Differential Mobile Robot for Intelligent Storage Management through ArUco Detection with ROS 2

Jaime Andres Escobar Chuquimia
Mechatronics Engineering
Bolivian Catholic University
jaime.escobar@ucb.edu.bo

Daniel Luis Choque Nacho
Mechatronics Engineering
Bolivian Catholic University
luis.choque.n@ucb.edu.bo

Leyla Bionda Lipa Huari
Mechatronics Engineering
Bolivian Catholic University
leyla.lipa@ucb.edu.bo

***Abstract*—This paper presents the design of a differential mobile robotic system for an intelligent storage organization, using visual marker(Arucos) detection, onboard camera capable of acquiring RGB and depth images allowing for both visual identification and 3D mapping of the environment using SLAM, and autonomous navigation with ROS 2 use the things said on a Raspberry Pi 5. Sensors such as Kinect, MPU, and laser sensors are integrated for obstacle planning and avoidance, and techniques such as OpenCV and PID logic are used for control.**

I. Introduction

Nowadays, the automation of logistics processes represents a key challenge in improving the operational efficiency of warehouses and distribution centers. This project proposes the development of a differential mobile robot with autonomous navigation capabilities and intelligent detection of QR and barcodes, using ROS 2 as the main development environment. Through the integration of sensors such as the Kinect camera, distance sensors, and an inertial measurement unit (MPU), the system is capable of identifying, classifying, and relocating boxes within a simulated warehouse, optimizing space usage and reducing human intervention. This document details the design, implementation, and testing carried out on a Raspberry Pi 5-based platform, as well as an analysis of related technologies and adopted solutions from the state of the art.

A. Objectives

To develop an autonomous robotic system for storage organization, capable of identifying, classifying, and locating boxes labeled with QR or barcodes in designated locations within a warehouse, using ROS 2 and a Raspberry Pi 5 as the main platform.

II. Related Work

A. Article 1: Obstacle avoidance technique for mobile robots in collaborative human-autonomous robot warehouse environments

One of the main challenges addressed in the study is the safe navigation of mobile robots in warehouse environments shared with humans and constantly moving objects. The paper proposes a system that combines computer vision, depth sensors, and fuzzy logic to improve the robot's ability to detect and react intelligently to different types of obstacles, without compromising the environment's productivity.

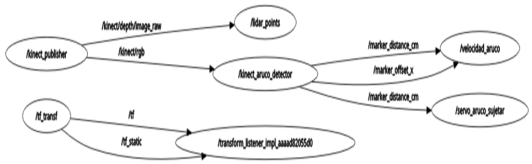

Figure 1. Diagrama de nodos

he system uses a Kinect camera that simultaneously provides color images and depth maps. The RGB images are processed by the YOLOv3 model to identify key objects such as people or boxes, while the depth information is used to construct a three-dimensional map of the environment. With these data, a node plans the robot's trajectory and generates the commands required for its movement. Subsequently, a fuzzy logic node analyzes the type of detected object and adjusts the robot's speed accordingly; for example, if a person is detected, it slows down to maintain a safe distance, thus ensuring safer and more adaptive navigation.
Valuable insights from this study can be applied to the development of the proposed project. The incorporation of fuzzy logic to adjust the robot's speed based on the distance and type of detected object is an efficient strategy for achieving safe navigation in dynamic environments. Additionally, the use of YOLOv3 is limited to classifying specific objects captured by the RGB camera, such as people or fragile equipment, allowing for the definition of differentiated safety zones. This enables the robot to adapt its behavior more intelligently and safely according to what lies ahead. Finally, ground filtering is highlighted as a useful technique to avoid false obstacle detections, improving the overall accuracy of the system.

B. Articulo 2: Path Planning of Mobile Robots Based on QR Code

In the industrial field, path planning for mobile robots has traditionally relied on methods such as line following, magnetic fields, or infrared sensors. Although functional, these approaches are often expensive, inflexible, and require specific physical infrastructure. To address these limitations, the study proposes a more economical and adaptable alternative: the use of QR codes distributed throughout the environment as reference points for autonomous navigation.

The solution involves organizing QR codes in a grid over the robot's workspace. Using a camera, the robot detects and decodes the codes to determine its exact location and plan its route accordingly. This strategy enables precise navigation without the need for complex sensors or additional infrastructure. The results demonstrated the effectiveness of the system in controlled environments, especially when implemented on the Kobuki platform with ROS and a Raspberry Pi, achieving reliable and low-cost autonomous movements.
This approach offers several opportunities for application in the proposed project. For example, QR codes could be placed on the warehouse floor to guide the robot to specific locations, complemented by a vision system on Raspberry Pi based on OpenCV for detection and decoding. Route planning can then be carried out using algorithms that interpret this information, enabling the determination of optimal paths with minimal investment. Key advantages include cost reduction by eliminating the need for advanced sensors, ease of maintenance in structured environments like warehouses, and the flexibility to modify routes simply by repositioning the QR codes.

