INTERNATIONAL JOURNAL OF COMPUTERS
Issue 3, Volume 5, 2011

An Autonomous Fuzzy-controlled Indoor Mobile Robot for Path Following and Obstacle Avoidance

Mousa T. AL-Akhras, Mohammad O. Salameh, Maha K. Saadeh, and Mohammed A. ALAwairdhi

***Abstract*—**This paper provides the design and implementation details of an autonomous, battery-powered robot that is based on Fuzzy Logic. The robot is able to follow a pre-defined path and to avoid obstacles, after avoiding an obstacle, the robot returns back to the path. The proposed system is divided into two main modules for path following and for obstacle avoidance and line search. Path following controller is responsible for following a pre-defined path, when an obstacle is detected, obstacle avoidance and line search controller is called to avoid the obstacle and then to return to the path. When the robot finds the path again, path follower controller is called.

LEGO Mindstorms NXT robot was used to realise the proposed design. The detailed design steps of the robot are provided for readers who are interested in replicating the design. For the implementation of the path following, Fuzzy Logic was employed. Fuzzy Logic controller takes the robot's light and ultrasonic sensory readings as input and sends commands to the robot motors to control the robot's speed and direction.

An extensive set of experiments considering both simple and complicated scenarios for path following and obstacle avoidance were conducted and the results proved the effectiveness of the system. Images of such scenarios are provided for reference and many videos were uploaded and the links are given in the paper for interested readers.

Keywords Fuzzy Logic, LEGO Mindstorms NXT, Robot, Obstacle Avoidance, Path Following.

I. Introduction

THE aim of this paper is to provide the design and implementation details of an autonomous, mobile indoor robot controlled by Fuzzy Logic (FL). The purpose of this controlling system is to enable a robot to:

1. Follow a pre-defined path like the one that leads visitors to different parts of an establishment like a hospital.
2. Avoid any obstacle the robot detects on its way.

The scenario that is adopted in this paper is depicted in Fig. 1. Utilising FL, the robot follows a predefined path until if finds an obstacle, it then avoids the obstacle and returns back to the path. This continues with other obstacles until the end of the path. The robot shown in Fig. 1 is provided with 3 light sensors and one ultrasonic sensor. The design and implementation details of this robot are provided in the rest of the paper.

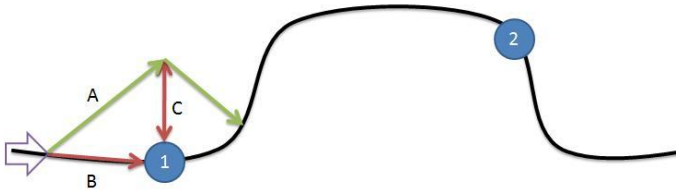

Fig. 1: General fuzzy-controlled robot system.

The biggest challenge that has motivated us to use FL is to enable smooth movement for the robot which is a major factor that motivated researchers to use FL in controlling problems.

It was decided that a robot control scheme can be used to process sensory information and provide control action decisions. Data is collected via an ultrasonic sensor (also infra red sensor can be used) and light sensors. Sensory data are transmitted to the robot's brain where all of the calculations are taken place. These data have to be processed in the brain of the robot using a Fuzzy Inference System (FIS) and then out to the motors which in turn perform robot drive and steering commands. A FL controller consists of a rule-base that fuzzifies range values and produces defuzzified control actions. The control actions are crisp values that represent the robot's speed and steering directions.

Any information about the robot's environment is provided solely by the attached sensors. No feedback of kinematic or dynamic information (such as position and velocity) is employed. The use of such basic sensors allows us to examine the power of FIS when constrained by minimal observability. From a practical point of view, this constraint is realistic relative to most autonomous mobile systems with limited computational resources. It has been demonstrated that such

Manuscript received March 11, 2011: Revised version received April 24, 2011.

Mousa AL-Akhras is an Assistant Professor in Artificial Neural Networks & Communications with the Computer Information Systems Department, King Abdullah II School for Information Technology (KASIT), The University of Jordan, , P.O. Box 13835, Amman, 11942, Jordan (phone: +962-6-5355-000 extension 22602; fax: +962-6-3500-233; e-mail: mousa.akhras@ju.edu.jo).

Mohammad Salameh is a Robotics Trainee and Programmer at the National Education Center for Robotics (NECR), Jubilee Center for Excellence in Education, Amman, Jordan (e-mail: m.omar82@gmail.com).

Maha Saadeh is a M.Sc. student with the Computer Science Department, King Abdullah II School for Information Technology (KASIT), The University of Jordan, Amman, Jordan (e-mail: saadeh.maha@yahoo.com).

Mohammed ALAwairdhi is an Assistant Professor in Software Engineering & Evolution with the Department of Computer Science, College of Computer and information Sciences, Al - Imam Muhammad ibn Saud Islamic University, Saudi Arabia (e-mail: mawairdhi@imamu.edu.sa).

systems exhibit satisfactory performance when utilising reactive and/or behavioural control strategies.

The rest of paper is organised as follows: section II reviews the related literature. Section III gives design and implementation details of the proposed system. Section IV gives details about the conducted experiments and results. Section V concludes the paper and presents avenues for future work.

II. Literature Review

In this section the needed literature is reviewed. Section II.A reviews several attempts for path following and obstacle avoidance. Section II.B gives the reader background information on Fuzzy Logic. Section II.C introduces LEGO Mindstorms NXT.

A. Path Following and Obstacle Avoidance

Pires and Nunes [1] presented an obstacle avoidance, behaviour-based architecture for a Reactive Shared-Control (RSC) system designed as a FL controller. The main aim was to provide easier and safer robotic wheelchair navigation. An improvement to the previous architecture was introduced by Bento et al. [2] through combining a FL implementing path tracking behaviour with the obstacle avoidance behaviour to enhance the navigation of the wheelchair.

To overcome the uncertainty of the unknown and dynamic robot behaviour, an adaptive robust fuzzy controller was proposed by Giap et al. [3], Lyapunov function theory was used to guarantee the stability of the tracking errors. A more sophisticated and intelligent path following method based on human driving behaviour for mobile robots with nonholonomic constraints was proposed by Liu [4]. The method suggests the use of two FL controllers: Robot Position Controller (RPC) and Robot Orientating Controller (ROC). The values of the position error and orientation error determine which fuzzy controller is used to control the movement of the robot.

