Indonesian Journal of Science & Technology

Journal homepage: http://ejournal.upi.edu/index.php/ijost/

The Mobile Robot Control in Obstacle Avoidance Using Fuzzy Logic Controller

M. Khairudin[1], R. Refalda[1], S. Yatmono[1], H. S. Pramono[1], A. K. Triatmaja[1], A Shah[2]

[1]Dept. of Electrical Engineering, Universitas Negeri Yogyakarta, Yogyakarta, Indonesia
[2]Faculty of Technical and Vocational, Universiti Pendidikan Sultan Idris, Perak, Malaysia

Correspondence: E-mail: : moh_khairudin@uny.ac.id

ABSTRACTS

A very challenging problem in mobile robot systems is mostly in obstacle avoidance strategies. This study aims to describe how the obstacle avoidance system on mobile robots works. This system is designed automatically using fuzzy logic control (FLC) developed using Matlab to help the mobile robots to avoid a head-on collision. The FLC designs were simulated on the mobile robot system. The simulation was conducted by comparing FLC performance to the proportional integral derivative (PID) controller. The simulation results indicate that FLC works better with lower settling time performance. To validate the results, FLC was used in a mobile robot system. It shows that FLC can control the velocity by braking or accelerating according to the sensor input installed in front of the mobile robot. The FLC control system functions as ultrasonic sensor input or a distance sensor. The input voltage was simulated with the potentiometer, whereas the output was shown by the velocity of DC motor. This study employed the simulation work in Simulink and Matlab, while the experimental work used laboratory scale of mobile robots. The results show that the velocity control of DC motors with FLC produces accurate data, so the robot could avoid the existing obstacles. The findings indicate that the simulation and the experimental work of FLC for mobile robot in obstacle avoidance are very close.

ARTICLE INFO

Article History:
Received 14 Nov 2019
Revised 04 Des 2019
Accepted 20 Jan 2020
Available online 26 May 2020

Keywords:
Experiment;
fuzzy logic controller;
mobile robot;
obstacle

1. INTRODUCTION

Though the number of head-on collisions is only 2.2% among transportation accidents, this type of crash contributes to 10% of death rates from road traffic incidents. Around 75% of traffic accidents are caused by human (driver) error (Basjaruddin *et al.*, 2016). This kind of accident can be influenced by inaccurate environmental recognition, bad decision-making, low performance, non-performance mistakes, and others with percentages of 40.6; 34.1; 10.3; 7.1; and 7.9, respectively.

The number of accidents can actually be reduced by installing an obstacle avoidance system, a device that helps drivers to lower the risk of collisions. This system helps minimize human errors due to sleepy condition or concentration problems behind the wheel (Faisal *et al.*, 2013). The errors caused by unfocused drivers happen frequently. In the case of traffic jams, drivers may unconsciously step the velocity or gas pedal resulting in a crash. It urges the need for a real-time object detector (Khairudin *et al.*, 2019; Mohamed *et al.*, 2016). Thus, several intelligent controllers make use of fuzzy logic as the system (Amelia *et al.*, 2019; Mojaveri & Moghimi, 2017).

The obstacle avoidance system functioning as a distance sensor is installed in the front part of a vehicle to identify input. This sensor detects the distance of other vehicles in the front. If the sensor detects that the front vehicle is quite far, the car moves fast when the driver steps on the gas pedal. On the other hand, if the sensor notices that the front vehicle is in a close distance, even after the driver steps on the gas pedal, the vehicle will not go at high velocity. It will stop or slow down.

The obstacle avoidance system is designed with a laboratory scale to improve the ability to avoid obstacles (Bhagat *et al.*, 2016). This ability is expected to be like that of humans so that studies on humanoid robots have been carried out in planning its movements with fuzzy Markov (Fakoor *et al.*, 2016). Several methods have been used to enhance mobile robot ability to avoid obstacles, (Oborski & Fedorczyk, 2015) and the most challenging issue is how mobile robots can perform obstacle avoidance with minimum cost (Ellili *et al.*, 2016).

The classic methods to control mobile robots in avoiding obstacles mainly employ road maps, potential field methods, decomposition of cells and several other methods (Szulczyński *et al.*, 2011). The intelligent control methods have also been applied using such as fuzzy logic controllers (Hong *et al.*, 2016; Bakdi *et al.*, 2017; Zuhrie *et al.*, 2017), neural networks (Budiharto, 2015), genetic algorithms (Mac *et al.*, 2017), and particle swarm optimization (Chołodowicz & Figurowski, 2017). The robot ability to obtain an accurate view of the existing obstacles to avoid has become the main issue from time to time (Terven *et al.*, 2016).

Several studies have been done to provide FLC design for mobile robots, but there are only a few studies on the comparison between the results of FLC implementation in mobile robots based on simulation and experimental works. The previous studies tend to explain FLC for a mobile robot based only on simulation works. There are also studies presenting FLC for mobile robot control with experimental works. However, only a small number of studies compare FLC to other methods for mobile robot performances in obstacle avoidance. Therefore, there is an urge to present the results of comparing the simulation and experimental works for mobile robot using FLC to avoid obstacles.

This study compares FLC to PID controller performances in a simulation work. It also presents the results of comparison which is rarely made by researchers. The findings show that FLC performance is better than PID controller. Mobile robots performance in avoiding obstacles is validated by comparing the results of simulation and experimental works. Then, it is found that the simulation work provides very relevant and more consistent results than the experimental work.

2. METHODS

2.1. Mobile Robot

Figure 1 presents the experimental setup of the mobile robot. The rig consisted of some parts, i.e. two DC motors as mobile robot actuators, ultrasonic sensors and input set points with potentiometers, and a processor. The ultrasonic sensor was installed in the front to detect the obstacles when the mobile robot was moving.

The working system was the installed ultrasonic sensor placed in the front of the mobile robot to detect obstacles, while the sensor data were processed by the microcontroller of ATmega328 Arduino Uno. Table 1 presents the details of the mobile robot specifications. The mobile robot moved with two wheels connected to a DC motor as an actuator. These two wheels were attached to the right and left on the backside and the lower front-mounted freewheel. Mobile robot simulations used Simulink and Matlab. Moreover, to obtain real-time data acquisition in implementing FLC control system on the mobile robot, the microcontroller was employed as the data processor.

In this study, FLC was designed, simulated and implemented to control a mobile robot for avoiding obstacles.

Table 1. Specification parameter of a mobile robot

Component	Spesification
Actuator	DC Motor 12V
Power supply	Powerbank 14000 mAH 5VDC Lipo Batery 3S 2200 mAH 12 VDC
Sensor	Sensor of distance measuring : Ultrasonic Sensor HC SR04
Step-down	Step-down Variable with 7 Segment
Processing Data	Microcontroller ATmega328, arduino uno
Bodyframe	3mm Acrylic
Motor driver	Type L298N
Body size	
Length	0.26 m
Height	0.15 m
Width	0.20 m

