SSRG International Journal of Electrical and Electronics Engineering
ISSN: 2348-8379/ https://doi.org/10.14445/23488379/IJEEE-V11I5P123

Volume 11 Issue 5, 261-267, May 2024

Original Article

An Investigation of Novel Control Strategy for An AMR Mapping and Inventory Management Using Lidar Sensor

Thi-Mai-Phuong Dao[1], Thu-Ha Nguyen[1], Duy-Thuan Vu[2], Thi-Duyen Bui[3], Duc-Hiep Nguyen[1], Ngoc-Khoat Nguyen[3*]

[1]*Faculty of Electrical Engineering, Hanoi University of Industry, Hanoi, Vietnam.*
[2]*Faculty of Energy Technology, Electric Power University, Hanoi, Vietnam.*
[3]*Faculty of Control and Automation, Electric Power University, Hanoi, Vietnam.*

**Corresponding Author : khoatnn@epu.edu.vn*

Received: 14 March 2024 Revised: 14 April 2024 Accepted: 12 May 2024 Published: 29 May 2024

Abstract *- Traditional material handling methods in industrial environments often involve manually operated carts and forklifts. These methods are labor-intensive, inherently hazardous, and characterized by repetitive tasks. Autonomous Mobile Robots (AMRs) have emerged as a promising solution to address these limitations, offering the potential for significant efficiency improvements in factories and warehouses. This paper presents a research study on an AMR system designed for material handling applications. The robot is built on the Raspberry Pi 3B+ embedded computing platform and utilises the LM298 power amplifier module. These hardware components are integrated to form a robust control system. The Raspberry Pi serves as the Central Processing Unit (CPU), receiving sensor data from the A1M8 Lidar sensor. The processed data is then transmitted to the Arduino Mega 2560 microcontroller, which controls the LM298 driver circuit. The Lidar sensor plays a critical role in constructing a map of the surrounding environment and providing essential data to the Raspberry Pi. The LM298 driver circuit effectively controls the motors, enabling the robot's movement. The Raspberry Pi's Broadcom BCM2837B0 processor, a quad-core A53 (ARMv8) 64-bit SoC operating at 1.4 GHz, ensures efficient data processing and control capabilities. Experimental results verify the applicability of the control system proposed in this study in achieving reliable and efficient material handling operations.*

Keywords *- AMR, AMR vehicle, ROS, Mapping, Lidar sensor.*

1. Introduction

Amidst the current drive for economic growth, factories and workshops are rapidly expanding in scale and capacity, accompanied by a corresponding increase in workshop size. However, the movement of goods and materials within workshops has become a labour-intensive and time-consuming task. Autonomous Mobile Robots (AMRs) are gaining widespread adoption globally, playing a pivotal role in reducing the labour burden on humans and delivering superior performance.

The AMRs, with their self-navigation capabilities, represent a class of mobile robots that operate without the need for human intervention. They are characterized by their ability to self-localize, navigate, and plan their movement based on information gathered from integrated sensors and control systems. AMRs are typically equipped with a suite of sensors, including Lidar, cameras, tactile sensors, and infrared sensors, to collect data from the surrounding environment. Utilizing data processing algorithms and control systems, AMRs are able to determine the location, create maps, as well as detect and avoid obstacles when moving. AMRs employ the popular Robot Operating System (ROS) environment to deploy autonomous functionalities. ROS provides software packages and tools that provide tools and methods for building programs that control self-directed robots [1].

The AMRs represent one of the fastest-growing scientific research fields today. Their applications span a vast spectrum of domains, including industrial automation, agriculture, healthcare, eldercare, planetary exploration, entertainment, search and rescue operations, transportation, personal services, product distribution, smart warehousing, construction, sports, self-driving cars, unmanned aerial vehicle applications, and numerous others. AMRs are particularly adept at assisting humans in performing tasks in hazardous or specialized environments [2].

This diverse range of applications ensures robust growth and economic impact. Even amidst the global industry disruption caused by the COVID-19 pandemic, the mobile robotics market was projected to continue its record-breaking growth, reaching USD 23 billion in 2021 and USD 54 billion by 2023 [3, 4]. Therefore, AMRs are widely employed and will serve as the focus of this paper's research [5-10].

