DESIGN OF A FUZZY LOGIC CONTROLLER FOR A WALL FOLLOWING ROBOT

Gabriel Chichi [1]
Faculty of Engineering, Environment and Computing
Coventry University
Coventry, England
chichig@uni.coventry.ac.uk

***Abstract*—In this paper, a Fuzzy Logic Controller (FLC) is designed for a PIONEER P3-DX mobile robot using the fuzzy Logic Toolbox in Matlab, primarily for wall following. The paper is arranged as follows; The first section contains the Design and Implementation of the FLC. The design justification is covered in the second section. In the third section, the performance analysis of the designed FLC is performed, Simulations are modelled V-rep and the results are discussed. The fourth section discusses the effectiveness of genetic fuzzy system FLCs in a literature review.**

I. DESIGN AND IMPLEMENTATION OF THE FUZZY LOGIC CONTROLLER

A. Fuzzy Logic Controller Overview

The Fuzzy Logic controller was designed in MATLAB, using the Fuzzy Logic Toolbox. It is designed to control a Pioneer P3-Dx mobile robot modelled in a V-Rep environment. The problem solved by the fuzzy logic controller is that of following a right wall at a constant distance. The FLC consists of three input functions; 'DistanceToForwardWall', 'DistanceToForwardRight' and 'DistanceToRightWall', with the corresponding outputs 'Left Velocity' and 'Right Velocity'. The FLC uses distance data, measured by the ultrasonic sensors to control the velocity in each wheel, thus controlling the steering.

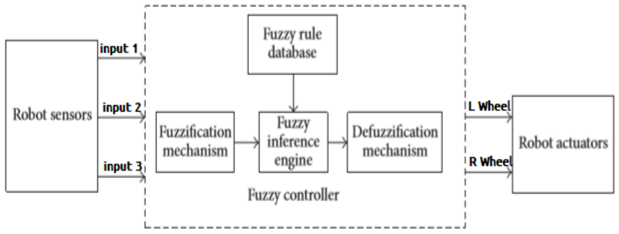


Figure 1. showing the structure of the system

*B. Braitenberg Vehicle (**BV**)*

The FLC was modelled with the principles of the Braitenberg Vehicle 2 "Fear", in mind [1]. BVs engage in autonomous motion based on data from sensors on the vehicle. They (the vehicles)

In Figure 2, Vehicles 2a and 2b are both BVs exhibiting phototaxis. 2a moves away from the light source and is said to be a 'Coward' (Fear), while vehicle 2b is said to be 'Aggressive' because it moves toward the light source [1]. Figure 3 shows the model used in this paper, the vehicle moves away from the wall rather than the sun in the same fashion as Vehicle 2a in figure 2.

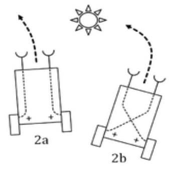


Figure 2 Braitenberg Vehicles 'Fear' and 'Aggression'

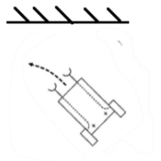


Figure 3 Modified 'Fear' Vehicle

The rules in the FLC control the actuators (wheels) based on the crisp data gotten from the ultrasonic sensors.

It utilizes the BV model, utilizing avoidance to ensure that there is never a collision with a wall [2]. Collisions are avoided by the rules set in the FLC based on input from sensory data captured from the ultrasonic sensors on the robot. The FLC rules are designed in a way that there is a threshold set right after the wall has been avoided, so the robot never strays too far away while it is engaging in wall following.

C. SIMULATION: Pioneer P3-DX Mobile Robot

This FLC designed in this study controls a pioneer P3-DX robot in a V-REP simulation environment.

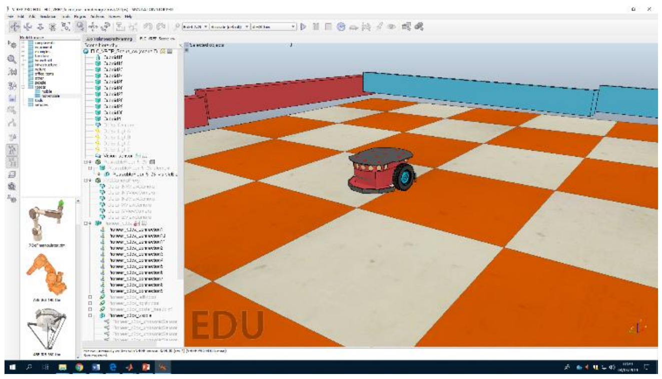


Figure 4 Pioneer 3-DX Mobile robot in V-Rep Environment

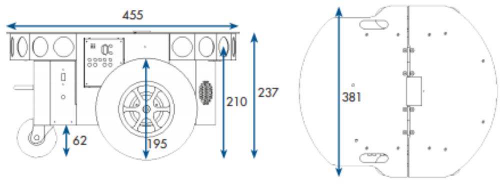


Figure 5 P3-DX Mobile Robot Dimensions in MM

TABLE I.
TABLE SHOWING PROPERTIES OF THE PIONEER P3-DX ROBOT

PIONEER P3-DX Mobile Robot Properties	
Dimensions	450x380x250
Sensors Used	Eight Ultrasonic Sensors in front
Sensory Coverage	180◦ (Front)
Rotation speed	300°/s
Max. speed	1.2 m/s
Supply voltage	5V @ 1.5A; 12V @ 2.5A

