BIRUNI: A ROS 2-BASED AUTONOMOUS MECANUM-WHEELED MOBILE ROBOT WITH MULTI-SENSOR FUSION, REAL-TIME SLAM, AND NAVIGATION IN GAZEBO FORTRESS -2025

BIRUNI: ROBOT TỰ HÀNH CÓ BÁNH XE ĐA HƯỚNG DỰA TRÊN ROS 2 VỚI KHẢ NĂNG KẾT HỢP NHIỀU CẢM BIẾN, ĐỊNH VỊ VÀ LẬP BẢN ĐỒ THỜI GIAN THỰC TRONG MÔI TRƯỜNG GAZEBO FORTRESS -2025

Elyounoussi Ossama, Nguyen Xuan Thuan, Vu Tien Dung*
School of Mechanical Engineering,
Hanoi University of Science and Technology (Hanoi, Vietnam)
**Corresponding author email: dung.vutien@hust.edu.vn*

ABSTRACT

This document presents Biruni, a ROS 2-based autonomous mecanum-wheeled mobile robot designed and evaluated in a high-fidelity Gazebo Fortress simulation environment. The robot integrates a multi-sensor perception architecture consisting of a 2D LiDAR, an RGB camera, an Inertial Measurement Unit (IMU), and wheel encoders. A unified Extended Kalman Filter (EKF) is implemented to fuse inertial and odometry data, generating a stable and drift-minimized state estimate for real-time navigation. Cartographer SLAM is used to produce consistent 2D occupancy maps, while the Navigation2 (Nav2) framework provides global and local planning, obstacle avoidance, and autonomous waypoint following. The complete system is structured around a modular ROS 2 package architecture with a fully validated TF tree, URDF/Xacro robot model, and synchronized sensor frames. Simulation results demonstrate accurate localization, reliable mapping, and smooth path execution with omnidirectional control using mecanum kinematics. The Biruni platform offers a reproducible foundation for education, research, and future deployment on physical hardware.

Keywords: *ROS 2; SLAM; Navigation; EKF; Sensor Fusion; Gazebo Fortress; Autonomous Mobile Robot; Mecanum Wheels.*

TÓM TẮT

Tài liệu này giới thiệu Biruni, một robot tự hành sử dụng bánh xe đa hướng dựa trên nền tảng ROS 2, được thiết kế và đánh giá trong môi trường mô phỏng Gazebo Fortress độ trung thực cao. Robot tích hợp kiến trúc nhận thức đa cảm biến bao gồm LiDAR 2D, camera RGB, thiết bị đo quán tính (IMU) và encoder gắn trên bánh xe. Bộ lọc Kalman Mở rộng (EKF) thống nhất được triển khai để kết hợp dữ liệu quán tính và dữ liệu đo quãng đường, tạo ra ước tính trạng thái ổn định và giảm thiểu trôi dạt cho điều hướng thời gian thực. Thuật toán định vị và lập bản đồ (SLAM) thời gian thực được sử dụng để tạo bản đồ vị trí 2D nhất quán, trong khi khung Navigation2 (Nav2) cung cấp khả năng lập kế hoạch toàn cục và cục bộ, tránh chướng ngại vật và theo dõi điểm dừng *tự động. Toàn bộ hệ thống được cấu trúc xung quanh kiến trúc gói ROS 2 dạng mô-đun với cây TF được xác thực đầy đủ, mô hình robot URDF/Xacro và các khung cảm biến được đồng bộ hóa. Kết quả mô phỏng chứng minh khả năng định vị chính xác, lập bản đồ đáng tin cậy và thực hiện đường đi mượt mà với điều khiển đa hướng sử dụng động học mecanum. Nền tảng Biruni cung cấp một nền tảng có thể tái tạo cho giáo dục, nghiên cứu và triển khai trong tương lai trên phần cứng vật lý.*

Từ khóa: *ROS 2; Định vị và lập bản đồ; Bộ lọc Kalman; Đa cảm biến; Môi trường Gazebo; Robot tự hành; Bánh xe đa hướng.*

1. INTRODUCTION

The development of autonomous mobile robots has gained significant attention in recent years due to their wide range of applications in logistics, industrial automation, service robotics, and research environments. These systems rely on the integration of mechanical design, perception, and intelligent software frameworks to achieve autonomy in dynamic and uncertain environments. A fundamental capability enabling such autonomy is Simultaneous Localization and Mapping (SLAM), which allows a robot to build a map of an unknown environment while simultaneously determining its position within it.

