2025 11th International Conference on Smart Computing and Communications (ICSCC)

Autonomous Indoor Navigator

Aleena Sojan , Cyril Thomas, Bhadra Raj, Sam Abraham Simon,Shajimon K John, Arukant A Jose
Muthoot Institute of Technology and Science, Ernakulam, Kerala
shajimonkjohn@mgits.ac.in

2025 11th International Conference on Smart Computing and Communications (ICSCC) | 979-8-3315-2562-0/25/$31.00 ©2025 IEEE | DOI: 10.1109/ICSCC66177.2025.11233830

***Abstract*—Autonomous navigation systems play a vital role in enabling mobile robots to navigate and operate efficiently in dynamic environments without human intervention. These systems integrate various technologies, including LiDAR-based mapping, sensor fusion, localization, path planning, and obstacle avoidance, to ensure accurate and safe movement. The core components include Simultaneous Localization and Mapping (SLAM) for map generation, Adaptive Monte Carlo Localization (AMCL) for real-time positioning, and the Navigation Stack (Nav2) for path planning and execution. Sensor inputs from LiDAR, encoders, and IMUs help in refining localization and odometry, while real-time obstacle detection mechanisms enhance navigation reliability.This work demonstrates a cost-effective and scalable approach to autonomous navigation, with future enhancements planned for multi-floor navigation, improved hardware integration, and product-level development.**

***Index Terms*—robot operating system, navigation, localization, path planning,mapping,LiDAR,object detection**

I. Introduction

Robotics is quickly developing for industrial applications and services, and guide robots are essential for indoor navigation in hospitals, museums, offices, and warehouses. The goal of this project is to create a cost-effective autonomous guide robot that can map, navigate, and guide users inside buildings. The robot uses a two-wheel drive platform with ROS 2 on a Raspberry Pi for top-level navigation and an Arduino Nano for motor control. ROS 2's modular design makes it easy to integrate different hardware and software modules, while Simultaneous Localization and Mapping (SLAM) builds an environment map, and NAV 2 does path planning and obstacle avoidance. For accurate navigation, the LiDAR sensor allows 360-degree scanning, while SPG30E-200K geared motors with encoders guarantee precise motion control.The design, hardware-software integration, mapping generation, and development challenges' solutions of the robot are discussed. Through tackling critical issues such as localization precision and obstacle evasion, this project provides an economical robotic solution. The fact that open-source software and commodity hardware are used makes the strategy viable for use in real situations. By this research, we hope to extend autonomous robotics with the creation of an affordable and viable guide robot for indoor spaces.

II. Motivation

This project on autonomous navigation using ROS can be built on in future projects and can aid the robotics community by publishing the changes we made. It is our hope, furthered discussed earlier, that this work will serve as a foundation to conduct more profound autonomous navigation research. Further, we were able to implement the project and subsequently understand deeply the developed algorithms implemented for the ROS navigation and trajectory planner developed by the pioneers in mobile robotics like Sebastian Thurn. Gaining this practical experience of working with ROS will be useful in our professional endeavors since ROS is becoming a standard framework for sophisticated robotic systems. With the knowledge gained from the project, we now appreciate a holistic view of a real life planning system, which broadens our minds and skills towards software engineering.

