BAREKENG: Journal of Mathematics and Its Applications
December 2024 Volume 18 Issue 4 Page 2093–2112
P-ISSN: 1978-7227 E-ISSN: 2615-3017
https://doi.org/10.30598/barekengvol18iss4pp2093-2112

PID PATH FOLLOWING CONTROL SYSTEM DESIGN ON UNMANNED AUTONOMOUS FORKLIFT PROTOTYPE

Teguh Herlambang[1*], Katherin Indriawati[2], Reza Maliki Akbar[3], Hendro Nurhadi[4]

[1]*Information System Department, FEBTD, Universitas Nahdlatul Ulama Surabaya*
Jl. Raya Jemursari 51-57, Surabaya, 60237, Indonesia

[2,3,4]*Department of Industrial Mechanical Engineering, Institut Teknologi Sepuluh Nopember*
Jl. Teknik Industri, Keputih, Sukolilo, Surabaya, 60111, Indonesia

*Corresponding author's e-mail: * **teguh@unusa.ac.id***

Article History:

Received: 23rd November 2023
Revised: 4th May 2024
Accepted: 1st July 2024
Published: 19th October 2024

Keywords:

Material Handling;
Unmanned Autonomous Forklift;
Rotary Encoder;
Odometry;
PID Control.

ABSTRACT

One of the technologies often used in material handling and material transport is forklift. Conventionally forklifts are operated by human operators. For efficiency, to improve security, safety, and occupational health and to minimize the risk of operators, forklifts can be automated. Therefore, the idea of unmanned autonomous forklift innovation was introduced. In this final project, a motion control system was designed for an unmanned autonomous forklift prototype both in simulation and hardware using PID control techniques, and the odometry system was equipped with a rotary encoder. In the simulation, the controlled variable was the distance and the manipulated variable was the force on the vehicle. Meanwhile, in the prototype, the controlled variable was distance and the manipulated variable was the motor pulse. In the simulation stage the PID control parameters were applied, $K_p = 1191.6$; $K_i = 340.69$; $K_d = 375.54$. *with an error of 0.32% in the simulation. The PID control parameters were applied to the prototype, that is,* $K_p = 0.97$; $K_i = 0.3$; $K_d = 1$. *The distance tests were done with variation of 50 cm to 200 cm (25 cm intervals). One variation of the distance experiment was done 5 times. The average absolute error resulted was 2.36 cm.*

How to cite this article:

T. Herlambang, K. Indriawati, R. M. Akbar and H. Nurhadi., "PID PATH FOLLOWING CONTROL SYSTEM DESIGN ON UNMANNED AUTONOMOUS FORKLIFT PROTOTYPE," *BAREKENG: J. Math. & App.*, vol. 18, iss. 4, pp. 2093-2112, December, 2024.

Journal homepage: *https://ojs3.unpatti.ac.id/index.php/barekeng/*
Journal e-mail: *barekeng.math@yahoo.com; barekeng.journal@mail.unpatti.ac.id*

Research Article • Open Access

1. INTRODUCTION

The process of loading, unloading, and transporting materials is one of the main problems for every production site in the industry, and it has a big impact on product costs [1]. Therefore, efforts to find a system continuously profitable and flexible become very important. The high demand for automation significantly changes industrial and warehouse operations [2].

Forklifts normally operated by humans need improving to accomplish efficient automated industrial tasks in material handling processes, while at the same time their operational safety must be ensured [3]. With rapid advances in sensor and computer technology, the chance of developing an unmanned autonomous forklift is high. The task of driving a forklift requires lengthy training for drivers to ensure safe and secure operation [4]. In addition, the work is quite tiresome since it is commonly monotonous repetition activities [5]. Therefore, it is an ideal candidate for automation.

With the recent decline in the price of sensors and computing devices, some researchers realize that mobile robot technology is ready enough to run forklifts in warehouses and/or industry. This also supports the development of the industrial revolution 4.0 [6]. By only receiving commands from a human supervisor, the unmanned autonomous forklift can safely carry out tasks that require interaction with people and objects in an unfamiliar work environment. This includes the ability to detect and manipulate pallets and their cargo, both on the ground and in storage areas, as well as to map a path and then to avoid obstacles. Even though several products are on the market, the costs are still high. Then there is still room for developing systems that are simple, user-friendly, and reasonably priced.

With those problems, this final project aims to design and develop a forklift able to move independently, to lift cargo and to move it to the target location. The forklift must be able to map the work environment and to move automatically or to allow human control intervention when needed.

2. RESEARCH METHODS

2.1 Kinematics and Dynamics of Forklift-Like Robot

The forklift kinematics and dynamics modeling is taken from the Forklift-Like Wheeled Robot model reference [7].

2.1.1 Kinematics Model

Figure 1 shows a forklift-like mobile robot model, wherever l is the distance between the front axle and the rear axle; w denotes the half distance between the two wheels on each axle, ρ the turning radius from the origin of the robot; $\varphi, \varphi_r, \varphi_l$ denotes the steering angle, the right wheel angle, and the left wheel angle. It is assumed that the steering system is designed without slip on the wheels or the slip is ignored. Thus, $\tan\varphi = \frac{l}{\rho}, \tan\varphi_r = \frac{l}{\rho - w}, \tan\varphi_l = \frac{l}{\rho + w}$ follows as in Figure 1.

The variable ρ is eliminated from the tangent function, resulting in $\tan\varphi_r = \frac{l\tan\varphi}{(l - w\tan\varphi)}$, $\tan\varphi_l = \frac{l\tan\varphi}{(l + w\tan\varphi)}$ the left wheel angle and the right wheel angle can easily be calculated to give l and w as a function of steering angle φ. The result is that the control of the steering angle φ is equal to φ_r and φ_l.

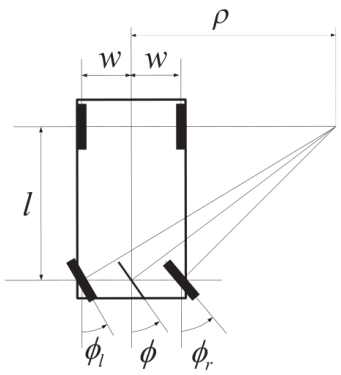

Figure 1. The Steering Shape of a Forklift-Like Wheeled Robot

Figure 2 shows the forklift kinematic parameters, where (x, y) is the position of the origin of the robot in the frame; (x_c, y_c) is the position of the robot's center of mass; lc is the distance from the origin of the robot (x, y) to the center of mass; v is the speed of the robot at (x, y); ρ is the origin robot's turning radius (x, y); φ and θ are the steering angle and the heading angle of the forklift.

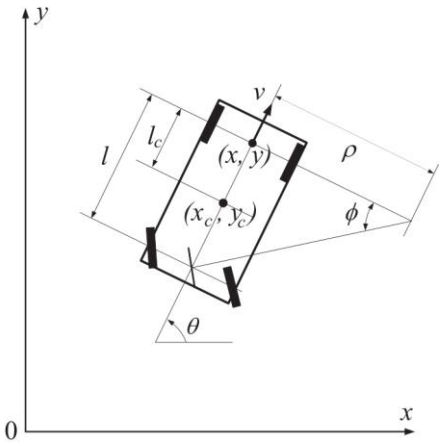

Figure 2. The Kinematic Parameters of the Forklift

Based on the condition of ignoring the wheel slip, the nonholonomic kinematic constraints can be reduced to:

$$\dot{x} = v\cos\theta \tag{1}$$

$$\dot{y} = v\sin\theta \tag{2}$$

$$\dot{\theta} = -\frac{v}{l}\tan\varphi$$

Where the last constraint is derived from $\dot{\theta} = -\frac{v}{\rho}$ with $\rho = \frac{l}{\tan\theta}$, as ρ is defined in **Figure 2**.