This paper proposes novel research and design for an AMR carrying an embedded computer and a Lidar sensor operating within a factory environment. The Lidar sensor utilized in this study is the A1M8 Lidar, which gathers spatial information surrounding the robot. Concurrently, the paper presents an intelligent control method based on Lidar data processing algorithms to ensure the robot can accurately determine its position recognize, and interact with objects in the working environment.

The research integrates systems for map building and navigation on the constructed map. This work developed an algorithm to identify the shortest path planning and move along the planned path. Subsequently, the research established and constructed an application based on the software packages available in the ROS community and developed additional software packages to suit practical requirements.

2. Hardware System Design and Dynamical Analysis

2.1. Hardware System Block Diagram

The AMR autonomous vehicle is constructed based on the following two hardware components. A LM298 power amplifier is used first. Then, a computer/laptop with Raspberry Pi 3B+ is employed. These hardware devices are interconnected according to the connection diagram in Figure 1. At the heart of the system lies the Raspberry Pi, serving as the central processing unit.

It receives signals from the Lidar sensor, processes them, and relays the processed signals to the Arduino Mega 2560 microcontroller for control of the LM298. The Lidar plays a crucial role in scanning the surrounding environment and transmitting the acquired map data to the Raspberry Pi for further processing. The LM298 acts as a buffer stage, amplifying and conditioning the control signals to drive the motors. The Lidar sensor A1M8 operates on the principle of emitting and receiving laser pulses to determine the distance and spatial information surrounding the sensor. The fundamental operational principle of the sensor employs Laser Imaging Detection and Ranging (Lidar) technology.

This sensor emits a narrow laser beam and tracks the time it takes for the laser beam to travel from the sensor to objects within the scanning range. Upon encountering an object, the laser beam is reflected and returned to the sensor. The A1M8 LiDAR sensor operates by employing a combination of filters and highly sensitive detectors. This system captures and measures the time-of-flight of a laser pulse, which is the time required for the pulse to reflect off a target and return to the sensor. The Lidar A1M8 employs a laser beam to measure distances. By varying the beam's angle, it constructs a 3D point cloud depicting the surrounding environment. Analysis of the generated point cloud data subsequently enables the localization and shape recognition of things which are in the sensor's field of view.

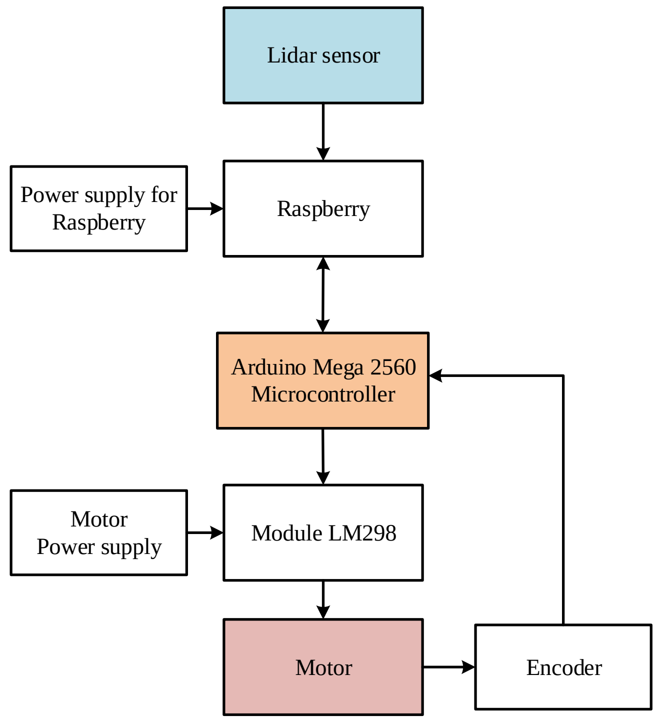

Fig. 1 Configuration of the hardware system

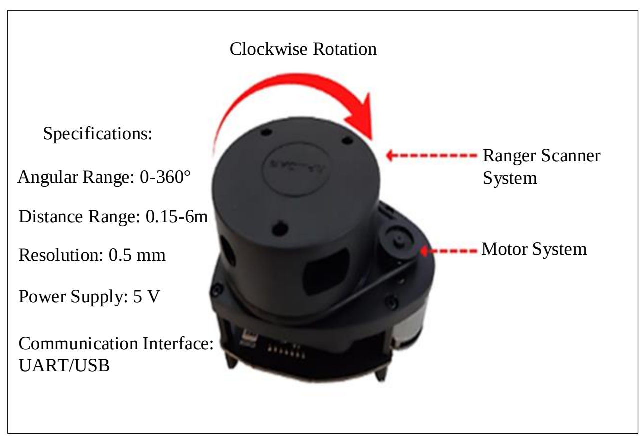

Fig. 2 Lidar sensor A1M8 [2]

2.2. Design of the AMR - Based Vehicle

In the design of autonomous vehicles, achieving simplicity, symmetrical structure, and ease of control is paramount. Utilizing basic geometric shapes with bilateral symmetry facilitates maneuverability. The design of the vehicle incorporates a two-tier architecture [11-14].