III. Related Works

ROS is an open-source software library that was developed by a worldwide community whose purpose is to present a common robotics research and development platform [11]. Before the invention of ROS, each robot necessitated a custom control system to be written, tested, and produced before any study can be initiated. ROS offers universal algorithms which may be executed on any robot setup irrespective of kinematic variations, as long as there is a ROS compliant driver developed for that robot. This significantly minimizes the initial setup for the creation of a new robot and also harnesses research through a common platform for sharing algorithms throughout the communityregardless of hardware access for each user. Among the numerous algorithms available within ROS, we utilized the Simultaneous Localization And Mapping [6], Navigation, Visualization [7] libraries We had also been attracted to high-speed pathfinding methods, hierarchical A*, including the one outlined by Botea et al [2]. SLAM refers to a class of algorithms for integrating different data from sensor to generate maps of an environment and at the same time find the robot location in the map. To know where the robot is, a robot will need a correct map; and to create a correct map, a robot needs to know where it is [8]. Choset and Nagatani proposed a solution to the problem in 2001 [3]. Ever since, SLAM's capacity for simultaneous mapping and localization has turned it into a technique of great significance in the area of robotics [4]. An analogous project was released by Willow Garage in 2010 where a mobile robot employed ROS to navigate autonomously over 26 miles of an office setting. [5] This is undoubtedly a gigantic achievement, yet there are certain distinctions between our implementation and the paper which has been cited. Most importantly, the robot employed in their experiments has an omni-directional drive system that is better suited for obstacle avoidance.The robot in the paper had much more advanced sensor suite and on-board processing capacity for localization, planning, and obstacle detection.

IV. System Architecture

The design of our autonomous navigation robot combines hardware and software elements that facilitate autonomous navigation and mapping of the environment. The system can be broken down into the following subsystems: Microcontrollers, Sensors, Power Supply, Motor Control, and Software Framework.Fig 1 shows the basic system architecture of our work.

A. Microcontrollers

1) Raspberry Pi: : The Raspberry Pi is the brain of the system, which executes ROS2 and handles navigation functions. Its capacity to collect sensor data, create maps, and run efficiant navigation algorithms makes it perfect for this project. The Raspberry Pi interacts with different components using ROS 2 nodes and topics.

2) Arduino Nano: The Arduino Nano is tasked with low-level motor control. It takes velocity commands from the Raspberry Pi and adjusts the geared motors in response. The Arduino also interprets encoder feedback to calculate wheel rotations and furnish information that will be used in odometry estimations.

B. Sensors

1) SLAMTEC RPLiDAR A1M8-360: :The LiDAR sensor is the main origin of environmental data,with 360-degree scanning at a fast refresh rate. It is attached to the Raspberry Pi and allows for SLAM by producing point cloud data, which ROS2 utilizes to construct a live map of the environment.

2) Motor Encoders: : The rotary encoders on the SPG30E-200K motors offer accurate wheel rotation feedback, which is essential for computing odometry and facilitating precise route planning and positioning.

C. Power Supply

A 12V Battery powers the robot, with energy management accomplished through targeted allocation to components:

o 5V supply: The Raspberry Pi needs a constant 5V supply, which is attained using through buck-boost converter which step down the 12V from the battery to 5V.

o 12V Supply: The motors and the L298N motor driver receive a direct 12V supply to ensure sufficient power for motion and control.

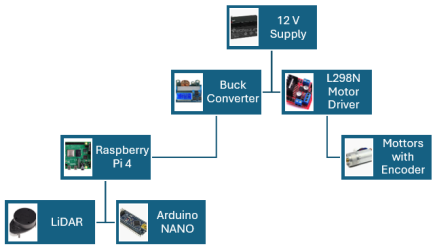

Fig. 1: System Architecture

V. Software Implementation

The ROS2 framework underlies the software implementation. This framework allows for the adaptable and structured integration of hardware and software components.Sensor integration, SLAM (Simultaneous Localization and Mapping), navigation, and communication between the Raspberry Pi and Arduino Nano.

A. Robot Operating System (ROS)

An Operating System (OS) is software that manages hardware and software resources, enabling interaction between the system and users. It handles tasks like running programs, managing files, and coordinating input/output operations. Similarly, the Robot Operating System (ROS) is middleware designed for robotic software development. Though not a standalone OS, it runs on systems like Linux and provides tools and libraries to simplify building complex robotic applications. ROS follows a modular structure where small independent programs, called nodes, handle specific tasks and communicate with each other.

