Autonomous Navigation For TurtleBot3 Robots in Gazebo Simulation Environment

Madhav Theeng Tamang
Intelligent Technologies Research Group, School of Architecture, Computing and Engineering, UEL
London, UK
u1430774@uel.ac.uk

Darshana Maheriya
Intelligent Technologies Research Group, School of Architecture, Computing and Engineering, UEL
London, UK
u2502796@uel.ac.uk

Mhd Saeed Sharif
Intelligent Technologies Research Group, School of Architecture, Computing and Engineering, UEL
London, UK
s.sharif@uel.ac.uk

Thamo Sutharssan
Department of Engineering UEL
London, UK
0000-0003-1836-0517

Abstract **— The fast-paced growth in the field of robotics has driven the creation of autonomous navigation systems that are necessary for robots to work autonomously in diverse environments. This research targets TurtleBot3, an immensely popular robotic platform that is known for its affordability and adaptability. The objective of TurtleBot3 is to improve the capabilities within the Gazebo simulation environment, an open-source robotic simulator that provides a realistic virtual environment for testing and developing navigation algorithms for robotics. The combination of intricate sensor technologies, leading-edge control systems, and innovative artificial intelligence strategies is the core of this research, enabling autonomous navigation in complicated surroundings. The initial focus of the study is the application and utilization of path planning algorithms, such as A* (A Star) and Rapidly exploring Random Tree (RRT) and using Simultaneous Localization and Mapping (SLAM) algorithms. Results from simulations demonstrate that the A* algorithm achieved a path accuracy rate of over 95% in static environments, while the RRT algorithm proved effective in dynamic, multi-dimensional spaces. The aim of this analysis is to enhance autonomous navigation algorithms using the Gazebo simulation environment. Conducting tests in a virtual setting minimizes the risks and costs compared to real-time testing. The outcomes of this research are anticipated to advance the field of robotics and hold significant value for applications in autonomous systems, including engineering, urban mobility, and domestic assistance.**

Keywords — Autonomous navigation, Gazebo simulation, Obstacle avoidance, Path planning algorithm, Robot Operating System (ROS), TurtleBot3

I. INTRODUCTION

The last few decades have noticed a large number of research in the robotics field [1],[2]. The project, “Autonomous Navigation for TurtleBot3 Robot in Gazebo Simulation Environment”, targets to take advantage of these advancements by creating a navigation system that facilitates the robot to move autonomously within dynamic and complex environments. The Gazebo simulation environment is a unique platform used to develop and test navigation algorithms. This virtual environment mirrors the physical world physics and material characteristics, along with different sensors and controllers. It is an ideal testbed for refining the navigation algorithms before deployment in physical robots [3], [4], [5]. The foundation of this research is incorporating multiple intricate sensor technologies, modern control systems, and artificial intelligence procedures that allow the TurtleBot3 to work on path planning, live obstacle detection, and flexible navigation techniques. Path planning algorithms for instance A*(A star) and Rapidly Exploring Random Tree (RRT) are important during navigation, as the algorithms calculate the most secure and efficient path to a designated location [5] , [6], [7].

The project utilizes SLAM algorithms to create a navigation map for the robot to navigate in an unknown environment, while locating itself within that map at the same time. To navigate in a virtual environment, this ability is crucial for TurtleBot3, that provides the fundamental mechanisms which is required for robots to choose the most efficient path independently and improve the capacity of the robot to manage in continuous changing and uncertain environments [4].

II. LITERATURE REVIEW

The analysis of independent navigation for TurtleBot3 in Gazebo simulation environment surrounds a broad area of studies that covers three primary areas of technology: creating and modifying path planning formulas, refining localization, and mapping techniques, and integrating multi-sensor data for improved robotic observation. The study aims to enhance TurtleBot3's autonomous navigation by refining path planning algorithms (such as A* and RRT) and SLAM techniques within the Gazebo simulation environment. This includes integrating advanced sensor data from LiDAR, cameras, and IMUs to improve obstacle avoidance and mapping accuracy.

