Implementation of path planning with obstacle avoidence using SLAM in Services Robot

Ryan Satria Wijaya[1], Dikki Wahyudi Tarigan[2], Senanjung Prayoga[3], Rifqi Amalya Fatekha[4]

{ryan@polibatam.ac.id}

Barelang Robotic (BRAIL),Department of Electrical Engineering,Politeknik Negeri Batam ,Batam,Indonesia[1234]

Abstract. Obstacle avoidance is an important aspect in service robot navigation. This research proposes an approach that integrates Lidar sensors with Simultaneous Localization and Mapping (SLAM) methods to improve the robot's ability to identify, understand, and avoid obstacles in dynamic environments. This data is then processed using SLAM algorithms to create a real-time map of the environment while estimating the robot's position within it. In this research, lidar is used to detect obstacles to be encountered by the robot. Dynamic environment mapping allows the robot to detect changes and adjust its path plan on the fly. An obstacle avoidance algorithm is designed to interact with SLAM data, so that the robot can adaptively change its movement path to avoid newly appearing or moving obstacles In robotic path planning, Dijkstra's algorithm can be applied to generate the shortest, most efficient route from the robot's current location to a target location. The algorithm operates on a predefined map or a dynamically updated map (as in SLAM), considering obstacles and the cost associated with traversing different parts of the environment. Dijkstra algorithm is chosen to determine the path to be traveled by the robot.Djikstra algorithm is chosen to determine the path to be traveled by the robot. The djikstra algorithm takes the robot through the obstacle barrier well from the starting position to the destination point with excellent.Combining SLAM with Lidar obstacle avoidance improves the robot's robustness in complex and rapidly changing environmental situations. Simulation experiments and field testing show that this approach is effective in improving the robot's performance in dealing with obstacles and optimizing its autonomous navigation. By utilizing Lidar and SLAM technologies, this research contributes to the development of reliable robot navigation systems in various application contexts.

Keywords: Lidar, Simultaneous Localization and Mapping (SLAM), Algorithm, navigation, obstacles, real-time.

1 Introduction

Obstacle avoidance technology is a method recently adopted by some robot makers to achieve the goal of intelligent robots. One common method for tackling this issue is SLAM, which enables robots to create a map of their surroundings while concurrently monitoring their location. Obstacle avoidance technology consists of two components: the first is a global obstacle avoidance approach that relies on pre-existing environmental data, and the second is a local obstacle avoidance approach based on sensor inputs[1]. Implementing SLAM with LiDAR sensors provides high accuracy in environmental mapping, which is crucial for path planning and robot navigation. LiDAR sensors can deliver high-resolution distance data, enabling accurate obstacle detection and detailed environmental information. With the map generated by SLAM, robots can plan an optimal path to reach specified goals while avoiding surrounding obstacles[2].

Path planning is a component that ensures the robot can move from the point of origin to the final destination without colliding with existing obstacles. The robot's capability to determine the most efficient route between two locations is referred to as path planning [3]. The Dijkstra algorithm is used to find the shortest path between the starting point and the destination. By varying nodes and adjusting weights within the system, real-time capacity in path planning is enhanced. ROS integrates the Dijkstra algorithm with the map produced by SLAM and provides the planned path for the robot, configured by RVIZ for path visualization.

This research aims to explore the implementation of SLAM-based path planning and obstacle avoidance for service robots. The author utilizes the A3 LiDAR sensor for environmental mapping and uses ROS (Robot Operating System) as the primary platform for system development and integration. Communication through ROS is used to visualize the received data and execute programs on the robot[4]. The implementation results are expected to demonstrate how the robot can effectively create maps, plan paths, and avoid obstacles in dynamic and unstructured environments. In this study, the obstacle avoidance performance for autonomous service robots is effectively verified through real-time autonomous simulation[5].

This implementation is expected to make a significant contribution to the field of autonomous robotics, especially in applications requiring safe and efficient navigation in unstructured environments. Improvements in path planning and obstacle avoidance techniques will enhance the robot's ability to operate in various practical scenarios, from household services to search and rescue missions[6]. By developing and testing this system, the author hopes to lay a stronger foundation for future research in autonomous robot navigation and provide solutions applicable to a wide range of real-world applications. This research also opens opportunities for further exploration of advanced sensors and optimization algorithms to improve the efficiency and reliability of robot navigation.