Among the various locomotion mechanisms used in mobile robots, mecanum wheels offer a unique advantage due to their omnidirectional movement capability, which enhances maneuverability in confined or structured spaces. This type of drive system is particularly suitable for indoor navigation, where flexible movement and precise control are required. However, the integration of mecanum kinematics with advanced SLAM algorithms and navigation systems presents both modeling and control challenges that require careful design and validation.

With the growing maturity of the Robot Operating System 2 (ROS 2) and its associated simulation environments, such as Gazebo Fortress (Ignition), open-source robotic development has become a practical platform for both educational and research purposes. These frameworks provide modular tools for robot modeling, sensor simulation, and algorithm testing within realistic virtual environments.

This study presents the design, simulation, and evaluation of “Biruni,” a four-wheeled mecanum autonomous mobile robot developed using ROS 2 Humble and Gazebo Fortress. The robot is equipped with a 2D LiDAR for planar environment perception, an Inertial Measurement Unit (IMU) for motion estimation and stability, and an Intel RealSense D435 RGB-D camera for visual sensing. The camera was successfully integrated into the simulation environment, allowing real-time image and depth data transmission to ROS 2 topics, although it was not yet utilized for autonomous decision-making. The primary SLAM framework used is Google Cartographer, which enables real-time map construction and localization. After the mapping phase, the Nav2 stack is implemented to achieve autonomous navigation and obstacle avoidance within the mapped environment.

The main objective of this work is to demonstrate the feasibility of integrating a mecanum drive system with modern ROS 2 SLAM and navigation frameworks using open-source tools. The proposed system’s ☞

performance is validated in a simulated environment, highlighting its capability to perform accurate mapping, localization, and path-planning tasks while maintaining smooth omnidirectional movement.

2. RELATED WORK

Autonomous mobile robots have been the focus of extensive research over the past two decades, particularly in the fields of localization, mapping, and navigation. Numerous open-source and commercial platforms have contributed to the development of robotic systems capable of performing complex tasks in structured and unstructured environments.

Early works such as the TurtleBot and Husky platforms have been widely adopted for research and education due to their modularity and integration with the Robot Operating System (ROS). These platforms have served as foundational tools for developing and validating SLAM and navigation algorithms in both simulated and real-world environments. However, most of these platforms employ differential or skid-steer drive mechanisms, which limit their maneuverability compared to omnidirectional systems.

Mecanum-wheel robots have attracted growing attention because of their ability to perform omnidirectional motion without changing orientation, which is advantageous in constrained spaces such as warehouses or indoor facilities. Studies such as those by [1] and [2] have explored kinematic modeling and control algorithms for mecanum-driven robots, demonstrating their potential for agile navigation and precise positioning. However, integrating mecanum motion control with robust SLAM and navigation frameworks remains a challenging task, particularly due to wheel slippage and odometry inaccuracies.

Recent advancements in the ROS 2 ecosystem have enabled more scalable and modular approaches to robotic system design. Research by [3] and [4] has demonstrated the effectiveness of ROS 2 middleware in supporting real-time communication and distributed control for autonomous systems. Furthermore, Google Cartographer has become one of the most reliable 2D SLAM algorithms in open-source robotics due to its real-time performance and multi-sensor fusion capabilities, as discussed in [5].

Several simulation-based studies have also utilized Gazebo or Ignition (Gazebo Fortress) environments for developing and testing autonomous robots prior to physical implementation. These environments provide physics-based simulation, sensor emulation, and ROS integration, enabling researchers to validate mapping and navigation algorithms efficiently. However, few studies have specifically focused on combining mecanum kinematics, Cartographer SLAM, and Nav2 navigation within ROS 2, particularly using open-source models designed from scratch.

