ISSN 1693-9220 (Print) | 2987-8713 (Online), Volume 20, No. 2 (2026), pp. 212-224

International Journal of Enterprise Modelling

Published by: International Enterprise Integration Association

Implementation of a ROS 2-Based Differential Robot Drive System in a Robot Control System

Ivan Rivaldo Butarbutar[1], Roni Permana Saputra[2], Hendrana Tjahjadi[3]

[1,3] Department of Electrical Engineering, Faculty of Engineering and Defense Technology, Indonesian Defense University, Bogor, Indonesia

[2] Center for Intelligent Mechatronics Research, National Research and Innovation Agency, Bandung, Indonesia

Abstract

This study aims to develop and evaluate a ROS2-based differential-drive robot control system using a low-cost embedded robotic architecture consisting of a Raspberry Pi 4 and Arduino Nano. The proposed system redesigns the previous robot architecture by simplifying the embedded controller configuration from multiple microcontrollers into a single Arduino Nano integrated with dual independent BLDC motor drivers. The system utilizes ROS2 communication topics including /cmd_vel, /target_rpm, /wheel_rpm, and /odom to enable real-time motion control, wheel speed feedback, and odometry estimation. Motor control is implemented using a GP8503 DAC module and ZS-X11H motor drivers, while wheel RPM feedback is acquired using rotary encoders. Experimental results show that the robot is capable of performing real-time differential-drive motion and successfully generating real-time wheel RPM feedback and odometry information. However, deviations of approximately 40–45% between target RPM and actual RPM were observed due to the absence of PID-based wheel speed correction, wheel slip, and mechanical load imbalance and motor nonlinearity. This research demonstrates the successful integration of ROS2 middleware with a simplified low-cost embedded robotic architecture as a foundation for future autonomous navigation and localization development.

Corresponding Author:

Ivan Rivaldo Butarbutar,
Department of Electrical Engineering,
Faculty of Engineering and Defense Technology,
Indonesian Defense University,
Kawasan IPSC Sentul, Jl. Anyar, Sukahati, Kec. Citeureup,
Kabupaten Bogor, Jawa Barat 16810, Indonesia,
ivanrivaldo2702@gmail.com

Article Info

Article history:

Received : Apr 15, 2026
Revised : May 26, 2026
Accepted : May 28, 2026

Keywords:

Arduino Nano;
DAC GP8503;
Differential Drive Robot;
Odometry;
Raspberry Pi;
ROS2.

Introduction

Differential-drive mobile robots remain widely used due to their simple kinematic structure, low computational requirements, and effective maneuverability in autonomous navigation systems (Kumar & Lee, 2023; Zhang et al., 2022). This platform serves as the foundation for many autonomous and teleoperated robotic systems. ROS2 provides scalable DDS-based middleware communication suitable for distributed and real-time autonomous robotic applications (Garcia et al., 2023; Macenski et al., 2022).

Implementing a ROS2-based robotic control system requires seamless integration between software and hardware, including reliable communication between the main processor and embedded microcontrollers (Serineka et al., 2024). In this study, a Raspberry Pi 4 is used as the main processor, while an Arduino Nano serves as the motor controller and encoder processing unit (Serineka et al., 2024). Low-cost embedded platforms such as Raspberry Pi and Arduino are increasingly adopted in robotics research because of their flexibility, modularity, and affordability (Sarker, 2021).

The robot platform used in this research is CARRIE (Collaborative Autonomous Robot for Rugged Industrial Environment), a differential-drive mobile robot previously developed for robotic research applications. However, the previous system architecture relied on multiple microcontrollers and a single motor driver configuration, resulting in higher communication complexity and synchronization overhead. Therefore, this research redesigns the embedded architecture by simplifying the controller configuration into a single Arduino Nano integrated with dual independent BLDC motor drivers. This modification improves system modularity, simplifies hardware integration, and enables independent wheel control.

The robot is manually controlled using a joystick through ROS2 communication topics including /cmd_vel, /target_rpm, /wheel_rpm, and /odom. The developed system focuses on motor control response, embedded communication, wheel RPM feedback acquisition, and odometry generation without implementing autonomous navigation algorithms such as SLAM or obstacle avoidance (Sai Prasad Reddy et al., 2025).

The objective of this research is to develop a ROS2-based differential-drive robot control system integrated with joystick teleoperation, wheel speed monitoring, and odometry estimation (Sungkono, 2023). Unlike previous studies that mainly focused on teleoperation or basic motor actuation, this research integrates ROS2 middleware, DAC-based motor control, encoder feedback, and differential-drive odometry into a unified low-cost embedded robotic platform.

Encoder-based wheel speed feedback and PID-based motor control can improve motion accuracy and reduce steady-state velocity error in differential-drive robotic systems (Sharma et al., 2022; Yoon et al., 2023). A modular robot architecture simplifies system scalability and integration between sensing, control, and navigation modules (Koubaa, 2023).Sensor fusion between odometry and IMU measurements improves localization accuracy and motion stability in autonomous robotic systems (H. Kim et al., 2021). Modern SLAM algorithms enable autonomous robots to perform simultaneous localization and mapping with improved robustness in dynamic environments (Campos et al., 2021; Lv et al., 2023). Adaptive Monte Carlo Localization (AMCL) is widely implemented in ROS2-based navigation systems for probabilistic robot localization (Alwan et al., 2024; Singh et al., 2022).