The foundation of the extensive scope of mobile robotics in domains, for instance domiciliary help, commercial automation, and urban transportation, is independent navigation. Improving potentiality of autonomous navigation, like those developed for the TurtleBot3, prepares the robots to execute complicated work individually and improving robot’s efficiency and applicability [5]. Gazebo is as a powerful three-dimensional dynamic counterfeit that can resemble several robotic models performing in various environments [8]. Gazebo facilitates the simulation of sensor data and enables a user to test robotic models in simulation environment [9]. Gazebo provides the platform to test the functionalities of robotic models in appropriate scenarios that prevents any physical damage to the robot [10].

Implementation of advanced algorithms such as path planning algorithms (e.g., RRT, A*) and SLAM techniques facilitate the robots to figure out and navigate within the space precisely. Studies by [11] focuses on the importance of these algorithms in improving the accuracy and reliability of independent robots in virtual environments which are crucial for successful deployment in the real-world.

This project is important for pointing out the primary complications related to physical world deployment. These complications include adapting to variable and unknown settings, assuring algorithms to operate through the calculated limitations of actual robotic platforms, and reliability of navigation systems during the failure of sensors.

Research Challenges:

- One of the primary challenges is the complicated algorithms required for efficient navigation and how these algorithms will perform in the physical world. SLAM algorithms and path planning are vigorous in a controlled and pre-defined simulation environment, but practical robotic applications often exceed hardware capabilities in their use of computational resources [12].
- Incorporating input data from multiple sensors (such as IMUs, LiDAR, and cameras) required to build precise judgements of the setting. The primary complication is handling the huge data generated by multiple sensors and assuring that the blending process is at proper time and precise, poses massive challenges. The research states the necessity for advancing data fusion strategies to manage complications and provide reliable results for navigating robots [11].
- Discrepancy between virtual environment and the physical world is a major challenge. Gazebo provides the best platform to assess but sometimes environmental circumstances and physical characteristics it mimics can go wrong to replicate those experienced in actual life situations. This disparity can direct navigation systems that work well in virtual environments but wobble with physical implementation, where uncertain changes such as light conditions and impacts of climate are common.
- Another challenge is maintaining the durability and reliability of independent navigation systems in changing and uncertain real-world environments. The research emphasizes the significance of creating navigation systems that can prepare the system for unpredictable environmental changes and can work well even when specific sensors fail, or data is not available.
- The scalability of navigation systems from controlled environments to more complex real-world applications remain a terrific challenge. As the working environment becomes more complicated, maintaining persistent performance and reliability of the navigation algorithms becomes more difficult.

The literature review of this research explains a solid analysis of diverse technological improvements, specifically in path planning, SLAM algorithms, and fusion of sensors. This study also determines continuous challenges related to transferring these simulated processes to execution in the physical world, considering computational restrictions, and assuring durability and reliability through different scenarios. Future work can aim to improve efficiency for less resource platforms, working on algorithms to better imitate existing situations, and adaptability of independent navigation systems in uncertain scenarios.

III. METHODOLOGY

Virtual environments provide a vital platform for experimenting with robotics research, especially for independent navigation. Gazebo simulation environment utilizes the potential of ROS (Robot Operating System) and helps in the formation, experimenting, and authentication of robotic models before deploying those to the physical world.

The first step towards the research starts with the configuration of the simulation environment and the TurtleBot3 model (e.g., Burger, Waffle, and Waffle Pi), implementation of multiple sensors and the employing navigation, path planning and SLAM techniques. These features are managed via ROS, using its framework and toolset such as the navigation stack and Rviz for visualizing and a few other required development tools.

