Shared Autonomy for an Omnidirectional Robot with SLAM and Collision Avoidance

Mireya Zapata[1(✉)], Alejandro Camino[1], Vannesa Vargas[2], and Pablo Ramos[2]

[1] Centro de Investigación en Mecatrónica y Sistemas Interactivos (MIST), Facultad de Ingenierías, Universidad Indoamérica, Machala y Sabanilla, Quito, Ecuador
mireyazapata@uti.edu.ec

[2] Departamento de Eléctrica, Electrónica y Telecomunicaciones, Universidad de las Fuerzas Armadas ESPE, Av. General Rumiñahui y Ambato, 171103 Sangolquí, Ecuador

Abstract. This paper describes the design and testing of an omnidirectional mobile robot (OMR) with Mecanum wheels, designed to implement assisted teleoperation within the Robot Operating System (ROS) framework. The platform has a 2D LiDAR sensor for environmental perception. Which enables real-time mapping and localization; in this case using the Hector SLAM algorithm to build 2D occupancy maps in real time, along with the Dynamic Window Approach (DWA) to ensure a safety layer that avoids obstacles while remote-controlling. The proposed shared-autonomy architecture demonstrates that combining SLAM-based mapping with reactive navigation makes teleoperation tasks much safer and more efficient. Experimental results show that the system allows the operator to control the mobile robot while still enabling it to avoid collisions on its own. The system is designed to support, not override, the operator's commands; instead, it only takes action when in close proximity with an obstacle. In contrast with other obstacle avoidance solutions, our proposal allows the user to retain navigation control as the obstacle avoidance algorithm takes action, additionally it supports real-time mapping; this allows for navigation on unknown scenarios while maintaining obstacle avoidance capabilities. Validation of the prototype proves the efficacy of this hybrid methodology for assisted teleoperation applications. The prototype shows high odometry precision (relative error <1%) and high mapping accuracy, demonstrating the system's robustness. Furthermore, testing shows that autonomous safety features in a teleoperation system, illustrating this hybrid methodology as a feasible solution for applications requiring accuracy and safety in intricate environments.

Keywords: Omnidirectional robot · mecanum wheel · collision avoidance · ROS · SLAM · Hector SLAM · teleoperation · shared autonomy · Dynamic Window Approach

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2027
O. Gervasi et al. (Eds.): ICCSA 2026, LNCS 16769, pp. 556–567, 2027.
https://doi.org/10.1007/978-3-032-30494-0_35

1 Introduction

Mobile robotics has witnessed significant growth, with an increasing variety of autonomous robots capable of carrying out complex tasks in diverse environments such as home, industry, military, and rescue scenarios. The ability of these systems to sense, learn, and engage with the world around them is a key enabling technology for recent advances in control, and robotic systems. Nevertheless, path planning in an unknown, dynamic environment remains a challenging problem for the increased autonomy and intelligence of mobile robots, due to a number of constraints imposed by the robot and its environment. The uncertainty characterizing real-world environments and the inherent limitations of sensor technology make the design of robust and intelligent navigation systems a major challenge today. In addition, the generation of environment maps is necessary for a wide range of applications, such as vehicle navigation, robot obstacle avoidance, search and rescue operations, and indoor and outdoor surveillance [1,2].

In this sense, omnidirectional mobile robots (OMRs) are a novel class of robots that can freely move in any direction and at any orientation, defying the non-holonomic constraints of traditional robots which need to reorient to change direction. This feature provides better maneuverability and accuracy, which is very useful in a limited-space working area and in a cluttered or dynamic environment, e.g., factory workshops, warehouses, or hospitals [3].

Teleoperation allows the human operator to directly control a robot from a remote location, being very effective in situations where the environment is hazardous or unreachable for human beings. In these architectures, the human performs high-level cognitive and localization tasks, while the robot handles low-level motion control. Shared autonomy methodologies for teleoperation have recently overcome key challenges such as network delays and model errors. Gulecyuz et al. [4] introduced a new shared autonomy mechanism that learns from teleoperated demonstrations during late-stage deployment, despite network-induced delays, and using confidence-aware arbitration balancing human and autonomous control. Likewise, Jabbour [5] proposed a two-term adaptive assistance scheme for compensating for model uncertainties, where operator input is considered as a corrective measurement and by integrating adaptive Kalman filtering in real-time for compensation of local errors and Procrustes analysis for correcting global misalignment.