Therefore, this research serves as an initial foundation for future autonomous robotic navigation development by providing a modular ROS2-based embedded control architecture capable of real-time wheel RPM monitoring and odometry estimation.

Methods

The process begins with a theoretical analysis to understand the operating principles and system requirements, followed by the creation of a block diagram illustrating the relationships between key components such as the joystick, Raspberry Pi, Arduino, GP8503 DAC, ZSX11H driver, BLDC motor, and encoder sensor. Once the system design was finalized, system programming was carried out to manage data communication from the joystick to the Raspberry Pi, command transmission to the Arduino via USB serial, and signal control to the DAC and motor driver.

The robot uses a differential-drive configuration with two independently controlled BLDC motors. Each BLDC motor is equipped with a rotary encoder with a resolution of 1200 pulses per

revolution (PPR), enabling high-resolution wheel RPM feedback acquisition and encoder-based odometry estimation within the ROS2 environment. The developed robot platform uses a wheel radius of 0.115 m and a wheelbase distance of 0.60 m for differential-drive motion calculations and odometry estimation. Serial communication between the Raspberry Pi 4 and Arduino Nano was configured using USB serial communication at 115200 baud. The control loop and encoder sampling processes were executed periodically to ensure stable real-time communication and wheel speed monitoring within the ROS2 environment.

The next stage is hardware implementation and system testing to ensure each component functions according to the design. If errors are found, a debugging process is carried out until the system works properly. Once the system runs stably, refinements and calibrations are performed to ensure that the motor response, encoder readings, and odometry data align with the target values in ROS2. The flow in this flowchart illustrates an iterative process, from design through testing, which is crucial for ensuring the robot control system functions optimally and adapts to changing field conditions.

Hardware

The hardware components of the CARRIE robot control system consist of several main parts that are interconnected and work in an integrated manner. The system design follows a modular robotics architecture that simplifies scalability and integration between sensing, control, and navigation modules (Koubaa, 2023). Arduino Nano functions as the embedded motor controller responsible for receiving wheel speed commands from the Raspberry Pi 4 through serial communication and generating motor control signals through the GP8503 DAC module. The controller also processes encoder pulse readings to obtain real-time wheel RPM feedback. The use of a single Arduino Nano simplifies communication synchronization compared to the previous multi-microcontroller architecture while reducing hardware complexity and improving modular integration. In addition, Arduino-based embedded systems are widely adopted in low-cost robotic applications due to their flexibility, ease of programming, compact form factor, and compatibility with various sensors and actuators (Sarker, 2021). ZS-X11H motor drivers are used to independently control the left and right BLDC motors of the differential-drive robot. Each driver receives analog voltage references generated by the GP8503 DAC module through the Arduino Nano controller. The use of dual independent motor drivers improves wheel velocity control stability and enables independent left-right wheel actuation, which is essential for differential-drive motion control and rotational maneuvering. Differential-drive mobile robots remain widely used due to their simple kinematic structure and effective maneuverability in autonomous navigation systems (Kumar & Lee, 2023; Zhang et al., 2022). Balanced wheel velocity control is important for maintaining stable linear motion in differential-drive robots, especially during rotational maneuvers and directional transitions (Hassan et al., 2024; Yoon et al., 2023). Compared to the previous robot architecture that used a single motor driver for both motors, this redesigned dual-driver configuration simplifies motor synchronization and improves independent wheel response during motion control. The GP8503 Digital-to-Analog Converter (DAC) module is used to generate precise analog voltage signals based on digital commands received from the Arduino Nano. In this system, the DAC output is connected to the ZS-X11H motor drivers to regulate motor speed continuously and smoothly. Unlike conventional PWM-based motor actuation, DAC-based control provides a more stable analog voltage reference and reduces signal ripple, resulting in smoother motor response and improved actuator stability.

The GP8503 module communicates with the Arduino Nano through the I2C interface and supports dual-channel analog output, allowing simultaneous control of the left and right motors. The use of DAC-based motor control can improve actuator response linearity and motor speed

stability in autonomous robotic systems (Lee et al., 2024; Wang et al., 2021). Each BLDC motor is equipped with a rotary encoder used as a wheel speed feedback sensor for real-time wheel RPM monitoring. Encoder pulse data is processed by the Arduino Nano to calculate the rotational speed of each wheel and is subsequently transmitted to ROS2 through serial communication. The wheel RPM information is utilized for differential-drive motion monitoring and odometry estimation. The encoder feedback mechanism enables the robot to estimate linear displacement and rotational orientation through dead-reckoning calculations. Encoder-based odometry is commonly used as a fundamental localization method in autonomous robotic systems and can later be integrated with additional sensors such as IMU and LiDAR for sensor fusion and SLAM applications (Campos et al., 2021; Lee et al., 2024; Lv et al., 2023; Zhou et al., 2021). The Raspberry Pi 4 acts as the main embedded computer responsible for executing the Robot Operating System 2 (ROS2) environment and managing communication between distributed robotic modules. Several ROS2 nodes are implemented on the Raspberry Pi 4, including teleoperation, differential-drive conversion, serial communication, wheel RPM feedback processing, and odometry estimation.