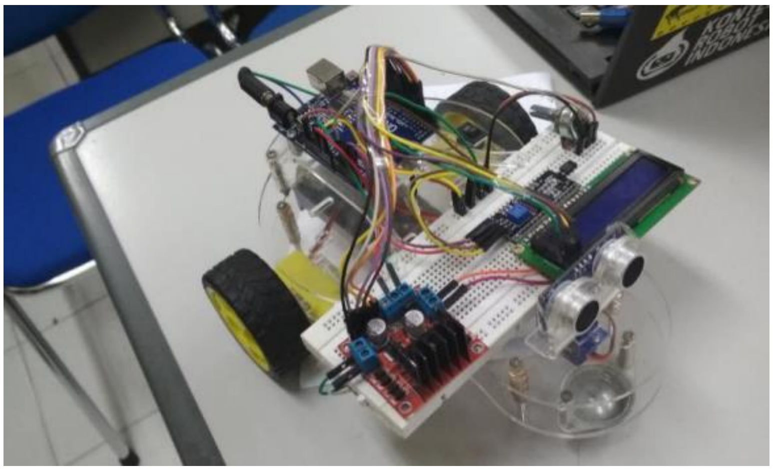


Figure 1. Experimental setup of mobile robot

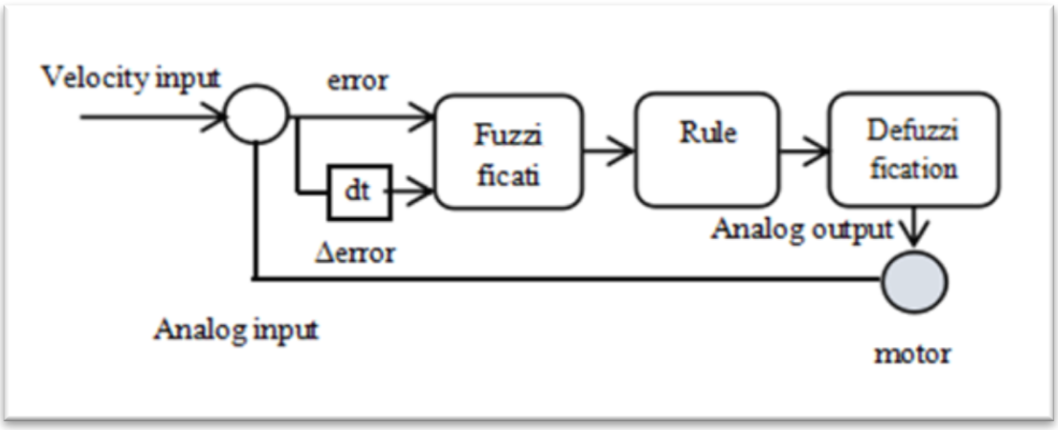


Figure 2. FLC design for mobile robot

2.2. Fuzzy Logic Controller Design

In designing the FLC control system, there several steps performed, such as fuzzification, membership function grouping, rule evaluation, and the defuzzification process. The FLC design procedure in **Figure 2** is further explained in the following steps.

a) Initialization

In this system, there are 3 fuzzy variables, namely Sensor1 in the form of the ultrasonic sensor, Analog Input, and DC Motor. The Fuzzy set refers to a group representing a certain condition of fuzzy variables. The fuzzy variables involve:

(1) Variable of Sensor1 that is divided into 5 fuzzy sets, namely Near, Quite Near, Medium, Quite Far. and Far.
(2) The variable of analog input which is classified into 5 fuzzy sets, namely No Input, Less Input, Medium Input, More Input, and Full Input.
(3) The DC Motor with 5 fuzzy set categories, namely Stationary, Slow, Medium, Quite Fast, and Fast.

The universe of discourse refers to the overall value that is refered to as a fuzzy variable.

Universe of discourse for
variable of Sensor1 = [0 300]
Universe of discourse for variable
of InputPotensio = [0 300]
Universe of discourse for variable
of DC Motor = [0 1500]

The fuzzy set domain refers to the whole value allowed in the universe of discourse that can be operated in this set.

The fuzzy set domain for variable of Sensor1 can be obtained as follows:

Near	= [-75:0:75]
QNear	= [0:75:150]
Medium	= [75:150:225]
QFar	= [150:225:300]
Far	= [225:300:375]

and domain fuzzy sets variable of InputPotensio can be calculted as follows:

NInput	= [-75:0:75]
LInput	= [0:75:150]
Medium Input	= [75:150:225]
MInput	= [150:225:300]
FInput	= [225:300:375]

Domain fuzzy sets variable of DCMotor can be determined as follows:

Stationary	= [-375.1 0 375.1]
Slow	= [0 375.1 750]
Medium	= [375.1 750 1127]
QFast	= [750 1127 1500]
Fast	= [1127 1500 1876]

Membership function (MF) is a curve showing the map of data input points into membership values ranging from 0 to 1. In detail, MF design for the input systems in Sensor1, Analog Input and DC motor output are shown in **Figures 3, 4 and 5**.

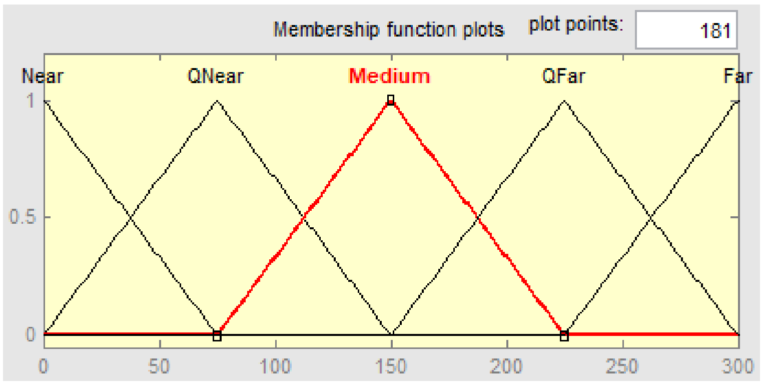


Figure 3. MF Design for input of Sensor1

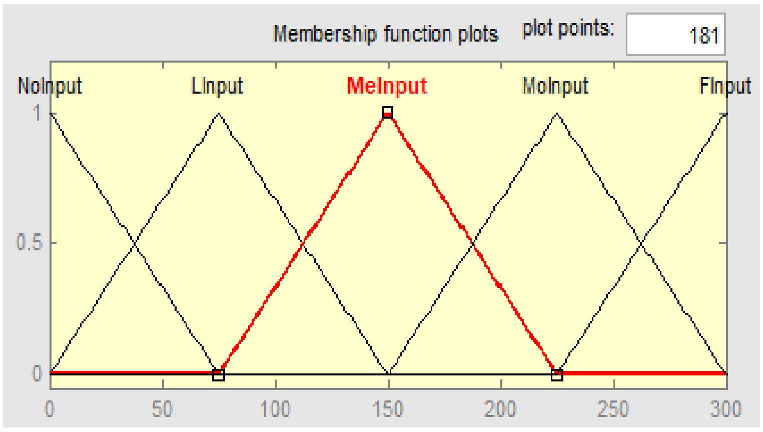


Figure 4. MF Design for input of Input Potensio

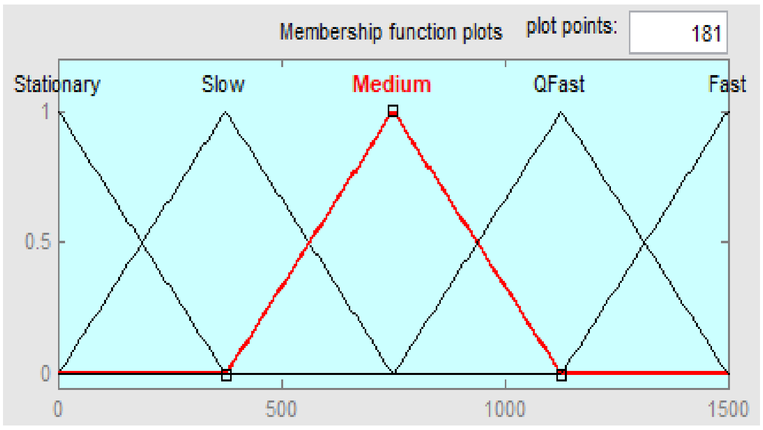


Figure 5. MF Design for input of DC motor output

b) Fuzzyfication

The next step is determining fuzzification on Sensor1 input. For Sensor1 input, the variable (x) is defined as an analog input to digital converter (ADC) value, for example the value of 50. The determination of fuzzification with Mamdani model on the Near MF to get the error value (μ) can be determined using the Equation (1).

$$\mu\ MF\ Near = \begin{cases} (b-x)/(b-a); & a \le x \le b \\ 0; & x \ge b \end{cases} \qquad (1)$$

where *x*, *b*, and *a* are the values of the Sensor1 input variable of ADC, upper limit and lower limit respectively. Thus, in case of near MF given *x*=50, *a*=0, and *b*=75, the resukt is $\mu\ Near = 0.33$.

