Jurnal Sains Informatika Terapan (JSIT)
E-ISSN: 2828-1659, Volume: 5, Issue: 1, Month: February, Year: 2026
Page: 347 - 353, Available online at: https://rcf-indonesia.org/home/

FUZZY STEERING CONTROL FOR WALL-FOLLOWING BEHAVIOR OF A MOBILE ROBOT IN WEBOTS SIMULATION

Danu Jaya Saputro [1], Siti Aisyah [2]
[1] Jurusan Teknologi Rekayasa Manufaktur dan Mekatronika, Politeknik Manufaktur Bandung, Bandung
[2] Program Studi Sains Data, Universitas Insan Cita Indonesia, Jakarta
Corresponding Author: [1] danu@polman-bandung.ac.id

Article Info

Article history:

Received: Jan 08, 2026
Revised: Feb 10, 2026
Accepted: Feb 18, 2026
Published: Feb 20, 2026

Keywords:

Wall-following robot
Webots simulation
Fuzzy logic
Proximity sensors

ABSTRACT

Navigation logic plays a crucial role in enabling mobile robots to move safely within structured environments. This study presents a simulation of a wall-following robot to navigate along an arena while avoiding frontal obstacles and the sides of the wall. The simulation is implemented in the Webots environment, where the E-puck robot utilizes infrared proximity sensors with fuzzy logic control algorithm. Initially, the crisp distance measurements obtained from sensors PS5, PS6, and PS7 are fuzzified into linguistic variables, namely Near, Medium, and Far. Based on the defined input and output membership functions, the fuzzy controller determines appropriate steering actions, including strong turning and forward motion. After a thorough fuzzy logic calculation, the result is a steering control of +0.183, which makes the robot slightly turn right. The proposed approach evaluates the relationship between sensor thresholds and the steering velocity control, demonstrating the effectiveness of fuzzy inference in regulating wall-following behavior.

1. INTRODUCTION

The autonomous navigation is essential to any mobile robot tasks including real world implementation like exploration and industrial use [1]. One of the most common robotics applications used in teaching, research, and development is a Wall following robot which the task is to travel along the predetermined boundary in a certain area[2]. While mobile robot has a wide range of applications, specifically in manufacturing field, the robot can distribute parts or materials from one point to another. This material handling application requires certain position and orientation recognized by the robot to perform pick and place operation. The renowned of industrial mobile robot is called Automated Guided Vehicles (AGV) that is programmed by the worker to follow the guided paths to load, transfer, and unload materials [3].

Putting a real-world condition of a robotics project into a simulation raises many benefits. The simulation system can identify potential obstacles and benefit from the use of different sensors, including range, vision, and force [4]. Additionally, to avoid high costs in configuring robotic systems, simulation offers excellent features of testing and experimenting to control disturbance in the physical world [5]. The provided benefits allow the simulator to test and evaluate the performance of advanced robot control algorithms [6].

This research employs a mini mobile robot called e-puck that is carried out through a robotics CAD software namely Webots. The purpose of the software development is to provide an open-source simulation for university education and corporate research centers. Undertaking simulation in robotics software encourage developers to take less time in obtaining the precise control of the robots, and to examine several constraints before turning the simulation into a physical robot. At the end of the research stage, it will eventually be benefit of fast prototyping of real robot developments.

Webots software enables users to carry out four fundamental stages in the robotic project creation as displayed in image 1. The initial phase is the modeling stage, where robotics designers can create the physical body of the robot as well as the robot simulation arena. Users can modify the physical model of the robot by arranging building blocks and organizing the properties like color, shape, and sensors. Moreover, the physical parameters of the robot's objects can be configured accordingly to support its simulation through features like mass distribution, bounding objects, friction, etc.

After the modeling stage is accomplished, users can continue to the programming stage. Here, the behavior of the robot is configured to read the measurable value from the sensors' reading and execute the commands through actuators' action. The third stage is the simulation stage, in which the virtual robot will perform the program around the robot arena. To achieve a proper simulation, it is essential to examine, fix, and improve the program repetitively. Lastly, the fourth stage is to transform the simulation into a real robot. If the controlled program of the robot is undertaken carefully and is calibrated appropriately, then it will perform roughly the same as the virtual robot simulation [7].

