Bulletin of Network Engineer and Informatics
E-ISSN 2986-8017 | P-ISSN 2987-4858
https://bufnets.tech
Vol. 4 No 1 – April 2026
DOI: 10.59688/6g8j1952

Performance Evaluation of PID Control for Distance Maintenance in Wall-Following Robots Navigating Dynamic Trajectories

Syaiful Mansur[1], Muhammad Ridwan Al Hafiz[2], Aknisa Alqibetia Sari[3], Ridho Akmal Sembiring[4]

[1,2,3,4]Program Studi Teknologi Rekayasa Elektronika,Politeknik Negeri Lampung Jl. Soekarno Hatta No.10, Kota Bandar Lampung 35141

e-mail:syaifulmansur@polinela.ac.id[1], mridwanalhafi05@gmail.com[2], aknisaalqibetiasark@gmail.com[3], ridhoakmalsembiring02@gmail.com[4]

Received : 19 April 2026	Accepted : 29 April 2026	Published : 01 May 2026

Abstract

Autonomous navigation in wall-following robots plays a crucial role in maintaining movement stability within environments characterized by diverse path geometries. Conventional control methods often encounter oscillations and increased collision risks, particularly on dynamic trajectories. This study evaluates the performance of Proportional-Integral-Derivative (PID) control in maintaining a robot's navigation distance relative to a fixed 10 cm setpoint. The system was developed utilizing an Arduino Mega 2560 microcontroller and HC-SR04 ultrasonic sensors for distance feedback. PID parameters were determined using the Ziegler-Nichols tuning method, yielding optimal values of $K_p = 2$, $K_i = 0.5$, *and* $K_d = 1$. *Testing was conducted on both linear paths and dynamic trajectories consisting of 90° turns. The results demonstrate that on linear paths, the system successfully maintains the target distance with high precision, achieving deviations ranging from* ± 0.56 *cm to* ± 1.13 *cm with zero collisions. In contrast, a significant performance trade-off was observed on dynamic trajectories, where a spike in transient error led to an average distance shift to 11.88 cm and 1–2 collisions per trial. These findings highlight that while PID control is highly effective under stable linear conditions, it possesses inherent limitations in responding to rapid geometric changes and mechanical inertia. Consequently, further development, such as adaptive control logic or high-sampling-rate sensing, is required to enhance system robustness on dynamic trajectories.*

Keywords: *PID Control, Wall-Following Robot, Dynamic Response, Navigation Error, Ultrasonic sensor.*

1. INTRODUCTION

Autonomous wall-following navigation represents a critical algorithm for rescue robots, enabling independent operation within post-disaster building environments[1]. This algorithm allows the robot to maintain its path orientation in complex environments without total dependence on a human operator. However, the implementation of conventional control strategies, such as on-off systems, frequently triggers high-oscillation movements (zig-zagging). Such erratic behavior increases the risk of collisions, particularly when navigating dynamic changes in path geometry[2].

The application of Proportional-Integral-Derivative (PID) control serves as a technical solution to mitigate oscillations and enhance navigational accuracy through continuous error correction. By precisely tuning the K_p, K_i , and K_d parameters, the robot is expected to achieve a smooth transient response[3][4][5][6]. While previous studies [1]-[6] have extensively discussed PID implementation for general navigation, there is a limited focus on the controller's robustness when faced with extreme, non-linear geometric shifts such as sharp 90-degree turns. Most existing research evaluates stability on linear paths or smooth curves, which may not fully represent the structural unpredictability of post-disaster sites.

This study addresses this gap by specifically evaluating the performance and stability limits of a PID-based prototype on high-curvature dynamic trajectories. The research focuses on the stability of the reference distance and analyzes how the controller manages the trade-off between error convergence and mechanical inertia during rapid direction changes. The results of this analysis are intended to provide a technical reference regarding the optimization of mobile robot navigation across both linear and dynamic trajectories for emergency response applications.

2. RESEARCH METHODOLOGY

2.1 System Architecture and Hardware

The robot's navigation system was developed using an Arduino Mega 2560 microcontroller as the central processing unit to integrate various sensors and actuators. The selection of the Arduino Mega was based on its sufficient I/O pin density, supporting the integration of three HC-SR04 ultrasonic sensors[7][8]. Distance data acquired from these sensors are processed to determine the robot's spatial position relative to the wall.