A. Environmental Setup and Robot Configuration

Assembling a simulation environment includes downloading and configuring ROS and the Gazebo simulation environment. ROS is a base that provides the required toolkits and packages for developing robotic software, and Gazebo provides a physically and visibly original setting [12], [14], [15]. Simulation environment setup and configuration involves the following steps:

- ROS Installation: ROS Noetic is the stable version of Ubuntu 20.04. It can be installed from Ubuntu's package management tools. For this research, a full desktop version has been chosen, which requires developing toolsets.
- Integration of Gazebo: Gazebo comes with ROS Noetic. Plugins and tools can be used to connect ROS with the Gazebo simulation environment.
- Setup of Workspace: A specific ROS (Catkin) workspace is needed that can manage packages and custom developments.
- Model Selection: The TurtleBot3 has a few variants (e.g., Burger, Waffle, and Waffle Pi), each model is equipped with various features and abilities. The performance and complexity of the tasks also depend on the model chosen. This model can be set up by configuring it in the launch files.
- URDF Model Configuration: The physical and optical characteristics including measurements, colors and hierarchy of mobile joint is determined by Unified Robot Description Format (URDF). Sensors such as LiDAR, camera, IMUs are also determined by URDF combined within the type of robot.
- Sensors simulation: For Autonomous navigation and SLAM, the required data can be obtained from

simulated sensors. Efficiency of constructed algorithms relies upon the precision of data of simulated sensors.

- Plugin Usage: Multiple ROS-Gazebo plugins can facilitate lifelike physics simulation and sensor data acquisition. Wheel motions, sensor data generation and intercommunication are managed by these plugins.
- Simulation Parameters: The simulation realism is balanced with computational demands by tuning parameters like the simulation time, physical characteristics (e.g., mass, friction), and update rates to a fine limit.

B. Sensor Simulation and Data Integration

The sensor simulation and integration of their data are crucial to obtain reliable and efficient navigation capabilities.

LiDAR Simulation: The use of LiDAR sensors meets the need for machine vision. This is required as the AGV (automated guided vehicle) robot must be able to identify and investigate steady and dynamic substances [16], [17]. LiDAR sensor is useful for detecting and avoiding obstacles and helps calculate the terrain. Gazebo has a built-in plugin that resembles precision and ranges of physical LiDAR sensor. The system processes data from the simulated LiDAR to generate a spot overcast or depth plan that shows the environment [13].

Camera Simulation: For visual navigation and identifying objects, cameras are used. These sensors are used to grab images or video streams that can mirror the optical elements of the Gazebo simulation environment. Data from this sensor is vital for SLAM and algorithms that rely on indications for navigation.

IMU Simulation: The IMU gives data about acceleration and location of the robot, which is used when GPS data is not accessible. Gazebo's IMU plugin mimics sensor noise to approximate the real inaccuracies of real-world sensors.

Data Integration: The generated data of sensors is integrated into ROS for processing and decision making, which is a crucial part of creating effective navigation algorithms. This integrated data reduces the unpredictability that could be possible due to replying on a single sensor data. Data fetched from sensors is constantly returned into simulation to modify the robot's movement according to the predefined environment.

C. Navigation and Path Planning

Path planning algorithms design the foundation of the autonomous navigation system, which allows the robot to navigate in dynamic surroundings safely. Path planning involves measuring the ideal route from current position to targeted location.

- Global Path Planning: This algorithm designs the path before the robot starts to move. Dijkstra's algorithm and A* algorithms are efficient to find the optimal path in a predefined environment.
- Local Path Planning: When Global Path planning is determined, local path planning will start handling the real-time adjustments, this is required to reduce collision of moving obstacles which were not considered initially. Dynamic Window Approach (DWA) evaluates the robot's velocity and location and adjusts the path accordingly to avoid obstacles in a dynamic environment.
- Simultaneous Localization and Mapping: SLAM techniques authorize the robot to build and modify the path in autonomous environment while tracing its location. Gmapping and Cartographer techniques are utilized depending on the requirements. Gmapping is used due to its robustness in managing noisy sensor data and it is efficient in creating accurate 2D maps. Cartographer is used for complicated settings. Google's cartographer provides real-time 3d mapping.
- Integration with ROS: Gmapping and cartographer are part of ROS packages, which makes it easier to integrate into the current ROS. SLAM techniques play a key role in performance improvement. Data fetched from Gmapping and cartographer is required to be adapted according to characteristics of the environment and hardware capabilities of the simulation system.

