Motion System of a Four-Wheeled Robot Using a PID Controller Based on MPU and Rotary Encoder Sensors

Muhamad Rian Sagita [1], Alfian Ma'arif [2,*], Furizal [3], Chokri Rekik [4], Wahyu Caesarendra [5], Rania Majdoubi [6]
[1, 2] Department of Electrical Engineering, Universitas Ahmad Dahlan, Yogyakarta, Indonesia
[3] Department of Informatics Engineering, Universitas Islam Riau, Pekanbaru, Indonesia
[4] Control and Energy Management Lab (CEM Lab), University of Sfax, Sfax Engineering School, Tunisia
[5] Faculty of Mechanical Engineering, Opole University of Technology, 76 Proszkowska St, 45-758, Opole, Poland
[5] Faculty of Integrated Technologies, Universiti Brunei Darussalam, Jalan Tungku Link, Gadong, BE1410, Brunei Darussalam
[6] SETIME Laboratory, Faculty of Sciences, Ibn Tofail University, Kenitra, Morocco
Email: [1] muhamad1700022080@webmail.uad.ac.id, [2] alfianmaarif@ee.uad.ac.id, [3] furizal.id@gmail.com, [4] chokri.rekik@enis.rnu.tn, [5] wahyu.caesarendra@ubd.edu.bn, [6] rania_majdoubi@um5.ac.ma
*Corresponding Author

***Abstract*—This research addresses the challenge of developing an effective motion system for a four-wheeled omnidirectional robot configured with wheels at a 45-degree angle, allowing for holonomic movement—motion in any direction without changing orientation. In this system, inverse kinematics calculates each wheel's angular velocity to optimize movement. PID control is implemented to stabilize motor speeds, while odometry guides and determines the robot's position using initial and target coordinates. The robot operates on a 12-volt power supply and two STM32F103C microcontrollers, utilizing an MPU6050 sensor to maintain orientation and optical rotary encoders for accurate positional tracking. Experimental results demonstrate that the robot achieves optimal motion on x and y axes with PID settings of kP = 0.8, kI = 1.0, and kD = 0.08. This configuration yields a rise time of 0.95 seconds, overshoot of 7.36%, and steady-state error of -0.5 RPM at a setpoint of 350 RPM. Using odometry, the robot successfully navigates various movement patterns with average position errors of 1.2% on the x-axis and 1.6% on the y-axis for rectangular patterns, 2.1% on the x-axis and 2.2% on the y-axis for zig-zag patterns, and 1.75% on the x-axis and 1.15% on the y-axis for triangular patterns. The MPU6050 sensor maintains orientation with an error of 0.65% in triangular patterns and 0.85% in rectangular patterns. Through inverse kinematics, PID control, and sensor integration, the robot reliably follows designated coordinate points.**

Keywords*—*Four-Wheeled Robot Motion System, Omnidirectional Wheeled Robot, Inverse Kinematics, Odometry

I. INTRODUCTION

In the field of wheeled robots, there are several types of robots with different kinds of wheels [1]-[3]. One of them is the omnidirectional robot, this type of wheeled robot can move in any direction without changing its orientation, and is known as a holonomic robot [4]-[8]. One type of wheel that is used in this wheeled robot is the omnidirectional poly roller wheel, which allows the robot to perform complex movements and reach the desired positions [9].

In order for an omnidirectional wheeled robot to move in any direction or specific predetermined coordinates, kinematic modeling based on the wheel placement configuration is required [10], [11]. There are two types of robot configurations using omnidirectional wheels: one with three wheels and one with four wheels, each arranged at specific angles relative to the frame axis of the robot [12].

One of the challenges faced when using omnidirectional wheels is the robot's movement system [13]-[16]. In the configuration of an omnidirectional wheeled robot, whether with three wheels or four wheels, the speed of each drive actuator is not uniform [17]. Furthermore, to maximize the robot's movement, the robot's position must be known in advance so that it can move to the next position [18]-[20]. To accurately determine the movement and position of an omnidirectional wheeled robot, a sensor is needed that can estimate the change in the robot's position relative to its initial position.

By understanding the kinematic model to be used, the speed and direction of the wheel rotation can be easily determined. Rotary encoders are commonly used to determine the actual position of the robot and the speed generated by each drive actuator [10], [21]. Similarly, the MPU6050 sensor is used to maximize and correct changes in the robot's orientation, allowing the robot to maintain its position and heading with minimal error [22], [23].

Therefore, in this research, a motion system for a omnidirectional four-wheel robot will be developed using inverse kinematic modeling. To determine the robot's actual position, the odometry method is used with rotary encoders and the MPU6050 sensor, which functions to detect changes in wheel rotation, allowing the actual position of the robot to be known and maintaining the robot's position and heading. Additionally, PID control is applied to each drive actuator so that each actuator can maintain its speed [24], [25].

II. METHOD

A. Kinematics Modeling

Motion kinematics studies the movement of an object or particle without considering the causes that make the object move [26]. In robotics studies, kinematics plays an important role in controlling the movement of robots, whether they are

manipulator robots, humanoids, animals, or wheeled robots [27]-[30]. In robot kinematic modeling, there are two types of kinematic models: inverse kinematics and forward kinematics [31]-[35].

As shown in Fig. 1, in this research uses four omnidirectional wheels arranged on the robot's base frame to form an "X" pattern, or at an angle of 45° relative to the *x* and *y* axes. W1, W2, W3, and W4 represent the omnidirectional wheels arranged on the robot's base frame with inclination angles α1, α2, α3, and α4 relative to the *x* and *y* axes. To determine the robot's linear velocity with respect to (x, y, θ), inverse kinematics can be used to convert the input (x, y, θ) into the speed and direction of rotation for each wheel. Based on Fig. 1, we can formulate the Equation (1) and (2).

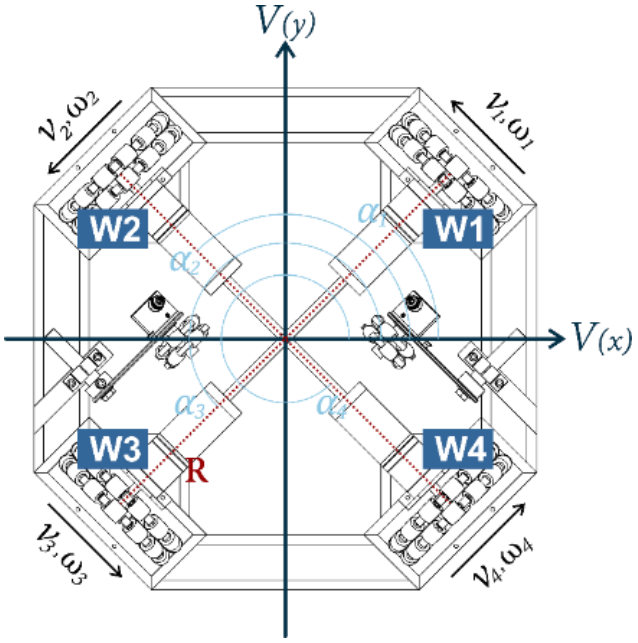


Fig. 1. Robot omnidirectional wheel placement

$$V_i = -\sin(\alpha_i)x + \cos(\alpha_i)y + R\theta \quad (1)$$

$$\begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{bmatrix} = \begin{bmatrix} -\sin\alpha_1 . V_x + \cos\alpha_1 . V_y + R . \theta \\ -\sin\alpha_2 . V_x + \cos\alpha_2 . V_y + R . \theta \\ -\sin\alpha_3 . V_x + \cos\alpha_3 . V_y + R . \theta \\ -\sin\alpha_4 . V_x + \cos\alpha_4 . V_y + R . \theta \end{bmatrix} \quad (2)$$

The robot's linear velocity is expressed in the vectors $V_{(x)}$ and $V_{(y)}$, which are the velocities along the x and y axes in the global reference frame. To determine the speed and direction of rotation for each wheel, the inverse kinematic model is used. This model utilizes the robot's position and orientation inputs (x, y, θ) to calculate the linear velocities v_i and angular velocities ω_i required for each wheel so that the robot can move according to the desired coordinates and orientation.

