IJRCS

International Journal of Robotics and Control Systems
Vol. 3, No. 2, 2023, pp. 233-244
ISSN 2775-2658
http://pubs2.ascee.org/index.php/ijrcs

ASCEE

Design and Application of PLC-based Speed Control for DC Motor Using PID with Identification System and MATLAB Tuner

Dodi Saputra [a], Alfian Ma'arif [a,1,*], Hari Maghfiroh [b], Phichitphon Chotikunnan [c], Safinta Nurindra Rahmadhia [d]

[a,d] Universitas Ahmad Dahlan, Yogyakarta 55191, Indonesia
[b] Department of Electrical Engineering, Universitas Sebelas Maret, Surakarta, Indonesia
[c] College of Biomedical Engineering, Rangsit University, Pathum Thani, Thailand
[1] alfianmaarif@ee.uad.ac.id
* Corresponding Author

ARTICLE INFO

Article history
Received August 07, 2022
Revised March 29, 2023
Accepted April 07, 2023

Keywords
PLC;
PID;
Tuning;
Motor DC;
MATLAB;
Tuner

ABSTRACT

Industries use numerous drives and actuators, including DC motors. Due to the wide-ranged and adjustable speed, DC motor is widely used in many industries. However, the DC motor is prone to external disturbance and parameter changes, causing its speed to be unstable. Thus, a DC motor requires an appropriate controller design to obtain a fast and stable speed with a small steady-state error. In this study, a controller was designed based on the PID control method, with the controller gains tuned by trial-and-error and MATLAB Tuner with an identification system. The proposed controller design was implemented using PLC OMRON CP1E NA20DRA in the hardware implementation. Each tuning method was repeated five times so that the system performances could be compared and improved. Based on hardware implementation results, the trial-error method gave acceptable results but had steady-state errors. On the other hand, the use of MATLAB Tuner provided fast system responses with no steady-state error but still had oscillations with high overshoot during the transition. Therefore, the PID controller gains acquired from MATLAB Tuner must be tuned finely to get better system responses.

1. Introduction

DC motor is one type of drive that is widely used in the industrial world [1]. This is because DC motors have many advantages over other motors [2], especially the wide-range speed variations that can be adjusted [3]. However, a change in load and other parameters or external disturbances can cause the speed of the DC motor to become unstable [4]. Therefore, a controller is needed [5]. Controlling the DC motor can make the DC motor performs more optimally [6].

Various control methods have been used for controlling DC motors, such as Fuzzy Logic Controller [7]–[10], Integral State Feedback [11][12], Sliding Mode Control [13][14], Neural Network [15], and PID Control [16]–[19]. However, PID controllers are more favorable in industries due to their easiness and reliability in controlling DC motor speed with a fast rise time, reduced error and reduced overshoot [20]. The PID control is a combination of three different control techniques, namely proportional, integral, and derivative controllers [21]. Each controller can be used simultaneously or

separately [22]. PID controllers are mostly applied to DC motors by administrating the control parameter values obtained from the trial-and-error method [23]; this process is repeated several times until the desired system performance is reached [24]. However, this is ineffective and time-consuming [25]. A faster tuning method is needed [26]. To obtain controlling parameters more quickly and efficiently, previous researchers have developed various tuning methods for PID controllers, such as Ziegler Nichols [27][28], Cohen – Coon [29], Genetic Algorithms [30], Particle Swarm Optimization [31], Fuzzy Tuning [32], Haris Hawk Optimization [33], Coronavirus Optimization [34] and MATLAB Tuner [35]. Among all tuning methods, MATLAB Tuner is the simplest yet most effective tuning method. It does not require additional programming since it is a MATLAB built-in feature, provides a detailed estimation of the system responses, and has an excellent user interface.

This research aims to produce a stable and constant DC motor rotational speed with reduced rise time, settling time, overshoot, and steady-state error. Before designing the controller and tuning its parameters, the DC motor system needs to be modeled first. In the research, determining the mathematical representation of the DC motor is carried out using a MATLAB system identification approach [36]. System identification is a method for obtaining a mathematical equation that represents the physical model of a system based on several experimental data [37]. Then, after the system's mathematical model is obtained, the augmented system, which consists of the DC motor and the proposed controller design, can be tuned using MATLAB Tuner.

PID controllers are usually embedded or programmed into controllers, processors, or supercomputers in practical hardware implementations [38]. Several commonly used devices are Arduino [39][40], ATMega microcontrollers, and Programmable Logic Controller (PLC) [41]. PLC has broad control functions for many purposes and applications; it can perform simple control functions to complex functions in the form of logic, timing and sequential [42][43]. Due to this reason, the research employed a PLC, specifically the PLC OMRON CP1E NA20DRA [44], to perform the DC motor speed control.

This research contributes to several designs and applications: applying a system identification approach to a DC motor controlled by the PID controller, designing a practical PID controller for speed control of DC motors, providing an analysis of practical hardware implementation of PID control method in DC motor, as well as a detailed performance comparison of using the trial-and-error method and MATLAB PID Tuner method.

2. Methods

2.1. System Design

Fig. 1 shows the overall system block diagram. The DC motor (JGA-25 370) is indirectly connected to a PLC (OMRON CP1E NA20DRA) as the controlled system and the motor speed controller through a serial connection of a motor driver and a voltage-to-PWM module and a feedback connection of a rotary encoder. The PLC is also connected to a personal computer (PC) that contains Human-to-Machine Interface (HMI), then the PC is connected to a laptop that contains MATLAB software.

The system begins with obtaining experimental data (paired input-output data) of the DC motor system through the PLC and the PC with installed HMI. The experimental data is then identified using a system identification approach so that a transfer function system model can be acquired and transmitted to a laptop with MATLAB software. Then, the subsequent procedure is inputting initialized data, the controller parameter values, into the PC. The controller (PLC) will process these data, generating an analog output. Then, the voltage-to-PWM module will convert these analog data into PWM signals, which the motor driver will receive and transmit to the DC motor. The DC motor speed is then adjusted and controlled by the output of the motor driver. Furthermore, the DC motor will rotate in the form of an angular rotation. The rotary encoder sensor then reads the number of rotations made by the DC motor and then translates the rotation into pulses. The pulse data is

forwarded to the PLC as input for PLC. The pulse data can then be read as Rotation Per Minute (RPM) data by performing a mathematical calculation.