B. Fuzzy Logic

Fuzzy Logic (FL) was introduced by Lotfi Zadeh in his 1965's seminal paper [5]. FL provides a method of reducing as well as explaining system's complexity and it is used for controlling purposes. Fuzzy sets are functions that map an input value, which might be a member of a set, to a number between zero and one (not only zero or one like in the crisp logic), indicating its degree of membership. A degree of zero means that the value is not in the set and a degree of one means that the value is completely represented in the set. Intermediate values indicate partial membership in the set.

Considering the example of determining a car's speed based on its distance from someone's home. The basic three processing steps of FL are explained as follows:

- **Fuzzification**: The translation of real-world concepts (values) to fuzzy world values using user-defined membership functions. For example the distance of over 100 km is translated into the value of 1 (*Far*). Distance less than 1 km is translated into the value of 0 (*NOT Far*). A distance between 1 km and 100 km is translated into a value between 0 and 1. The same is applied to the current speed which may be *slow*, *medium*, or *fast*.
- **Rule Evaluation**: Apply a set of rules like: If SPEED=SLOW and HOME=FAR then GAS=INCREASE.
- **Defuzzification**: After computing the fuzzy rules and evaluating the fuzzy variables, fuzzy values need to be translated back to real-world concepts. A membership function is needed for each output variable. For example GAS=INCREASE corresponds to applying 10% increase to the current gas level.

Several researchers have elaborated on the work of Zadeh, notably the FL controller introduced by Mamdani [6]. Since its introduction, FL control method has been successfully applied in enormous number of control problems, e.g., [7] [8] [9] [10] [11] [12] [13] [14].

C. LEGO Mindstorms NXT

LEGO Mindstorms NXT is a programmable robotics kit released by LEGO® in 2006 [15]. It replaced the first-generation LEGO Mindstorms, which was called the Robotics Invention System.

The main component of the kit is a brick-shaped computer called NXT brick. NXT brick can be connected to four sensors and it can control three motors via cables. The brick has a 100X64 pixel gray scale Liquid Crystal Display (LCD) and four buttons that can be used to navigate the user interface using hierarchical menus. NXT is supplied with power using 6 AA batteries. LEGO has released the firmware for NXT Intelligent Brick as open source. The technical specifications of the LEGO Mindstorms NXT as given in [16] are presented as follows:

- 32-bit ARM7 main microprocessor (256 KB cache memory, 64 KB RAM)
- 8-bit ATmega48 microcontroller at 4 MHz (4 KB cache memory, 512 Bytes RAM)
- 100x64 pixel LCD matrix display.
- A single USB 2.0 port (12 Mbit/s).
- Bluetooth (Class II) wireless connectivity, supporting the Bluetooth Serial Port Profile (SPP).
- 4 input ports, support for I2C.
- 3 output ports.

There are two ways to control LEGO NXT Robot:

1. Writing programs that will be executed on the NXT brick. NXT-G is the official programming environment that comes bundled with the NXT. NXT-G is adequate for basic programming tasks, like driving motors, incorporating sensor inputs, and performing basic calculations. There are other unofficial programming languages that can be used with NXT such as Not eXactly C (NXC) which is a simplified version of C. The list in [17] describes some of these languages.
2. NXT Remote Control (RC): NXT supports Bluetooth

communication and includes SPP [18], which can be considered as a wireless serial port; utilising this feature, RC programs can communicant and control the NXT.

III. The Proposed System

This section gives details about the proposed system. Section III.A explains the general framework of the system. Section III.B describes the used hardware design. Section III.C describes the implementation details.

A. General Framework

The general framework for an autonomous robot controller for path following and obstacle avoidance using FL is given in this section. The robot reads sensory data provided through the attached light and the ultrasonic sensors and sends commands to the motors to follow the path and to avoid the obstacles. The general flow of the system is shown in Fig. 2.

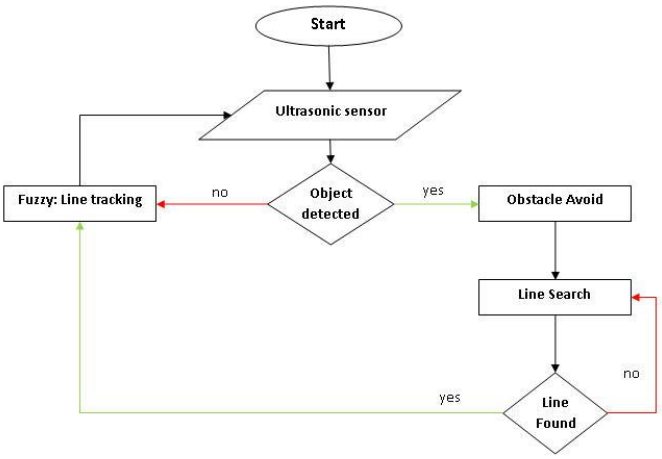

Fig. 2: System flowchart.

To track a line, a Path Following Controller (PFC) is employed. PFC uses a FIS to control the robot's movements. Consequently, the robot can move freely and smoothly on a pre-defined path. When the robot detects an obstacle through its ultrasonic sensor, it calls Obstacle Avoidance and Line Search (OALS) controller. OALS controller avoids an obstacle and then it searches for the line again to return to the path. The speed and direction of the robot motors will be decided based on whether PFC or OALS is in control, i.e. depending on whether there is an obstacle on the robot way or not. The general flow for the OALS controller is shown in Fig. 3.

OALS controller employs lights sensors, if a black line is found, OALS stops searching and control is given back to PFC to continue with path following. If the line is not found, rotation sensor is employed to change the robot's direction, when the robot rotates to the required degree of 30, it moves forward. Further rotation and move forwarded is performed until the path is found, or when the robot passes the specified distance which is 28 cm for the used robot, if the robot did not find the path within the specified distance, the robot will rotate 30 degrees to the left.

B. Hardware Design

The hardware structure for the proposed system is illustrated in Fig. 4. The design includes three light sensors used to track the path, one ultrasonic sensor to detect any obstacle on the path, and two motors (Left and Right). By moving the two motors in different directions (forward and backward), the steering angle can be controlled. Moving them in the same direction, but with different speeds can also control the steering angle. Moving them with the same speed moves the robot either forward or backward in a straight line. The actual robot design is shown in Fig. 5.

