Jurnal Polimesin
Department of Mechanical Engineering
State Polytechnic of Lhokseumawe
http://e-jurnal.pnl.ac.id/polimesin
e-ISSN : 2549-1999 No. : 1 Month : February
p-ISSN : 1693-5462 Volume : 23 Year : 2025

Processing dates: received on 2024-07-11, reviewed on 2024-12-02, accepted on 2024-12-18 and online availability on 2025-02-28

Implementation of a PID Controller in a DC Motor System: a ase study on a floor cleaning robot

Rizki Aulia Nanda[1,*], Karyadi[1], Ade Suhara[2], Tukino[3], Muhamad Guntur[1]
[1]Department of Mechanical Engineering, University of Buana Perjuangan Karawang, Karawang 41360, Indonesia
[2]Department of Industrial Engineering, University of Buana Perjuangan Karawang, Karawang, 41360, Indonesia
[3]Department of Information System, University of Buana Perjuangan Karawang, Karawang, 41360, Indonesia
*Corresponding author: rizki.aulianda@ubpkarawang.ac.id

Abstract

A floor-cleaning robot requires a stable and consistent movement to ensure effective cleaning. This study implements a Proportional-Integral Derivative (PID) control system using Arduino™ Uno to regulate the DC motor's speed and rotation. A well-tuned PID system ensures smooth and constant motion, optimizing the cleaning process. Without PID control, the robot's movement can become inconsistent, reducing its cleaning efficiency. The research focuses on designing and implementing a PID system to achieve constant speed and rotation through programming. The selected PID parameters are K_p = 0.2, K_i = 50, and K_d = 0, effectively making it a Proportional-Integral (PI) controller. Experiments were conducted under two conditions: (1) unloaded motor testing—where the motor's rotational stability was assessed without external load, and (2) loaded robot operation—where the robot was tested while cleaning a dirty floor. Results showed that the DC motor's unloaded speed was initially 5 RPM, stabilizing at 10.62 RPM. Under load, the robot's speed started at 0.2 RPM, peaked at 6.85 RPM, and maintained an average velocity of 4.85 cm/s. The robot demonstrated consistent motion in forward, left-turn, and right-turn maneuvers, achieving 13-degree rotations in 30 seconds. The robot was able to operate for 30 minutes, maintaining a stable speed between the 3rd and 22nd minute, before declining as battery power depleted. The findings confirm that the implementation of a PI controller effectively stabilizes motor rotation and enhances floor-cleaning efficiency. Additionally, this control method can be adapted to larger-scale robots and various DC/AC motor systems.

Keywords:
PID System,DC motor, mobile robot, Arduino UNO, tachometer.

1 Introduction

A Proportional Integral Derivative (PID) controller uses the system's feedback characteristics to calculate an instrumentation system's precision. In the industrial world, PID controllers are common conventional controllers. Depending on the amount of error found, the PID controller will instruct the control valve. An actuator that controls fluid flow in industrial processes is what the control valve does. The set point is the desired water level. The discrepancy between the actual water level and the set point is known as error [1][2]. The problem of this research is that when developing a robot car in regulating speed, the robot speed is often not constant due to Pulse Width Modulation (PWM) and the incoming electric current is not balanced, this research gap arises due to the robot car motion settings that do not use PID, so when the robot car moves it will produce a very fast motion and then it will decrease and there is no constant in rotation, therefore the purpose of this study is to implement a PID system so that the robot car can move according to low PWM until it reaches a critical point. An earlier study also covered the use of PID to control a DC motor in a robot. To enable the wheels to determine the robot's position automatically, mathematical calculations that serve as a reference for wheel rotation are required. The position of the PID value can be entered into a programming value using this fundamental math [3]. For example, in the study conducted by Vicky Mudeng and colleagues, who talked about how the PID control system uses mathematics to achieve balance in robot speed and movement position [4]. The mathematical values for a PID system that can regulate a DC motor are explained by Eq. 1.

$$u(t) = K_p e(t) + K_i \int_0^t e(t)dt + K_d \frac{de(t)}{dt} \quad (1)$$

The PID eq. is used to determine the DC motor control signal. Eq. (1) states that y(t) is the actual angle, u(t) is the PWM signal for the DC motor, and e(t) is the position error value difference between the set angle and output measured angle (actual angle). K_p, K_i, and K_d represented the proportional, integral, and differential coefficient values, respectively. PWM must be regulated in order to control a DC motor because the microcontroller, which regulates a DC motor's rotation, becomes the primary system for all output processes [5][6][7]. The technique used in electronic circuits to change direct current into alternating current is called pulse width modulation, or PWM. Additionally, by using this technique, alternating current is converted from a square wave to a sine wave [8]. According to Chandra Shekhar Gohiya's research, a DC motor's output circuit communicates between the voltage on the motor coil and a PWM signal that controls the motor. The microcontroller will then receive the PWM signal and use the computed PID system to control it. Fig. 1 shows the PID circuit system on the motor [9].

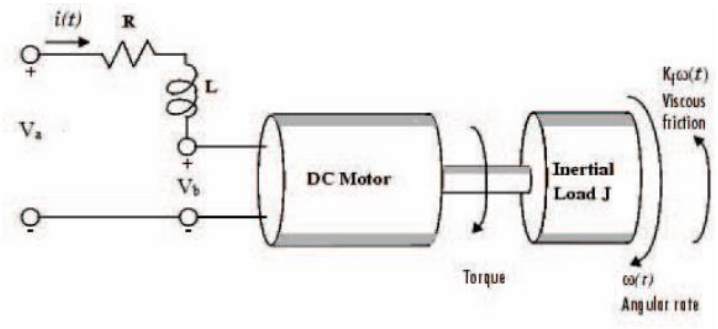


Fig. 1. Motor DC PWM Control [8]

The PWM is first set during the microcontroller programming process, and it serves as the primary logic gate for the DC motor's input of an electric current voltage. This is how PID settings work [10][11]. According to research by Ricard and colleagues, a robot's PID can be optimized to neutralize a DC motor's rotation, resulting in a harmonious and balanced combination of torque, speed, and rotation. To determine the impact of the PID entered on the microcontroller programming logic gate, measurements were made [12]. Therefore, a diagram illustrating the PID process's flow from input to output must be created [13]. Yatmano and friends' research also explains the importance of setting the robot car motion parameters in the floor cleaning process, which aims to provide stable motion at each coordinate point; each added PWM results in an increase in rotation, but the increase in rotation becomes constant if the robot works for the first 1 minute [14]. The measuring parameters are then used as the starting point for data retrieval. in Fig. 2 for the PID diagram.