A power supply is used in the system to power the electrical devices. Each device has different power supply requirements. For example, the PLC requires power from an AC power source of 220V. Meanwhile, other components, such as the DC motor and the rotary encoder, require DC power in different values. The flow of the power supply of PLC, rotary encoder, DC motor, and the voltage-to-PWM module is shown in the system schematic diagram in Fig. 2. Meanwhile, the power specifications of other components are listed in Table 1.

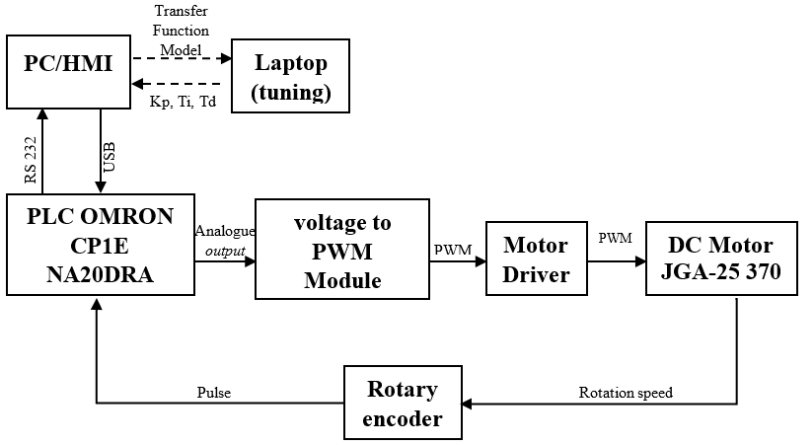

Fig. 1. System block diagram

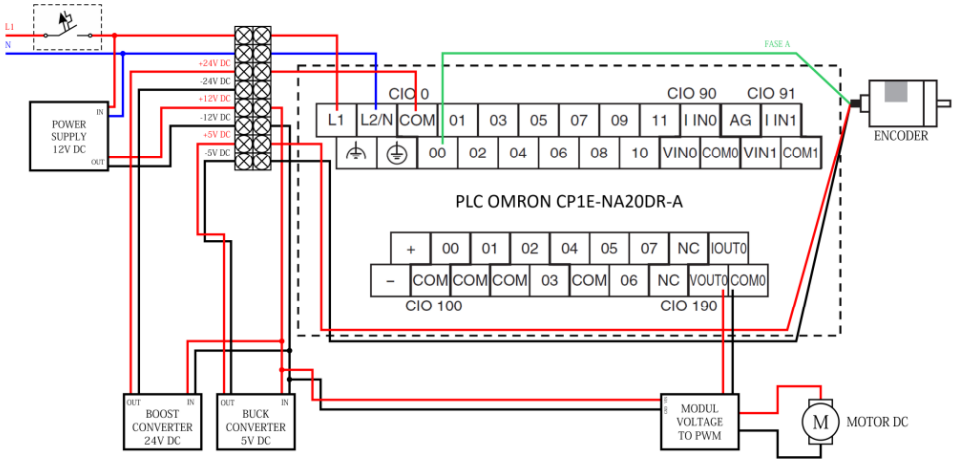

Fig. 2. Schematic diagram of the system

Table 1. Component's specification

No	Components	Input	Output
1	Rotary Encoder	5 VDC	Pulse 24 VDC
2	PLC	Supply 220 VAC CIO: 24 VDC	CIO 190: analog 0 – 5 V DC
3	Voltage to PWM module	Supply 12 V DC Input signal 0 – 5 V DC	PWM 0 -100 %
4	DC motor	PWM 12 V	Rotational speed

2.2. PID Controller

The PID controller is designed so that the rotational speed of the DC motor has a fast response with a reduced steady-state error and reaches the setpoint. To control the DC motor using the PID

control method, a rotary encoder sensor needs to read the number of rotations made by the DC motor, which will be forwarded along with the setpoint to the PLC for processing. The setpoint value is used as a reference and a control objective, resulting in a difference referred to as an error when compared to the measured rotational speed [45][46]. Then, the PLC generates an analog voltage based on the calculation that will be translated into PWM signals as input for the DC motor. This process is carried out continuously to get a small error value or an error value equal to zero, meaning that the measured rotational speed can be controlled accurately. Fig. 3 shows the block diagram of the proposed control method.

The PID-based PLC controller controls the DC motor using several controller parameters (gains). The success of the control method and the system performances are highly dependent on determining the controller parameter values. The research determines the controller parameter values by two different tuning methods: trial-and-error and the MATLAB Tuner.

In general, the parameters used in PID controllers are Kp, Ki, and Kd. However, in PLC Omron, the PIDAT (191) instruction uses different parameters, namely Proportional Band (PB), Integral Time (Tik), and Derivative Time (Tdk) [47]–[49]. Therefore, an equation is needed to change the parameters used by PIDAT (191), namely (1), (2), and (3).

$$PB = 100/Kp \tag{1}$$

$$Ki = Kp/Tik \tag{2}$$

$$Kd = Kp \times Tdk \tag{3}$$

where PB is Proportional Band (%), Tik is Integral Time (s), Tdk is Derivative Time (s), Kp is constant Proportional, Ki is constant Integral, Kd is constant Derivative.