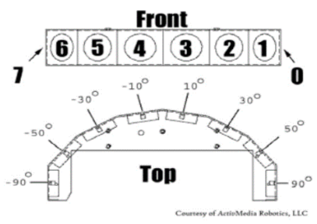


Figure 6 P3-DX Mobile Robot Sonar Ring

Only four ultrasonic sensors were utilized out of a possible 8, because the purpose of the study is a right wall following robot. Sensors 3, 4, 6 and 7 shown in Figure 6 were used. The mean of sensor 3 and 4 was compute to feed the 'DistanceToFrontWall' input in the FLC, while sensors 5 and 7 were used to feed 'DistanceToForwardRightWall' and 'DistanceToRightWall' respectively.

D. Screenshots of the FLC

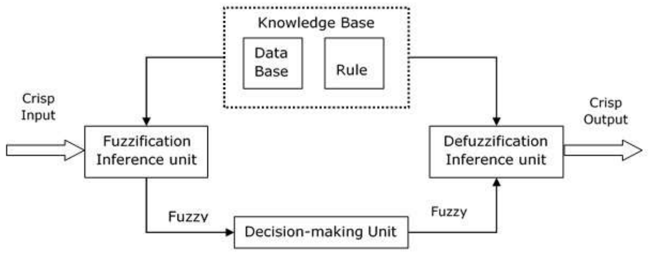


Figure 7 FIS Structure

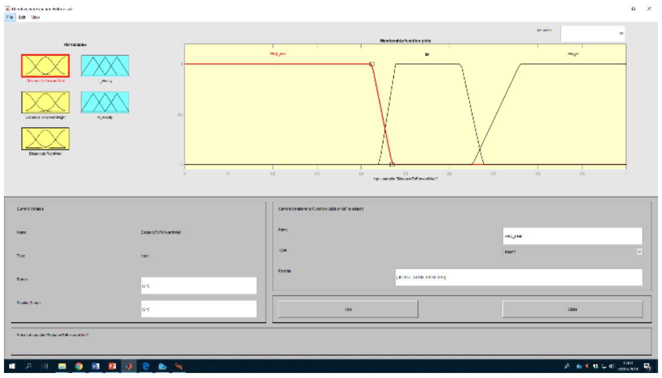


Figure 8 MFs for input 'DistanceToForwardWall', which correspond to the sets "Very near", 'far' and 'very far'

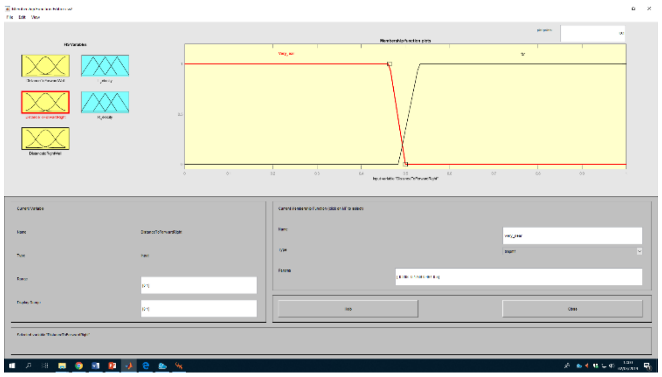


Figure 9 MFs for input 'DistanceToFowardRight', which correspond to the sets "Very near" and 'Very Far'

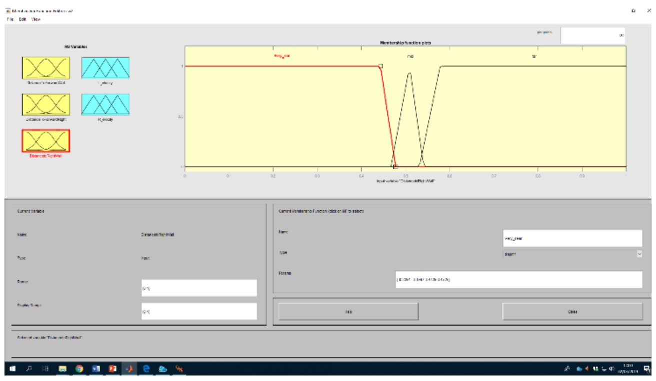


Figure 10 MFs for input 'DistanceToRightWall', which correspond to the sets "Very near", 'mid' and 'Far'

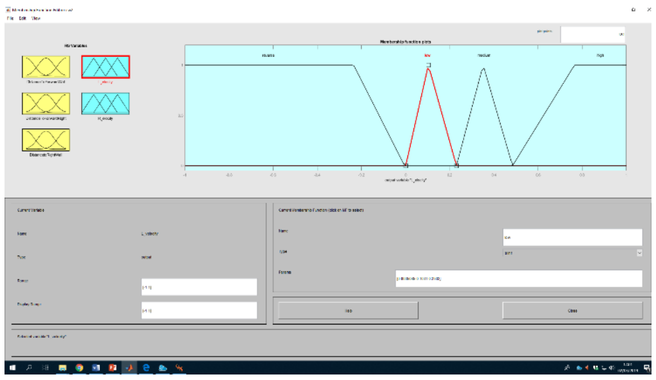


Figure 11 MFs for output 'L velocity', which correspond to the sets "Reverse", 'low' and 'medium' and 'high'