Meanwhile, fuzzification using Mamdani model in MF Qnear, Medium, and QFar to get the error value (μ) is determined using the Equation (2).

$$\mu\ MF\ QNear/Medium/QFar = \begin{cases} 0; & x \le a\ atau\ x \ge c \\ (x-a)/(b-a); & a \le x \le b \\ (c-x)/(c-b); & b \le x \le c \end{cases} \quad (2)$$

Therefore, for medium MF given *x*=160, *a*=75, *b*=150 and *c*=225, the result is $\mu\ Medium = 0.86$, whereas for MF QNear given *x*=50, *a*=0, *b*=75, and *c*=150, the result is $\mu\ QNear = 0.66$. While for MF QFar given *x*=275, *a*=150, *b*=225 and *c*=300, the result is $\mu\ QFar = 0.33$.

The determination of fuzzification with Mamdani model on Far MF to get the error value (μ) used the Equation (3).

$$\mu\ MF\ Far = \begin{cases} 0; & x \le a \\ (x-a)/(b-a); & a \le x \le b \\ 1; & x \ge b \end{cases} \quad (3)$$

It means Far MF given with *x*=250, *a*=225, and *b*=300, obtained $\mu\ Far = 0.33$.

For determining fuzzification using Mamdani model in Analog input with the same technique to get the value of delta_error (μ), it is obtained with the following value.

1) Analog input of InputPotensio for MF of NoInput.

In case of MF NoInput given *x*=50, *a*=0, and *b*=75, the result is $\Delta\mu\ NInput = 0.33$.

2) Analog input of InputPotensio of MF Less Input, Medium Input, and More Input. In case of MF MeInput given *x* = 160, *a* = 75, *b* = 150, and *c* = 225, the result is $\mu\ MeInput = 0.86$. Whereas for MF LInput given x = 50, *a* = 0 , *b* = 75, and *c* = 150, we obtained $\mu\ LInput = 0.66$. In case of MF MoreInput given *x*=275, *a*=150 , *b*=225, and *c*=300, the result is $\mu\ MoInput = 0.33$.
3) Analog input of InputPotensio of MF Full Input. In case of MF Full Input, given *x*=250, *a*=225, and *b*=300, the result is $\Delta\mu\ FullInput = 0.33$.

The evaluation rule is determined using the logic operation of AND. Meanwhile, $\alpha - predicate$, resulted from logical operations of AND is obtained by taking the smallest membership value between elements in the related set. It can be seen at Equation (4).

$$\mu A \cap B = \min(\mu A(x), \mu B(y)) \quad (4)$$

where the value of x and y are the input variables.

In this case for the rule with Sensor1 condition and Analog Input are 200 and 100, respectively. The rule can be constructed as follows:

[R1] IF SENSOR1 Near AND InputPotensio NoInput THEN DCMOTOR Stationary
It can be found with the Equation (5).

$$\alpha - predicate_1 = \mu Sensor1Near \cap InputPotensioNoInput$$

$$\begin{aligned} \alpha - predicate_1 &= \mu Sensor1Near \cap InputPotensioNoInput \\ &= \min(\mu Sensor1Near(200), \mu InputPotensioNoInput(100)) \\ &= \min(0;0) \\ &= 0 \end{aligned} \quad (5)$$

$\alpha pred_1 = \alpha - predicate_1$, *a* and *b* are lower limit and upper limit for maximum output, respectively. In order to obtain the defuzzification of maximum luminance for rule1, it can determine with Equation (6).

$$z_1 = b - (\alpha pred_1 * (b-a)) \quad (6)$$

Thus when *a*=0 dan *b*= 375, the result is:

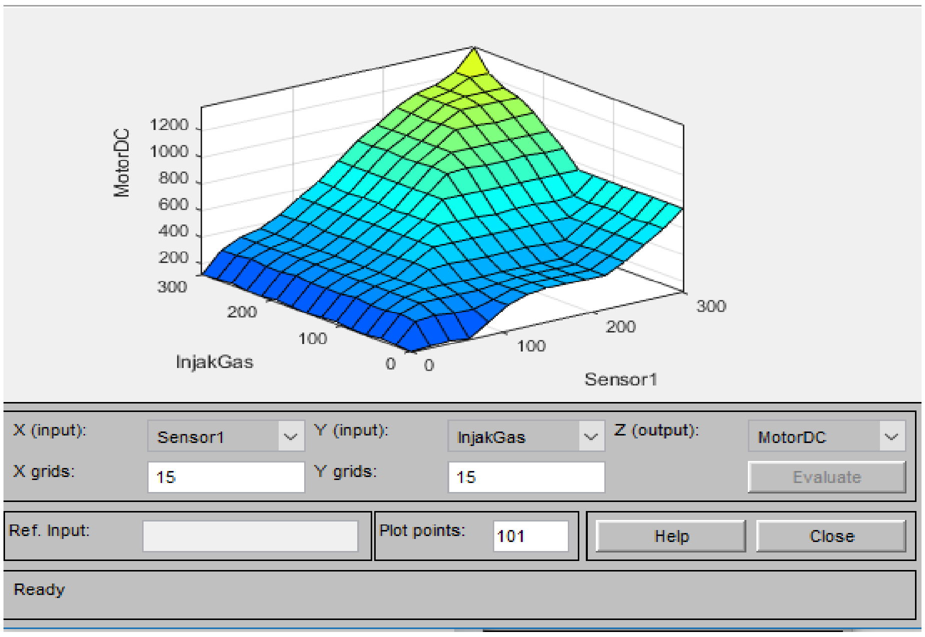
$$\begin{aligned} z_1 &= b - (\alpha pred_1 * (b-a)) \\ &= 0 - (0 * (375.1 - 0)) \\ &= 0 - 0 \\ &= 0 \end{aligned}$$

Through the same technique, other rules are obtained:

[R2] IF SENSOR1 Near AND InputPotensio LInput THEN DCMOTOR Stationary
[R3] IF SENSOR1 Near AND InputPotensio MeInput THEN DCMOTOR Stationary
[R4] IF SENSOR1 Near AND InputPotensio MoInput THEN DCMOTOR Stationary
[R5] IF SENSOR1 Near AND InputPotensio FInput THEN DCMOTOR Stationary
[R6] IF SENSOR1 QNear AND InputPotensio NoInput THEN DCMOTOR Stationary
[R7] IF SENSOR1 QNear AND InputPotensio LInput THEN DCMOTOR Slow
[R8] IF SENSOR1 QNear AND InputPotensio MeInput THEN DCMOTOR Slow
[R9] IF SENSOR1 QNear AND InputPotensio MoInput THEN DCMOTOR Slow
[R10] IF SENSOR1 QNear AND InputPotensio FInput THEN DCMOTOR Slow
[R11] IF SENSOR1 Medium AND InputPotensio NoInput THEN DCMOTOR Slow
[R12] IF SENSOR1 Medium AND InputPotensio LInput THEN DCMOTOR Slow
[R13] IF SENSOR1 Medium AND InputPotensio MeInput THEN DCMOTOR Medium
[R14] IF SENSOR1 Medium AND InputPotensio MoInput THEN DCMOTOR Medium
[R15] IF SENSOR1 Medium AND InputPotensio FInput THEN DCMOTOR Medium
[R16] IF SENSOR1 QFar AND InputPotensio NoInput THEN DCMOTOR Slow
[R17] IF SENSOR1 QFar AND InputPotensio LInput THEN DCMOTOR Slow
[R18] IF SENSOR1 QFar AND InputPotensio MeInput THEN DCMOTOR Medium
[R19] IF SENSOR1 QFar AND InputPotensio MoInput THEN DCMOTOR QFast
[R20] IF SENSOR1 QFar AND InputPotensio FInput THEN DCMOTOR QFast
[R21] IF SENSOR1 Far AND InputPotensio NoInput THEN DCMOTOR Medium
[R22] IF SENSOR1 Far AND InputPotensio LInput THEN DCMOTOR Medium
[R23] IF SENSOR1 Far AND InputPotensio MeInput THEN DCMOTOR Medium
[R24] IF SENSOR1 Far AND InputPotensio MoInput THEN DCMOTOR QFast
[R25] IF SENSOR1 Far AND InputPotensio FInput THEN DCMOTOR Fast