This study contributes to this research area by presenting the design and simulation of an autonomous mecanum-wheeled robot – Biruni – built entirely using open-source tools and ROS 2 Humble. The robot integrates LiDAR, IMU, and RGB-D camera sensors to achieve mapping, localization, and navigation using Cartographer and Nav2 frameworks. The work aims to bridge the gap between theoretical research and practical implementation of omnidirectional SLAM-capable robots in modern robotic ecosystems.

3. SYSTEM ARCHITECTURE AND IMPLEMENTATION

The Biruni autonomous mobile robot was developed using a modular architecture based on ROS 2 Humble [6] and simulated using Gazebo Fortress (Ignition) [7]. The system is organized into three primary packages: biruni_description, biruni_gazebo, and biruni_navigation, each corresponding to a specific functional layer within the robot's software stack.

a. Robot Description and Simulation

The biruni_description package defines the robot's complete structural and kinematic model using URDF/Xacro. The main file, robot_3d.urdf.xacro, specifies a four-wheeled mecanum configuration equipped with an IMU and an Intel RealSense D435 RGB-D camera [8]. All mechanical, inertial, and sensor elements needed for accurate simulation are included.

The biruni_gazebo package provides the simulation environment and plugins required for Gazebo Fortress [7]. It includes world files, sensor configurations, and the biruni_fusion.launch.py script, which spawns the robot and initializes all necessary ROS-Gazebo bridges. Through the ros_gz_image bridge [9], the RealSense D435 produced synchronized RGB, depth, and infrared image streams; although this data was not yet used for navigation decisions, its complete integration validated sensor functionality within the simulated environment.

b. SLAM, Localization, and Navigation

The biruni_navigation package contains all components required for mapping, localization, and autonomous navigation. It is organized into three major subsystems:

Mapping: The mapping.launch.py file launches the Cartographer SLAM framework [10], which constructs a 2D occupancy grid using LiDAR data. A dedicated RViz configuration is included for real-time visualization of scan matching and map generation.

Localization: The localization.launch.py script initializes the Adaptive Monte Carlo Localization (AMCL) algorithm [11] to estimate the robot's pose on the pre-generated map. All required parameter files and an RViz configuration are automatically loaded.

Navigation: The navigation.launch.py file integrates the Nav2 framework [12], enabling global and local path planning, costmap generation, recovery behaviors, and dynamic obstacle avoidance. A unified navigation parameter file defines the configuration of the global planner, local planner, controller server, and behavior tree nodes. RViz is launched alongside Nav2 for visual feedback.

All three subsystems rely on an Extended Kalman Filter (EKF) implementation provided by the robot_localization package [13], which fuses wheel odometry, IMU data, and LiDAR-derived pose estimates to improve state estimation accuracy.

The operational workflow follows four stages:

1. Spawn simulation: biruni_fusion.launch.py initializes the robot and environment in Gazebo Fortress.

2. Mapping: mapping.launch.py constructs the global 2D occupancy map using Cartographer.

☞

3. Localization: localization.launch.py validates pose estimation using AMCL.

4. Navigation: navigation.launch.py activates the Nav2 stack for autonomous motion and obstacle avoidance.

Through this architecture, Biruni achieved stable mapping performance and consistent autonomous navigation in complex simulated environments, successfully detecting and avoiding obstacles while following planned trajectories.

4. RESULTS AND DISCUSSION

This section presents the outcomes of the Biruni robot's mapping, localization, and autonomous navigation tests conducted entirely in a single simulated playground-style environment. The world included open spaces and simple structures such as slides and fixed obstacles, making it suitable for evaluating basic SLAM and navigation behaviors without the complexity of crowded or narrow environments.

Figures 1 and 2 provide an overview of the simulation environment and the Biruni robot model, including the sensors used for mapping and navigation.