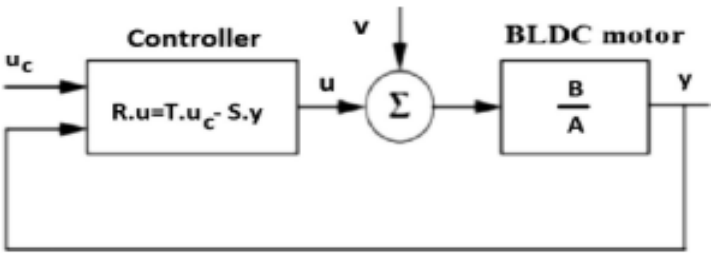


Fig. 2. PID Diagram[15]

A microcontroller system, a motor driver to regulate a DC motor's rotational speed and CW and CCW directions, and the actual DC motor device are the key components that control a PID system. The microcontroller will be connected to the DC motor through the assembly of all these parts [16][17][18]. According to research by Parikh and colleagues, the PID system can keep the motor rotating continuously while maintaining constant power, voltage, and current consumption. Nonetheless, this study illustrates how PID and the fuzzy approach can be paired to provide a motor's rotation system with constant rotation. Use an encoder or an RPM reading measuring device to get rotation data. The Matlab program is used for the mathematics of PID input. Thus, in this study, two types of graphs—fuzzy graphs and PID graphs—can be used to read a motor's rotation [19]. There are several differences from previous studies as well as some similarities from the selected literature studies that will be applied. The components used in this study are the same: DC motor, motor drive, and Arduino Uno microcontroller. The methodology used for data collection—using a tachometer to determine whether the DC motor is loaded or not—distinguishes this study from its updates. The problem in a DC motor greatly affects the rotation; non-constant rotation often makes the DC motor easily damaged, and energy consumption becomes more wasteful. Therefore, this study will discuss the stability of the DC motor by applying PID but does not discuss the energy used after and before the PID is used. To prevent waste from entering the robot, the next update will look at how the floor-cleaning robot car can maintain a stable speed. Several sources, each with varying data, are cited in this research method; therefore, each will be discussed in detail in the research methodology subsection. In this study, a PID system is required to stabilize the DC motor rotation so that the robot can move smoothly and precisely. However, PID implementation can be utilized in other systems with the intended output value.

2 Materials and methods

Before delving into the research methodology, please take note of Fig. 3, which represents the form and configuration of the robot employed in this study. From the robot's dimensions, it can be seen that it has a length of 682 mm, a width of 304 mm, and a height of 156 mm. The robot has several parts, such as a body, frame, wheels, a floor-cleaning broom, and electrical parts.

Fig. 4 shows the robot results that correspond to the design. Next, you can see the parts that will be utilized to move a floor-cleaning robot vehicle.

This robot is divided into multiple sections, including an aluminium frame, a waste storage area, a floor cleaning area, and a microcontroller system section. The dimensions of this robot are 24 cm in height, 35 cm in width, and 60 cm in length.

The mother of all system controls, the Arduino UNO microcontroller, has 11 digital pins, 5 analog pins, 1 pair of RX and TX pins, and a 5 V DC pin. This microcontroller is used to operate the floor-cleaning robot car [20][21][22]. Next up is the LN298N brand motor driver, which has four pins total and two channels: two CW channels (A and B), two CCW channels (A and B), and two PWM channels (A and B). The third component powers the robot wheels with a DC motor that has a torque of 20 kg/cm and a current of 12 V DC [23]. Next, a tachometer is a device that measures a wheel's rotational mass (RPM) on a robot car. Table 1's final part is the battery, which has a capacity of 6800 mAh and a voltage of 12.56 V DC 20A. It powers every robot.Table 1 lists the parts that make up the floor-cleaning robot car.

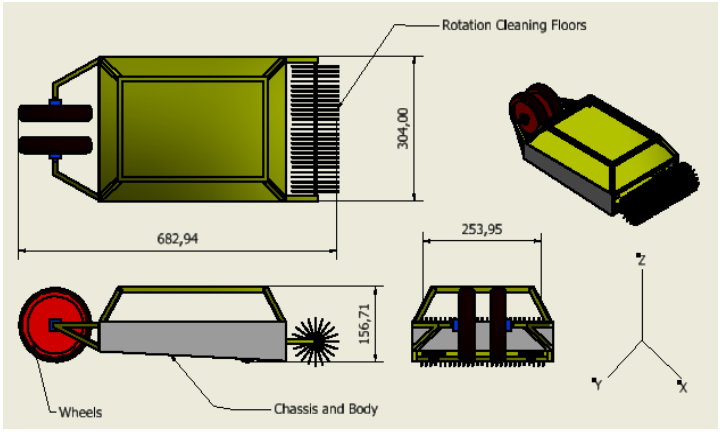


Fig. 3. Design of robot

Fig. 4. Robot floor cleaning

Table 1. Component of robot

No	Name	Specification
1	Arduino Uno	Arduino has a function to control all electronic systems according to the control in the programming that has been set, including the PID system.
2	Motor Driver	The Motor Driver has a function to control the rotation of the DC motor on the wheels and the broom with an electric voltage of 12 VDC.
3	Motor DC	DC motor to drive the wheels with speed, rpm and torque
4	Bluetooth	Robot systems can be controlled or their movements can be monitored using Bluetooth's telecommunications capabilities.
5	Battery	Stores and distributes electrical energy to all robotic components with a capacity of 12 VDC and a performance current of 5000 mAh.

This part can move the robot automatically while mopping the floor. Fig. 5 shows how the research is conducted.

Assembling the electronic system from all of the parts listed in Table 1 is the first step in the research flow shown in Fig. 5. Fig. 6 shows the schematic of every circuit.