Communication in ROS is handled through topics, which use a publish-subscribe model, allowing nodes to exchange information without direct connections. This design supports scalability and easy system modification. In addition, ROS includes services, which use a request-response pattern suitable for tasks that need confirmation or immediate replies, such as sensor data retrieval or executing commands. For longer operations that require feedback and the option to cancel, ROS uses actions—ideal for tasks like navigation where ongoing updates and control are necessary.

Simulation in **ROS** is vital for safe and efficient robot testing. **Gazebo** simulates real-world environments, while **RViz** visualizes sensor data and robot behavior. Together, they help develop and validate robotic systems without physical risks.

B. SLAM and Localization Algorithms

The Mapping and localization operations in our system are based on SLAM Toolbox and AMCL. The environmental information is collected by using the SLAMTEC RPLiDAR A1M8-360 sensor, which offers scanning with a 360-degree range to assist the SLAM algorithms in map generation and updating. For this project, we particularly picked the Cartographer package in ROS 2 because it achieves a high level of accuracy in indoor mapping and fast low-latency processing to provide consistent navigation within dynamic environments.

1) SLAM Toolbox (Graph-Based SLAM): SLAM Toolbox builds a 2D occupancy grid map, which it uses to represent a graph containing robot locations .

Loop Closure: SLAM Toolbox checks permanently for loop closure, bringing the robot back onto previously explored points to account for drift.

Graph Optimization: SLAM's error minimization attempts to make transformations observed in measurements match the predicted transformations.

2) AMCL (Adaptive Monte Carlo Localization): SLAM Toolbox creates a 2D occupancy grid map with a graph where robot positions are nodes and movements are edges or transformations. AMCL employs a particle filter where particles are potential robot poses and are weighted according to how likely they are to match sensor data.

Particle Weighting:Particles are assigned weights based on the likelihood of matching the observed data:

Pose Estimation: The robot's position is determined by computing the weighted average of particles:

3) Mapping and Path Planning: Mapping and path planning allow the robot to form a map and drive to a target in a safe manner.

Mapping with SLAM Toolbox: SLAM Toolbox generates a 2D occupancy grid representing obstacles and free space. The probability of each cell being occupied is determined using the formula:

Path Planning using ROS Nav2: ROS Nav2 provides global and local planning. The global planner (e.g., A*) computes the initial path, and the local planner adjusts it in real-time

C. Navigation 2

Nav 2 is a central part of the ROS 2 architecture intended to resolve the challenges associated with autonomous navigation in dynamic and unfamiliar environments. As a central part of ROS 2, Nav 2 presents a collection of packages and libraries that provide robots with the capability to move, perceive, and react to their environments in real time. It provides a strong tool set and algorithms for path planning, obstacle avoidance, and sensor processing, which are needed for the creation of Dependable self-driving navigation systems.

The navigation server is the core of **Nav2**, enabling autonomous robot movement through real environments. It manages path planning, motion control, and execution, ensuring smooth navigation even in uncertain conditions. Key components like planners, controllers, recovery behaviors, and smoothers work together to improve accuracy, efficiency, and obstacle avoidance, making **Nav2** a robust framework for real-world navigation.

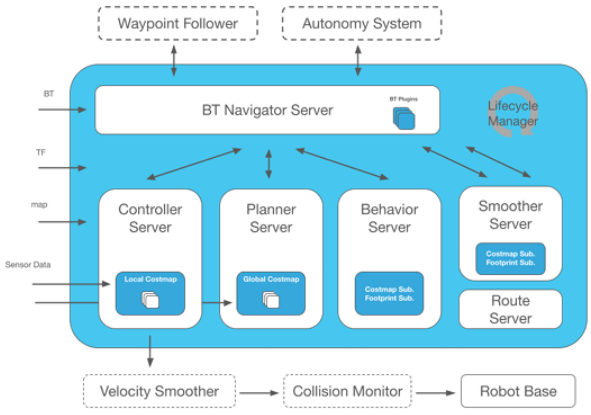

Fig. 2: Navigation Server [18]

The planner server computes optimal paths from the robot's current position to its goal, considering the environment. Depending on configuration, paths may prioritize shortest distance, full coverage, or predefined routes, aiming for safe and efficient navigation.