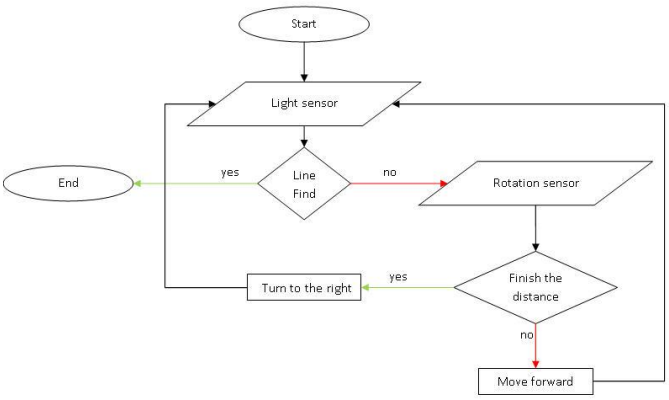

Fig. 3: OALS controller flowchart.

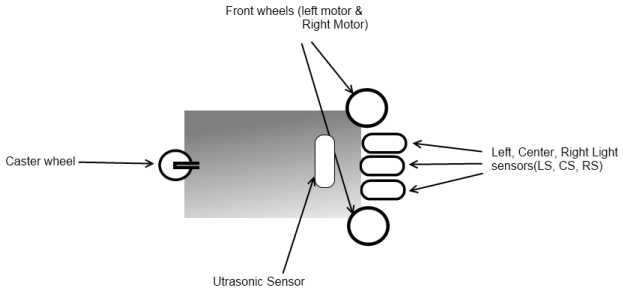

Fig. 4: Hardware structure.

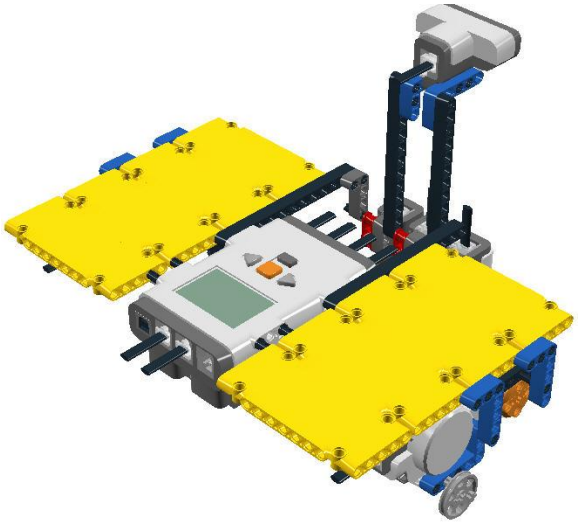

Fig. 5: Robot design.

To allow the robot design to be replicated by interested researchers, an exact step by step design is provided through LEGO Digital Designer and a video file is uploaded to:

http://www.ju.edu.jo/sites/Academic/mousa.akhras/Material/FuzzyRobot/NXTDesign.mp4

This MP4 file can be played using many media players such as Windows Media Player or Winamp.

C. System Implementation

In this section, the implementation details of the main system modules are given.

1) Module One: Path Following Controller:

Path Following Controller (PFC) controls the movement of the robot over a pre-defined path. As shown in Fig. 6, PFC FIS has four inputs and two outputs. The input parameters are the values read by the Left light Sensor (LS), Centre light Sensor (CS), Right light Sensor (RS), and ultrasonic sensor. The outputs are the commands for the Left and Right motors to control the speed and the steering angle of the robot.

Converting a real-world input value into a fuzzy value that determines the degree to which it belongs to each fuzzy set, according to the membership function, is a process called fuzzification.

Since we use LEGO Mindstorms NXT light sensor, each light sensor domain is [0 to 1000]. Fig. 7 shows the mapping from real-world light sensor (Left sensor is taken as example) into a fuzzy value using two membership functions for *black* and *white*. *Black* is a Triangle function with vertices (0, 0), (0, 1) and (570, 0). Any reading greater than 570 has a 0 membership value of *black*. *White* is a Trapezoid function with vertices (550, 0), (750, 1), (1000, 1) and (1000, 0). These vertices are shown in Fig. 7.

For LEGO Mindstorms NXT ultrasonic sensor domain is [0 to 250]. Fig. 8 shows the membership function for the obstacle distance measured using the ultrasonic sensor, with three membership functions, *Too near*, *Near* and *Far*. *Too near* is a triangle defined with the vertices (0, 0), (0, 1) and (10, 0). *Near* is a triangle defined with the vertices (8, 0), (15, 1), and (30, 0). *Far* is a Trapezoid function with vertices (25, 0), (108, 1), (250, 1) and (250, 0). These vertices are shown in Fig. 8.

After the fuzzification process, an inference system, like Mamdani [6] is used. The four inputs' fuzzy values are evaluated according to the fuzzy rules listed in Table I. In this step the values are applied to the rules antecedents. The minimum fuzzy value will be chosen since AND operator is used. The results from the antecedent evaluation are applied to the consequent part of the rules. Then the membership functions of all rules consequent are combined into a single fuzzy set in rules aggregation step. This set is used in the defuzzification step to obtain the final output.

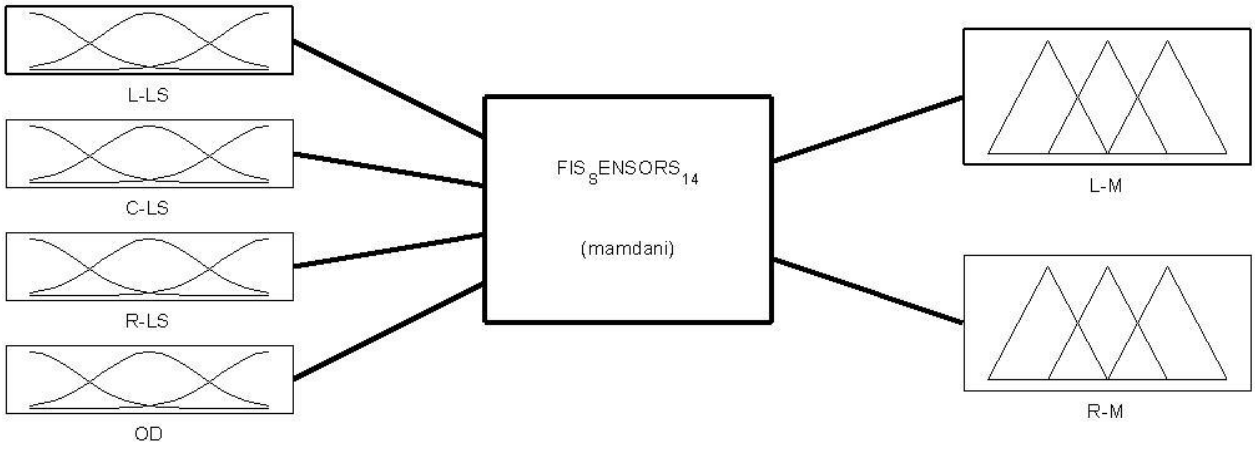

Figure 6: Fuzzy system for PFC.

