

Mobile Robot Retrofit for the ROS2 Framework

Bruno Pinto da Cunha Lima^{1*}, Victor Santos Matos¹

¹SENAI CIMATEC University; Salvador, Bahia, Brazil

This article presents the retrofit of a TurtleBot2i mobile robotic platform to support the ROS2 framework. The motivation for this migration lies on the discontinuation of ROS1 and the growing need for educational and research platforms to align with current technologies. The retrofit process encompassed both hardware and software adaptations: the robot was upgraded with a new power system and additional sensors, including a Velodyne LiDAR and an Intel RealSense camera, while its software environment was updated to Ubuntu 22.04 and ROS2 Humble, with the robot model and drivers adapted to ensure full integration. The results confirm the platform's compatibility with the ROS2 ecosystem, enabling advanced perception and SLAM algorithms. These improvements enhance the applicability of the TurtleBot2i in educational and research contexts, ensuring its continued relevance in mobile robotics development.

Keywords:Robotics. ROS2. Retrofit. TurtleBot2i.

The TurtleBot2i is used in educational and research contexts due to its versatility, open architecture, and active developer community [1]. These factors have established the TurtleBot as an important tool in robotics education and scientific experimentation.

In this context, the platform is based on the Robot Operating System (ROS), an open-source framework that provides libraries and tools for the development of robotic applications, promoting interoperability, modularity, and reusability. ROS has been widely adopted in the robotics community for facilitating the integration of sensors, actuators, and algorithms into complex robotic systems.

However, given the continuous advancement of technology, modernizing legacy platforms becomes an alternative to maintain compatibility with current tools. Regular updates are essential to ensure compatibility with current technologies and standards. In this scenario, it is worth noting that the mobile platform version TurtleBot2i was officially archived in 2022 and no longer receives updates [2].

Despite its discontinuation, the TurtleBot2 remains a reference in academic research, as evidenced by several studies. The work of [3] implemented potential field control on the TurtleBot2 using ROS Melodic and Ubuntu 18.04. The study in [4] developed an indoor navigation system, also in ROS1. The research in [5] applied a visual simultaneous localization and mapping (SLAM) algorithm for autonomous navigation, integrating neural networks as real-time object detectors based on an RGB-D camera, also operating in ROS1 Melodic.

All of these studies share the characteristic of using a platform based on an outdated software ecosystem, built on ROS1 and older versions of the Ubuntu operating system. It is important to highlight that, according to [6], ROS1 has been discontinued and replaced by ROS2, with a recommendation for use in new developments.

ROS2 represents a significant evolution compared to ROS1 [7], incorporating substantial improvements in critical areas such as node-to-node communication, security, support for real-time systems, integration with multiple robots, and operation in distributed environments. While ROS1 was essential in standardizing robotic development, ROS2 provides a more robust, scalable foundation prepared for the contemporary challenges of robotics.

The migration of platforms from ROS1 to ROS2 involves the concept of technological retrofit,

Received on 12 April 2026; revised 11 July 2026.

Address for correspondence: Bruno Pinto da Cunha Lima.
Av. Orlando Gomes, 1845 - Piatã, Salvador – BA – Brazil,
Zipcode: 41650-010. E-mail: cunhalimabruno@gmail.com.

J Bioeng. Tech. Health 2026;9(8):817-823
© 2026 by SENAI CIMATEC University. All rights reserved.

which is a strategy aimed at extending the lifespan of existing robotic systems by modernizing their software components [8] and, in some cases, also their hardware [9]. This approach allows for minimizing the disposal of still-functional hardware while enabling the incorporation of more recent technologies into already established platforms. This was exemplified in the study by Eon et al. (2020) [10], which describes the retrofit of a humanoid robot, maintaining its original physical structure but replacing the controller and performing electronic and mechanical upgrades.