Behind the autonomy and transparency of teleoperation, there is the importance of the real-time pose tracking and map-building system capabilities. The simultaneous localization and mapping (SLAM) problem is fundamental in mobile robotics and enables a robot to construct a representation (map) of an unknown environment while concurrently tracking its location and orientation within that representation [2]. LiDAR sensors have become increasingly popular for SLAM, mainly due to their accuracy, robustness, wide field of view, and insensitivity to light conditions. Recent research has analyzed different 2D LiDAR SLAM algorithms, exemplifying the trade-off between filter-based methods, e.g.,

Hector SLAM and Gmapping, and graph-based methods such as Cartographer and KartoSLAM [6,7].

Moreover, reactive obstacle avoidance is essential for safe navigation in dynamic and rough environments, particularly when robots come in contact with humans. Reactive control strategies enable the robot to adapt its path while traveling based on real-time sensor information to avoid collisions [8]. In this sense, there has been a rising interest on local planning algorithms such as the Dynamic Window Approach (DWA) taking into account its computational efficiency and kinodynamic feasibility [9]. Recent extensions of DWA have been introduced to address its drawbacks in more complex scenarios. Zhang et al. [10] proposed an enhanced DWA based on SSA for UAV path planning in a multi-dynamic obstacle environment, where the path length, safety distance, and iteration efficiency were notably improved. However, integrating real-time LiDAR-SLAM perception with the kinematic control of omnidirectional mobile robots (OMR) in dynamic environments remains a worthwhile problem to be investigated [11]. Sensor fusion solutions have proven effective for overcoming localization issues in indoor environments without GPS signals. Lee et al. [12] showed that wheel-visual-inertial odometry outperforms other standalone odometry methods in feature-poor environments.

In this paper, the challenge of providing a dependable and secure robotic system in a cluttered and dynamic environment is addressed by developing an OMR control software using ROS. The purpose is to build a teleoperation platform that couples real-time localization and mapping system with reactive obstacle avoidance for safe navigation under human control, reconciling the autonomy efficiency with the intelligent supervision of a human operator.

The remainder of this paper is organized as follows. Section 2 details the hardware and software architecture of the implemented system, along with an explanation of the proposed experiments to validate the system's performance. Section 3 presents the results obtained in the evaluation process, presenting real-life testing data and providing visual representations of the experiments. Section 4 discusses the results and its limitations. Finally, Sect. 5, concludes the work and analyzes how the proposed solution contributes to teleoperation-obstacle avoidance sector.

2 Methods

The following section presents the applied research methodology for this project, divided into four major parts: hardware platform implementation, software architecture design, system integration, and experimental validation.

2.1 Hardware Platform Implementation

The mobile robot, shown in Fig. 1, is an OMR with four Mecanum wheels (MD60, 152 mm diameter), each powered by a 24 V DC motor (MD60P18P, 100 W, 175 rpm nominal) with a quadrature encoder.

The four RoboClaw Solo 30A motor controllers manage the low-level PID velocity control for the four motors, with gains adjusted using the BasicMicro Motion Studio software.

A Raspberry Pi 4 Model B+ (4 GB RAM) serves as the main computer on board, running Ubuntu Server 20.04 LTS and ROS Noetic. Environmental perception is provided by an RPLidar A2M8 2D LiDAR sensor (16m range, 8000 samples/s).

The system is powered by a 60 V, 20 Ah lithium main battery, stepped down to 24 V for the motors and 5 V for the logic components. A wireless joystick allows remote operation.

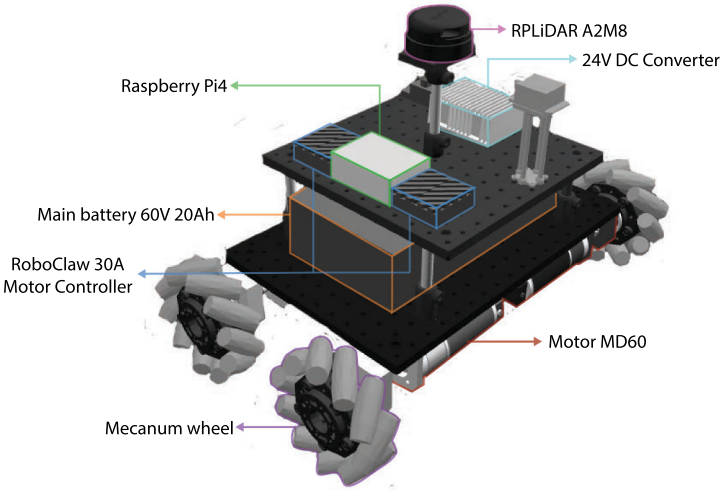

Fig. 1. Prototype of the omnidirectional mobile robot and its 3D model. Physical prototype with motor labels (M1-M4).