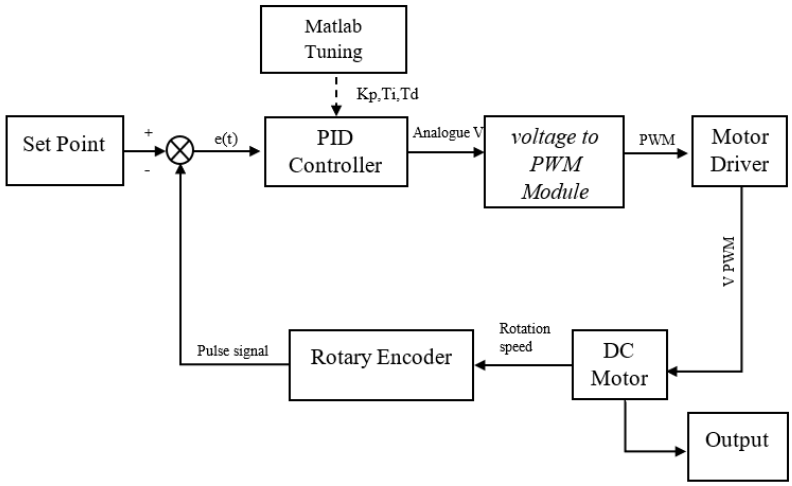

Fig. 3. Proposed control method

2.3. MATLAB System Identification

System identification is a method for making a mathematical equation of a physical system. The system identification model approach is conducted by taking input data and output data from a system. Then, an algorithm analyzes and processes this data as a mathematical model representing the relation between the system's input and output. One of the mathematical representations that resulted from the system identification process is the transfer function; later, the transfer function model will be used in the tuning process.

The principle of system identification is to find an estimated mathematical pattern based on the imported experimental data, which is the system's input and output. In MATLAB, this process can be

operated through one of the built-in toolboxes provided by the software (Fig. 4). This toolbox contains detailed system identification process functions, such as preprocessing and estimation. The experimental data can be imported directly into the toolbox's interface, and the mathematical operations used for the system identification algorithm can be selected. The toolbox also contains functions for analyzing the performance of the estimated system model that makes the toolbox a complete package for system identification.

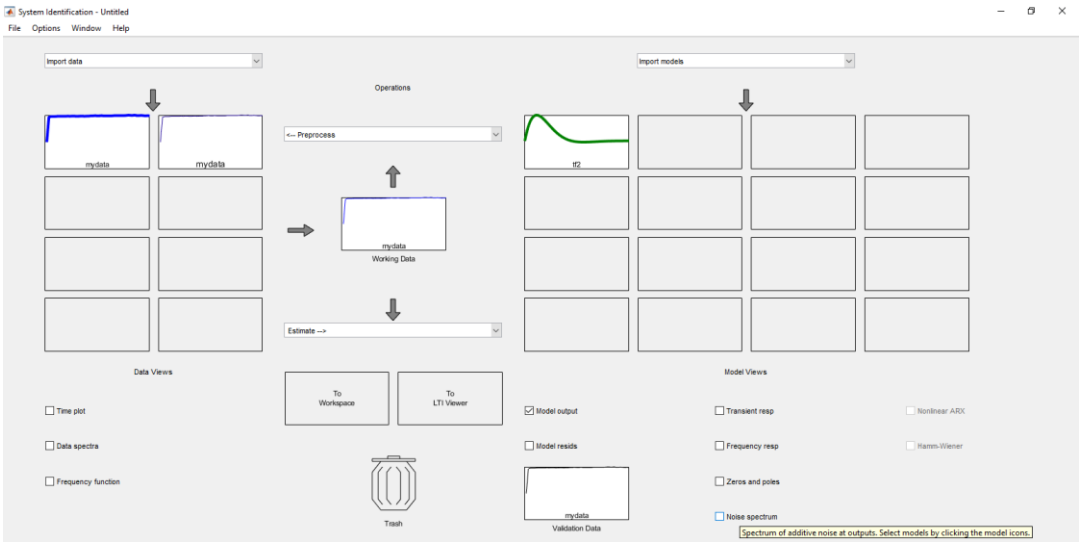

Fig. 4. MATLAB System Identification Toolbox

2.4. MATLAB PID Tuner

The transfer function model obtained from the system identification process is then inputted through the command window and will be processed for designing the suitable PID controller parameters (tuning process). MATLAB provides tuning functions for the PID controller with the PID Tuner Toolbox, as shown in Fig. 5.

The toolbox helps to tune the PID controller by providing parameter values with the best system performance specifications. These parameter values can be entered into the augmented system, especially in the controller gains, to start controlling the DC motor system. The controller parameter values obtained from the tuning process performed on the PID tuner toolbox are shown in Table 2.

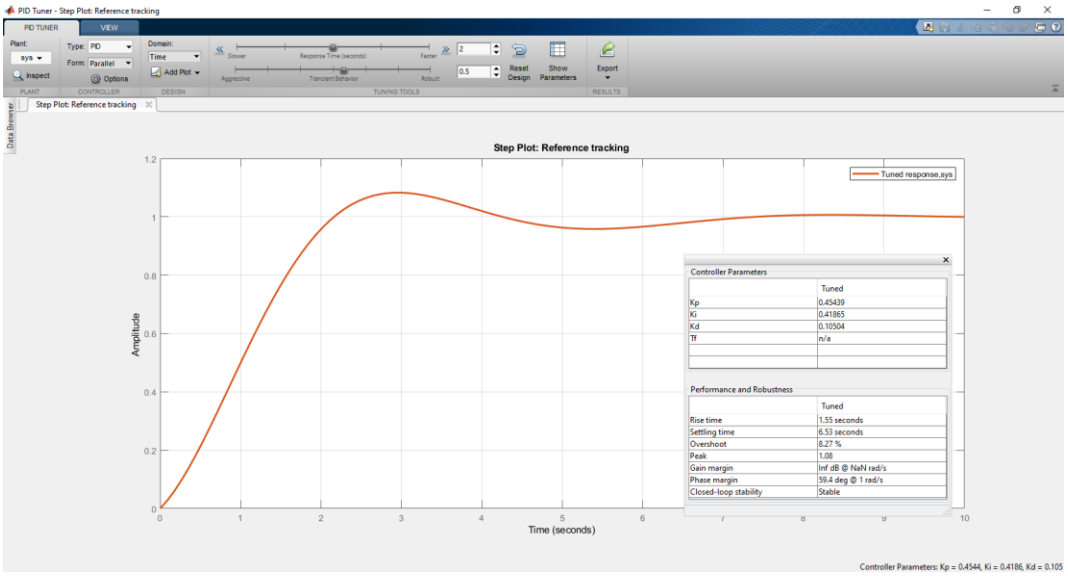

Fig. 5. MATLAB PID Tuner

Table 2. MATLAB-based PID Tuner Results: Controller Gains and Expected System Performances

