Speed and steering control system for self-driving car prototype

Rafsanjani\textsuperscript{1,2}, I P D Wibawa\textsuperscript{1}, and C Ekaputri\textsuperscript{1}

\textsuperscript{1} Electrical Engineering, School of Electrical Engineering, Telkom University, Bandung, Indonesia

E-mail: 2rafsanjani.official@gmail.com

Abstract. A self-driving car is a vehicle that can run autonomously using a control. There are two systems that are controlled in this research. The first is speed control functions to regulate the speed and movement of the self-driving car prototype by adjusting the PWM value on the DC motor and second, the steering control uses the Ackerman steering system with a servo motor as an actuator. Both are arranged using fuzzy logic control methods that adapt from the habits in driving a car, the goal self-driving car prototype can walk to follow the track, maintain the robot car in the middle of the lane, can adjust the speed when turning and can stop when there are obstacles or traffic lights. The results of the control design testing in this study are, the average error value in the simulation is 0.771008 for servo angle and 0.392072 for speed. The error value in the programming algorithm is 0.149712 for servo angles and 0.198168 for PWM DC motors. Robot cars in accordance with the logic of the fuzzy rules made. Self-driving car prototypes can run turns and follow trajectories with a success rate of 93.33%. The distance between the robot car and the track is 1.83 cm inside the track and 0.03 cm outside the track. The self-driving car prototype can adjust its speed and can stop when there are obstacles with an average distance of 29.89 cm.

1. Introduction
A self-driving car is a car that can move automatically in its environment. Some of the research that has been done uses a lot of PID controls to design a control system on a self-driving car and some research does separately between speed control and steering control[1]. In this research, a prototype of the car is designed, the construction of which is adapted from the real car. This research uses an approach such as when driving a real car. When driving there are different conditions and different actions in each condition such as for example when turning right but the position of the car is on the left side of the road so the steering must be rotated a lot towards the right but when turning right and the position of the car on the right then the steering just need to rotate slightly to the right and vice versa. There are also conditions where the turning angle that is passed varies the speed when turning also must be adjusted so as not to slip. Based on these conditions, the fuzzy logic control method was chosen to be able to produce actions that correspond to these conditions.

2. Literature Review

2.1. Self-Driving Car Prototype
A self-driving car prototype is a prototype vehicle that can move unmanned in its environment. Starting from the steering system and the driving system adapting to the vehicle[2]. Vehicle system which generally has four wheels two wheels are positioned in front and two wheels in the rear. The front wheels and rear wheels have a certain distance which is usually called the wheelbase (L) as well as the right wheel and the left wheel, commonly called front tread (t). The two front wheels have a function as a determinant of the direction of movement of the car while the two rear wheels function as a determinant of the movement and speed of the car or called two-wheel-drive (2WD)[3].

2.2. Ackerman Steering Geometry

Ackerman steering geometry is a geometric arrangement in the steering wheel of a car or other vehicle designed to solve the problem of the inside and outside of the turn which is capable of tracing a circle with a different radius. Ackerman’s geometry is composed of several constituent components, namely kingpins, steering arms, pivot point linkage, and tie rod[4][5]. Kingpins is a ball joint that functions for the steering wheel axis connected to each front wheel, the steering arms are an extension of the tie rod connector and also determine the difference in the turning angle between the right wheel and the left wheel. The linkage pivot point is a ball joint as a rotating axis connecting the steering arms and tie rod.

2.3. Fuzzy Logic Controller

Fuzzy Logic Controller is a method for determining an uncertain or vague value. Fuzzy logic was first introduced by Dr. Lofti Zadeh in 1965 concerning the fuzzy set theory. There are three main processes in designing fuzzy logic controllers, namely:

1. Fuzzification
2. Fuzzy inference
3. Defuzzification

The main processing of FLC occurs in the Fuzzy inference process where input into the process is a value that has gone through the process of fuzzification and also the rules that have been made. In the final process, defuzzification returns the value of the results of the Fuzzy inference process again to be a value.

3. Implementation

3.1. Block Diagram.

This section will explain the design of the system to achieve the goal. This self-driving car prototype system basically controls two systems, namely the steering system and also the rear-wheel-drive system. The block diagram for this system can be seen in Figure 1 and the hardware diagram can be seen in Figure 2 and hardware diagram can be seen in Figure 2.
The input from this system is obtained from the camera capture and then processed on the PC to obtain values. The value obtained is the value of the middle error is the difference from the midpoint of the path to the midpoint of the camera (ICETIR accepted #199), then the turning angle value is the angle value of the turn on the road (ICETIR accepted #200) and the distance between robot cars and obstacles (blue box obstacle and traffic light) (ICETIR accepted #201). The values that have been obtained are sent using serial communication to the microcontroller to control the actuator. This system controls two actuators, namely a servo motor and a DC motor.