The case study by Lyman and colleagues (2019) [9] demonstrates how obsolete robots from the 1980s were repurposed with new control systems and modern interfaces, allowing their use in practical robotics and automation courses. This approach not only reduces costs but also provides students with direct access to the internal components of the robots, promoting deeper learning aligned with industry demands. Similarly, Petrone and colleagues (2025) [11] presented the use of the UR10 manipulator in ROS2, showing its control architecture without physical modifications, as a modernization strategy compatible with current industrial applications.

This article details the retrofit of the TurtleBot2i for full integration with ROS2, aiming to modernize its software ecosystem while promoting sustainable practices through the reuse of functional hardware as a strategy aligned with contemporary needs in robotics education and research.

The remainder of this paper is organized as follows. Section 2 introduces the TurtleBot2i robotic platform. Section 3 details the methodology of the retrofit process, covering both hardware and software adaptations. Section 4 presents the results obtained with the upgraded platform, highlighting its validation in ROS2. Finally, Section 5 concludes the paper by summarizing the main contributions and outlining directions for future work.

The TurtleBot2i Robotic Platform

The TurtleBot2i platform employed in this study was originally configured with an Intel NUC6CAYH

computer, running Ubuntu 16.04 LTS and the ROS1 Kinetic framework [12]. Both software components have been officially discontinued and no longer receive security updates or technical support.

In addition to its mobile base, the platform integrates the Pincher MK3 robotic arm, featuring four degrees of freedom (DOF) [13], which enables the manipulation of small objects. For perception and visual data acquisition, the system utilizes two depth cameras: the Orbbec Astra and the RealSense SR300, as illustrated in Figure 1.

Given these characteristics, the TurtleBot2i can be classified as a legacy system, which imposes significant limitations on scalability and maintainability. The obsolescence of its software environment hinders compatibility with modern libraries, integration with emerging technologies, and adherence to current security protocols. These constraints negatively impact the platform's lifecycle and its potential for reuse in contemporary research and educational contexts.

Materials and Methods

The retrofit process of the TurtleBot2i for ROS2 involved adaptations in both the hardware and software of the platform.

The method adopted for the retrofit followed a structured process divided into three main stages.

Figure 1. TurtleBot2i operating with ROS1.



First, the limitations of the original TurtleBot2i configuration were analyzed, identifying obsolete components and opportunities for modernization. Based on this assessment, the retrofit was planned in two complementary dimensions: hardware and software. The hardware stage focused on adapting the power supply and expanding the platform's perception capabilities with additional sensors, while the software stage addressed the use in ROS2, the adaptation of robot descriptions, and the integration of drivers and communication structures. Finally, the upgraded system was validated through functional tests to ensure proper operation and compatibility with the ROS2 framework.

Hardware

The physical retrofit of the TurtleBot2i involved modifications to its base to accommodate a new power configuration. A custom support structure was designed using CAD software (Figure 2) and subsequently fabricated via 3D printing to house a lithium-ion battery (model NH2054, 14.4V), selected for its capacity to meet the energy demands of the upgraded system (Figure 3).

As previously mentioned, the Astra depth camera was originally integrated into the TurtleBot2i platform. During the retrofit process, two additional sensors were incorporated: the Velodyne VLP-16 LiDAR and the Intel RealSense D435 camera, significantly enhancing the robot's perception capabilities.

To ensure efficient communication and adequate power distribution among all components including the Intel NUC mini-PC, lithium-ion battery, Astra camera, Velodyne LiDAR, RealSense D435, and the Kobuki mobile base, the system was reconfigured with a focus on integration and reliability.

Figure 4 shows the diagram of the connections and components of the robot's new configuration.

Software

Aiming at its modernization and extension of service life, the platform was updated to run the Ubuntu 22.04 operating system, chosen for its

Figure 2. 3D design of the mount for the new battery.

