algorithms based optimum path planning for mobile robot", International Conference on Control, Automation, Robotics and Vision, ICARCV, Pages no. 1565-1570, 2010.**
8. **Han K. M., "Collision free path planning algorithms for robot navigation problem", Master Thesis, University of Missouri-Columbia, 2007.**
9. **Mohanty P. K., and Parhi D. R., "Controlling the Motion of an Autonomous Mobile Robot Using Various Techniques: a Review", Journal of Advance Mechanical Engineering, Volume 1, Pages no. 24-39, 2013.**
10. **Ge S. S., Cui Y. J., "New potential functions for mobile robot path planning", IEEE T. Robot. Autom., Volume 5, Pages no. 615- 620, 2000.**
11. **Li L., Ye T., Tan M., "Present state and future development of mobile robot technology research", Robot, Volume 5, Pages no. 475-480, 2002.**

12. Saska M., Macas M., Preucil L., Lhotska L., "Robot Path Planning using Particle Swarm Optimization of Ferguson Splines", Proceedings of the IEEE Conference on Emerging Technologies and Factory Automation, Prague, Pages no. 833-839, 2006.
13. Boschian V., Pruski A., "Grid modeling of robot cells: A memory efficient approach", J. Intell. Robot. Syst., Volume 2, Pages no. 201-223, 1993.
14. Zheng T. G., Huan H., Aaron S., "Ant Colony System Algorithm for Real-Time Globally Optimal Path Planning of Mobile Robots", Acta. AutomaticaSinica, Volume 3, Pages no. 279-285, 2007.
15. K. S. Fu, R. C. Gonzalez, and C. S. G. Lee, "Robotics: Control, Sensing, Vision, and Intelligence", McGraw-Hill, 1987.
16. Petros A. Maragos, and Ronald W. Schafer, "Morphological Skeleton Representation and Coding of Binary Images", IEEE Transactions on acoustics speech, and signal processing, vol. 34, no. 5, 1986.
17. Xinguo Deng, Yangguang Yao, Jia Chen, andYufeng Lin, "Combining breadth-first with depth-first search algorithms for VLSI wire routing ", Advanced Computer Theory and Engineering (ICACTE), 2010 3rd International Conference on, vol. 6, pp. V6-482 - V6-486, 2010.
18. C. Lin, A. B. Rad, W. L. Chan, and K. Y. Cai,"A robust fuzzy PD controller for automatic steering control of autonomous vehicles," PD controller for automatic steering control of autonomous vehicles," IEEE International Conference on Fuzzy Systems, vol. 1, pp. 549-554, 2003.