The uppermost stratum of the system houses the control circuitry, motor driver units, and a computer, which is integrated into the system. The lower tier accommodates the lifting frame for the Lidar sensor. The real technical specifications are presented in Appendix 1.

The AMRs are designed for autonomous operation and receive control signals transmitted by the Robot Operating System (ROS). These signals typically specify the desired Linear velocity (v) and Rotational velocity (ω). The embedded program on the AMR is then responsible for interpreting these commands and controlling the vehicle's actuators to achieve the specified velocities.

Fig. 3 Mechanical design of the AMR vehicle

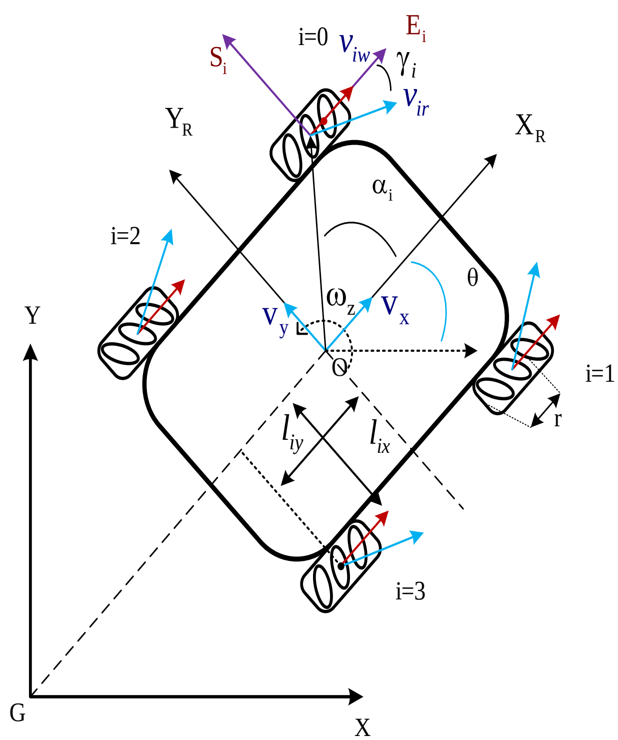

Fig. 4 Four-wheel construction of the AMR vehicle robot

The data relating to the velocity of the robot obtained from the sensors must be converted into the desired translational velocities v_x, v_y, and rotational speed ω. This work aims to achieve perfect velocity matching between the vehicle's center and the target velocity. To address the unique challenges posed by the four-wheel mecanum wheel configuration with its offset rollers, a multi-differential system is implemented.

This system functions similarly to conventional differential systems by regulating rotational forces applied to each wheel. However, unlike conventional differentials, the multi-differential system allows for independent lateral movement of each wheel, granting the vehicle improved maneuverability [5]. The determination of each wheel's specific angular speed is based on kinetic analysis. Four rotational velocities corresponding to the four wheels are computed in (1).

$$\begin{cases} \omega_1 = \dfrac{1}{r}(v_x - v_y - (l_x - l_y)\omega), \\ \omega_2 = \dfrac{1}{r}(v_x + v_y + (l_x + l_y)\omega), \\ \omega_3 = \dfrac{1}{r}(v_x + v_y - (l_x + l_y)\omega), \\ \omega_4 = \dfrac{1}{r}(v_x - v_y + (l_x + l_y)\omega), \end{cases} \tag{1}$$

From (1), it is straightforward to deduce the following:

$$\begin{bmatrix} v_x \\ v_y \\ \omega_z \end{bmatrix} = \frac{r}{4}\begin{bmatrix} 1 & 1 & 1 & 1 \\ -1 & 1 & 1 & -1 \\ -\dfrac{1}{l_x+l_y} & \dfrac{1}{l_x+l_y} & -\dfrac{1}{l_x+l_y} & \dfrac{1}{l_x+l_y} \end{bmatrix}\begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \omega_4 \end{bmatrix} \tag{2}$$