To provide a comprehensive mechanical context for the performance analysis, the physical and technical specifications of the robot prototype are summarized in Table 1.

Table 1: Specifications of the Robot Prototype
[Source: Author, 2026]

No	Parameter	Specification
1	Microcontroller	Arduino Mega 2560
2	Sensors	3 x HC-SR04 Ultrasonic Sensors
3	Actuators	2 x DC Gear Motors JGA25-370 12V
4	Total Mass	1.6 kg
5	Dimensions (L x W x H)	25 x 15.5 x 16 cm
6	Wheel Diameter	7.5 cm
7	Power Source	11.1V Li-Po Battery

The system's mass and dimensions are critical factors that contribute to the mechanical inertia. As discussed in the subsequent sections, this inertia significantly affects the robot's momentum and its ability to execute rapid corrective Pulse Width Modulation (PWM) changes generated by the PID controller, particularly during sharp 90° turns.

2.2 Control System Block Diagram

The control system follows a closed-loop feedback principle, where sensor data is continuously updated to correct the motor velocity. The ultrasonic sensors detect the actual distance (y), which is compared against a reference value or setpoint (r). The difference between these two values generates an error signal (e) that is processed by the PID controller to produce the Pulse Width Modulation (PWM) control signal.

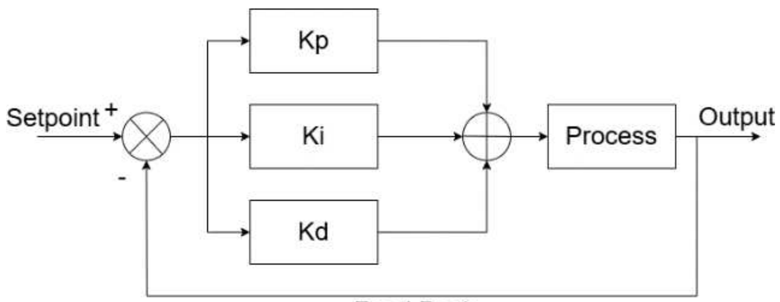

Figure 1. PID Control System Block Diagram
[Source: Author, 2026]

2.3 PID Algorithm Implementation

Implementation on a microcontroller requires the conversion of the continuous PID equation into a discrete-time form suitable for processing within each program iteration cycle[9][10]. Utilizing the Euler approximation method, the control law implemented in the firmware is defined as follows:

$$Output_{PID} = K_p \cdot e(n) + K_i \cdot \sum e(n) \cdot \Delta t + K_d \cdot \frac{e(n) - e(n-1)}{\Delta t} \quad (1)$$

- $e(n)$ is the current error;
- $e(n-1)$ *is the previous error;*

- Δt is the sampling time.

2.4 Tunning Ziegler-Nichols Method

The controller gains were determined using the Ziegler-Nichols (Type II) oscillation method to achieve an optimal transient response[11]. The tuning procedure was conducted through the following experimental stages:

1. Identification of Ultimate Gain K_u: the K_i and K_d parameters were set to zero, while K_p was gradually increased until the robot reached a state of continuous oscillation (marginal stability). The ultimate gain (K_u) was identified at 3.33.
2. Determination of Ultimate Period (T_u): The time period of one full oscillation at the K_u state was recorded as 0.8 seconds, serving as the system's natural frequency reference.
3. Parameter Calculation: Based on K_u and T_u, the initial parameters were calculated using standard PID formulas as shown in Equations (2), (3), and (4):

$$K_p = 0{,}6 \times K_u = 2 \tag{2}$$

$$K_i = \frac{2 \times K_p}{T_u} = 5 \tag{3}$$

$$K_d = \frac{K_p \times T_u}{8} = 0.2 \tag{4}$$

However, the initial parameters from the Z-N formulas resulted in excessive overshoot and instability during the robot's transition at 90° turns due to mechanical inertia. Therefore, a fine-tuning process was conducted to optimize the response. The integral gain (K_i) was reduced to 0.5 to minimize cumulative error oscillation, and the derivative gain (K_d) was increased to 1 to provide stronger damping against rapid distance changes. Consequently, the final optimal values adopted for this study are $K_p = 2, K_i = 0.5$, and $K_d = 1$.