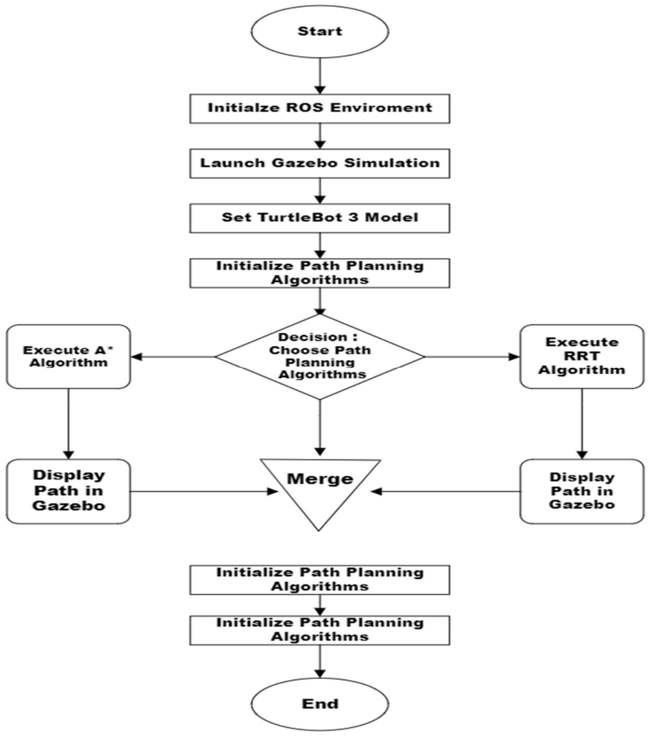

Fig. 1. Flow chart of the proposed model

D. Mathematics behind the algorithms

Some of the approved path planning algorithms in robotics are A* (A star) and RRT (Rapidly exploring Random Tree). The concepts of math behind each algorithm are explained here, demonstrating its implementation and application of TurtleBot3.

A* Algorithm: This algorithm identifies the minimal route between current location to destined location. This algorithm uses path-cost function and heuristic. Heuristic measures the path from start node to target node.

- Cost Function (g(n)): Describe the route cost starting with initial node to n node.
- Heuristic (h(n)): It evaluates minimal cost of beginning to target node.
- F-Score (f(n)): To give preference to a node in the queue, this function is used in (1). It is the summation of g(n) and h(n):

$$f(n)=g(n)+h(n) \qquad (1)$$

Algorithm Steps:

- Initiate the priority line commencing with the opening node.
- Loop till the priority list is vacant:
- Pick up n node with the smallest f(n) from the open list.
- If node n has arrived at the goal, build the route again.
- For every neighbour m of n, measure g(m), the possible smallest cost to node m.
- When m is not in list or the possible g(m) is smaller than the observed g(m), rectify (m), set the parent of m to n, and calculate f(m) again, then add m to the list.

RRT Algorithm:

This algorithm works by gradually developing a tree from the initial node to the target location via random sampling. This algorithm is advantageous in a multidimensional setting. The RRT includes the edges and nodes, commencing with the first node and then generates random nodes in the given surroundings.

Algorithm Steps:

- Start the tree with the first node.
- Loop until the target has arrived or till k iterations:
  - Create a fresh point q_{rand} randomly.
  - Look for the closest node q_{near} to q_{rand} in the tree.
  - Make a different node q_{new} by spanning from q_{near} to q_{rand} by specified length.
  - Build q_{new} to the tree when there is no destruction to the route from q_{near} to q_{new}.
  - If accessible, attempt to link q_{new} straight to the target.

