Artificial Intelligence, Medical Engineering and Education
Z.B. Hu et al. (Eds.)

doi:10.3233/ATDE231347

Design of an Improved Autonomous Tracked Vehicle Based on SLAM Algorithm

Xiaozhe YANG, Pizheng CHEN, Huiting LU
The Faculty of Intelligent Manufacturing, Nanning University, Nanning, 530299, China

Abstract. In the design of autonomous vehicles, relying solely on the SLAM algorithm for obstacle avoidance poses a certain response delay issue, which may affect the real-time safe driving of unmanned vehicles. To address this issue, a solution that identifies simulated obstacles by running YOLOv5 on the Jetson module is proposed to achieve optimized obstacle avoidance processing. At the same time, a tracked vehicle was designed using the ROS robot development system to achieve the specific effects of this solution. In the experiment, when the tracked vehicle encounters simulated obstacles during driving, it can automatically turn and deviate from the original driving trajectory, proving that this visual obstacle avoidance scheme is effective.

Keywords. Yolov5; Ros robot development; SLAM; Visual obstacle avoidance

1. Introduction

With the development of technology, autonomous driving technology has become a special concern in current society. It also represents an important direction in today's technological development to a certain extent [1,2]. To achieve autonomous driving technology, advanced sensors and increasingly optimized autonomous navigation technology are indispensable.

Autonomous navigation involves a series of core technologies such as environmental perception, map creation, autonomous positioning, and motion planning. The implementation of autonomous navigation technology for robots relies on research in many technical fields, such as the combination of perception technology, automation control technology, dynamic decision-making, path planning technology, etc., in order to build a complete robot system [5-7]. The combination of various advanced technologies enriches the functions of robots and provides them with more flexible obstacle avoidance methods, no longer limited to lidar or ultrasonic detectors [8,9]. Adding a camera to the robot not only allows for real-time acquisition of images in front for mapping, but also allows for visual recognition to check for obstacles ahead [10,11].

More and more mobile robots are extracting deep semantic features to perceive the surrounding environment through deep learning methods [12,13]. Continuous improvement and optimization are very important [14]. The method of environmental perception and autonomous navigation based on multi-sensor fusion has become a hot topic in current research [15]. It faces many challenges, and using improved algorithms can effectively solve various practical problems encountered.

2. Overall Design and Selection of Hardware and Software

Based on SLAM algorithm improvement and optimization for local obstacle avoidance, the main research method is to achieve the positioning, mapping, and path planning functions of tracked robots in the ROS system platform [16].

Use path planning algorithm to achieve global planning in the experimental site with simulated obstacles and use YOLOv5 visual recognition model to perform local obstacle avoidance to achieve dynamic obstacle avoidance of tracked robots during driving.

2.1 Overall Design Plan

The tracked vehicle is mainly composed of tracks, encoding motors, frame, battery, ROS robot STM32 drive board, Raspberry Pi 4B development board, and LiDAR. By installing cameras on the tracked robot and running YOLOv5 visual recognition on the Jetson module, the problem of the tracked robot being unable to quickly and effectively avoid obstacles in local avoidance is improved[17].

2.2 Hardware Selection of Tracked Vehicles

The RPLIDAR A1 LiDAR model can perform 360^0 laser ranging scans in all directions, up to 12 meters, and generate a two-dimensional raster map.

The ROS robot STM32 driver board consists of an STM32 chip, a USB driver chip, an attitude sensor module, a power processing chip module, a motor driver chip module, and a power circuit.

Raspberry Pi 4B is a Linux based microcomputer that can be used to run on the ROS development platform and achieve distributed multi machine communication through WiFi.

Jetson nano is a platform that supports the development of applications such as deep learning, machine learning, and computer vision, and can load YOLOv5 visual recognition onto robots on mobile platforms.

The GY-85 nine axis sensor is used to measure the speed, direction, and gravity of the equipment, and convert the data into PPM signals for feedback to the STM32 driver board.

A DC motor with an encoder can convert angular displacement or angular velocity into digital pulse signals, which are fed back to the Raspberry Pi through the STM32 driver board [18].