2.1.2 Dynamics Model

The forklift-like mobile robot motion equations are derived by applying Lagrange's equations. In this derivation, the potential energy of the robot is assumed to be zero. The kinetic energy of the robot under consideration is the sum of the translational energy of the robot's mass, the rotational energy of the four wheels around the horizontal axis, the rotational energy of the robot about the vertical axis passing through the robot's center of mass, and the rotational energy of the four wheels around the vertical axis of the robot passing through the robot's center of mass. Referring to **Figure 2**, the average speed of the rear wheels is equal to φ times that of the front wheels due to steering.

Since the kinetic energy is the same as the Lagrangean L for the forklift-like robot, the robot's motion equation can be derived using the Lagrange equation [8] for v and φ.

$$M\dot{q} + c + Dq = f \tag{4}$$

Where q˙ is a $2x1$ state vector defined as $\dot{q} \equiv (v, \varphi^{\cdot})\, T$; f is a 2×1 input vector defined as $f \equiv (fv, \tau\varphi)$ T; matrix $M_{2\times2}$ positive-definite symmetric, matrix 2×1 Coriolis and centrifugal force vector c, and matrix 2×2 viscous damping, D is defined as follows:

$$M = \begin{bmatrix} \left(m + \frac{2}{r^2}Jh\right) + \frac{2}{r^2}J_h \sec^2\varphi \\ +\frac{1}{l^2}(l_c^2 m + J_b + 4J_v)\tan\varphi - \frac{2}{l}J_v \tan\varphi \end{bmatrix} \quad \begin{matrix} -\frac{2}{l}J_v \tan\varphi \\ \\ 2J_v \end{matrix} \tag{5}$$

$$C = \begin{Bmatrix} \left(2\left[\frac{1}{l^2}(l_c^2 m + J_b + 4J_v) + \frac{2}{r^2}J_h\right]\tan\varphi \sec^2\varphi v\dot{\varphi} - \frac{2}{l}J_v \sec^2\varphi\dot{\varphi}^2\right) \\ -\left[\frac{1}{l^2}(l_c^2 m + J_b + 4J_v) + \frac{2}{r^2}J_h\right]\tan\varphi \sec^2\varphi v^2 \end{Bmatrix} \tag{6}$$

$$D = \begin{bmatrix} d_v & 0 \\ 0 & d_\varphi \end{bmatrix} \tag{7}$$

Where r is the radius of the wheel; m is the mass of the robot; J_b is the robot's moment of inertia about a vertical axis passing through the robot's center of mass; J_h and J_v are the moment of inertial mass of the wheel about the horizontal and vertical axes passing through the center of mass of the wheel, respectively; d_v and d_φ denote the coefficient of *viscous damping*; f_v and τ_φ denote the driving force and the steering torque.

2.1.3 Trajectory Generator

In this modeling, the kinematic control is applied to create a real-time trajectory. The e_B tracking error is determined based on the xy base frame as shown in **Figure 3**.

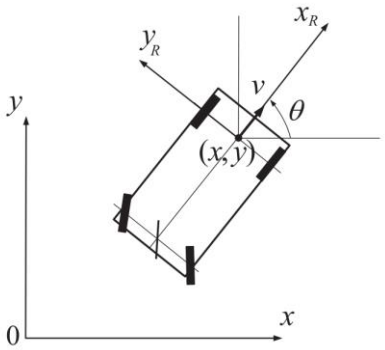

Figure 3. Two Different and Moving Robot Frames

$$e_B \equiv \begin{Bmatrix} e_{Bx} \\ e_{By} \\ e_{B\theta} \end{Bmatrix} = \begin{Bmatrix} x_d - y \\ y_d - y \\ \theta_d - \theta \end{Bmatrix} \tag{8}$$

Of which x_d, y_d, θ_d is the desired trajectory of x, y, θ in the base frame.

After that, the e_B *tracking error* on the base frame is transformed into e_R, *the local robot frame tracking error* x_R, y_R is connected to the origin robot as shown in **Figure 3**.

$$e_R \equiv \begin{Bmatrix} e_{Bx} \\ e_{By} \\ e_{B\theta} \end{Bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{Bmatrix} e_{Bx} \\ e_{By} \\ e_{B\theta} \end{Bmatrix} \tag{9}$$

Then, the kinematic error equation is derived by differentiating **Equation (9)**, referring to the kinematic constraints (1) – (3)

$$\dot{e_x} = \dot{\theta} e_y - v + v_d \cos e_\theta \tag{10}$$

$$\dot{e_y} = -\dot{\theta} e_x + v_d \sin e_\theta \tag{11}$$

$$\dot{e_\theta} = -\dot{\theta}_d - \dot{\theta} \tag{12}$$

By considering the tracking error in the local robot frame, the robot's movement is controlled from the robot's point of view, just like driving a car. The kinematic control scheme is designed based on the kinematic error **Equation (10)** - **Equation (12)**. The kinematic control scheme calculates the reference speed v and the reference time rate of the heading angle $\dot{\theta}$ of the robot.

$$v_r = v_d \cos e_\theta + k_x e_x \tag{13}$$

$$\dot{\theta}_r = \dot{\theta}_d + \frac{k_y}{k_z} v_d e_y + k_\theta \sin e_\theta \tag{14}$$

2.1.4 Simplifying the Dynamic Model

For the simplicity of the control design, the dynamic model is simplified. In practice the steering angle is usually not large for a 4 wheeler while in motion. Then all $tan\,\varphi$ can be neglected in nonlinear dynamics (4); However, the effect of the v^2 term on the steering dynamics (φ) may be too significant for high speed driving. Thus, the dynamic model (4) can be simplified to

$$\dot{v} = \frac{1}{m_v} (-d_v v + f_v) \tag{15}$$

$$\ddot{\varphi} = \frac{1}{J_\varphi} \left(-d_\varphi \dot{\varphi} + c_\varphi \tan\varphi \sec^2\varphi\, v^2 + \tau_\varphi\right) \tag{16}$$