The input and output systems are shown in Fig. 6. The output system has two motor drivers, and three DC motors, while the left and right wheels are moved by two DC motors on one motor driver and the floor cleaning broom system is moved by one motor driver and one DC motor. The input system, on the other hand, has an Arduino Uno that controls all systems and has input PID data that has been set in the C language programming function. The Arduino, motor driver, and DC motor are all connected to the battery; Table 1 lists the system names based on the numbers.

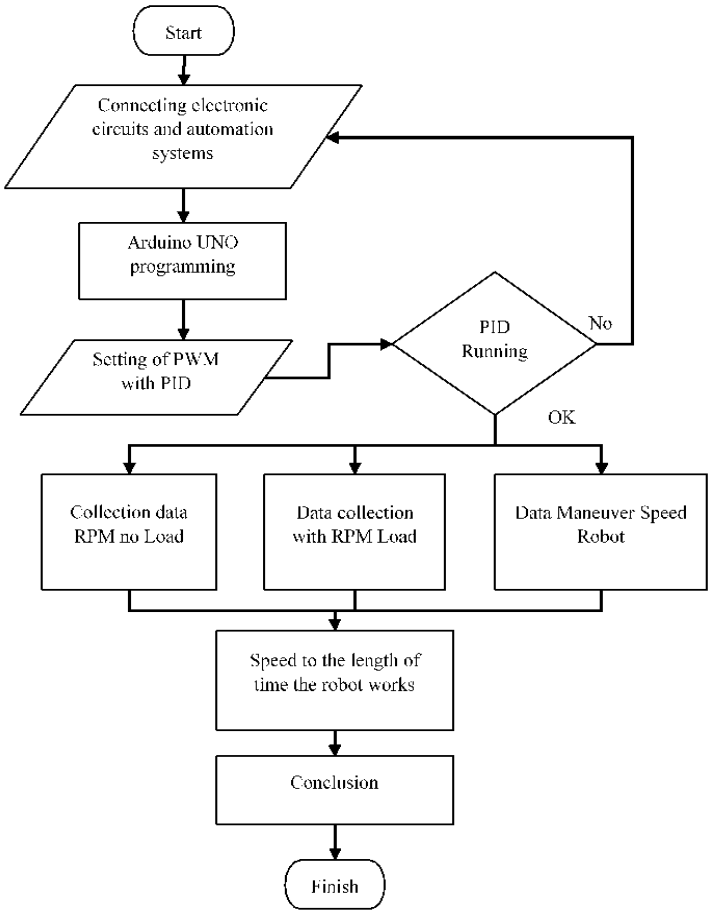


Fig. 5. Flow chart of research

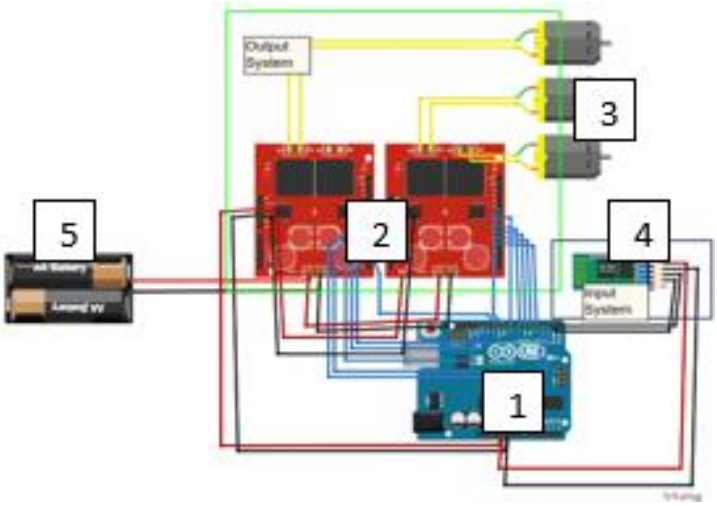


Fig. 6. Schematic circuit

2.1 Input PID to Arduino UNO

The system PID is entered into the Arduino IDE's programming since, in essence, speed control uses C programming to regulate the DC motor's rotational speed via a PWM signal [24][25]. The PID diagram can be seen in Fig. 7, which is the PID input design for Arduino UNO programming.

As a reference for the error difference that occurs in the first measurement before inputting the PID in Arduino programming, K_p data input begins with a value of 0.2. The set PWM value, Ki, is then 50. The input value will be programmed into the Arduino programming as a "float" function since K_d is **a** 0 derivative. Refer to Fig. 8 for information on the K_p, K_i, and K_d inputs when using the "float" function [26]. The article "Autonomous Self-Reconfigurable Floor Cleaning Robot" discusses the use of the PID value in relation to the mathematical computation of speed. Nonetheless, the value is derived from the PID derivative's mathematical computation, which adapts to the newly created floor cleaning robot. Only PI or Kp and Ki can use the value of 0 acquired from K_d because it does not give a function. Nevertheless, the Arduino system programming still uses the value of 0 obtained for the calculation process and motion stability of the floor-cleaning robot car. The article "Brushless DC motor tracking control using self-tuning fuzzy PID control and model reference adaptive control [15]" is cited in the mathematical derivation using the PID eq., which yields the values K_p = 0.2, Ki = 50, and K_d = 0 for this investigation. Thus, it merely becomes a PI function. In Fig. 8, the value is displayed. The reason the K_d value in the PID system can be set to 0 is so that the system only uses PI (Proportional-Integral) control. This is usually done if the derivative component is not needed, for example, in a system that does not require a fast response or if the noise in the measurement signal is too large because K_d tends to magnify the effect of noise.