The linear velocity in x – direction is calculated in (3).

$$v_x(t) = (\omega_1 + \omega_2 + \omega_3 + \omega_4).\frac{r}{4} \tag{3}$$

The linear velocity in y – the direction is computed in (4).

$$v_y(t) = (-\omega_1 + \omega_2 + \omega_3 - \omega_4).\frac{r}{4} \tag{4}$$

The rotational speed should be computed as follows:

$$\omega_z(t) = (-\omega_1 + \omega_2 - \omega_3 + \omega_4).\frac{r}{4(l_x + l_y)} \tag{5}$$

Where: v_x and v_y denote two elements of wheel velocity obtained in the x and y directions; l_x is half the front wheel distance; l_y is half the distance between the two wheels (front and rear sides); ω_z denotes the rotational speed; ω_1, ω_2, ω_3, ω_4 are the four rotational velocities corresponding to the four wheels; and r in length, unit denotes the symmetric radius for each wheel of the robot.

3. Design of Control Algorithm

3.1. Algorithmic Diagram for the Whole System

3.1.1. Flow Chart to Motion Control

Upon completion of each timer operation cycle, the microcontroller acquires velocity values (v, ω) from global variables and calculates the required number of pulses for each left and right motor. Subsequently, it outputs the pulse signals to the driver. If an encoder is present in the system, the encoder signal is fed back to the pulse calculation unit to generate the appropriate number of pulses for the motors.

During operation, if new signals are received, data from the Raspberry Pi is transferred to the buffer via USB. Upon receiving a sufficient amount of data (bytes) in the USB buffer, an interrupt signal triggers the microprocessor. This signal prompts the microprocessor to perform the following actions:

(i) Load data structure from a buffer object;
(ii) Compute the necessary velocity values based on the retrieved data. and
(iii) Update the global variables with the calculated velocity values.

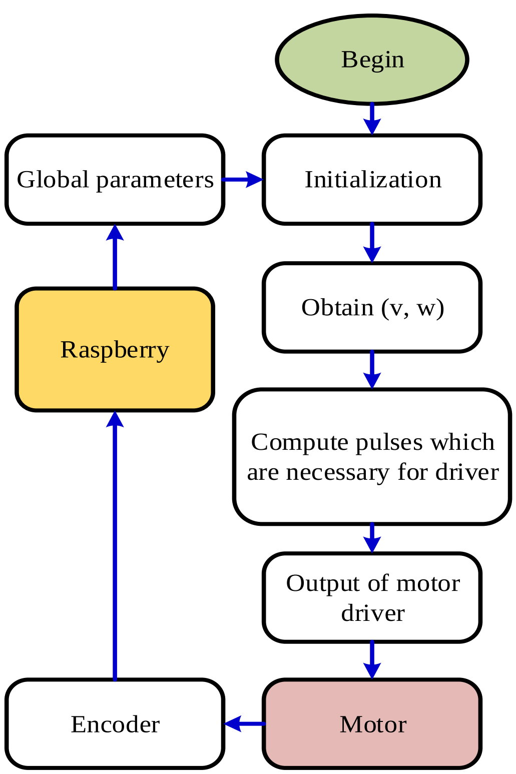

Fig. 5 Motor control – an algorithmic block representation

3.1.2. SLAM Mapping Diagram

To implement Simultaneous Localization and Mapping (SLAM), we utilize the hector_slam package for the AMRs. This is achieved by collecting data from the Rplidar sensor and converting the base frame (where the sensor is mounted) to a laser frame with the part structure, as shown in Figure 6.

The SLAM process involves a network of interconnected nodes that work together to achieve accurate navigation and mapping. The core nodes involved in this process are:

(1) Rplidar_node: This node operates the Lidar sensor and acquires sensor data. The sensor data is then published to the robot via the topic “Rplidar_msgs/LaserScan”.
(2) Key_teleop*:* This node facilitates robot movement control by publishing velocity data. The published velocity data guides the robot’s motion. AMR_core: This node receives data from Key_teleop, including the robot’s angular deviation and movement speed. While publishing the “Odom” message for odometry and robot state estimation, it also publishes the relative coordinates converted in the order odom → base_footprint → base_link → base_scan.
(3) Hector_mapping: This node generates a map based on input data comprising “tf” and “scan” messages. The output of this node is an occupancy grid map (OccupancyGrid).
(4) Map_server: This node stores the processed map from the gmapping node into files named “map.pgm” and “map.yaml”. These files represent the map used for navigation and path planning.