The results of rule evaluation design of surface performance for rule evaluation can be be seen in **Figure 6**. After the defuzzification process is finished, the output value (Z) can be determined by employing Equation (7).

$$Z = \frac{\alpha pred_1 * z_1 + \alpha pred_2 * z_2 + \alpha pred_3 * z_3 + \dots\dots\dots + z_{25} + \alpha pred_{25}}{\alpha pred_1 + \alpha pred_2 + \alpha pred_3 + \dots\dots\dots + \alpha pred_{25}} \tag{7}$$



X (input): Sensor1
Y (input): InjakGas
Z (output): MotorDC
X grids: 15
Y grids: 15
Evaluate
Ref. Input:
Plot points: 101
Help
Close
Ready

Figure 6. Surface performance for RULE evaluation

By subtituting the value of each variable, the result can be obtained as

$$Z = \frac{\begin{array}{c}0+0+0+0+0+0+0+0+0+0+0+0+288.62+\\164.6+0+0+0+331.7+330.85+0+0+0+0+0+0+0\end{array}}{\begin{array}{c}0+0+0+0+0+0+0+0+0+0+0+0+0.52+0.33+\\0+0+0+0.66+0.33+0+0+0+0+0+0+0\end{array}}$$

$$Z = \frac{1115.77}{1.84}$$

$$Z = 606.4$$

In order to obtain the number of output values (Z) of 606.4, the Equation (7) can be applied to control the velocity of output DC Motor.

3. RESULTS AND DISCUSSION

In this study, to assess the prepared FLC system, a simulation work is done. The simulation work uses the Simulink and Matlab programs. The process of measuring system through simulation work is shown in **Figure 7**. In the block diagram, there is an input step to adjust the source of desired volt. The automatic driver control system is simulated with a DC motor system as a plant that is controlled through FLC. In this simulation work, FLC performance is compared to the PID controller. To make sure that the design of FLC can be implemented in hardware, the next step is to assess the FLC with simulation work using Simulink and Matlab.

The Fuzzy Inference System process in this study employs the Mamdani method. Every consequence of IF-Then rules must be represented by a fuzzy set with a monotonous membership function, and as a result, the output of the inference from each rule is presented explicitly (crisp) based on the α-predicate. The final result is presented in a weighted average. Meanwhile, in the FIS design, there is a display showing the output results from rule that has been prepared with the surface display as shown in **Figure 6**.

In **Figure 6** as a display surface, it can be seen that the results of RULE are linear starting from 0 to 1500. However, in the picture, it looks smaller because the RULE does not adjust to the linear stability.

3.1. Simulation works

In the block diagram system using an FLC-based control system, FLC control system is connected to the plant in the form of DC motor as a simulation of the anti-collision control system. The performance of FLC system in this simulation work is compared to PID control system. PID control system is determined by Ziegler Nichols method, in which the values of *Kp*, *Ki* and *Kd* were 15, 30 and 0, respectively.

The large PID gain control system produces a big change in the output of a particular error value. However, if the gain is too large, the system needs a long time to reach a steady-state condition. Conversely, if the gain is small, the output response may also be small, making the controller to be less responsive or sensitive. It makes the controller response increasingly slower if there is any interference. The Integral is set to 30, which is directly proportional to the magnitude and duration of the error. Integral in PID controller is the sum of errors each time and accumulated with the offset that has been previously corrected. The integral term accelerates the transfer of processes to the set point and eliminated steady-state errors that occur on proportional controllers. Since the integral responds to accumulated errors, it can cause overshoot. Finally, the derivative value is set to 0 to determine the slope of the error at each time and multiplied by the change at each time with the derivative gain.

The simulation work is done by integrating the Simulink block diagram in the FLC system that has been designed through the fuzzy inference system in the form of file .fis. The results of Simulink simulation between FLC and PID controller performance are shown in **Figure 7**.

In **Figure 8**, it can be seen that there is a comparison between the control system performance on mobile robots using FLC and PID controller. The mobile robot system is given the input step with a value of 1500 rpm. The input step is set with the time of 0, i.e. the line started directly from the value of 0. Moreover, the initial value is set to 0 to start the step directly from the value 0, and the Final Value is set in 1500 rpm that the result of the direct output is 1500 rpm.