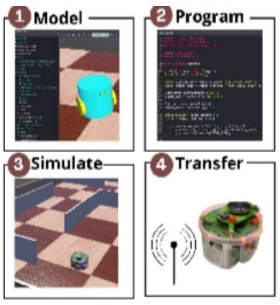


Figure 1. Four fundamental stages in Webots

2. RELATED RESEARCH

2.1. Wall-following robot

A wall-following robot is a mission where a mobile robot equipped with ultrasonic sensors navigates through an environment. The circumstance generally occurs where the robot partially recognises the environment, and without any preplanned path. The robot does not necessarily need to know and model the environment. If a robot can perform reactive navigation, in which it can sense along the wall, then it can achieve its mission [8].

The primary task of a wall-following robot is to move forward until it meets the boundary of objects, such as an arbitrarily shaped wall, within the determined arena. By utilizing a distance-ranging sensor, the robot can maintain a safe distance while closely surrounding the wall [2]. Technically, the number of sensors attached can vary depending on the type of robot. Afterward, the sensor readings obtained can be utilised to predict what moves of the robot will come next [9]. Following that, the robot will spin on itself to the left or right side to be perpendicular to the wall. This cycle repeats continuously while exploring the rather unknown environment [7].

2.2. Robot Model

The mobile robot chosen is the E-puck model developed by EPFL University in Switzerland. The e-puck robot is installed with several component devices, including stepper motors, LEDs, accelerometers, sensors, and a camera. An e-puck employs two stepper motors, on which the two wheels are mounted. These electrical motors can break a full rotation into a maximum of 1000 steps. To support this feature, an incremental encoder is utilized to count the number of steps. Additionally, the e-puck robot is equipped with devices such as LEDs to emit light for state feedback, accelerometers to measure the total force applied to the robot, an object collision detector, and a camera to observe the environment. Most importantly, the robot has 8 infrared sensors (IR) that can predict the distance by measuring the light of the nearby environment [7].

Prior research employs the e-puck robot to move along the provided curved black line on the white floor. To achieve its mission, the e-puck is programmed to gain infrared sensor readings to follow the alignment with the light reflection produced by the black line [10]. Moreover, [11] stated that this robot is fully observable and suited to any stochastic, sequential, and continuous experiment. The conducted research utilizes the e-puck robot to perform A* heuristic search, where it can connect links on each subsequent checkpoint between forward and rotation movements.

Figure 2. E-puck robot

II.3. Sensor

A component device that is significantly important in this research is the eight infrared sensors (IR) attached to the e-puck robot. Basically, sensors are frequently used for the detection process while conducting robot measurement and control [12]. More specifically, the infrared sensors can be used to accept the range of visible light and to count the amount of light the robot receives from the surrounding environment. Alternatively, they are very useful for detecting obstacles ahead. By producing the infrared light, the IR sensors will quantify the bouncing light when there is a nearby object. [7]. In this research, the e-puck utilizes the proximity sensor (PS5, PS6, PS7) to determine the upcoming direction of the robot.

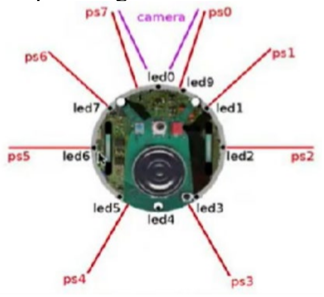
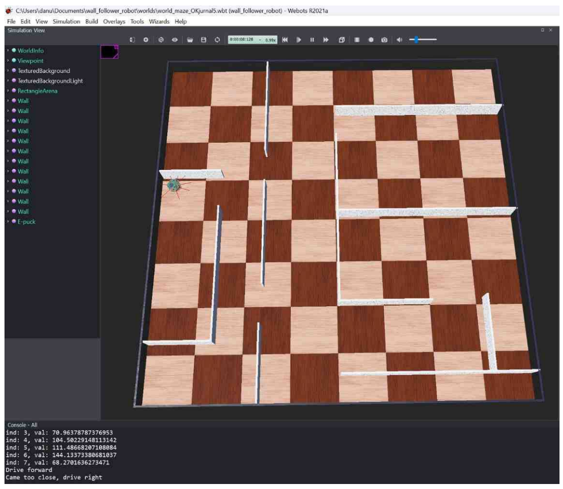