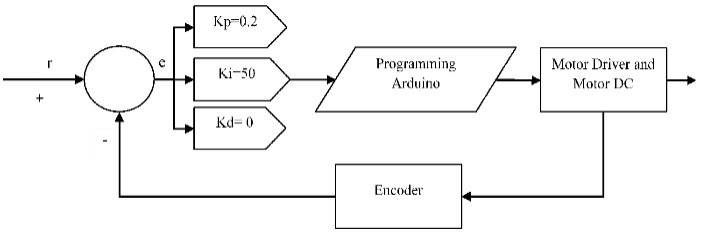


Fig. 7. Circuit PID System

```
float Kp          = 0.2;
float Ki          = 50;
float Kd          = 0;
```

Fig. 8. Input PID

Since the fundamental formula for calculating PID is K_p+K_i, an error parameter is required to determine the value that should be set as a reference for the floor cleaning robot to move steadily. Take note of Fig. 9, which illustrates the PID calculations for a stable PWM on a robot car.

```
//PID sinyal posisi
 error   = targetPos - PWMPosRead;
 integral += error * dt;
 derivative = (error - prevError)/dt;
 control = (Kp*error) + (Ki*integral) + (Kd*derivative);
```

Fig. 9. PID calculation in programming

To determine the difference value that occurs in the PWM for the output, the first value that must be considered is the difference value between the last measurement value and the value that has been set. Next, compute the derivative, determine planning, calculate the integral error over time (dt = 0.006 s), and then make control decisions. The error, integral, and derivative will be multiplied by all K_p, K_i, and K_d values. The PI system is run to observe the output that results from programming it to determine its parameters. Rechecking the circuit system's connectivity is essential if the operating system malfunctions, as a malfunctioning cable may be the cause [27]. The measurement procedure is completed if it is successful. The data collection subchapter will explain the various steps that makeup data collection.

2.2 Data extraction

The robot wheel's last rotation without a load was measured to start the first data collection process. This robot used to have a 30-minute workday. Therefore, for 30 minutes, data collection without load is conducted by hanging the robot wheel's components and leaving the wheel running. After that, the consistency of a PWM that has been set using PID is examined. From Fig. 10, you can see the final round data collection process without load. The goal of the final rotation without load test is to confirm that the PID system has correctly entered the Arduino programming by hanging the wheel

and measuring the rotation to observe the constant value and the level of RPM that occurs. The goal of the final rotation without a load test is to confirm that the PID system has correctly entered the Arduino programming by hanging the wheel and measuring the rotation to observe the constant value and the level of RPM that occurs.

Fig. 10. No-load rotational data acquisition

Next, record the robot wheel's rotational data after it has been loaded. The robot was first measured for 30 minutes while it moved forward, 30 minutes while it turned left, and 30 minutes while it turned right. As a result, the test can be used to determine the degree of left and right turning because the robot wheels' settings have been established. Fig. 11 explains how to take a rotation when a load is applied. Testing with a load involves placing the robot on the floor and testing its DC motor to see if it rotates steadily at a set speed and can move in the direction and under the control of the robot.

Fig. 11. Measurement with load

The robot is positioned and left to operate when measuring while carrying a load, but the data collection procedure is still completed. Procedure for gathering data that calculates speed by measuring travel time and **distance [28].** Nevertheless, the time and distance traveled also have an impact on how quickly the DC motor will rotate. The robot car has a working duration of thirty minutes, and its data is summarized in one **second [29]**[30], yielding an average RPM value and speed in units per minute based on thirty data points. The results and conclusions chapter will cover every piece of data that has been gathered, providing a detailed explanation of each piece of data about the research method and flow.

3 Results and Discussion.

The rotation of the DC motor without a load was the first piece of data to be measured, as the research method and flow explain. Determine which portions of the data have been taken first. The position of the wheel rotation direction for which data was collected is displayed in Table 2.

Table 2. Combination of rotation Motor DC

Measurement	A		B		Time(s)
	CW	CCW	CW	CCW	
M1	1	0	1	0	30
M2	0	1	1	0	30
M3	0	1	0	1	30

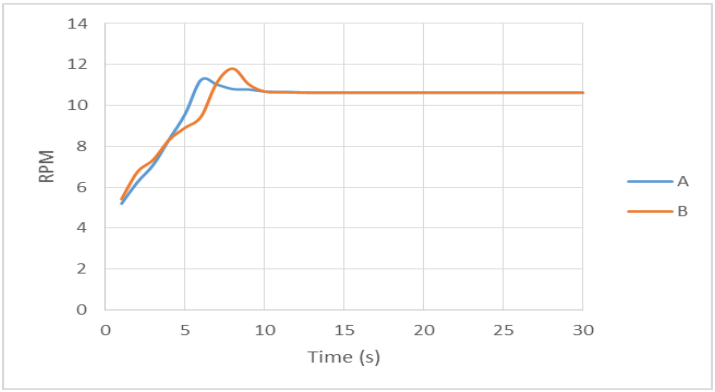
Table 2 explains that a single-motor driver can be used with two channels or a single motor. Consequently, the left DC motor is driven by channel A, and the right DC motor is driven by channel B. Three measurements—M1, M2, and M3—were made to determine whether the DC motor rotates in a CW or CCW direction. In Table 2, indicator "1" indicates the DC motor's on position, while indicator "0" indicates the motor's off position. Table 3 displays the measurement results.

Without a load, the DC motor was measured, and Table 3 displays the findings. Channel A CW or CCW and channel B CW or CCW measurements M1, M2, and M3 display data collected for 30 seconds. The data thus decreases in the first second, increases reaches its peak in the seventh second, then decreases once more, and stays constant at RPM 10.62, as can be observed from the table. The duration of a continuous rotation is 13 to 30 seconds. A graph based on the measurement results is shown in Fig. 12, with an OUTPUT rotation range of 5 RPM–10 RPM–11 RPM.

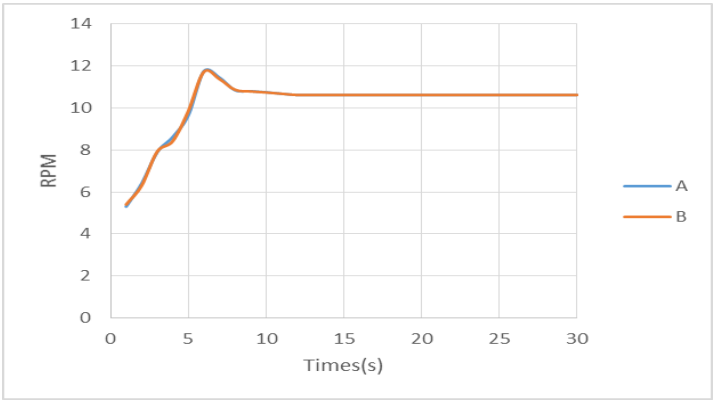
Thus, it can be concluded that the DC motor rotation stays in a constant state by the programming input on the PID. The PID, which has been set and has a constant rotation constant, provides a reference that the rotation will be higher in the first second but will be constant in the following seconds. Table 3's zero value indicates that because the CW section is turned on when the rotation obtains data, the CCW rotation does not obtain this data.