2 Method Research

This research uses the SLAM method on service robots related to obstacle avoidance data acquisition, with the LiDAR sensor functioning as a scanner that generates data from tests conducted by the author. LIDAR and SLAM work hand-in-hand to provide highly accurate, real-time mapping and localization. LIDAR offers detailed spatial data that SLAM processes to

construct and update maps in real time, allowing the robot to navigate effectively and adapt to changes in its environment. These technologies were chosen for their ability to provide precision, adaptability, and efficiency in dynamic and unknown environments, making them essential for autonomous systems that require robust and real-time mapping capabilities. The software application module uses ROS to operate the robot and its hardware. ROS serves as middleware in robotics research. It provides modules that can be directly implemented by robotics researchers and the community, and it allows for the development of new modules[7].

2.1 Overview

The robot employed in this study is a service robot equipped with four mecanum wheels, which are designed to move in any direction. These wheels consist of a central hub surrounded by several rollers that move freely at a 45° angle to the circumference of the wheel[8]. The mobile robot created is designed to provide visualization of the shape of the room into the form of a map. The sensor used is for mapping this room is LIDAR. LIDAR provides precise, real-time environmental data, while SLAM uses this data to simultaneously map the surroundings and localize the robot. Together, they enhance real-time mapping by ensuring accurate navigation and obstacle avoidance in dynamic environments.To access the LIDAR sensor, a mini PC is used. LIDAR data is sent to minipc to be processed into a map. minipc and this monitor are integrated in the [9].The author uses an A3 LiDAR sensor for mapping and obstacle detection. The LiDAR sensor is connected to a Jetson Nano, which accesses maps and obstacles, and is integrated with ROS and an ESP 32 microcontroller that controls the robot.

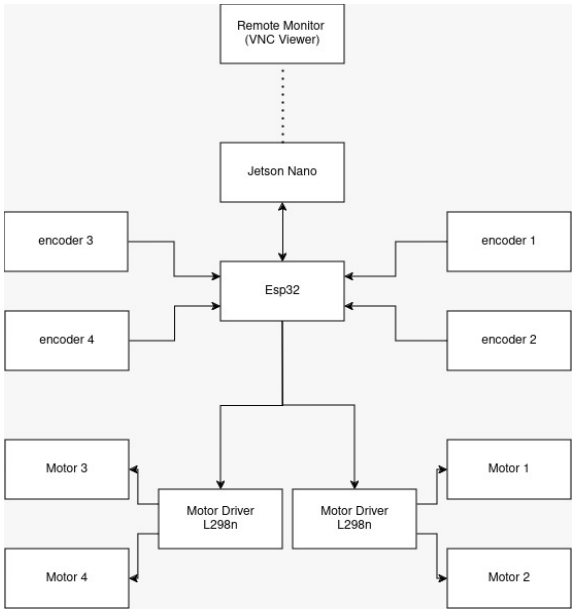

Fig.1. Block diagram

Figure 1 describes the additional software used to control the Jetson Nano as a monitor accessed from a laptop/PC, in order to activate the program on the Jetson Nano. The Jetson Nano serves as the top-level component for running SLAM and path planning. The ESP 32 serves as the input for odometry, which is calculated from four encoders and then sent to the Jetson Nano. The ESP 32 also calculates the motor speed based on the robot speed input from the Jetson Nano. This motor speed is converted into a PWM value as input for the motor driver. The motor drivers then convert these PWM values into voltage values for each motor.

2.2 Design Robots

In this research, the authors developed a protective casing for the robot using a two-tier aluminum plate. The first level is used to place hardware components such as the ESP 32, motor driver, DC motor, and step-down converter. On the second level, the author placed the Jetson Nano and RP LiDAR. The overall weight of the robot is 3.3 Kg. With a very light weight and adequate components, it is expected that the robot can navigate well. This robot has dimensions of 300mm long, 270mm wide, and 250mm high.The prototype 2D design of the service robot illustrates in Figure 2.