Figure 3. Infrared sensor (IR) on E-puck robot

2.4. Robot Arena

The world file in this experiment contains the whole environment of the simulation, including the robot shape, the form of the ground, the form of the wall and the obstacle shape. Additionally, there are two window features, namely a simulation window to represent the 3D simulation and a robot window to display the e-puck sensor and actuator values. Furthermore, the robot arena in this research is designed as a labyrinth that allows the e-puck robot to traverse along the wall and avoid the obstacles while obtaining the infrared sensor readings in the output console.



```
Console - All
ind: 3, val: 70.96378787376953
ind: 4, val: 104.50229148113142
ind: 5, val: 111.48668207108084
ind: 6, val: 144.13373380681037
ind: 7, val: 68.2701636273471
Drive forward
Came too close, drive right
```

Figure 4. Robot arena

2.5. **Fuzzy Logic**

Fuzzy logic is a computational approach to translate a quantity between 0 and 1 expressed using linguistic terms because it is similar to the way humans think. This logic system can represent human knowledge in mathematical form based on the experiment conducted [12]. Fuzzy logic is generally chosen due to its benefits, including low cost and simple control. The displayed triangular curve can determine to what extent a value is right or wrong [13]. Additionally, a fuzzy model can be used to interpret imprecise information and uncertain data. In more detail, the fuzzy model depicts the gradual transition from membership to non-membership objects, from 0 to 1. This approach is illustrated in Image 5, where the fuzzy set membership function defines "fit" according to the "weight" input. The more weight is gained, the closer the fit goes up from 0 to 1 and vice versa [14].

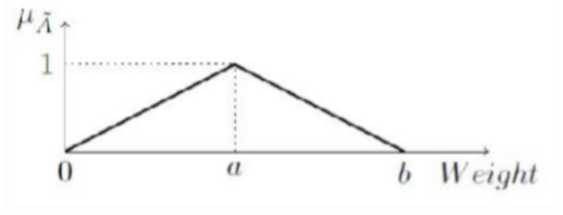


Fig 5. The illustration of Fuzzy membership function

3. RESEARCH METHODS

In fuzzy logic control, several operational steps include fuzzification, rule base, and defuzzification. Figure 6 is a diagram block of a fuzzy logic control. The Sugeno fuzzy method is utilised in this research, where a linguistic rule determines the control action that needs to be executed in response to the measured sensor value [15].

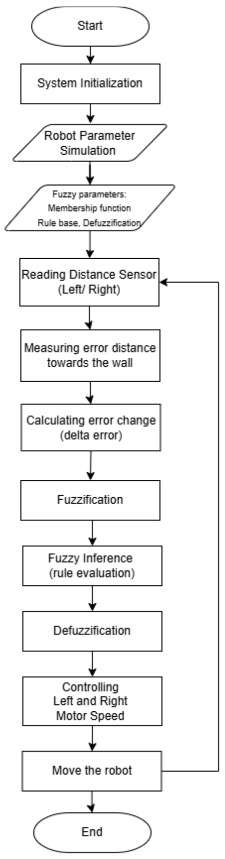

Start
System Initialization
Robot Parameter Simulation
Fuzzy parameters: Membership function Rule base, Defuzzification
Reading Distance Sensor (Left/ Right)
Measuring error distance towards the wall
Calculating error change (delta error)
Fuzzification
Fuzzy Inference (rule evaluation)
Defuzzification
Controlling Left and Right Motor Speed
Move the robot
End

Figure 6. Wall following robot flowchart

a. Fuzzification

To initiate the fuzzy logic, the first stage involves determining the crisp input value. The fuzzy value measurement is affected by the membership function designed by the experts. Afterward, the mapping of real or crisp values can be identified as Near (N), Medium (M), and Far (F), which reflects the principle of fuzzification. Moreover, it is imperative to define these linguistic terms using triangular and trapezoidal membership functions to ensure smooth transitions between different distance conditions.

i. Membership function input

Input Variable (Proximity Sensors)

• Left Distance (LD)	: PS5
• Left-Front Distance (LFD)	: PS6
• Front-Distance (FD)	: PS7

Domain sensor: $x \in [0,160]$

Linguistic Terms

- Near (N)
- Medium (M)
- Far (F)

Mathematics equation
Near (Trapezoidal)

$$\mu N(x) = \begin{cases} 0, & x \leq 80 \\ \frac{x-80}{30}, & 80 < x \leq 110 \\ 1, & x > 110 \end{cases} \quad (1)$$