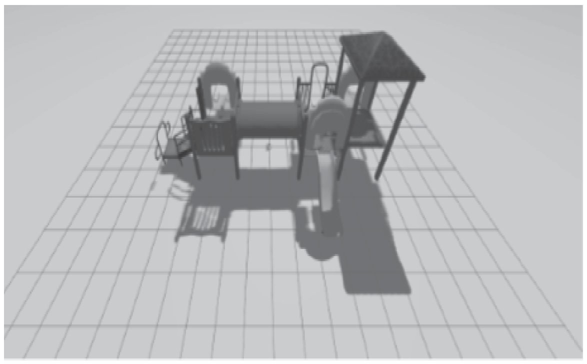
Figure 1. The world and robot spawn in Gazebo

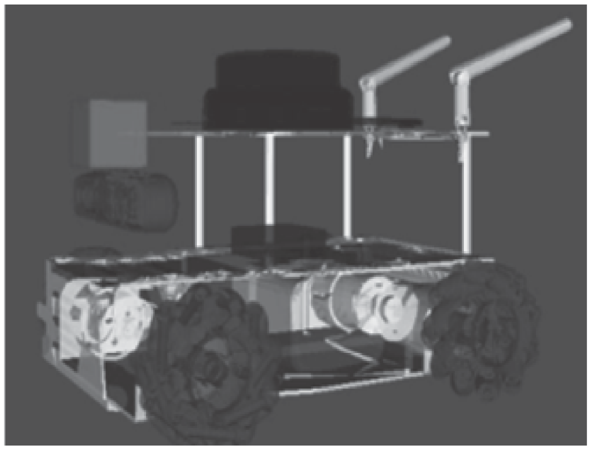
Figure 2. Clear robot model with all sensors

a. Mapping Performance

Mapping was performed in the playground environment using the Cartographer SLAM framework [10]. Two independent mapping sessions were executed following similar exploration paths. The resulting occupancy grids showed:

• Accurate reconstruction of the playground layout, with clear representation of structural boundaries and static elements.

• Stable LiDAR scan alignment, resulting in clean wall outlines with no significant drift.

• High repeatability between the two sessions, with only minor variations due to differences in the robot's driving trajectory.

Figures 3 and 4 illustrate the mapping process and the resulting occupancy grid as visualized in RViz. Listing 1 shows the automatically generated YAML configuration file corresponding to the map, which is used by the ROS 2 map_server node to load the occupancy grid.

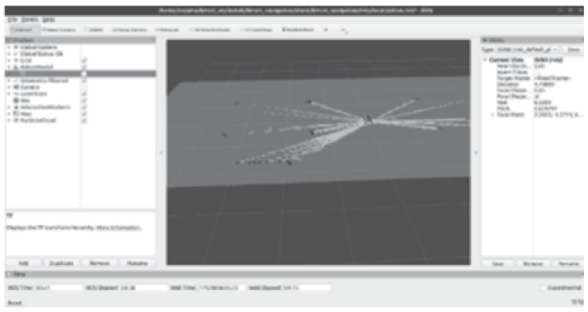
Figure 3. Partial map generated during the Cartographer mapping process

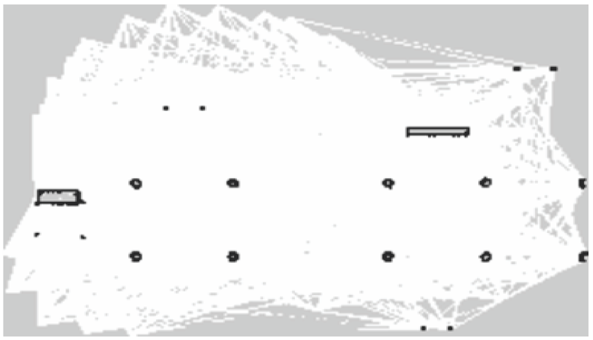
Figure 4. Final occupancy grid of the playground environment

Listing 1. Map YAML configuration used in Gazebo for the playground world.

```
image: my_map.pgm
mode: trinary
resolution: 0.05
origin: [-3.22, -5.67, 0]
negate: 0
occupied_thresh: 0.65
free_thresh: 0.25
```

Cartographer successfully generated a consistent and accurate map of the playground, producing an occupancy grid suitable for both localization and navigation. No loop closure events occurred, and the SLAM performance matched expectations for this world.

b. Localization Accuracy

Localization was evaluated using AMCL [11] on the previously generated map. In this environment:

• The robot achieved fast and reliable initial localization, typically converging within a few seconds.