Of which

$$m_v = m + \frac{4}{r^2} J_h \tag{17}$$

$$J_\varphi = 2J_v \tag{18}$$

$$c_\varphi = \frac{1}{l^2}(l_c^2 m + J_b + 4J_v) + \frac{2}{r^2} J_h \tag{19}$$

2.2 Methodology Flowchart

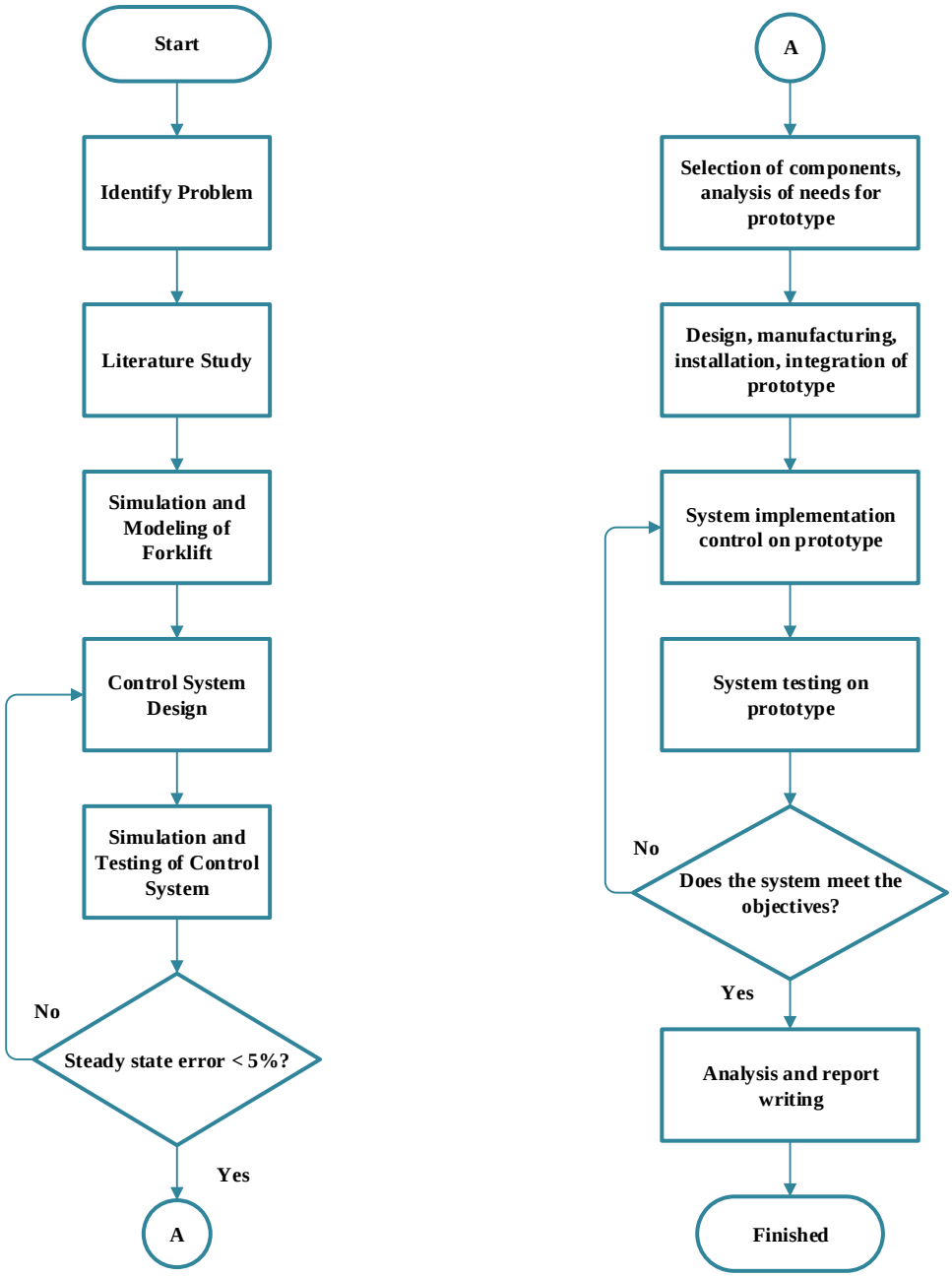

Figure 4. Research Methodology Flowchart

The methodology used to achieve the objectives of this final project was explained through the flowchart as shown in **Figure 4** describing the stages of the research starting from identifying the problem, studying the literature, up to making the modelling along with the simulation of the unmanned autonomous forklift to be developed and studied. Modelling is the key-part in developing the control system design, then it is simulated and tested until a steady-state error criterion of less than 5% is fulfilled. After designing and testing the control system for the forklift, a needs analysis was carried out, component selection was done, and its prototype design was developed. The prototype design was realized by manufacturing, installing and integrating the system as a whole both in terms of mechanics and electronics. When all installed and integrated, the control system on the unmanned autonomous forklift prototype was implemented and then tested for a certain distance. During the testing, of course, the testing data were obtained for the unmanned autonomous forklift prototype, after that, those data were processed to be interpreted as information on the performance and success of the unmanned autonomous forklift prototype.

2.2.1 Literature Study

A study of the existing problems was carried out using the literature study method applied in the previous studies. Several references describing the design and development of various unmanned autonomous forklifts in the previous studies are used as a reference in this final project [9][10].

2.2.2 Forklift Modelling

The forklift modeling is taken from the model reference of Forklift-Like Wheeled Robot [7], as the kinematic and dynamic modeling is derived for a forklift with four wheels. This robot is a car-like type where the system is non-linear and coupled. Kinematically, the time rate of the heading angle is proportional to the driving speed and the tangent angle of the steering angle. Thus, the heading angle of the car-like robot cannot be controlled when the driving speed is zero [11]. So, the steering angle as well as the driving speed must be controlled simultaneously to fulfill the directional control of the car-like robot [12].

The forklift-like mobile robot modeling is almost similar to car-like mobile robot. The main difference is the rear-steering system, while car-like robots use the front-steering system. The dynamics of the car-like robot and forklift-like robot are non-linear and coupled. Forklift-like robots are also included in the non-minimum-phase dynamic systems, which makes the control even more complex and complicated

2.2.3 Control System Design, Simulation and Testing

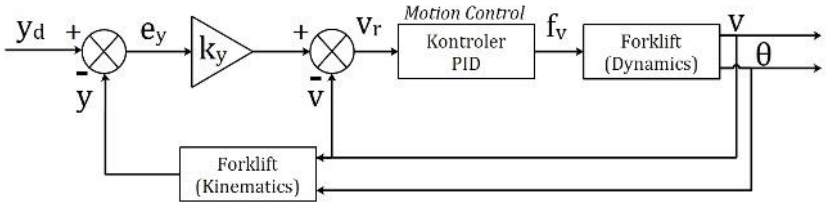

Figure 5. Motion control block diagram of forklift prototype

Figure 5 shows a motion control block diagram for an unmanned autonomous forklift prototype. Each block has an equation taken from reference [7]. In the forklift kinematics block, there are Equation (1), Equation (2), and Equation (3) which are the kinematic equations of the forklift-like robot itself. Since the forklift moves on a straight trajectory, parameter $l = 4\ m$, then the value $\theta = 90^o$, thus

$$\dot{x} = 0 \tag{20}$$

$$\dot{y} = v \tag{21}$$

$$\dot{\theta} = 0 \tag{22}$$

The forklift dynamics is used as a plant with equations (15) and (16) and both equations are simplifications of Equation (4). Variable v_r is obtained from the trajectory generator, functioning as a real-time trajectory generator referring to Equation (13), that is

$$v_r = k_x \times e_x \tag{23}$$

The motion control block uses the common PID equation, written as follows [13].

$$f_v = K_p e(t) + K_i \int_0^t e(t)dt + K_d \frac{de(t)}{dt} \tag{24}$$

With equation e, that is

$$e = v_r - v \tag{25}$$

After deriving the equation mathematically is done, the modeling simulation is carried out on the Matlab Simulink application [14]. The simulation follows according to the motion control block diagram in Figure 5 so that the equations in Figure 5 are transformed in the form of blocks, subsystems in Simulink. The overall Simulink block diagram is as in the attachment.

When the simulation was carried out the control system was tested to meet the system criteria with a steady-state error of less than 5% **[15], [16]**. Therefore, fine tuning was done during the control system testing to obtain Kp, Ki, Kd parameters matching the criteria as set. The performance of the control system was also tested in terms of the desired distance (setpoint) and the distance traveled.

2.2.4 Selection of Components and Analysis for Equipment Needs

After the simulation is carried out, the data resulted from the simulation are obtained, and useful for determining the components and equipments needed in the system.

The main components needed in making this unmanned autonomous forklift prototype are:

a. Car-like 4WD robot prototype and its tires
b. DC motor with gearbox, 4 pieces (2 for position control is enough)
c. Arduino Mega 2560 microcontroller, 1 piece
d. FC-03 module, 2 pieces
e. L298N Drivers, 1 piece

In practice, the 4WD car-like robot is used as a prototype system, for the search for a rotary encoder shall be in accordance with the physical conditions of the forklift prototype and also if a rotary encoder is installed the concern is that whether it can damage the forklift prototype or not. Besides, additional sensors and their amplifier circuits are also needed. Because the models are not exactly the same, the similarities are only on the front, that is, the placement of the accelerating motor, so the rear-steering function cannot be applied. Therefore the system can only move forward (or backward if the program is adjusted) according to the predetermined distance.