The Raspberry Pi communicates with the Arduino Nano through USB serial communication and processes joystick input data into linear and angular velocity commands. ROS2 provides scalable DDS-based middleware communication suitable for distributed and real-time autonomous robotic applications (Garcia et al., 2023; Macenski et al., 2022) .In addition, the Raspberry Pi 4 supports wireless communication and remote system monitoring, making it suitable for modular robotic development and future autonomous navigation implementation.

A wireless USB joystick is used as the manual teleoperation interface for robot motion control. The joystick is connected directly to the Raspberry Pi 4 and generates user input commands that are translated into linear velocity and angular velocity values within the ROS2 environment. These motion commands are published through the /cmd_vel topic and subsequently processed by the differential-drive control node. Joystick-based teleoperation is commonly utilized for robot motion validation, actuator testing, and embedded system verification before implementing autonomous navigation algorithms (Khan et al., 2023; Pereira et al., 2021). In this research, joystick teleoperation is used to evaluate the responsiveness of the differential-drive control system and real-time communication performance between ROS2 and the embedded controller.

The robot system is powered using a 24-volt battery supply that provides electrical power for the BLDC motors, motor drivers, Arduino Nano, DAC module, and Raspberry Pi 4. Stable voltage distribution is essential to maintain motor performance, sensor reliability, and communication stability during robot operation. The battery system is designed to support simultaneous operation of multiple embedded components and actuator loads while minimizing voltage fluctuations during dynamic motor acceleration and directional changes. Reliable embedded robotic systems require stable voltage regulation to ensure consistent sensor and controller performance during operation (Lee et al., 2024).

The physical hardware interconnection of the developed robotic system is illustrated in Figure 1. The schematic diagram presents the integration between the Raspberry Pi 4, Arduino Nano, GP8503 DAC module, ZS-X11H motor drivers, BLDC motors, rotary encoders, joystick input device, and power supply system used in the differential-drive robot platform.

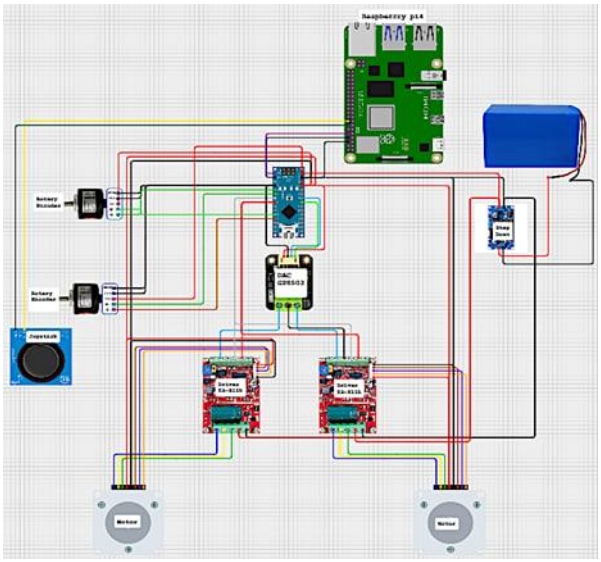

Figure 1. Hardware Interconnection and Differential-Drive Robot Control Schematic

As shown in Figure 1, the Raspberry Pi 4 functions as the primary embedded computer responsible for executing the ROS2 environment and processing joystick teleoperation commands. The Raspberry Pi communicates with the Arduino Nano through USB serial communication, where motion commands are transmitted in the form of target wheel RPM values.

The Arduino Nano acts as the embedded motor controller that processes the received target RPM data and generates analog voltage references through the GP8503 DAC module. These analog signals are used to control two independent ZS-X11H motor drivers connected to the left and right BLDC motors.

Each motor is equipped with a rotary encoder used to obtain real-time wheel RPM feedback. Encoder pulse data is processed by the Arduino Nano and transmitted back to ROS2 for wheel speed monitoring and odometry estimation. The entire robotic system is powered by a 24-volt battery supply that supports motor actuation and embedded control operation.

Figure 2. Developed CARRIE Differential-Drive Mobile Robot Platform

Figure 2 shows the developed CARRIE differential-drive mobile robot platform used in this research. The robot utilizes two independently controlled BLDC motors integrated with rotary encoders for wheel RPM feedback acquisition and odometry estimation. The embedded control

system consists of a Raspberry Pi 4, Arduino Nano, GP8503 DAC module, and dual ZS-X11H motor drivers integrated within a modular low-cost robotic architecture.

Software and ROS2 Integration

The software architecture of the CARRIE robot is developed using Robot Operating System 2 (ROS2) running on a Raspberry Pi 4 under the Linux operating system. ROS2 functions as middleware that manages communication between distributed robotic modules through a publisher-subscriber communication mechanism. ROS2 enables scalable and real-time communication between robotic subsystems using DDS-based middleware architecture (Garcia et al., 2023; Macenski et al., 2022).