C. Article 3: ROS 2 Robot With SLAM (Stewart Church, 2024)

Their experimentation highlights the importance of choosing lightweight SLAM backends when working with embedded platforms like the Raspberry Pi. Instead of relying on the full Nav2 stack, they opted for decoupled SLAM modules with reduced overhead, improving responsiveness and energy efficiency. This approach aligns with the requirements of your system, which also operates under hardware constraints. Additionally, their analysis of IMU–camera synchronization and drift correction is especially relevant for combining Kinect and MPU data in environments with variable lighting or motion blur. [1].

D. Article 4: Simultaneous Localization and Mapping for Warehouse Applications (Choudhary et al., 2024)

This work stands out for its practical contribution in real warehouse setups, where robots face repetitive layouts and occlusions due to shelving units. The authors evaluate the robustness of Cartographer in these constrained layouts and provide tuning parameters such as scan matching weights, motion filters, and adaptive voxel filtering. These findings can directly inform the setup of SLAM in your robotic system, especially when navigating between aisles or operating in partially mapped sections of the warehouse. [2]

E. Article 5: A Mobile Mapping System for ROS 2-based Autonomous Robots (VAULT Prototype, 2025)

The VAULT system exemplifies how sensor fusion and modular design can enhance both scalability and real-time performance. By distributing computational loads across composed ROS 2 nodes and selectively activating sensors depending on environmental context, the system achieves smooth indoor-outdoor transitions. While your project focuses on indoor environments, the architectural flexibility presented in this paper may be beneficial for extending functionality in larger warehouse layouts or future upgrades that include manipulator arms or inventory counting. [3]

F. Article 6: High-Precision Localization Using AMCL and QR Codes (Chen et al., 2025)

Chen et al. provide detailed results on the effectiveness of QR-assisted localization under varying conditions, such as lighting changes and sensor noise. Their evaluation shows that supplementing SLAM with strategically placed QR markers improves position correction and reduces cumulative drift over long distances. This insight could be applied to your system by placing QR markers at known coordinates along warehouse pathways, effectively anchoring your robot's estimated pose during long navigation tasks or when loop closure fails. [4]

G. Article 7: Autonomous Navigation with RL in ROS 2 (Kashyap Konathalapalli, 2025)

The authors not only train their agents in simulation but also validate them on a TurtleBot3 in real environments, observing better obstacle anticipation and smoother velocity transitions compared to traditional reactive planners. Although your robot currently uses PID and visual control, their findings support the feasibility of adding a learning-based layer in the future. This would allow your robot to adapt its behavior based on context (e.g., more cautious in high-traffic zones), potentially increasing both safety and efficiency in shared storage areas. [5]

III. Methodology

The process begins with perception, where the robot uses its front-facing camera (Kinect) to visually scan the boxes distributed throughout the warehouse. Using computer vision techniques implemented with OpenCV, the QR or barcode labels attached to each box are detected and interpreted.
In the classification stage, the content of the code is analyzed to identify the type of product (e.g., "cookies" or "beverages") and assign it to a specific, predefined storage zone within the warehouse.
Next, the system performs planning and navigation. A map of the environment is generated in real time using SLAM (Simultaneous Localization and Mapping) techniques, leveraging the depth information provided by the Kinect sensor. Based on this map, the optimal path to the target zone is calculated using algorithms such as A* or Wavefront, ensuring an efficient and collision-free trajectory.
During the manipulation and relocation phase, the robot aligns itself with the box using visual servoing (visual control), ensuring precise positioning before moving it to its newly assigned location within the warehouse.