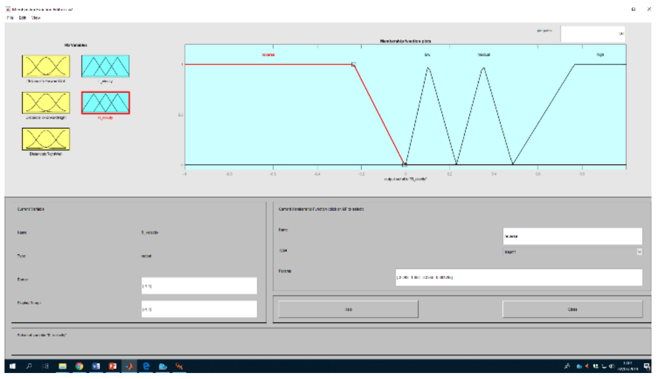


Figure 12 MFs for output 'R *velocity', which correspond to the sets "Reverse", 'low' and 'medium' and 'high'*

II. DESIGN JUSTIFICATION OF THE FUZZY LOGIC CONTROLLER

A. *Fuzzy Rules for Steering*

Initially an exhaustive method of rule selection was employed, all possible combinations of the linguistic terms of all inputs were written out and given steering outputs thought to be suitable at the time. The rules were summed up to 65 but tested very poorly in the V-Rep simulation, the robot failed to follow the right wall consistently, crashing into the walls and stopping abruptly. The robot also failed to turn corners and went often off course.

An iterative method of rule selection was used next. At the first iteration, the minimum rule needed for the robot to move (Rule one) was set and the simulation run, more rules were added with each iteration to enable navigation control. Rules were added and removed based on the performance of the robot in the simulation to improve steering at corners and maintaining a constant distance to the wall on the right.

At the end of the iterations, 20 rules in total were designed for the Fuzzy System. The rules help the robot navigate in the environment. Speed and steering are both achieved by the controlling the power supplied to each wheel of the robot. When the power supplied to both wheels is equal, the robot moves forward without turning. Steering is achieved by supplying unequal amounts of power to each wheel. The degree of steering is also dependent on the power disparity between wheels, the robot turns become sharper as the disparity increases. Listed below in Table II are the steering combinations used in the study.

TABLE II.
TABLE SHOWING STEERING USED IN THE FIS

Left Wheel Velocity	Right Wheel Velocity	Steering
Low	Low	Forward(slow Speed)
Medium	Medium	Forward (Normal Speed)
Reverse	Low	Hard Left
Low	High	Left
High	Low	Right

Rules 1, 4 and 15 propel the robot forward without turning at any angle. Rule 1 and 15 activate when there are no considerably close walls. The robot moves until a wall is detected, then aligns itself and begins the wall following process.

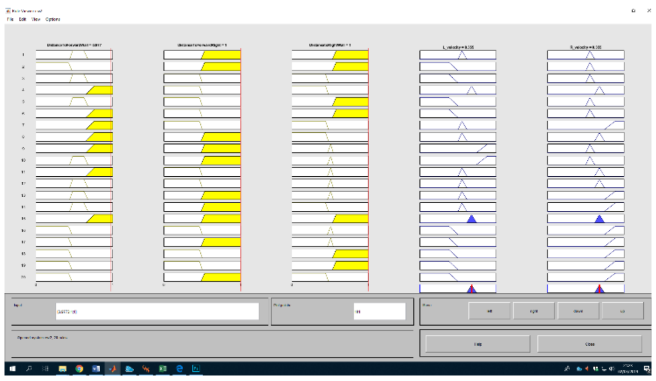


Figure 13 Rule activation for rules 1 and 15

Rules 2-6 and 16-20 activate whenever there is a wall in close proximity ('Very Near') that would cause a collision, the rules make the robot execute a 'Hard Left' turn. This is essential when avoiding and aligning to right walls. The rules also make getting out of tight corners easier, since the robot rotates on a single spot.

Front Wall	Front Right	Right Wall	LEFT WHEEL	RIGHT WHEEL	STEERING
FAR	FAR	FAR	LOW	LOW	FORWARD
VERY FAR	VERY NEAR	VERY NEAR	MEDIUM	MEDIUM	FORWARD
VERY FAR	FAR	FAR	MEDIUM	MEDIUM	FORWARD
VERY NEAR	FAR	FAR	REVERSE	LOW	HARD LEFT
FAR	VERY NEAR	VERY NEAR	REVERSE	LOW	HARD LEFT
FAR	VERY NEAR	FAR	REVERSE	LOW	HARD LEFT
VERY FAR	VERY NEAR	FAR	REVERSE	LOW	HARD LEFT
VERY NEAR	VERY NEAR	MID	REVERSE	HIGH	HARD LEFT
VERY NEAR	FAR	MID	REVERSE	HIGH	HARD LEFT
VERY NEAR	VERY NEAR	FAR	REVERSE	HIGH	HARD LEFT
VERY NEAR	VERY NEAR	FAR	REVERSE	HIGH	HARD LEFT
VERY NEAR	FAR	VERY NEAR	REVERSE	HIGH	HARD LEFT
VERY FAR	VERY NEAR	VERY NEAR	LOW	HIGH	LEFT
VERY FAR	FAR	VERY NEAR	LOW	MEDIUM	LEFT
VERY FAR	VERY NEAR	MID	LOW	MEDIUM	LEFT
FAR	VERY NEAR	MID	LOW	MEDIUM	LEFT
FAR	FAR	MID	LOW	HIGH	LEFT
FAR	FAR	VERY NEAR	LOW	HIGH	LEFT
VERY FAR	FAR	MID	HIGH	LOW	RIGHT
FAR	FAR	MID	HIGH	LOW	RIGHT