2.5 Real-Time Data Acquisition

To ensure repeatability and analytical accuracy, error data were recorded in real-time during the testing phase[12][13]. The system utilized the Serial Communication feature of the Arduino Mega 2560 connected to a laptop. The error was calculated at each iteration using the following equation:

$$e(n) = Setpoint - Jarak_{aktual} \tag{5}$$

The data were transmitted via the Serial Monitor in a tabulated format for subsequent processing into response graphs.

2.6 Differential Navigation Configuration

The PID output was used to modulate the speed of the left and right DC motors via a motor driver to produce corrective movements. If the robot is too close to the wall (positive error), the PID output decreases the speed of the outer motor while increasing the inner motor speed to steer the robotaway, and vice versa. The PWM programming logic is defined as:

$$PWM_{kiri} = BaseSpeed + Output_{PID} \tag{6}$$

$$PWM_{kanan} = BaseSpeed - Output_{PID} \tag{7}$$

2.7 Testing and Data Collection

Performance evaluation was conducted under two scenarios: a linear path and a dynamic trajectory (90° turn). Testing was performed with a fixed setpoint of 10 cm. Success indicators were measured based on the robot's ability to minimize error (error convergence) and the frequency of collisions during changes in path geometry.

3. RESULT AND DISCUSSION

3.1 Error Calculation Analysis

The accuracy of the control system is highly dependent on the precision of the error calculations performed by the microcontroller during each iteration cycle (loop). Mathematically, the error calculation is defined as follows:

$$e(n) = Setpoint - Actual\_Distance \quad (8)$$

3.2 Alalysis of System Response to K_p Variation (K_i= 0, K_d = 0)

Initial testing was conducted by varying the proportional constant (K_p) from 1 to 10, while K_i and K_d parameters were set to zero. The objective of this test was to observe the influence of proportional control on the system's response characteristics.

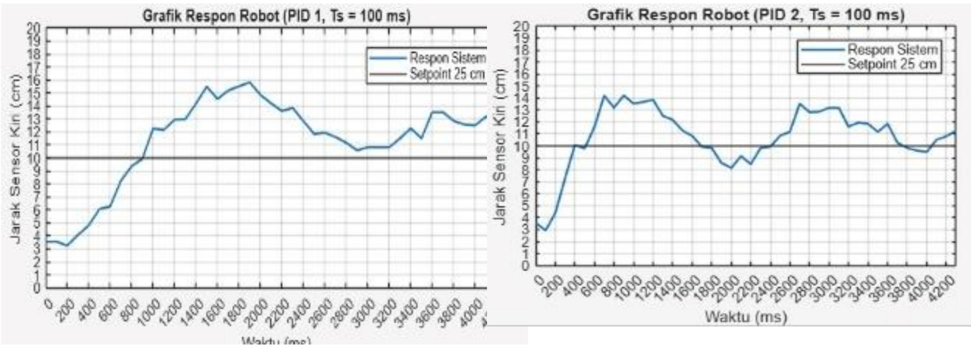

Figure 2. Response Graphs for $K_p = 1$ and $K_p = 2$

[Source: Author, 2026]

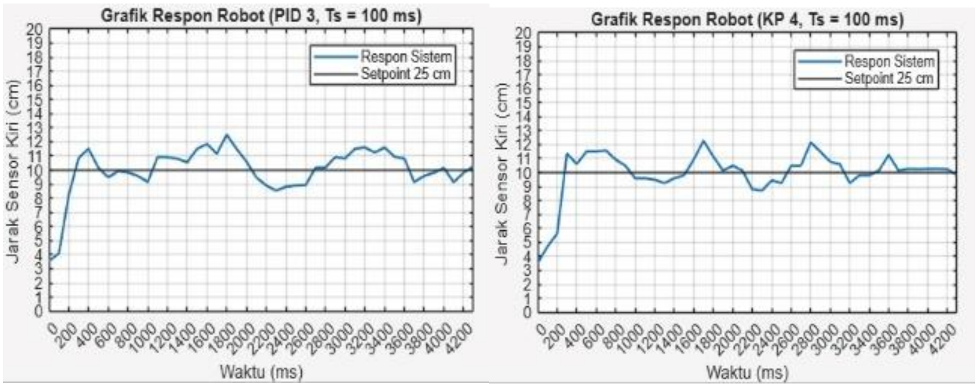

Figure 3. Response Graphs for $K_p = 3$ and $K_p = 4$

[Source: Author, 2026]