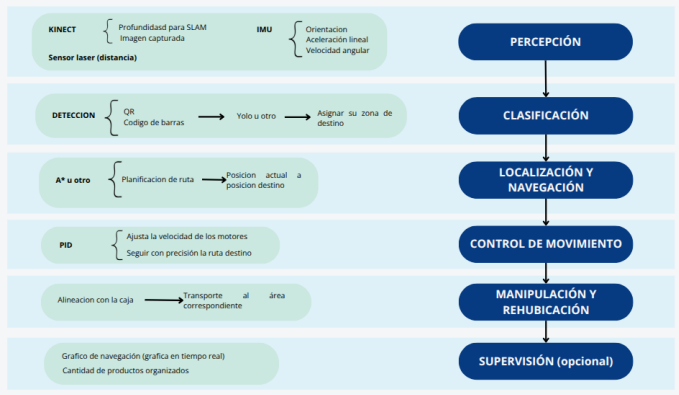

Figure 2. Diagrama del sistema

A. Approach/Design

The proposed system is designed under a modular and distributed architecture, combining sensory perception, real-time decision-making, and actuator control within the ROS 2 framework. The approach follows a hierarchical division of responsibilities: the **Raspberry Pi 5** acts as the master processor executing high-level SLAM, object detection (QR/ArUco), and navigation planning, while a **Raspberry Pi Pico** microcontroller handles low-level motor control through a dual PID loop.

Perception is achieved through a **Kinect v1** sensor, which provides RGB and depth data to support object detection, environment mapping, and pose estimation. An **MPU6050 IMU** assists with orientation data, improving localization and drift compensation in SLAM. All visual computation and decision-making are carried out onboard using **OpenCV**, SLAM algorithms, and PID control laws. Communication between components follows the ROS 2 publisher-subscriber model, while serial UART links are used between the Raspberry Pi 5 and Pico for real-time motor commands.

B. Data Collection

Data collection focused on two types of sensory information:

- **Visual and depth data** from the Kinect sensor, used to detect QR and ArUco markers and reconstruct the 3D environment.
- **Inertial data** from the MPU6050 module, used to estimate robot orientation and enhance pose estimation during SLAM.

Additionally, the motor encoders provided real-time velocity feedback, essential for PID tuning. Data from encoder RPMs in response to step input voltages were used to identify the transfer function of the motor system. All data were logged using ROS 2 tools such as `ros2 bag` for post-experiment analysis.

C. Experimental Setup / Algorithm Design

The experimental setup consists of:

- A **Raspberry Pi 5** running ROS 2 and managing the Kinect, MPU6050, QR detection, SLAM mapping, and path planning.
- A **Raspberry Pi Pico** receiving velocity commands via UART and controlling two **JGA25-370B** DC motors through a **L298N H-bridge**.
- Power distribution using two independent **18650 Li-ion battery packs**, one for logic/sensor circuitry and the other for motor power.
- A **Kinect v1** connected via USB for RGB-D imaging.
- An **MPU6050 IMU** connected to the Raspberry Pi 5 through I2C.

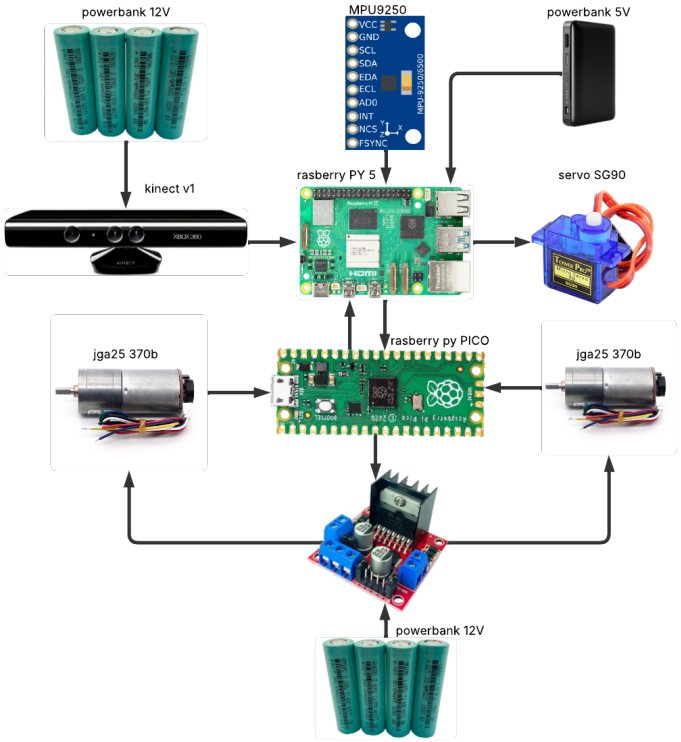

Figure 3. conexion entre compnentes electronicos

The Raspberry Pi 5 sends linear and angular velocity commands to the Pico via UART. The Pico calculates the appropriate PWM signals using PID controllers, one per motor, based on encoder feedback. The PID gains were calculated by identifying the motor transfer function through step input analysis. Control logic includes visual servoing for marker alignment, SLAM-based navigation planning, and dual PID velocity control.

This setup ensures clear separation of perception and control layers, improves system scalability, and allows for independent testing and debugging of each module.