Figure 14 Rules sorted by steering

Rules 7-8 and 11-14 make the robot execute a left turn, while rules 9-10 make the robot execute a right turn. The steering functions 'Left' and 'Right' make sure the robot does not crash into the right wall and maintains a distance of at least 40 centimetres from the wall.

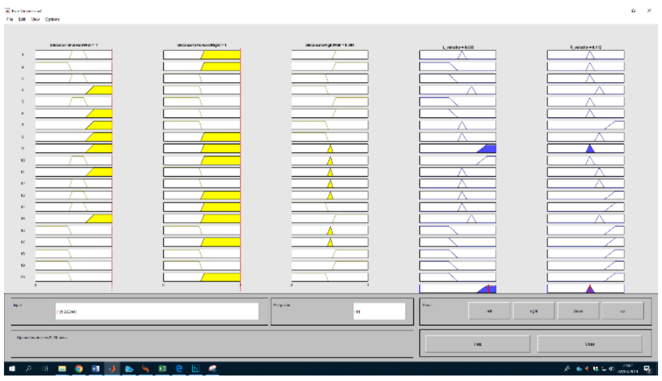


Figure 15 Rule activation for 'right'

B. *Fuzzy Membership Functions (MFs)*

Triangular and trapezoidal membership functions were used for the input and output functions. The membership functions in the inputs were non-identical because different ranges were needed for all three inputs. The sensor readings change very rapidly so MFs with steep gradients worked better with the rules. Maximum memberships were needed for MFs on either extreme ends so trapezoidal MFs were employed. The 'Mid' MF responsible for keeping the robot at a constant distance of at least 40 cm, so a triangular MF was used for its precision.

The Defuzzification method used is the centroid of area method, while maximum was used for Aggregation. The FLC was of the Mandani type.

C. DESCRIPTION AND EXPLANATION

III. ANALYSIS OF THE CONTROLLER

A. *Overview: Description of surface plot*

Figure 16 and 17 below show the relationship between DistancetoRightWall, DistanceToForwardWall and power to both motors. In Figure 16 the velocity to the right wheel increases when the distance to the right or forward wall is low, this is to prevent the robot from crashing into the wall and to turn at corners. In figure 17, the spike in velocity between MF 0.4-0.5 corresponds to the 'Mid' MF in DistancetoRightWall. This spike means that the robot maintains a constant distance to the wall, never straying and going off course.

Figures 18 and 19 show the surface plot showing the relationship between DistancetoForwardRight, DistanceToForwardWall and both motors. The power in the left wheel drops to a minimum when the forward wall is near so the robot can turn left.

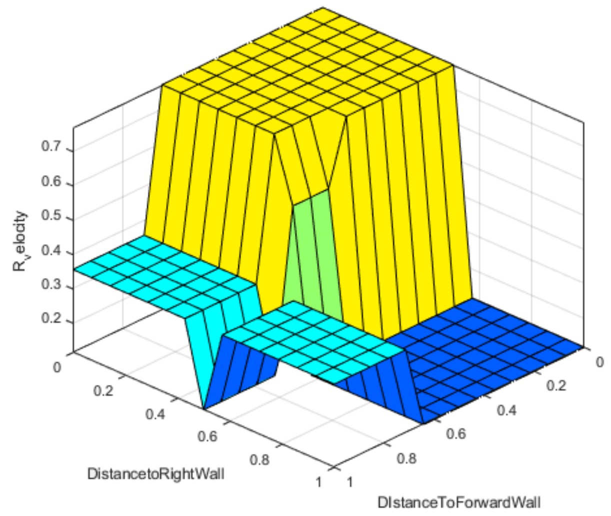


Figure 16 Surface plot showing the relationship between DistancetoRightWall, DistanceToForwardWall and R Velocity

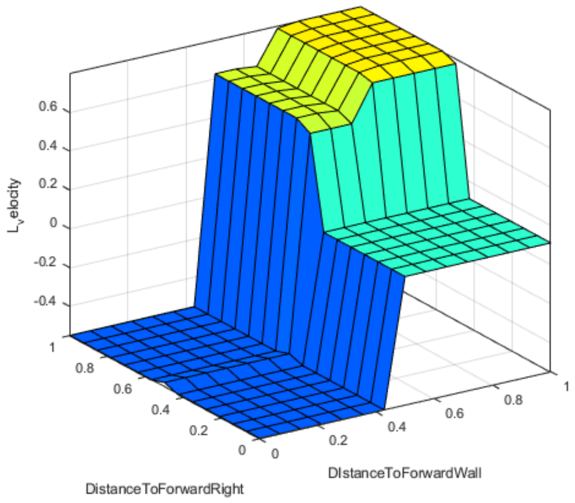


Figure 18 Surface plot showing the relationship between DistancetoForwardRight, DistanceToForwardWall and L Velocity