• Pose estimation remained stable during navigation, as the environment provided sufficient geometric cues for consistent particle distribution.

• Only minor fluctuations occurred during rapid rotations, which is expected when relying on LiDAR-only observations.

Figures 5-7 illustrate the localization process: Figure 5 shows the robot in RViz on the map before the estimated pose was set; Figure 6 shows the robot immediately after setting the estimated pose, with the particle arrows displayed around it; and Figure 7 shows the robot with the particle arrows and obstacles highlighted with a purple aura, representing obstacle detection.

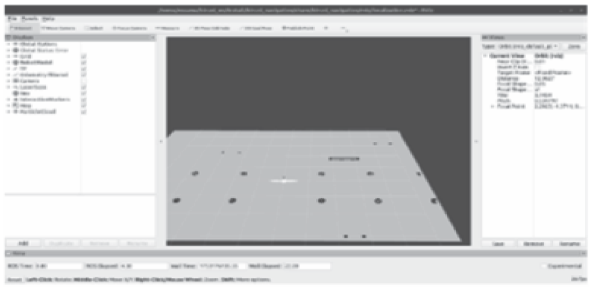
Figure 5. Robot in RViz before setting the estimated pose

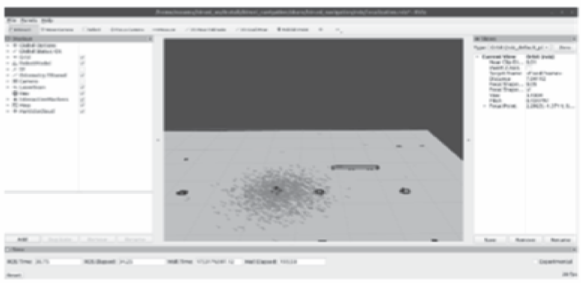
Figure 6. Robot after setting the estimated pose, with particle arrows displayed

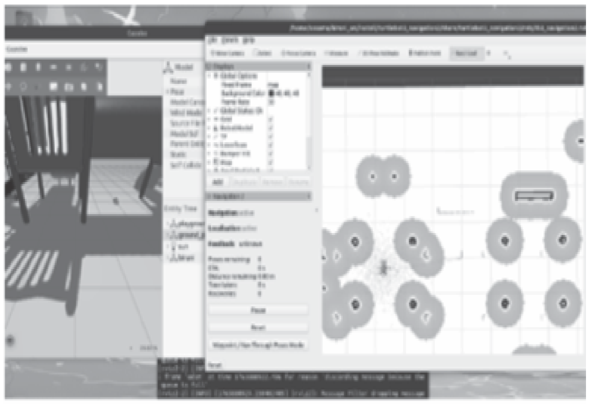

Figure 7. Robot with particle arrows and obstacles highlighted with a purple aura

AMCL performed robustly and maintained a consistent estimate of the robot's pose throughout all test runs.

c. Navigation Performance

Autonomous navigation was performed using the Nav2 framework [12], with goals placed at different locations within the playground. Navigation tests demonstrated:

Successful path planning on the generated map, with smooth global paths.

Accurate trajectory following, supported by the mecanum drive's ability to perform lateral and diagonal adjustments.

Stable movement behavior, with no erratic oscillations or path divergence during standard motion.

Figures 8-12 illustrate the robot's movement in RViz while navigating in Gazebo, from the start position until reaching the final goals.