2.2.5 Design, Manufacture, Installation and System Integration

The following is a three-dimensional computer-aided design image of the prototype. In this final project the forklift prototype is represented by a 4WD car-like robot in **Figure 6**.

a. The forklift prototype : length : 4.27 m; width : 1.19 m; height : 2.5 cm; mass : 1000 kg; wheel diameter : 600 cm; number of motors : 3 pcs; number of tyres : 4 pcs

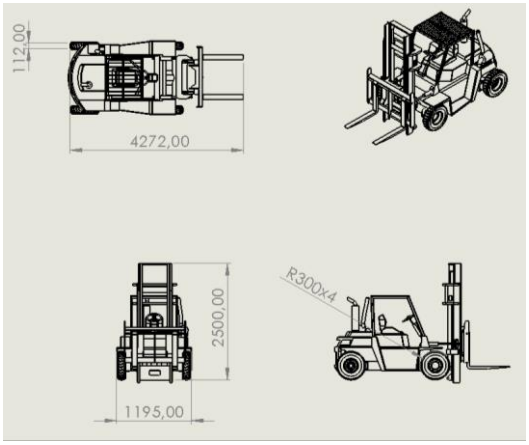

Figure 6. CAD forklift

b. Car-like robot 4WD : length : 25.7 cm; width : 14.92 cm; height : 3.1 cm; mass : 670 gram; wheel diameter : 66.10 cm; number of motors : 4 pcs; number of tyres : 4 pcs.

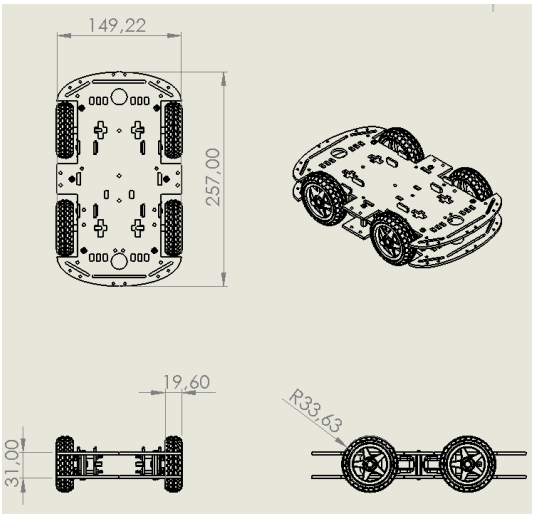

Figure 7. CAD car-like robot 4WD

Before the electronic design is made, an analysis of the planning of the components needed for the forklift is carried out in **Figure 7**.

On the unmanned autonomous forklift prototype, there are three driving motors, that is,

a. Front motor for acceleration
b. Motor fork, to lift and lower goods
c. Rear motor for a steering or rudder

However, since the rotary encoder could not be installed on the forklift prototype considering its physical condition, a 4WD car-like robot prototype was used. Thus, it only takes two motors in front of it to accelerate the movement. So, the electronic schematic design is as follows in **Figure 8** and **Table 1**.

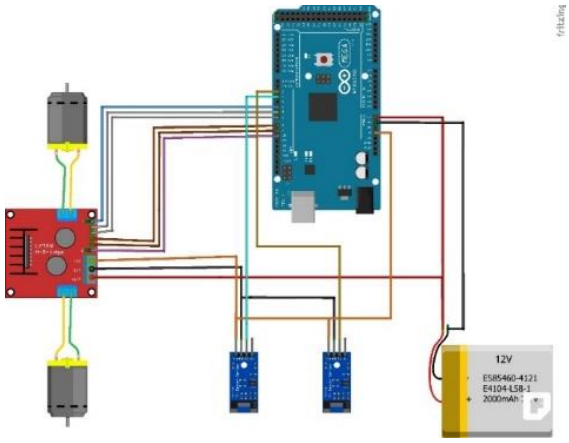

Figure 8. Electronic Schema

Table 1. List of System Inputs and Outputs

Input	Output
D10 Arduino	ENA
D9 Arduino	IN1
D8 Arduino	IN2
D5 Arduino	ENB
D7 Arduino	IN3
D6 Arduino	IN4
D3 Arduino	D0 FC-03 (right)
D2 Arduino	D0 FC-03 (left)
OUT1, OUT2	Motor +, Motor – (right)
OUT3, OUT4	Motor +, Motor – (left)
+12V	Vin Arduino, +12V driver
-12V	GND
5V Arduino	VCC FC-03, 5V driver
GND Arduino	GND FC-03, GND driver

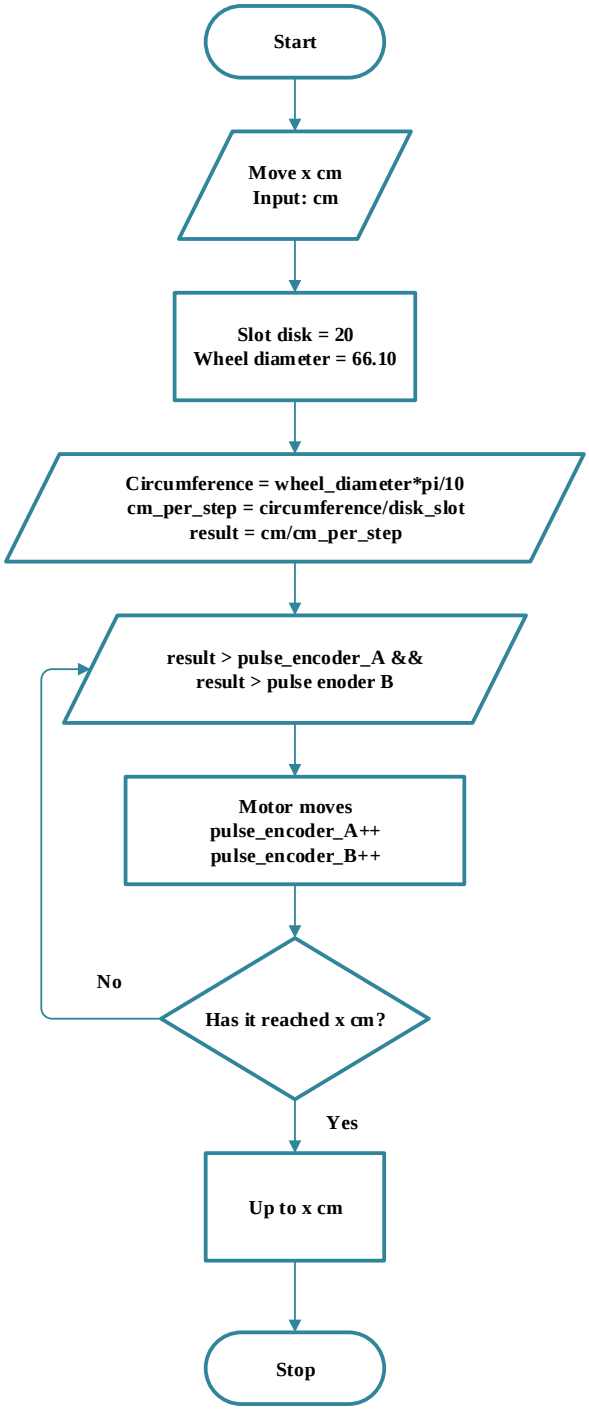

Figure 9. Odometry Program Flowchart

The odometry program is developed to reach a certain position with a distance of cm units. The program input is the value of the distance to travel. Then, the program processes where there is a slot disk encoder constant which is worth 20, meaning 20 holes on the rotary encoder disk. The wheel diameter entered into the program is 66.10 cm. Then, it comes to the odometric program. To get the travel results, the circumference of the wheel must be known then it is made into centimeters per step, which is the formula for the circumference of the wheel divided by the number of rotary encoder disk holes. After that the program calculates the encoder rotation according to the calculations already made and entered previously. If the result value is still greater than the encoder pulse, then the motor continues to move. The microcontroller will iterate continuously until the result value is less than the encoder pulse. If the result value is met, the robot will stop at a predetermined distance. The flowchart is shown as in **Figure 9**.