2.3 Software Platform - ROS Robot Development Platform

ROS is the abbreviation for robot operating system, which was proposed by Stanford University in the STAIR project in 2007, which is mainly driven by Willow Garage and developed in collaboration with other institutions. It is now widely used in robot development and provides a complete set of tools and frameworks for building, deploying, and maintaining robot systems. In the ROS system, massive, developed resource packages are collected to directly implement software functions, which increases the heavy usage rate of the software and makes the development of robots more convenient.

3. Map Construction and Path Planning

3.1 SLAM Mapping

SLAM is an abbreviation for Synchronous Localization and Mapping, first proposed by Hugh Durrant Whyte and John J. Leonard, mainly used to solve the problem of real-time localization and map building for robots running in unknown environments. Using SLAM can effectively solve the three major problems of where the robot is located, where the robot should go, and how the robot should go.

3.2 Gmapping Algorithm Mapping

Gmapping is an open-source SLAM algorithm based on 2D LiDAR that uses RBPF particle filter algorithm to separate the positioning and mapping processes, first performing localization and then mapping. The advantage of Gmapping is that it requires less computation and higher accuracy when building maps for small scenes. It is suitable for indoor robot mapping experiments in small scenes due to its low frequency requirements for LiDAR. Obtain the Gmapping function package on the ROS development platform, subscribe to the robot's depth information, IMU information, and odometer information, and complete the configuration of some necessary parameters to create a probability based 2D grid map.

3.3 Path Planning

After obtaining a map of the surrounding environment through the SLAM algorithm, the path planning algorithm enables the robot to find a collision free and safe path from the starting point to the endpoint within the specified range area.

The path planning algorithm can use the A * algorithm, which was originally published in 1968 by Peter Hart, Nils Nilsson, and Bertram Raphael of Stanford Research Institute. The A * algorithm is an extension of the Dijkstra algorithm, which improves the Dijkstra algorithm by proposing a heuristic function method that adds an estimated cost from the endpoint to the original one.

$$\boldsymbol{f}(\boldsymbol{n}) = \boldsymbol{g}(\boldsymbol{n}) + \boldsymbol{h}(\boldsymbol{n}) \tag{1}$$

In the function of the A * algorithm, *f(n)* represents the comprehensive priority of the node, which is also the node closest to the starting point and the estimated distance to the endpoint. *g(n)* represents the distance between the node and the starting point, while *h(n)* represents the estimated distance between the node and the endpoint. There are three main methods for estimating the distance, namely Manhattan distance, Euclidean distance, and Chebyshev distance.

In the Manhattan distance calculation function, it represents the sum of the horizontal and vertical distances between node a and target node b.

$$\boldsymbol{h}(\boldsymbol{n}) = |\boldsymbol{x_a} - \boldsymbol{x_b}| + |\boldsymbol{y_a} - \boldsymbol{y_b}| \tag{2}$$

The Euclidean distance calculation function represents the linear distance between node a and target node b.

$$\boldsymbol{h}(\boldsymbol{n}) = \sqrt{(\boldsymbol{x_a} - \boldsymbol{x_b})^2 + (\boldsymbol{y_a} - \boldsymbol{y_b})^2} \tag{3}$$

The Chebyshev distance calculation function represents the maximum difference in coordinates between node a and target node b.

$$\boldsymbol{h}(\boldsymbol{n}) = \boldsymbol{max}(|\boldsymbol{x_a} - \boldsymbol{x_b}|, |\boldsymbol{y_a} - \boldsymbol{y_b}|) \tag{4}$$

3.4. Improvement Based on SLAM Algorithm

In the path planning of mobile robots, a global map is usually established using LiDAR, and then mobile navigation obstacle avoidance is achieved by combining global and local path planning. However, the Dynamic Window Algorithm (DWA) requires robot position, velocity, angular velocity, and surrounding obstacle information when processing dynamic obstacle avoidance, and is susceptible to sensor noise. Inaccurate sensors or sensor information not updated in a timely manner can affect the effectiveness of the DWA algorithm, and the obstacle avoidance effect for non-convex obstacles is also poor. Additionally, it may not be possible to avoid sudden obstacles when the robot is moving at high speeds. Therefore, this article proposes to add a camera to the robot and use the YOLOv5 algorithm for dynamic obstacle recognition to assist in dynamic obstacle avoidance during the movement process.