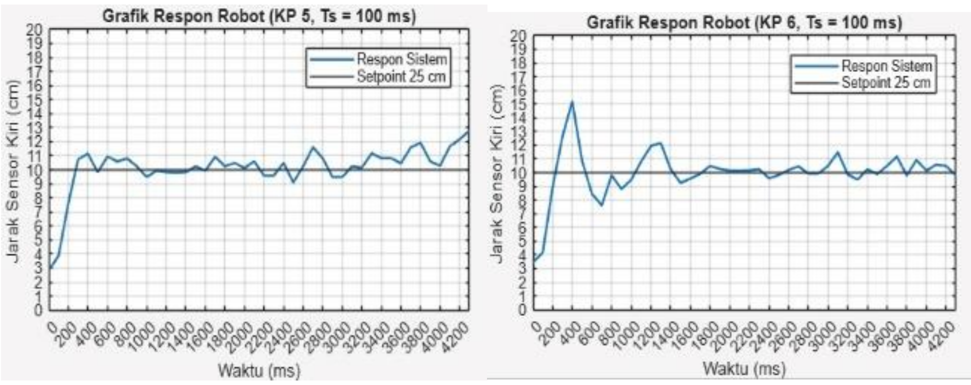

Figure 4. Response Graphs for $K_p = 5$ and $K_p = 6$

[Source: Author, 2026]

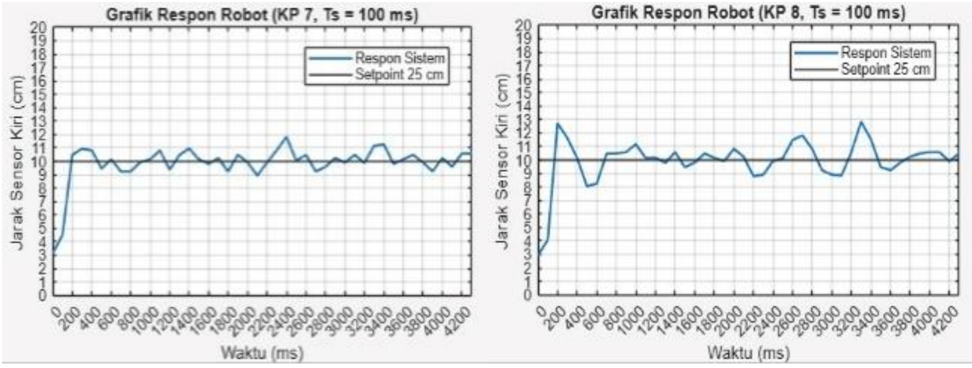

Figure 5. Response Graphs for $\boldsymbol{K_p = 7}$ and $\boldsymbol{K_p = 8}$
[Source: Author, 2026]

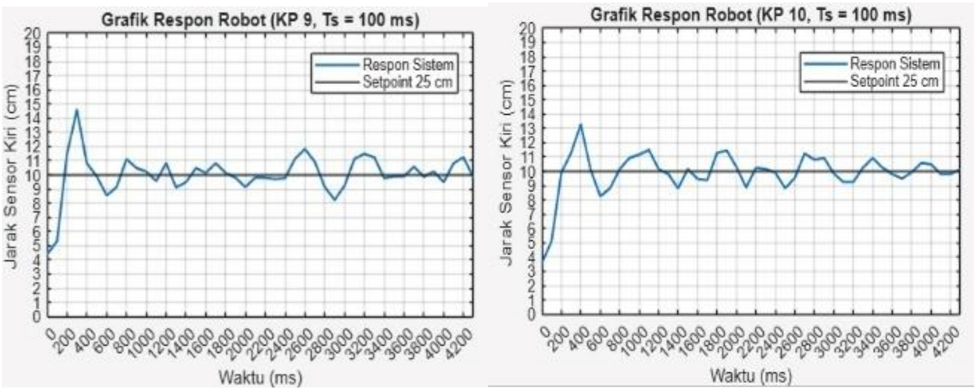

Figure 6. Response Graphs for $\boldsymbol{K_p = 9}$ and $\boldsymbol{K_p = 10}$
[Source: Author, 2026]