2.2 Software Architecture and ROS Implementation

The software architecture, shown in Fig. 2, is a modular ROS-based system. The key nodes and their interactions are as follows:

- **rplidarNode.cpp**: Interfaces with the RPLidar sensor, publishing raw laser scan data to the `/scan` topic (sensor_msgs/LaserScan).
- **joystick_bypass_dwa.py**: Captures joystick commands and publishes them as raw velocity commands on the **`/cmd_vel_joystick`** topic (geometry_msgs/Twist).
- **advanced_safety_mini_dwa.py**: Implements the shared autonomy core. It subscribes to `/scan` and **`/cmd_vel_joystick`**, using the DWA algorithm

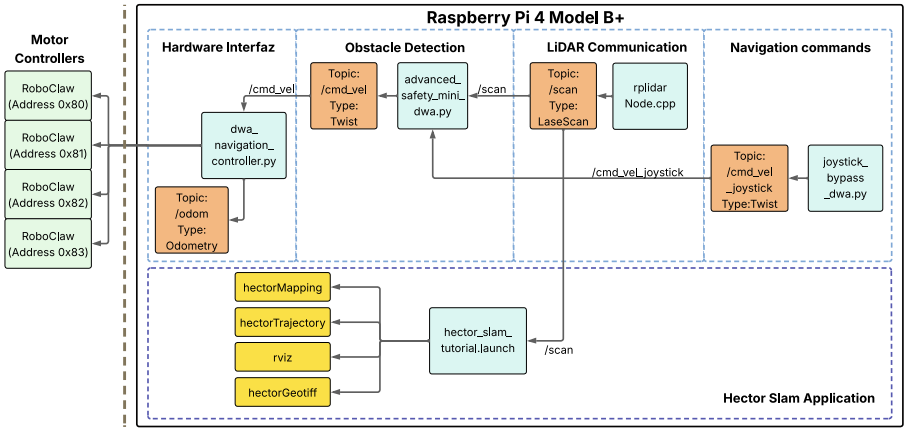

Fig. 2. ROS software architecture for the omnidirectional mobile robot, showing nodes (blue), topics (orange), and data flow. (Color figure online)

to generate safe velocity commands when in proximity with obstacles. These are published on the `/cmd_vel` topic, effectively filtering the operator's commands to avoid imminent collisions.

- **dwa_navigation_controller.py**: Subscribes to `/cmd_vel`, computes the inverse kinematics for the four Mecanum wheels based on Eq. 1, and sends individual velocity setpoints to each RoboClaw controller via serial communication. It also computes odometry from wheel encoders and publishes it on the `/odom` topic (nav_msgs/Odometry).
- **hector_mapping** package: Subscribes to the `/scan` topic to perform SLAM, generating a 2D occupancy grid map and publishing the robot's estimated pose (`/slam_out_pose`). This map is visualized in RViz.

The inverse kinematics that relates the robot's velocity vector $\mathbf{U} = [v_x, v_y, \omega]^T$ to individual wheel angular velocities ω_i is given by:

$$\begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \omega_4 \end{bmatrix} = \frac{1}{r} \begin{bmatrix} 1 & -1 & -(l_x + l_y) \\ 1 & 1 & l_x + l_y \\ 1 & -1 & l_x + l_y \\ 1 & 1 & -(l_x + l_y) \end{bmatrix} \begin{bmatrix} v_x \\ v_y \\ \omega \end{bmatrix} \tag{1}$$

where r is the wheel radius, and l_x and l_y are the distances from the robot's center to the wheel axles along the x and y axes.

An important clarification of the proposed system needs to be established. As mentioned previously the obstacle avoidance layer is designed to support the operator and not completely override the operator's command. In this context, the **advanced_safety_mini_dwa.py** node receives the raw velocity command of the joystick `/cmd_vel_joystick`, and as long as this node does not detect an obstacle inside the threshold region, the raw joystick command is directly passed to the final output velocity command `/cmd_vel`. In the case that the algorithm

detects an obstacle inside the region, the joystick command is processed to represented a desired pose instead of a velocity command, taking the direction of the desired joystick velocity and projecting a point in this direction; this pose is the target position that the DWA algorithm takes as reference to compute its velocity command.

