Implementation of Slam Using Mobile Robot in ROS2

Atchaya H[1], Jeevitha J[2], Varshini A[3], Dr.M.S.Suresh Kumar[4]
[1,2,3]*UG-Scholars, (Sl. Gr), Department of Robotics and Automation, Sri Ramakrishna Engineering College, Coimbatore, India*
[4]*Assistant Professor (Sl. Gr), Department of Robotics and Automation, Sri Ramakrishna Engineering College, Coimbatore, India*

Abstract— For autonomous mobile robots to effectively navigate and understand unfamiliar environments, a fundamental capability is Simultaneous Localization and Mapping (SLAM). This project focuses on the implementation of a SLAM system using a mobile robot within the Robot Operating System 2 (ROS2) framework. The primary objective is to enable the robot to construct a real-time map of its surroundings while simultaneously estimating its precise position within that environment.

To achieve accurate localization and mapping, the system integrates sensor data obtained from LiDAR along with onboard odometry. The ROS2 architecture, known for its improved communication, scalability, and real-time performance, facilitates seamless interaction between various nodes responsible for sensing, mapping, and robot control. SLAM algorithms such as Gmapping and Hector SLAM are utilized to process sensor data and generate occupancy grid maps in real time.

Visualization and monitoring are carried out using tools like RViz2, allowing users to observe the robot's movement and map formation dynamically. The mobile robot platform, such as TurtleBot, is chosen due to its affordability, ease of integration, and strong compatibility with ROS2, making it ideal for research and educational purposes.

This project highlights the practical implementation of SLAM in modern robotics and provides a strong foundation for advanced applications such as autonomous navigation, path planning, and obstacle avoidance. Overall, the study demonstrates how ROS2-based SLAM systems significantly enhance the autonomy, adaptability, and intelligence of mobile robots. The modular design approach allows individual components—such as mapping, localization, and sensor processing—to be independently developed, tested, and optimized. This flexibility makes the system highly scalable and adaptable to different robotic platforms and application requirements.

In addition, performance evaluation of the SLAM system is carried out based on parameters such as mapping accuracy, computational efficiency, and localization stability. The system is tested in various indoor environments with different levels of complexity to validate its effectiveness. The results demonstrate that the ROS2-based SLAM implementation provides reliable and consistent performance, even in partially unknown or changing environments, thereby reinforcing its suitability for real-world autonomous robotic applications.

I.INTRODUCTION

Simultaneous Localization and Mapping (SLAM) is an important idea in robotics that allows a robot to move through unfamiliar environments without any prior data. This makes it possible for the robot to build a map and assess its own location—an essential capacity for real-world autonomous functioning. SLAM is essential in modern robotics for uses like autonomous vehicles, automating storage facilities, and servicing robots.

Robots can effectively understand their environment and make decisions by integrating sensor data with smart algorithms, thus improving efficiency and reducing reliance on human involvement. The TurtleBot platform is a popular choice for implementing SLAM because it is affordable, modular, and works with the Robot Operating System (ROS). It includes sensors like LiDAR, cameras, and wheel encoders that are crucial for gathering environmental and positional data. The Robot Operating System (ROS) offers a versatile framework for the development of robotic applications. It facilitates communication among various modules via nodes, topics, and services, simplifying the integration of SLAM algorithms such as Gmapping and Hector SLAM for mapping and localization purposes.

II. OBJECTIVE

The main aim of this project is to implement Simultaneous Localization and Mapping (SLAM) using the Robot Operating System (ROS) on a TurtleBot to enable autonomous navigation in unknown environments. The system is designed to allow the robot to generate a real-time map of its surroundings while simultaneously determining its position within that map. This project improves efficiency, accuracy, and safety in robotic applications by minimizing human intervention and enabling intelligent decision-making. By integrating sensors such as LiDAR and applying SLAM algorithms, the robot can perform autonomous exploration and navigation tasks. The system promotes the use of smart, user- friendly robotic technology for applications such as indoor mapping, surveillance, and service robotics.

The primary objective of this project is to design and implement a Simultaneous Localization and Mapping (SLAM) system using a mobile robot within the ROS2 framework. The system aims to enable the robot to generate a real-time map of an unknown environment while accurately estimating its own position within that space. To achieve this, the project focuses on integrating sensor data from LiDAR and onboard odometry to ensure precise mapping and localization. It also utilizes the modular and efficient communication capabilities of ROS2 to coordinate various functional nodes responsible for sensing, mapping, and control. Additionally, the project involves implementing SLAM algorithms such as Gmapping and Hector SLAM to generate occupancy grid maps, along with visualization tools like RViz2 for real-time monitoring. The overall objective is to evaluate the system's performance in terms of accuracy, efficiency, and robustness, while establishing a strong foundation for advanced robotic applications such as autonomous navigation, path planning, and obstacle avoidance. In addition to the core implementation, another objective of this project is to enhance the reliability and adaptability of the SLAM system in dynamic and partially unknown environments.