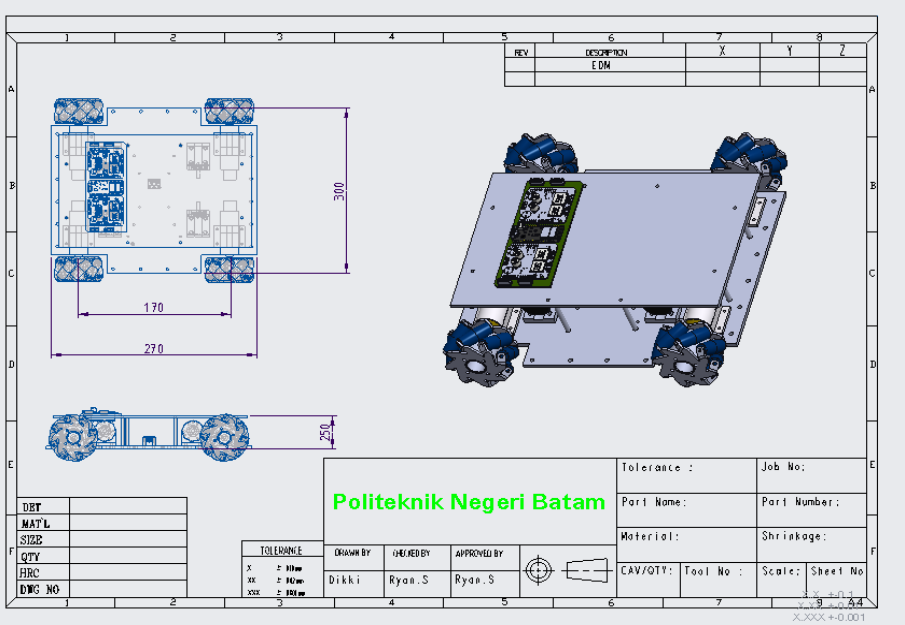

Fig.2. 2D Prototype design services robot

2.3 Ros

The Robot Operating System is an open-source framework that provides a suite of tools and libraries useful for developing programs or applications for robotic systems[10]. ROS has gained widespread recognition both in academia and industry due to its versatility, community support, and active development. It is commonly used in research, education, prototyping, and commercial robotic applications across various domains, including autonomous vehicles, drones, industrial robots, and service robots. ROS provides a flexible and modular architecture that simplifies the development of complex robotic systems. It follows a distributed computing model, where various processes (nodes) communicate with each other through a publish-subscribe messaging system.

A collection of nodes that communicate with each other is organized into packages. A package consists of nodes and other libraries[11].

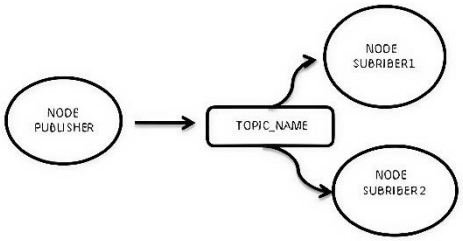

Fig.3. Comunication node

Figure 3 depicts nodes in the Robot Operating System (ROS) can be classified into two types: publishers and subscribers. Nodes communicate by exchanging information on a topic. When a

publisher node sends data to a topic, the subscriber nodes that are subscribed to that topic receive the data. A callback function is defined to be invoked each time data is received, allowing the subscriber to process the received data according to the requirements developed by the author.

2.4 Algoritma Slam dan Obstacle Avoidance

This article selects to use Dijkstra's algorithm for route planning rather than other algorithms such as A*, which are often used in robotic path planning. Dijkstra's algorithm is well suited for scenarios that require finding optimal paths from a single source to all nodes in a graph, which can be advantageous in dynamic environments where conditions change frequently. Unlike A*, which is designed for faster pathfinding with heuristic optimization, Dijkstra's algoritm ensures the shortest path is found in a comprehensive manner.
The way the author tried the slam algorithm and the obstacle barrier will be explained as follows:

2.4.1 Mapping using SLAM

SLAM plays a crucial role in the development of service robots by providing the capability to understand and interact with their environment autonomously[12]. Obstacles and barriers pose significant challenges that SLAM technology must overcome to ensure safe and efficient navigation. In this article, the author employs SLAM using the gmapping method. This method offers high mapping accuracy by utilizing lidar as a navigation scanner and odometry to determine position, orientation, and objects based on direct measurements of wheel rotations and movements [13]. Using the map generated by SLAM, the robot is able to determine the most efficient route to its destination while circumventing obstacles. Dijkstra navigation algorithm is implemented to find the shortest path without colliding with barriers. On the other hand, Gmapping uses a closed-loop approach, where if the robot returns to its initial position, the formed map will be updated [14]. Here is a detailed figure 4 explanation of map creation using SLAM:

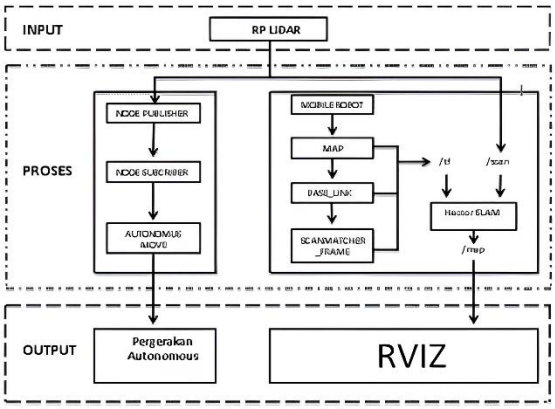

Fig.4. Diagram SLAM

Figure 4 depicts the working process of an autonomous robot using SLAM method based on Lidar sensor. The Rplidar used in this study is A3M12, which serves as the input data processed

by publisher and subscriber nodes to enable autonomous movement of the robot. Rplidar also provides input to the robot to generate SLAM maps that are visualized in RVIZ.

2.4.2 Obstacle Avoidance

Obstacle avoidance relies on real-time SLAM data to make decisions by continually updating the map of the robot's environment and its location within it. The SLAM system generates a detailed map and tracks the robot's location, which is then fed into the obstacle avoidance algorithm. This data allows the algorithm to identify potential obstacles and dynamically adjust the robot's path to avoid collisions. For effective obstacle avoidance, the real-time integration of SLAM data ensures that the robot can respond to environmental changes and navigate safely. Obstacle avoidance is a technique used in robotics to prevent robots from crashing into obstacles while moving. The process involves several stages that include sensor input, data processing, and decision-making to produce an appropriate action output.

A. Input by Lidar

Lidar is highly effective because it can generate accurate 3D maps of the robot's surroundings, enabling better obstacle detection and avoidance[15].One kind of such sensor is a 2- D lidar sensor renowned for its accuracy in measuring distances[16].

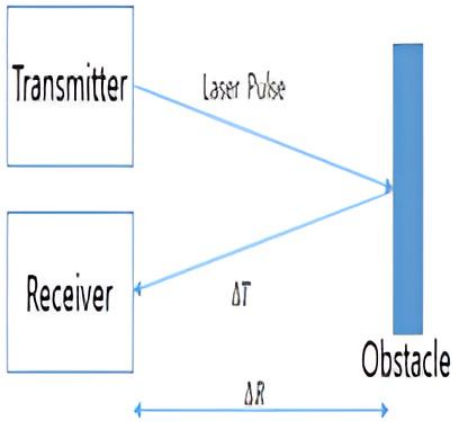

Fig.5. Principle of lidar work

Figure 5 illustrates the moment when laser pulse signals are emitted and the pulses reflected from objects are received by the receiver (17). Data collected by the LiDAR will be processed to generate points (point cloud) representing the surrounding environment. This point cloud can then be used to construct a 2D map of the surroundings. The LiDAR sensor is positioned at the front of the service robot and scans the front area with a scanner range of 400 to 1400 and a maximum distance of 12 meters.

B. Process

The TEB local planner utilizes the Timed Elastic Band (TEB) Algorithm, an adaptation of the elastic band method. Unlike the original approach, which relies on contraction and repulsion forces, the TEB algorithm incorporates timing constraints to enhance local path planning. It requires detailed information on kinematics, dynamics, geometric shape, acceleration, and velocity constraints. The TEB local planner is implemented in ROS as the Teb_local_planner package[18]. Sensors perform data collection about the robot's surroundings, including the distance to nearby objects. Object detection algorithms identify the identify the position and extent of obstacles along the robot's path. Determine safe alternate paths or calculate direction changes to avoid obstacles.

$$\text{Waypoints : } \mathrm{X_i} = (X_i, Y_i, \beta_i)^T \in R^2 \times s^1$$

Xi: The X coordinate on the horizontal axis in a two-dimensional Cartesian space (R2).

Yi: The Y coordinate on the vertical axis in a two-dimensional Cartesian space (R2).

βi: The orientation angle or heading at that position, expressed in the unit circle (S1).S1 represents the 360-degree orientation space (or 2π radians), so βi indicates the direction or orientation of the vehicle at that point. Overall, the waypoint Xi defines the position and orientation of the vehicle at a certain point in the path planned by the local planner[19]. Output data from the TEB Local Planner is the path and the data is forwarded to the next process after the TEB has generated the desired path and speed,this data is passed to the robot controller, which manages the robot's physical movements of the robot.

C. Output(Robot Movement)

The motor driver receives the speed commands and moves the robot according to the path planned by TEB.The action taken by the robot driver sends instructions to motors or actuators to change the direction or speed of the robot, if there is a path change the robot may slow down, stop, or change direction to avoid a collision.