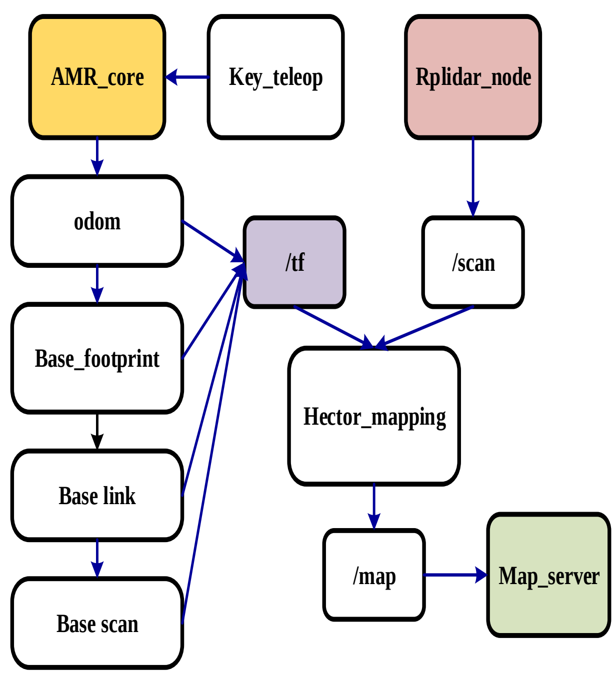

Fig. 6 SLAM mapping diagram

3.1.3. Navigation Diagram for the AMRs

Upon acquiring environmental and localization information from the SLAM system, the robot can be assigned target locations. However, these targets must be confined within the mapped environmental region to ensure successful navigation. The robot's navigation system then assumes responsibility for guaranteeing safe and autonomous motion in the real environment. This entails effectively avoiding both dynamic and static obstacles while closely adhering to a pre-defined trajectory that guides the robot from its initial position to the desired target within the mapped range.

The robot's navigation system relies on the following elements: a local planner and a global planner [15-18]. These components work together to achieve efficient navigation. Path planning for the robot involves mapping the surrounding environment using data from the perception system. Leveraging a digital map representation of the environment, the global planner within the robot's control system is tasked with strategically determining the optimal path for navigation. This process entails meticulously calculating the shortest path between the robot's current location, designated as the initial position, and the designated target location.