Medium (Triangular)

$$\mu M(x) = \begin{cases} 0, & x \leq 50 \\ \frac{x-50}{30}, & 50 < x \leq 80 \\ \frac{110-x}{30}, & 80 < x \leq 110 \\ 0, & x > 110 \end{cases} \quad (2)$$

Far (Trapezoidal)

$$\mu F(x) = \begin{cases} 1, & x \leq 50 \\ \frac{80-x}{30}, & 50 < x \leq 80 \\ 0, & x > 80 \end{cases} \quad (3)$$

ii. Membership function output

Table 1. Linguistic Terms Output

Term	Meaning	Parameter
SL	Strong Left	(-1.0, -1.0, -0.6, -0,5)
L	Left	(-1.0, -0.5, 0)
F	Forward	(-0.5, 0, 0.5)
R	Right	(0, 0.5, 1.0)
SR	Strong Right	(0.5, 0.6, 1.0, 1,0)

iii. The steering control output
The steering control output is defined as the difference between the left and right wheel velocities. A positive ΔV results in a right turn, while negative ΔV causes a left turn. This convention is adopted to ensure consistency with the kinematic behavior of the differential drive robot.

$$\Delta V = V\text{left} - V\text{right} \quad (4)$$

Table 2. Steering Control Output

ΔV	Condition	Direction
ΔV < 0	V_left < V_right	Turn left
ΔV > 0	V_left > V_right	Turn right
ΔV = 0	equal	Move forward

b. Rule Base
Following the prior step, the rule base is responsible for collecting fuzzy logic rules based on human knowledge to connect input and output variables. Then, the created rules base will proceed the fuzzy logic input to obtain a decision. Thus, by deciding conclusion of fuzzy output, the collected rules of fuzzy output and membership function have been determined.

In this research, the mobile robot uses a proximity sensor to read the distance to the wall. If the sensor detects the wall, the original red sensor line will change into a green line, as illustrated in Figure 7.
Prox_sensor[ind] = robot->getDistanceSensor(ps_name);

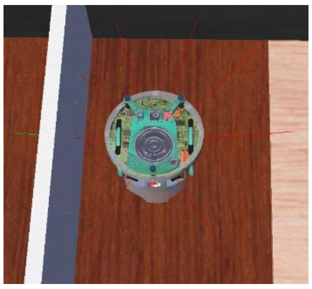
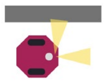
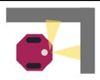
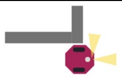
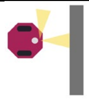


Figure. 7. Wall detection by proximity sensors
Following that, the action of the wall-following robot is classified into four categories, depicted as follows:

Table 3. Wall Following Robot Direction

No	Wall position		Illustration	Direction
1	Left wall	-		Move forward
2	Left wall	Front wall		Turn right
3	-	-		Turn left
4	-	Front wall		Turn right

To accomplish all directions, there are three conditions for several proximity sensors, as follows:

```
left_wall   = prox_sensors[5]->getValue() > 80;
left_corner = prox_sensors[6]->getValue() > 80;
front_wall  = prox_sensors[7]->getValue() > 80;
```

The output variable of the fuzzy controller is the steering control, represented by the difference in wheel velocities. Five linguistic terms are defined for the output: Strong Left, Left, Forward, Right, and Strong Right.

Table 4. Fuzzy Rule Base for Wall-Following Robot

Rule	LD	LFD	FD	Output (ΔV)	Robot Action
R1	Medium	Far	Far	Forward	Move forward
R2	Near	Far	Far	Right	Follow left wall
R3	Far	Far	Far	Left	Return wall
R4	Medium	Medium	Near	Right	Avoid front wall
R5	Near	Medium	Near	Strong Right	Sharp turn right

c. Defuzzification

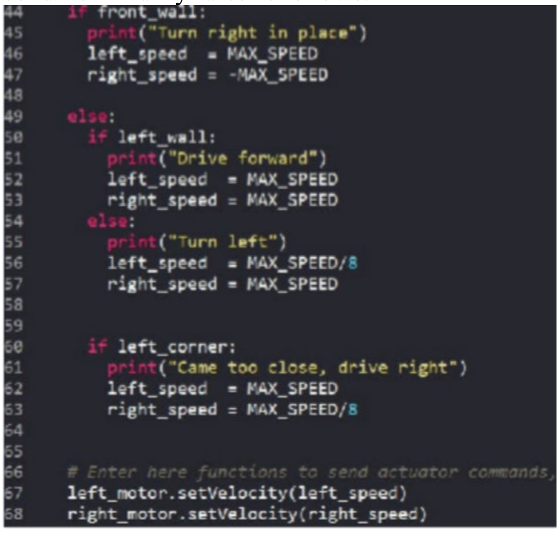
The defuzzification process will convert fuzzy output into a crisp value. The output of the membership function designs this process. It can use the centroid or the average weighted technique [15].