IV. Experimental Setup/Algorithm Design

The robotic system was designed using a modular ROS 2 architecture, where each component is executed as an

independent node communicating via topics. The hardware platform includes a Raspberry Pi 5 as the main processing unit, interfaced with a Kinect camera for RGB and depth sensing, a laser distance sensor for close-range obstacle detection, and an MPU for orientation data. DC motors with encoders are controlled via a PID algorithm to ensure stable and precise motion, with motor feedback being continuously processed to adjust velocities in real time.

Navigation and mapping are achieved using a SLAM-based approach that fuses data from the Kinect depth camera and IMU. The depth data is continuously published to a dedicated topic, which is then subscribed to by nodes responsible for building the occupancy grid and for obstacle planning. For path planning, an A* or Wavefront algorithm is implemented to determine optimal trajectories from the robot's current location to predefined storage zones, avoiding static and dynamic obstacles based on real-time sensor input.

The system also incorporates visual object detection using ArUco markers placed on target objects. The RGB and depth streams from the Kinect are processed using OpenCV to detect the markers and calculate their relative position. Specifically, the system publishes the image depth from the Kinect, and several nodes subscribe to the corresponding topic to receive depth frames. A custom algorithm processes this information to extract the distance between the robot and the ArUco markers. These values are then transformed into positional data that allow the robot to detect and interact with objects, enabling accurate visual servoing and object manipulation within the warehouse environment.

V. Mathematical Models

Two PID controls were implemented to move the JGA25-370B motors and the robot in the given direction. The control was achieved by obtaining a transfer function, thus obtaining the PID gains for the encoder motor.

To find the mathematical model, a step function was used as input to find the motor's transfer function. Each step function input was input to the motor as a voltage signal based on the microcontroller used, with the encoder readings being plotted as an output function relative to the step function input to the motor. Some noise was obtained when the motor was given maximum voltage.

As a result of this using Matlab Tools, the transfer function was obtained, with an adjusted data estimate of 97.83

$$G(s) = \frac{77.22s + 0.0008115}{s^2 + 2.972s + 1.407e - 05} \tag{1}$$

The following PID gains were used for both motors.

Kp = 6.12522
Ki = 3.06929
Kd = 1.405

The following performance was obtained:

RiseTime: 0 seconds
SettlingTime: 0.1 seconds
Overshoot: 1.0

VI. Results and Discussion

During the conducted tests, the correct operation of each of the implemented nodes in the ROS 2 system was verified. The robot was programmed to detect ArUco visual markers using the node `/kinect_aruco_detector`, which processes the image from the `/kinect/kinect_ros2` camera node and publishes the estimated distance to the marker on the topic `/marker_distance_cm`, as well as the relative pose on `/marker_pose`. This information was crucial for determining the robot's relative position to the marker in X, Y, Z coordinates, and rotation angles, enabling automatic target tracking.

The `/marker_follower` node received the distance data and executed the control logic to follow the marker, generating linear and angular velocity commands published on `/cmd_vel`. Subsequently, the `/dual_motor_pid_node` processed these commands, applying independent PID control for each motor based on encoder readings. As a result, stable and controlled locomotion was achieved, adjusting the speeds of the left and right motors in real-time, which were published on the topics `/rpm_left` and `/rpm_right`.

In additional tests, an alternative flow was also validated where the speed command was manually defined from the `/setpoint_vel` node, demonstrating the versatility of the system in both automatic operation and direct control mode. In all cases, the modular architecture based on ROS topics allowed for clear communication between components, ensuring coherent robot behavior in different scenarios.

The results show that visual detection, pose estimation, tracking control, and motor regulation through PID controllers were correctly implemented and functional in all tested configurations.

VII. Conclusion

El robot desarrollado demostró una integración efectiva entre hardware embebido, algoritmos de control y percepción visual en el entorno ROS 2. Durante su implementación y pruebas, se validó la funcionalidad completa de los módulos diseñados, desde la detección visual de marcadores ArUco hasta la ejecución autónoma de trayectorias mediante controladores PID aplicados sobre motores DC con encoder.

A nivel de percepción, el uso de la cámara Kinect combinada con OpenCV permitió obtener una estimación precisa de la posición y orientación del marcador, lo cual fue clave para el seguimiento visual. La arquitectura modular del sistema, basada en nodos y tópicos, facilitó el flujo de información entre componentes como el nodo de detección `/kinect_aruco_detector`, el nodo de seguimiento `/marker_follower` y el controlador de motores `/dual_motor_pid_node`. Esta organización permitió depurar y validar cada parte de forma independiente, asegurando escalabilidad para futuras mejoras.