Table 3. Results of measuring DC motor rotation without load

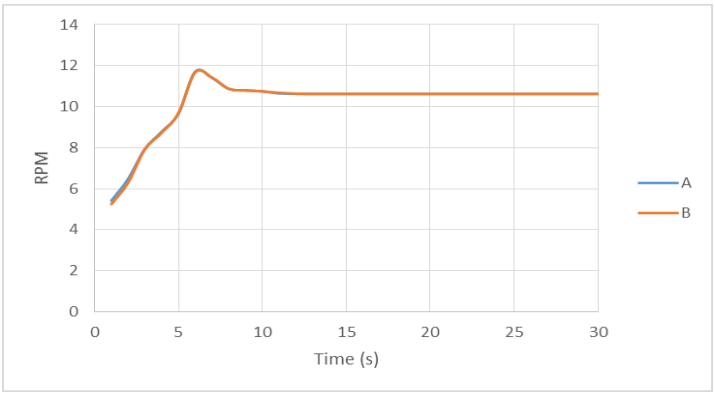
Times(s)	M1				M2				M3			
	A		B		A		B		A		B	
	CW	CCW	CW	CCW	CW	CCW	CW	CCW	CW	CCW	CW	CCW
1	5,2	0	5,42	0	0	5,3	5,4	0	0	5,25	0	5,42
2	6,25	0	6,75	0	0	6,4	6,3	0	0	6,3	0	6,48
3	7,1	0	7,35	0	0	7,89	7,92	0	0	7,92	0	7,94
4	8,35	0	8,325	0	0	8,62	8,42	0	0	8,73	0	8,78
5	9,58	0	8,91	0	0	9,65	9,85	0	0	9,68	0	9,67
6	11,25	0	9,45	0	0	11,75	11,72	0	0	11,68	0	11,71
7	11,02	0	11,12	0	0	11,45	11,38	0	0	11,42	0	11,41
8	10,8	0	11,8	0	0	10,85	10,87	0	0	10,88	0	10,87
9	10,78	0	11,03	0	0	10,8	10,79	0	0	10,788	0	10,8
10	10,68	0	10,68	0	0	10,75	10,74	0	0	10,75	0	10,75
11	10,66	0	10,65	0	0	10,68	10,68	0	0	10,68	0	10,65
12	10,65	0	10,63	0	0	10,62	10,62	0	0	10,64	0	10,63
13-30	10,62	0	10,62	0	0	10,62	10,62	0	0	10,62	0	10,62



(a)



(b)



(c)

Fig. 12. No-load rotation measurement graphs (a) M1 (b) M2 and (c) M3

Fig. 12 in (a) shows that the measurement at point M1 indicates that the data is not arbitrary or consistent, but rather rotates steadily from the 10th to the 30th second. The highest RPM is found at the point where the data is closest to the 10th, equal to 11.25 RPM with CW rotation, and channel B has a peak rotation of 11.12 RPM and so forth. For both channel A and channel B with CW rotational motion, the rotational constant is RPM 10.62.

Table 4. Combination of rotation Motor DC on Load

Measurement	A		B		Time(s)	Position
	CW	CCW	CW	CCW		
M1	1	0	1	0	30	Forward
M2	0	1	1	0	30	Turn Left
M3	1	0	0	1	30	Turn Right

The robot will be put on the floor and tested for constant rotation using the load to be analyzed once the DC motor's constant rotation is known without a load, and it is assumed that the rotation is constant. Table 4 illustrates the movement and measurement process; it is nearly identical to Table 2, where measurements were also taken of M1, M2, and M3. However, since the robot was moving during the test, M2 and M3's positions were turning left and right, respectively, showing that when the robot turns, the rotation that has been input into the PID system produces a degree.

The explanation for the robot car's position in Table 4 goes as follows: **P**osition M1 displays the wheels' location between channels A and B. The robot will advance in CW condition, and when it reaches position M2, it will turn left as channel A displays CCW and channel B displays CW. The robot will turn to the right because M3 measurements indicate that channel A is pointing in the CW direction and channel B is pointing in the CCW direction. In order to determine whether PID input has been provided in the Arduino Uno programming, measurements are made to see a graph that rotates continuously. Take note that the measurement results when the robot is given a load are displayed in Table 5.

Table 5 shows that on both Channel A and Channel B, the DC motor's RPM position begins at the first second, with a value of 0.25 RPM and 0.15 RPM CW rotation. The rotation for the first measurement (M1) is the rotation that moves the robot forward. The robot rotates steadily at 6.85 RPM for 30 seconds as the RPM rises, reaching its maximum at the 11th position. An initial rotation that remains balanced results from the discrepancy between channel A and channel B in the CW rotation.

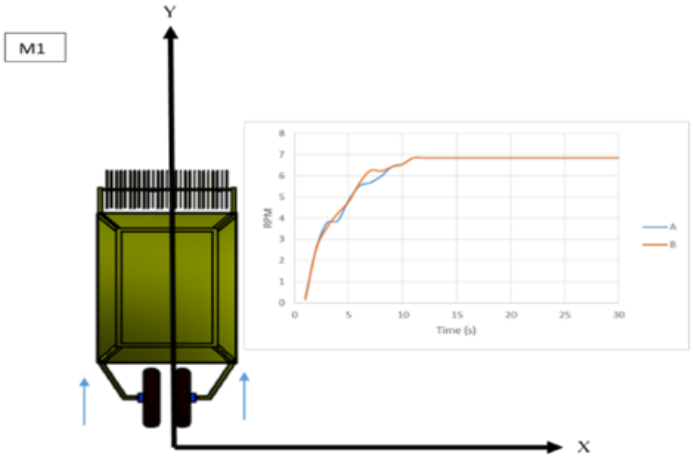
The second measurement (M2) explains why the CCW rotation on channel A displays an RPM of 0.24 in the first second. There is a discrepancy in the increase at constant RPM; on channel A, the constant RPM is at 11 seconds, but on channel B, it is at 12 seconds.