After doing all the designs, manufacture, installation, and system integration are carried out, combining mechanics, electronics, and forklift prototype programs in **Figure 10**.

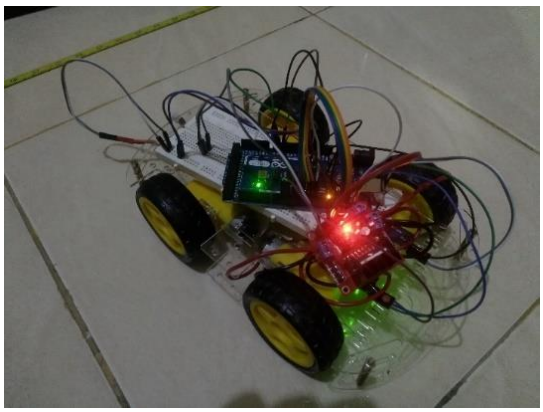

Figure 10. Prototype with all Integrated

2.2.6 Implementation of Control System on Prototype

The odometry program made has not used the additional form of a PID controller program, so that the output of the program has no controllers value, it only refers to the value in the program's while loop. By adding the PID controller program to the previous odometry program, the setpoint value can be maintained by the programmed controller. In its implementation the values of the gain parameters K_p, K_i, and K_d are entered. Then the setpoint value is also entered in in **Figure 11**. The PID controller program will calculate the error from the encoder pulses, the controller will execute it on the PWM motor. If the desired setpoint is reached, the brake function on the motor driver is activated, so that the forklift reaches the predetermined distance.

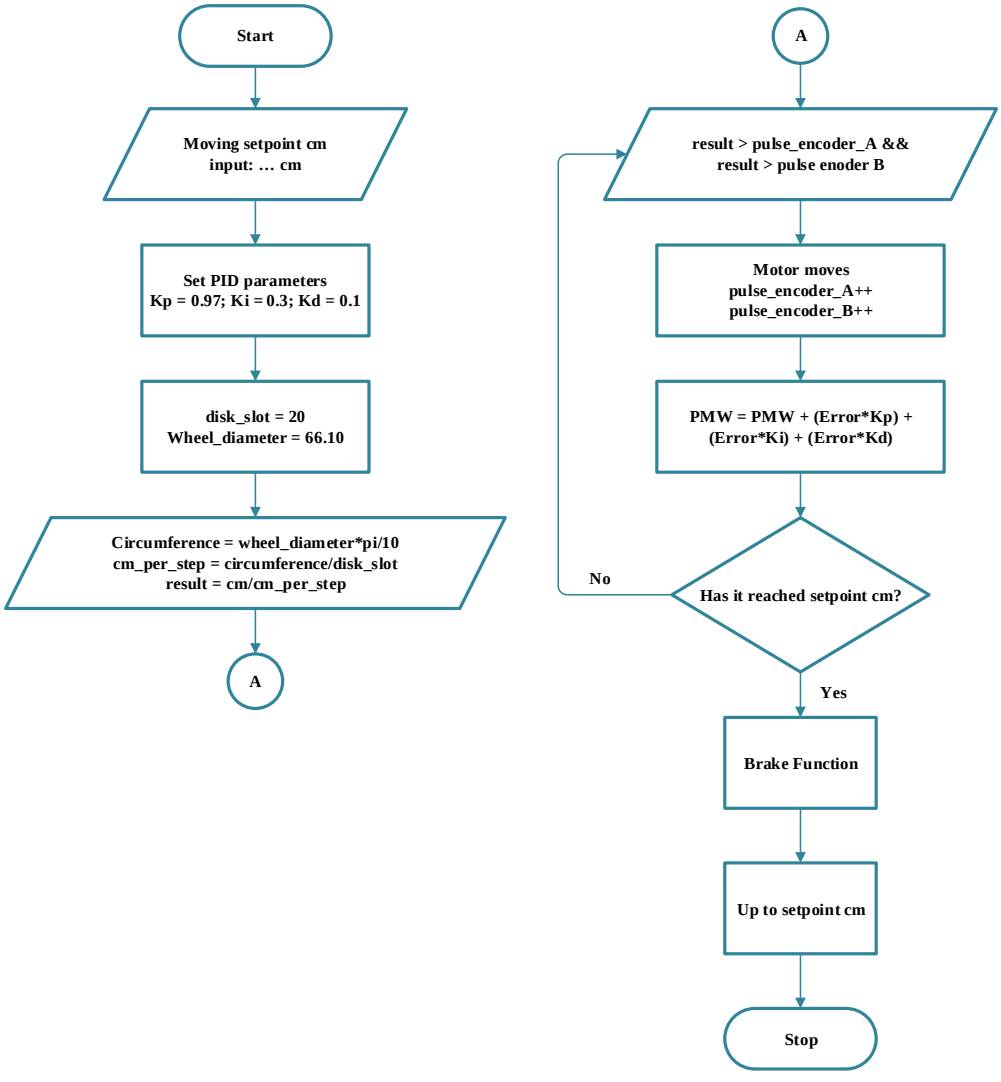

Figure 11. Implementation of Control System on Prototype

2.2.7 System Testing

The system testing is conducted by experimenting on the autonomous forklift robots with the predetermined methods, referring to the existing theories and literature review.

In this final project, what was developed and researched was a forklift prototype so that the testing was carried out on a smaller media. The 1:10 scale forklift of the original size was tested on a factory miniature table with an area of 2.4 m × 2.4 m in in **Figure 12**. The existing workstations on the factory table were adjusted to the existing area, not exactly the same as the PT. Altman. So, there are 6 points, that is, four for machining, one raw material, one place for storing goods having been completed in in **Figure 13**.

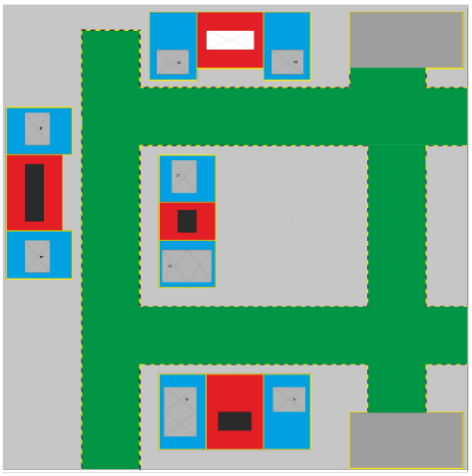

Figure 12. Table design of factory miniature

Figure 13. Result of Finished Factory Miniature Table

3. RESULTS AND DISCUSSION

3.1 Simulation Results

In the simulation, kinematic and dynamic parameters were entered to be processed in the workspace using m files, so that Simulink could read the parameters. The kinematic and dynamic parameters entered are $r = 0.3;\ m = 1000;\ l = 4;\ lc = 2;\ J_b = 300;\ J_h = 20;\ J_v = 10;\ dv = 200;\ dphi = 10$. In addition to the kinematic and dynamic parameters, the values for the gain parameter $k_x = 0.5$ were also entered; $k_y = 0.5;\ kz = 1;\ k_\theta = 1$. The trajectory was set from the trajectory generator under the conditions of $v_d = 5t\ m/s$ for $0 \leq t \leq 6\ s$ and $v_d = 30\frac{m}{s}$ for $t \geq 6$ s and $\theta_d = 0.2t$ for all $t \geq 0\ s$. The trajectory was a straight track with a setpoint of 50 meters. The simulation stop time on Simulink was set at 100 seconds.

In **Figure 14**, Fine tuning was carried out on the motion controller, by configuring the PID control parameters, namely $K_p = 1191.6;\ K_i = 340.69;\ K_d = 375.54$. The simulation result was a straight motion on the y axis.