The controller server ensures the robot accurately follows the planned path by translating it into real-time control commands. It continuously adjusts the robot's motion in response to dynamic environments, ensuring smooth and reliable execution.

VI. Hardware Integration

Hardware integration is essential for ensuring precision and reliability in navigation and control operations. The LiDAR sensor is strategically positioned to provide a full 360-degree view of the surroundings, enabling high-resolution scanning and real-time obstacle detection. It connects to the central processing unit via USB, utilizing dedicated drivers and software libraries to process and interpret LiDAR data efficiently.

To control movement, the L298N motor drivers regulate the speed and direction of the DC motors based on commands from the central unit. Motors equipped with encoders provide real-time feedback on rotation and speed, which is crucial for accurate odometry calculations. This feedback allows the system to monitor and adjust the robot's position and movement with precision.

A high-capacity battery pack powers all components, ensuring stable operation. The battery is connected to a power distribution board, which regulates and distributes power efficiently. The integration process begins with careful planning of wiring and connections to maintain signal integrity and minimize interference. Each component undergoes thorough testing and calibration to ensure optimal performance and seamless operation.

VII. Results and Discussion

Developing and refining a self-navigating robot was a systematic process that combined simulation, hardware integration, and real-world mapping to achieve dependable autonomous navigation. The simulation stage provided a safe platform to test algorithms for localization, mapping, and path planning, helping to identify and correct limitations early. Hardware implementation, involving components such as the Raspberry Pi 4, Arduino Nano, LiDAR A1M8, and motor encoders, ensured accurate sensing, control, and communication between modules. Real-world mapping and navigation trials confirmed the robot's ability to generate reliable environmental maps, localize itself effectively, and adjust paths dynamically when obstacles appeared. Overall, the results validated that the system could navigate smoothly in both static and dynamic conditions, demonstrating robustness, accuracy, and adaptability for indoor applications.

A. Simulation of Autonomous Navigation Platform

Before deploying the robot, we conducted a simulation to ensure the robot could navigate proficiently. The virtual

environment served as a primary testing ground and accurately reproduced operating conditions. Using Gazebo's building editor, we meticulously constructed an environment containing significant features, including walls, corridors, and obstacles, so the robot could navigate real-world challenges.

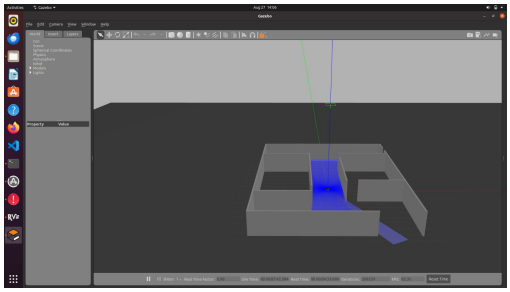

Fig. 3: Gazebo Simulation of the simulated world

The Gazebo simulator was merged with RViz,within the ROS2 ecosystem. We conducted a full evaluation by physically driving the robot in the simulation to validate its strategies for obstacle detection, localization, and navigation. Some primary measurements—such as how accurately the robot followed the planned route, whether it collided with obstacles, and how quickly it achieved the designated goals—were extensively observed and computed.Fig 3 and Fig 4 is obtained after simulation in Gazebo and corresponding visualization in Rviz2

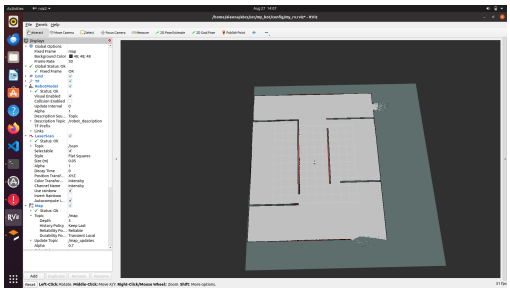

Fig. 4: Robot Visualization of simulated word using RVIZ

In the ROS2 environment, the robot successfully navigated to various targets, dynamically avoiding obstacles and adjusting its path. This tested the accuracy of the map and the efficiency of NAV2 in planning and collision avoidance. Simulation helped identify issues early, ensuring a reliable and low-risk transition to real-world deployment.