Table 5. Results of measuring DC motor rotation with load

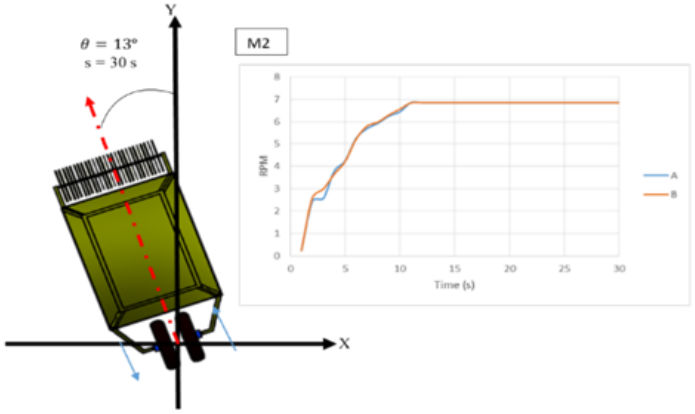
Times (s)	M1				M2				M3			
	A		B		A		B		A		B	
	CW	CCW	CW	CCW	CW	CCW	CW	CCW	CW	CCW	CW	CCW
1	0,25	0	0,15	0	0	0,24	0,22	0	0,26	0	0	0,28
2	2,56	0	2,55	0	0	2,44	2,58	0	2,52	0	0	2,55
3	3,78	0	3,56	0	0	2,56	2,98	0	2,88	0	0	2,68
4	3,88	0	4,23	0	0	3,78	3,65	0	3,78	0	0	3,89
5	4,85	0	4,78	0	0	4,25	4,22	0	4,55	0	0	4,62
6	5,52	0	5,65	0	0	5,25	5,22	0	5,65	0	0	5,82
7	5,68	0	6,25	0	0	5,69	5,8	0	6	0	0	6,28
8	5,98	0	6,23	0	0	5,93	5,98	0	6,25	0	0	6,3
9	6,43	0	6,42	0	0	6,25	6,3	0	6,32	0	0	6,48
10	6,56	0	6,55	0	0	6,44	6,56	0	6,55	0	0	6,48
11-30	6,85	0	6,85	0	0	6,85	6,84	0	6,85	0	0	6,78

Channel B displays a CCW rotation in the first second of 0.22. Nevertheless, the robot's speed of movement is essentially unaffected by this. The robot will move and turn to the left, as indicated by its CW and CCW rotations. On channels A and B, the RPM drops by 1.56% from the first to the tenth second during turning. The robot's axis of motion does not travel linearly, which causes the drop. However, the robot's mobility is not greatly affected by the 1.56% difference because the robot can move steadily from the eleventh to the thirty-first seconds.

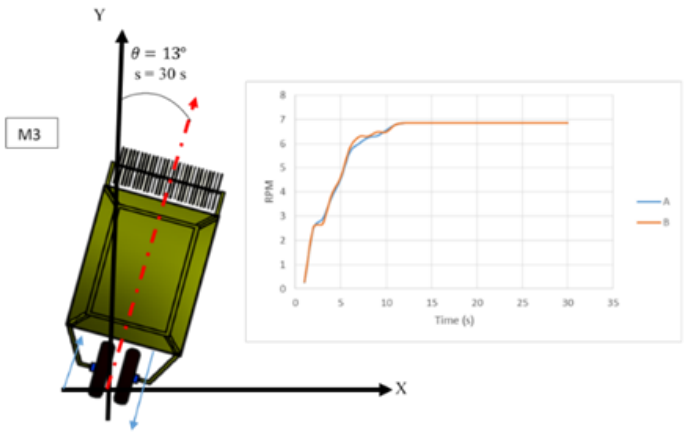
The third measurement (M3) reveals that the CW rotation in channel A is 0.26 RPM in the first second, while the CCW rotation in channel B produces an initial rotation of 0.28 RPM. The constant rotation in channel A is found at 12 seconds, while the constant rotation in channel B is found at 12 seconds, with a constant value of 6.85 RPM. so that 6.85 RPM is used for the M1, M2, and M3 constant rotation tests. Fig. 13 displays the robot's position and graph. This measurement indicates that the robot is moving in the direction of a right turn. Channel A and Channel B saw a decrease of 3.38% in the M3 test, which was the same as in the M2 test and resulted in nonlinear wheel rotation. Despite this, the robot's stability can still be maintained because the decrease happened between seconds 1 and 10 and finally stabilized between seconds 11 and 30, just like in the M2 test.



(a)



(b)



(c)

Fig. 13. Robot Position and Motor DC Rotation Graph (a) M1 (b) M2 (c) M3

Due to the CW rotation used by Channels A and B, Figs 13a, b, and c show a measurement (M1) of DC motor rotation with a load-producing straight motion. The robot rotates steadily at 6.85 RPM with balanced movement over the first second till the peak point. Following the first measurement (M1), which demonstrates a constant final rotation for optimal operation, the second measurement (M2) measures the balanced DC motor rotation that occurs when the robot moves to the left for 30 seconds, resulting in the same peak rotation of 6.85 RPM on channels A and B. Nevertheless, it generates a 13-degree turning angle in 30 seconds. The identical test was performed in the third measurement (M3), wherein Channel A and Channel B received CW and CCW rotations, respectively, resulting in a peak rotation of 6.85 RPM, which caused the robot to turn to the right. This test is run once while the robot's battery lasts until it runs out; during the test, the robot can clean the floor for more than an hour, but the data is only collected at the critical point of the robot's movement, as previously explained; at this point, moving forward and turning the robot produces different effects, which can also affect the application of PID; however, this study does not discuss the impact of power due to the application of PID in the robot's movement process during the floor cleaning. Thus, it has been effectively proved that the PID can be determined based on the movement of the robot. All that is needed to measure the robot's position degree is to draw an arc perpendicular to the robot. Examine Fig. 14, which shows a speed graph of the robot tested for thirty minutes till the robot car's battery ran out.