B. Odometry System

Odometry is a technique that uses data from actuator movements to estimate changes in coordinate positions over time. Odometry is used to estimate the coordinates of the position relative to the initial position [10], [36], [37].

A rotary encoder convert wheel rotations into distance measurement, each full rotation of the rotary encoder generates a number of pulses and can be converted into linear distance using Equation (3) and (4).

$$C = \pi . D \quad (3)$$

$$d = \frac{N . (\pi . D)}{ppr} \quad (4)$$

As in Equation (3) and (4) D is wheel circumference, ppr is pulse per revolution of rotary encoder, we can calculate the distance d from the number of pulses N. By utilizing rotary encoder sensors, odometry can calculate the distance traveled by each wheel and combine this information to determine the robot's position and orientation at any given time.

The rotary encoder sensors provide crucial data for continuously estimating the robot's changes in position (x, y) and the orientation (θ), allowing the control system to make necessary corrections to keep the robot on its desired path, based on Fig. 1, each wheel is have the rotary internal encoder and we can use (3)-(5) to determine the robot actual position in (x, y) and the orientation (θ).

$$x = \left(\frac{1}{2}\right)(r_4 - r_2)\cos 45° + \left(\frac{1}{2}\right)(r_1 - r_3)\sin 45° \quad (5)$$

$$y = \left(\frac{1}{2}\right)(r_4 - r_2)(-\sin 45°) + \left(\frac{1}{2}\right)(r_1 - r_3)\cos 45° \quad (6)$$

$$\theta = Robot\ orientation\ from\ MPU6050\ sensor \quad (7)$$

The odometry system with omnidirectional wheels can be used to determine the position and orientation of the robot in the *x* and *y* coordinates. As in Equation (5) and (6) r_1, r_2, r_3, r_4 is the distance reading from rotary encoder sensors. The use of a 45° angle simplifies the calculation of the wheel speed projection on the *x* and *y* axes. As shown in Fig. 2, re_1, re_2 represent traveled distance readings from two external rotary encoders placed at a 45° angle to the x and y axes. The external rotary encoders convert the wheel rotations into distance measurements in centimeters.

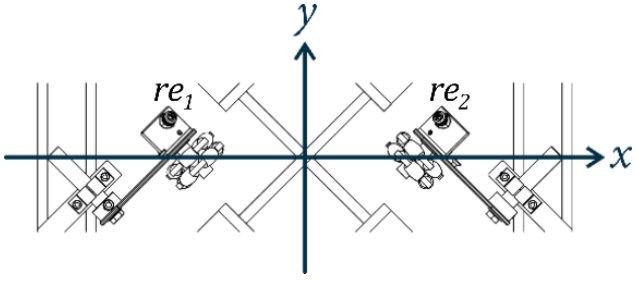


Fig. 2. External rotary encoder placement

The linear velocity components on the x and y axes can be calculated using sine and cosine functions to ensure the correct projection of wheel speed on each axis. This relationship is expressed in the Equation (8) and (9).

For robot's orientation (θ), is obtained from the output value of the MPU6050 sensor [22], [23], [38]. In this way,

the odometry system can estimate the robot's relative position to its initial position with higher accuracy, compensating for potential errors due to wheel slippage.

$$x = \left(\frac{1}{2}\right) re_1 . \cos 45° + \left(\frac{1}{2}\right) re_2 . \sin 45° \quad (8)$$

$$y = \left(\frac{1}{2}\right) re_1 . (-\sin 45°) + \left(\frac{1}{2}\right) re_2 . \cos 45° \quad (9)$$

C. PID Control

PID control is a highly effective and easy-to-implement control technique [39]-[41]. PID control consists of three main parameters: P (Proportional), I (Integral), and D (Derivative [42]-[45]). Each of these parameters has its own advantages and disadvantages.

In designing a PID control system, the trial and error method is often used due to the interdependence of the parameters Kp, Ki, and Kd [46]-[48]. To achieve optimal control action, a trial-and-error approach is required to combine P, I, and D until suitable values for Kp, Ki, and Kd are found [17], [49]. In this research, PID control is used to maintain the revolutions per minute (RPM) of each wheel at the predetermined set point. The formula for calculating the PID output is in Equation (10) [50]-[53].

$$u(t) = K_p e(t) + K_i \int e(t)\, dt + K_d \frac{de(t)}{dt} \quad (10)$$

Where $u(t)$ is the output of the PID controller, applied as a control signal to the wheel drive motor, and $e(t)$ is the difference between the set point RPM and the actual RPM obtained from the rotational speed measured by the rotary encoder that continuously calculated and used to generate control signal. The parameter K_p controls the impact of the current error, K_i controls the impact of the accumulated error, and K_d controls the impact of the rate of change of the error on the control output signal, by combining these three parameters, the PID control system can reduce the overall error and maintain the wheel's RPM at the desired value.

III. System Design

In this research, the movement system of the four-wheeled omnidirectional robot uses inverse kinematics with the odometry method to determine the robot's actual position. The MPU6050 sensor [54]-[56] is used as a reference for the robot's orientation, and PID control is used to correct the error generated by each wheel. This section will explain about the control system of the robot.

A. System Block Diagram

The system block diagram as shown in Fig. 3, shows the interaction between the main components in the control system of the omnidirectional four-wheel robot. The NRF24L01 module [57]-[59] is used for wireless communication between the robot and the PC, with data displayed via MATLAB. The STM32F103C8(1) microcontroller [60], [61] processes data from the MPU6050 sensor for robot orientation and manages communication with the NRF24L01 module. The STM32F103C8(2) microcontroller controls the PG45 DC motors based on signals from external and internal rotary encoders that measure wheel rotation and position.

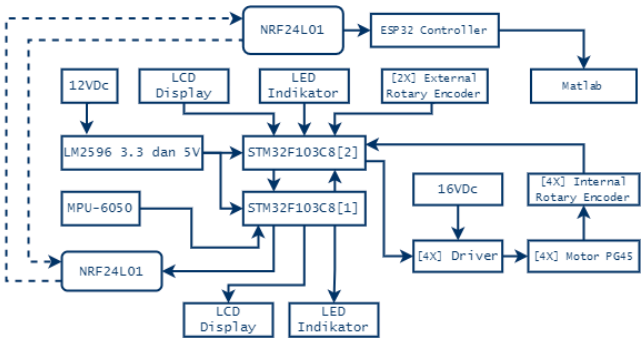


Fig. 3. Block diagram omnidirectional four-wheel robot

The system uses two main power supplies: 12VDC and 16VDC, with an LM2596 converter module providing 3.3V and 5V. The PG45 motors are controlled by drivers receiving signals from the STM32F103C8 [2]. An LCD display and LED indicators are used to show system information and provide visual feedback on the system's status.