III. BLOCK DIAGRAM

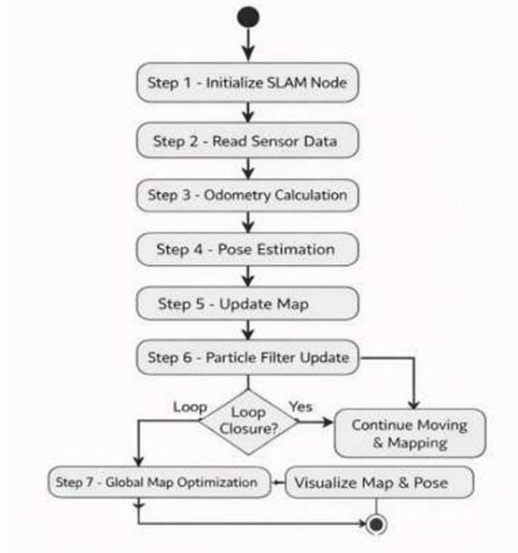

Fig. 1. Block diagram of the implementation of slam using mobile robot in ros2

The flowchart represents the working process of a SLAM algorithm in a mobile robot. It begins with initializing the SLAM node, followed by reading sensor data such as LiDAR and odometry inputs from the robot. This data is then used to calculate odometry and estimate the robot's current pose (position and orientation). Based on this estimated pose, the system updates the map of the environment. A particle filter is then applied to refine localization by handling uncertainty in sensor data. The system continuously checks for loop closure— if the robot revisits a known location, it corrects and optimizes the map for better accuracy; otherwise, it continues mapping and movement. Finally, global map optimization is performed, and the updated map along with the robot's pose is visualized, completing the SLAM cycle.

I. Raspberry Pi 5

The system executes SLAM and ROS 2 algorithms to enable the robot to simultaneously map its environment and determine its position, while ROS 2 provides a structured framework for handling nodes, topics, and real-time robotic operations. It processes data collected from sensors such as LiDAR, cameras, and IMUs, analyzes this information, and controls the robot's movement by sending appropriate commands to motors and actuators. It facilitates communication between different components of the robot by allowing

seamless data exchange between sensors, processing units, and control modules through ROS 2communication mechanisms like publishers, subscribers, services, and actions.

Fig. 2. Architecture of Raspberry Pi.

II. Raspberry pi Camera (Module v1)

The Raspberry Pi Camera captures high-quality images and videos, providing clear visual data that can be used for various applications such as photography, recording, and image processing. Itsupplies visual input for computer vision tasks, enabling the system to perform functions like object detection, recognition, tracking, and analysis in real- time environments. Additionally, it supports real-time monitoring and video streaming, making it highly useful in robotics, surveillance systems, and automation projects where continuous observation is required.

In addition to image capture and streaming, the Raspberry Pi Camera plays a crucial role in enhancing autonomous decision-making in robotic systems. By integrating with computer vision libraries such as OpenCV and ROS2-based vision nodes, the camera enables the robot to interpret its surroundings more intelligently. It can assist in tasks like obstacle detection, lane following, face recognition, and gesture analysis, thereby improving interaction with the environment.

Furthermore, the camera's compact size, low power consumption, and easy compatibility with embedded systems make it an ideal choice for real-time applications, ensuring efficient performance in resource-constrained robotic and automation platforms.

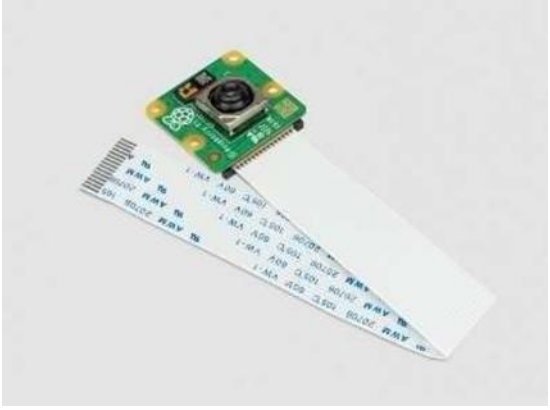

Fig. 3. Raspberry pi Camera (Module v1)