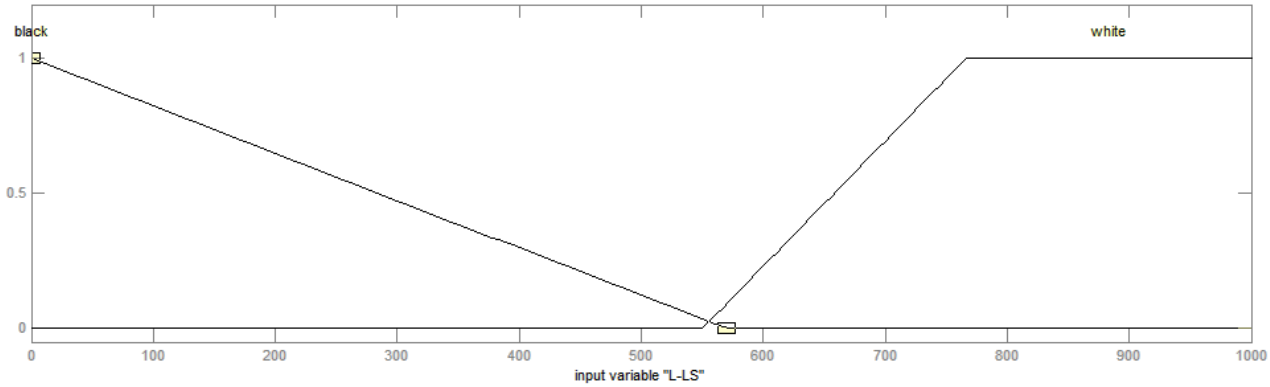

Figure 7: Fuzzy membership functions for the left light sensor input.

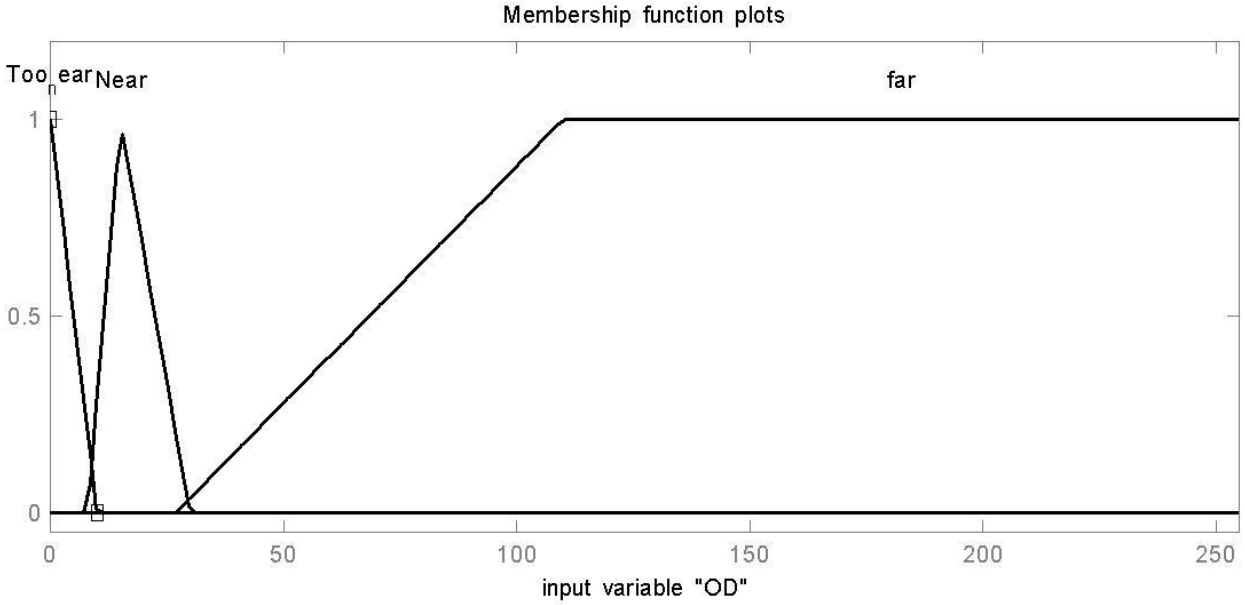

Figure 8: Fuzzy membership functions for the obstacle distance input.

Table I: Fuzzy Rules for PFC.

1. If (OD is Too_near) then (L-M is stop)(R-M is stop)
2. If (OD is Near) then (L-M is stop)(R-M is stop)
3. If (L-LS is white) and (C-LS is black) and (R-LS is white) and (OD is far) then (L-M is fm)(R-M is fm)
4. If (L-LS is black) and (C-LS is black) and (R-LS is white) and (OD is far) then (L-M is rs)(R-M is fs)
5. If (L-LS is black) and (C-LS is white) and (R-LS is white) and (OD is far) then (L-M is rm)(R-M is fm)
6. If (L-LS is white) and (C-LS is black) and (R-LS is black) and (OD is far) then (L-M is fs)(R-M is rs)
7. If (L-LS is white) and (C-LS is white) and (R-LS is black) and (OD is far) then (L-M is fm)(R-M is rm)
8. If (L-LS is black) and (C-LS is black) and (R-LS is black) and (OD is far) then (L-M is fs)(R-M is fs)
9. If (L-LS is white) and (C-LS is white) and (R-LS is white) and (OD is far) then (L-M is fs)(R-M is fs)

The outputs from the fuzzy system are power for the left and right motors. As discussed earlier, the amount and direction of the two motors control the speed and the steering angle of the robot, the power domain for the motor is [-100 to 100], negative sign represents backward movement. The membership function for the left motor is shown in Fig. 9 where the Left motor is taken as example.