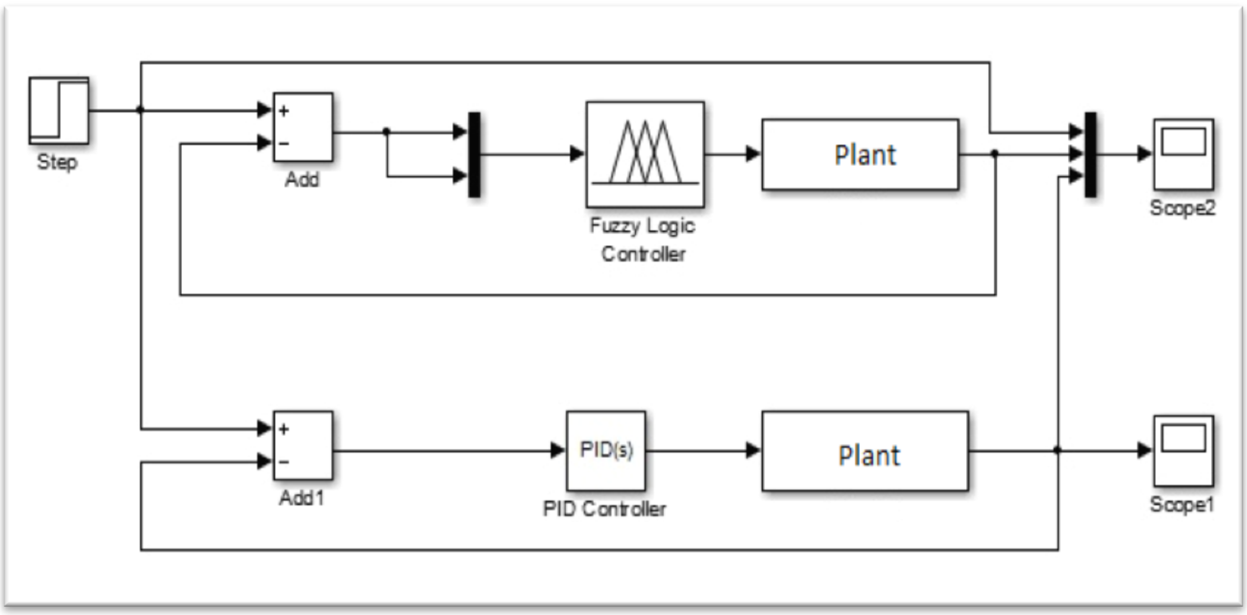


Figure 7. Schematic simulation work

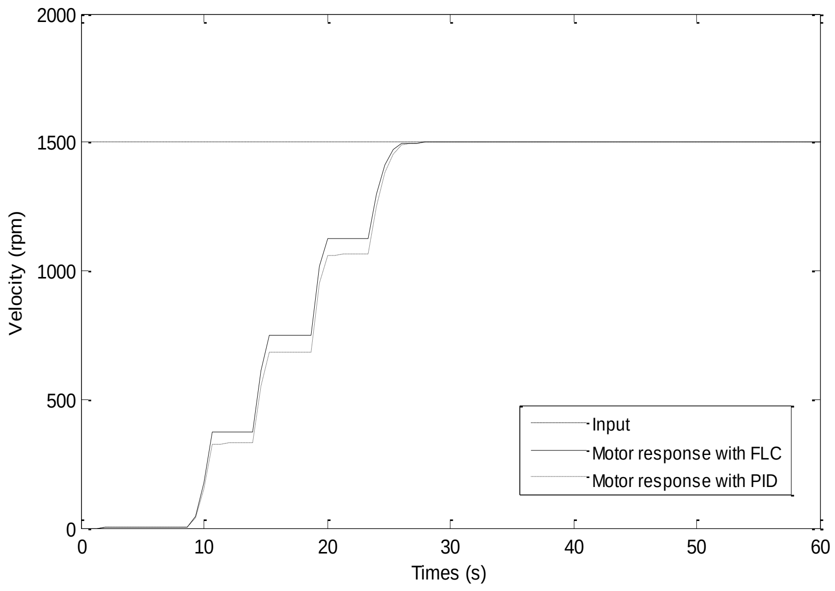


Figure 8. Performance comparison for FLC and PID

The results show that the mobile robot follows the given input values. In system with FLC, the accuracy with settling is 24.50 s and overshoot is 0%. On the other hand, the mobile robot system using the PID control system indicates that the performance with settling is 24.80 s and overshoot is 0%. These results indicate that the system with FLC provides better performance than the system with PID control. Based on these results, FLC control system can be officially used to be implemented in real-time systems and hardware through experimental work.

The simulation works in this study have been in line with previous study conducted by other researchers (Ellili *et al.*, 2016) on how FLC for mobile robot can produce output responses without overshoot. However, the previous study showed higher settling time compared to this study. It means that this study presents FLC design with better performance compared to the previous study in terms of faster settling time and no overshoot.

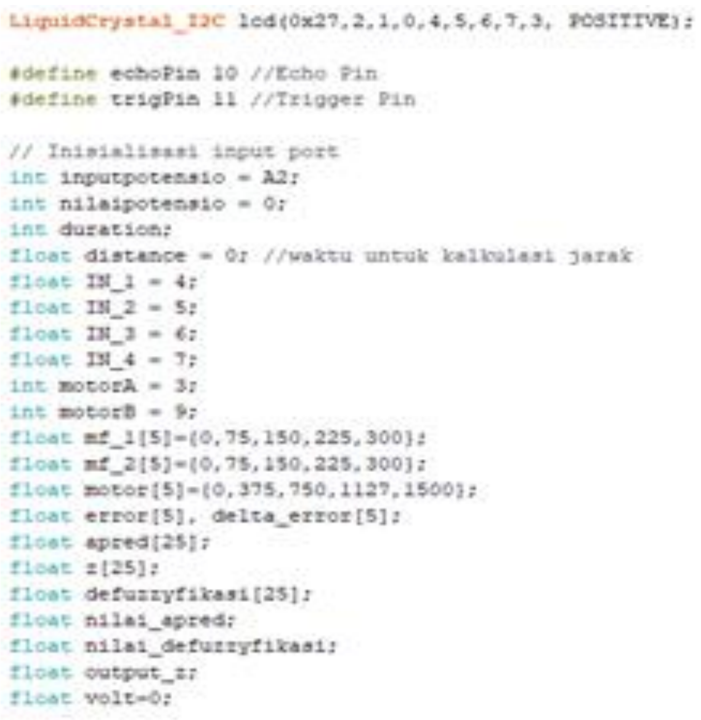
3.2. Experimental works

Implementation of the FLC control system on collision avoidance systems can be done by several mini computers as data processing. In this study, the data processing for FLC system is carried out using the Arduino microcontroller. The laboratory-scale system for the mobile robot control system to avoid the obstacle contains ultrasonic distance sensor input to measure the distance between the mobile robot and the obstacle, the input voltage simulated with a potentiometer, and the system output, i.e. the vehicle velocity scaled by the laboratory with DC motor velocity.

The preparation of the FLC application program using the Arduino microcontroller is certainly the same as the FLC process in general. The FLC process includes fuzzification, membership functions, rule evaluation, and defuzzification. The preparation of FLC application program using the Arduino microcontroller begins with the initiation of parameters and variables as shown in **Figure 9**.

The arrangement of the collision avoidance system program of FLC uses Arduino UNO application with ultrasonic sensor input and potentiometer as the substitution of gas stepping and the output system, i.e. the vehicle velocity scaled up with DC motor velocity.

Figure 9 presents the initialization of the program including defining variables in the form of float data types, integers and using arrays. Meanwhile, the float data type of **Figure 9** involves the variables of *mf_1*, *mf_2*, distance, error, delta_error, apread, defuzzification, and output.



```
LiquidCrystal_I2C lcd(0x27,2,1,0,4,5,6,7,3, POSITIVE);

#define echoPin 10 //Echo Pin
#define trigPin 11 //Trigger Pin

// Inisialisasi input port
int inputpotensio = A2;
int nilaipotensio = 0;
int duration;
float distance = 0; //waktu untuk kalkulasi jarak
float IN_1 = 4;
float IN_2 = 5;
float IN_3 = 6;
float IN_4 = 7;
int motorA = 3;
int motorB = 9;
float mf_1[5]={0,75,150,225,300};
float mf_2[5]={0,75,150,225,300};
float motor[5]={0,375,750,1127,1500};
float error[5], delta_error[5];
float apred[25];
float z[25];
float defuzzyfikasi[25];
float nilai_apred;
float nilai_defuzzyfikasi;
float output_z;
float volt=0;
```

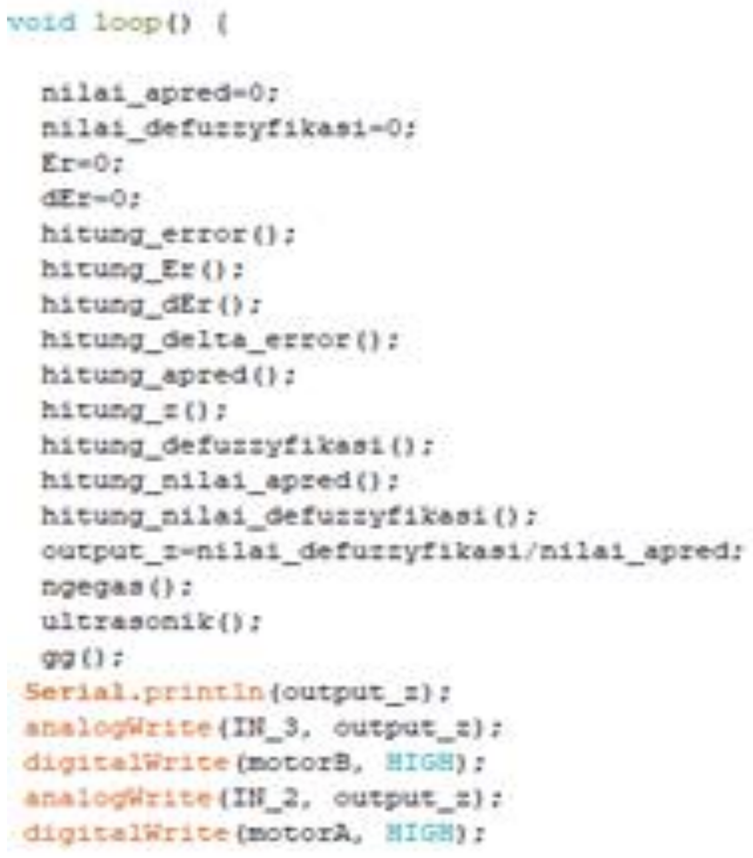
Figure 9. Coding for Initiation program

Integer data types include the value of the analog input, motorA, motor, Er and Der. The data variables that are used in array form included *mf_1*, *mf_2*, and *motor*. Then, the fuzzification process is carried out through the source code in details as shown in **Figure 10.**

Figure 10 explains the program for the fuzzification process with input errors. The fuzzification process program with delta error input uses the same program script as in the error input.