This research employs the Mamdani inference technique, where the fuzzy rules are evaluated using the minimum operator, and the aggregation process uses the maximum operator. The final crisp output is obtained through the centroid defuzzification method. The fuzzy rule is represented with syntax "IF ANTECEDENT …THEN CONSEQUENT…". "If antecedent" illustrates the condition of a system obtained from the input variable, while "then the consequent" is the output variable.

a. Input Variable = label (the distance of the left sensor = near), meaning the distance of the right sensor is the input variable, and "near" is a label of the membership function.

b. Output Variable = label (motor acceleration = strong right), meaning the motor acceleration is the output variable, and "strong right" is a label of the membership function.

In this wall-following simulation, it utilizes nested conditional fuzzy rules as follows:



```
    if front_wall:
      print("Turn right in place")
      left_speed  = MAX_SPEED
      right_speed = -MAX_SPEED

    else:
      if left_wall:
        print("Drive forward")
        left_speed  = MAX_SPEED
        right_speed = MAX_SPEED
      else:
        print("Turn left")
        left_speed  = MAX_SPEED/8
        right_speed = MAX_SPEED


      if left_corner:
        print("Came too close, drive right")
        left_speed  = MAX_SPEED
        right_speed = MAX_SPEED/8


    # Enter here functions to send actuator commands,
    left_motor.setVelocity(left_speed)
    right_motor.setVelocity(right_speed)
```

Figure 8. Conditions of Fuzzy logic rules

4. RESULTS AND DISCUSSION

The proposed fuzzy wall-following controller was evaluated through simulation in a structured indoor environment consisting of straight walls and corners. The mobile robot was equipped with proximity sensors to measure the distance to the surrounding walls and obstacles. Here is a deliberate step of fuzzy logic to determine the mobile robot's direction.

i. Sensor Condition (Crisp Input)

Left sensor (S_L)	= 146
Left Front Sensor (S_LF)	= 81
Front sensor (S_F)	= 0
Domain sensor	: [1,160]

ii. Input Membership Function (Parameter)

Table 5. Input Membership Function

Term	Parameter
Far	(0, 0, 50,80)
Medium	(50, 80, 110)
Near	(80, 110, 160, 160)

iii. Fuzzification

a. Left Sensor = 146
- Far , µF(146)=0, (146>80)
- Medium, µM(146)=0, (146>110)
- Near, µN(146)=1, (146>110)

Then SL = Near (1.0)

b. Left-Front Sensor = 81
- Far, µF(81)=0, (81>80)
- Medium, 80 < x ≤ 110
  $$\mu M(81) = \frac{110-81}{30} = \frac{29}{30} = 0{,}967$$
- Near, 80 < x ≤ 110
  $$\mu M(81) = \frac{81-80}{30} = \frac{1}{30} = 0{,}03$$

Then SF = Medium (0.967) + Near(0.03)

c. Front Sensor = 70
- Far, 50 < x ≤ 80
  $$\mu F(70) = \frac{80-70}{30} = \frac{10}{30} = 0{,}33$$
- Medium, 50 < x ≤ 80
  $$\mu M(70) = \frac{70-50}{30} = \frac{20}{30} = 0{,}67$$
- Near, 70 ≤ 80
  $$\mu N(70) = 0$$

Then SF = Far (0.33) + Medium(0.67)

iv. Rule Base

Table 6. Rule Base

	IF Left	AND Left-Front	AND Front	THEN ΔV
R1	Near	Medium	Far	Right
R2	Near	Medium	Medium	Right
R3	Near	Near	Medium	Strong Right
R4	Near	Near	Far	Right

v. Inference (Mamdani)
- Rule R1
  $\alpha_1 = \min(1.0, 0.967, 0.33) = 0.33$
  Output: Right (0.67)
- Rule R2
  $\alpha 2 = \min(1.0, 0.967, 0.67) = 0.67$
  Output: Right
- Rule R3
  $\alpha 3 = \min(1.0, 0.03, 0.67) = 0.03$
  Output: Strong Right
- Rule R4
  $\alpha 4 = \min(1.0, 0.03, 0.33) = 0.03$
  Output: Right

vi. Output Aggregation

Forward	: activation at 0.67
Right	: activation at max (0.33, 0.03) = 0.33
Strong right	: activation at 0.03

vii. Defuzzification (Centroid)