B. Hardware Integration

The integration of the motor drivers, motors, and encoders demonstrated a highly reliable mobility system for the robot. The L298N motor drivers successfully regulated the 12V supply to the SPG30E-200K motors, ensuring smooth operation and consistent torque delivery. Encoder feedback from the motors was processed in real-time, which significantly improved odometry estimation and reduced positional errors during long runs. This accurate motion control proved essential for precise trajectory following, enabling the robot to execute planned paths effectively and avoid deviations, thereby validating the robustness of the motor subsystem.

The LiDAR A1M8 sensor produced excellent results in both environmental scanning and obstacle detection. It provided a continuous 360° view of the surroundings with reliable accuracy, even in cluttered indoor spaces. This sensor data was effectively integrated into the SLAM algorithms running on the Raspberry Pi 4, which resulted in the generation of clear and detailed occupancy grid maps. These maps enabled efficient localization and route planning, ensuring that the robot could navigate dynamically changing environments while maintaining a stable and consistent position estimate. The successful synchronization of LiDAR data with odometry confirmed the effectiveness of the chosen localization and mapping strategy.

The overall system integration showcased the capability of the Raspberry Pi 4 and Arduino Nano to manage high-level processing and low-level control tasks, respectively. The Raspberry Pi handled computation-intensive modules of ROS2 such as SLAM, AMCL, and Nav2, while the Arduino ensured precise motor actuation through PWM signals. Power was reliably delivered by the lead-acid battery, which supported long-duration experiments without voltage fluctuations or performance degradation. The combined performance of these components confirmed that the hardware design was both efficient and scalable, ultimately enabling the robot to achieve robust autonomous navigation in complex indoor environments.

C. Mapping of the Real-World

The LiDAR A1M8 sensor used 2D laser scanning to achieve a detailed representation of the environment around it.Fig 5 depicts the laser scan obtained.

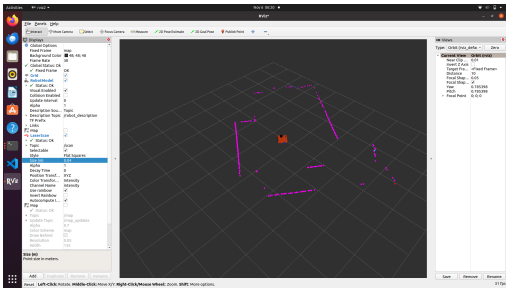

Fig. 5: 2D Laser Scan of real-world

Mapping played a crucial role in enabling autonomous navigation. As the robot moved through its environment, the LiDAR A1M8 continuously scanned its surroundings, capturing real-time spatial data on walls, structures, and obstacles. This data was processed using ROS2 mapping packages powered by SLAM algorithms to generate a high-resolution 2D map.

The resulting map accurately outlined open areas for safe movement and marked obstacles for collision avoidance. It served as a reference model for the navigation system, supporting precise localization and enabling the robot to choose

optimal routes and dynamically reroute when obstacles were detected.Fig 6 shows the map generated of a room.