III. METHODOLOGY

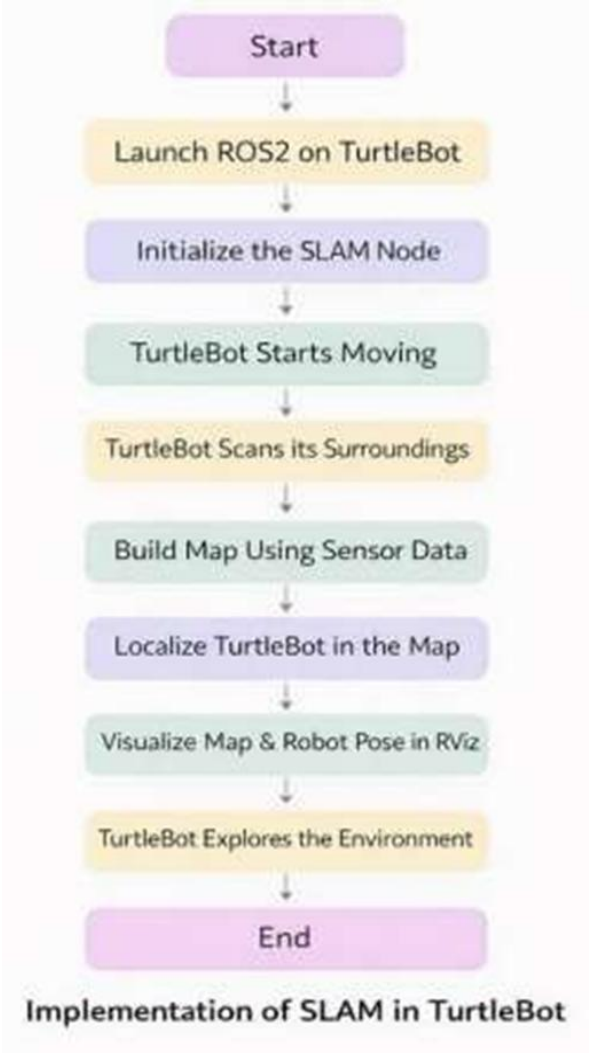

Fig. 4. Methodology to be adopted

IV. IMPLEMENTATION OF ALGORITHM

STEP 1: Start
The process begins by initializing the system and powering up all hardware components of the TurtleBot. Sensors, connections, and required modules are checked to ensure proper functionality. This step prepares the robot for executing the SLAM process without errors.

STEP 2 – Launch ROS2 on TurtleBot
ROS2 is launched to establish communication between various nodes in the system. It acts as the middleware that enables data exchange between sensors, processing units, and control modules. This step ensures a synchronized and efficient working environment.

STEP 3 – Initialize the SLAM Node
The SLAM node is started by loading the required algorithm such as Gmapping or Hector SLAM. Necessary parameters and configurations are set during initialization. This prepares the system to perform mapping and localization tasks.

STEP 4 – TurtleBot Starts Moving
The TurtleBot begins moving in the environment either manually or autonomously. Movement is essential to collect data from different areas. This allows the robot to explore and understand its surroundings.

STEP 5 – TurtleBot Scans its Surroundings
As the robot moves, sensors like LiDAR continuously scan the environment. These sensors collect real-time information about obstacles, distances, and free space. The collected data is used as input for SLAM processing.

STEP 6 – Build Map Using Sensor Data
The SLAM algorithm processes the sensor data to construct a map of the environment. It generates an occupancy grid map that represents obstacles and free areas. The map is continuously updated as new data is received.

STEP 7 – Localize TurtleBot in the Map
The robot estimates its current position and orientation within the generated map. This is done using odometry data combined with sensor inputs. Accurate localization helps the robot navigate effectively.

STEP 8 – Visualize Map & Robot Pose in RViz
The generated map and the robot's position are displayed in RViz for real-time visualization. Users can monitor the mapping progress and robot movement. This step is useful for debugging and performance evaluation.

STEP 9 – TurtleBot Explores the Environment
The robot continues to move and explore new areas of the environment. It refines the map and improves localization accuracy over time. This ensures better coverage and completeness of the map.

STEP 10 – End
Once sufficient mapping is completed, the SLAM process is terminated. The final map and robot trajectory are saved for future use. This data can be used for navigation, path planning, and obstacle avoidance tasks.

The overall methodology of implementing SLAM using a mobile robot in ROS2 involves a systematic process where the robot initializes its system and establishes communication through ROS2 to coordinate various functional nodes. The SLAM algorithm is then activated, enabling the robot to begin movement and collect real-time data from sensors such as LiDAR and odometry. The integration of mapping and localization ensures that the robot can accurately understand and navigate unknown environments. Visualization tools like RViz are used to monitor the robot's movement and map generation in real time. Throughout the process, the robot incrementally refines both its map and position through continuous exploration and data updates. Finally, the completed map is stored and can be utilized for advanced applications such as autonomous navigation, path planning, and obstacle avoidance, demonstrating the effectiveness of the ROS2-based SLAM system.