No	Proportional (Kp)	Integral (Ti)	Derivative (Td)	Robust Time	Transient Behavior
1	0.26188	0.8808	0.22002	2.316	0.568
2	0.2682	0.89591	0.22398	2.295	0.57
3	0.27458	0.91173	0.22793	2.274	0.572
4	0.28102	0.92752	0.23188	2.251	0.574
5	0.28827	0.9469	0.23673	2.232	0.576

2.5. Human-Machine Interface

Applying and implementing the controller design, along with inputting the controller gains from the tuning process, is conducted using PLC. Therefore, a visual interface is needed to apply and monitor the PLC-integrated control system. The visual interface, commonly known as Human-Machine Interface (HMI), is programmed to bridge between operation/control commands from humans and monitoring status of the machine, as shown in Fig. 6. The HMI includes a feature that functions to input PID controller gains, namely Kp, Ki, and Kd. It also contains features enabling inputting the sampling time, setpoint, autotuning and a PID On/Off button. The HMI also displays rotational speed graphs and motor speed data in RPM. Fig. 7 shows the prototype of the proposed system hardware, which has been made in the research.

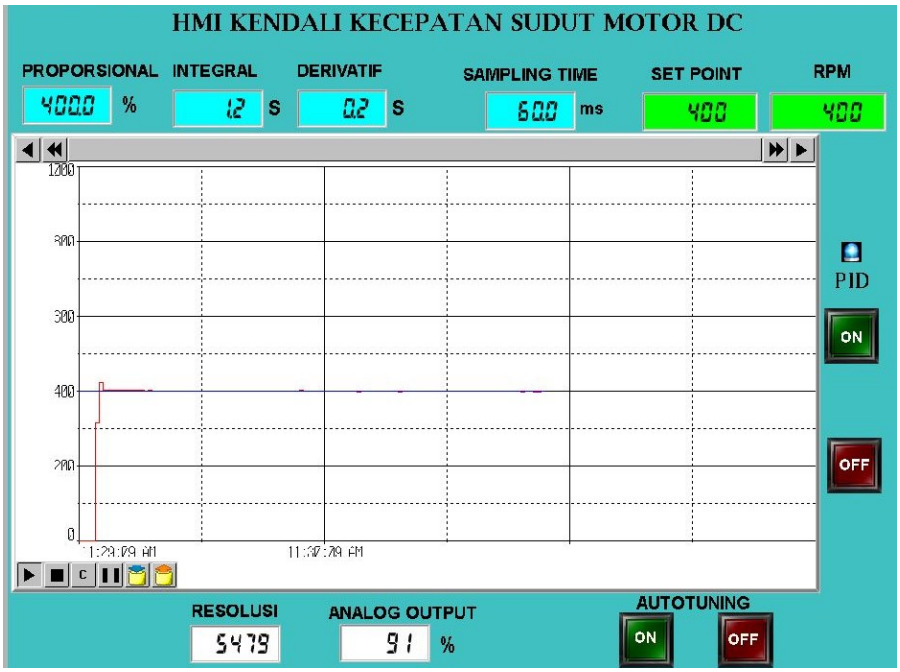

Fig. 6. Human-Machine Interface

3. Results and Discussions

The results and discussion of this study consist of the performance test results of DC motor rotational speed control using the trial-and-error method and MATLAB Tuner. Each test was carried out in five replications.

3.1. System Identification

The DC motor system was run in an open-loop test to obtain experimental data for system identification. The DC motor system's input and output were recorded and imported into the MATLAB system identification toolbox. According to the system identification results, the transfer function model was then obtained, as presented in (4).

$$\frac{2.671}{s^2 + 1.471s + 1.104} \quad (4)$$

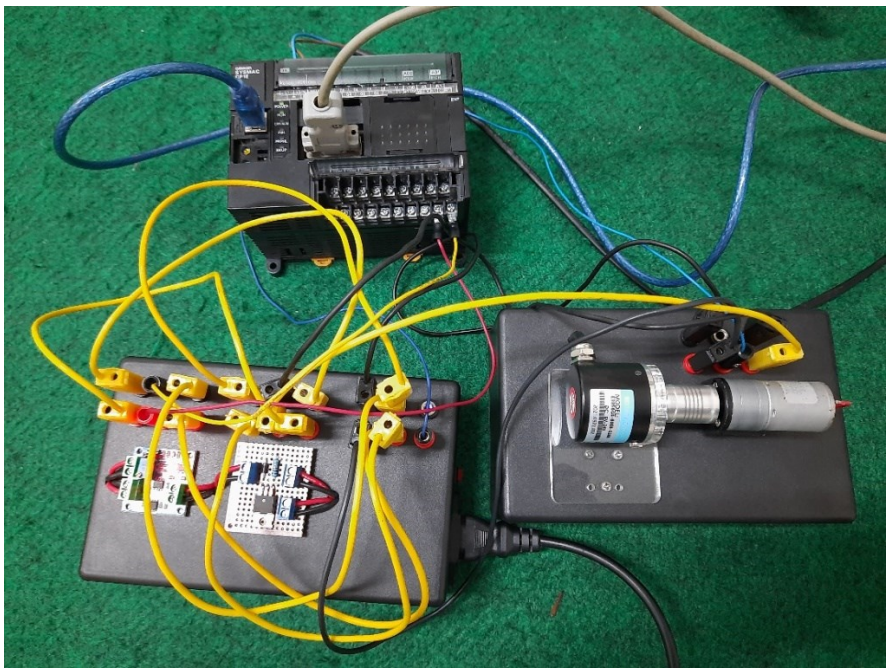

Fig. 7. System Hardware

3.2. Manual PID Tuning

A tuning for PID controller parameters that is done manually is also commonly known as the trial-and-error method [50]. The term trial-and-error refers to repetitive attempts in the problem-solving process until a desired result is achieved. In other words, the trial-and-error method in the PID tuning process is conducted by repeatedly inputting several combinations of PID parameter values until a desired system response is achieved or the performance specifications criteria are met.