Analysis:

- Low K_p values (1–4): The system response tended to be sluggish, taking a significant amount of time to reach the setpoint.
- Moderate K_p values (5–7): The system began to demonstrate faster response times with manageable overshoot.
- High K_p values (8–10): The response became very rapid; however, it induced significant oscillations and overshoot.

These results indicate that increasing K_p accelerates the system response but potentially compromises stability if the value is excessively high.

3.3 Analysis of Graph Characteristics and Overshoot

Visualization of the system response relative to the setpoint revealed significant overshoot phenomena under dynamic trajectory conditions. The response graphs indicate that on linear paths, the robot achieves a steady-state condition in a short duration without meaningful deviation.

However, when the robot encounters sharp turns, a spike in transient error occurs, leading to overshoot. Mechanically, the inability of the actuators (DC motors) to execute instantaneous PWM changes due to inertia causes the robot to exceed the safe distance threshold before the derivative component can stabilize the position.

3.4 Analysis of Real-Time Error Graphs

Real-time error graphs were obtained through data acquisition using serial communication between the microcontroller and the computer.

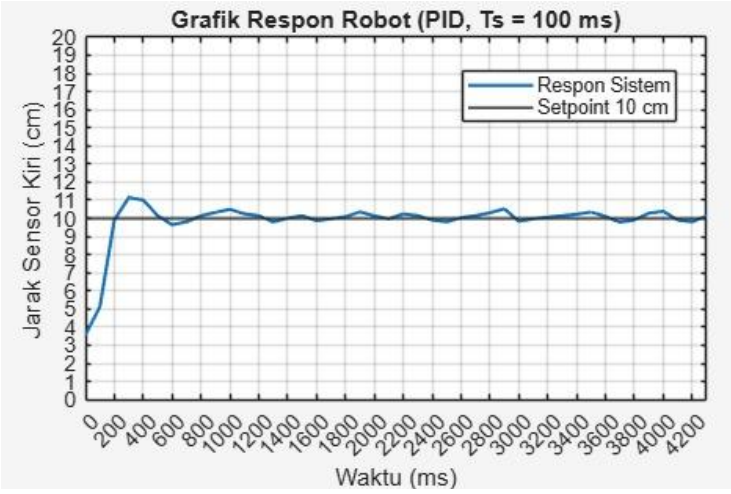

Figure 7 Final PID Test Results Graph
[Source: Author, 2026]

Based on the graph, two primary characteristics were identified:

- Steady-State Error Elimination: The graph shows the error value approaching zero on linear paths, indicating the success of the integral component (K_i) in eliminating static error.
- Transient Error and Overshoot: Spikes in the graph represent the system's response to disturbances or path changes. The derivative component (K_d) plays a role in damping these spikes.

3.5 Experimental Results on Linear and Dynamic Trajectories

On linear trajectories, the PID control system demonstrated stable performance with high precision. The robot successfully maintained its distance from the wall within a range of 10.56 cm to 11.13 cm relative to a fixed 10 cm setpoint, with a Mean Absolute Error (MAE) of 0.88 cm and a deviation of approximately ±0.56 cm to ±1.13 cm. Furthermore, all tests showed zero collisions, indicating that the system can respond to errors smoothly under constant path conditions.

The precision achieved in this study on linear paths shows a significant improvement compared to previous wall-following implementations. For instance, Iqbal et al [14] reported that while PID control effectively smooths robot movement, it still faces challenges in maintaining very tight distance tolerances on complex arenas. Our results demonstrate that with the specific Ziegler-Nichols tuning of $K_p = 2, K_i = 0.5$, and $K_d = 1$, the steady-state error can be further minimized.