The servo motor serves as the driver of the steering system. The input from the servo angle is the middle error value and also the turn angle value so that the self-driving car prototype can maintain the position in the middle of the road and turn in line. While the DC motor serves to regulate the value of the robot's car speed basically influenced by the distance to the obstacle and is also influenced by the value of the steering angle so that when the robot car turns the speed can adjust so as not to slip.

3.2. Hardware Designing

Ackerman steering as explained in chapter two of Ackerman Steering serves to determine the direction of motion of a self-driving car prototype made. As a driver, one servo motor is used which is connected to the constituent components of Ackerman steering[6]. Figure 3 shows the constituent parts of the Ackerman steering system. The rear-wheel-drive system functions to move forward and backward of the self-driving car prototype that is used but in this research the function used is only an advanced function. The rear-wheel-drive system is composed of several gears arranged in such a way as to increase and transmit power from the rear-wheel-drive system. As a driver there is one DC motor that rotates clockwise so that the self-driving car prototype can move forward. In figure 4 shows the constituent parts of the rear-wheel-drive system.

This robot car has a size ratio of 1: 10, which means that the size of the robot is 10 times smaller than the actual car. Some measures that must be considered in the use of self-driving car prototypes to adjust the design of the width, length and turning angle of the road. In figure 5 shows the shape and size self-driving car prototype.
Figure 5. Size and Specification

From the systems described above, we can specify the specifications of the self-driving car prototypes used this time described in table 1.

| NO | System                                | Specification                          |
|----|---------------------------------------|----------------------------------------|
| 1  | Self-driving car prototype drive       | Two-wheel drive (2WD)                  |
|    |                                       | Rear-wheel drive (RWD)                 |
| 2  | Range steering angle                  | 30 degree (60 - 120)                   |
| 3  | Turning radius                        | 171 cm                                 |
| 4  | Wheelbase                             | 26 cm                                  |
| 5  | Front tread                           | 19 cm                                  |
| 6  | Dimension                             | 39cm x 19cm x 27.5cm                   |
| 7  | Overall weight                        | 1,835 kg                               |
| 8  | Tire spokes                           | 3cm                                    |

3.3. Control Design

Fuzzy logic toolbox from Matlab is used to design fuzzy logic controller systems before being translated into the Arduino IDE programming language and as a comparison of the Arduino IDE programming algorithm. While Arduino IDE is used as a microcontroller programming device to process data and implement on hardware that has been designed in this research is a self-driving car prototype. The Arduino IDE programming algorithm is said to be in accordance with the rules of fuzzy logic controller rules if the output results are in accordance with the output of the fuzzy logic controller logic simulation in MatLab[7]. Process:

1. Fuzzification

The first step in FLC is the process of changing the input value of firm value (crisp) into a form of fuzzy set. In this process there is a calculation of the membership function value. Membership function (MF) is a curve that determines how each point in the input space is mapped to the value of membership (or membership level) between 0 and 1. value (or membership level) between 0 and 1.

There are three linguistic variables for each input. The servo angle output has seven linguistic variables and for PWM values the DC motor has four linguistic variables. Furthermore, determine the range and value of the membership function for each input as follows:

1. ME membership function with range [-8 8]
   - Left trapezoid function [-7 -8 -3 0]
   - The function of the center triangle [-1 0 1]
   - Right trapezoid function [0 3 8 9]
2. SB membership function with range [60 120]
   - Left trapezoid function [59 60 80 90]
• Straight triangle function [75 90 105]
• Right trapezoid function [90 100 120 121]
3. Distance membership function with range [39 79]
  • Near trapezoid function [38 39 49 59]
  • Medium triangle function [49 59 69]
  • Far trapezoidal function [49 59 79 80]
After the value is determined then membership functions are made in the graph. Figures 6 through 8 show the graph of membership functions of each input.

![Figure 6. Membership Function Input 1](image1)

![Figure 7. Membership Function Input 2](image2)

![Figure 8. Membership Function Input 3](image3)

![Figure 9. Servo Output](image4)

![Figure 10. DC Motor Output](image5)

2. Fuzzy Inference

Fuzzy Inference is the process of formulating a mapping from input given to output using fuzzy logic. The mapping then provides the basis from which decisions can be made or patterns that are seen [the court of law]. The basic rules used in fuzzy logic are "if ... then ... the basic rules are determined according to the desired output[8].
Table 2. Code for Table 3

| ME (Middle Error) | SB (Turning Angle) | Distance |
|-------------------|--------------------|----------|
| Left = 0          | Left = 0           | Near = 0 |
| Center = 1        | Straight = 1       | Medium = 1|
| right= 2          | Right = 2          | Far = 2  |