Mathematics involved:

- Euclidean distance is used to measure the q_near distance as shown in (2):

$$Distance = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \qquad (2)$$

- To build q_new steering Function is used as shown in (3):

$$qnew=qnear+step\_size\cdot qrand-qnear/\|qrand-qnear\| \qquad (3)$$

- The above computations make sure that the created route moves in the direction of undiscovered spaces and can be easily navigated through Gazebo simulated environments.

IV. EXPERIMENTAL RESULTS

The findings of the experiments in this research were the efficiency and reliability of the system and showed the scope of improvement by executing autonomous navigation algorithms. Here are some key insights from multiple scenarios, focusing on the performance of navigation techniques, precision of sensors, and overall stability of the system.

A. Simulation Setup

Experimentations were executed in a Gazebo simulation environment with Burger model of TurtleBot3 robot in house environment to determine the performance. This environment has multiple stagnant and dynamic objects. The TurtleBot3 robot was assembled with LiDAR and cameras.

- Step 1: The first step is to open a terminal and enter 'roscore' command. This command will start the main control and management process of ROS. This includes multiple important features that are necessary for ROS nodes to communicate.
- Step 2: Keep the first terminal open and then open one more terminal and enter 'roslaunch turtlebot3_gazebo turtlebot3_house.launch' command to launch the TurtleBot3 robot. Gazebo simulator will be initiated by this command and load the inbuilt housing environment for TurtleBot3. The TurtleBot3 robot will then be placed in the simulation environment presented in Fig. 2.

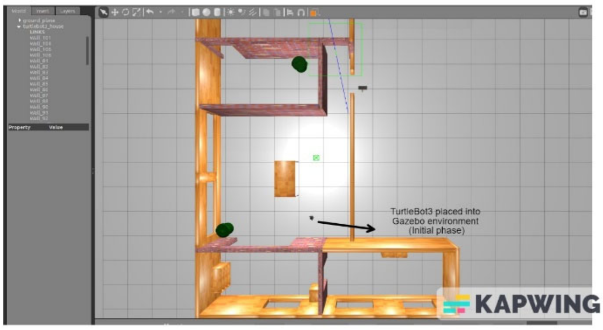

Fig. 2. Gazebo of turtlebot3 in initial phase

Step 3: Open a new terminal and enter the 'rosrun' command with path to the python file to run the python script. This command tells the ROS to execute the Python file. Fig. 3 and Fig. 4 shows that the robot has started moving towards the destination.

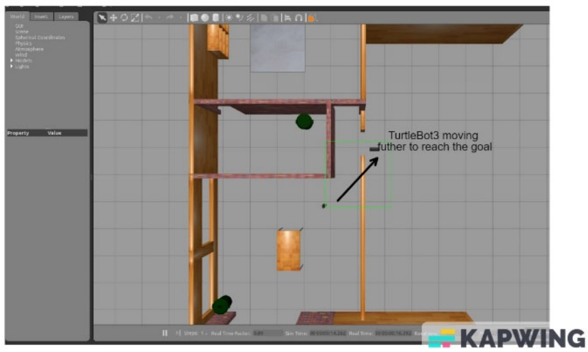

Fig. 3. The Turtlebot3 moving further

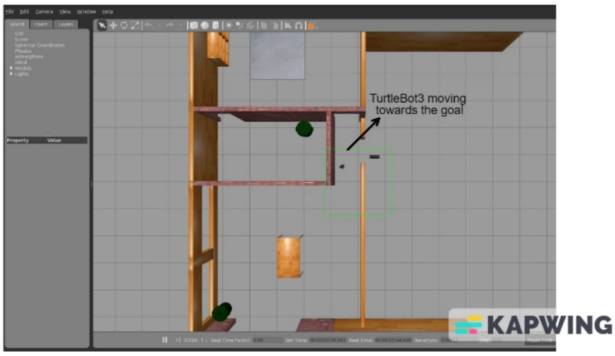

Fig. 4. The Turtlebot3 navigates to the Target spot

B. Path Planning and Obstacle Avoidance

Results achieved from the algorithms indicate higher reliability. The robot effortlessly drove independently to the goal nodes with mean path fluctuation below 5% from the optimum route, that demonstrated the efficacy of the A* algorithm in steady environments. TurtleBot3 modified the route in moving obstacles environment, suggested how well DWA algorithm worked.