Furthermore, when compared to advanced control strategies such as the Fuzzy Tuned PID Controller[15] and Fractional-order PID[16], our conventional PID system remains highly competitive for linear trajectories. While [15] and [16] offer superior adaptability in dynamic path tracking, our system provides a lower computational burden for the Arduino Mega 2560 platform while maintaining a precision level that is sufficient for rescue robot prototypes. However, the 1–2 collisions observed on 90° turns in this study align with the findings in[16], which suggest that standard PID controllers possess inherent limitations in handling non-linear geometric changes compared to higher-order control methods.

Overall, the combination of K_p, K_i, and K_d parameters produced a fast and stable response on linear paths. The integral component effectively reduced steady-state error, while the derivative component helped dampen fluctuations, although it remained sub-optimal in handling rapid error changes on dynamic trajectories due to the mechanical inertia of the 1.6 kg chassis and sensor latency.

Table 2: Linear Path Test Results (Setpoint 10 cm)
[Source: Author, 2026]

Trial	Collisions	Average Distance (cm)	Result
1	0	11.06	Success (Collision-free)
2	0	10.98	Success (Collision-free)
3	0	10.56	Success (Collision-free)
4	0	11.13	Success (Collision-free)
5	0	10.88	Success (Collision-free)

Table 3: 90° Turn Path Test Results (Setpoint 10 cm)
[Source: Author, 2026]

Trial	Collisions	Average Distance (cm)	Result
1	2	12,10	Completed (Collision)
2	1	11,75	Completed (Collision)
3	1	12,40	Completed (Collision)
4	0	11,90	Success (Collision-free)
5	0	11,25	Success (Collision-free)

4. CONCLUSION

The implementation of the PID algorithm in the wall-following robot navigation system has been proven to significantly improve response stability compared to conventional control systems. Through the configuration of optimal parameters $K_p = 2$, $K_i = 0{,}5$, and $K_d = 1$, the system successfully maintains the reference distance relative to the wall with high precision on linear trajectories, exhibiting a minimal average deviation of ± 0.56 cm to ± 1.13 cm. Error data analysis demonstrates that the integration of the integral component plays a crucial role in eliminating steady-state error, thereby ensuring the convergence of the robot's position exactly at the desired setpoint for this specific 1.6 kg hardware configuration.

Performance degradation on dynamic trajectories was identified as a consequence of high transient errors that exceeded the system's mechanical response capabilities. Although the PID controller effectively dampens oscillations, the increased collision frequency (1–2 times) during sharp 90° turns indicates limitations in the derivative action's ability to compensate for sensor data acquisition latency and the robot's mass inertia. These findings confirm that for environments with complex geometries, the conventional PID control scheme possesses inherent limitations in anticipating non-linear changes in path angles.

ACKNOWLEDGMENTS

The authors would like to express their deepest gratitude to the Electronics Engineering Technology Study Program, Department of Information Technology, Politeknik Negeri Lampung, for providing the necessary laboratory facilities, technical resources, and academic environment that made this research possible. We also extend our sincere appreciation to the faculty members and colleagues who provided insightful guidance and support throughout the development of this autonomous robot prototype. Their contribution was instrumental in the successful completion of this study.

REFERENCES

[1] R. Ramesh, D. Selvalakshmi, M. Naven Karthick, S. L. Nikileshkumar, S. Cibivarshan, and K. Prasanth, "Design and Implementation of Autonomous Maze-Solving Robot based on an Enhanced Maze Solving Techniques," *2nd Int. Conf. Autom. Comput. Renew. Syst. ICACRS 2023 - Proc.*, pp. 1892–1897, 2023, doi: 10.1109/ICACRS58579.2023.10405127.