B. Control System Diagran

The movement system of the four-wheeled omnidirectional robot can be seen in Fig. 4.

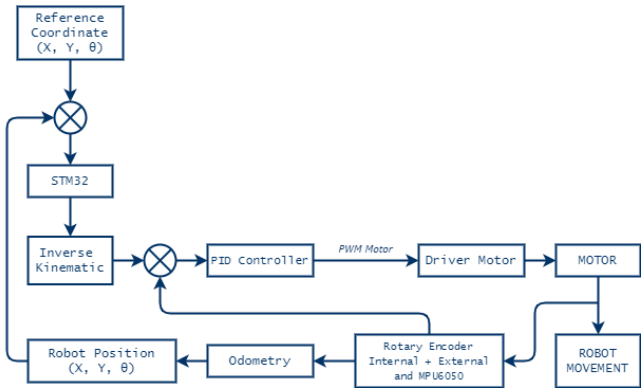


Fig. 4. System control block diagram

The control system diagram in Fig. 4, explains the workflow of controlling the four-wheeled omnidirectional robot. The system starts with the reference coordinate input (x, y, θ), which represents the desired position and orientation of the robot. This input is sent to the STM32 microcontroller, which then performs inverse kinematic calculations as in (1) and (2) to convert the reference coordinates into the speed and direction of each wheel, results of the inverse kinematic calculations are passed to the PID controller as in (8), which is responsible for correcting the error between the desired speed and the actual wheel speed. The control signal from the PID controller in the form of PWM is sent to the motor driver, which then drives the motor.

Internal and external rotary encoders, as well as the MPU6050 sensor, measure the actual position and orientation of the robot. This data is used by the odometry system to calculate the robot's actual position in coordinates (x, y, θ) as in (3)-(7). The actual position is then fed back to the STM32 microcontroller to form a feedback loop, ensuring that the robot moves according to the desired reference coordinates.

C. Control System Diagran

The four-wheeled omnidirectional robot and the actual implementation of this design, the 3D design includes the placement of components such as wheels, motors, and sensors on the robot frame, and can be shown in Fig. 5.

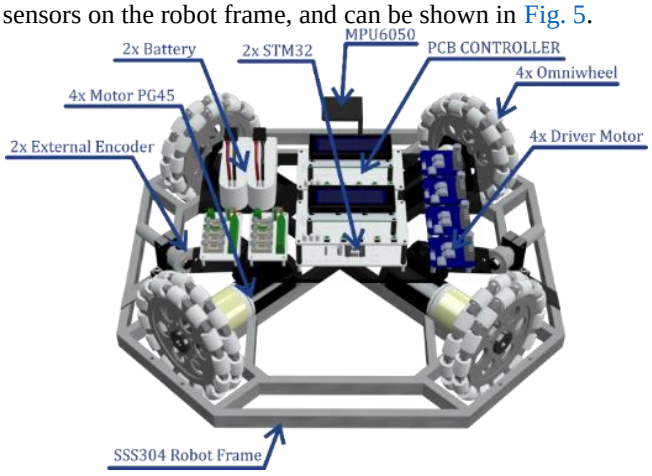


Fig. 5. Render 3D design four-wheeled omnidirectional robot

This visualization helps in understanding the physical layout of the implemented system and compares the planned design with the actual outcome. While the implementation from the 3D designs shows the actual assembly of each components can be shown in Fig. 6.

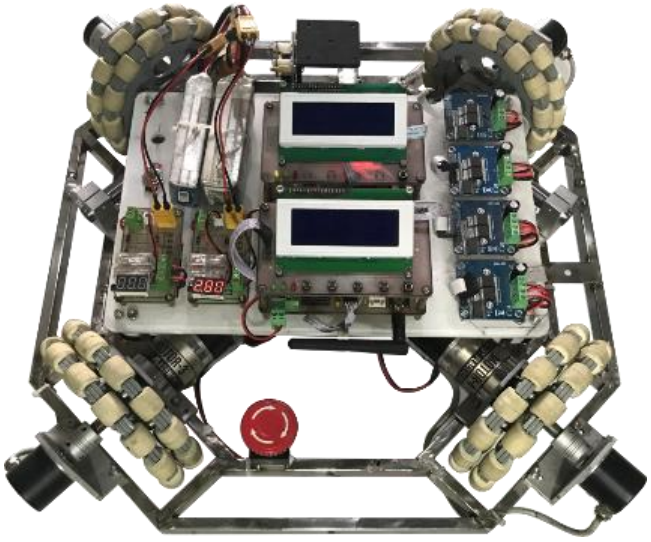


Fig. 6. Visualization omnidirectional four-wheel robot result and discussion

D. Sensor MPU6050 Test

After calibrating the MPU6050 sensor, tests were conducted by rotating the sensor on its axis to specific angle positions. Table 1 shows the results of testing at five different angle positions. The average error produced by the MPU6050 sensor is -0.972°. These results indicate that the MPU6050 sensor can be used to detect the orientation or heading of the robot with sufficient accuracy.

E. PID Control Response

The PID control testing was conducted using the trial and error method to find the parameters of K_p, K_p, and K_p that provide the best response in maintaining the motor speed at the set point [17], [47]. After several experiments, the best results were obtained with parameters $K_p = 0.65$, $K_i = 1.85$, and $K_d = 0.3$, as shown in Fig. 7.

Table 1. Sensor Result of MPU6050

Number	MPU6050 Angle (°)		
	Reference	Reading Result	Error
1	0	0.76	-0.76
2	45	45.80	-0.80
3	90	90.75	-0.75
4	135	136.25	-1.25
5	180	181.30	-1.30
Average Error			**-0.972**

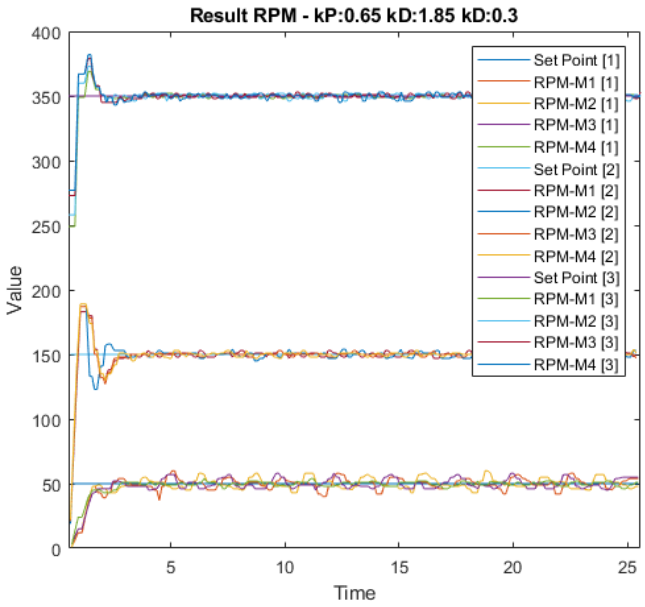


Fig. 7. PID control graph $K_p = 0.65$, $K_i = 1.85$, and $K_d = 0.3$

The analysis results show that PID control with parameters $K_p = 0.65$, $K_i = 1.85$, and $K_d = 0.3$, provides a good response at various target speeds. At the lower target speed (50 RPM), the settling time is relatively longer with moderate overshoot. However, at higher target speeds (150 RPM and 350 RPM), the system shows faster settling times and higher overshoot at 150 RPM compared to 350 RPM. The steady-state error is relatively small at all speeds, indicating the system's ability to maintain the desired speed with good accuracy.