The mapping from a fuzzy value into a real-world power motor value uses five membership functions for reverse medium (*rm*), reverse small (*rs*), *stop*, forward small (*fs*), and forward medium (*fm*). All of the above functions are defined as triangles. *rm* is defined with vertices (-100, 0), (-100, 1), and (-60, 0). *rs* is defined with vertices (-66.67, 0), (-33.33, 1), and (0, 0). *stop* is defined with vertices (-3.33, 0), (0, 1), and (3.33, 0). *fs* is defined with vertices (0, 0), (33.33, 1), and (66.67, 0). *fm* is defined with vertices (60, 0), (100, 1), and (100, 0).

Fig. 10 shows an example of how the fuzzy rules of the implemented FIS are evaluated to find a relation between the input and the output and deciding on the speed and direction for the left and right motors based on the sensory readings.

Assume that the four inputs have the following values: L-LS = 640, C-LS = 280, R-LS = 600, and OD = 200.

The first step is to fuzzify the input values according to the fuzzy membership functions shown in Fig. 7 and Fig. 8. As a result from this step, each input will have a new fuzzy value corresponds to the original value read by the sensor. The fuzzy values are:

- L-LS = 0.0 black and 0.5 white
- C-LS = 0.6 black and 0.0 white
- R-LS = 0.0 black and 0.2 white
- OD = 0.0 too-near, 0.0 near, and 1.0 far

After input fuzzification the rules are evaluated according to the fuzzy values shown in Table I, consequently the third rule is fired in this case as illustrated in. Fig. 10.

Since we use Mamdani inference, the minimum input fuzzy value is selected, 0.2 in this case, which is used as a horizontal cutting value in the output membership functions that result from the rules evaluation, *far* in this case, so we clip the *far* membership function at 0.2. The final step is to defuzzify the output fuzzy value, in this example the result of the defuzzufication step is 24.8 for both L-M and R-M. This value will be sent to the left and right motors respectively.

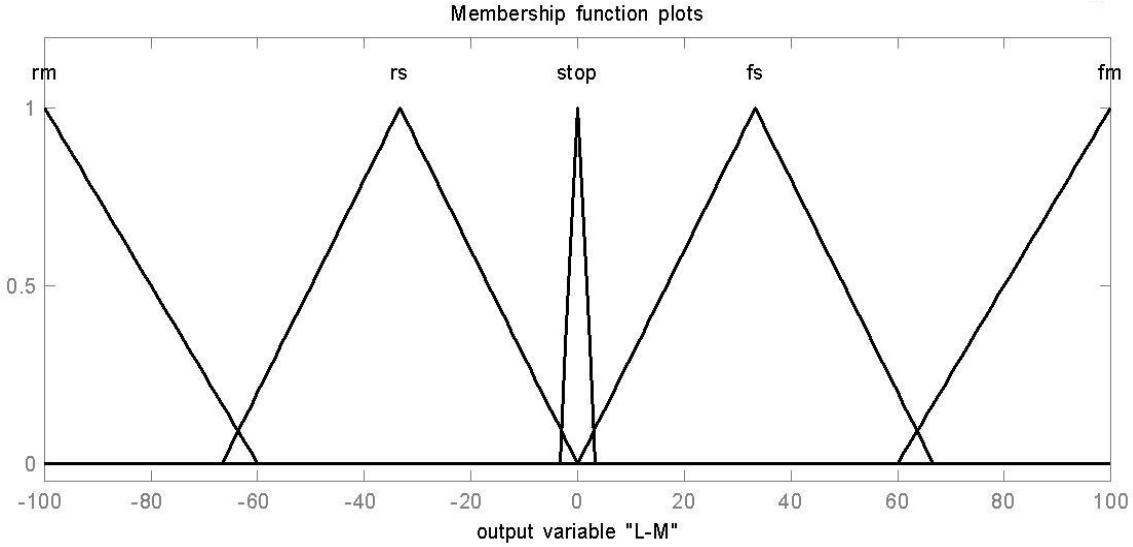

Fig. 9: Fuzzy membership functions for the left motor output.

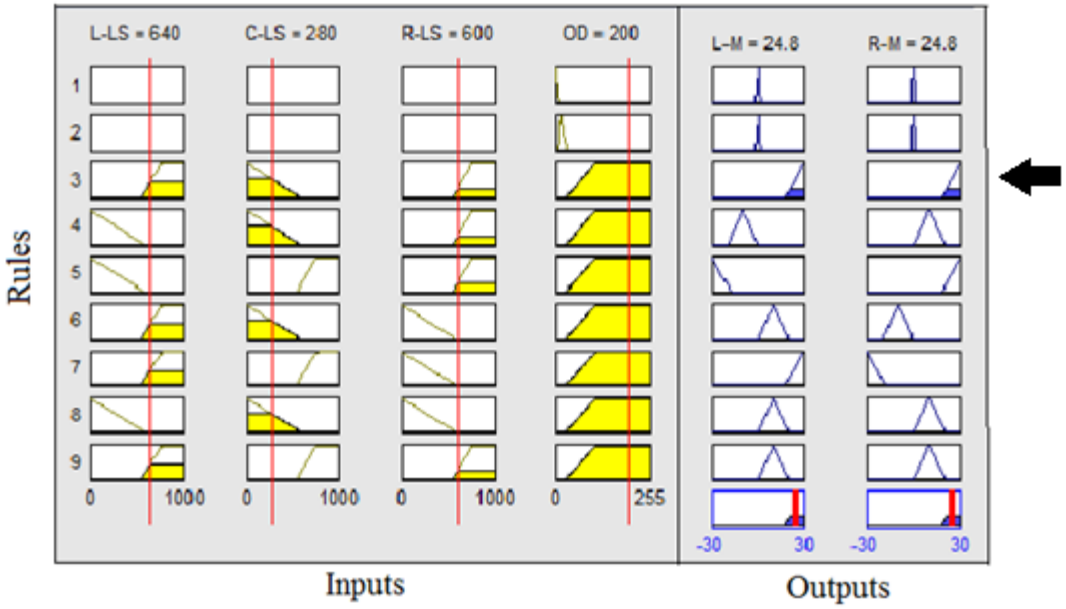

Fig. 10: Example of using PFC FIS.