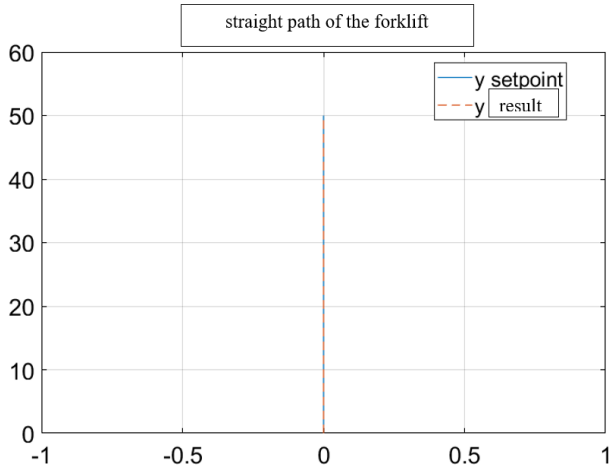

Figure 14. Simulation Result as Forklift Straight Trajectory

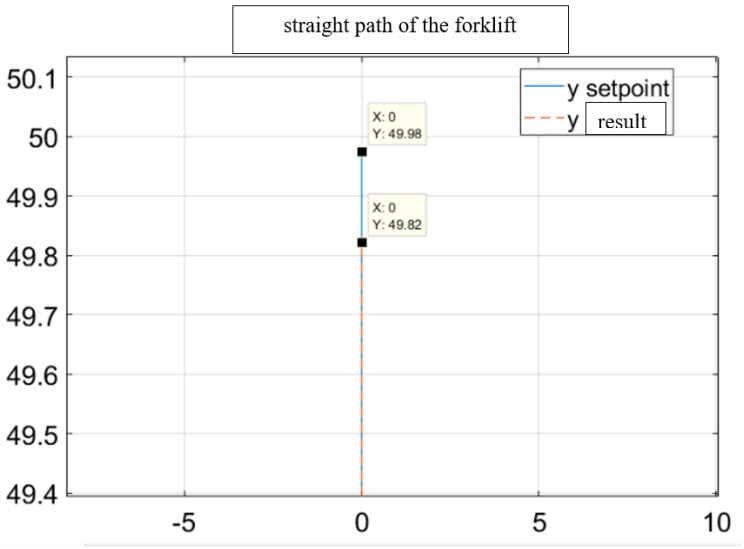

Figure 15. Forklift Track Chart Zoom

In **Figure 15,** the simulation results shows an error of 0.32% obtained from the difference in the value of y setpoint = 49.98 m and that of y result = 49.82.

In in **Figure 16**, the control signal response at speed (v) as the output of the control system is shown in Figure with a delay time of 4.12 seconds, a rise time of 7.46 seconds, a peak time of 10.03 seconds, an overshoot of 6.73%, and a settling time of 23.09 seconds.

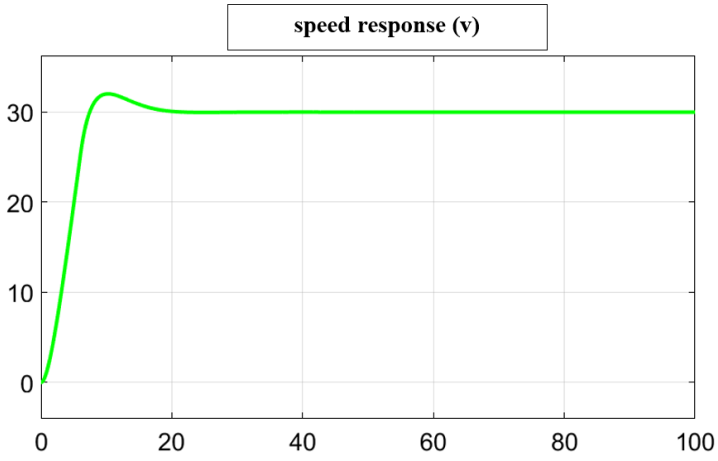

Figure 16. Speed Response Signal

Referring to a straight line with a distance of 50 meters, a travel time of 100 seconds, a graph of the distance travelled against the travel time can be obtained in **Figure 17** and **Figure 18**. On the y axis, $v = 0.5\ m/s$ is obtained, while on the x axis, $v = 0\ m/s$ (constant) for the forklift does not experience a change in angle, in accordance with a straight trajectory.

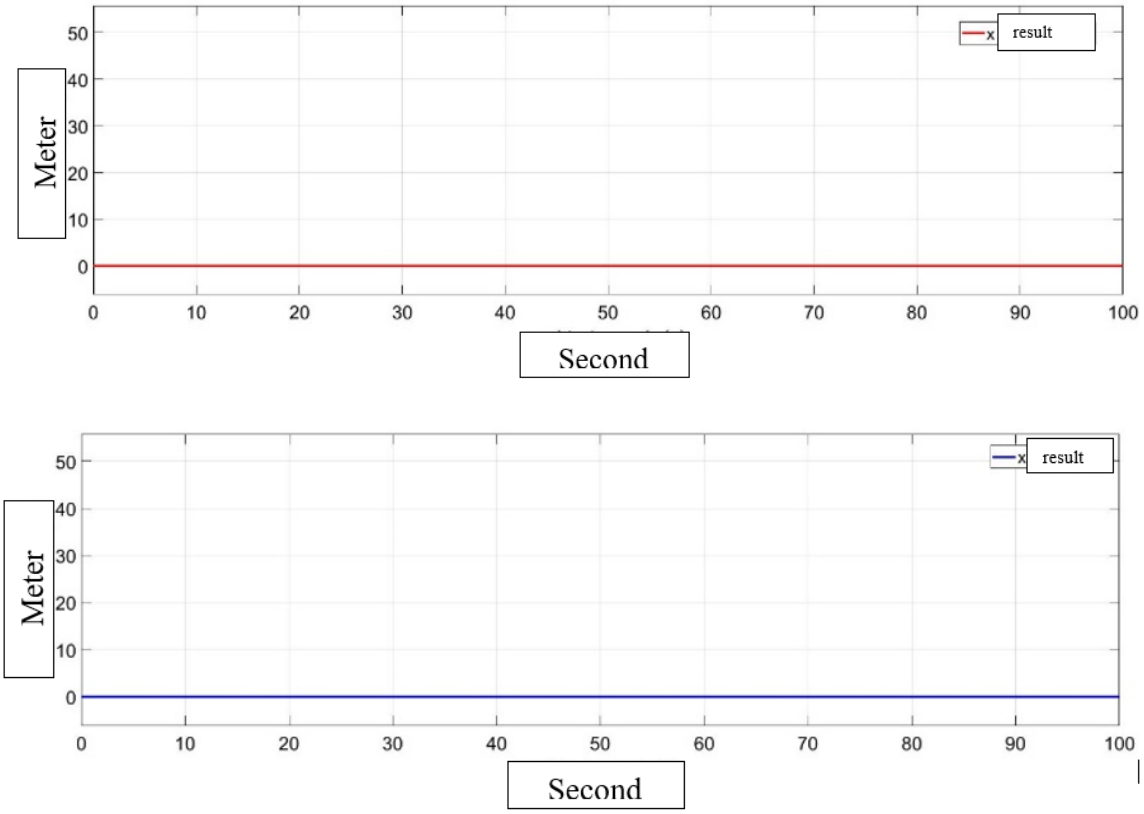

Figure 17. Graph of Distance Travelled Against Travel Time on the *X* Axis a) Travel Result, b) Setpoint