2.3 Experimental Validation

Three sets of experiments were conducted to validate the system's performance:

1. **Odometry Accuracy**: The robot was programmed to perform linear displacements of 400 cm in the x and y directions (10 repetitions each) and rotational maneuvers ($90°, 180°, 270°, 360°$) with five repetitions each. Odometry estimates were compared against real displacements and IMU data (for rotations) to compute absolute and relative errors.
2. **Mapping Precision**: Three independent mapping runs of an L-shaped corridor (dimensions specified in Fig. 3) were performed using Hector SLAM. The lengths of eight distinct segments in the generated maps were measured using RViz's measure tool and compared to ground-truth manual measurements.
3. **Obstacle Avoidance Effectiveness**: The robot was teleoperated directly towards a static obstacle from varying initial distances (2.2 m to 4.2 m). Each repetition measured the time taken by the DWA algorithm to avoid the obstacle interpreted as the time difference between the algorithm taking control of the velocity command to it returning full control to the operator.

3 Results

The experimental results are presented below, focusing on the three key performance metrics.

3.1 Odometry Accuracy

The odometry system demonstrated high precision. For linear displacements, the average errors were minimal, as shown in Table 1. For rotations, all relative errors remained below 1%, with the highest error (0.62%) observed for the 360° rotation. The results confirm the robustness of the odometry estimation for navigation tasks.

3.2 Cartography Accuracy with Hector SLAM

To assess the accuracy of the maps, three independent mappings of an L-shaped corridor were conducted. The experimental space layout with eight rectilinear segments a-h (see Fig. 3). The lengths of the segments were measured using the RViz-measure tool and compared with manual ground-truth measurements. The full details for each segment across the three runs are given in Table 2, and Fig. 4 shows one of the resulting occupancy grid maps in RViz.

Table 1. Absolute and relative errors for linear and angular odometry.

Test Type	Real Value	Mean Value	Rel. Error (%)
Δx	400 cm	401.7 cm	0.42
Δy	400 cm	403.8 cm	0.95
$\Delta\theta$ (90°)	90°	90.6°	0.43
$\Delta\theta$ (180°)	180°	180.6°	0.11
$\Delta\theta$ (270°)	270°	270.3°	0.32
$\Delta\theta$ (360°)	360°	358.4°	0.62

These results show great metric accuracy in the cartographic reconstruction. Analyzing the eight measured segments, the relative error is always less than 0.35%, with a mean value of 0.19%. Although the largest absolute errors are attained for the two longest segments (d and h, with –1.70 cm and –2.20 cm, respectively), the corresponding relative errors are still small (0.17% and 0.13%), so the important conclusion is that the magnitude of the error is scaled with the measured distance.

The reproducibility of the three separate runs for each segment demonstrates the robustness of the mapping, confirming that Hector SLAM produces metrically accurate environment maps that can be used in an application with a mobile robot.

These results are consistent with Zhao et al. [7], who showed that the mapping accuracy of Hector SLAM in structured environments is very high.

Table 2. Measurements of the eight segments (a-h) within the mapped space, as obtained from RViz.

Segment of the path	Ground Truth (cm)	Map Measurements			Mean value (cm)	Rel. Error (%)
		1	2	3		
a	188.2	187.6	187.6	187.5	187.6	0.34
b	179.8	178.6	179.5	180.0	179.4	0.24
c	221.0	220.9	220.7	220.6	220.7	0.12
d	1011.3	1010.3	1007.6	1010.9	1009.6	0.17
e	178.4	177.7	178.4	178.3	178.1	0.15
f	176.5	176.4	175.6	176.3	176.1	0.23
g	176.5	176.2	177.1	176.8	176.7	0.11
h	1747.5	1745.9	1745.3	1744.7	1745.3	0.13