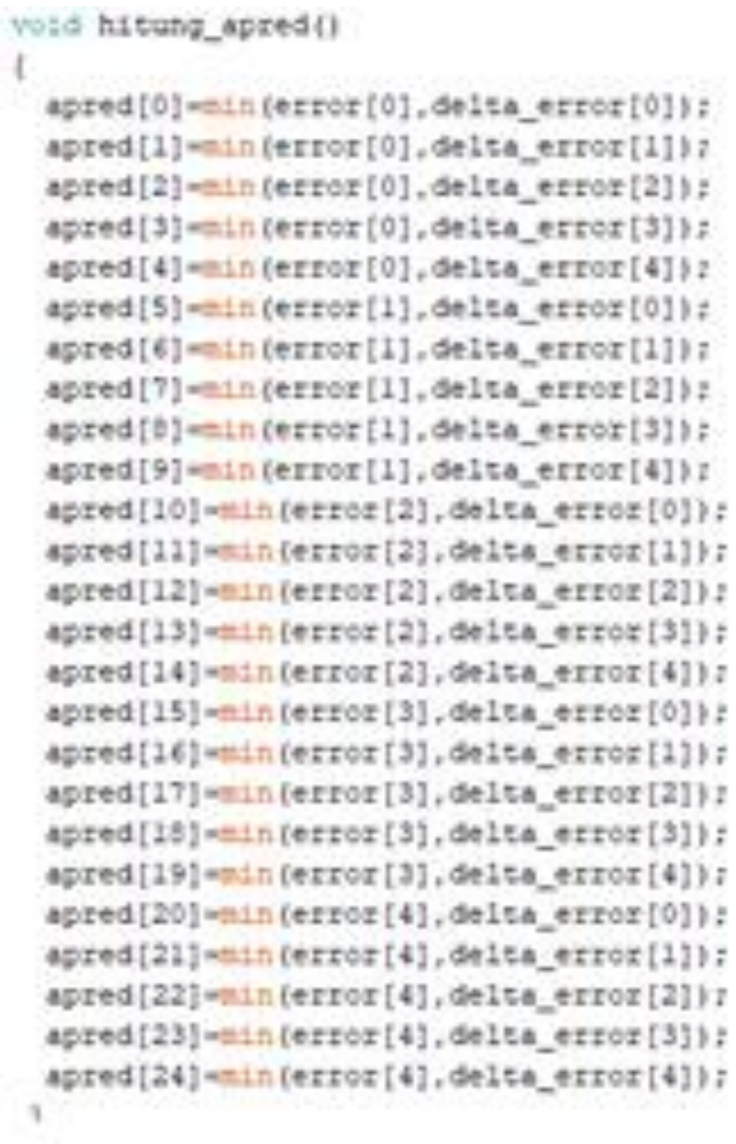
The next step is the calculation process of α-predicate values. **Figure 11** explains the program for determining the α-predicate value by using AND operator so that the smallest value from the input error and input delta error comparison can be chosen.

The steps for preparing FLC on the Arduino system also end with the defuzzification process. It is followed by the programming process to determine the defuzzification value (z) based on the desired RULE. The next process is determining the output of FLC system to get the value of Pulse Width Modulation (PWM) as the final process. **Figure 12** explains the preparation of the program to find FLC Output value, namely the PWM value (Z). To make editing process easier, the error value, delta error and PWM need to be monitored through LCD display. Testing with proteus software is done before hardware analysis. Figure 13 presents the results of the assess program using proteus software.



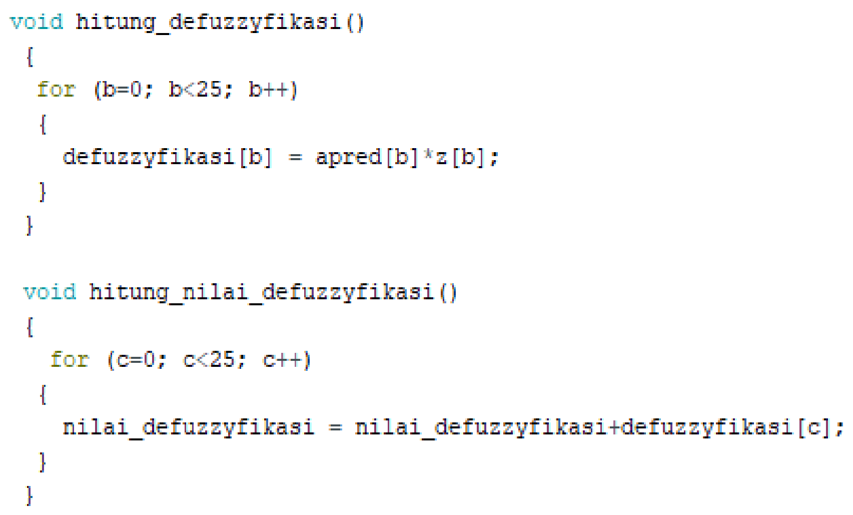
```
void loop() {

   nilai_apred=0;
   nilai_defuzzyfikasi=0;
   Er=0;
   dEr=0;
   hitung_error();
   hitung_Er();
   hitung_dEr();
   hitung_delta_error();
   hitung_apred();
   hitung_z();
   hitung_defuzzyfikasi();
   hitung_nilai_apred();
   hitung_nilai_defuzzyfikasi();
   output_z=nilai_defuzzyfikasi/nilai_apred;
   ngegas();
   ultrasonik();
   gg();
  Serial.println(output_z);
  analogWrite(IN_3, output_z);
  digitalWrite(motorB, HIGH);
  analogWrite(IN_2, output_z);
  digitalWrite(motorA, HIGH);
```

Figure 10. Coding for Fuzzification program



```
void hitung_apred()
{
  apred[0]=min(error[0],delta_error[0]);
  apred[1]=min(error[0],delta_error[1]);
  apred[2]=min(error[0],delta_error[2]);
  apred[3]=min(error[0],delta_error[3]);
  apred[4]=min(error[0],delta_error[4]);
  apred[5]=min(error[1],delta_error[0]);
  apred[6]=min(error[1],delta_error[1]);
  apred[7]=min(error[1],delta_error[2]);
  apred[8]=min(error[1],delta_error[3]);
  apred[9]=min(error[1],delta_error[4]);
  apred[10]=min(error[2],delta_error[0]);
  apred[11]=min(error[2],delta_error[1]);
  apred[12]=min(error[2],delta_error[2]);
  apred[13]=min(error[2],delta_error[3]);
  apred[14]=min(error[2],delta_error[4]);
  apred[15]=min(error[3],delta_error[0]);
  apred[16]=min(error[3],delta_error[1]);
  apred[17]=min(error[3],delta_error[2]);
  apred[18]=min(error[3],delta_error[3]);
  apred[19]=min(error[3],delta_error[4]);
  apred[20]=min(error[4],delta_error[0]);
  apred[21]=min(error[4],delta_error[1]);
  apred[22]=min(error[4],delta_error[2]);
  apred[23]=min(error[4],delta_error[3]);
  apred[24]=min(error[4],delta_error[4]);
 }
```

Figure 11. Coding to obtain α-predicate



```
void hitung_defuzzyfikasi()
 {
  for (b=0; b<25; b++)
  {
    defuzzyfikasi[b] = apred[b]*z[b];
  }
 }

 void hitung_nilai_defuzzyfikasi()
 {
   for (c=0; c<25; c++)
  {
    nilai_defuzzyfikasi = nilai_defuzzyfikasi+defuzzyfikasi[c];
  }
 }
```

Figure 12. Coding to obtain FLC output

Figure 13 presents the microcontroller of Arduino Uno circuit assembled to 1 potentiometer input, 2 ultrasonic sensor inputs, and DC motor driver, which had been linked to DCmotor A and B outputs.

After assessing the system using Simulink and Matlab as well as proteus for the FLC system of the mobile robot, experimental work is conducted. It consists of a plant in the form of a mobile robot involving two DC motors and motor drivers. The mobile robot system is equipped with an ultrasonic sensor input as a proximity sensor and input voltage simulated with a potentiometer.