2) Module Two: Obstacle Avoidance and Path Searching:

Obstacle Avoidance and Line Searching (OALS) controller is activated when an obstacle is detected. To avoid an obstacle, OALS controller steers the robot in a specific angle, moves forward and then it attempts to find the path again.

To steer the robot in the needed angle one of the robot wheels must stop and the other must rotate. The amount of wheel rotation depends on the angle that is required to steer the robot, the diameter of the wheel and the distance between the wheels.

The robot turnover result from the movement of the robot motors inverse the direction of rotation, the robot is turnover on the circumference of a circle diameter equal to the distance between the wheels Let *d* be the distance between the two wheels in mm as shown is Fig. 11. Our robot has a distance between the wheels equals 230 mm.

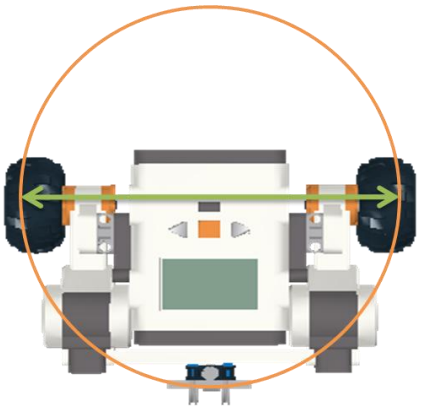

Fig. 11: Example of using PFC FIS.

The circumference of a circle can be calculated as

$$c=\pi * d \quad (1)$$

For the shown circle, the circumference equals c= (22/7)*230=723 mm.

Note that the distance crossed by a robot depends on the usage wheel circumference. When you move the wheel one rotation of 360°, the robot moves a distance equal to the circumference of the wheel. For the used robot, the wheel's diameter, *d*, equals 56 mm, therefore the circumference (c) of the wheel can be calculated as, c= (22/7)*56=176 mm. This means that when the wheel rotates 360°, the robot moves 176 mm.

We will describe how the robot turnovers 30° and how the robot moves distance about 28 cm.

Rotating the robot 30° covers, cc, of the circumference. cc can be calculated as (30/360)*723 = 60 mm. Recall that rotating 360° moves the robot 176 mm, therefore, 60mm*360°/176mm= 123° is the number of degrees the wheel is required to rotate so that the robot rotates 30°.

We need to move the robot forward a distance of 28 cm (280 mm), the rotation angle can be calculated as:

$$Rotation\ Amount = d*360°/c \quad (2)$$

Where

d is the required distance

c is the robot circumference

To move the robot 280 mm, both wheels should move forward, 360° rotation moves the robot 176 mm, therefore, for the used robot, the required degrees equals to (280/176)*360°=572°. i.e., the wheel needs to rotate 572° to move the robot 280 mm.

IV. Experiments and Results

An extensive set of simple and complicated scenarios for path following and obstacle avoidance were performed to test the effectiveness of the system.

At the beginning of the research, several wrong attempts were encountered before fine-tuning the system. These attempts were recorded and uploaded to:

http://www.ju.edu.jo/sites/Academic/mousa.akhras/Material/FuzzyRobot/Wrong01.mp4

Another 22 attempts were also uploaded; just replace the number "01" in the address above by "02" to "23".

Several successful experiments considering different scenarios were conducted; these experiments are reported and detailed as follows:

1. Scenario 1: Simple Path Follow

In this experiment, a simple path following scenario is considered. No obstacles were placed on the path. The path is shown in Fig. 12. The robot successfully followed the path and the experiment was repeated in the reverse direction and it was successful as well. Both videos were uploaded to:

http://www.ju.edu.jo/sites/Academic/mousa.akhras/Material/FuzzyRobot/Case01_01.mp4

And

http://www.ju.edu.jo/sites/Academic/mousa.akhras/Material/FuzzyRobot/Case01_02.mp4

2. Scenario 2: Following a Path with Cuts

In this experiment, a path following with cuts scenario is considered. The path has cuts to increase the difficulty level. No obstacles were placed on the path. The path is shown in Fig. 13. The robot successfully followed the path and the experiment was recorded and a video file was uploaded to:

http://www.ju.edu.jo/sites/Academic/mousa.akhras/Material/FuzzyRobot/Case02_01.mp4

3. Scenario 3: Following a Path with Loops and Cuts

In this experiment, a path following scenario is considered. The path has loops and cuts to increase the difficulty level. No obstacles were placed on the path. The path is shown in Fig. 14. The robot successfully followed the path and the experiment was recorded and a video file was uploaded to:

http://www.ju.edu.jo/sites/Academic/mousa.akhras/Material/FuzzyRobot/Case03_01.mp4

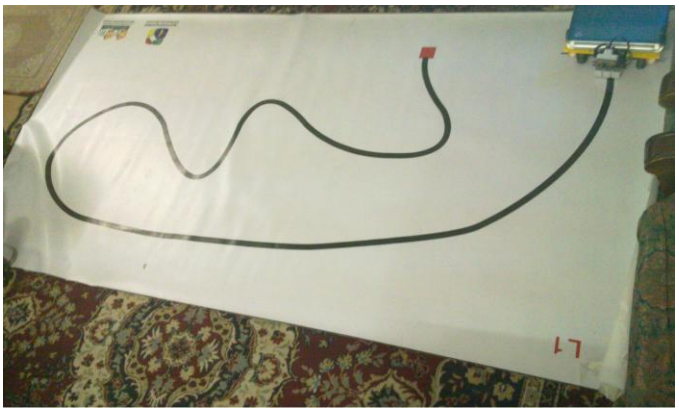

Fig. 12: Scenario 1: Simple path follow

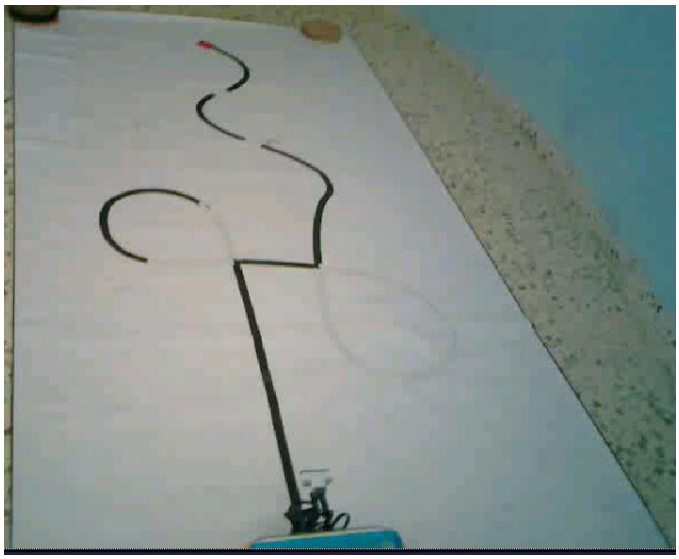

Fig. 13: Scenario 2: Following a path with cuts

4. ***Scenario 4: Following A Path with Cuts and one obstacle***

In this experiment, a path with cuts and one obstacle is considered. The path is shown in Fig. 15. The robot successfully followed the path and avoided the obstacle and the experiment was recorded and uploaded to:

http://www.ju.edu.jo/sites/Academic/mousa.akhras/Material/FuzzyRobot/Case04_01.mp4