C. SLAM Accuracy and Map Quality

Implementing SLAM techniques resulted in a high accuracy in mapping and localization. The TurtleBot3 effectively maintained precise localization throughout the navigation tasks.

- Map Accuracy: The maps generated by mapping algorithms showed an average accuracy rate of 95% in a known simulation environment. Slight disparity was observed near low-contrast entities, occasionally leading to minor curves in the map.

- Localization Precision: The TurtleBot3 robot kept its localization failure under 0.5 meters during testing, which is tolerable range for nearly all jobs for indoor navigation. For replicating environment like TurtleBot3 in Gazebo which covers robotics and physics, to grasp how the replication of environment connects to real-world operation, the idea of 'Sim Time' vs 'Real Time' is key.

The graph in Fig. 5. shows that the x-axis represents simulation time in seconds, ranging from 120 to 50 seconds and the y-axis 'variable value' represents the number of iterations, from 100 to 200. The graph shows that the algorithm is performing nearly consistently with a steady increase in iterations per second of simulation time. The slight distortion to the line is due to minor variations in the simulation environment.

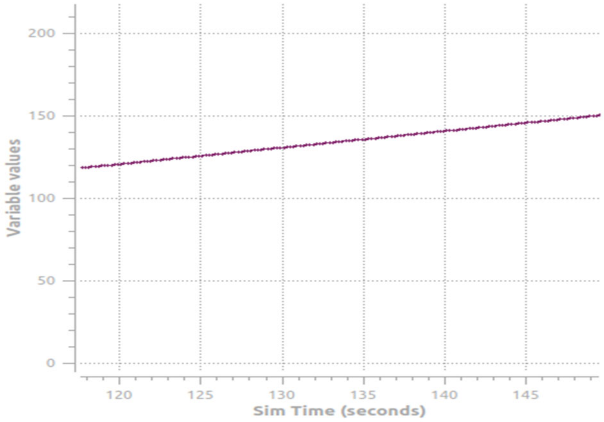

Fig. 5. SIM Time vs. Real-time of developed Turtlebot3

In simulations involving any robotic system in virtual environments, shown in Fig. 5, to observe the performance of algorithms, interrelation of Simulation Time (SIM Time) and the number of iterations is essential. The graph in Fig. 6 indicate that simulation is working close to real-time. Both the graphs indicate that simulation is stable and relatively constant throughout the simulation.

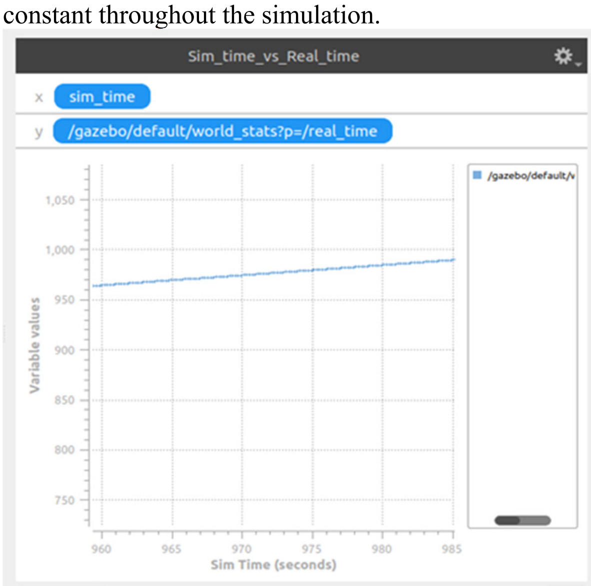

Fig. 6. SIM Time vs Iterations of A* and RRT Turtlebot3 (Gazebo)

V. Analysis and Discussion

A. Key Findings

The integration of advanced path planning and SLAM techniques greatly improved TurtleBot3's autonomous navigation capabilities. The Gazebo simulation environment proved to be an ideal platform for testing and refining these navigation algorithms prior to real-world deployment, enabling the robot to navigate accurately and reliably in dynamic environments.