The mobile robot system with FLC to avoid obstacles contains a proximity sensor installed on the front part of the robot and the input voltage regulator system with a potentiometer. The proximity sensor can detect the obstacle distance in front whether near or far. If the sensor detects the front obstacles in a far range, the mobile robot goes fast. On the contrary, when the sensor detects that the obstacle is near, the mobile robot does not go fast and keeps silent or moves with low velocity despite having full voltage. The mobile robot system in this experimental work is shown in **Figure 1**.

The test results on experimental work with FLC performance are displayed in the form of errors, delta errors, PWM values, and DC motor velocity. **Figure 14** shows the error and delta error as FLC input. The steady-state values for errors and delta errors of 298 rpm are fulfilled at 24 seconds. After 31 seconds, the ultrasonic sensor detects an obstacle. Thus, it appears in the delta error after braking. It makes the graph decrease around 36 seconds. In 37 seconds, the ultrasonic sensor does not detect the obstacle and the braking is over that caused the delta error signal began to rise. This operation process operates to 44 seconds where the mobile robot operation process is terminated at the stop point, so the delta error value is close to zero.

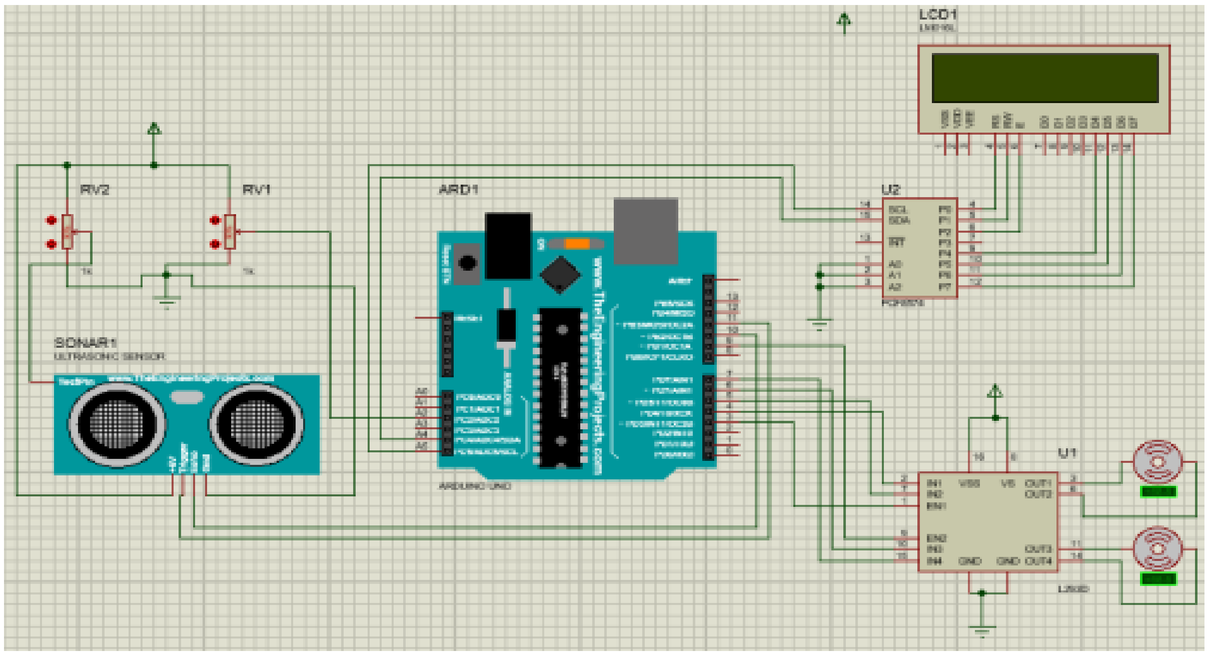


Figure 13. Schemcatic of Simulation work in proteus

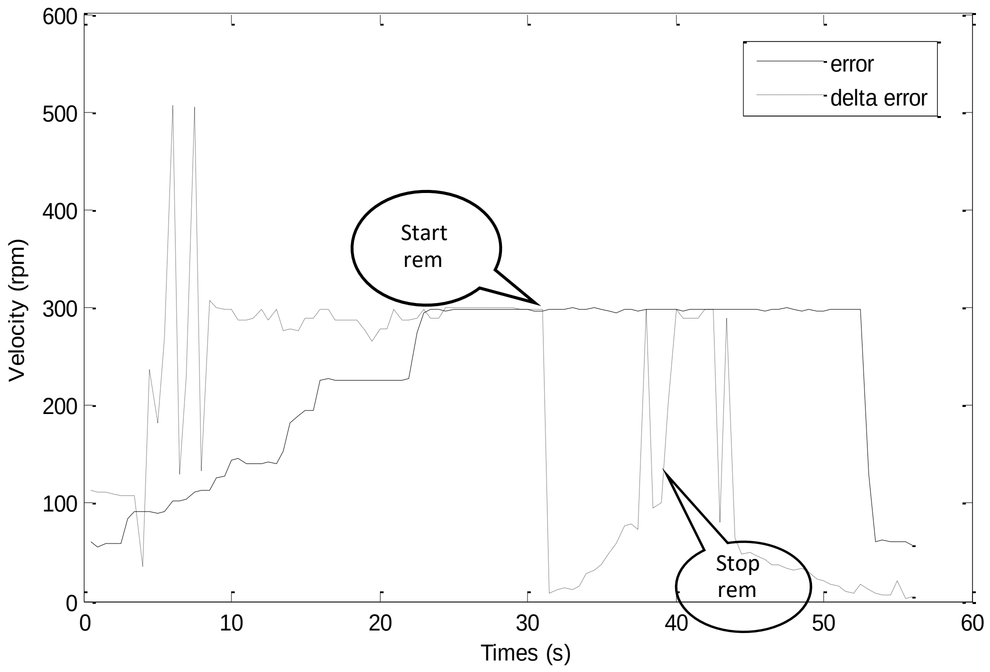


Figure 14. Error and delta_error for FLC

Figure 15 shows PWM data that is almost similar to PWM performance generated by FLC to be inputted on the DC motor driver as the mobile robot actuator. It also presents PWM value as the output of FLC. The steady-state value for PWM is given with the maximum value of 250 achieved at 24 seconds, in which the mobile robot is expected to run at high velocity. In the 31 seconds, the ultrasonic sensor detects the obstacle. Thus it decreases to approximately 36 seconds. After 37 seconds, the ultrasonic sensor does not detect the obstacle, as it ends the braking which makes PWM signal begin to increase again till reaching the steady-state value. This process runs to 44 seconds where the mobile robot operation ends in the stop point, and the PWM value returns to zero.

Figure 16 shows the velocity response of the motor of the mobile robot system. The phenomenon of motor velocity response in the mobile robot system is also similar to the PWM value inputted to the motor driver. **Figure 16** also shows the response of motor velocity as the mobile robot system performance controlled by FLC. The steady-state value for the response of the motor velocity is 1500 rpm achieved at 24 seconds, in which the mobile robot is expected to run at high velocity. At 31 seconds, the ultrasonic sensor detects the obstacle. Thus, the value of the motor velocity is declining. After 36 seconds, the motor velocity starts to rise again until it reaches the steady-state value. This operation process completes in 44 seconds in the stop point and the motor velocity response becomes zero. Based on simulation work and experimental work, it can be seen that the results are very close in terms of the settling time and overshoot indicators, i.e. around 24 s and 0%, respectively.

The performance of obstacle avoidance is shown in **Table 2**. The test is performed with variations of obstacle distances namely near, quite near, medium, quite far, and far. The experiment is carried out several times. Based on the results of the obstacle reading, it can be concluded that the FLC system can read the obstacle accurately with the obstacle reading level of 100% correctly. The FLC system can also take action, and the mobile robots can avoid obstacles accurately.