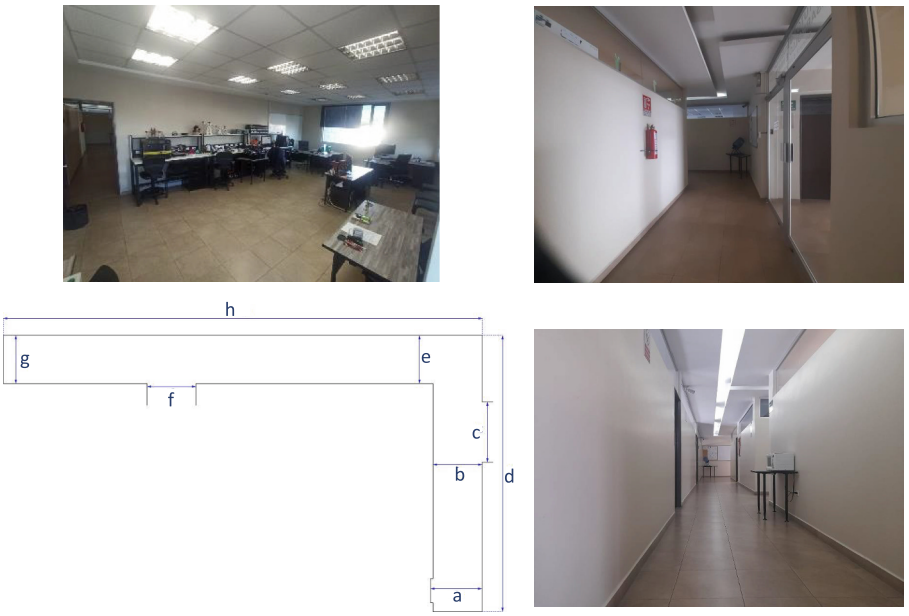

Fig. 3. Layout of the L-shaped corridor used as the testing scenario. The segments labeled with letters (from a to h) indicate the locations where measurements were taken during the tests.

3.3 Obstacle Avoidance with DWA

The DWA-based safety layer consistently intervened to prevent collisions. The reaction time of the algorithm was stable, averaging around 5.36 s across four trials with varying initial distances (see Table 3). This consistency suggests the avoidance behavior is governed by the fixed safety parameters (e.g., 0.8 m safe distance) rather than the initial distance. Figure 5 shows a teleoperation session where the DWA algorithm adjusted the commanded velocity in real time to avoid an obstacle, while Hector SLAM continued to build the map.

The observed performance is consistent with recent DWA enhancements reported in the literature. Zhang et al. [13] demonstrated that incorporating gradient information into DWA improves obstacle awareness, particularly for non-convex obstacles. While our implementation uses the standard DWA formulation, the consistent reaction times validate its suitability as a non-intrusive safety layer for shared autonomy applications.

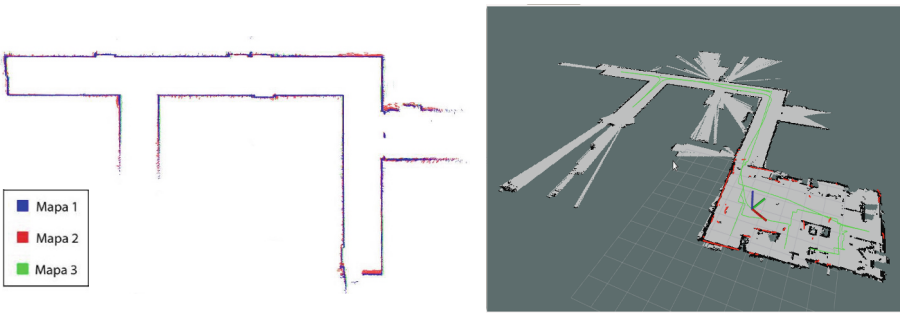

Fig. 4. Occupancy grid map of the L-shaped corridor generated by Hector SLAM, visualized in RViz. The overlay of the three obtained maps demonstrates the reproducibility of the measurements.

Table 3. Reaction time of the DWA algorithm for obstacle avoidance from varying initial distances.

Trial	Initial Distance (m)	DWA Reaction Time (s)
1	2.20	5.37
2	2.20	5.50
3	3.20	5.16
4	4.20	5.40

4 Discussion

The results confirm the viability of the shared-autonomy approach for the Mecanum-wheeled robot. The strongly coupled integration of Hector SLAM and a DWA-based safety layer in a ROS environment was successful on performing assisted teleoperation.