Table 3. Rules

| FLC Rules | Servo Motor Angle | Output 2 (PWM DC motor) | FLC Rules | Output 1 (Servo Motor Angle) | Output 2 (PWM DC motor) |
|-----------|-------------------|-------------------------|-----------|----------------------------|-------------------------|
| 000       | LL                | ST                      | 112       | L                          | FS                      |
| 001       | LL                | N                       | 120       | R                          | ST                      |
| 002       | LL                | N                       | 121       | R                          | N                       |
| 010       | RL                | ST                      | 122       | R                          | N                       |
| 011       | RL                | N                       | 200       | LO                         | ST                      |
| 012       | RL                | N                       | 201       | LO                         | SL                      |
| 020       | RO                | ST                      | 202       | LO                         | SL                      |
| 021       | RO                | SL                      | 210       | LL                         | ST                      |
| 022       | RO                | SL                      | 211       | LL                         | N                       |
| 100       | L                 | ST                      | 212       | LL                         | N                       |
| 101       | L                 | N                       | 220       | RL                         | ST                      |
| 102       | L                 | N                       | 221       | RL                         | N                       |
| 110       | S                 | ST                      | 222       | RL                         | N                       |
| 111       | S                 | N                       |           |                            |                         |

3. Defuzzyfication

The last process is Defuzzyfication where in this process the input obtained is from the results of Fuzzy Inference. Inversely proportional to the fuzzyfication process, defuzzyfication changes the set of fuzzy default values to crisp values. Figure 11 shows the results defuzzyfication.

![Figure 11. Results of Defuzzyfication](image-url)
4. Experiments and Results
Control design testing consists of 3 tests, including:

1. Testing the Fuzzy Logic Algorithm on Arduino IDE
   The results of this test are the average error of simulation on MatLab is 0.771008 for the servo angle value and 0.392072 for the PWM value of the DC motor. While the average error of the programming algorithm in Arduino is 0.149712 for the servo angle value and 0.198168 for the PWM value of the DC motor.

2. Testing the relationship of PWM values, rear-wheel RPM, voltage and speed
   This test aims to see the linearity between the PWM, the voltage coming out of the motor driver, the RPM generated at the rear wheels and the speed of the robot car. Then compared with the results of the count and the results obtained as in figure12-14.

3. Whole system testing
   The results of this test are robot cars can execute curves from the range of 60-120 degrees with a success rate of 93.333%. The robot car is in accordance with the fuzzy rules that are made, the speed of the robot car adjusts to the steering angle shown in figure 15. It was found that the deviation of the robot car towards the inner track was 1.83 cm and the outer track was 0.03 cm. Robot cars can stop when there are obstacles/traffic lights with an average distance of 29.89 cm.
5. Conclusions
The control of the self-driving car prototype that is made is in accordance with the design of the self-driving car prototype that can follow the trajectory but the self-driving car prototype several times is still out of the track because the self-driving car prototype does not have the right direction when turning. Self-driving car prototypes can slow down when turning to adjust the steering angle and can stop when there are obstacles or traffic lights.

6. Acknowledgments
This research was funded by the Telkom University for research collaboration with Hanbat National University and Research Institute for Tea and Cinchona.

7. References
[1] M. A. Sotelo, “Lateral control strategy for autonomous steering of Ackerman-like vehicles,” Rob. Auton. Syst., vol. 45, no. 3–4, pp. 223–233, 2003.
[2] T. H. S. Li, S. S. Jian, and M. C. Tsai, “Design and implementation of fuzzy trajectory following and planning control for mobile robots,” IEEE Reg. 10 Int. Conf. Electr. Electron. Technol., no. 01, pp. 453–458, 2001.
[3] M. V. Rajasekhar and A. K. Jaswal, “Autonomous vehicles: The future of automobiles,” 2015 IEEE Int. Transp. Electrific. Conf. ITEC-India 2015, pp. 1–6, 2016.
[4] R. Mattheai et al., “Autonomous Driving,” in Handbook of Driver Assistance Systems, Cham: Springer International Publishing, 2016, pp. 1519–1556.
[5] I. Tejado, V. Milanés, J. Villagrá, J. Godoy, H. HosseinNia, and B. M. Vinagre, “Low speed control of an autonomous vehicle by using a fractional PI controller,” IFAC Proc. Vol., vol. 44, no. 1 PART 1, pp. 15025–15030, 2011.
[6] F. FAHMIZAL, B. B. MURTI, D. B. PRATAMA, and A. MAYUB, “Kendali Logika Fuzzy pada Car Like Mobile Robot (CLMR) Penjejak Garis,” ELIKOMIKA J. Tek. Energi Elektr. Tek. Telekomun. Tek. Elektron., vol. 6, no. 3, p. 451, 2018.
[7] C. Mathworks, Fuzzy Logic Toolbox™User’s Guide R 2015 b. 2015.
[8] J. Angeles, Fundamentals of Robotic Mechanical Systems: Theory, Methods, and Algorithms. 2003.