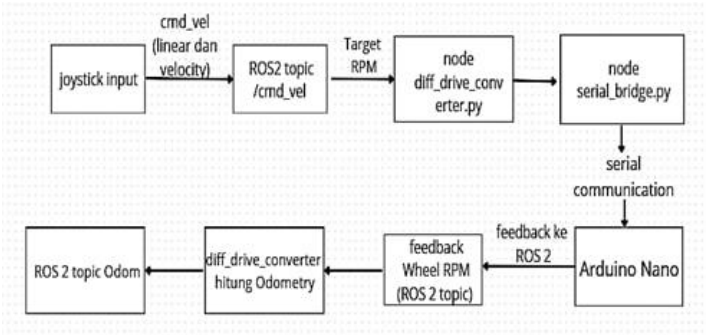

Figure 3. ROS2-Based Communication and Differential-Drive Control Architecture

Differential Drive Kinematics

The motion of the developed mobile robot is based on a differential-drive mechanism consisting of two independently controlled wheels mounted on the left and right sides of the robot platform. Robot motion is generated by controlling the angular velocity of each wheel independently, allowing the robot to move linearly, rotate, and perform turning maneuvers. Differential-drive robots are widely used in mobile robotic applications because of their simple kinematic structure and efficient maneuverability in constrained environments(Ferreira et al., 2023; Kumar & Lee, 2023).

In the developed system, linear velocity (v) and angular velocity (\omega) commands are obtained from the ROS2 topic /cmd_vel through joystick teleoperation. These velocity commands are converted into left and right wheel angular velocities using differential-drive inverse kinematics. The wheel angular velocities are expressed as:

$$\text{Left wheel angular velocity} \quad : \omega_L = \frac{v - \frac{L}{2}\omega}{r} \tag{1}$$

$$\text{Right wheel angular velocity: } \omega_R = \frac{v + \frac{L}{2}\omega}{r} \tag{2}$$

where (\omega_L) and (\omega_R) represent the left and right wheel angular velocities, (v) represents the linear velocity of the robot, (\omega) represents the robot angular velocity, (L) is the wheelbase distance between the left and right wheels, and (r) is the wheel radius. In the developed robot platform, the wheel radius is 0.115 m and the wheelbase distance is 0.60 m.

Odometry Estimation

Robot position and orientation estimation are performed using encoder-based odometry through a dead-reckoning approach. Rotary encoders mounted on each BLDC motor continuously provide wheel RPM feedback data, which is processed to estimate the robot displacement and heading over time. Odometry is one of the most commonly used localization methods in differential-

drive mobile robots because it can estimate robot motion without requiring external positioning systems (Batista et al.,2022; Ferreira et al., 2023).

The odometry estimation process calculates the robot position coordinates (x,y) and orientation (\theta) based on the linear velocity and angular velocity generated from encoder feedback. The robot pose update equations are expressed as follows:

$$xt + 1 = xt + vcos(\theta)\Delta t \quad (3)$$

$$y_{t+1} = y_t + vsin\ (\theta)\Delta t \quad (4)$$

$$\theta_{t+1} = \theta_t + \omega\Delta t \quad (5)$$

where (x) and (y) represent the robot position coordinates, (\theta) represents robot orientation, (v) represents linear velocity, (\omega) represents angular velocity, and (\Delta t) represents the sampling interval.

The calculated odometry information is published through the ROS2 topic /odom for robot motion monitoring and trajectory estimation. In this research, odometry is utilized primarily for wheel motion monitoring and position estimation during manual teleoperation. The generated odometry data can also serve as a foundational component for future implementation of autonomous navigation, localization, and SLAM algorithms integrated within the ROS2 ecosystem (Campos et al., 2021; Lv et al., 2023; Macenski et al., 2022).

Results and Discussion

The developed robot control system was successfully implemented through the integration of hardware and software based on the ROS2 framework. Communication between the Raspberry Pi 4 and Arduino Nano operated successfully through USB serial communication, enabling real-time transmission of motion commands and wheel RPM feedback within the embedded robotic system.

During the control process, joystick input commands were translated into linear and angular velocity values through the joy2twist.yaml node in ROS2. The generated motion commands were published through the /cmd_vel topic with a maximum linear velocity reference of 0.5 m/s. This velocity limit was intentionally configured to ensure stable teleoperation performance and safe real-time experimental operation during system testing.

The linear and angular velocity commands from /cmd_vel were subsequently converted into left and right wheel target RPM values through the differential-drive conversion node implemented within the ROS2 environment. The calculated target RPM values were transmitted to the Arduino Nano through serial communication, where the embedded controller generated analog motor control signals using the GP8503 DAC module to regulate the ZS-X11H motor drivers.

Simultaneously, the rotary encoders attached to each BLDC motor continuously measured the actual wheel rotational speed and transmitted wheel RPM feedback data back to ROS2 through the /wheel_rpm topic. The wheel RPM feedback was then processed by the odometry node to estimate robot position coordinates (x,y) and orientation (\theta), which were subsequently published through the /odom topic for real-time motion monitoring.

Experimental observations showed that the robot achieved a measured linear velocity of approximately 0.39 m/s with a very small angular velocity (angular.z), indicating relatively stable forward motion with minimal rotational deviation. This result demonstrates that the communication flow from /cmd_vel to target RPM generation, wheel RPM feedback acquisition, and odometry estimation operated consistently within the ROS2-based embedded robotic architecture.

Although wheel RPM feedback was continuously monitored using rotary encoders, the current implementation does not yet apply PID-based closed-loop wheel speed correction. Therefore, the encoder feedback mechanism primarily functions for wheel speed monitoring and odometry estimation rather than direct motor speed compensation. Previous studies have shown

that PID-based wheel speed control and encoder feedback can improve motion accuracy and reduce steady-state velocity error in differential-drive robotic systems (Sharma et al., 2022; Yoon et al., 2023).