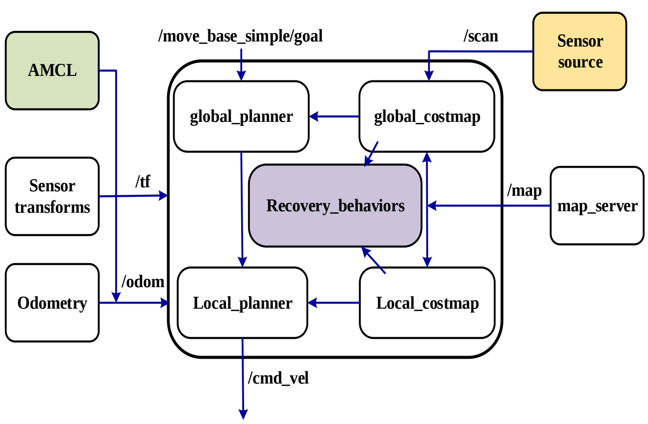

Fig. 7 Navigation diagram for the AMR proposed in this study

- Odometry: Odometry utilizes encoder readings to determine the number of pulses generated by the encoder during wheel rotation. By applying kinematic principles, the displacement of the wheel can be calculated. This information is then employed to determine the wheel's relative position compared to its initial location.
- Sensor transforms: Sensor transforms encompass the transformation of sensor data from its local frame to a common reference frame, enabling the integration of sensor measurements into a unified coordinate system. This typically involves sensors like the encoder and Lidar.
- AMCL: AMCL is an algorithm that estimates the location of the robot within a map relying on sensor data. It employs a probabilistic approach to maintain multiple hypotheses about the robot's location, gradually refining its estimation as sensor measurements become available.
- Map_server: Map_server serves as a ROS node that publishes a 2D map, typically generated from laser scans or other mapping sensors. This map provides a representation of the robot's environment and is utilized for navigation purposes.
- Sensor source: Sensor source refers to the origin of sensor data, in this case, laser scans. Laser scanners emit laser beams and measure the reflected light to create a detailed representation of the surrounding environment.
- /move_base_simple/goal: This topic serves as the input for navigation commands. It specifies the desired destination for the robot, typically in the form of a pose (x, y, orientation) within the map.
- /cmd_vel: The /cmd_vel topic publishes the linear and angular velocity commands for the robot. The base controller utilizes this information to regulate the robot's movement according to the desired motion.
- Global_costmap: Global_costmap represents a 2D map that has been integrated and processed by the global planner. It incorporates obstacle information from the robot's sensors and serves as the basis for path planning.
- Global_planner serves as the strategic decision-maker within the robot's navigation system. Its primary function entails the computation of an optimal path for the robot to traverse, facilitating its movement towards the designated goal location. It utilizes various path-planning algorithms to determine the most efficient and collision-free route within the environment.
- Local_costmap: Local_costmap represents a smaller windowed region within the global map that is centered around the current location of the robot. It provides a localized view of the environment for more immediate obstacle avoidance and reactive navigation.
- Local_planner: Local_planner employs algorithms like DWA to generate a collision-free path within the local_costmap. It considers the robot's immediate surroundings and provides velocity commands to ensure smooth and safe navigation.
- /tf provides the transform information between various components in the robot system. which includes the relationship between the robot's base frame, sensor frames, and end-effector frames.

4. Simulation and Experiment Results

4.1. Mapping Process and Map Results

Figure 8 illustrates the mapping process. The black area represents the projection of an obstacle onto the 2D plane of the map, while the red lines represent laser scans extracted from LaserScan at a fixed height above the ground.

4.2. Robot Navigation Process

To initialize the robot's position, it is mandatory first to inform it of its starting location and then specify its destination. Subsequently, the global planner will generate a path from the starting position to the desired destination.