2.5 Robots Devices

There are several hardware and software used for this research, including:

2.5.1 Hardware Devices

Table 1 provides a summary of the hardware components used in the service robot :

Table 1. Hardawe

Hardware	Qty
NVDIA Jetson Nano 2GB	1
Motor driver L298N	2
Motor DC encoder	4
RPlidar A3M12	1
ESP 32 wifi	1
Meccanum wheels	4
Battery Lippo 12V	2
Stepdown DC	1

Table 1 explenetion the NVIDIA Jetson Nano is essential for the development and functioning of autonomous robots, providing the computing power and flexibility needed to handle a variety of tasks including, mapping the environment: Using sensor data from LIDAR, and other sources, Jetson Nano helps the robot create a map of its environment. The L298N motor driver is a DC motor driver module that is most widely used or used in the electronics world which functions to control the speed and direction of rotation of a DC motor [20]. A DC motor with an encoder integrates a direct current (DC) motor with an encoder. The encoder, attached to the motor, delivers feedback on the motor's position, speed, and direction[21]. RPLidar A2M12 is a 360 degree lidar sensor developed by SLAMTEC, RPLidar A2M12 Mini PC intel NUC Arduino Atmega 2560 this sensor can perform 360 degree 2D scanning with a maximum scan radius of 12 meters. Lidar is used as a navigation system for agricultural robots. RPLIDAR is attached to the autonomous robot for collision avoidance and obstacle detection. Lidar is used in this test to enable the robot to move to avoid obstacles and localize indoors[22].ESP 32 is widely used in IoT (Internet of Things) applications due to its rich feature set, including builtin Wi-Fi and bluetooth capabilities.

2.5.2 Software Devices

The software utilized in the service robot is described in detail, including the Arduino IDE (Integrated Development Environment). This platform is employed for writing and uploading code to the board. Developed in Java, the Arduino IDE includes C/C++ libraries that simplify input and output operations. IDE plays a role to write programs, compile them into binary code and upload them into the microcontroller memory [23]. Melodic's ROS offers an extensive collection of tools and libraries for the development, simulation, and deployment of robotic applications. Nodes communicate with each other through topics, which are channels that carry messages of a certain type, such as sensor data, commands or images[24]. Visual Studio Code is a powerful and versatile code editor that serves a variety of development needs[25].The author uses visual studio code to develop and modify programs for robot needs. VNC Viewer is a versatile tool for accessing remote desktops, providing a variety of features for controlling and managing remote computers. VNC Viewer ensures efficient and reliable remote connections across multiple platforms and devices. Angry IP scanner is used for open source network scanners to detect and find open devices and display the host name of the found device to connect to.

3. Result and Discussion

The implementation of Path Planning for obstacle avoidance based on SLAM (Simultaneous Localization and Mapping) on service robots involves several key components, including environment mapping, path planning, and robot navigation. The results of this implementation include the robot's success in navigating an unfamiliar environment, avoiding obstacles, and reaching the specified destination. With the visualization in RVIZ, the author can monitor the process in real time and ensure the robot navigation goes according to plan. In the data collection and measurement on the actual using a roll meter and compared with the actual movement of the robot displayed by the ROS visual.

3.1 Obstacle Avoidance Testing

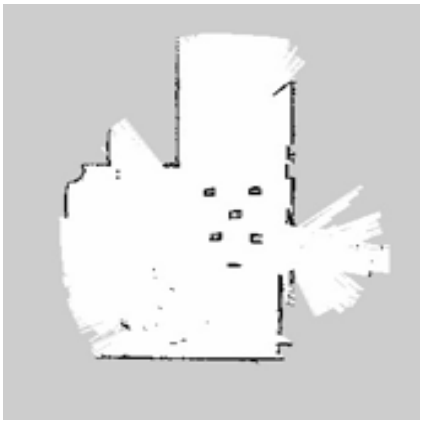

Fig.7A. Slam environment maps 1

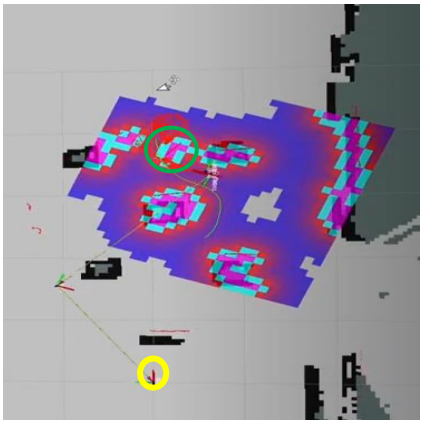

Fig.7B. Obstacle avoidance in RVIZ map 1

Fig 7A explained the slam map of the first experiment, where obstacle positions were created using cartons and brail (Polibatam robotics workshop) was chosen as the research site.
Fig 7B demonstrates the position of the robot when avoiding obstacles and has been visualized in 2D di RVIZ with lidar as input. The yellow line is the starting position of the robot and the green line is an obstacle that is placed suddenly to test the local TEB obstacle planner works well, the range from the beginning start to the destination point was found to be 4,758 meters with a travel time of 17.67 seconds.