System Analysis

DAC and Wheel RPM analysis

Table 1 presents the experimental results of the relationship between DAC output voltage, measured actuator voltage, and wheel rotational speed obtained from both tachometer measurements and encoder feedback through the ROS2 monitoring system. The experiments were conducted on the left and right BLDC motors to evaluate the consistency of wheel speed response within the differential-drive robotic platform.

Table 1. Comparison of DAC data with rpm measured manually and automatically

Dac Value (mV)	Manual		Automatic			
	V Out 1 (Left)	V Out 0 (Right)	Left Rpm	Right Rpm	Left Rpm	Rpm Right
0	0.02	0.03	0.00	0.00	0.00	0.00
250	0.25	0.26	6.27	7.04	6.50	7.00
500	0.51	0.52	19.58	19.74	19.30	19.50
750	0.75	0.75	36.56	36.36	36.20	36.70
1,000	1.09	1.11	56.25	56.41	56.50	56.80
1,250	1.25	1.26	74.16	74.71	73.80	73.40
1,500	1.51	1.51	90.75	89.90	90.60	90.40
1,750	1.76	1.77	106.6	106.90	104.8	104.10
2,000	2.1	2.15	120.0	120.90	118.4	117.90
2,250	2.26	2.26	128.0	128.70	128.1	128.20
2,500	2.51	2.52	140.7	138.60	139.1	138.00
2,750	2.76	2.76	148.9	148.10	147.3	147.10
3,000	3.00	3.05	156.8	156.60	155.5	156.50
3,250	3.25	3.25	168.20	166.30	164.0	165.20
3,500	3.50	3.51	176.1	173.10	173.3	174.60
3,750	3.75	3.75	185.9	184.90	181.5	184.70
4,000	4.00	4.00	195.9	194.80	186.6	190.20

As shown in Table 1, the measured DAC output voltage increased proportionally with the configured DAC setpoint values. When the DAC setpoint was configured at 0 mV, the measured output voltage was approximately 0.02–0.03 V, while a DAC setpoint of 4000 mV produced an output voltage of approximately 4.00 V. This proportional increase indicates that the GP8503 DAC module and ZS-X11H motor driver responded consistently to the generated analog control signals. Stable DAC-based actuation behavior is important for maintaining proportional motor response and motion consistency in embedded robotic systems (Lee et al., 2024; Wang et al., 2021).

The tachometer measurements and encoder feedback data obtained through the Serial Monitor also exhibited highly similar response characteristics across the tested DAC range. For example, at a DAC value of 1000 mV, the tachometer measured wheel rotational speeds of approximately 56.25–56.41 RPM, while the encoder feedback values ranged from approximately 56.50–56.80 RPM. The relatively small deviation between the tachometer measurements and encoder feedback indicates that the rotary encoder system provides reliable real-time wheel RPM monitoring for differential-drive motion analysis and odometry estimation (Ahmed et al., 2022).

In addition, both left and right wheel RPM values demonstrated relatively balanced rotational characteristics throughout the experiments. Balanced wheel rotational speed is essential

for maintaining stable linear motion and minimizing directional deviation in differential-drive robotic systems (Choi et al., 2022; Rahman et al., 2024). The experimental results therefore demonstrate that the integration between the DAC module, motor drivers, BLDC motors, encoder sensors, and ROS2 communication framework operated consistently during real-time motor control testing.

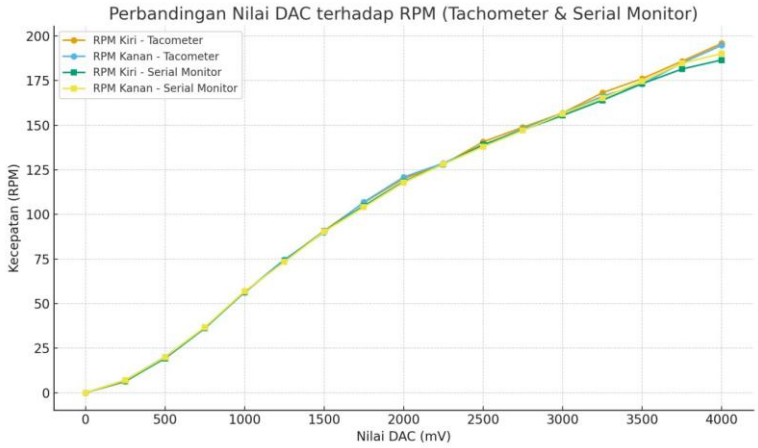

Figure 4. Comparison of DAC Voltage and Wheel RPM Measurements from Tachometer and Encoder Feedback

Figure 4 shows that both tachometer measurements and encoder feedback data follow a highly linear trend with respect to increasing DAC values. The proportional relationship between DAC voltage and wheel rotational speed indicates that the motor driver system responded consistently to variations in the analog control signal generated by the DAC module.