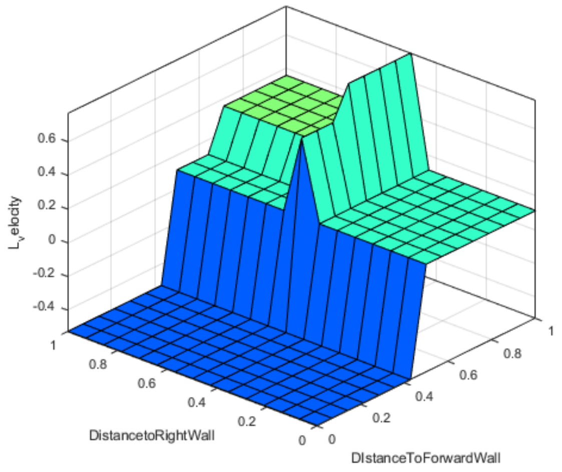


Figure 17 Surface plot showing the relationship between DistancetoRightWall, DistanceToForwardWall and L Velocity

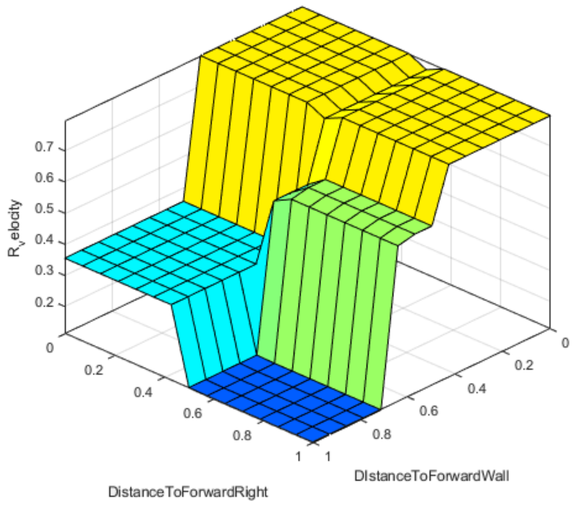


Figure 19 Surface plot showing the relationship between DistancetoForwardRight, DistanceToForwardWall and R Velocity

B. The Scene

The FLC created controlled a simulated Pioneer 3dx robot in the Virtual Robot experimentation platform (V-rep) [5]. The FLC was run in Matlab and was connected to the robot in V-rep via Matlab's Remote API. All simulations were performed in V-rep at a the default dt of 50ms. For a faster experience 100dt is advised, because the simulation starts acting erratically at 200dt.

The simulation took place on a 5x5 meter floor surrounded by walls. The walls were detectable by the ultrasonic sensors on the Pioneer 3DX robot.

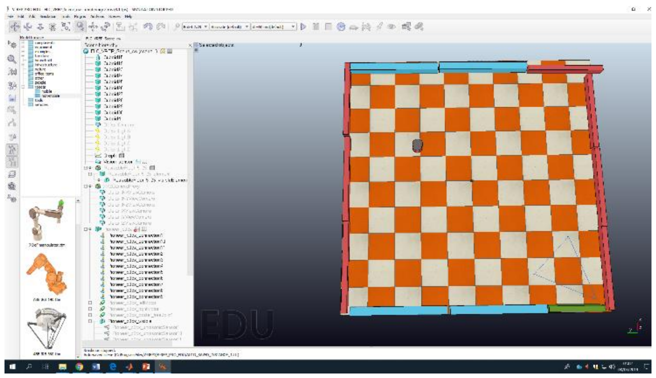
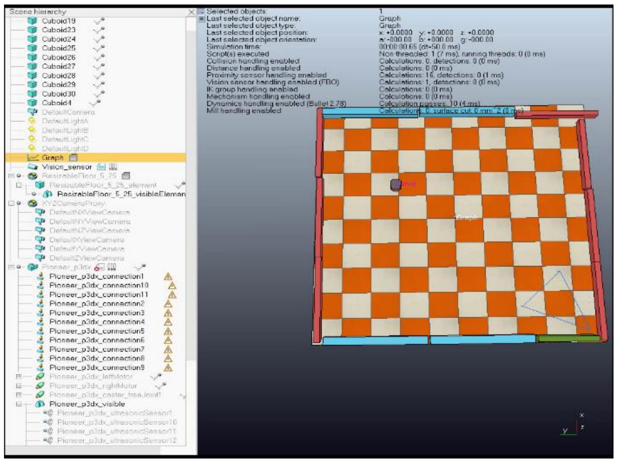


Figure 20. The scene of the simulation in V-rep, a 5x5 floor with surrounding walls.