The high precision of the odometry system (relative errors <1%) is noteworthy, as it mitigates one of the primary drawbacks of Mecanum wheels—slippage. This accuracy is crucial for reliable localization within the SLAM framework. Similarly, the mapping accuracy (0.19% mean relative error) confirms that Hector SLAM, using only LiDAR data, can generate metrically consistent maps of indoor environments, providing valuable situational awareness to the remote operator. The consistent performance of the DWA algorithm, with its reaction times, validates its use as a non-intrusive safety layer. It effectively acts as an "assistant," modifying the operator's commands only when necessary to prevent a collision, thereby reducing cognitive load and enhancing safety without completely removing human control. This is clearly evidenced in Fig. 5, where the

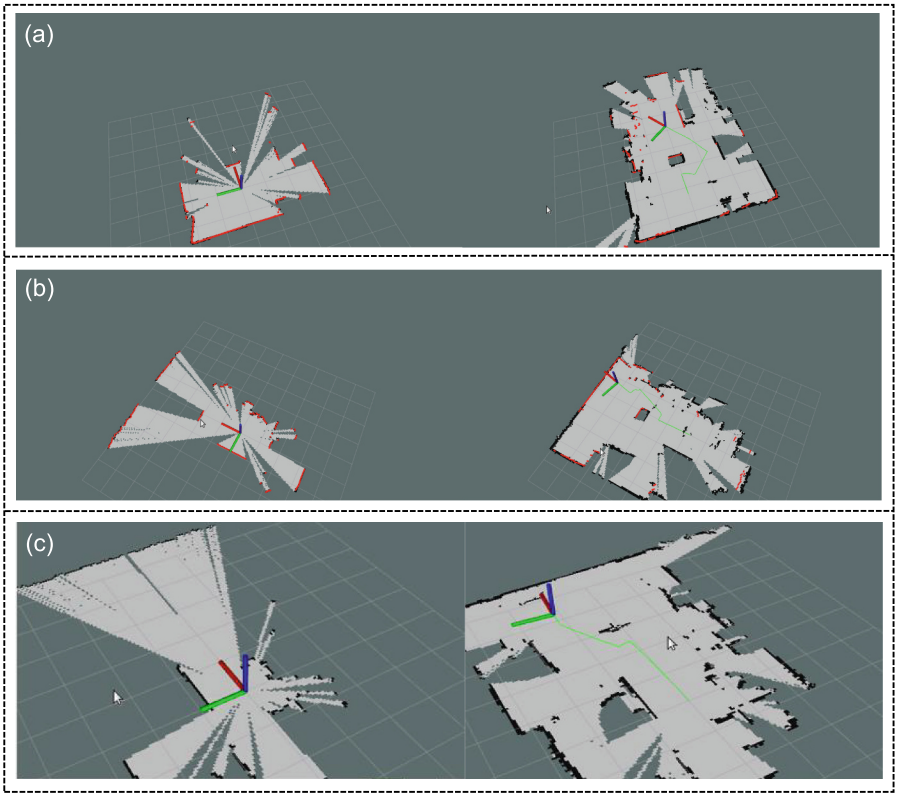

Fig. 5. Experimental validation of obstacle avoidance with DWA in RViz. (a) System initiates mapping under manual joystick control while ensuring safe navigation. (b-c) In obstacle-dense environments, DWA adjusts trajectories in real time, maintaining safe and fluid paths, while Hector SLAM generates coherent and detailed maps. (Color figure online)

executed trajectory (green line) smoothly deviates from the direct path to avoid an obstacle detected by the LiDAR (red points).

However, the study has limitations. The inherent susceptibility of Mecanum wheels to slippage on uneven terrains remains a challenge for odometry, though the use of SLAM partially corrects for this. Hector SLAM, while robust, can drift during rapid rotations, and the 2D LiDAR is blind to obstacles above or below its scan plane, such as table overhangs. Additionally for the DWA implementation, the velocity command transition logic using distance threshold might not be sufficient in the presence of dynamic obstacles. Moreover, although the conversion from joystick velocity command to reference pose seemed to be sufficient in the testing environment of this study, it might not prove so efficient in more diverse and cluttered environments, as the forward reference point is

set at a constant distance from the robot it could place the pose inside of an obstacle.

Future work will focus on sensor fusion by integrating an IMU and depth cameras for more robust localization and perception. There is room for improvement on the logic for when to activate the obstacle avoidance layer, future work fill focus on smoother and safer transition. Furthermore by improving the transformation from joystick command to forward reference point the performance of the DWA algorithms would highly improve. Besides that, migrating to ROS 2 would also provide better real-time performance and support for more complex, multi-sensor fusion algorithms and localization algorithms.