- At the target speed of 50 RPM, the average rise time is 0.7 seconds, settling time is 25.2 seconds, peak time is 10.0 seconds, overshoot is 16.00%, and steady-state error is 0.8 RPM.
- At the target speed of 150 RPM, the average rise time is 1.0 seconds, settling time is 2.6 seconds, peak time is 1.0 seconds, overshoot is 24.67%, and steady-state error is -0.3 RPM.
- At the target speed of 350 RPM, the average rise time is 1.0 seconds, settling time is 1.6 seconds, peak time is 1.4 seconds, overshoot is 7.36%, and steady-state error is -0.5 RPM.

F. Omnidirectional Four-Wheel Robot Movement

The testing of the omnidirectional four-wheel robot's movement system aims to ensure that the entire system functions properly. The testing is conducted by providing variations in the reference coordinate inputs (x, y) and

creating specific patterns to determine whether the robot can move to the predetermined coordinate points and to observe the influence of the MPU6050 sensor in maintaining the robot's orientation.

1) *Movement on x axis with coordinate (200, 0)*

This test aims to evaluate the capability of the four-wheeled omnidirectional robot to reach predetermined coordinates at *x* axes. The plot result without and with the MPU6050 sensor can be seen in Fig. 8.

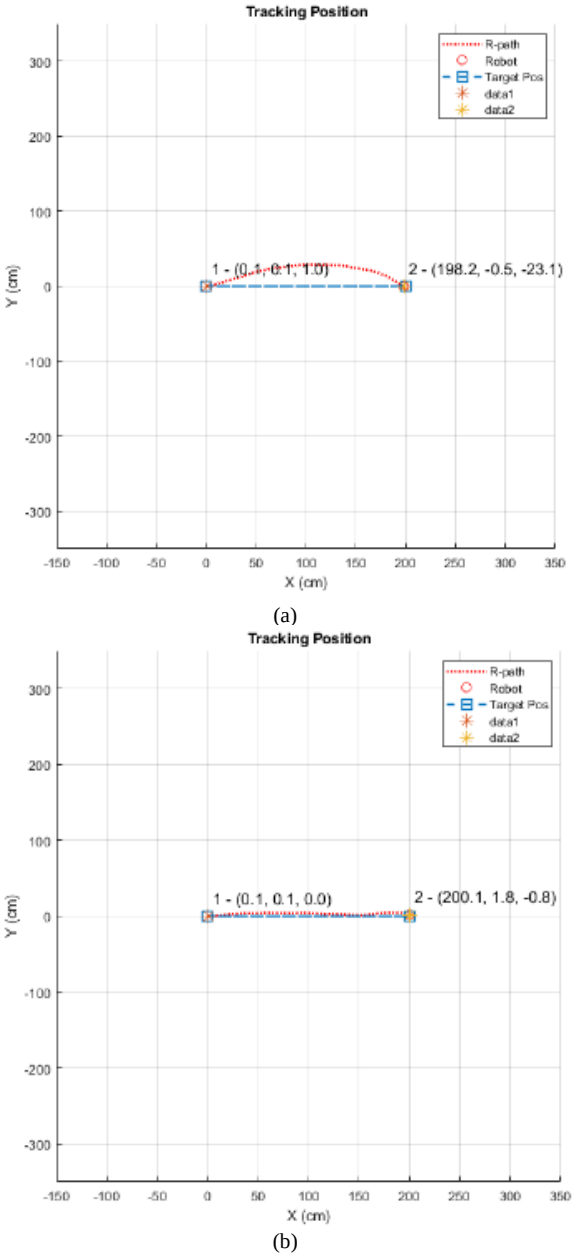


Fig. 8. Plot (a) without and (b) with MPU6050 sensor on *x*-axis (200, 0)

From Fig. 8 shows that the use of the MPU6050 sensor significantly reduces positional and orientation errors. In tests without the MPU6050, the positional error reached 1.8 cm on the *x*-axis and 23.08° in orientation. In contrast, with the MPU6050, the positional error reduced to -0.1 cm on the *x*-axis, -0.1 cm on the *y*-axis, and 0.78° in orientation. The plots with the red dashed line also indicate that the robot with the MPU6050 follows a path closer to the planned trajectory compared to without the MPU6050.

2) *Diagonal Movement with coordinate (200, 200)*

In the diagonal movement test, the robot was tested with and without the MPU6050 sensor. For the target coordinates (150, 150), the positional error in the test without the MPU6050 was 3.3 cm on the *x*-axis, -1.0 cm on the *y*-axis, and -0.55° in orientation. In contrast, with the MPU6050, the positional error reduced to 1.2 cm on the *x*-axis, 1.9 cm on the *y*-axis, and -1.17° in orientation. The plot with the red dashed line results also show that the robot with the MPU6050 follows a path closer to the planned trajectory compared to without the MPU6050. The Diagonal movement without and with the MPU6050 sensor shown in Fig. 9.

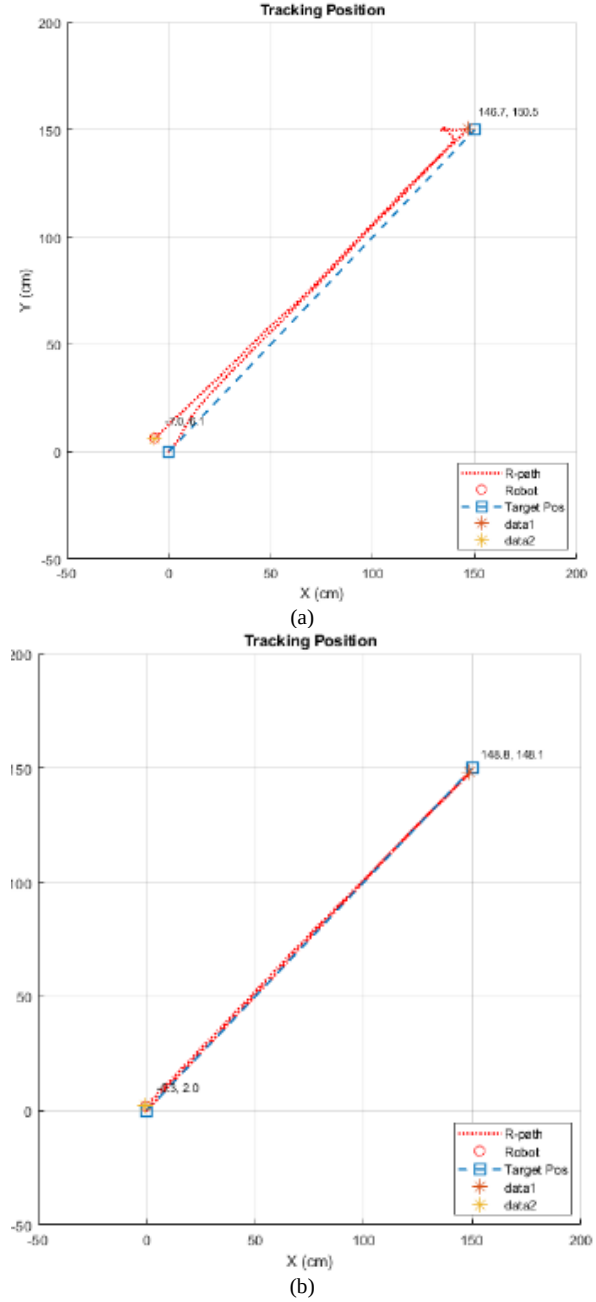
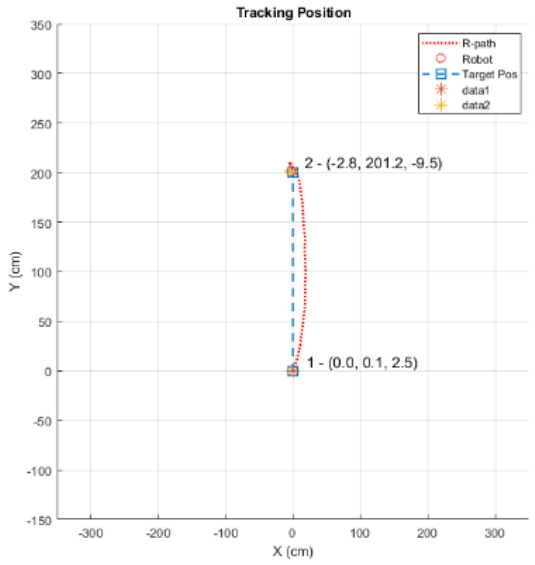