C. The Simulation

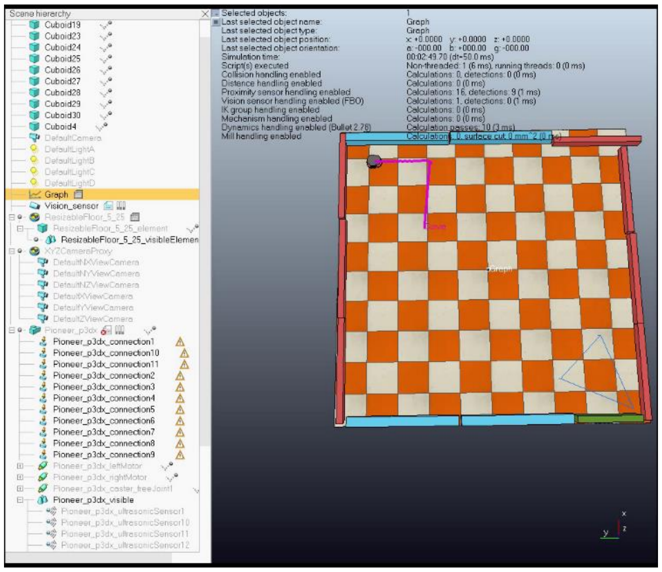
The purpose of this paper is to design a fuzzy logic controller to guide a robot to successfully follow the wall on the right.

At the beginning of the simulation, the robot was placed in an arbitrary position in the room. The robot moved forward, following instructions from the FLC (rules 1 and 15 activated). Upon detection of a wall in front, the robot slowed down, turned left and initiated the wall following sequence by aligning itself to the right wall. The robot then followed the wall up until its first corner where it made the necessary adjustments, turning the corner and realigning itself with the wall on its right side. The simulation is represented in Figures: xxx, a pink trail was added to the robot so the path could be easily traced. A link to the video file of the simulation will be put in the appendices and the FLC and V-rep scene file will be placed there so the simulation can be easily reproduced



```
Selected objects:                          1
Last selected object name:                 Graph
Last selected object type:                 Graph
Last selected object position:             x: +0.0000   y: +0.0000   z: +0.0000
Last selected object orientation:          a: -000.00   b: +000.00   g: -000.00
Simulation time:                           00:00:00.65 (dt=50.0 ms)
Script(s) executed                         Non-threaded: 1 (7 ms), running threads: 0 (0 ms)
Collision handling enabled                 Calculations: 0, detections: 0 (0 ms)
Distance handling enabled                  Calculations: 0 (0 ms)
Proximity sensor handling enabled          Calculations: 16, detections: 0 (1 ms)
Vision sensor handling enabled (FBO)       Calculations: 1, detections: 0 (0 ms)
IK group handling enabled                  Calculations: 0 (0 ms)
Mechanism handling enabled                 Calculations: 0 (0 ms)
Dynamics handling enabled (Bullet 2.78)    Calculation passes: 10 (4 ms)
Mill handling enabled                      Calculations: 0, surface cut: 0 mm^2 (0 ms)
```

Figure 21 The start of the simulation, the pioneer robot is at rest



```
Selected objects:                          1
Last selected object name:                 Graph
Last selected object type:                 Graph
Last selected object position:             x: +0.0000   y: +0.0000   z: +0.0000
Last selected object orientation:          a: -000.00   b: +000.00   g: -000.00
Simulation time:                           00:02:49.70 (dt=50.0 ms)
Script(s) executed                         Non-threaded: 1 (6 ms), running threads: 0 (0 ms)
Collision handling enabled                 Calculations: 0, detections: 0 (0 ms)
Distance handling enabled                  Calculations: 0 (0 ms)
Proximity sensor handling enabled          Calculations: 16, detections: 9 (1 ms)
Vision sensor handling enabled (FBO)       Calculations: 1, detections: 0 (1 ms)
IK group handling enabled                  Calculations: 0 (0 ms)
Mechanism handling enabled                 Calculations: 0 (0 ms)
Dynamics handling enabled (Bullet 2.78)    Calculation passes: 10 (3 ms)
Mill handling enabled                      Calculations: 0, surface cut: 0 mm^2 (0 ms)
```

Figure 22 The pioneer robot at the start of turning a corner after alignment with the right wall

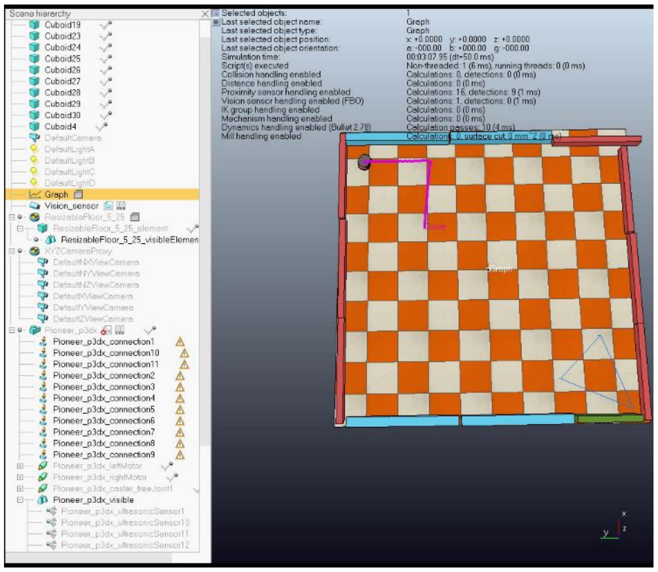

```
Selected objects:                          1
Last selected object name:                 Graph
Last selected object type:                 Graph
Last selected object position:             x: +0.0000   y: +0.0000   z: +0.0000
Last selected object orientation:          a: -000.00   b: +000.00   g: -000.00
Simulation time:                           00:03:07.95 (dt=50.0 ms)
Script(s) executed                         Non-threaded: 1 (6 ms), running threads: 0 (0 ms)
Collision handling enabled                 Calculations: 0, detections: 0 (0 ms)
Distance handling enabled                  Calculations: 0 (0 ms)
Proximity sensor handling enabled          Calculations: 16, detections: 9 (1 ms)
Vision sensor handling enabled (FBO)       Calculations: 1, detections: 0 (1 ms)
IK group handling enabled                  Calculations: 0 (0 ms)
Mechanism handling enabled                 Calculations: 0 (0 ms)
Dynamics handling enabled (Bullet 2.78)    Calculation passes: 10 (4 ms)
Mill handling enabled                      Calculations: 0, surface cut 0 mm^2 (0 ms)
```