4. Experimental Verification

4.1. Train YOLOv5 Model

Collect relevant simulated obstacle images. This article collected 106 similar images of the wheels of the robot car and used the open-source image annotation tool (labelimg) to annotate the images. After labeling the images, use YOLOv5 code to train the model to obtain a visual recognition model for such items.

4.2. YOLOv5 Code Modification

Visual obstacle avoidance requires real-time detection in YOLOv5 to determine the position of the detected obstacle by recording the real-time position coordinates of the marked block diagram. Under YOLOv5 code, by obtaining the coordinated position of the marked block diagram through the *box_label* module recorded by the p1 and p2 arrays, the specific position coordinates of obstacles on the image can be obtained.

The coordinates of the center point of the obstacle in the marked block diagram can be calculated from the coordinates of the upper left and lower right corners. Set a standard safe distance pixel so that the tracked robot can offset in a certain direction to avoid the position of the obstacle when detecting it. Use a camera to capture the center point position coordinates (320320) of frame 640 * 640 and compare them with the x-axis values of the obstacle center point position coordinates and ensure that the pixel distance between these two points is maintained outside the safe pixel distance. Otherwise, make the tracked robot turn and offset, increasing the position between two points and maintaining it beyond a safe pixel distance. The modified code can output the center position coordinates of the marked block diagram in real-time in the image, and the coordinates will also be sent to the Raspberry Pi end for processing.

4.3. System Integration

Deploying a deep learning environment with a USB camera and running YOLOv5 code in the Jetson nano to detect obstacles in front of the car and connecting the Jetson nano to the Raspberry Pi 4B using a USB data cable. When the USB camera detects obstacles in front, the YOLOv5 code is used to select the frame and send binary messages through serial communication in a timely manner to transmit the position coordinates of the frame selection center to the control system of the car, Raspberry Pi 4B, for data processing. Figure 1 shows this process. Run the ROS system in Raspberry Pi 4B to receive center point coordinate messages sent by Jetson Nano, and process binary messages to convert them into coordinate position messages. Then, create a new message node to subscribe to the processed coordinate position information, and use the if function to determine whether the center point position coordinates meet the safety pixel distance. If not, publish the speed message cmd to the ROS system_ Vel, changes the rotational speed of the tracked robot to move away from obstacles.

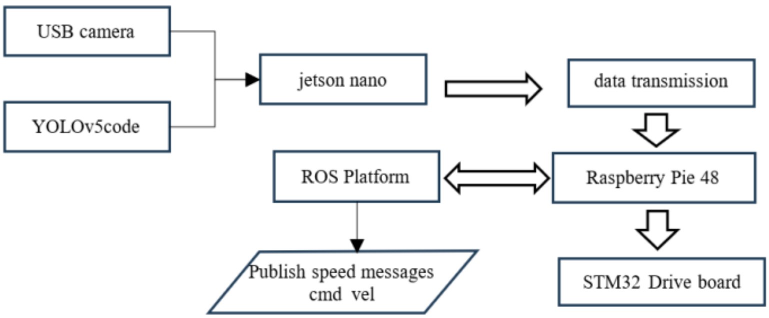

Figure 1. Visual Local Obstacle Avoidance System Diagram

4.4. Testing and Experimentation

Download the ROS robot platform from the Ubuntu system or the Ubuntu system running virtual machines on a PC. Start the power supply of the tracked vehicle, connect it to the tracked vehicle through the Ubuntu system ssh command on the PC, start the STM32 base plate, and run the LiDAR. The process of drawing is shown in Figure 2. Start the keyboard control node in the ROS terminal of the virtual machine, control the tracked robot to move, complete the mapping of the surrounding environment, and save the map.