5 Conclusion

In this paper, we have presented the design, implementation, and validation of an omnidirectional mobile robot, based on Mecanum wheels, for shared autonomy teleoperation in the ROS environment. The main contribution is the tight integration of Hector SLAM for real-time mapping, with a minimally invasive reactive obstacle avoidance layer based on DWA, that plays the role of non-intrusive safety assistant for a human operator.

Experimental results verified that the system can simultaneously obtain high precision odometry and mapping (with relative errors less than 1% and 0.2%, individually), and perform efficient collision avoidance. The results demonstrate that this hybrid shared autonomy approach improves the safety and efficiency of teleoperation in challenging scenarios, providing a feasible solution to practical applications such as remote inspection in hazardous regions or high-precision logistics. The ROS implementation is open source, providing a solid basis for future research and development in autonomous mobile robots.

References

1. Li, Z., Chamran, K., Alobaedy, M.M., Sheikh, M.A., Siddiqui, T., Ahad, A.: An optimization on 2d-slam map construction algorithm based on lidar. J. Intell. Robotic Syst. **110**(4) (2024). https://doi.org/10.1007/s10846-024-02123-1
2. Payá, L., Gil, A., Reinoso, O.: A state-of-the-art review on mapping and localization of mobile robots using omnidirectional vision sensors. J. Sensors **2017**, 1–20 (2017). https://doi.org/10.1155/2017/3497650
3. Burkacki, M., Łysy, I., Suchoń, S., Chrzan, M., Kowolik, R.: Systematic review of mecanum and omni wheel technologies for motor impairments. Appl. Sci. **15**(9), 4773 (2025). https://doi.org/10.3390/app15094773
4. Güleçyüz, B.: Enhancing shared autonomy in teleoperation under network delay: transparency- and confidence-aware arbitration. IEEE Robot. Autom. Lett. **10**(10), 9654–9661 (2025). https://doi.org/10.1109/lra.2025.3596436
5. Elio, J.: Shared-autonomy control for improving Human-Robot collaboration in haptic teleoperation. Theses, Université de Bordeaux (Nov 2025). https://theses.hal.science/tel-05460108

6. Ran, Y., Xu, X., Tan, Z., Luo, M.: A review of 2d lidar slam research. Remote Sensing **17**(7), 1214 (2025). https://doi.org/10.3390/rs17071214
7. Zhao, J., Liu, S., Li, J.: Research and implementation of autonomous navigation for mobile robots based on slam algorithm under ros. Sensors **22**(11), 4172 (2022). https://doi.org/10.3390/s22114172
8. Azizi, M.R., Rastegarpanah, A., Stolkin, R.: Motion planning and control of an omnidirectional mobile robot in dynamic environments. Robotics **10**(1), 48 (2021). https://doi.org/10.3390/robotics10010048
9. Lin, Z., Taguchi, R.: Faster implementation of the dynamic window approach based on non-discrete path representation. Mathematics **11**(21), 4424 (2023). https://doi.org/10.3390/math11214424
10. Zhang, L., Li, Y., Yu, Y., Zhao, Y., Song, W., Retscher, G.: Improved dynamic window approach for uav local path planning in multi-dynamic obstacles environments using sparrow search algorithm. Trans. Emerging Telecommun. Technol. **36**(10), e70284 (2025). https://doi.org/10.1002/ett.70284, https://onlinelibrary.wiley.com/doi/abs/10.1002/ett.70284
11. Rivai, M., Hutabarat, D., Nafis, Z.M.J.: 2d mapping using omni-directional mobile robot equipped with lidar. TELKOMNIKA (Telecommun. Comput. Electr. Control) **18**(3), 1467 (2020). https://doi.org/10.12928/telkomnika.v18i3.14872
12. Lee, C., Hur, S., Kim, D., Yang, Y., Choi, D.: Insufficient environmental information indoor localization of mecanum mobile platform using wheel-visual-inertial odometry. J. Mech. Sci. Technol. **38**(9), 5007–5015 (2024). https://doi.org/10.1007/s12206-024-0836-z
13. Zhang, Z., Xue, Y., Figueroa, N., Åkesson, K.: Gradient field-based dynamic window approach for collision avoidance in complex environments. In: 2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 19669–19674 (2025). https://doi.org/10.1109/IROS60139.2025.11246091