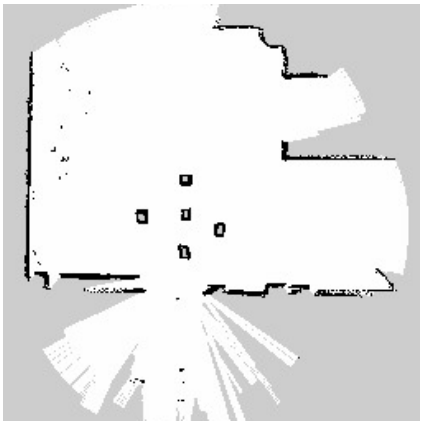

Fig.8a. Slam environment map 2

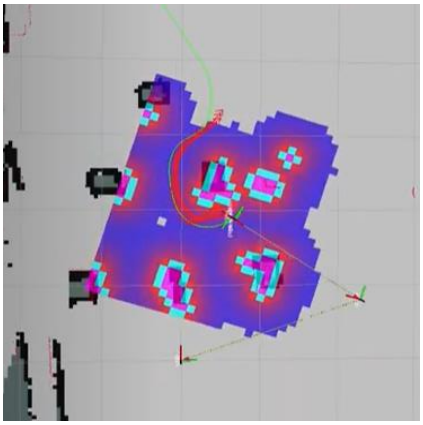

Fig.8b. Obstacle avoidance in RVIZ map 2

Fig 8A depicts the slam map of the second experiment, where the obstacle positions were changed to suit the author's needs.
Fig 8B demonstrates the position of the robot when avoiding obstacles and has been visualized in 2D di RVIZ with lidar as input. The yellow line is the starting position of the robot and the green line is an obstacle that is placed suddenly to test the local TEB obstacle planner works well, range from the beginning start to the goal was found to be 5,342 meters, with a travel time of 21.73 seconds.

Fig.9a. Slam environment maps 3

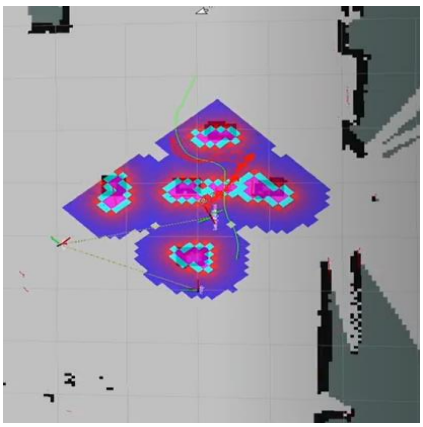

Fig.9b. Obstacle avoidance in RVIZ map 3

Fig 9A depicts the slam map of the third experiment, where the obstacle positions were changed to suit the author's needs.
Fig 10B demonstrates the position of the robot when avoiding obstacles and has been visualized in 2D di RVIZ with lidar as input. The yellow line is the starting position of the robot and the green line is an obstacle that is placed suddenly to test the local TEB obstacle planner works well, range from the beginning start to the goals destination point was found 4.887 meters, with a movement time of 20.94 seconds.

Table 2 Test result obstacle

Testing NO	Distance(m)	Time(s)	Velocity(m/s)
1	4.758	17.67	0.27
2	5.342	22.50	0.21
3	4.887	20.94	0.22

Table 2 shows the data obtained from 3 experiments to avoid obstacles. Where from the experiments obtained different data because the velocity and distance taken by the robot are different.