In this study, the test was carried out 5 times with the same setpoint value but different control parameters. The list of PID parameters obtained through the trial-and-error method is resumed in Table 3. The system response of each combination of the controller gains was observed and was then analyzed to assess the controller's performance, as shown in Fig. 8. Meanwhile, the detailed system performance of each combination of controller gains obtained from the trial-and-error method can be seen in Table 4. According to the system responses and performance results, the trial-and-error method could provide a stable DC motor rotational speed control. It can be seen that, from five times of trials, the average SSE, rise-time, peak-time, settling time, and maximum overshoot are: -0.8%, 3.269s, 16s, 11.111s, and 5.150, respectively.

Table 3. Controller Gains Obtained from Manual PID Tuning

No	Setpoint	Sampling Time	Proportional (PB)	Integral (Tik)	Derivative (Tdk)
1	400	60	400	0.4	0.1
2	400	60	300	0.8	0.2
3	400	60	400	1.2	0.2
4	400	60	500	0.8	0.2
5	400	60	600	1.2	0.2

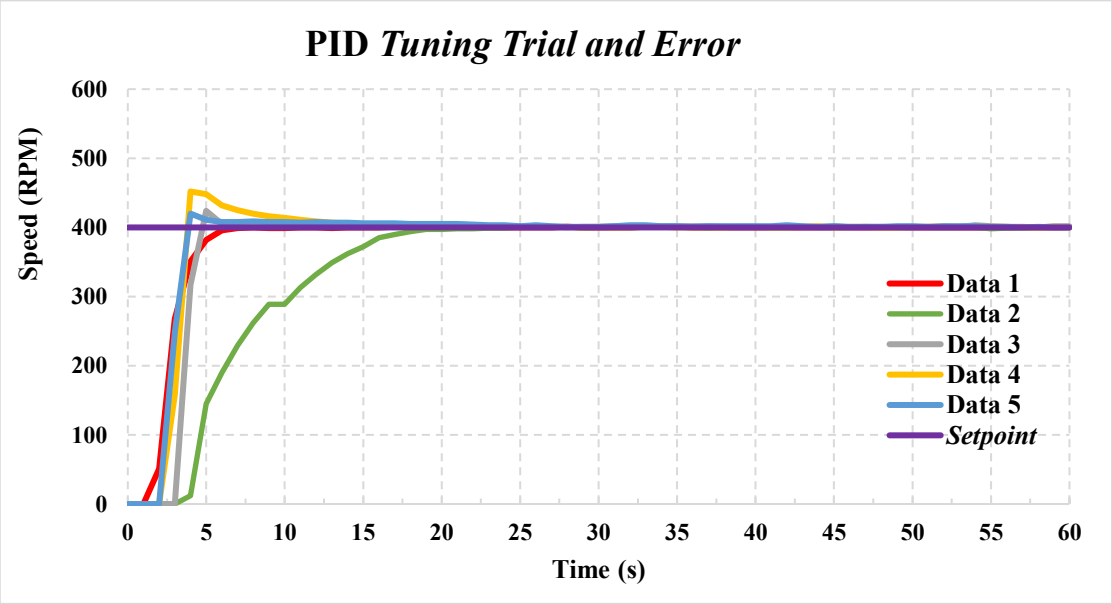

Fig. 8. System Responses of Applying Manual PID Tuning

Table 4. System Performance Results of Applying Manual PID Tuning

No	Tuning Trial and Error	Steady State Error (%)	Rise Time (s)	Peak Time (s)	Settling Time (s)	Maximum overshoot (s)
1	Data 1	0	2.506	18	6.714	0.250
2	Data 2	-1	9.636	46	18.500	1.500
3	Data 3	-2	1.276	6	6.842	6.000
4	Data 4	-1	1.434	5	13.500	13.000
5	Data 5	0	1.490	5	10.000	5.000
	Mean	-0.8	3.269	16	11.111	5.150

3.3. PID MATLAB Tuner

The MATLAB-tuned PID controller's performance is tested by inputting the controller parameters derived from the MATLAB software on the HMI. Initialization of desired response time and transient behavior settings is necessarily made to obtain the suitable controller parameters.

Similar to the previous test, the test was carried out five times with the same setpoint value but different control parameters. The obtained combinations of controller parameters are resumed and listed in Table 5. The system response of each combination of the controller gains was observed and was then analyzed to assess the controller's performance, as seen in Fig. 9. By using the PID controller with the MATLAB Tuner, the speed performance of the DC motor is listed in Table 6.

Table 5. PID with MATLAB Tuner

No	Setpoint	Sampling Time	Proportional (PB)	Integral (Tik)	Derivative (Tid)
1	400	60	381.8 %	0.2 s	0.1 s
2	400	60	372.8 %	0.2 s	0.1 s
3	400	60	364.1 %	0.3 s	0.1 s
4	400	60	355.8 %	0.3 s	0.1 s
5	400	60	346.8 %	0.3 s	0.1 s

As can be seen from the system responses and performance results, the steady-state error or SSE was equal to zero. However, the speed oscillated before reaching a steady state. The oscillated speed resulted in big overshoots; the average overshoot was 105%.

These results had better SSE, rise time, and peak time than results from the manual tuning using the trial-and-error method. On the other side, its settling time value was slightly bigger, and the overshoot was unacceptable. However, these can be adjusted by fine-tuning the PID gains using MATLAB Tuner.