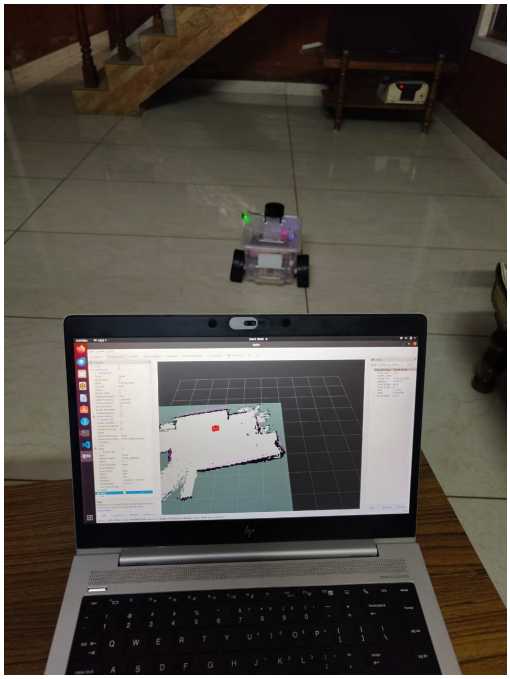

Fig. 6: Mapping

To validate map accuracy, we ran tests in known environments. The LiDAR-based mapping closely matched the actual layout, showing high precision and reliability. The system consistently detected obstacles and chose smooth paths, even in crowded areas.

This accurate mapping was crucial for autonomous operation. With ROS2 and LiDAR working together, the robot could localize itself, plan paths efficiently, and adapt in real-time when faced with unexpected obstacles, ensuring stable and responsive navigation.Fig 7 also depicts the map generated using our Robot.

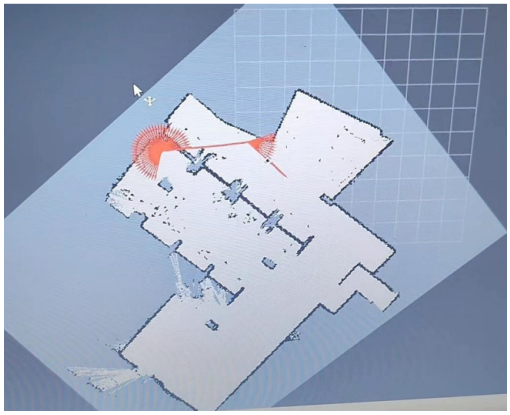

Fig. 7: 2D map of an office environment created by our robot

D. Navigation and Path Planning

The robot successfully navigated from start to goal positions in various indoor environments, including both open and obstacle-rich settings. Across multiple test runs, the system demonstrated reliable localization, accurate path tracking, and effective real-time obstacle avoidance.

Navigation paths visualized in RViz confirmed close adherence to the planned trajectories, with only minor deviations observed during dynamic re-routing. The integration of global and local costmaps allowed for smooth transitions and responsive adjustments in the presence of both static and moving obstacles.Fig 8 shows the image of obtained costmap. These results affirm the robustness and adaptability of the ROS2 Navigation Stack in enabling fully autonomous indoor navigation.

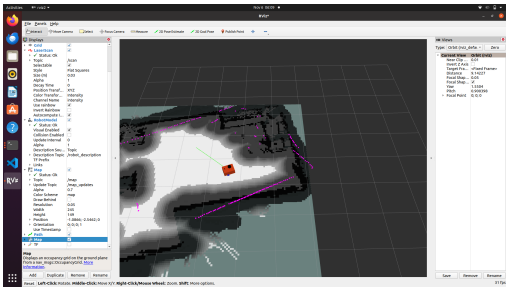

Fig. 8: Costmap

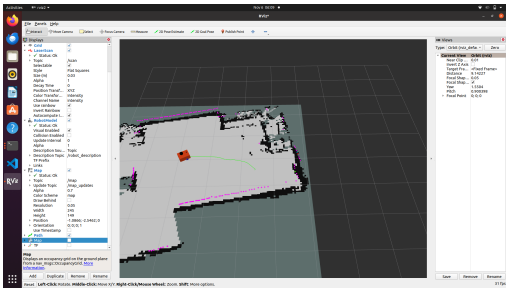

Fig. 9: Path planning

The green path line in Figure 9 visually indicates the robot's computed path, depicting the successful accomplishment of the navigation task. The robot consistently executed this planned route, exhibiting seamless cornering and rapid reaction to real-time barriers. This was achieved through the unobtrusive fusion of LiDAR data for the detection of obstacles, real-time costmap revision, and adaptive path planning algorithms, providing seamless and uninterrupted navigation. Path planning experiments proved to be highly accurate, with the robot always arriving at its destination targets without significant delay or detour. Its responsiveness to adapt and dynamically replot upon encountering obstacles showcased the capability of the navigation system. Such findings confirmed the reliability of the system for use in real-world scenarios, especially in settings such as warehouses and other dynamic environments where adaptive and efficient path planning is critical.