Table.3 Result test RPLidar

Testing Measurment RPLidar A3M12 in M												
NO	Actual Measurment (M)				Measurment in ROS				Error (M)			
	0°	90°	180°	270°	0°	90°	180°	270°	0°	90°	180°	270°
1	0.85	0.7	0.91	0.76	0.84	0.69	0.91	0.76	0.01	0.01	0	0
2	1.16	1.16	1.21	1.24	1.16	1.16	1.21	1.24	0	0	0	0.03
3	2.52	1.87	2.14	186	2.5	1.87	2.15	1.86	0.02	0	0.01	0
4	1.66	1.53	1.7	1.54	1.65	1.53	171	1.55	0.01	0	0.01	0.01
Average Error Value									0.006875			

Table 3 describes the results of lidar sensor measurements with obstacle barriers. where samples were taken 4 times and measurements were taken manually and checked displayed by ROS. where the averageerror obtained between actual and ros is 0.0068 meters or 6.8 millimeters.

4. Conclusion

In this study, a path planning system integrating the Dijkstra algorithm for obstacle avoidance based on SLAM (Simultaneous Localization and Mapping) was successfully implemented and tested on a service robot. The results of the study indicate that this approach is effective in providing accurate and efficient navigation in unstructured environments.With three time experiment data shows :

- The velocity generated by the robot will affect the movement time of the robot.If the speed generated is 0.27 m/s, the time taken to reach the target is faster and the distance traveled is further, and conversely, if the speed generated is lower, the time taken to reach the goal point is delayed..
- The average error produced by the lidar to read the obstacle is 6.8 millimeters from the actual distance.

References

[1] Peng, Y., Qu, D., Zhong, Y., Xie, S., Luo, J., & Gu, J. (2015, August). The obstacle detection and obstacle avoidance algorithm based on 2-d lidar.In 2015 IEEE international conference.

[2] Cherubini, A., & Chaumette, F. (2011, September). Visual navigation with obstacle avoidance. In 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems (pp. 1593-1598).

[3] Haq, R., Purwanto, D., & Mardianto, R. (2023, August). Microcontroller-Based Multi-Objective Genetic Algorithm for Mobile Robots Path Planning. In 2023 3rd International Conference on Electronic and Electrical Engineering and Intelligent System (ICE3IS) (pp. 122-126). IEEE.
[4] Gilliam, B., Sahai, Q., & Chandrasekaran, B. (2023, March). Path Planning and Mapping of an Autonomous Agricultural Robot Using Robot Operating System (ROS) and Gazebo. In 2023 15th International Conference on Computer and Automation Engineering (ICCAE) (pp. 528-533).
[5] Takahashi, S., & Nomura, H. (2024, May). LiDAR-only based SLAM and Ackermann Drive Navigation System, Using ROS Gmapping. In 2024 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC) (pp. 126-131). IEEE.
[6] J. Jumiyatun, "Pengendalian kecepatan motor DC menggunakan sensor encoder dengan kendali PI," ECTP, vol. 4, no. 1, pp. 23–27, Apr. 2017, doi: 10.33019/ecotipe.v4i1.15. D. Yanderson and H. M. Saputra, "Validasi RPLidar untuk Pengukur Jarak pada Mobile Robot".
[7] Wen, R., & Tong, M. (2017, August). Mecanum wheels with Astar algorithm and fuzzy PID algorithm based on genetic algorithm. In 2017 International Conference on Robotics and Automation Sciences (ICRAS) (pp. 114-118). IEEE.
[8] McInerney, Ian. Simplistic Control of Mekanum Drive. FCR Team 2022.
[9] R. W. Wolcott and R. M. Eustice, "Visual localization within lidar maps for automated urban driving," in 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, Sep. 2014, pp. 176–183.
[10] Rahman, A. (2020). Penerapan slam gmapping dengan robot operating system menggunakan laser scanner pada turtlebot. Jurnal Rekayasa Elektrika, 16(2).
[11] Liu, Z. (2021, April). Implementation of SLAM and path planning for mobile robots under ROS framework. In 2021 6th International Conference on Intelligent Computing and Signal Processing (ICSP) (pp. 1096-1100). IEEE.
[12] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "Ros: an open-source robot operating system," in ICRA Workshop on Open Source Software, 2009.
[13] G. Grisetti, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with rao-blackwellized particle filters," IEEE transactions on Robotics, vol. 23, no. 1, pp. 34–46, 2007.
[14] Madhavan, T. R., & Adharsh, M. (2019, January). Obstacle detection and obstacle avoidance algorithm based on 2-D RPLiDAR. In 2019 International Conference on Computer Communication and Informatics (ICCCI) (pp. 1-4). IEEE.
[15] Song, H., Lee, K., & Kim, D. H. (2018, December). Obstacle avoidance system with LiDAR sensor based fuzzy control for an autonomous unmanned ship. In 2018 Joint 10th International Conference on Soft Computing and Intelligent Systems (SCIS) and 19th International Symposium on Advanced Intelligent Systems (ISIS) (pp. 718-722). IEEE.
[16]"Wiki.ros.org. (2020). teb_local_planner" - ROS Wiki. [online] Available at: http://wiki.ros.org/teb local planner [Access: 9-Nov2020].
[17] I. R. Muttaqin and D. B. Santoso, "Prototype Pagar Otomatis Berbasis Arduino Uno Dengan Sensor Ultrasonic Hc-SR04," Elektro, vol. 6, no. 2, p. 41, Sep. 2021, doi: 10.30736/je-unisla.v6i2.695.
[18] Nash, A., & Koenig, S. (2013). Any-angle path planning. AI Magazine, 34(4), 85-107.
[19] D. Yanderson and H. M. Saputra, "Validasi RPLidar untuk Pengukur Jarak pada Mobile Robot".
[20] J. Jumiyatun, "Pengendalian kecepatan motor DC menggunakan sensor encoder dengan kendali PI," ECTP, vol. 4, no. 1, pp. 23–27, Apr. 2017, doi: 10.33019/ecotipe.v4i1.15.
[21] J. Arifin, L. N. Zulita, and H. Hermawansyah, "Perancangan murottal otomatis menggunakan mikrokontroller arduino mega 2560," J. n.a Infotama, vol. 12, no. 1, Feb. 2016, doi: 10.37676/jmi.v12i1.276.