5. ***Scenario 5: Following the same path with no obstacle, with one obstacle and with two obstacles***

In this experiment, a path was used several times with more than one scenario.

i. **No obstacles were placed on the path**. The robot successfully followed the path and the experiment was recorded and uploaded to:

http://www.ju.edu.jo/sites/Academic/mousa.akhras/Material/FuzzyRobot/Case05_01.mp4

ii. **One obstacle was placed on the path.** The robot successfully followed the path and avoided the obstacle and the experiment was recorded and uploaded to:

http://www.ju.edu.jo/sites/Academic/mousa.akhras/Material/FuzzyRobot/Case05_02.mp4

iii. **Two obstacles were placed on the path.** The robot successfully followed the path and avoided the obstacles and the experiment was recorded and uploaded to:

http://www.ju.edu.jo/sites/Academic/mousa.akhras/Material/FuzzyRobot/Case05_03.mp4

iv. **The camera was placed on the robot to see the path from the robot's angle.** The robot successfully followed the path and avoided the obstacle. Fig. 16 shows the path from the robot's angle. The whole experiment was recorded and uploaded to:

http://www.ju.edu.jo/sites/Academic/mousa.akhras/Material/FuzzyRobot/Case05_04.mp4

V. Conclusions and Future Work

In this paper we proposed the design and implementation for an autonomous, path following and obstacle avoidance fuzzy-controlled robot system. Fuzzy logic was used to control the robot's movements on a pre-defined path. The full design and implementation details were given in the paper.

An extensive set of experiments for simple and complicated scenarios for path following and obstacle avoidance were conducted and the results proved the effectiveness of the system. Images of different scenarios were provided for reference and many videos were uploaded and the links were given in the paper for interested readers.

As a future work we will implement this design on a wheel chair robot and evaluate its effectiveness in following complicated paths and multiple obstacles using robots other than LEGO Mindstorms NXT robot kit.

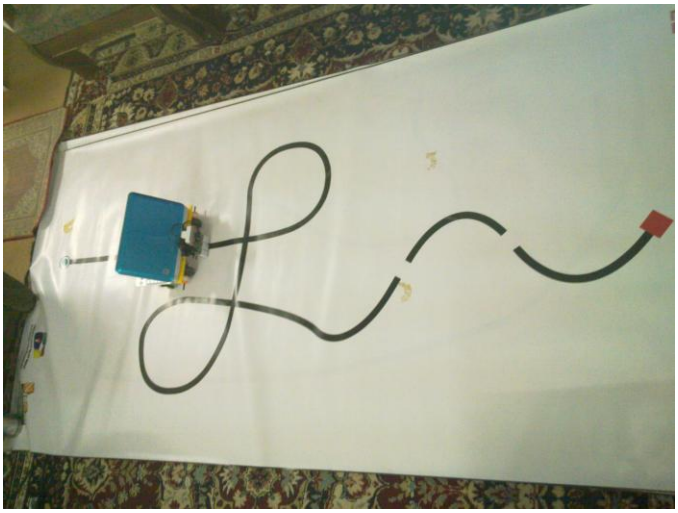

Fig. 14: Scenario 3: Following a path with cuts and loops

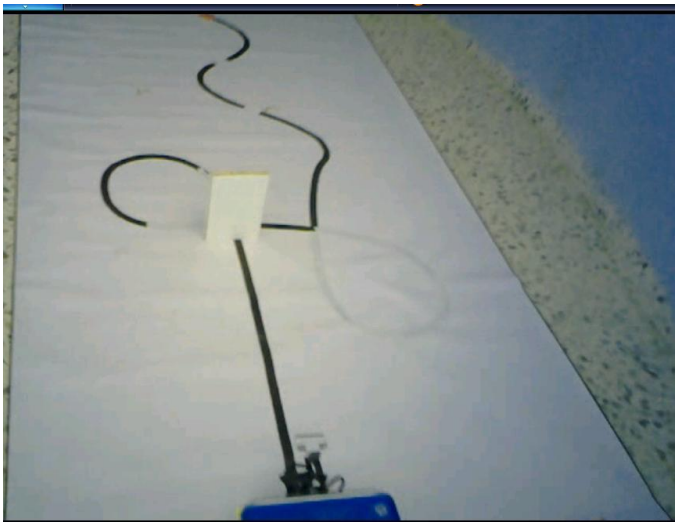

Fig. 15: Scenario 4: Following a path with cuts and one obstacle

Fig. 16: Scenario 5: Seeing the path from the robot's angle

References

[1] G. Pires, and U. Nunes, "A Wheelchair Steered through Voice Commands and Assisted by a Reactive Fuzzy-Logic Controller," *Journal of Intelligent and Robotics Systems*, vol. 34, pp. 301– 314, July 2002.