The RPM curves obtained from tachometer measurements and encoder feedback remain closely aligned across the tested operating range, with an average deviation of approximately ±2–5 RPM. This small deviation demonstrates that the encoder feedback system is sufficiently accurate for real-time wheel speed monitoring and odometry estimation within the ROS2-based robotic platform. Accurate wheel RPM feedback is important for maintaining reliable velocity estimation and Accurate wheel RPM feedback is important for maintaining reliable velocity estimation and improving future implementation of PID-based wheel speed control in mobile robotic systems (Patel, 2023).

Although minor differences between tachometer measurements and encoder feedback were observed at higher DAC values, the overall linear response trend indicates that the developed embedded motor control system operates consistently and reliably during real-time differential-drive motion control experiments.

Target RPM and Wheel RPM error analysis

The performance of the wheel RPM feedback system was evaluated by comparing the target wheel RPM generated from the ROS2 differential-drive conversion node with the actual wheel RPM measured through encoder feedback. Experimental measurements showed that the target wheel RPM value was approximately 44.56 RPM, while the measured wheel RPM obtained from the encoder feedback system was approximately 25.19 RPM.

The relative error between the target RPM and actual wheel RPM was calculated using the following equation:

$$Error\% = \frac{(T - A)}{T} \times 100\% \quad (6)$$

where (T) represents the target wheel RPM and (A) represents the actual wheel RPM measured through encoder feedback.

Substituting the experimental values into the equation yields:

$$\text{Error}(\%) = \frac{44.56 - 25.19}{44.56} \times 100\% \approx 43.5\% \quad (7)$$

The experimental results indicate that the measured wheel RPM achieved approximately 57% of the target RPM value, resulting in a relatively large deviation of approximately 43.5%. This deviation suggests the presence of scaling mismatch, actuator limitations, or mechanical losses within the embedded motor control system.

Several factors may contribute to the observed wheel RPM error, including the absence of PID-based closed-loop motor speed correction, nonlinear motor response characteristics, wheel slip, voltage drop under load conditions, and uncertainty in encoder scaling or differential-drive conversion parameters. Similar wheel velocity deviations are commonly observed in embedded differential-drive robotic systems that rely on open-loop motor actuation and encoder-based monitoring (Sharma et al., 2022; Yoon et al., 2023).

Although a relatively large wheel RPM deviation was observed, the experimental results demonstrate that the wheel RPM feedback system operated consistently and successfully provided real-time encoder feedback for wheel speed monitoring and odometry estimation within the ROS2-based robotic platform.

Cmd_vel and Odometry Velocity Analysis

The relationship between the commanded robot velocity (cmd_vel) and the actual robot motion estimated through odometry was evaluated to analyze the overall performance of the differential-drive control system. During the experiments, the maximum linear velocity command published through the ROS2 topic /cmd_vel was configured at 0.5 m/s using joystick teleoperation.

Experimental observations showed that the odometry system estimated an actual robot linear velocity of approximately 0.3 m/s during forward motion. The measured odometry velocity was therefore lower than the commanded velocity reference, indicating the presence of velocity tracking deviation within the embedded robotic control system.

The relative velocity error between the commanded velocity and the odometry-estimated velocity can be expressed as follows:

$$\text{Velocity Error}(\%) = \frac{v_{cmd} - v_{odom}}{v_{cmd}} \times 100\% \quad (8)$$

Substituting the experimental values into the equation yields:

$$\text{Velocity Error}(\%) = \frac{0.5 - 0.3}{0.5} \times 100\% = 40\% \quad (9)$$

The experimental results indicate a velocity deviation of approximately 40% between the commanded velocity and the odometry-estimated robot velocity. This deviation is consistent with the wheel RPM deviation discussed in the previous section, indicating that the observed error propagates from wheel actuation to odometry estimation within the overall differential-drive control system.

Several factors may contribute to the observed velocity deviation, including wheel slip, mechanical load imbalance, motor response nonlinearity, voltage drop under dynamic load conditions, and uncertainty in wheel radius calibration or encoder scaling parameters. In addition, the absence of PID-based closed-loop wheel velocity correction may reduce the ability of the system to maintain accurate wheel speed tracking during robot motion.

Despite the observed velocity deviation, the experimental results demonstrate that the ROS2-based communication system, wheel RPM feedback mechanism, and encoder-based odometry estimation operated consistently during real-time differential-drive robot operation. The

generated odometry information was successfully published through the /odom topic and can serve as a foundational component for future implementation of autonomous navigation and localization algorithms within the ROS2 ecosystem.

Evaluation of Results and System Limitations

The experimental results demonstrated that the developed ROS2-based differential-drive robotic system successfully achieved real-time motor actuation, wheel RPM monitoring, and encoder-based odometry estimation. The DAC output voltage exhibited a highly proportional relationship with wheel rotational speed, indicating consistent actuator response characteristics throughout the tested operating range. In addition, the wheel RPM measurements obtained from the external tachometer and encoder feedback system showed highly similar response trends, with only small deviations of approximately ±2–5 RPM across most DAC values. This result indicates that the encoder system provides sufficiently reliable real-time wheel RPM feedback for motion monitoring and odometry estimation within the embedded robotic platform.

Despite the observed linearity between DAC voltage and wheel rotational speed, significant deviations were identified between the target wheel RPM generated by the ROS2 differential-drive conversion node and the actual wheel RPM measured through encoder feedback. Experimental measurements showed that a target wheel RPM of approximately 44.56 RPM produced an actual wheel RPM of approximately 25.19 RPM, resulting in a relative error of approximately 43.5%.