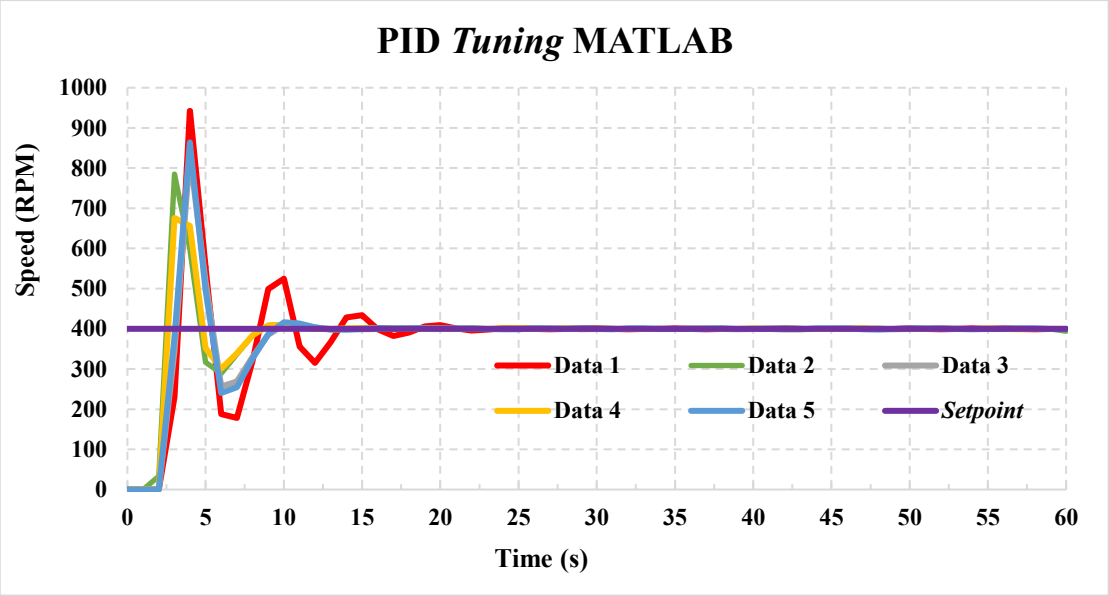

Fig. 9. System Responses of Applying PID with MATLAB Tuner

Table 6. System Performance Results of Applying PID with MATLAB Tuner

No	Tuning	Steady State Error (%)	Rise Time (s)	Peak Time (s)	Settling Time (s)	Overshoot (RPM)
1	Data 1	0	1.009	5	18.796	135.500
2	Data 2	0	0.426	4	11.429	96.250
3	Data 3	0	0.915	5	12.334	108.250
4	Data 4	0	0.473	4	11.286	69.000
5	Data 5	0	0.844	5	12.413	116.000
	Mean	0	0.733	4.6	13.252	105.000

4. Conclusions

This research contributes to several designs and applications: applying a system identification approach to a DC motor controlled by the PID controller, designing a practical PID controller for speed control of DC motors, providing an analysis of practical hardware implementation of PID control method in DC motor, as well as a detailed performance comparison of using a trial-and-error method and MATLAB PID Tuner method. The design and hardware implementation of PID-based DC motor speed control using PLC OMRON CP1E NA20DRA was successfully done in the study. According to the hardware implementation results, both methods could control the DC motor system with different performance specifications. However, MATLAB PID Tuner had better system performance results than the trial-and-error method in general. The trial-and-error method provided acceptable results with steady-state error. Meanwhile, MATLAB PID Tuner gave fast system responses with no SSE even though the speed oscillated and had a big overshoot during the transient state prior to fine-tuning.

Author Contribution: All authors contributed equally to the main contributor to this paper. All authors read and approved the final paper.

Funding: This research was funded by Universitas Ahmad Dahlan.

Conflicts of Interest: The authors declare no conflict of interest.

References

[1] B. N. Kommula and V. R. Kota, "Direct instantaneous torque control of Brushless DC motor using firefly Algorithm based fractional order PID controller," *Journal of King Saud University - Engineering Sciences*, vol. 32, no. 2, pp. 133–140, Feb. 2020, https://doi.org/10.1016/j.jksues.2018.04.007.

[2] D. Potnuru, K. Alice Mary, and C. Sai Babu, "Experimental implementation of Flower Pollination Algorithm for speed controller of a BLDC motor," *Ain Shams Engineering Journal*, vol. 10, no. 2, pp. 287–295, Jun. 2019, https://doi.org/10.1016/j.asej.2018.07.005.

[3] K. Vanchinathan and N. Selvaganesan, "Adaptive fractional order PID controller tuning for brushless DC motor using Artificial Bee Colony algorithm," *Results in Control and Optimization*, vol. 4, p. 100032, Sep. 2021, https://doi.org/10.1016/j.rico.2021.100032.

[4] A. M. Zaki, M. El-Bardini, F. A. S. Soliman, and M. M. Sharaf, "Embedded two level direct adaptive fuzzy controller for DC motor speed control," *Ain Shams Engineering Journal*, vol. 9, no. 1, pp. 65–75, Mar. 2018, https://doi.org/10.1016/j.asej.2015.10.003.

[5] B. Hekimoglu, "Optimal Tuning of Fractional Order PID Controller for DC Motor Speed Control via Chaotic Atom Search Optimization Algorithm," *IEEE Access*, vol. 7, pp. 38100–38114, 2019, https://doi.org/10.1109/ACCESS.2019.2905961.

[6] 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 Engineering Journal*, vol. 9, no. 3, pp. 341–352, Sep. 2018, https://doi.org/10.1016/j.asej.2016.02.004.

[7] A. Lotfy, M. Kaveh, M. R. Mosavi, and A. R. Rahmati, "An enhanced fuzzy controller based on improved genetic algorithm for speed control of DC motors," *Analog Integrated Circuits and Signal Processing*, vol. 105, no. 2, pp. 141–155, Nov. 2020, https://doi.org/10.1007/s10470-020-01599-9.

[8] H. R. Patel, "Fuzzy-based metaheuristic algorithm for optimization of fuzzy controller: fault-tolerant control application," *International Journal of Intelligent Computing and Cybernetics*, vol. 15, no. 4, pp. 599–624, Sep. 2022, https://doi.org/10.1108/IJICC-09-2021-0204.

[9] I. Suwarno, Y. Finayani, R. Rahim, J. Alhamid, and A. R. Al-Obaidi, "Controllability and Observability Analysis of DC Motor System and a Design of FLC-Based Speed Control Algorithm," *Journal of Robotics and Control (JRC)*, vol. 3, no. 2, pp. 227–235, Feb. 2022, https://doi.org/10.18196/jrc.v3i2.10741.

[10] A. L. Shuraiji and S. W. Shneen, "Fuzzy Logic Control and PID Controller for Brushless Permanent Magnetic Direct Current Motor: A Comparative Study," *Journal of Robotics and Control (JRC)*, vol. 3, no. 6, pp. 762–768, Dec. 2022, https://doi.org/10.18196/jrc.v3i6.15974.

[11] N. R. Setiawan, A. Ma'arif, and N. S. Widodo, "DC Motor Controller Using Full State Feedback," *Control Systems and Optimization Letters*, vol. 1, no. 1, pp. 7–11, Mar. 2023, https://doi.org/10.59247/csol.v1i1.3.