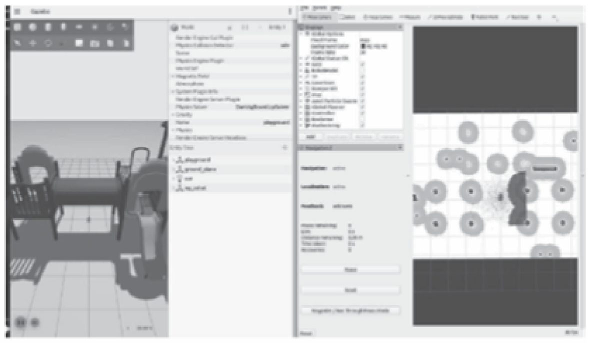

Figure 8. Robot starting navigation in RViz

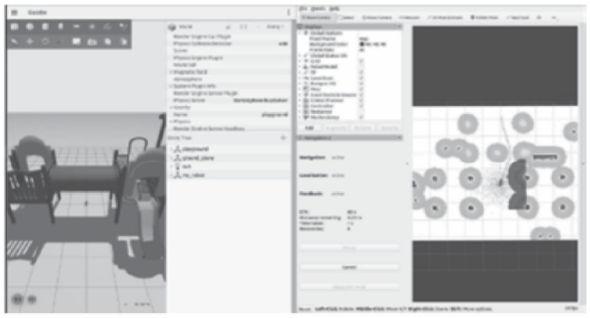

Figure 9. Robot moving along the planned path

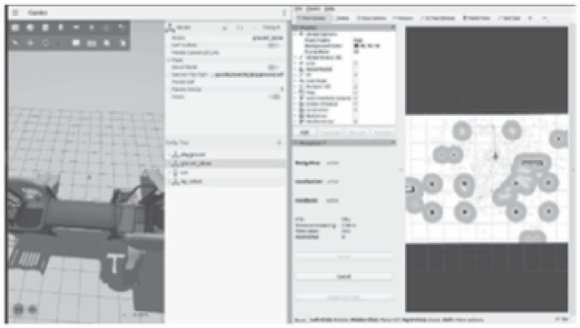

Figure 10. Robot performing a lateral adjustment

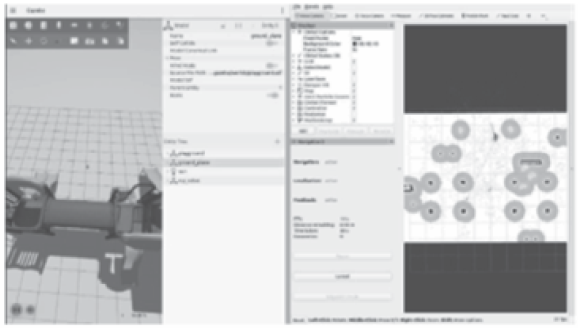

Figure 11. Robot approaching the goal

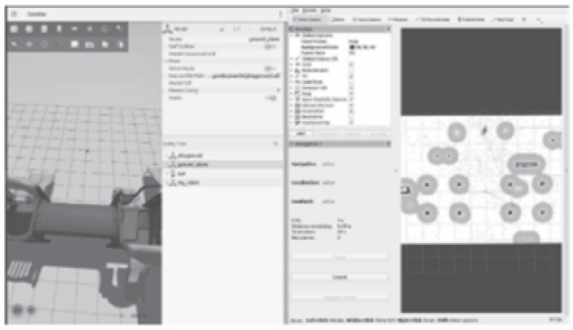
Figure 12. Robot at the final goal position

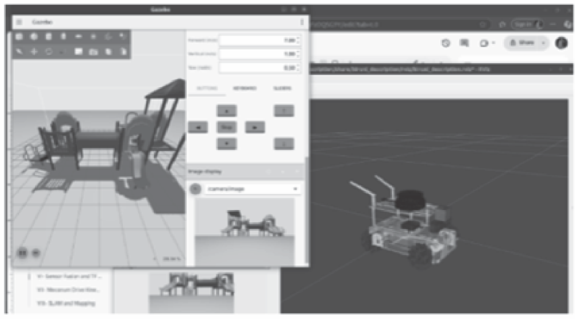
Figure 13. RGB-D camera image from the Intel RealSense D435 visualized in RViz

Overall, the robot navigated reliably and smoothly throughout all test runs, demonstrating robust path following and stable control under the Nav2 framework.

d. Obstacle Interaction

Obstacle avoidance was evaluated using the fixed structures present in the playground world. Results showed:

• Consistent detection of playground equipment using the 2D LiDAR.
• Correct inflation of obstacles on the local costmap.
• Smooth avoidance maneuvers, even though the environment lacked challenging obstacle distributions.