The absolute error between the target and actual wheel RPM values is expressed as:

$$\text{Absolute Error} = T - A \tag{10}$$

Substitusi:

$$\text{Absolute Error} = 44.56 - 25.19 = 19.37 \text{ RPM}$$

The relative error calculation is expressed as:

$$\text{Relative Error}(\%) = \frac{T - A}{T} \times 100\% \tag{11}$$

Substitusi:

$$\text{Relative Error}(\%) = (44.56 - 25.19) / 44.56 \times 100\% \approx 43.5\%$$

A similar deviation was also observed between the commanded robot velocity and the odometry-estimated velocity. When the commanded linear velocity (v_{cmd}) was configured at 0.5 m/s, the odometry system estimated an actual robot velocity of approximately 0.3 m/s, corresponding to a relative velocity deviation of approximately 40%. The consistency between the wheel RPM deviation and odometry velocity deviation indicates that the observed error propagates throughout the overall control system, from wheel actuation to encoder-based odometry estimation. Several factors may contribute to the observed deviations, including scaling mismatch in target RPM conversion, uncertainty in encoder calibration parameters, wheel slip, nonlinear motor response characteristics, voltage drop under dynamic load conditions, and limitations in the open-loop motor actuation mechanism. Since the DAC voltage and wheel RPM relationship remained highly proportional throughout the experiments, the observed deviations are likely associated with conversion scaling parameters and actuator response limitations rather than instability in the DAC or encoder feedback system itself. Such deviations may reduce localization accuracy and negatively affect autonomous navigation and path-following performance in mobile robotic systems (Nguyen et al., 2021; Silva et al., 2023). Future system development may incorporate PID-based closed-loop motor speed correction and sensor fusion techniques integrating encoder odometry with IMU measurements to

improve motion accuracy, localization robustness, and overall navigation stability within the ROS2 robotic framework (D. Kim et al., 2024; Zhou et al., 2021).

Overall, the experimental results confirm that the developed ROS2-based embedded robotic architecture successfully achieved consistent real-time communication, differential-drive motion control, wheel RPM feedback acquisition, and odometry estimation, thereby providing a functional foundation for future autonomous robotic navigation research and development.

Conclusion

This research successfully developed and implemented a ROS2-based differential-drive mobile robot control system using a Raspberry Pi 4 and Arduino Nano within a low-cost embedded robotic architecture. The developed system successfully enabled real-time communication and motion control through ROS2 topics, including /cmd_vel, /wheel_rpm, and /odom. Experimental results demonstrated a proportional relationship between DAC output voltage and wheel rotational speed, while encoder feedback measurements showed only small deviations of approximately ±2–5 RPM compared with tachometer measurements, indicating reliable real-time wheel RPM feedback and stable differential-drive robot operation. Although the developed system successfully achieved real-time robot motion control and odometry estimation, deviations of approximately 40–45% were still observed between commanded and measured wheel velocities. These deviations were mainly caused by wheel slip, actuator limitations, calibration uncertainty, and the absence of PID-based closed-loop motor control. Nevertheless, the developed ROS2-based robotic platform provides a functional foundation for future implementation of PID-based wheel speed control, IMU sensor fusion, and autonomous navigation methods such as SLAM and AMCL.

References

Ahmed, S., Rahman, M., & Noor, F. (2022). Encoder Calibration for Accurate Mobile Robot Odometry. *Sensors*, *22*(18), 6912. https://doi.org/10.3390/s22186912

Alwan, H., Mahmood, A., & Kareem, Z. (2024). Improved AMCL Localization for Differential Drive Robots in Indoor Environments. *Sensors*, *24*(2), 611. https://doi.org/10.3390/s24020611

Batista, L. G., Salarolli, P. F., Gamarra, D. F. T., De Almeida, G. M., Vivacqua, R. P. D., & Cuadros, M. A. S. L. (2022). *Odometry and speed control of a 4WD mobile robot integrated with ROS 2*.

Campos, C., Elvira, R., Rodriguez, J., Montiel, J., & Tardos, J. (2021). ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual-Inertial, and Multi-Map SLAM. *IEEE Transactions on Robotics*, *37*(6), 1874–1890. https://doi.org/10.1109/TRO.2021.3075644

Choi, H., Kim, J., & Lee, S. (2022). Stability Analysis of Differential Drive Mobile Robots Under Wheel Velocity Disturbance. *Sensors*, *22*(11), 4211. https://doi.org/10.3390/s22114211

Ferreira, M., Moreira, L., & Lopes, A. (2023). Differential drive kinematics and odometry for a mobile robot using TwinCAT. *Electronic Research Archive*, *31*(4), 1789–1803. https://doi.org/10.3934/era.2023092

Garcia, M., Torres, J., & Perez, A. (2023). Real-Time Communication Performance Evaluation of ROS2 for Autonomous Robotic Systems. *Robotics and Autonomous Systems*, *163*, 104392. https://doi.org/10.1016/j.robot.2023.104392

Hassan, M., Rahman, T., & Islam, S. (2024). Wheel Odometry Error Compensation for Mobile Robots Using Adaptive Calibration. *IEEE Access*, *12*, 22114–22128. https://doi.org/10.1109/ACCESS.2024.3361182