Fig. 9. Plot diagonal movement (a) without and (b) with MPU6050 sensor (200, 200)

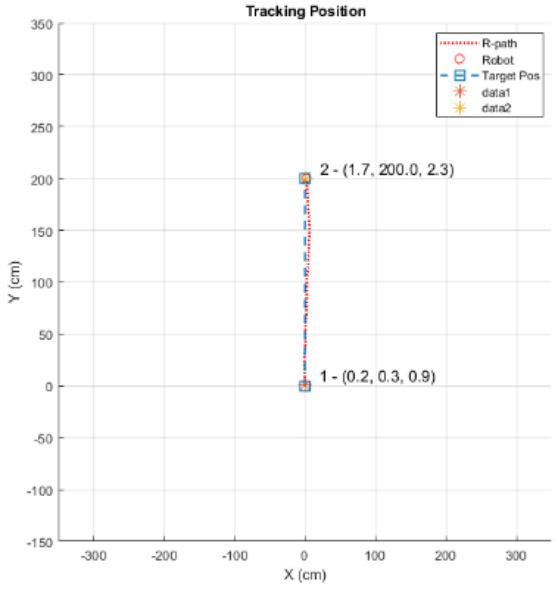
3) *Movement on y axis with coordinate (0, 200)*

In the y-axis movement test, the robot was tested with and without the MPU6050 sensor. For the target coordinates (0,

200), the positional error in the test without the MPU6050 was 2.8 cm on the *x*-axis, -1.2 cm on the *y*-axis, and 9.51° in orientation. In contrast, with the MPU6050, the positional error reduced to -1.7 cm on the *x*-axis, 0.0 cm on the *y*-axis, and 2.29° in orientation. The plot with red dashed line results also show that the robot with the MPU6050 follows a path closer to the planned trajectory compared to without the MPU6050. Fig. 10 a show result without using the MPU6050 sensor, ad Fig. 10 b show result with using the MPU6050.



(a)



(b)

Fig. 10. Plot (a) without and (b) with MPU6050 sensor on *y*-axis (0, 200)

Movement test with square pattern

This test aims to evaluate the capability of the four-wheeled omnidirectional robot to follow a predetermined rectangular path and observe the influence of the MPU6050 sensor in maintaining the robot's orientation. The result can be shown in Fig. 11. The test results show that the robot, aided by the MPU6050 sensor, can follow the specified rectangular path with a high degree of accuracy. From the table and plot, we can observe the positional and orientation errors as follows:

- At the initial position (0, 0), the positional error is -0.2 cm on the x-axis, 0.0 cm on the y-axis, and 0.47° in orientation.
- At data point 1 (200, 0), the positional error is 2.4 cm on the x-axis, 0.1 cm on the y-axis, and -1.56° in orientation.
- At data point 2 (200, 140), the positional error is 0.1 cm on the x-axis, 2.4 cm on the y-axis, and -0.33° in orientation.
- At data point 3 (0, 140), the positional error is -1.6 cm on the x-axis, 1.6 cm on the y-axis, and 0.95° in orientation.
- At data point 4 (0, 0), the positional error is 0.5 cm on the x-axis, 1.6 cm on the y-axis, and -1.92° in orientation.

Overall, the average error from the expected positions is approximately 1.2% for the x-axis, 1.6% for the y-axis, and 0.85% for orientation.

4) Movement test with triangle pattern

After this experiment to make a robot move in triangle pattern, results show that the robot, aided by the MPU6050 sensor, can follow the specified triangular path with a high degree of accuracy. From plot in Fig. 12, we can observe the positional and orientation errors as follows:

- At the initial position (0, 0), the positional error is -0.1 cm on the x-axis, -0.1 cm on the y-axis, and -0.42° in orientation.
- At data point 1 (150, 150), the positional error is -0.1 cm on the x-axis, 1.7 cm on the y-axis, and -0.24° in orientation.
- At data point 2 (150, 0), the positional error is 2.3 cm on the x-axis, 1.6 cm on the y-axis, and -0.79° in orientation.
- At data point 3 (0, 0), the positional error is 3.4 cm on the x-axis, -1.2 cm on the y-axis, and -1.21° in orientation.

Overall, the average error from the expected positions is approximately 1.75% for the x-axis, 1.15% for the y-axis, and 0.65% for orientation.

5) Movement test with triangle pattern

This test aims to evaluate the omnidirectional four-wheel robot ability to follow a predetermined zig-zag path and to observe robot ability to maintaining the robot's orientation. The test results show that the robot, aided by the MPU6050 sensor, can follow the specified zig-zag path with a high degree of accuracy. From the resulting plot shown in Fig. 13, we can observe the positional and orientation errors as follows:

- At the initial position (0, 0), the positional error is -0.1 cm on the x-axis, -0.1 cm on the y-axis, and -1.2° in orientation.
- At data point 1 (50, 50), the positional error is 1.2 cm on the x-axis, 2.0 cm on the y-axis, and -2.34° in orientation.
- At data point 2 (100, 0), the positional error is -1.3 cm on the x-axis, 4.1 cm on the y-axis, and -1.1° in orientation.
- At data point 3 (150, 50), the positional error is -3.3 cm on the x-axis, 2.9 cm on the y-axis, and -1.2° in orientation.
- At data point 4 (200, 0), the positional error is 1.8 cm on the x-axis, -1.7 cm on the y-axis, and 1.32° in orientation.
- At data point 5 (0, 0), the positional error is 3.7 cm on the x-axis, -0.2 cm on the y-axis, and 0.98° in orientation.

Overall, the average error from the expected positions is approximately 2.1% for the x-axis, 2.2% for the y-axis, and 1.5% for orientation.