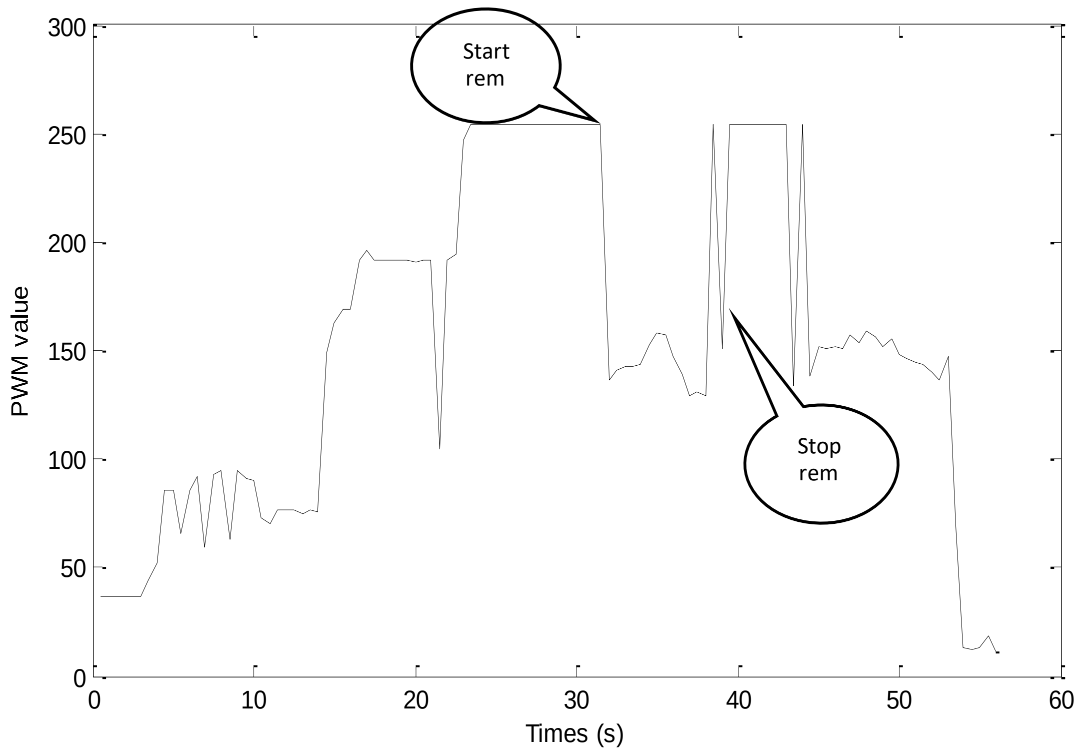


Figure 15. PWM signal response

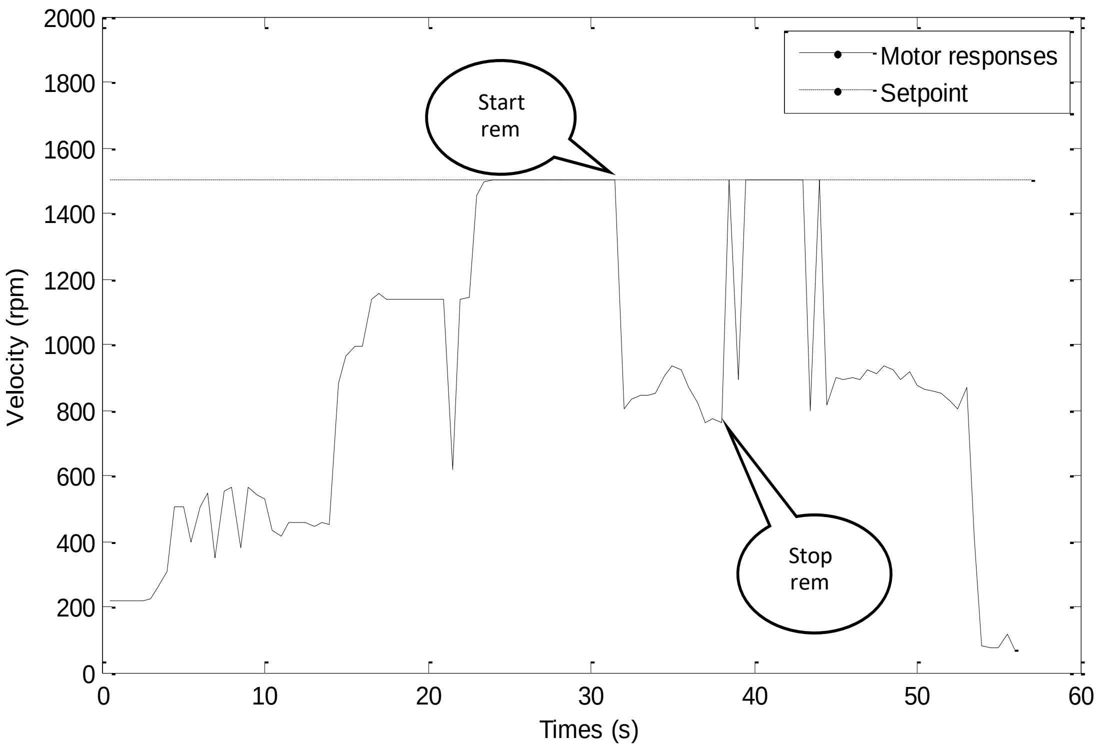


Figure 16. Velocity responses of DC motor

Table 2. Performance of obstacle detection

No	Obstacle distance	Experiment					average of detection success (%)
1	Near	1	2	3	4	5	100
2	Near	s	s	s	s	s	100
3	Quite Near	s	s	s	s	s	100
4	Quite Near	s	s	s	s	s	100
5	Medium	s	s	s	s	s	100
6	Medium	s	s	s	s	s	100
7	Quite Far	s	s	s	s	s	100
8	Quite Far	s	s	s	s	s	100
9	Far	s	s	s	s	s	100
10	Far	s	s	s	s	s	100

Note: s = success, f = false

The FLC control system for the obstacle avoidance system on the mobile robot functions appropriately. Moreover, the FLC system shows better and more accurate performance compared to the PID controller. Therefore, it can be concluded that the FLC control system is highly feasible to be applied to the mobile robot system to avoid obstacles. The results describe the fruitfulness of mobile robot in all areas for detecting and avoiding obstacles. In general, the ability of mobile robot to detect obstacles can be found in previous studies, but very few robots can detect obstacles as quickly as avoiding obstacles. Through the FLC design in this study, it is shown that mobile robot has the capability with good performance in detecting obstacles and avoiding obstacles in a fast time. The process of detecting and subsequently avoiding obstacles that are far away may be easy to do, yet this study has also conducted with experimental work for various distances in the position of obstacles namely near, quite near, medium, quite far and far distances. The results also present the comparison between FLC design with simulation works and that with experimental works for mobile robot in obstacle avoidance. Both works show similar results, which suggest that the validity of FLC design for mobile robot control is confirmed. In addition, FLC has a better performance compared to the PIC control scheme.

4. CONCLUSION

This study has presented the FLC control system for mobile robots in avoiding obstacles. This mobile robot system uses ultrasonic sensor input and input voltage regulation. The performance of FLC control on the mobile robot system to avoid obstacles has been simulated and compared with the PID control system. The controller performance is shown by the settling the time and overshoot values for the two types of controller. To validate the FLC system design, the FLC on experimental work is done. Both simulation and experimental works show very close results. This means that the FLC control system for the obstacle avoidance system on the mobile robot can function accurately. In conclusion, the FLC system shows a better performance compared to the PID controller.

5. ACKNOWLEDGEMENTS

The study was supported by Kemenristedikti (the grant of Penelitian Terapan with the contract number of 194/SP2H/LT/DRPM/2019). We thank Universitas Negeri Yogyakarta for providing facilities to conduct this study.

6. AUTHORS' NOTE

The author(s) declare(s) that there is no conflict of interest regarding the publication of this article. Authors confirmed that the data and the paper are free of plagiarism.