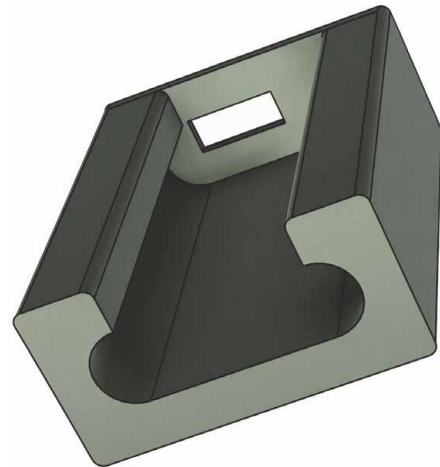
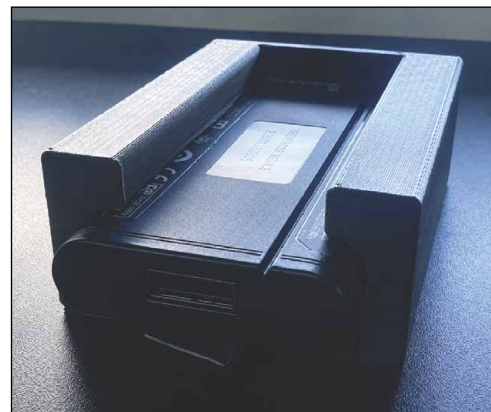


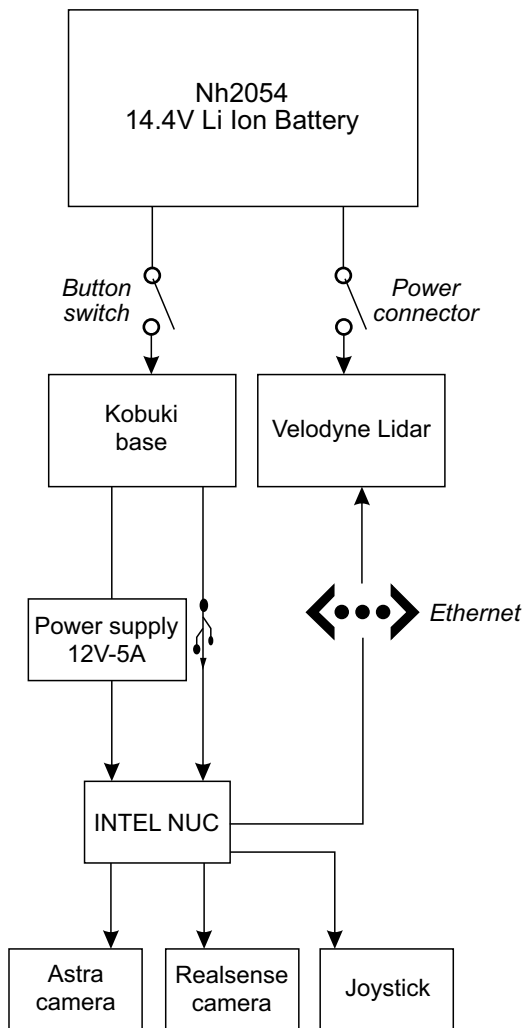
Figure 3. Battery installed in the printed mount.



stability and compatibility with ROS2, specifically the Humble distribution.

The architectural changes introduced by ROS2 were fundamental to this retrofit. While ROS1 uses a centralized system with a ROS Master, which can be a single point of failure and has limitations in communication, ROS2 adopts a decentralized, peer-to-peer communication model based on the Data Distribution Service (DDS). This architecture offers significant improvements in scalability, security, and real-time capabilities [14].

Adapting the robot's 3D model was a crucial step to enable accurate system visualization and facilitate debugging within the ROS2 environment. The old URDF (Unified Robot Description Format) model was converted to ROS2 format and updated

Figure 4. Connection diagram.

to include the new sensors, ensuring correct representation in both simulation and visualization.

During the migration, some adjustments were also required in the communication structure. One important change was the replacement of XML-based launch files used in ROS1 by Python-based launch files in ROS2. This modification allowed a more flexible configuration of nodes, parameters, and remappings, improving system maintainability, making the platform more robust for both educational and research applications compared to the previous version.