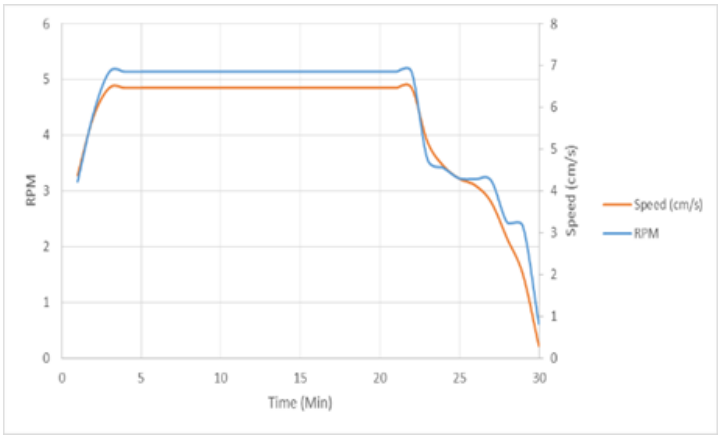


Fig. 14. Robot rotation and speed graphs

Fig. 14 illustrates how the robot's rotation and speed decrease as the battery energy decreases. The floor cleaning robot has a working endurance of 30 minutes; the critical point occurs in the third to 22nd minute; the robot maintains a constant speed of 4.85 cm/s at 6.85 RPM. The robot rotates at 4.22–5.85 RPM and moves at a speed of 3.28–4.33 cm/s in the first and second minutes. Battery power begins to decrease from the 23rd to the 30th minute, resulting in a speed variation of 3.88 cm/s to 0.22 cm/s at a rotation speed of 4.75-0.82 RPM. The rotation and speed of the floor cleaning robot will remain constant if the PID input in the Arduino IDE programming is functioning properly. The test shows that the PID system settings implemented in the PI system, it has an impact on the floor cleaning process, which is good and does not leave any dirt residue.

This study makes reference to the work of Ricard Bitriá and Jordi Palacín [12], whose findings clarify how RPM and PWM affect the way PID is implemented in microcontroller systems. There are variations in data collection, though, and this study also addresses the impact of power when the PID keeps the robot's DC motor rotating continuously. According to research by Ricard and colleagues, the robot car's rotation reaches its ideal state when the RPM increases from the first to the peak second and then stays constant in the following second. The RPM graph gradually

indicates that the speed (RPM) is diminishing as the battery power begins to drain. Thus, this study demonstrates the effectiveness of PID in maximizing DC motor rotation.

4 Conclusions.

This study demonstrates the implementation of a PID (PI) controller in regulating the speed and motion of a floor-cleaning robot. The control process was evaluated through three measurement methods (M1, M2, and M3), confirming stable and predictable motor performance under load conditions. Key findings include:

1. Stable Speed Control: Without a load, the DC motor maintained a stable rotation of 10.62 RPM. Under load, it reached a peak speed of 6.85 RPM, stabilizing between 3 to 22 minutes before decreasing due to battery depletion.
2. Directional Control: The robot maintained a 13-degree turn per 30 seconds in both left-turn (CCW) and right-turn (CW) conditions, demonstrating effective maneuverability.
3. Operational Efficiency: The robot functioned efficiently for 30 minutes, with motion stability contributing to effective floor cleaning without leaving residual debris.

Although the PID implementation ensures a steady and predictable cleaning motion, further research is recommended to evaluate its impact on energy consumption and battery efficiency. Future studies could optimize PID tuning to extend operational time and improve power utilization for enhanced robotic performance

References.

[1] V. M. Hernández-Guzmán and J. Orrante-Sakanassi, "PID control of robot manipulators actuated by BLDC motors," Int. J. Control, vol. 94, no. 2, pp. 267–276, 2021, doi: 10.1080/00207179.2019.1590648.

[2] W. P. Aung, "Analysis on Modeling and Simulink of DC Motor and its Driving System Used for Wheeled Mobile Robot," World Acad. Sci. Eng. …, vol. 88704, no. December, pp. 299–306, 2007, [Online]. Available: http://urrg.eng.usm.my/eee351/PB4.pdf

[3] M. M. Maung, M. M. Latt, and C. M. Nwe, "DC Motor Angular Position Control using PID Controller with Friction Compensation," Int. J. Sci. Res. Publ., vol. 8, no. 11, pp. 149–155, 2018, doi: 10.29322/ijsrp.8.11.2018.p8321.

[4] V. Mudeng, B. Hassanah, Y. T. K. Priyanto, and O. Saputra, "Design and Simulation of Two-Wheeled Balancing Mobile Robot with PID Controller," Int. J. Sustain. Transp. Technol., vol. 3, no. 1, pp. 12–19, 2020, doi: 10.31427/ijstt.2020.3.1.3.

[5] T. Allam, M. Raju, and S. S. Kumar, "Design of PID controller for DC Motor Speed Control Using Arduino Microcontroller," Int. Res. J. Eng. Technol., pp. 791–794, 2016, [Online]. Available: www.irjet.net

[6] H. Feng, W. Ma, C. Yin, and D. Cao, "Trajectory control of electro-hydraulic position servo system using improved PSO-PID controller," Autom. Constr., vol. 127, no. December 2020, p. 103722, 2021, doi: 10.1016/j.autcon.2021.103722.

[7] S. Pookkuttath, M. R. Elara, M. Mohan Rayguru, Z. S. Saldi, V. Sivanantham, and B. Ramalingam, "Snail: An Eco-Friendly Autonomous Steam Mopping Robot for Cleaning and Disinfection of Floors," Mathematics, vol. 11, no. 5, pp. 1–17, 2023, doi: 10.3390/math11051086.

[8] C. S. Gohiya, S. S. Sadistap, S. A. Akbar, and B. A. Botre, "Design and development of digital PID controller for DC motor drive system using embedded platform for mobile robot," Proc. 2013 3rd IEEE Int. Adv. Comput. Conf. IACC 2013, pp. 52–55, 2013, doi: 10.1109/IAdCC.2013.6514193.

[9] A. Joon and W. Kowalczyk, "Design of autonomous mobile robot for cleaning in the environment with obstacles," Appl. Sci., vol. 11, no. 17, pp. 1–13, 2021, doi: 10.3390/app11178076.

[10] A. A. Mahfouz, A. A. Aly, and F. A. Salem, "Mechatronics Design of a Mobile Robot System," Int. J. Intell. Syst. Appl., vol. 5, no. 3, pp. 23–36, 2013, doi: 10.5815/ijisa.2013.03.03.