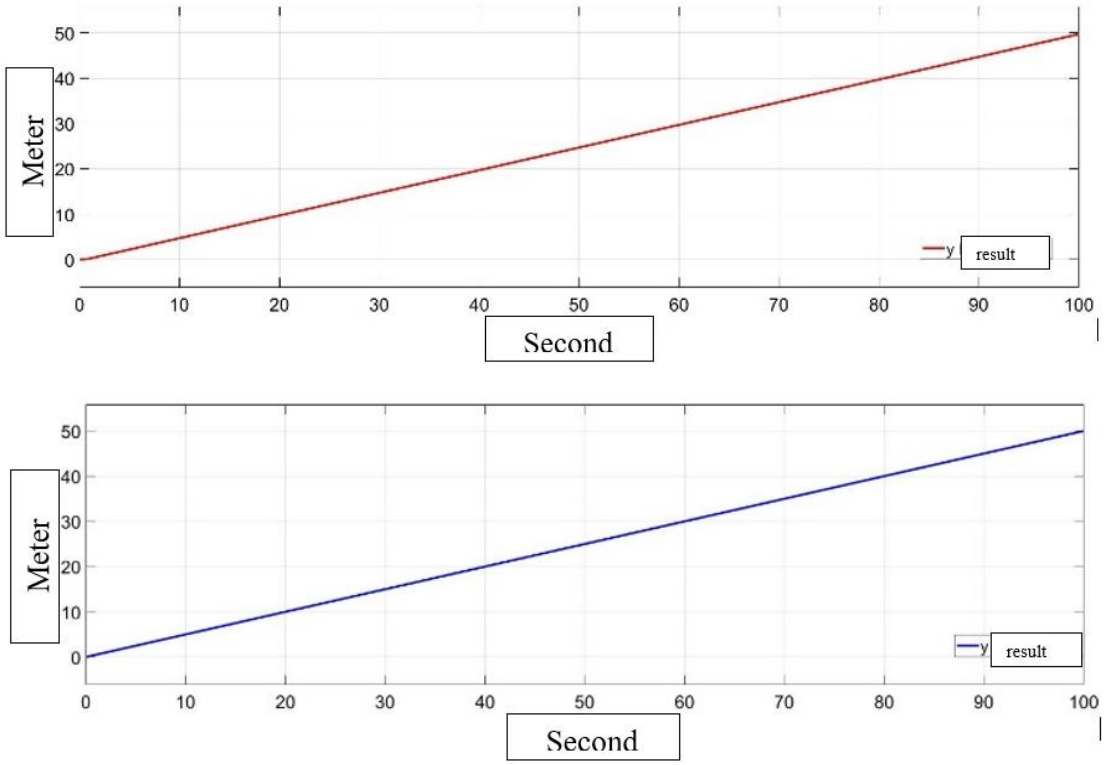

Figure 18. Graph of Distance Travelled Against Travel Time on the y Axis a) Travel Result, b) Setpoint

3.2 FC-03 Module Sensor Calibration

The problem frequently encountered when using the FC-03 sensor module is its poor signal conditioning. Another thing is that when this sensor is paired with the Arduino Mega microcontroller, the sensor reads 4 times more pulses than the encoder generates. This sensor is very sensitive to disturbances at the VCC and GND pins in **Figure 19**. Therefore a filter is needed to eliminate this interference.

At the start of the pulse signal there is a rebound, so actually the digital signal is not completely square. The data received is not perfect.

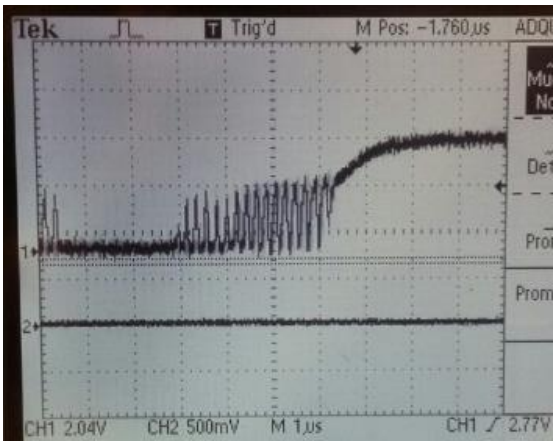

Figure 19. The Initial Pulse has a Rebound

This also happens at the end of the pulse, there is also a rebound. The Arduino is very sensitive and reads this rebound as a good pulse, which is in fact it is not a correct pulse in **Figure 20.**

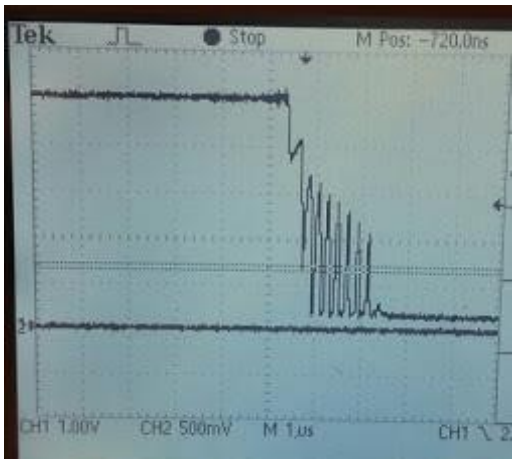

Figure 20. The Pulse at the End has a Rebound

For its anticipation, a 100nF capacitor (104) can be added, then a program with anti-debouncing is made. With that capacitor, the rebound signals can be filtered in **Figure 21**.

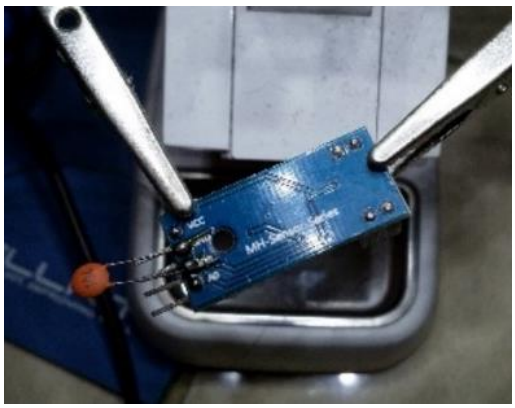

Figure 21. Added capacitor to reduce rebound

After adding the capacitor and modifying the program, the following results are obtained in **Figure 22** and **Figure 23**:

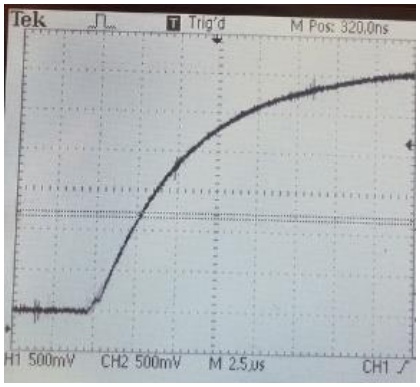

Figure 22. For the Pulse at Start, the Rebound is Gone

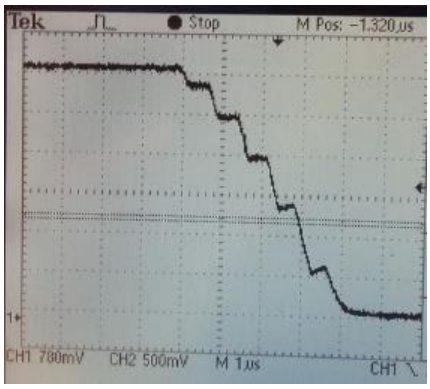

Figure 23. For the pulse at the end, the rebound gets less

3.3 Distance Testing on Prototype

The distance test was carried out using a serial communication program. The PID parameter tuning by trial and error method results in, for each parameter, $K_p = 0.97; K_i = 0.3; K_d = 1$ in **Figure 24** and **Figure 25**

```
double Kp=0.97, Ki=0.3, Kd=1;
double Setpoint, Input, Output;
PID myPID(&Input, &Output, &Setpoint, Kp, Ki, Kd, DIRECT);
// Constants for Interrupt Pins

#define pi 3.14159265358979323846264338327 95
const byte MOTOR_A = 3;  // Motor 2 Interrupt Pin - INT 1 - Right Motor
const byte MOTOR_B = 2;  // Motor 1 Interrupt Pin - INT 0 - Left Motor
```