The retrofit also required the installation and configuration of new ROS2 packages to support the added sensors. For instance, Velodyne LiDAR integration relied on dedicated drivers and point

cloud processing packages, while Intel RealSense support was enabled through specific ROS2 packages maintained by the community. These additions expanded the set of published topics, especially those related to depth perception and mapping, and demanded updates in the robot's URDF description to correctly integrate the new devices. It is worth noting that the Kobuki mobile base already had ROS2-compatible drivers [15], which served as the foundation for building the TurtleBot2i software in ROS2, with the necessary configurations developed to integrate the additional hardware.

Validation

The objective of this validation was to verify whether the retrofitted TurtleBot2i could operate correctly within the ROS2 framework, ensuring the proper integration of both legacy and newly added components. The evaluation emphasized confirming sensor compatibility, consistency of published data, and the robot's ability to perform essential tasks such as perception and mapping. This stage sought to demonstrate that, after modernization, the platform remains suitable for educational and research purposes.

Rather than conducting a quantitative assessment mapping algorithm performance, the validation focused on functional aspects to confirm the readiness of the system for use in ROS2. The key objective was to ensure full interoperability of all sensors and the Kobuki mobile base within the new software architecture, thereby validating the viability of the retrofit for advanced applications. The generated map served as practical evidence of the robot's perception and modeling capabilities, reinforcing that the modernization successfully reestablished the platform's utility in contemporary research and education contexts, while also opening possibilities for more detailed investigations of precision and error in localization and mapping systems.

The validation was carried out in laboratory conditions. The procedure involved teleoperating

the robot to generate maps using SLAM algorithms, verifying the publication of sensor data, and rendering the updated URDF model in ROS2 visualization tools. Compatibility of the Kobuki base, depth cameras, and LiDAR sensor was evaluated through parameter configuration and topic monitoring to ensure proper communication in ROS2.

The robot successfully publishes data from all integrated sensors including depth cameras, the LiDAR sensor, and odometry enabling robust environment perception and modeling. These capabilities support applications such as localization, mapping, autonomous navigation, and interaction with the environment. Figure 5 illustrates the final configuration of the robot with all sensors integrated.

The system receives control inputs such as wheel speed commands and joystick-based teleoperation. Its outputs include odometry data, orientation and acceleration from the IMU, point clouds from the cameras, collision and fall detection from the Kobuki base, as well as status information from physical buttons, battery level, and general diagnostics. Ensuring compatibility of all peripherals with ROS2 required specific adaptations, including parameter reconfiguration and validation tests to confirm the consistency of data published by each device.

The robot's 3D model was successfully rendered in the ROS2 visualization environment (RViz), as shown in Figure 6. The updated model includes the robot's geometry and all embedded sensors, enabling simulation and analysis in a virtual environment. This required converting the original URDF file, correcting deprecated syntax, and incorporating the newly added sensors.

Additionally, the new configuration enabled the use of SLAM algorithms with both the original and newly integrated sensors. In laboratory tests, the robot was manually teleoperated to map an indoor environment, resulting in an accurate spatial representation, as shown in Figure 7. One of the main challenges during the retrofit was adapting all sensors to the ROS2 framework, particularly ensuring compatibility of the Astra camera,

Figure 5. Final TurtleBot2i configuration.