Dynamic obstacle testing was not performed. However, the Intel RealSense D435 RGB-D camera was successfully integrated, and ROS 2 correctly published image data from all sensors. Figure 13 shows the RGB-D image stream as visualized in RViz, confirming proper sensor operation and image transfer during navigation.

Overall, the system reliably detected and avoided static obstacles, ensuring safe navigation within the playground.

e. System Integration Stability

Across all launch sequences (SLAM, localization, and navigation), the system demonstrated:

• Stable multi-node execution, with no crashes or communication failures.

• Reliable sensor fusion using the EKF [13].

• Predictable behavior during repeated sessions, confirming the correctness of the launch and configuration structure.

To verify that the EKF was correctly fusing odometry, IMU, and LiDAR inputs, the TF tree was inspected during operation. Figure 14 illustrates the full transform structure, confirming continuous publication of the map → odom → base_link chain and proper alignment of all sensor frames, ensuring consistent and stable state estimation throughout the system.

The TF tree in the Biruni robot is composed of the following main links:

☞

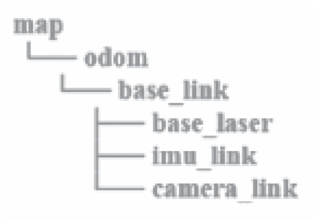

Figure 14 shows TF tree of the Biruni robot during operation, showing correct connectivity of all sensor frames and the EKF-generated transforms.

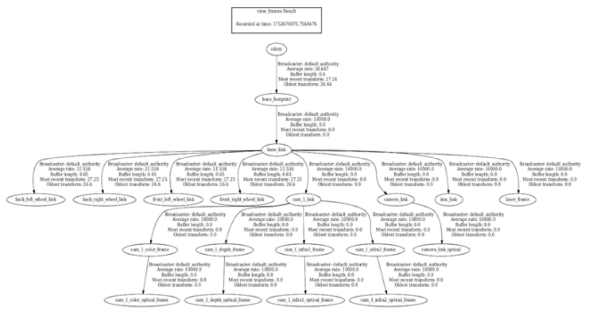
Figure 14. TF tree graph during operation

The modularity of the ROS 2 architecture allowed each phase – mapping, localization, and navigation – to be executed independently or in sequence, improving the clarity, repeatability, and reliability of the testing process.

5. CONCLUSION

This work presented the development and evaluation of Biruni, a four-wheeled mecanum autonomous mobile robot implemented in ROS 2 Humble and simulated in Gazebo Fortress. The system was built entirely using open-source tools, including a custom URDF/Xacro description, a structured simulation workflow, and a complete navigation pipeline integrating SLAM, localization, sensor fusion, and autonomous navigation.

Testing was performed in a playground-style simulation environment. Cartographer SLAM successfully produced a stable 2D occupancy grid suitable for downstream tasks, while AMCL provided reliable localization across multiple trials. Using the Nav2 framework, the robot achieved consistent goal-directed navigation and robust obstacle avoidance relying solely on 2D LiDAR data. The Intel RealSense D435 RGB-D camera was fully integrated at the sensing level, although its information was not yet incorporated into navigation.

Overall, the system demonstrated a robust and modular architecture capable of performing all fundamental behaviors required of an autonomous mobile robot. Even within a simple environment, Biruni achieved stable mapping, accurate localization, and reliable navigation, validating both the robot model and the ROS 2-based software stack. This work represents a complete end-to-end implementation and establishes a solid foundation for future extensions, including operation in more complex environments, integration of RGB-D perception, and deployment on physical hardware.

6. FUTURE WORK

Several extensions are planned to further enhance the capabilities of Biruni and support the transition from simulation to real-world deployment:

Integration of RGB-D Perception: Future versions will incorporate depth information from the Intel RealSense D435 into obstacle detection, local planning, and scene understanding. This includes experimenting with voxel-based costmaps, layered obstacle representations, and depth-assisted localization.