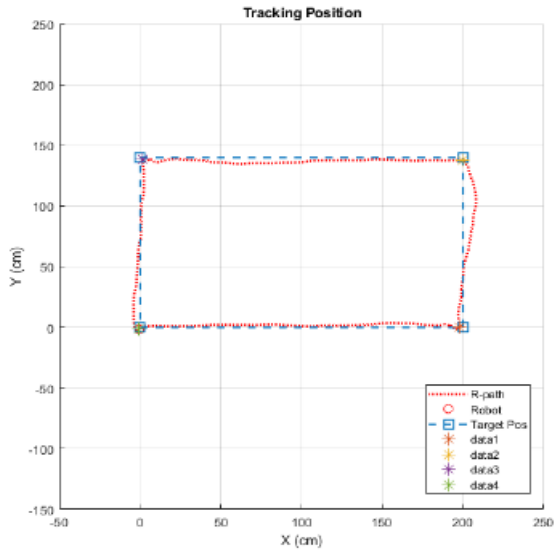


Fig. 11. Plot square pattern movement

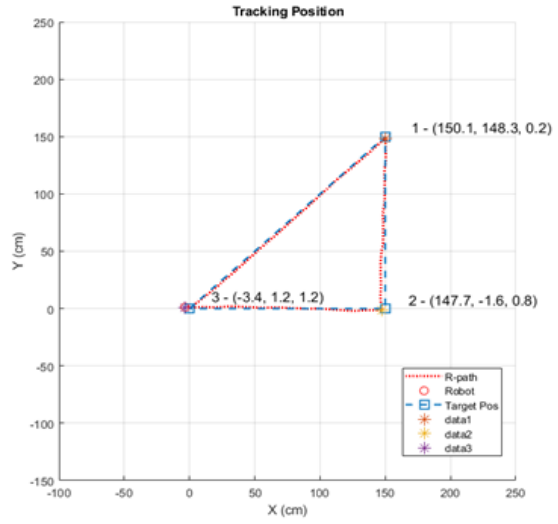


Fig. 12. Plot triangle pattern movement

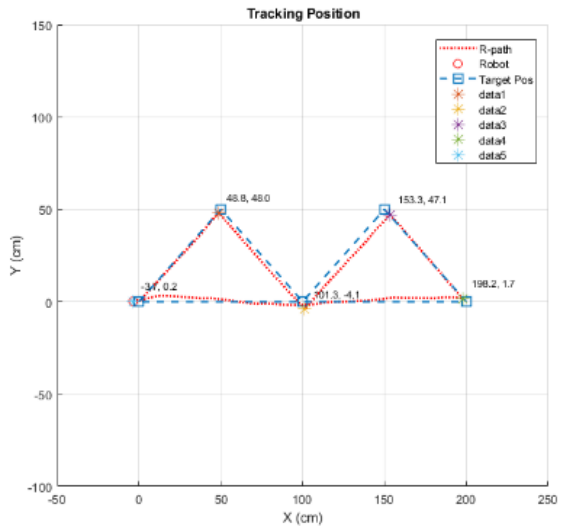


Fig. 13. Plot double triangle pattern movement

IV. CONCLUSION

After conducting several stages of testing in this study titled "Movement System of a Four-Wheeled Robot Using PID Controller Based on MPU and Rotary Encoder Sensors," it can be concluded that the robot is able to move using inverse kinematic to control the rotation direction and speed of each wheel on the x and y coordinates with an optimal PID control at $K_p = 0.65$, $K_i = 1.85$, and $K_d = 0.3$, which produces a rise time of 0.95 seconds, overshoot of 7.36%, and a steady-state error of -0.5 RPM at a set point of 350 RPM; the robot is capable of optimizing its movement towards specific coordinate points using odometry from the rotary encoder sensor and the MPU6050 sensor with an average error of 1.2% on the *x*-axis and 1.6% on the *y*-axis for the rectangular pattern, 2.1% on the *x*-axis and 2.2% on the *y*-axis for the zig-zag pattern, and 1.75% on the *x*-axis and 1.15% on the *y*-axis for the triangular pattern; the robot is able to maintain its orientation using the MPU6050 sensor with an average error in the triangular pattern of 0.65% and in the rectangular pattern of 0.85%; and by using inverse kinematic to determine the speed and direction of wheel rotation, combined with PID control, rotary encoder sensor, and MPU6050 sensor, the robot has successfully followed the predetermined reference coordinate points.

ACKNOWLEDGMENT

This research was assisted by Robotic Development Community Ahmad Dahlan University (RDC UAD) friends, especially the SAGOTRA team, and my research supervisor Mr. Ir. Alfian Ma'arif, S.T., M.Eng., I also thank you for the encouragement from friends and my parents in conducting this research.

REFERENCES

[1] M. Bjelonic, V. Klemm, J. Lee, and M. Hutter, “A Survey of Wheeled-Legged Robots,” *Robotics in Natural Setting*, pp. 83–94, 2023, https://doi.org/10.1007/978-3-031-15226-9_11.

[2] L. Tagliavini, G. Colucci, A. Botta, P. Cavallone, L. Baglieri, and G. Quaglia, “Wheeled Mobile Robots: State of the Art Overview and Kinematic Comparison Among Three Omnidirectional Locomotion Strategies,” *Journal of Intelligent & Robotic Systems*, vol. 106, no. 3, p. 57, 2022, https://doi.org/10.1007/s10846-022-01745-7.

[3] H. Taheri and C. X. Zhao, “Omnidirectional mobile robots, mechanisms and navigation approaches,” *Mechanism and Machine Theory*, vol. 153, p. 103958, 2020, https://doi.org/10.1016/j.mechmachtheory.2020.103958.

[4] M. Ben-Ari and F. Mondada, “Elements of robotics,” *Springer International Publishing*, 2018, https://doi.org/10.1007/978-3-319-62533-1.

[5] D. Rotondo, V. Puig, F. Nejjari and J. Romera, "A Fault-Hiding Approach for the Switching Quasi-LPV Fault-Tolerant Control of a Four-Wheeled Omnidirectional Mobile Robot," *IEEE Transactions on Industrial Electronics*, vol. 62, no. 6, pp. 3932-3944, 2015, https://doi.org/10.1109/TIE.2014.2367002.

[6] A. Krishnan and P. Sudarshan, "Self-Localization and Waypoints following of Holonomic Three Wheeled Omni-Directional Mobile Robot," *2021 IEEE International Conference on Distributed Computing, VLSI, Electrical Circuits and Robotics (DISCOVER)*, pp. 253-258, 2021, https://doi.org/10.1109/DISCOVER52564.2021.9663644.

[7] D. U. Rijalusalam and I. Iswanto, “Implementation Kinematics Modeling and Odometry of Four Omni Wheel Mobile Robot on The Trajectory Planning and Motion Control Based Microcontroller,” *Journal of Robotics and Control (JRC)*, vol. 2, no. 5, pp. 448-455, 2021, https://doi.org/10.18196/jrc.25121.

[8] E. Savaee and A. Rahmani Hanzaki, “A New Algorithm for Calibration of an Omni-Directional Wheeled Mobile Robot Based on Effective Kinematic Parameters Estimation,” *Journal of Intelligent & Robotic Systems*, vol. 101, no. 2, p. 28, 2021, https://doi.org/10.1007/s10846-020-01296-9.

[9] H. P. Oliveira, A. J. Sousa, A. P. Moreira and P. J. Costa, “Modeling and Assessing of Omni-Directional Robots with Three and Four Wheels,” *Contemporary Robotics - Challenges and Solutions*, 2009, https://doi.org/10.5772/7796.

[10] A. Sofwan, H. R. Mulyana, H. Afrisal and A. Goni, "Development of Omni-Wheeled Mobile Robot Based-on Inverse Kinematics and Odometry," *2019 6th International Conference on Information Technology, Computer and Electrical Engineering (ICITACEE)*, pp. 1-6, 2019, https://doi.org/10.1109/ICITACEE.2019.8904418.