Fig. 8 AMR vehicle scans signals that match the constructed 2D map

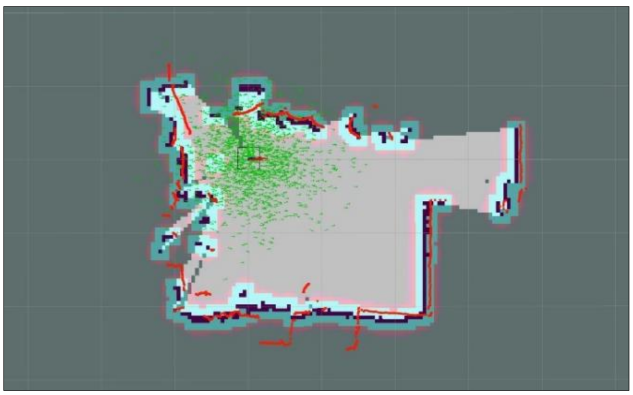

Fig. 9 Plan routes for AMR vehicles

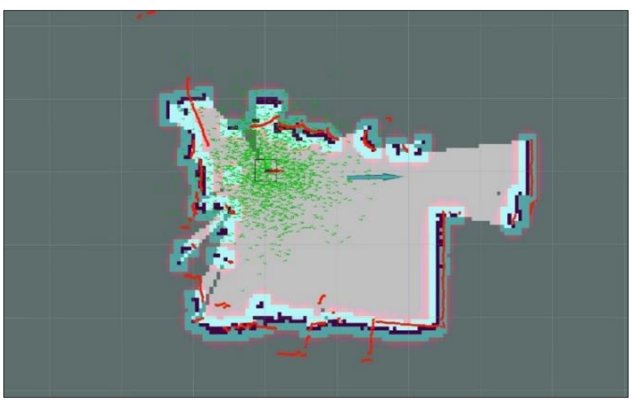

Fig. 10 The AMR vehicle tracks a scheduled route and reaches the destination

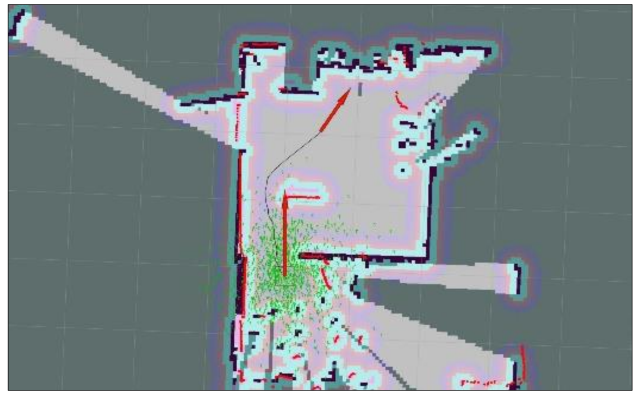

Fig. 11 Re-planning process for the AMR in case of obstacles

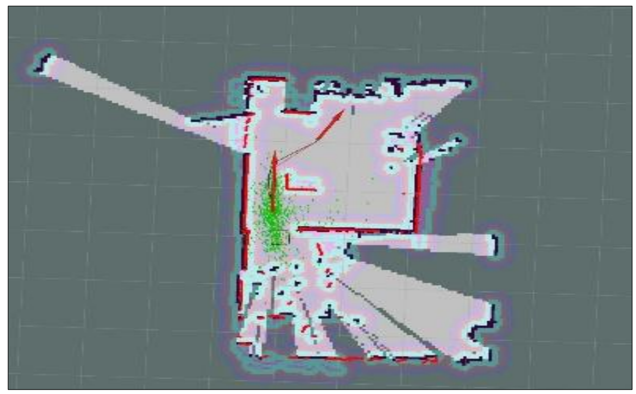

Fig. 12 The AMR is moving to avoid obstacles

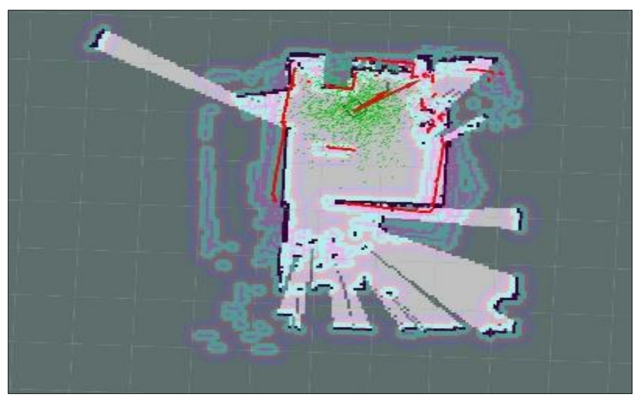

Fig. 13 An illustration of the AMR reaching the destination

In Figure 9, the blue area represents the local costmap area. The robot will then move autonomously according to the path plan calculated by the local planner. Figure 10 illustrates the robot's successful arrival at the predetermined destination.

4.3. Control of the AMR when Obstacles Occur

Obstacle avoidance during navigation is achieved using the Dynamic Window Approach (DWA) algorithm. Figure 11 illustrates the path re-planning process that occurs when obstacles are encountered during navigation. This dynamic adaptation to environmental changes is crucial for ensuring the robot's ability to reach its destination safely. The robot's navigation system exhibits the capability of dynamically re-planning its path when obstacles are detected. This real-time adaptation to environmental changes ensures the robot's ability to safely navigate and reach its intended destination (see Figure 12). The robot has reached its destination, as shown in Figure 13. The AMR's ability to promptly detect and respond to obstacles highlights the effectiveness of the proposed control algorithm. The real-time path re-planning capabilities enable the AMR to navigate around obstacles and reach its intended destination successfully. This achievement underscores the high feasibility of the proposed control algorithm in practical applications.

5. Conclusion and Future Work

Through comprehensive computational analysis, simulations, experiments, and discussions presented in this paper, we have achieved significant advancements in developing a novel control algorithm for steering robots within the ROS framework. The implemented algorithm effectively utilizes a Lidar sensor on an autonomous vehicle to perform map scanning and stable navigation. This study successfully identified the time-varying parameters governing the controller. This identification was achieved by leveraging information regarding the desired linear and angular velocities of the autonomous vehicle. By implementing these time-varying parameters, the controller is able to maintain the stability of the vehicle even when navigating across a variety of complex trajectories. Additionally, an embedded computer has been integrated into the vehicle, reducing its overall size and enhancing its maneuverability. The presented research lays the foundation for the successful construction and development of autonomous vehicles at a higher level. Future research directions will focus on refining the overall system for more comprehensive and stable operation across diverse and even challenging environments.