The center of Membership function:

Forward	= 0
Right	= 0.5
Strong Right	= 0.8

$$\Delta V = \frac{(0.67\ x\ 0) + (0.33\ x\ 0.5) + (0.03x0.8)}{0.67 + 0.33 + 0.03} =$$

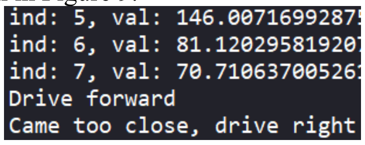
$$\Delta V = \frac{0.165 + 0.024}{1.03} = \boxed{+0.183}$$

viii. Interpretation

According to the steering control result, the robot slightly turns right.

ΔV = Vleft – Vright = +0.183

The fuzzy logic step and calculation reflect a similar outcome on the output console of the webots simulation in Figure 9.



```
ind: 5, val: 146.0071699287
ind: 6, val: 81.12029581920
ind: 7, val: 70.71063700526
Drive forward
Came too close, drive right
```

Fig. 9 Webot Output Console

The simulation results demonstrate that the robot is capable of maintaining a stable distance from the wall while navigating along straight paths. When the robot approaches a corner or detects an obstacle in the front direction, the fuzzy controller smoothly adjusts the steering control to avoid collision without abrupt changes in motion.

Compared to threshold-based control, the fuzzy logic approach provides smoother trajectory transitions, particularly when the robot encounters a wall corner or slight variations in distance measurements. The overlapping membership functions allow gradual steering corrections, reducing oscillations and improving overall navigation stability.

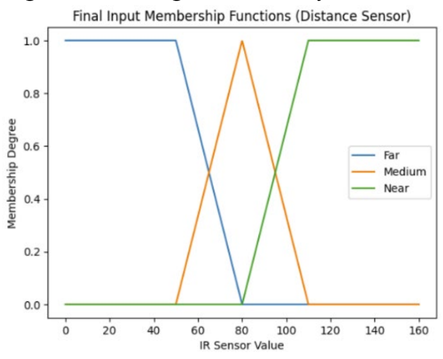


Figure 10. Input Membership Functions (Distance Sensor)

Since the e-puck infrared proximity sensor produces higher values as the robot approaches the wall, it can be seen in the Near and Far membership illustrated in Figure 10. The Near membership function is defined over the upper range of sensor readings, while the Far membership function corresponds to lower values.

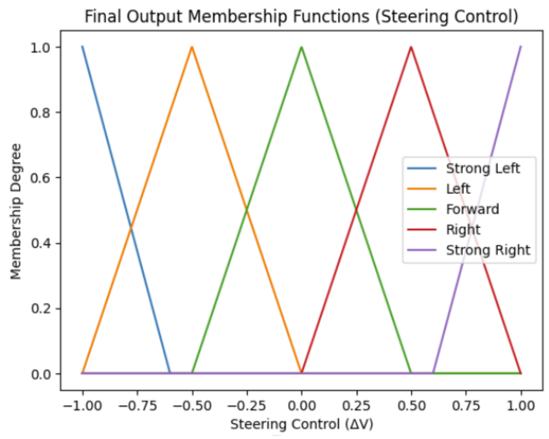


Figure. 11. The Output Membership Functions

Furthermore, Figure 11 displays the range of output membership. As an illustration, the robot can make a sharp turn left when the steering control indicates -1.00 with the membership degree at 1.0 reflected by the blue line. This is followed by other directions depicted by other color lines. Overall, the robot successfully recovers wall-following behavior when the wall is temporarily lost, indicating the robustness of the proposed fuzzy controller under varying environmental conditions.

5. CONCLUSION

The implementation of a wall-following robot using fuzzy logic control was successfully implemented in Webots simulation. According to the fuzzy logic-based control algorithm, it can be concluded that the wall-following robot simulation runs smoothly to traverse along the maze-like arena. Additionally, the chosen proximity sensors, PS5, PS6, and PS7, effectively carry out sensor reading gain. The closer the distance to the wall is, the higher the number will be, close to 160.