[11] L. O. M. Idris, S. A. D. Prasetyowati, I. M. I. Subroto, “Forward and Inverse Kinematic on Wheeled Soccer Robot,” *Journal of Telematics and Informatics*, vol. 6, no. 4, pp. 258-268, 2019, http://dx.doi.org/10.12928/jti.v6i4.

[12] H. P. Oliveira, A. J. Sousa, A. P. Moreira, and P. J. Costa, “Dynamical models for omni-directional robots with 3 and 4 wheels,” *Proceedings of the Fifth International Conference on Informatics in Control, Automation and Robotics - Volume 2: ICINCO*, pp. 189-196, 2008, https://doi.org/10.5220/0001489201890196.

[13] R. T. Yunardi, D. Arifianto, F. Bachtiar, and J. I. Prananingrum, “Holonomic Implementation of Three Wheels Omnidirectional Mobile Robot using DC Motors,” *Journal of Robotics and Control (JRC)*, vol. 2, no. 2, pp. 65-71, 2021, https://doi.org/10.18196/jrc.2254.

[14] M. Fiedeń and J. Bałchanowski, “A Mobile Robot with Omnidirectional Tracks—Design and Experimental Research,” *Applied Sciences*, vol. 11, no. 24, p. 11778, 2021, https://doi.org/10.3390/app112411778.

[15] J. Palacín, D. Martínez, E. Rubies, and E. Clotet, “Suboptimal Omnidirectional Wheel Design and Implementation,” *Sensors*, vol. 21, no. 3, p. 865, 2021, https://doi.org/10.3390/s21030865.

[16] M. R. Azizi, A. Rastegarpanah, and R. Stolkin, “Motion Planning and Control of an Omnidirectional Mobile Robot in Dynamic Environments,” *Robotics*, vol. 10, no. 1, p. 48, 2021, https://doi.org/10.3390/robotics10010048.

[17] M. Kamaludin and W. S. Aji, “Manuver Robot Manual Menggunakan PID pada Robot Manual KRAI 2018,” *Buletin Ilmiah Sarjana Teknik Elektro*, vol. 1, no. 3, p. 91, 2019, https://doi.org/10.12928/biste.v1i3.978.

[18] M. B. Alatise and G. P. Hancke, "A Review on Challenges of Autonomous Mobile Robot and Sensor Fusion Methods," *IEEE Access*, vol. 8, pp. 39830-39846, 2020, https://doi.org/10.1109/ACCESS.2020.2975643.

[19] C. S. Tan, R. Mohd-Mokhtar and M. R. Arshad, "A Comprehensive Review of Coverage Path Planning in Robotics Using Classical and Heuristic Algorithms," *IEEE Access*, vol. 9, pp. 119310-119342, 2021, https://doi.org/10.1109/ACCESS.2021.3108177.

[20] P. K. Panigrahi and S. K. Bisoy, “Localization strategies for autonomous mobile robots: A review,” *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 8, pp. 6019–6039, 2022, https://doi.org/10.1016/j.jksuci.2021.02.015.

[21] K. H. Ng, C. F. Yeong, E. L. M. Su and A. R. Husain, "Implementation of cascade control for wheeled mobile robot straight path navigation," *2012 4th International Conference on Intelligent and Advanced Systems (ICIAS2012)*, pp. 503-506, 2012, https://doi.org/10.1109/ICIAS.2012.6306067.

[22] I. Rifajar and A. Fadlil, “The Path Direction Control System for Lanange Jagad Dance Robot Using the MPU6050 Gyroscope Sensor,” *International Journal of Robotics and Control Systems*, vol. 1, no. 1, pp. 27–40, 2021, https://doi.org/10.31763/ijrcs.v1i1.225.

[23] A. Febriawan and W. S. Aji, “Rotating Control on Robots Indonesian Abu Robot Contest with PID and IMUBNO055 Controls,” *Buletin Ilmiah Sarjana Teknik Elektro*, vol. 2, no. 1, p. 14, 2020, https://doi.org/10.12928/biste.v1i3.987.

[24] K. -C. Chang *et al*., "Shortest Distance Maze Solving Robot," *2020 IEEE International Conference on Artificial Intelligence and Information Systems (ICAIIS)*, pp. 283-286, 2020, https://doi.org/10.1109/ICAIIS49377.2020.9194913.

[25] J. -X. Zhang and G. -H. Yang, "Low-Complexity Tracking Control of Strict-Feedback Systems With Unknown Control Directions," *IEEE Transactions on Automatic Control*, vol. 64, no. 12, pp. 5175-5182, 2019, https://doi.org/10.1109/TAC.2019.2910738.

[26] I. Carlucho, M. D. Paula, G. G. Acosta, “Double Q-PID algorithm for mobile robot control,” *Expert Systems with Applications*, vol. 137, pp. 292-307, 2019, https://doi.org/10.1016/j.eswa.2019.06.066.

[27] Y. Zhao, J. Wang, G. Cao, Y. Yuan, X. Yao and L. Qi, "Intelligent Control of Multilegged Robot Smooth Motion: A Review," *IEEE Access*, vol. 11, pp. 86645-86685, 2023, https://doi.org/10.1109/ACCESS.2023.3304992.

[28] J. Carpentier and P.-B. Wieber, “Recent Progress in Legged Robots Locomotion Control,” *Current Robotics Reports*, vol. 2, no. 3, pp. 231–238, 2021, https://doi.org/10.1007/s43154-021-00059-0.

[29] T. Mikolajczyk *et al*., “Recent Advances in Bipedal Walking Robots: Review of Gait, Drive, Sensors and Control Systems,” *Sensors*, vol. 22, no. 12, p. 4440, 2022, https://doi.org/10.3390/s22124440.

[30] G. Singh and V. K. Banga, “Kinematics and trajectory planning analysis based on hybrid optimization algorithms for an industrial robotic manipulators,” *Soft computing*, vol. 26, no. 21, pp. 11339–11372, 2022, https://doi.org/10.1007/s00500-022-07423-y.

[31] I. Doroftei and B. Stirbu, “Design, modeling and control of an omni-directional mobile robot,” *Solid State Phenomena*, vol. 166–167, pp. 173–178, 2010, https://doi.org/10.4028/www.scientific.net/SSP.166-167.173.

[32] M. Hijikata, R. Miyagusuku, and K. Ozaki, “Wheel Arrangement of Four Omni Wheel Mobile Robot for Compactness,” *Applied Sciences*, vol. 12, no. 12, p. 5798, 2022, https://doi.org/10.3390/app12125798.

[33] H. Hadfield, L. Wei, and J. Lasenby, “The Forward and Inverse Kinematics of a Delta Robot,” *Advances in Computer Graphics*, pp. 447-458, 2020, https://doi.org/10.1007/978-3-030-61864-3_38.

[34] H. Ye, D. Wang, J. Wu, Y. Yue, and Y. Zhou, “Forward and inverse kinematics of a 5-DOF hybrid robot for composite material machining,” *Robotics and Computer-Integrated Manufacturing*, vol. 65, p. 101961, 2020, https://doi.org/10.1016/j.rcim.2020.101961.

[35] N. T. Dantam, “Robust and efficient forward, differential, and inverse kinematics using dual quaternions,” *The International Journal of Robotics Research*, vol. 40, no. 10-11, pp. 1087-1105, 2021, https://doi.org/10.1177/0278364920931948.

[36] Kok Seng Chong and L. Kleeman, "Accurate odometry and error modelling for a mobile robot," *Proceedings of International Conference on Robotics and Automation*, vol. 4, pp. 2783-2788, 1997, https://doi.org/10.1109/ROBOT.1997.606708.