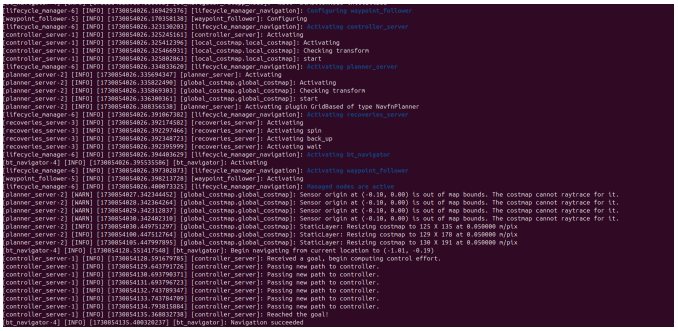

Fig. 10: Log output of the autonomous navigation system, showing node activations, costmap adjustments, path planning, and successful goal-reaching by the robot.

Fig 10 is the result obtained when the robot reaches the destination,The smooth blending of simulation, hardware, and software elements yielded a robust autonomous navigation system. Its excellent performance in both simulation and real-world settings proved the system's feasibility for real-world applications. Possible future enhancements would include optimizing path-planning algorithms for improved dynamic obstacle handling or expanding the system to accommodate multi-floor navigation, potentially using behavior trees for enhanced decision-making.

Conclusion and Future works

The successful implementation, design, and evaluation of the autonomous navigation robot in ROS2 proved the system to be reliably functional in dynamic environments. From simulation to hardware integration, real-world mapping, and path planning, the project successfully developed a robust and efficient navigation system. The simulated outcomes validated the precision of the system, while the integration of essential hardware components like the LiDAR A1M8, Raspberry Pi 4, L298N motor driver, and SPG30E-200K motors guaranteed real-time processing and accurate movement.

Visualization of paths in RViz and real-time dynamic adaptation of the robot's path further highlighted the effectiveness of the system in real-world applications. The successful outcomes verified that the autonomous navigation system is highly suitable for real-world deployment, where precision, adaptability, and reliability are key considerations.

The autonomous navigation robot demonstrated effective performance in indoor environments, validating its potential for real-world applications such as warehouse automation, healthcare monitoring, and indoor service robotics.

Future enhancements include integrating additional sensors like depth cameras to improve environmental perception, developing advanced path planning algorithms for complex obstacle management, and testing the system in more challenging environments. Extension to support lift APIs would enable seamless multi-floor navigation, making the system ideal for deployment in multi-story buildings. These improvements aim to enhance the robot's versatility, efficiency, and real-world applicability.

Acknowledgement

This project was supported by Trizlabz Pvt. Ltd. and the Muthoot Institute of Technology and Science. We would also like to express our gratitude towards our guide, Dr. Shajimon K. John, for his insightful inputs and knowledge. Furthermore, we appreciate the OSAM funding arranged by our college, which helped in the successful completion of this project.

References

[1] Berenson, D.; Abbeel, P.; and Goldberg, K. "A Robot Path Planning Framework that Learns from Experience." IEEE International Conference on Robotics and Automation (ICRA), May, 2012.

[2] Botea, A.; Muller, M.; and Schaeffer, J. "Near Optimal Hierarchical Path-Finding." Journal of Game Development 2004.