Figure 23 Pioneer robot aligning with the wall after turning at the corner

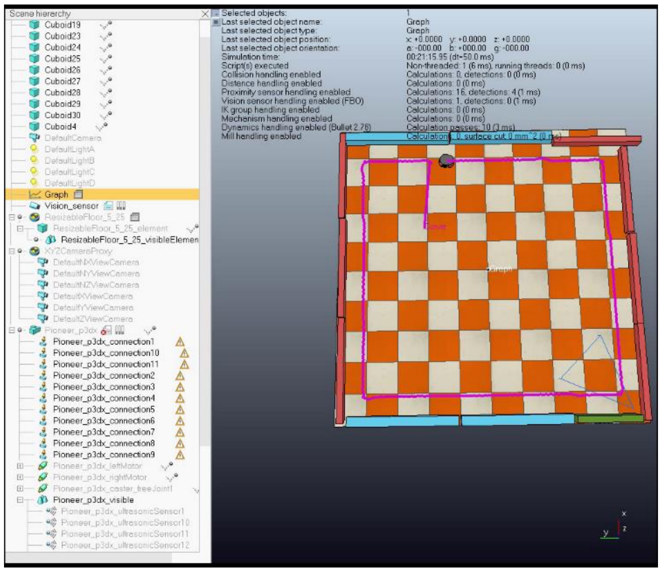

```
Selected objects:                          1
Last selected object name:                 Graph
Last selected object type:                 Graph
Last selected object position:             x: +0.0000   y: +0.0000   z: +0.0000
Last selected object orientation:          a: -000.00   b: +000.00   g: -000.00
Simulation time:                           00:21:15.95 (dt=50.0 ms)
Script(s) executed                         Non-threaded: 1 (6 ms), running threads: 0 (0 ms)
Collision handling enabled                 Calculations: 0, detections: 0 (0 ms)
Distance handling enabled                  Calculations: 0 (0 ms)
Proximity sensor handling enabled          Calculations: 16, detections: 4 (1 ms)
Vision sensor handling enabled (FBO)       Calculations: 1, detections: 0 (1 ms)
IK group handling enabled                  Calculations: 0 (0 ms)
Mechanism handling enabled                 Calculations: 0 (0 ms)
Dynamics handling enabled (Bullet 2.78)    Calculation passes: 10 (3 ms)
Mill handling enabled                      Calculations: 0, surface cut 0 mm^2 (0 ms)
```

Figure 25 Pioneer Robot after completing a full path around the roo

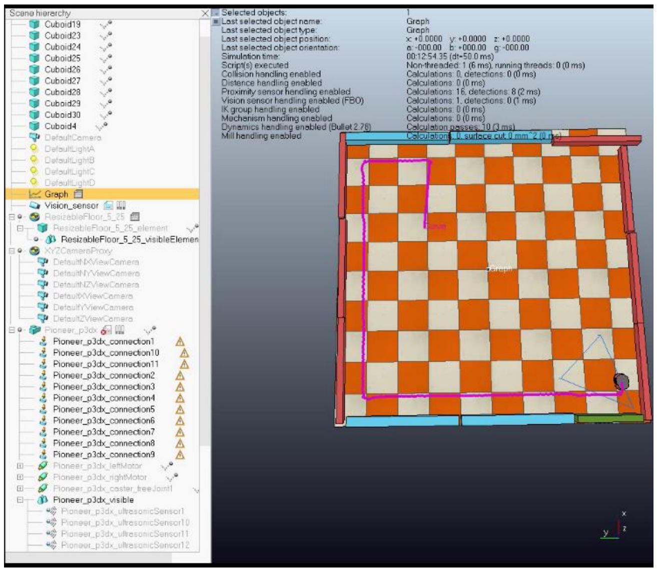
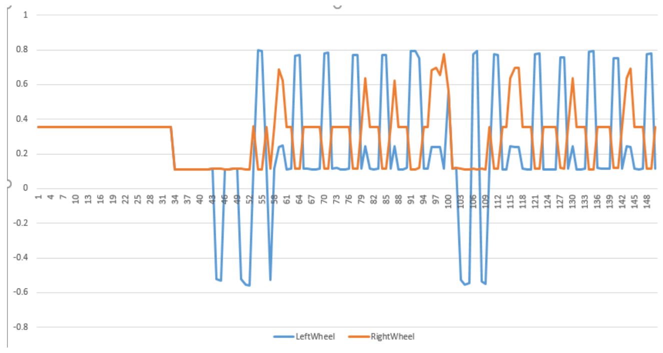

```
Selected objects:                          1
Last selected object name:                 Graph
Last selected object type:                 Graph
Last selected object position:             x: +0.0000   y: +0.0000   z: +0.0000
Last selected object orientation:          a: -000.00   b: +000.00   g: -000.00
Simulation time:                           00:12:54.35 (dt=50.0 ms)
Script(s) executed                         Non-threaded: 1 (6 ms), running threads: 0 (0 ms)
Collision handling enabled                 Calculations: 0, detections: 0 (0 ms)
Distance handling enabled                  Calculations: 0 (0 ms)
Proximity sensor handling enabled          Calculations: 16, detections: 8 (2 ms)
Vision sensor handling enabled (FBO)       Calculations: 1, detections: 0 (1 ms)
IK group handling enabled                  Calculations: 0 (0 ms)
Mechanism handling enabled                 Calculations: 0 (0 ms)
Dynamics handling enabled (Bullet 2.78)    Calculation passes: 10 (3 ms)
Mill handling enabled                      Calculations: 0, surface cut 0 mm^2 (0 ms)
```