[11] H. Khan, S. Khatoon, and P. Gaur, "Comparison of various controller design for the speed control of DC motors used in two wheeled mobile robots," Int. J. Inf. Technol., vol. 13, no. 2, pp. 713–720, 2021, doi: 10.1007/s41870-020-00577-8.

[12] R. Bitriá and J. Palacín, "Optimal PID Control of a Brushed DC Motor with an Embedded Low-Cost Magnetic Quadrature Encoder for Improved Step Overshoot and Undershoot Responses in a Mobile Robot Application," Sensors, vol. 22, no. 20, 2022, doi: 10.3390/s22207817.

[13] B. N. Kommula and V. R. Kota, "Direct instantaneous torque control of Brushless DC motor using firefly Algorithm based fractional order PID controller," J. King Saud Univ. - Eng. Sci., vol. 32, no. 2, pp. 133–140, 2020, doi: 10.1016/j.jksues.2018.04.007.

[14] S. Yatmono, M. Khairudin, H. S. Pramono, and A. Asmara, "Development of Intelligent Floor Cleaning Robot," J. Phys. Conf. Ser., vol. 1413, no. 1, 2019, doi: 10.1088/1742-6596/1413/1/012014.

[15] A. A. El-samahy and M. A. Shamseldin, "Brushless DC motor tracking control using self-tuning fuzzy PID control and model reference adaptive control," Ain Shams Eng. J., vol. 9, no. 3, pp. 341–352, 2018, doi: 10.1016/j.asej.2016.02.004.

[16] D. Somwanshi, M. Bundele, G. Kumar, and G. Parashar, "Comparison of fuzzy-PID and PID controller for speed control of DC motor using LabVIEW," Procedia Comput. Sci., vol. 152, pp. 252–260, 2019, doi: 10.1016/j.procs.2019.05.019.

[17] S. Ekinci, B. Hekimoğlu, and D. Izci, "Opposition based Henry gas solubility optimization as a novel algorithm for PID control of DC motor," Eng. Sci. Technol. an Int. J., vol. 24, no. 2, pp. 331–342, 2021, doi: 10.1016/j.jestch.2020.08.011.

[18] T. S. Tamir et al., "Comparative Study of Four Speed Controllers of Brushless DC Motors for Industrial Applications," IFAC-PapersOnLine, vol. 53, no. 5, pp. 59–64, 2020, doi: 10.1016/j.ifacol.2021.04.124.

[19] P. Parikh, S. Sheth, R. Vasani, and J. K. Gohil, "Implementing Fuzzy Logic Controller and PID Controller to a DC Encoder Motor - 'a case of an Automated Guided Vehicle,'" Procedia Manuf., vol. 20, pp. 219–226, 2018, doi: 10.1016/j.promfg.2018.02.032.

[20] Arhami, A. N. Rizki, and K. Rudi, "Structural Analysis of Mobile Robot Frame for Spinach Water Seed Planting Using Finite Element Method," in International Conference on Experimental and Computational Mechanics in Engineering, 2021, pp. 177–186.

[21] R. A. Nanda, T. Supriyono, R. A. R. Ma'arof, and F. M. Dewadi, "Analisis Chassis Mobil Robot Penanaman Bibit Kangkung Menggunakan Metode Elemen Hingga," J. Tek. Mesin Mech. Xplore, vol. 2, no. 2, pp. 1–8, 2022.

[22] R. A. Nanda, A. Arhami, and R. Kurniawan, "Perancangan Dan Pengujian Model Mobil Robot Penanam Bibit Kangkung," Rona Tek. Pertan., vol. 13, no. 2, pp. 14–28, 2020, doi: 10.17969/rtp.v13i2.16982.

[23] R. Parween, L. T. L. Clarissa, M. Y. Naing, N. A. F. B. M. Fuad, and M. R. Elara, "Modeling and Analysis of the Cleaning System of a Reconfigurable Tiling Robot," IEEE Access, vol. 8, pp. 137770–137782, 2020, doi: 10.1109/ACCESS.2020.3009120.

[24] K. Vanchinathan and N. Selvaganesan, "Adaptive fractional order PID controller tuning for brushless DC motor using Artificial Bee Colony algorithm," Results Control Optim., vol. 4, no. February, p. 100032, 2021, doi: 10.1016/j.rico.2021.100032.

[25] M. M. Sabir and J. A. Khan, "Optimal Design of PID Controller for the Speed Control of DC Motor by Using

Metaheuristic Techniques," Adv. Artif. Neural Syst., vol. 2014, pp. 1–8, 2014, doi: 10.1155/2014/126317.
[26] R. Parween, M. Vega Heredia, M. M. Rayguru, R. Enjikalayil Abdulkader, and M. R. Elara, "Autonomous Self-Reconfigurable Floor Cleaning Robot," IEEE Access, vol. 8, pp. 114433–114442, 2020, doi: 10.1109/ACCESS.2020.2999202.
[27] P. Megantoro et al., "Autonomous and smart cleaning mobile robot system to improve the maintenance efficiency of solar photovoltaic array," Bull. Electr. Eng. Informatics, vol. 12, no. 6, pp. 3288–3297, 2023, doi: 10.11591/eei.v12i6.5950.
[28] R. A. Nanda, K. Karyadi, R. Roban, and F. M. Dewadi, "RPM MEASUREMENT COMPARISON USING A THERMOMETER AND LM393 MICROCONTROLLER," Int. J. Mech. Eng. Technol. Appl., vol. 5, no. 1, pp. 51–62, 2024.
[29] R. A. Nanda, K. Karyadi, and R. Roban, "Use of Mini Solar Panels for Battery Charging in the Mini Robot Warehouse," Circuit J. Ilm. Pendidik. Tek. Elektro, vol. 8, no. 1, pp. 1–15, 2024.
[30] N. R. A. Tukino, W. S. Gunawan, and T. S. Prasetyo Sri Yulianto Joko, "Analysis Transfer Data Image Processing and Face Recognition Using Camera ESP32CAM Web Browser IoT," ICI-EL, vol. 17, no. 06, p. 717, 2023.