[22] SVD, S. V. D., Pencahayaannya, B. I., & Wajah, U. P. (2021). Jurnal Ilmiah Setrum Article In Press.
[23] Taniguchi, H., & Nakasho, K. (2021, November). Visual Studio Code Extension and Auto-completion for Mizar Language. In 2021 Ninth International Symposium on Computing and Networking (CANDAR) (pp. 182-188). IEEE.
[24] Liu, Y., & Anshus, O. J. (2009, May). Improving the performance of vnc for high-resolution display walls. In 2009 International Symposium on Collaborative Technologies and Systems (pp. 376-383). IEEE.
[25] Fusic, S. J., Ramkumar, P., & Hariharan, K. (2018, March). Path planning of robot using modified dijkstra Algorithm. In 2018 National Power Engineering Conference (NPEC) (pp. 1-5).

FORMULIR LOOGBOOK BIMBINGAN DAN PENGAJUAN SEMINAR PROPOSAL/SIDANG TUGAS AKHIR*

Nama : Dikki Wahyudi Tarigan
NIM : 4222011004
Pembimbing I : Ryan Satria Wijaya S.Tr.T, M.Tr.T
Pembimbing II* : -
Judul : Implementasi Path Planning saat menghindari rintangan objek berdasarkan slam Pada Service robot

NO	Hari/tgl	Rincian kegiatan	TTD Pembimbing I & II	
1	Kamis 7-2-24	Development Package Jetson nano (2 GB)		
2	Kamis 15-2-24	Development / Replace Lidar A2 ke model A3M2		
3	Kamis 22-2-24	Melakukan Integrasi Package		
4	Kamis 29-2-24	Test running Rplidar with Jetson nano		
5	Kamis 7-3-24	Test running rp lidar untuk mendeteksi objek		
6	Kamis 14-3-24	Mencoba test Slam Pada Lingkungan bnail		
7	Kamis 21-3-24	Test running Rplidar dengan Slam Lingkungan		
8	Kamis 16-5-24	Test running Rplidar dengan mapping yang dibuat		
9	Kamis 23-5-24	Test running rp lidar dengan mapping yang dibuat		
10	Kamis 30-5-24	Mengganti mapping untuk test running Robot		
11	Kamis 6-6-24	Melakukan Preventive Maintenance terhadap electrical		
12	Minggu 6-6-24	Melakukan Percobaan Pengambilan data slam		
13	Kamis 13-6-24	Mengumpulkan data yang diperlukan		
14	Sabtu 15-6-24	Menyelesaikan Penulisan journal.		

Berdasarkan hasil bimbingan yang telah dilaksanakan selama 6 bulan dan telah disetujui oleh dosen pembimbing, maka dengan ini saya mengajukan diri sebagai peserta Seminar Proposal /Sidang Tugas Akhir*.

Batam, 19 Juni 2024
Peserta

Dikki Wahyudi Tarigan
NIM: 4222011004

**Hapus yang tidak perlu.*
Jumlah bimbingan minimal 10 kali. Dalam satu minggu maksimal bimbingan yang dihitung adalah 2 kali.