[12] A. Apte, V. A. Joshi, H. Mehta, and R. Walambe, "Disturbance-Observer-Based Sensorless Control of PMSM Using Integral State Feedback Controller," *IEEE Transactions on Power Electronics*, vol. 35, no. 6, pp. 6082–6090, Jun. 2020, https://doi.org/10.1109/TPEL.2019.2949921.

[13] A. Durdu and E. H. Dursun, "Sliding Mode Control for Position Tracking of Servo System with a Variable Loaded DC Motor," *Elektronika ir Elektrotechnika*, vol. 25, no. 4, pp. 8–16, Aug. 2019, https://doi.org/10.5755/j01.eie.25.4.23964.

[14] A. Ma'arif and A. Çakan, "Simulation and Arduino Hardware Implementation of DC Motor Control Using Sliding Mode Controller," *Journal of Robotics and Control (JRC)*, vol. 2, no. 6, pp. 582–587, 2021, https://doi.org/10.18196/jrc.26140.

[15] J. Peng and R. Dubay, "Identification and adaptive neural network control of a DC motor system with dead-zone characteristics," *ISA Transactions*, vol. 50, no. 4, pp. 588–598, Oct. 2011, https://doi.org/10.1016/j.isatra.2011.06.005.

[16] S. Ekinci, B. Hekimoğlu, and D. Izci, "Opposition based Henry gas solubility optimization as a novel algorithm for PID control of DC motor," *Engineering Science and Technology, an International Journal*, vol. 24, no. 2, pp. 331–342, Apr. 2021, https://doi.org/10.1016/j.jestch.2020.08.011.

[17] Q. Ariyansyah and A. Ma'arif, "DC Motor Speed Control with Proportional Integral Derivative (PID)

Control on the Prototype of a Mini-Submarine," *Journal of Fuzzy Systems and Control*, vol. 1, no. 1, 2023, https://ejournal.ptti.web.id/index.php/jfsc/article/view/26.

[18] R. Rikwan and A. Ma'arif, "DC Motor Rotary Speed Control with Arduino UNO Based PID Control," *Control Systems and Optimization Letters*, vol. 1, no. 1, pp. 17–31, Mar. 2023, https://doi.org/10.59247/csol.v1i1.6.

[19] Z. B. Abdullah, S. W. Shneen, and H. S. Dakheel, "Simulation Model of PID Controller for DC Servo Motor at Variable and Constant Speed by Using MATLAB," *Journal of Robotics and Control (JRC)*, vol. 4, no. 1, pp. 54–59, Feb. 2023, https://doi.org/10.18196/jrc.v4i1.15866.

[20] R. P. Borase, D. K. Maghade, S. Y. Sondkar, and S. N. Pawar, "A review of PID control, tuning methods and applications," *International Journal of Dynamics and Control*, vol. 9, no. 2, pp. 818–827, Jul. 2020, https://doi.org/10.1007/s40435-020-00665-4.

[21] E. S. Ghith, F. Abdel, and A. Tolba, "Design and Optimization of PID Controller using Various Algorithms for Micro-Robotics System," *Journal of Robotics and Control (JRC)*, vol. 3, no. 3, pp. 244–256, May 2022, https://doi.org/10.18196/jrc.v3i3.14827.

[22] M. Filo, S. Kumar, and M. Khammash, "A hierarchy of biomolecular proportional-integral-derivative feedback controllers for robust perfect adaptation and dynamic performance," *Nature Communications*, vol. 13, no. 1, pp. 1–19, Apr. 2022, https://doi.org/10.1038/s41467-022-29640-7.

[23] M. Saad, A. H. Amhedb, and M. Al Sharqawi, "Position Control of Real Time DC Motor Using LabVIEW," *Journal of Robotics and Control (JRC)*, vol. 2, no. 5, pp. 342–348, Sep. 2021, https://doi.org/10.18196/jrc.25104.

[24] M. S. Chehadeh and I. Boiko, "Design of rules for in-flight non-parametric tuning of PID controllers for unmanned aerial vehicles," *Journal of the Franklin Institute*, vol. 356, no. 1, pp. 474–491, Jan. 2019, https://doi.org/10.1016/j.jfranklin.2018.10.015.

[25] P. Mitra, C. Dey, and R. K. Mudi, "Fuzzy rule-based set point weighting for fuzzy PID controller," *SN Applied Sciences*, vol. 3, no. 6, pp. 1–34, Jun. 2021, https://doi.org/10.1007/s42452-021-04626-0.

[26] Z. Qi, Q. Shi, and H. Zhang, "Tuning of digital PID controllers using particle swarm optimization algorithm for a CAN-Based DC motor subject to stochastic delays," *IEEE Transactions on Industrial Electronics*, vol. 67, no. 7, pp. 5637–5646, Jul. 2020, https://doi.org/10.1109/TIE.2019.2934030.

[27] A. Ghosh, A. K. Ray, M. Nurujjaman, and M. Jamshidi, "Voltage and frequency control in conventional and PV integrated power systems by a particle swarm optimized Ziegler–Nichols based PID controller," *SN Applied Sciences*, vol. 3, no. 3, pp. 1–13, Mar. 2021, https://doi.org/10.1007/s42452-021-04327-8.

[28] V. V. Patel, "Ziegler-Nichols Tuning Method: Understanding the PID Controller," *Resonance*, vol. 25, no. 10, pp. 1385–1397, Oct. 2020, https://doi.org/10.1007/s12045-020-1058-z.

[29] S. B. Joseph, E. G. Dada, A. Abidemi, D. O. Oyewola, and B. M. Khammas, "Metaheuristic algorithms for PID controller parameters tuning: review, approaches and open problems," *Heliyon*, vol. 8, no. 5, p. e09399, May 2022, https://doi.org/10.1016/j.heliyon.2022.e09399.

[30] T. Hui, W. Zeng, and T. Yu, "Core power control of the ADS based on genetic algorithm tuning PID controller," *Nuclear Engineering and Design*, vol. 370, p. 110835, Dec. 2020, https://doi.org/10.1016/j.nucengdes.2020.110835.