Khan, M., Ahmed, R., & Islam, T. (2023). Real-Time Teleoperation of Differential Drive Robots Using ROS2. *IEEE Access*, *11*, 78122–78135. https://doi.org/10.1109/ACCESS.2023.3298712

Kim, D., Lee, J., & Park, Y. (2024). Real-Time Sensor Fusion for Differential Drive Robot Navigation Using Extended Kalman Filter. *IEEE Access*, *12*, 42011–42025. https://doi.org/10.1109/ACCESS.2024.3375011

Kim, H., Park, J., & Choi, S. (2021). Odometry Error Analysis for Differential Drive Mobile Robots. *Sensors*, *21*(14), 4823. https://doi.org/10.3390/s21144823

Koubaa, A. (2023). *Robot Operating System (ROS): The Complete Reference*. Springer.

Kumar, A., & Lee, S. (2023). Differential Drive Navigation and Motion Control for Autonomous Robots. *IEEE Access*, *11*, 55421–55435. https://doi.org/10.1109/ACCESS.2023.3278451

Lee, D., Kim, S., & Park, H. (2024). High-Efficiency BLDC Motor Control for Autonomous Mobile Robots. *Electronics*, *13*(1), 201. https://doi.org/10.3390/electronics13010201

Lv, X., Zhao, Y., & Wang, P. (2023). Recent Advances in SLAM for Autonomous Mobile Robots: A Review. *Robotics*, *12*(4), 98. https://doi.org/10.3390/robotics12040098

Macenski, S., Foote, T., Gerkey, B., Lalancette, C., & Woodall, W. (2022). Robot Operating System 2: Design, Architecture, and Uses in Autonomous Robots. *Science Robotics*, *7*(66), eabm6074. https://doi.org/10.1126/scirobotics.abm6074

Nguyen, T., Hoang, P., & Tran, D. (2021). Path-Following Control for Differential Drive Mobile Robots. *Robotics*, *10*(2), 56. https://doi.org/10.3390/robotics10020056

Patel, R. (2023). Real-Time Encoder Feedback for Robotic Motion Control. *IEEE Access*.

Pereira, L., Costa, P., & Silva, R. (2021). Teleoperation Framework for ROS2-Based Mobile Robots. *Robotics*, *10*(4), 112. https://doi.org/10.3390/robotics10040112

Rahman, M., Hasan, T., & Karim, A. (2024). Motion Stability Enhancement for Differential Drive Robots Using Adaptive Wheel Speed Compensation. *IEEE Access*, *12*, 51123–51138. https://doi.org/10.1109/ACCESS.2024.3381924

Sai Prasad Reddy, M., Pranav, K., Rahman, W., Vijay Kumar, B., Jain, A., & Gurulakshmi, A. B. (2025). Speed Control of BLDC Motor using PWM and Arduino Uno. *E3S Web of Conferences*, *619*. https://doi.org/10.1051/e3sconf/202561902007

Sarker, V. (2021). Low-Cost Mobile Robot Platform Using Raspberry Pi. *Robotics*.

Serineka, I. G. P., Piarsa, I. N., & Raharja, I. M. S. (2024). Design and Construction of an Arduino Nano-Based Robotic Arm. *Jurnal Syntax Admiration*, *5*(11), 4526–4532. https://jurnalsyntaxadmiration.com/index.php/jurnal/article/view/1625

Sharma, V., Gupta, R., & Mehta, A. (2022). Closed-Loop Speed Control for Differential Drive Mobile Robots Using PID Algorithms. *Sensors*, *22*(9), 3441. https://doi.org/10.3390/s22093441

Silva, J., Costa, A., & Lima, R. (2023). Autonomous Path Tracking for Differential Drive Robots Using ROS2. *IEEE Access*, *11*, 92122–92137. https://doi.org/10.1109/ACCESS.2023.3307711

Singh, R., Patel, D., & Kumar, P. (2022). ROS2-Based Adaptive Monte Carlo Localization for Autonomous Mobile Robots. *IEEE Access*, *10*, 113522–113535. https://doi.org/10.1109/ACCESS.2022.3214478

Sungkono, S. (2023). Improvement of Motion Accuracy in a Differential Drive Mobile Robot through Odometry-Compass Sensor Fusion Implementation. *ELKHA Jurnal Teknik Elektro*, *15*(1).

Wang, T., Li, Q., & Sun, Y. (2021). Performance Analysis of BLDC Motors for Mobile Robotic Applications. *IEEE Access*, *9*, 128331–128342. https://doi.org/10.1109/ACCESS.2021.3112274

Yoon, J., Park, M., & Choi, K. (2023). Real-Time Closed-Loop Velocity Control for Autonomous Mobile Robots. *IEEE Access*, *11*, 99112–99126. https://doi.org/10.1109/ACCESS.2023.3310842

Zhang, Y., Chen, X., & Li, J. (2022). Kinematic Modeling and Control of Differential Drive Mobile Robots. *Robotics and Autonomous Systems*, *152*, 104070. https://doi.org/10.1016/j.robot.2022.104070

Zhou, Y., Li, H., & Wang, Z. (2021). IMU and Odometry Sensor Fusion for Mobile Robot Localization. *Sensors*, *21*(6), 2145. https://doi.org/10.3390/s21062145