[2] M. H. Iqbal and W. S. Aji, "Implementasi wall following dengan sistem kendali pid pada prototype robot rescue," *Semin. Nas. Tek. Elektro, Sist. Informasi, dan Tek. Inform.*, pp. 129–134, 2021, [Online]. Available: https://ejurnal.itats.ac.id/snestik/article/view/1770/1644

[3] M. Muharrom, "Perancangan Robot Pemadam Api dengan Pengontrolan Gerak Metode Proportional Integral Derivative (PID) Menggunakan Sensor Sonar Berbasis Mikrokontroller," *J. Inform. Dan Teknol. Komput.*, vol. 1, no. 3, pp. 1–14, 2022, [Online]. Available: https://researchhub.id/index.php/jitek/article/view/83/58

[4] Z. Li, Z. Tu, M. Zhang, Y. Lu, and H. Qiao, "Brain-Inspired Visual Language Navigation Robot Position Deviation Correction," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)* , vol. 15209 LNAI, pp. 273–287, 2025, doi: 10.1007/978-981-96-0789-1_20.

[5] M. Amirul Ummah F. A. B, "Rancang Bangun Robot Penghindar Halangan Dengan Metode PID," 2023, *Universitas 17 Agustus 1945 Surabaya, Surabaya*. doi: https://doi.org/10.55606/jtmei.v2i3.2352.

[6] H. Sihombing, E. Setiawan, and S. R. Akbar, "Sistem Kendali Robot Beroda Wall-Follower pada Tembok Lurus dan Berbelok menggunakan Metode PID," *J. Pengemb. Teknol. Inf. dan Ilmu Komput.*, vol. 6, no. 11, pp. 5129–5138, 2022, [Online]. Available: http://j-ptiik.ub.ac.id

[7] F. Ahmad Fauzi, E. Mulyana, R. Mardiati, and A. Eko Setiawan, "Fuzzy Logic Control for Avoiding Static Obstacle in Autonomous Vehicle Robot," *Proceeding 2021 7th Int. Conf. Wirel. Telemat. ICWT 2021*, 2021, doi: 10.1109/ICWT52862.2021.9678436.

[8] M. Sobirin and A. S. Hidayat, "Rancang Bangun Robot Wall Follower Dengan Kendali PID Berbasis Sensor Ultrasonik," *J. Kaji. Tek. Elektro*, vol. 10, no. 1, pp. 79–87, 2025, doi: 10.52447/jkte.v10i1.7839.
[9] M. S. Alam, A. I. Gullu, and A. Gunes, "Fiducial Markers and Particle Filter Based Localization and Navigation Framework for an Autonomous Mobile Robot," *SN Comput. Sci.*, vol. 5, no. 6, 2024, doi: 10.1007/s42979-024-03090-y.
[10] A. Nurul, T. Winarno, and A. Komarudin, "Sistem Navigasi Robot Berkaki Menggunakan Sensor Lidar Dengan Metode PID," 2021, *jurnal.polinema.ac.id*. doi: 10.33795/elk.v8i1.234.
[11] A. P. Afflin and E. Utami, "Tuning Control Aktual Temperatur Pada Reboiler Menggunakan Metode Ziegler-Nichols," pp. 1288–1297, 2022.
[12] A. M. M. Sizkouhi, M. Rahimifard, and R. R. Selmic, "Covert Attack and Detection Through Deep Neural Network on Vision-Based Navigation Systems of Multi-Agent Autonomous Vehicles," in *2022 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 2022, pp. 2583–2590. doi: 10.1109/SMC53654.2022.9945210.
[13] H. Meng, S. Zhang, W. Zhang, and Y. Ren, "Optimizing actual PID control for walking quadruped soft robots using genetic algorithms," 2024, *nature.com*. doi: 10.1038/s41598-024-77100-7.
[14] M. H. Iqbal and W. S. Aji, "Wall Following Control System with PID Control and Ultrasonic Sensor for KRAI 2018 Robot," *Int. J. Robot. Control Syst.*, vol. 1, no. 1, pp. 1–14, 2021, doi: 10.31763/ijrcs.v1i1.206.
[15] M. R. Razali, A. A. M. Faudzi, and A. U. Shamsudin, "Position and Angular control using Fuzzy Tuned PID Controller for Mobile Robot Path Tracking," *2022 IEEE 5th Int. Symp. Robot. Manuf. Autom. ROMA 2022*, 2022, doi: 10.1109/ROMA55875.2022.9915684.
[16] J. Du, "Design of Fractional-order PID Controller for Path Tracking of Wheeled Mobile Robot," 2021. doi: 10.1109/CAC53003.2021.9727828.