[31] H. Feng, W. Ma, C. Yin, and D. Cao, "Trajectory control of electro-hydraulic position servo system using improved PSO-PID controller," *Automation in Construction*, vol. 127, p. 103722, Jul. 2021, https://doi.org/10.1016/j.autcon.2021.103722.

[32] P. Sarkhel, N. Banerjee, and N. B. Hui, "Fuzzy logic-based tuning of PID controller to control flexible manipulators," *SN Applied Sciences*, vol. 2, no. 6, pp. 1–11, Jun. 2020, https://doi.org/10.1007/s42452-020-2877-y.

[33] V. K. Munagala and R. K. Jatoth, "A novel approach for controlling DC motor speed using NARXnet based FOPID controller," *Evolving Systems*, vol. 14, no. 1, pp. 101–116, Feb. 2023, https://doi.org/10.1007/s12530-022-09437-1.

[34] M. A. Shamseldin, R. Barbosa, and I. Jesus, "Optimal Coronavirus Optimization Algorithm Based PID Controller for High Performance Brushless DC Motor," *Algorithms*, vol. 14, no. 7, p. 193, Jun. 2021, https://doi.org/10.3390/a14070193.

[35] L. Wang, *PID control system design and automatic tuning using MATLAB/Simulink*. John Wiley & Sons, 2020, https://doi.org/10.1002/9781119469414.

[36] E. A. Abioye *et al.*, "IoT-based monitoring and data-driven modelling of drip irrigation system for mustard leaf cultivation experiment," *Information Processing in Agriculture*, vol. 8, no. 2, pp. 270–283, Jun. 2021, https://doi.org/10.1016/j.inpa.2020.05.004.

[37] T. Kuntoro Priyambodo, A. Majid, Z. Saad, and S. Shouran, "Validation of Quad Tail-sitter VTOL UAV Model in Fixed Wing Mode," *Journal of Robotics and Control (JRC)*, vol. 4, no. 2, pp. 179–191, Apr. 2023, https://doi.org/10.18196/jrc.v4i2.17253.

[38] S. S. Khairullah and A. N. Sharkawy, "Design and Implementation of a Reliable and Secure Controller for Smart Home Applications Based on PLC," *Journal of Robotics and Control (JRC)*, vol. 3, no. 5, pp. 614–621, Sep. 2022, https://doi.org/10.18196/jrc.v3i5.15972.

[39] A. Kherkhar, Y. Chiba, A. Tlemçani, and H. Mamur, "Thermal investigation of a thermoelectric cooler based on Arduino and PID control approach," *Case Studies in Thermal Engineering*, vol. 36, p. 102249, Aug. 2022, https://doi.org/10.1016/j.csite.2022.102249.

[40] P. B. de Moura Oliveira, J. D. Hedengren, and E. J. Solteiro Pires, "Swarm-Based Design of Proportional Integral and Derivative Controllers Using a Compromise Cost Function: An Arduino Temperature Laboratory Case Study," *Algorithms*, vol. 13, no. 12, p. 315, Nov. 2020, https://doi.org/10.3390/a13120315.

[41] J. Możaryn, J. Petryszyn, and S. Ozana, "PLC based fractional-order PID temperature control in pipeline: design procedure and experimental evaluation," *Meccanica*, vol. 56, no. 4, pp. 855–871, Apr. 2021, https://doi.org/10.1007/s11012-020-01215-0.

[42] J. Mellado and F. Núñez, "Design of an IoT-PLC: A containerized programmable logical controller for the industry 4.0," *Journal of Industrial Information Integration*, vol. 25, p. 100250, Jan. 2022, https://doi.org/10.1016/j.jii.2021.100250.

[43] Z. Wang, Y. Zhang, Y. Chen, H. Liu, B. Wang, and C. Wang, "A Survey on Programmable Logic Controller Vulnerabilities, Attacks, Detections, and Forensics," *Processes*, vol. 11, no. 3, p. 918, Mar. 2023, https://doi.org/10.3390/pr11030918.

[44] F. A. Aziz and R. D. Puriyanto, "Rancang Bangun Mesin Pengecat Dinding Otomatis Berbasis PLC CP1E-NA20DR-A," *Buletin Ilmiah Sarjana Teknik Elektro*, vol. 1, no. 3, pp. 118–130, 2019, https://doi.org/10.12928/biste.v1i3.1050.

[45] A. Rajagukguk, W. Arafanaldy, A. Anhar, and N. Nurhalim, "Pitch Blade Control Prototype Design for Vertical Axis Wind Power Plant," *Jurnal Ilmiah Teknik Elektro Komputer dan Informatika*, vol. 8, no. 1, pp. 157–166, May 2022, https://doi.org/10.26555/jiteki.v8i1.23662.

[46] P. Sutyasadi, "Control Improvement of Low-Cost Cast Aluminium Robotic Arm Using Arduino Based Computed Torque Control," *Jurnal Ilmiah Teknik Elektro Komputer dan Informatika*, vol. 8, no. 4, pp. 650–659, Dec. 2022, https://doi.org/10.26555/jiteki.v8i4.24646.

[47] M. F. Al Andzar and R. D. Puriyanto, "PID Control for Temperature and Motor Speed Based on PLC," *Signal and Image Processing Letters*, vol. 1, no. 1, pp. 7–13, Mar. 2019, https://doi.org/10.31763/simple.v1i1.150.

[48] A. Setiawan and A. Ma'arif, "Stirring System Design for Automatic Coffee Maker Using OMRON PLC and PID Control," *International Journal of Robotics and Control Systems*, vol. 1, no. 3, pp. 390–401, Oct. 2021, https://doi.org/10.31763/ijrcs.v1i3.457.

[49] D. Prasetiyo and W. Sapto Aji, "Irrigation Sluice Control System Using Algorithm Based DC Motor PID And Omron PLC," *Control Systems and Optimization Letters*, vol. 1, no. 1, pp. 19–26, Mar. 2023, https://doi.org/10.59247/csol.v1i1.5.

[50] G. R. Wicaksono and R. D. Puriyanto, "Programmable Logic Controller (PLC) Based Paint Viscosity Control System," *Buletin Ilmiah Sarjana Teknik Elektro*, vol. 3, no. 1, pp. 1–9, 2021, https://doi.org/10.12928/biste.v3i1.1109.