V. MOBILE ROBOT AND SENSOR INTEGRATION

The integration of a mobile robot with sensors plays a vital role in the implementation of Simultaneous Localization and Mapping (SLAM) using ROS2. In this system, the mobile robot is equipped with sensors such as LiDAR, cameras, and Inertial Measurement

Units (IMU), which continuously collect environmental data. These sensors are interfaced with the ROS2 framework, where nodes are used to handle data acquisition, processing, and communication between different components.

The robot acts as a data collection platform, while ROS2 serves as the middleware that ensures efficient and real-time communication among various modules. The sensor data is published to specific topics, and SLAM algorithms subscribe to these topics to process the information. This integration enables the robot to perceive its surroundings and estimate its position within the environment. Hence, the seamless coordination between the robot hardware and ROS2 software forms the foundation of an autonomous navigation system.

VI. RASPBERRY PI EDGE DETECTION PROCESS

The SLAM process in ROS2 is a crucial component that allows the mobile robot to build a map of an unknown environment while simultaneously determining its position within that map. Initially, sensor data such as laser scans or visual inputs are collected and preprocessed to remove noise and improve accuracy. These inputs are then fed into SLAM algorithms such as Cartographer or Gmapping. The SLAM algorithm identifies features from the sensor data and uses them to estimate the robot's pose. As the robot moves, it continuously updates the map by adding new observations and refining previous estimates. This iterative process involves techniques such as scan matching, loop closure detection, and pose graph optimization. Through these operations, the system reduces localization errors and improves map consistency.

Thus, SLAM enables the robot to operate autonomously in unknown environments by providing accurate localization and real-time map generation, which are essential for navigation and path planning.

VII. MAP GENERATION AND LOCALIZATION

Map generation and localization are core outputs of the SLAM process in ROS2. As the robot navigates through the environment, it constructs a two-dimensional or three- dimensional map representing obstacles and free space. This map is continuously updated based on incoming sensor data, ensuring that it reflects the current state of the environment.

Localization refers to the robot's ability to determine its position and orientation within the generated map. ROS2 utilizes transformation libraries such as TF2 to maintain relationships between different coordinate frames, including the map frame, odometry frame, and robot base frame. By combining sensor data and motion information, the system achieves accurate pose estimation.

VIII. VISUALIZATION AND OUTPUT DISPLAY IN ROS2

The output of the SLAM system is visualized using tools available in the ROS2 ecosystem, such as RViz2. RViz2 provides a graphical interface where users can observe the real-time map, robot position, sensor data, and movement trajectory. The visualization interface displays laser scans, occupancy grids, and coordinate frames, allowing users to monitor the SLAM process interactively. Additionally, the system can publish mapping data and robot states to web- based dashboards or monitoring tools for remote access.

This visualization capability improves system understanding, debugging, and performance evaluation. It also enables users to verify mapping accuracy and ensure proper robot operation in real-time scenarios. In addition to real-time visualization, the system also supports recording and playback of sensor data using ROS2 bag files. This feature allows developers to analyze previously captured data and evaluate system performance under different conditions without requiring continuous robot operation.

IX. NAVIGATION AND PATH PLANNING

Navigation is an essential extension of SLAM in ROS2, where the generated map is used for path planning and obstacle avoidance. The navigation stack in ROS2 processes the map and determines the optimal path from the robot's current location to a target position.

Using algorithms such as A* and Dijkstra, the system computes a collision-free path while considering dynamic obstacles. The robot then follows this path using control algorithms that adjust its motion based on sensor feedback.

This integration of SLAM with navigation allows the

robot to move autonomously in complex environments, making it suitable for applications such as warehouse automation, service robotics, and exploration tasks. Moreover, the navigation system incorporates localization feedback and dynamic obstacle handling to ensure safe and efficient robot movement. By continuously updating the robot's position using localization techniques such as Adaptive Monte Carlo Localization (AMCL).

VIII. VISUALIZATION AND MONITORING IN ROS2

The output of the object detection system is visualized using tools such as RViz2. RViz2 provides a graphical interface to display camera feed, detected objects, and robot movement in real time.

The system can also be integrated with web-based dashboards to display processed data remotely. The interface shows bounding boxes, object labels, and positional information, allowing users to monitor system performance. This improves debugging, analysis, and system validation.

Visualization plays a crucial role in understanding how the robot perceives its environment and ensures accurate detection and tracking performance.