References

[1] Gerald Cook, and Feitian Zhang, *Mobile Robots: Navigation, Control and Sensing, Surface Robots and AUVs*, IEEE Press, 2020. [Google Scholar] [Publisher Link]

[2] Francisco Rubio, Francisco Valero, and Carlos Llopis-Albert, "A Review of Mobile Robots: Concepts, Methods, Theoretical Framework, and Applications," *International Journal of Advanced Robotic Systems*, vol. 16, no. 2, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[3] Manuel Cardona, Allan Palma, and Josue Manzanares, "COVID-19 Pandemic Impact on Mobile Robotics Market," *2020 IEEE Andescon*, Ecuador, pp. 1-4, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[4] Rplidar A1: Low Cost 360 Degree Laser Range Scanner, Introduction and Datasheet, Shanghai Slamtec. Co. Ltd., 2016. [Online]. Available: https://www.generationrobots.com/media/rplidar-a1m8-360-degree-laser-scanner-development-kit-datasheet-1.pdf

[5] Lentin Joseph, and Jonathan Cacace, *Mastering ROS for Robotics Programming: Design, Build, and Simulate Complex Robots using the Robot Operating System*, 2nd ed., Packt Publishing Ltd., 2018. [Google Scholar] [Publisher Link]

[6] Giuseppe Fragapane et al., "Planning and Control of Autonomous Mobile Robots for Intralogistics: Literature Review and Research Agenda", *European Journal of Operational Research*, vol. 294, no. 2, 2021, pp. 405-426. [CrossRef] [Google Scholar] [Publisher Link]

[7] Shuzhi Sam Ge, and Frank L. Lewis, *Autonomous Mobile Robots: Sensing, Control, Decision Making, and Applications*, CRC Press, 2006. [Google Scholar] [Publisher Link]

[8] Wikipedia, Automated Guided Vehicle. [Online]: Available: https://en.wikipedia.org/wiki/Automated_guided_vehicle

[9] Roland Siegwart, Illah R. Nourbakhsh, and Davide Scaramuzza, *Introduction to Autonomous Mobile Robots*, The MIT Press, 2nd ed., 2011. [Google Scholar] [Publisher Link]

[10] ROS.org, Costmap_2D. [Online]. Available: http://wiki.ros.org/costmap_2d

[11] Kaveh Azadeh, Rene De Koster, and Debjit Roy, "Robotized and Automated Warehouse Systems: Review and Recent Developments," *Transportation Science*, vol. 53, no. 4, pp. 917-1212, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[12] Takeshi Shimmura, Ryosuke Ichikari, and Takashi Okuma, "Human-Robot Hybrid Service System Introduction for Enhancing Labor and Robot Productivity," *Advances in Production Management Systems, Towards Smart and Digital Manufacturing*, pp. 661-669, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[13] Samyeul Noh, Jiyoung Park, and Junhee Park, "Autonomous Mobile Robot Navigation in Indoor Environments: Mapping, Localization, and Planning," *2020 International Conference on Information and Communication Technology Convergence (ICTC)*, Korea (South), pp. 908-913, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[14] N. Dinesh Kumar et al., "Mapping and Navigation of Autonomous Robots with Lidar for Indoor Applications", *Mapping and Navigation of Autonomous Robot with LiDAR for Indoor Applications*, pp. 1-11, 2023.

[15] Oliveira Junior, and Alexandre de, "*Combining Particle Filter and Fiducial Markers in a Slam-based Approach to Indoor Localization of Mobile Robots*," Master Thesis, IPB Digital Library, 2022. [Google Scholar] [Publisher Link]

[16] Morgan Quigley, Brian Gerkey, and William D. Smart, *Programming Robots with ROS: A Practical Introduction to the Robot Operating System*, O'Reilly Media Inc., USA, 2015. [Google Scholar] [Publisher Link]

[17] ROS.org, Move_base. [Online]. Available: http://wiki.ros.org/move%20_base

[18] Dang Thai Son et al., "The Practice of Mapping-based Navigation System for Indoor Robot with RPLidar and Raspberry Pi," *International Conference on System Science and Engineering (ICSSE)*, Vietnam, pp. 279-282, 2021. [CrossRef] [Google Scholar] [Publisher Link]

Appendix 1. Technical Specifications of the AMR Vehicle

Vehicle Length: 245 mm; Vehicle Width: 250 mm; Vehicle Height: 260 mm; Distance between Tier 1 and Tier 2: 100 mm; Camera Mount Height: 50 mm.