[3] Choset, H.; Nagatani, K. (2001). "Topological simultaneous localization and mapping (SLAM): Toward exact localization without explicit localization." Robotics and Automation, IEEE Transactions on, 17(2), 125-137.

[4] Dissanayake, M.W.M.G.; Newman, P.; Clark, S.; Durrant-Whyte, H. F.; Csorba, M. (2001). "A solution to the simultaneous localization and map building (SLAM) problem." Robotics and Automation, IEEE Transactions on, 17(3), 229-241.

[5] Marder-Eppstein, E.; Berger, E.; Foote, T.; Gerkey, B.; and Konolige, K. "The Office Marathon: Robust navigation in an indoor office environment," Robotics and Automation (ICRA), 2010 IEEE International Conference on , vol., no., pp.300-307, 3-7 May 2010.

[6] Durrant-Whyte, H.; Bailey, T. (2006). "Simultaneous Localization and Mapping (SLAM): Part I The Essential Algorithms." Robotics and Automation Magazine 13 (2): 99–110.

[7] Hershberger, D.; Faust, J. "rviz Package Summary" [Online]. Available: http://www.ros.org/wiki/rviz

[8] Leonard J.J Durrant-Whyte, H.F. (1991). "Simultaneous map building and localization for an autonomous mobile robot." Intelligent Robots and Systems '91. 'Intelligence for Mechanical Systems, Proceedings IROS '91. IEEE/RSJ International Workshop on, 1442-1447 vol.3.

[9] "Navigation" [Online]. Available: http://www.ros.org/wiki/navigation

[10] "p2os" [Online]. Available: http://www.ros.org/wiki/p2os [

[11]] "ROS Documentation" [Online]. Available: http://www.ros.org/wiki/

[12] A. Thakur and P. Rajalakshmi, "LiDAR-Based Optimized Normal Distribution Transform Localization on 3-D Map for Autonomous Navigation," in IEEE Open Journal of Instrumentation and Measurement, vol. 3, pp. 1-11, 2024, Art no. 8500211, doi: 10.1109/OJIM.2024.3412219.

[13] J. Zhang and K. Zhao, "3D localization and pose tracking system for an indoor Autonomous Mobile Robot," 2015 IEEE International Conference on Mechatronics and Automation (ICMA), Beijing, China, 2015, pp. 2209-2214, doi: 10.1109/ICMA.2015.7237829.

[14] M. Nieuwenhuisen, J. Stückler and S. Behnke, "Improving indoor navigation of autonomous robots by an explicit representation of doors," 2010 IEEE International Conference on Robotics and Automation, Anchorage, AK, USA, 2010, pp. 4895-4901, doi: 10.1109/ROBOT.2010.5509689.

[15] S. Noh, J. Park and J. Park, "Autonomous Mobile Robot Navigation in Indoor Environments: Mapping, Localization, and Planning," 2020 International Conference on Information and Communication Technology Convergence (ICTC), Jeju, Korea (South), 2020, pp. 908-913, doi: 10.1109/ICTC49870.2020.9289333.

[16] Z. Mo, W. Hao, Z. Wang, Y. Huang and H. Wu, "A Method for Evaluating the SLAM Algorithm Based on the ROS_Gazebo Simulation Platform," 2022 International Conference on Intelligent Manufacturing and Industrial Big Data (ICIMIBD), Changsha, China, 2022, pp. 1-4, doi: 10.1109/ICIMIBD58123.2022.00012.

[17] J. Krushnan, H. Thangaraj, K. Thiruvalluvan, S. Balasubramanian, V. P. Uppalapati and F. Schrödel, "MicroROS Based Controller and RViz Visualization for Robot Manipulation as an Educational Module," 2024 32nd Mediterranean Conference on Control and Automation (MED), Chania - Crete, Greece, 2024, pp. 406-411, doi: 10.1109/MED61351.2024.10566213.

[18] Navigation concepts. *https://navigation.ros.org/concepts/index.html#navigation-servers*. [Online; accessed April 2023].