- Efficiency of Path Planning Algorithms: The A* and DWA algorithms are efficient in predefined environments. However, in unknown environments, the system struggled sometimes, which suggests a need for refinement in algorithms.

- Obstacle Avoidance and SLAM: These algorithms +provide accurate mapping and localization, improving the effectiveness of the navigation system. Gmapping gave high-fidelity maps, required for navigation and path planning.
- Computational Demands: The performance of the system under different computational loads displayed that the TurtleBot3 robot is optimized for live processing on average hardware, but limited resources can affect the performance in complicated surroundings.

B. Discussion

The result showed that the combination of A* and DWA algorithms provided great results for path planning, while SLAM algorithms ensured accurate localization and mapping. Continued research and improvement are essential to conquer current limitations and unlock new possibilities in robotics.

- Adaptability of Algorithms: The moving obstacle avoidance analysis underlined the urgent requirements of flexible and location-aware navigation algorithms. Future work could include employing machine learning specifically reinforcement learning and deep learning algorithms to predict and adapt to dynamic movements.
- Testing in real-world: To validate the system's performance, algorithms must be demonstrated in real-world surroundings. Testing in the physical world will help in figuring out any disparity between the physical world and the virtual world. This will help in providing insights for future enhancements.
- Scalability and Robustness: To find out the system's scalability and durability, experiment the algorithms in an unstructured and complicated environment. Carrying out trials of the system in different complicated situations —like outdoors or areas with lots of people—would play a crucial role in evaluating whether the system is ready to operate in the physical world.

VI. CONCLUSION AND FUTURE WORK

The methods used to simulate TurtleBot 3 in Gazebo showed that incorporating sensor data was efficient, and path planning and obstacle avoidance techniques were reliable. ROS provides adaptable and strong testbed for self-driving navigation algorithms to develop and test in a restrained yet lifelike environment. The following areas for future work to refine self-driving navigation systems:

- Machine Learning Techniques: Utilizing machine learning algorithms (such as deep learning and reinforcement learning) can enhance flexibility and improve decision-making. The system can be improved and become more adaptable to new environments by using these algorithms.
- Energy Efficiency: Optimization of algorithms for energy efficiency is important to extend the operating time of the robots, specifically when using in real-world operations where available energy is limited and expensive. This involves rectifying path planning and SLAM techniques to lessen computational overhead.
- Real-world Application: Testing in the real-world is crucial to know how robots work outside of controlled environments. Consider that future work could be tested outdoors with a more complex and unpredictable environment.
- Multi-robot System: Delve into how multiple robots can work together in Gazebo that might create fresh uses and improve the ways of developing things. Exploring appropriate SLAM and shared navigation techniques would play a significant part in this area.
- Advanced Sensor Integration: Advanced sensor integration techniques will be explored to enhance the robot's perception and decision-making capabilities. Using new sensor technologies and data fusion techniques could help improve this capability.

REFERENCES

[1] T. A. Salih, M. T. Ghazal, and Z. G. Mohammed, "Development of a dynamic intelligent recognition system for a real-time tracking robot," IAES International Journal of Robotics and Automation (IJRA), vol. 10, no. 3, p. 161, Sep. 2021, doi: 10.11591/ijra.v10i3.pp161-169.

[2] H. M. Qassim and W. Z. W. Hasan, "A review on upper limb rehabilitation robots," Applied Sciences (Switzerland), vol. 10, no. 19, pp. 1–18, Oct. 2020, doi: 10.3390/app10196976.