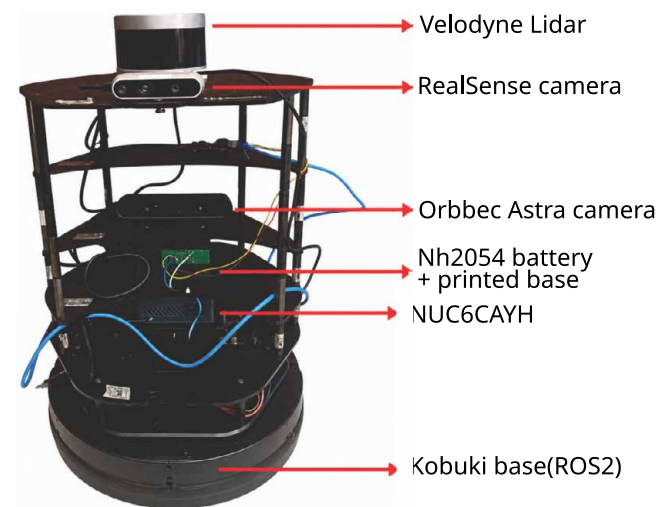


Figure 6. 3D model of the robot with integrated cameras and lidar sensor.

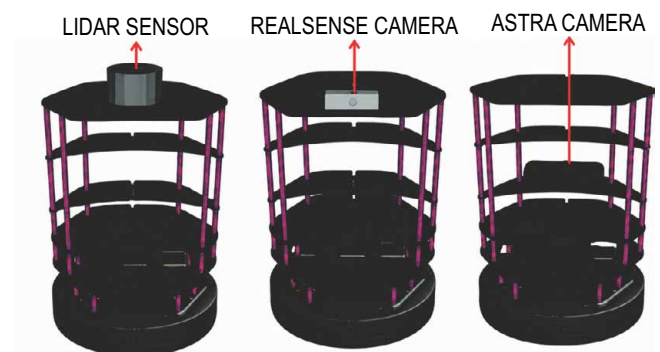
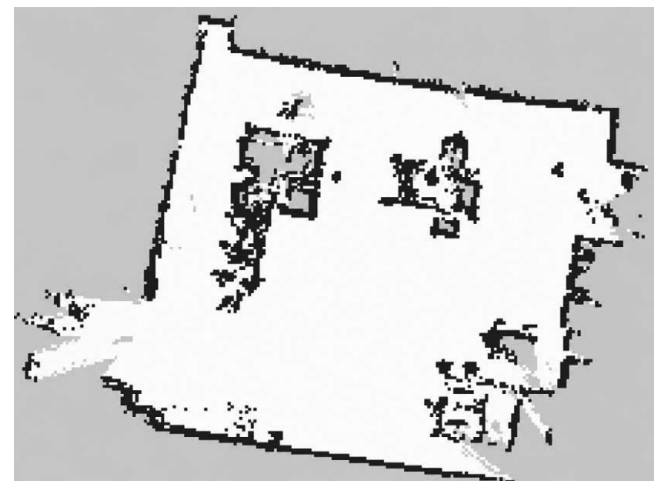


Figure 7. Environment map generated using the Velodyne LiDAR.



Velodyne LiDAR, and RealSense D435. This process involved extensive research and testing to guarantee seamless integration.

These results demonstrate that the retrofitted TurtleBot2i retains its capabilities for teleoperation, mapping, and localization, while gaining improved compatibility, scalability, and integration with modern ROS2 tools. The migration also enhances the platform's applicability in educational and research settings that demand up-to-date technologies and active community support.

Conclusion

The retrofit of the TurtleBot2i to ROS2 extended the robotic platform's lifespan and modernized its development environment, aligning it with current demands in research and education. The transition to ROS2 provided improvements in performance, security, and interoperability, while enabling the integration of additional sensors thereby enhancing its capabilities in autonomous navigation, mapping, and environmental interaction.

These advances reinforce the platform's potential to continue being used in academic and experimental contexts, even in light of the discontinuation of its original ROS1-based configuration. The update also contributes to technological sustainability by avoiding the disposal of functional hardware through the modernization of its software architecture.

As an outcome of this work, the use of the platform is proposed for experimentation with advanced techniques in perception, navigation, and coordination of mobile robots. Future research may explore its application in dynamic and unstructured environments, where safe and efficient operation is critical.