Figure 24. Parameters K_p, K_i, K_d in the Program

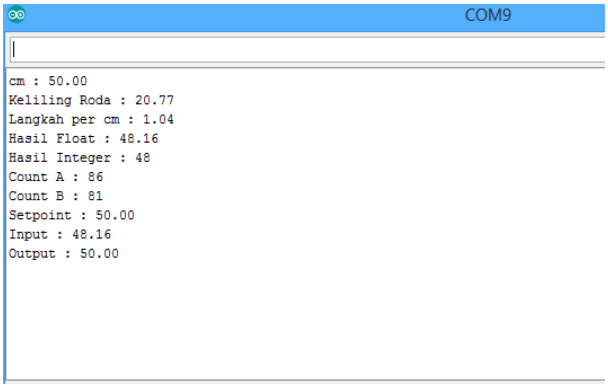

Figure 25. The Output of the Serial Monitor Program by PID

The distance test was carried out with a variation of the distance of 50 cm to 200 cm (25 cm intervals). One variation of the distance experiment was carried out 5 (five) times in Table 2. The following results were obtained:

Table 2. Distance Test Results

No	Experiment	Desired distance (cm)	Actual distance (cm)	Absolute error	Average of absolute error
1	1	50	55.1	5.1	1.74
	2	50	52.1	2.1	
	3	50	49.8	0.2	
	4	50	51.2	1.2	
	5	50	50.5	0.5	
2	1	75	80.1	5.1	2.54
	2	75	78.3	3.3	
	3	75	74.6	0.4	
	4	75	75.3	0.3	
	5	75	Hi.4	4.4	
3	1	100	103.2	3.2	3.68
	2	100	101.5	1.5	
	3	100	99.4	0.6	
	4	100	105.8	5.8	
	5	100	108.5	8.5	
4	1	125	126.7	1.7	1.7
	2	125	128.4	3.4	
	3	125	123.4	1.6	
	4	125	130.1	5.1	
	5	125	124.9	0.1	

No	Experiment	Desired distance (cm)	Actual distance (cm)	Absolute error	Average of absolute error
5	1	150	153.4	3.4	3.76
	2	150	159.7	9.7	
	3	150	160.2	10.2	
	4	150	148.4	1.6	
	1	50	55.1	5.1	
6	1	175	175.4	0.4	1.24
	2	175	178.4	3.4	
	3	175	177.5	2.5	
	4	175	173.4	1.6	
	5	175	176.5	1.5	
7	1	200	201.5	1.5	1.88
	2	200	204.3	4.3	
	3	200	199.5	0.5	
	4	200	198.7	1.3	
	5	200	205.4	5.4	

The results of the average distance test error on the prototype with variation in distance of 50 cm, 75 cm, 100 cm, 125 cm, 150 cm, 175 cm, and 200 cm were as follows:

a. At a distance of 50 cm with an average error of 1.74 cm (3.48%)
b. At a distance of 75 cm with an average error of 2.54 cm (3.39%)
c. At a distance of 100 cm with an average error of 3.68 cm (3.68%)
d. At a distance of 125 cm with an average error of 1.7 cm (1.36%)
e. At a distance of 150 cm with an average error of 3.76 cm (2.51%)
f. At a distance of 175 cm with an average error of 1.24 cm (0.71%)
g. At a distance of 200 cm with an average error of 1.28 cm (0.64%)

The average error result for the whole experiment was 2.36 cm.

4. CONCLUSIONS

Based on the results of the data analysis done, the following conclusions are drawn:

a. The unmanned autonomous prototype PID motion control system with a straight trajectory was successfully made using the parameter values $K_p = 1191.6; K_i = 340.69; K_d = 375.54$, with a path error of 0.32%.
b. The design of PID motion control system was successfully developed for an unmanned autonomous forklift prototype with a straight trajectory which varies from 50 cm to 200 cm with 25 cm intervals, using parameter values $K_p = 0.97; K_i = 0.3; K_d = 1$, with an average absolute error of 2.36 cm.

REFERENCES

[1] S. Chopra and P. Meindl, *Supply Chain Management: Strategy, Planning, and Operation*. Pearson, 2013.
[2] T. Bock and T. Linner, *Site Automation: Automated/Robotic On-Site Factories*, 1st edition. Cambridge University Press, 2016.

[3] G. Swartz, *Forklift Safety: A Practical Guide to Preventing Powered Industrial Truck Incidents and Injuries*, Second edition. Government Institutes, 1999.
[4] R. Jefferies, *Forklift Operator Training*. CreateSpace, 2012.
[5] Washington State Departement of Labor & Industries, *Forklift Safety Guide*. 2015.
[6] U. Elangovan, *Smart Automation to Smart Manufacturing: Industrial Internet of Things*. Momentum Press, 2019.
[7] H.-H. Lee, "Modeling and Trajectory Control of a Forklift-Like Wheeled Robot," in *Volume 4A: Dynamics, Vibration, and Control*, American Society of Mechanical Engineers, Nov. 2014. doi: 10.1115/IMECE2014-37081.
[8] H. A. Van Essen and H. Nijmeijer, "Non-linear model predictive control for constrained mobile robots," in *2001 European Control Conference (ECC)*, IEEE, Sep. 2001, pp. 1157–1162. doi: 10.23919/ECC.2001.7076072.
[9] M. Abdellatif, M. Shoeir, O. Talaat, M. Gabalah, M. Elbably, and S. Saleh, "Design of an Autonomous Forklift Using Kinect," *MATEC Web of Conferences*, vol. 153, p. 04005, Feb. 2018, doi: 10.1051/matecconf/201815304005.
[10] T. A. Tamba, B. Hong, and K.-S. Hong, "A path following control of an unmanned autonomous forklift," *Int J Control Autom Syst*, vol. 7, no. 1, pp. 113–122, Feb. 2009, doi: 10.1007/s12555-009-0114-y.
[11] T. Herlambang, F. A. Susanto, D. Adzkiya, A. Suryowinoto, and K. Oktafianto, "Design of Navigation and Guidance Control System of Mobile Robot with Position Estimation Using Ensemble Kalman Filter (EnKF) and Square Root Ensemble Kalman Filter (SR-EnKF)," *Nonlinear Dynamics and Systems Theory*, vol. 22, no. 4, pp. 390–399, 2022.
[12] Teguh Herlambang, Reizano Amri, and Sri Hartatik, "Estimasi Posisi Mobile Robot Menggunakan Metode Akar Kuadrat Unscented Kalman Filter (AK-UKF)," *Technology Science and Engineering Journal*, vol. 1, no. 2, 2017.
[13] T. Herlambang, Subchan, and H. Nurhadi, "Design of motion control using proportional integral derivative for UNUSAITS AUV," *International Review of Mechanical Engineering*, vol. 12, no. 11, pp. 928–938, 2018, doi: 10.15866/ireme.v12i11.15758.
[14] T. Herlambang, S. Subchan, H. Nurhadi, and D. Adzkiya, "Motion Control Design of UNUSAITS AUV Using Sliding PID," *Nonlinear Dynamics and Systems Theory*, vol. 20, no. 1, pp. 51–60, 2020.
[15] T. Herlambang and H. Nurhadi, "Design of a Sliding PID Controller for The Surge and Roll Motion Control of UNUSAITS AUV," *International Journal of Computing Science and Applied Mathematics*, vol. 3, no. 2, p. 61, 2017.
[16] F. Yudianto, T. Herlambang, F. A. Susanto, A. Suryowinoto, and B. P. Tomasouw, "Design of ROV Straight Motion Control Using Proportional Sliding Mode Control Method," *BAREKENG: Jurnal Ilmu Matematika dan Terapan*, vol. 16, no. 3, pp. 1051–1058, Sep. 2022.