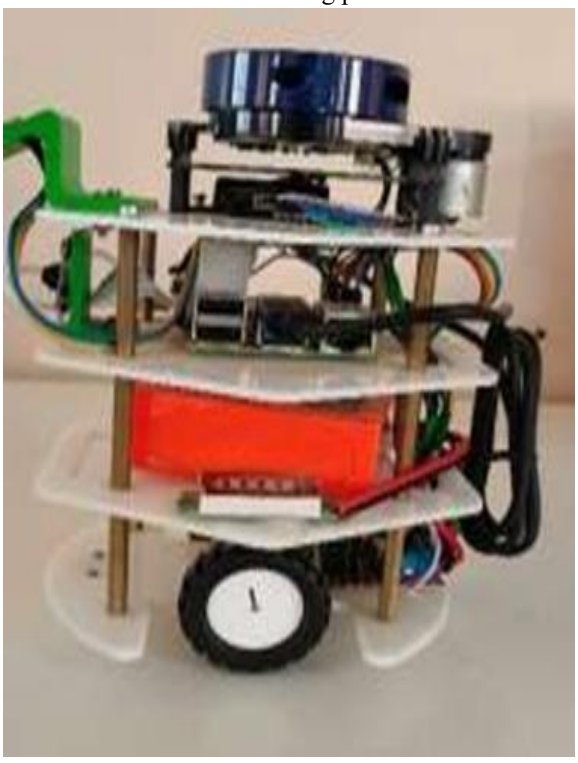

Fig. 5.Mobile robot

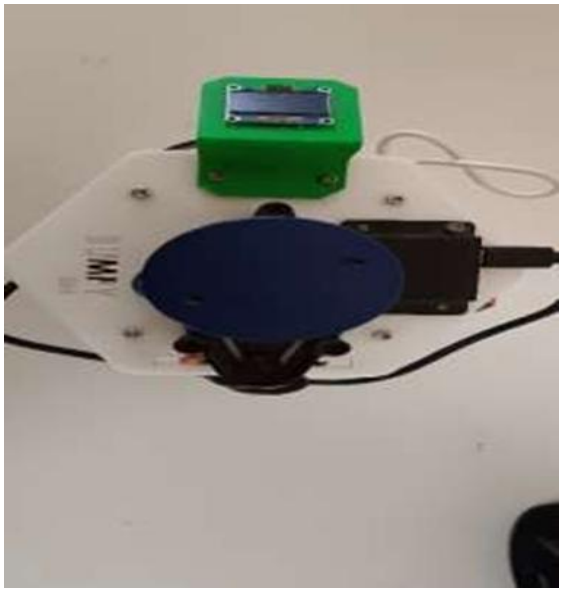

Fig. 6. Lidar

VI. CONCLUSION

From an academic perspective, this project can serve as a modular research platform for experimenting with different SLAM algorithms, navigation strategies, and control techniques. It encourages innovation and provides a foundation for further studies in autonomous robotics and intelligent systems. Finally, future developments can focus on optimizing computational efficiency and reducing power consumption to make the system more sustainable and suitable for embedded platforms. Lightweight algorithms and energy-efficient hardware will play a crucial role in advancing autonomous robotic systems.

REFERENCE

[1] Durrant-Whyte, H., & Bailey, T. (2006). Simultaneous Localization and Mapping (SLAM): Part I. IEEE Robotics & Automation Magazine.

[2] Thrun, S., Burgard, W., & Fox, D. (2005). Probabilistic Robotics. MIT Press.

[3] Grisetti, G., Stachniss, C., & Burgard,W.(2007). Improved techniquesfor grid mapping with Rao-Blackwellized particle filters. IEEE Transactions on Robotics.

[4] Kohlbrecher, S., Meyer, J., Graber, T., Petersen, K., & Klingauf, U. (2011). Hector SLAM for autonomous navigation.

[5] Hess, W., Kohler, D., Rapp, H., & Andor, D.

(2016). Real-time loop closure in 2D LiDAR SLAM. Google Cartographer.
[6] Quigley, M., et al. (2009). ROS: An Open-Source Robot Operating System. ICRA Workshop.
[7] Macenski, S., Foote, T., Gerkey, B., Lalancette, C., & Woodall, W. (2022). The ROS2 Navigation Stack (Nav2). IEEE.
[8] Zhang, J., & Singh, S. (2014). LOAM: LiDAR Odometry and Mapping in real time. Robotics Science and Systems.
[9] Fox, D., Burgard, W., & Thrun, S. (1999). Markov Localization for Mobile Robots.
[10] Open Robotics (2020). ROS2 Documentation and Tutorials.