Figure 24 Pioneer robot halfway around the room

Figure 26 Graph showing power supplied to wheels after 150 iterations



IV. LITERATURE REVIEW

Fuzzy logic systems can be combined with genetic algorithms to form fuzzy genetic algorithms. Genetic algorithms are inspired by occurrences in nature. They are very successful in solving optimization problems by utilizing bio-inspired operators such as mutation, crossover and selection [6].

Genetic algorithms can be used to construct the rules of the fuzzy controller [7] or used to optimize already created membership functions to produce give optimal results [[8]

Knowledge bases can be learnt using evolutionary algorithms with three main approaches: Michigan, Pittsburgh, and IRL [11]. The solutions generated by genetic algorithms are usually very robust, and they are a very powerful tool for the generation of fuzzy rule base, optimization of fuzzy rule bases, generation of membership functions, and tuning of membership functions [10].

V. APPENDIX

TABLE III.
TABLE SHOWING LINGUISTIC TERMS FOR THE INPUT "DISTANCETOFORWARDWALL"

Linguistic Term	Distance
Very Near	<47cm
Far	44.09 – 67.62cm
Very Far	>65.21cm

TABLE IV.
TABLE SHOWING LINGUISTIC TERMS FOR THE INPUT "DISTANCETOFORWARDRIGHT"

Linguistic Term	Distance
Very Near	<50cm
Far	>48.41cm

TABLE V.
TABLE SHOWING LINGUISTIC TERMS FOR THE INPUT "DISTANCETORIGHTWALL"

Linguistic Term	Distance
leVery Near	<47.76cm
Mid	46.7 – 54.1cm
Far	>53.13

TABLE VI.
TABLE SHOWING LINGUISTIC TERMS FOR THE OUTPUTS "L_VELOCITY" AND "R_VELOCITY"

Linguistic Term	Wheel Velocity
Reverse	<0
Low	0-0.23
Medium	0.23-0.48
High	>0.48

Video link to Simulation Run.
Googledrive containing simulation materials

REFERENCES

[1] V. Braitenberg, Vehicles, Experiments in Synthetic Psychology. Cambridge, Mass.: M.I.T.P., 1986.
[2] https://en.wikipedia.org/wiki/Braitenberg_vehicle
[3] Yang, X., Patel, R.V.
[4] Moallem, M. J Intell Robot Syst (2006) 47:101. https://doi.org/10.1007/s10846-006-9055-3
[5] [5] Coppeliarobotics.com. (2019). Coppelia Robotics V-REP: Create. Compose. Simulate. Any Robot.. [online] Available at: http://www.coppeliarobotics.com/ [Accessed 3 Mar. 2019].
[6] Mitchell, Melanie (1996). An Introduction to Genetic Algorithms. Cambridge, MA: MIT Press. ISBN 9780585030944.
[7] Chia-Feng Juang and Chia-Hung Hsu, "Reinforcement Ant Optimized Fuzzy Controller for Mobile-Robot Wall-Following Control", IEEE Transactions on Industrial Electronics, vol. 56, no. 10, pp. 3931-3940, 2009. Available: 10.1109/tie.2009.2017557.
[8] F. Herrera, M. Lozano and J. Verdegay, "Tuning fuzzy logic controllers by genetic algorithms", International Journal of Approximate Reasoning, vol. 12, no. 3-4, pp. 299-315, 1995. Available: 10.1016/0888-613x(94)00033-y [Accessed 4 March 2019].
[9]] O. Cordo´n, F. Herrera, F. Hoffmann, L. Magdalena, Genetic fuzzy systems: evolutionary tuning and learning of fuzzy knowledge bases, Advances in Fuzzy Systems—Applications and Theory, vol. 19, World Scientific, 2001
[10] [2001, O. Cordón, F. Herrera, F. Gomide, F. Hoffmann and L. Magdalena, Ten years of genetic-fuzzy systems: a current framework and new trends, Proceedings of Joint 9th IFSA World Congress and 20th NAFIPS International Conference, pp. 1241–1246, Vancouver - Canada, 2001.