Operation in Complex and Dynamic

Environments: Testing will be expanded beyond the simple playground world to include larger indoor layouts, cluttered environments, and dynamic obstacles. These scenarios will allow evaluation of loop closure performance, global planning robustness, and navigation stability under more challenging conditions.

Real-World Hardware Deployment: The entire software stack will be deployed on a physical Biruni robot, including sensor calibration, mecanum wheel tuning, and validation of odometry accuracy. Special attention will be given to SLAM performance and localization drift under real sensor noise.

Advanced Localization and Mapping: Future work will explore multi-sensor fusion methods such as LiDAR-IMU SLAM, visual SLAM, and multi-modal EKF configurations to improve robustness in sparse-feature or dynamic environments.

Improved Autonomy and Higher-Level Behaviors: Additional autonomous capabilities -such as multi-goal navigation, autonomous exploration, and behavior-tree-based decision-making- will be investigated to enable more advanced robot behaviors.

Performance Optimization and Codebase Refinement: Further optimization of the robot model, launch configuration, and parameter tuning will be carried out to improve computational efficiency, simulation stability, and reproducibility across different ROS 2 systems.❖

References:

[1]. A. Zeidis, "*Investigation of the kinematics and dynamics of mecanum-wheel mobile platforms*". Scientific Journal of Riga Technical University, 2019. [Online]. Available: https://www.db-thueringen.de/servlets/MCRFileNodeServlet/dbt_derivate_00049732/1521-4001_99_2019_12_e201900173.pdf

[2]. A. Moreno-Caireta, "*Model predictive control for a mecanum-wheeled robot navigating among obstacles*". ISA Transactions, 2021. [Online]. Available: Model Predictive Control for a Mecanum-wheeled Robot Navigating among Obstacles - ScienceDirect

[3]. Y. Ding et al., "*ROS2 real-time performance optimization and evaluation*". Chinese Journal of Mechanical Engineering, 2023. [Online]. Available: https://cjme.springeropen.com/articles/10.1186/s10033-023-00976-5

[4]. T. Kronauer et al., "*Latency analysis of ROS2 multi-node systems*". arXiv preprint, 2021. [Online]. Available: https://arxiv.org/abs/2101.02074

[5]. W. Hess, D. Kohler, H. Rapp, and D. Andor, "*Real-time loop closure in 2D LiDAR SLAM*". In: Proc. IEEE Int. Conf. Robotics and Automation (ICRA), 2016. [Online]. Available: https://google-cartographer.readthedocs.io

[6]. Open Robotics, "*ROS 2 Humble Hawksbill documentation*", 2022. [Online]. Available: https://docs.ros.org/en/humble/

[7]. GazeboSim Community, "*Gazebo Fortress (Ignition) documentation*", 2021. [Online]. Available: https://gazebosim.org/docs/fortress/

[8]. Intel Corporation, "*Intel RealSense SDK (librealsense)*", 2019. [Online]. Available: https://github.com/IntelRealSense/librealsense

[9]. ROS Simulation Team, "*ros_gz_bridge: Gazebo–ROS 2 transport bridge*", 2023. [Online]. Available: GitHub - gazebosim/ros_gz: Integration between ROS (1 and 2) and Gazebo simulation

[10]. Cartographer Authors, "*Cartographer ROS integration*". Google, 2016. [Online]. Available: https://google-cartographer.readthedocs.io

[11]. D. Fox, "*Adapting the sample size in particle filters through KLD-sampling*". International Journal of Robotics Research, 2003.

[12]. S. Macenski, M. Fisher, C. Vargas, and F. Martín, "*The Marathon 2: A navigation system*". IEEE Robotics & Automation Magazine, vol. 29, no. 2, pp. 92-105, 2022.

[13]. T. Moore and D. Stouch, "*A generalized extended Kalman filter implementation for the Robot Operating System*". In: Proc. 13th Int. Conf. Intelligent Autonomous Systems (IAS), 2014.