[37] G. -S. Cai, H. -Y. Lin and S. -F. Kao, "Mobile Robot Localization using GPS, IMU and Visual Odometry," *2019 International Automatic Control Conference (CACS)*, pp. 1-6, 2019, https://doi.org/10.1109/CACS47674.2019.9024731.

[38] A. J. Moshayedi, A. Abbasi, L. Liao and S. Li, "Path planning and trajectroy tracking of a mobile robot using bio-inspired optimization algorithms and PID control," *2019 IEEE International Conference on Computational Intelligence and Virtual Environments for Measurement Systems and Applications (CIVEMSA)*, pp. 1-6, 2019, https://doi.org/10.1109/CIVEMSA45640.2019.9071596.

[39] 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, 2021, https://doi.org/10.1007/s40435-020-00665-4.

[40] S. A. Al-Samarraie and I. I. Gorial, “Assessment of FLC, PID, Nonlinear PID, and SMC Controllers for Level Stabilization in Mechatronic Systems,” *Journal of Robotics and Control (JRC)*, vol. 5, no. 6, pp. 1845–1861, 2024, https://doi.org/10.18196/jrc.v5i6.23639.

[41] J. Zhang and L. Guo, "Theory and Design of PID Controller for Nonlinear Uncertain Systems," *IEEE Control Systems Letters*, vol. 3, no. 3, pp. 643-648, 2019, https://doi.org/10.1109/LCSYS.2019.2915306.

[42] A. Ianni and N. Rossi, “SIR-PID: A Proportional–Integral–Derivative Controller for COVID-19 Outbreak Containment,” *Physics*, vol. 3, no. 3, pp. 459–472, 2021, https://doi.org/10.3390/physics3030031.

[43] A. Eltayeb, G. Ahmed, I. H. Imran, N. M. Alyazidi, and A. Abubaker, “Comparative Analysis: Fractional PID vs. PID Controllers for Robotic Arm Using Genetic Algorithm Optimization,” *Automation*, vol. 5, no. 3, pp. 230–245, 2024, https://doi.org/10.3390/automation5030014.

[44] R. C. Chourasia, N. Verma, P. Singh, and S. Kant, "Speed Control of DC Motor using PID Controller," *Global Journal of Technology and Optimization*, vol. 13, no. 1, pp. 1–4, 2022, https://www.hilarispublisher.com/open-access/speed-control-of-dc-motor-using-pid-controller-76415.html.

[45] F. F. Rahani and P. A. Rosyady, "Quadrotor Altitude Control using Recurrent Neural Network PID," *Buletin Ilmiah Sarjana Teknik Elektro*, vol. 5, no. 2, pp. 279–290, 2023, https://doi.org/10.12928/biste.v5i2.8455.

[46] O. Dogru *et al*., "Reinforcement learning approach to autonomous PID tuning," *Computers & Chemical Engineering*, vol. 161, p. 107760, 2022, https://doi.org/10.1016/j.compchemeng.2022.107760.

[47] A. Ma'Arif, H. Nabila, Iswanto, and O. Wahyunggoro, "Application of Intelligent Search Algorithms in Proportional-Integral-Derivative Control of Direct-Current Motor System," *Journal of Physics: Conference Series*, vol. 1373, no. 1, 2019, https://doi.org/10.1088/1742-6596/1373/1/012039.

[48] M. Bodson, "Speed Control for Doubly Fed Induction Motors With and Without Current Feedback," *IEEE Transactions on Control Systems Technology*, vol. 28, no. 3, pp. 898-907, 2020, https://doi.org/10.1109/TCST.2019.2898372.

[49] S. Balamurugan and A. Umarani, "Study of Discrete PID Controller for DC Motor Speed Control Using MATLAB," *2020 International Conference on Computing and Information Technology (ICCIT-1441)*, pp. 1-6, 2020, https://doi.org/10.1109/ICCIT-144147971.2020.9213780.

[50] M. A. Shamseldin, "Design of Auto-Tuning Nonlinear PID Tracking Speed Control for Electric Vehicle with Uncertainty Consideration," *World Electric Vehicle Journal*, vol. 14, no. 4, p. 78, 2023, https://doi.org/10.3390/wevj14040078.

[51] X. Miao, C. Hu, and Y. Qiao, "A Novel Two Variables PID Control Algorithm in Precision Clock Disciplining System," *Electronics*, vol. 13, no. 19, p. 3820, 2024, https://doi.org/10.3390/electronics13193820.

[52] Z. Bingul and K. Gul, "Intelligent-PID with PD Feedforward Trajectory Tracking Control of an Autonomous Underwater Vehicle," *Machines*, vol. 11, no. 2, p. 300, 2023, https://doi.org/10.3390/machines11020300.

[53] K. Mohamed, K. Sayed, and A. Kassem, "Analysis, Performance Improvement and Control Design of Heliostat System Based on Fractional Order PID Controller," *Sohag Engineering Journal*, 2024, https://doi.org/10.21608/sej.2024.275518.1056.

[54] S. A. Khomami and S. Shamekhi, "Persian sign language recognition using IMU and surface EMG sensors," *Measurement*, vol. 168, p. 108471, 2021, https://doi.org/10.1016/j.measurement.2020.108471.

[55] I. Miftahussalam, E. S. Julian, K. Prawiroredjo, and E. Djuana, "Wheelchair control system with hand movement using accelerometer sensor," *Microelectronic Engineering*, vol. 278, p. 112018, 2023, https://doi.org/10.1016/j.mee.2023.112018.

[56] A. A. Rafiq, W. N. Rohman, and S. D. Riyanto, "Development of a Simple and Low-cost Smartphone Gimbal with MPU-6050 Sensor," *Journal of Robotics and Control (JRC)*, vol. 1, no. 4, pp. 136-140, 2020, https://doi.org/10.18196/jrc.1428.

[57] L. R. K. and V. Vijayaraghavan, "A Self-Powered, Real-Time, NRF24L01 IoT-Based Cloud-Enabled Service for Smart Agriculture Decision-Making System," *Wireless Personal Communications*, vol. 124, no. 1, pp. 207–236, 2022, https://doi.org/10.1007/s11277-021-09462-4.

[58] W. Li, W. Wei and C. Chuixin, "Design of Wireless Wind Control Cooling System Based on nRF24L01," *2021 IEEE International Conference on Information Communication and Software Engineering (ICICSE)*, pp. 327-330, 2021, https://doi.org/10.1109/ICICSE52190.2021.9404144.

[59] Azhari, T. I. Nasution, S. H. Sinaga, and Sudiati, "Design of Monitoring System Temperature And Humidity Using DHT22 Sensor and NRF24L01 Based on Arduino," *Journal of Physics: Conference Series*, vol. 2421, no. 1, p. 012018, 2023, https://doi.org/10.1088/1742-6596/2421/1/012018.

[60] Y. Kang, Y. Xu, Y. Wang, Y. Wu, and Q. Tan, "Underground transient electromagnetic real-time imaging system for coal mine water disasters," *Measurement*, vol. 203, p. 111709, 2022, https://doi.org/10.1016/j.measurement.2022.111709.

[61] I. Yashan, I. Bekh, and S. Mamilov, "Developing hardware and software complex for monitoring the relative level of Co in the human exhalation," *AIP Conference Proceedings*, p. 090017, 2023, https://doi.org/10.1063/5.0148002.