[3] A. W. Winkler, C. D. Bellicoso, M. Hutter and J. Buchli, "Gait and trajectory optimization for legged systems through phase-based end-effector parameterization", IEEE Robot. Autom. Lett., vol. 3, no. 3, pp. 1560-1567, Jul. 2018.

[4] C. D. Bellicoso, F. Jenelten, C. Gehring and M. Hutter, "Dynamic locomotion through online nonlinear motion optimization for quadrupedal robots", IEEE Robot. Autom. Lett., vol. 3, no. 3, pp. 2261-2268, Jul. 2018.

[5] J. Collins, S. Chand, A. Vanderkop and D. Howard, "A Review of Physics Simulators for Robotic Applications," in IEEE Access, vol. 9, pp. 51416-51431, 2021.

[6] J. Qi, H. Yang and H. Sun, "MOD-RRT*: A Sampling-Based Algorithm for Robot Path Planning in Dynamic Environment," in IEEE Transactions on Industrial Electronics, vol. 68, no. 8, pp. 7244-7251, Aug. 2021.

[7] M. A. R. Pohan, B. R. Trilaksono, S. P. Santosa and A. S. Rohman, "Path Planning Using Combined Informed Rapidly- Exploring Random Tree Star and Particle Swarm Optimization Algorithms," in IEEE Access, vol. 12, pp. 56582-56608, 2024.

[8] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: part I," in IEEE Robotics & Automation Magazine, vol. 13, no. 2, pp. 99-110, June 2006.

[9] R. Tornese et al., "ROS-based simulation environment for obstacle avoidance in autonomous navigation," in 2022 IEEE Conference on Control Technology and Applications, CCTA 2022, IEEE, Aug. 2022, pp. 17–22, doi: 10.1109/CCTA49430.2022.9965786.

[10] A. Dobrokvashina, S. Sulaiman, T. Gamberov, K. H. Hsia, and E. Magid, "New Features Implementation for Servosila Engineer Model in Gazebo Simulator for ROS Noetic," Proceedings of International Conference on Artificial Life and Robotics, vol. 28, pp. 154–157, Feb. 2023, doi: 10.5954/icarob.2023.os6-3.

[11] A. O. Prasad et al., "Design and development of software stack of an autonomous vehicle using robot operating system," Robotics and Autonomous Systems, vol. 161, p. 104340, Mar. 2023, doi: 10.1016/j.robot.2022.104340.

[12] Karur, K., Sharma, N., Dharmatti, C. and Siegel, J.E., "A survey of path planning algorithms for mobile robots". Vehicles, 3(3), pp.448-468, 2021.

[13] E. Marder-Eppstein, E. Berger, T. Foote, B. Gerkey and K. Konolige, "The Office Marathon: Robust navigation in an indoor office environment," 2010 IEEE International Conference on Robotics and Automation, Anchorage, AK, USA, pp. 300-307, 2010.

[14] Fairchild, C. and Harman, T.L., ROS Robotics By Example: Learning to control wheeled, limbed, and flying robots using ROS Kinetic Kame. Packt Publishing Ltd., 2017.

[15] Nguyen, L.A., Harman, T.L. and Fairchild, C., September. Swarmathon: a swarm robotics experiment for future space exploration. In 2019 IEEE International Symposium on Measurement and Control in Robotics (ISMCR), pp. B1-3, 2019.

[16]. V. Arulkumar, M. A. Lakshmi and B. H. Rao, "Super Resolution and Demosaicing based self learning Adaptive Dictionary Image Denoising framework," 2021 5th International Conference on Intelligent Computing and Control Systems (ICICCS), 2021, pp. 1891-1897, doi: 10.1109/ICICCS51141.2021.9432182.

[17]. S. Anantha Babu, R. Joshua Samuel Raj, Arul Xavier V.M and N. Muthukumaran, "DCT based Enhanced Tchebichef Moment using Huffman Encoding Algorithm (ETMH)," 2021 Third International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV), Tirunelveli, India, March 2021, pp. 522-527.