En cuanto al control del movimiento, se aplicó un enfoque riguroso mediante la obtención de la función de transferencia del sistema motor-encoder, logrando ajustes adecuados de

parámetros PID. Esto resultó en una locomoción precisa, suave y estable durante el seguimiento de objetivos visuales.

El sistema demostró un comportamiento confiable en todos los ejercicios planteados: control básico con comandos `/cmd_vel`, detección y localización de marcadores, y aproximación controlada a objetivos. Estas pruebas confirman que el diseño es robusto y adaptable, cumpliendo con los requisitos de un sistema móvil autónomo para gestión de almacenamiento.

Como conclusión general, se validó que el enfoque propuesto basado en tecnologías accesibles como Raspberry Pi, ROS 2 y visión artificial, permite construir soluciones eficientes y de bajo costo para aplicaciones logísticas. Este desarrollo abre la puerta a mejoras futuras como la integración de SLAM, manipulación robótica y aprendizaje automático para tareas de clasificación y optimización dinámica de rutas.

VIII. References

References

[1] Stewart, R. and L. Church: *ROS 2 Robot With SLAM: Lightweight Implementation on Raspberry Pi Platforms*. Robotics and Automation Letters, 9(2):1120–1127, 2024.

[2] Choudhary, A., V. Patil, P. Shah, and R. Rode: *Simultaneous localization and mapping for warehouse applications*. International Journal of Robotics and Control, 12(1):45–52, 2024. https://www.researchgate.net/publication/SLAM_for_Warehouse_Applications.

[3] Fernández, M., Q. Li, and R. Rusu: *A mobile mapping system for ros 2-based autonomous robots*. In *Proceedings of the 2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1805–1812. IEEE, 2025.

[4] Chen, J., T. Huang, M. Wei, and Y. Zhang: *High-precision localization for transport robots in industrial environments based on improved amcl and qr code assistance*. Journal of Intelligent Robotic Systems, 109(3):450–463, 2025.

[5] Kashyap, R. and S. Konathalapalli: *Autonomous navigation of ros 2-based turtlebot3 in static and dynamic environments using intelligent approaches*. In *2025 IEEE International Conference on Mechatronics and Automation (ICMA)*, pages 720–727. IEEE, 2025.

IX. Annexes

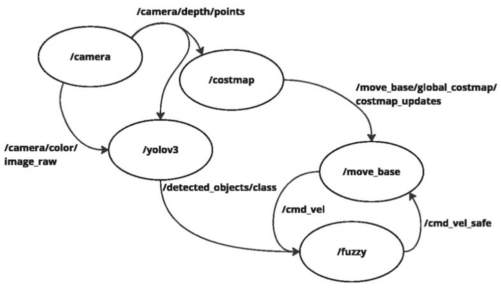

Figure 4. Nodes Diagram

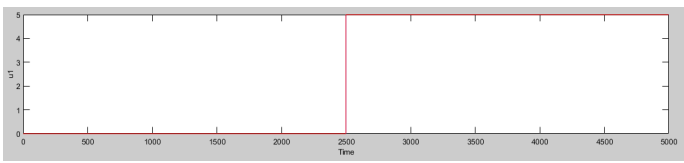

Figure 5. Input step function set to JGA25-370B Motor

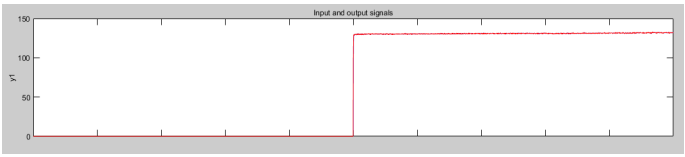

Figure 6. Output function with respect to JGA25-370B Motor

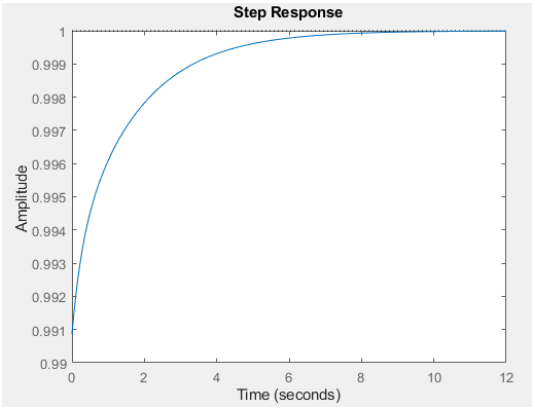

Figure 7. Simulation PID