[2] L. Bento, G. Pires, and N. Nunes, "A Behavior Based Fuzzy Control Architecture for Path Tracking and Obstacle Avoidance," in *Proceedings of the 5th Portuguese Conference on Automatic Control, (Controlo 2002)*, Aveiro, Portugal, 2002 pp. 341–346.
[3] N. Giap, J.-H. Shin, and W.-H. Kim, "Adaptive Robust Fuzzy Control for Path Tracking of a Wheeled Mobile Robot," *Artificial Life and Robotics*, vol. 13, pp. 134–138, 2008.
[4] N. Liu, "Intelligent Path Following Method for Nonholonomic Robot Using Fuzzy Control," in *Second International Conference on Intelligent Networks and Intelligent Systems, 2009 (ICINIS '09)*, pp. 282–285, 2009.
[5] L. A. Zadeh, "Fuzzy sets," *Information and Control*, vol. 8, pp. 338–353, 1965.
[6] E. Mamdani, "Application of fuzzy logic to approximate reasoning using linguistic synthesis," *IEEE Transactions on Computers*, vol. C-26, no. 12, pp. 1182–1191, 1977.
[7] A. Katbab, "Fuzzy logic and controller design-a review," in *Proceedings of IEEE Southeastcon '95. 'Visualize the Future'*, Raleigh, NC, Mar. 1995, pp. 443–449.
[8] Y.-M. Park, U.-C. Moon, and K. Lee, "A self organizing fuzzy logic controller for dynamic systems using a fuzzy auto-regressive moving average (farma) model," *IEEE Transactions on Fuzzy Systems*, vol. 3, pp. 75–82, February 1995.
[9] R. Resende, A. Lavelha, A. Yamakami, and I. Bonatti, "A fuzzy algorithm to solve the problem of QoS unicast routing in IP networks," in *International Telecommunications Symposium*, 2006, pp. 856 –861.
[10] J. I. Corcau, and E. Stoenescu, "Fuzzy logic controller as a power system stabilizer," *International Journal of Circuits, Systems and Signal Processing*, vol. 1, no. 3, pp. 266-273, 2007.
[11] M. Roscia, d. Zaninelli, and g. Lazaroiu, "Sustainable urban model through fuzzy logic weights," International Journal of Energy And Environment, vol. 2, no. 1, pp. 66-72, 2008.
[12] S. G. Matthews, S. Coupland, and S.-M. Zhou, "An integrated stereo vision and fuzzy logic controller for following vehicles in an unstructured environment," in *The 2008 UK Workshop on Computational Intelligence, UKCI 2008*, De Montfort University, Leicester, UK, 2008, pp. 135–140.
[13] M. S. Islam, N. Amin, M. Zaman, and M. S. Bhuyan, "Fuzzy based PID controller using VHDL for transportation application," *International Journal of Mathematical Models and Methods in Applied Sciences*, vol. 2, no. 2, pp. 143-147, 2008.
[14] O. Hachour, "The proposed fuzzy logic navigation approach of autonomous mobile robots in unknown environments," *International Journal of Mathematical Models and Methods in Applied Sciences*, vol. 3, no. 3, pp. 204-218, 2009.
[15] R. U. Pedersen, *TinyOS Education with LEGO MINDSTORMS NXT*, Springer Berlin Heidelberg, September 2007, pp. 231–241.
[16] LEGO Group what is NXT? [Online]. Available: http://mindstorms.lego.com/en-gb/Wrappers/whatisnxt_wrapper.aspx, access date 20 March 2011.
[17] B. Bagnall. Maximum LEGO NXT: Building Robots with Java Brains Variant Press. 2007.
[18] *LEGO MINDSTORMS NXT Bluetooth Developer Kit*. LEGO Group, edition 1.00, 2006.

Mousa T. AL-Akhras obtained his B.Sc. and M.Sc. degrees in computer science from the University of Jordan, Amman, Jordan in 2000 and 2003, respectively. Then he worked as teaching assistant at the University of Jordan. He earned his Ph.D. degree in 2007 from De Montfort University, Leicester, UK. His Ph.D. specialization is artificial neural networks & communications.

He works as an Assistant Professor in the Computer Information Systems Department, King Abdullah II School for Information Technology at the University of Jordan since 2007. His research interests include problems in the area of artificial intelligence and particularly artificial neural networks. His research interests include Voice over IP, multimedia communication, robotics, genetic algorithm, fuzzy logic, and statistics. He is also interested in the area of electronic learning (e-learning), and mobile learning (m-learning).

Dr. AL-Akhras is a member of IEEE and he was elected as a secretary for general activities of the IEEE executive committee, Jordan Section, region 8 for the years 2010-2011. He was also elected as a vice-chair for Computational Intelligence/Computer Joint Societies Chapter, Jordan section for the years 2010-2011. He is also a member of IEEE CIS & RAS Societies. He is in the organizing and technical committees for a number of local and international conferences. Also, he serves as a reviewer and a member of the editorial board in a number of local and International Journals. He is a member of the Jordan Society for Scientific Research (JSSR), he also serves as a judge in the national and Arabic robotic contest (First LEGO League).

Mohammad O. Salameh obtained his B.Sc. degree in Computer Science from King Abdullah II School for Information Technology (KASIT), the University of Jordan in 2005, and then he received his M.Sc. Degree in Computer Science from Faculty of Graduate Studies and Scientific Research, Al-Balqa' Applied University in 2009.

He works as a Robotics Trainee and Programmer in the National Education Center for Robotics (NECR), Jubilee Center for Excellence in Education, Amman, Jordan. His research interests include Robotics Design and Programming, like LEGO Mindstorm & PIC-Microcontroller, Evolutionary Computation, Image processing and Mobile Phone applications.

Maha K. Saadeh obtained her B.Sc. degree in computer science from the University of Jordan, Amman, Jordan in 2009. Currently she is pursuing her M.Sc. degree from the same university. Her research interests are: Wireless networks, network security, image processing, and robotics.

Mohammed A. ALAwairdhi obtained his B.Sc. degrees in information Systems from King Saud University, Saudi Arabia in 1996. Then he received his M.Sc. degree in Computer Science from California State University, USA in 2000. He then worked as lecturer in computer & information programs department, Institute of Public Administration from April 2000 to June 2003. He then worked as a lecture, College of Computer and information Sciences, Al - Imam Muhammad ibn Saud Islamic University, April 2004 to September 2005. He earned his Ph.D. degree in 2009 from De Montfort University, Leicester, UK. His Ph.D. thesis title is "A Reengineering Approach for Software Systems Complying with the Utilization of Ubiquitous Computing Technologies".

He works as an Assistant Professor in the college of Computer and Information Sciences, Al - Imam Muhammad ibn Saud Islamic University since March 2010. He is also vice dean of Graduate Studies and Research, College of Computer and Information Sciences, Al - Imam Muhammad ibn Saud Islamic University since December 2010. His research interests include ubiquitous computing (sensor-based applications, sensor networks, smart environments, and RFID). Software engineering, evolution, and re-engineering. Business Process Reengineering: (IT role, modeling, simulation), and Human Computer Interaction (culture and cyberspace).