References

1. ROBOTIS. TurtleBot3 overview [Internet]. ROBOTIS e-Manual. Available from: ROBOTIS e-Manual – TurtleBot3 Overview; <https://emanual.robotis.com/docs/en/platform/turtlebot3/overview/>
2. Interbotix. TurtleBot2i [Internet]. GitHub. Available from: Interbotix – TurtleBot2i; <https://github.com/Interbotix/turtlebot2i>
3. Okura J, Teixeira B, Spelta C, Faccio FM. Controle por campos potenciais em robô TurtleBot2 usando ROS. In: Anais do XXIII Congresso Brasileiro de Automática (CBA); 2020; Salvador, Brasil. Salvador: SBA; 2020. Available from: IEEE Xplore; <https://ieeexplore.ieee.org/document/10349365>
4. Ferreira MF, Silva RS, Girelli CM. Navegação indoor autônoma com ROS para robôs móveis. In: Anais do Congresso de Engenharia de Aplicações; 2020; São Paulo, Brasil. São Paulo: CEA; 2020. Available from: IEEE Xplore; <https://ieeexplore.ieee.org/document/9039877>
5. Pinheiro JR, Almeida M, Vasconcelos M. SLAM visual com YOLOv3 em ROS1. In: Encontro Nacional de Inteligência Artificial; 2021; Curitiba, Brasil. Curitiba: ENIA; 2021. Available from: IEEE Xplore; <https://ieeexplore.ieee.org/document/10417221>
6. Open Robotics. ROS Noetic end of life announcement [Internet]. ROS Blog. Available from: ROS Noetic end of life announcement; <https://www.ros.org/blog/noetic-eol/>
7. RoboticsBackend. ROS1 vs ROS2: practical overview [Internet]. RoboticsBackend. Available from: ROS1 vs ROS2 – Practical Overview; <https://roboticsbackend.com/ros1-vs-ros2-practical-overview/>
8. InfoQ Brasil. Migrando do ROS1 para o ROS2: experiências do Autoware e da Rover Robotics [Internet]. InfoQ; 2020. Available from: InfoQ Brasil; <https://www.infoq.com/br/news/2020/02/migrating-ros1-ros2/>
9. Lyman G, Wilcox J, Sanford R. Robotics retrofit: renovating outdated robotics platforms to meet current curriculum requirements driven by industry demand. In: Proceedings of the 2019 Conference for Industry and Education Collaboration; 2019; Washington, DC. Washington: American Society for Engineering Education; 2019. p. 1-10.
10. Eon J, Rohmer E, Laliberté T. Humanoid robot retrofit: controller and mechanical modernization. J Integr Des Process Sci [Internet]. 2020;24(1):45-58. Available from: Taylor & Francis; <https://www.tandfonline.com/doi/full/10.1080/10255842.2020.1714997#d1e159>
11. Petrone L, Souza MR, Almeida C. ROS2 control para manipulador UR10 em ambientes industriais. In: Simpósio Brasileiro de Automação Inteligente (SBAI); 2025; Florianópolis, Brasil. IEEE; 2025. Available from: IEEE Xplore; <https://ieeexplore.ieee.org/document/10886909>
12. Interbotix. TurtleBot2i installation guide [Internet]. GitHub. Available from: TurtleBot2i Installation Guide; <https://github.com/Interbotix/turtlebot2i/blob/master/INSTALLATION.md>

13. TurtleBot. TurtleBot 2 platform [Internet]. TurtleBot.com. Available from: TurtleBot 2 Platform; <https://www.turtlebot.com/turtlebot2/>
14. Open Robotics. Changes between ROS 1 and ROS 2 [Internet]. ROS 2 Design. Available from: Changes between ROS 1 and ROS 2; <https://design.ros2.org/articles/changes.html>
15. Kobuki developers. kobuki_ros2 package repository [Internet]. GitHub. Available from: Kobuki ROS repository; https://github.com/kobuki-base/kobuki_ros2