Furthermore, the fuzzy logic control functions optimally to categorize parameters into Near, Medium, and Far, breaking the sensor number down to 160, 110, 80, 50, and 0. Finally, the output membership functions successfully produced the direction of the robot's steering control movement, including the strong left, left, forward, right, and strong right.

ACKNOWLEDGEMENTS

Gratitude was conveyed to the Center for Research and Community Service (P3M) of the Bandung Manufacturing Polytechnic for encouraging this independent research to be carried out properly. Gratitude was also expressed to the Department of Manufacturing Automation and Mechatronics for facilitating the equipment and equipment used in this study.

REFERENCES

[1] T. Dash, "Automatic navigation of wall following mobile robot using Adaptive Resonance Theory of Type-1," *Biol. Inspired Cogn. Archit.*, vol. 12, pp. 1–8, 2015, doi: 10.1016/j.bica.2015.04.008.

[2] H. Suwoyo and F. H. Kristanto, "Performance of a Wall-Following Robot Controlled by a PID-BA using Bat Algorithm Approach," *Int. J. Eng. Contin.*, vol. 1, no. 1, pp. 56–71, 2022.

[3] G. Mikell, *Automation, Production Systems, and Computer-Integrated Manufacturing*. Lehigh University, 2015.

[4] A.Sh. Khusheef, G. Kothapalli, M. Tolouei_Rad, "Simulation of a mobile robot navigation system". *ICMS,* pp 1-2, 2011.

[5] M. T. Torriti, T. Arrendondo, P.C.Pizzaro, "Survey and Comparative Study of Free Simulation Software for Mobile Robots". *Robotica,*pp 1-3, 2014.

[6] P.C.Pizzaro, T. V.Arredondo, M. T.Torriti, "Introductory Survey to Open-Source Mobile Robot Simulation Software", IEEE LARS vol 1 pp 1-3, 2010.

[7] O. Michel, F. Rohrer, and N. Heiniger, "Cyberbotics ' Robot Curriculum," 2009.

[8] R. Braunstingl, J. Mujika, J. Pedro, J. M. Arizmendianieta, E.- Mondragon, and G. I. Espaiia, "A Wall Following Robot With A Fuzzy Logic Controller Optimized By A Genetic Algorithm 1 Description Of The Mobile Robot," 1995.

[9] S. Bansal and J. Kaur, "Wall Following Robot Navigation Using Machine Learning Techniques," *SSRN*, 2024.

[10] A. Lorenza, B. Hidayat, L. S. Alia, L. A. Nurachanta, and M. S. I. Syahputra, "Simulation Understanding Line Follower Robot C Program with Webots Simulasi Pemahaman Robot Pengikut Garis Program C dengan Webots," vol. 2, pp. 7–18, 2022.

[11] G. A. Vargas, O. G. Rubiano, R. A. Castillo, O. F. Avilés, and M. F. Mauledoux, "Simulation of e-puck path planning in webots," *Int. J. Appl. Eng. Res.*, vol. 11, no. 19, pp. 9772–9775, 2016.

[12] N. Zenita, K. B. K, A. Mujib, and N. Laily, "Journal of Application and Science Implementation of a 3-wheeled Wall Following Robot Navigation System using Coppelia," vol. 3, no. December 2021, pp. 63–72, 2022.

[13] M. R. Kenawas, P. Risma, T. Dewi, S. Muslimin, and Y. Oktarina, "JOURNAL OF APPLIED SMART ELECTRICAL NETWORK AND SYSTEMS (JASENS) Fuzzy Logic Controller sebagai Penentu Gerak Mobile Robot Pembasmi Hama," vol. 1, no. 1, pp. 19–24, 2020.

[14] C. F. Riman and P. E. Abi-char, "Fuzzy Logic Control for Mobile Robot Navigation in Automated Storage," vol. 12, no. 5, pp. 313–323, 2023, doi: 10.18178/ijmerr.12.5.313-323.

[15] M. M. Aditya, A. B. Insani, J. Al Kausar, and A. Silvia, "Sistem Kendali Fuzzy pada Mobile Robot ANNUAL RESEARCH SEMINAR 2016," vol. 2, no. 1, pp. 231–234, 2016.