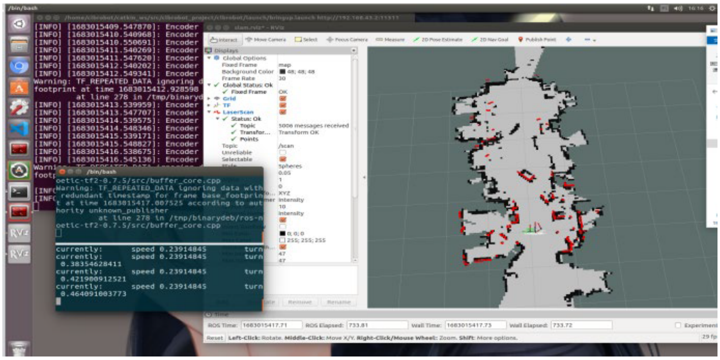

Figure 2. SLAM Mapping Process

The construction of the map is completed by connecting the virtual machine to the terminal of the tracked robot using SSH. The code in the navigate.launch file is activated to read the established map and automatically expand the contours of obstacles in the grid environment map.

Run the autonomous navigation algorithm shown in Figure 3 in the ROS terminal, use the robot simulation platform to view the working status of various sensors of the tracked robot, and calibrate the position of the tracked robot using the 2D Pose Estimate tool. Start the Jetson nano module to run YOLOv5 code to detect obstacles ahead. The actual environment is shown in Figure 4.

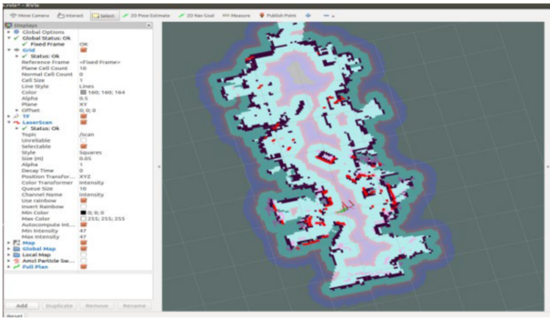

Figure 3. Autonomous Navigation Environment Map **Figure 4**. Actual Environment Map

When dynamic obstacles appear during driving, the camera on the tracked robot takes real-time pictures of the front and uploads them to the Jetson nano module for image processing.

（a） (b)

Figure 5. Effect diagram of obstacle avoidance process

As shown in the figure 5, when a wheel appears in front of the tracked robot, the YOLOv5 code detects and marks the block diagram. Because the block diagram position of the wheel is within a safe pixel distance, the tracked robot begins to turn and offset the wheel obstacle. When the block diagram position of the wheel is outside the safe pixel distance, stop turning and continue to complete the global path planning route.

From Figure 5, it can be seen that the tracked robot has turned and deviated from its original driving trajectory, proving that the visual obstacle avoidance module is an effective solution. However, the steering method of the tracked robot is too rigid, which only determines that the left or right turning movement cannot be well coordinated with the global path planning for driving. Additionally, when there are moving obstacles on both sides at the same time, it can cause the tracked robot to shake, leading to the problem of unsmooth driving of the tracked robot. These issues are currently deficiencies in visual obstacle avoidance, and we hope to strengthen code development in the future to better improve the occurrence of such problems.

5. Conclusion and Outlook

For improving the performance of autonomous vehicles, YOLOv5 can be run on the Jetson module to identify simulated obstacles and improve the automatic steering and obstacle avoidance functions of tracked vehicles. The research results of this article can achieve left and right turn obstacle avoidance for simple obstacles, but the visual obstacle avoidance system is not yet perfect. Subsequent improvement research will focus on the following aspects: achieving visual obstacle avoidance while also considering the global path planning of tracked robots. Integrating visual obstacle avoidance with path planning algorithms is expected to yield better obstacle avoidance solutions.

Visual obstacle avoidance is also an important research field in future unmanned driving, as it can more flexibly and effectively handle obstacle avoidance. Visual recognition can also achieve real-time analysis of road conditions, thereby further enhancing the safety guarantee of unmanned driving.

Acknowledgment

This project is supported by the project of 2022 Nanning University Educational reform （2022XJJG22）

References

[1] Bai, Y. Accurate obstacle prediction method in unmanned vehicle driving. [J]. Advances in Transportation Studies, 2022, 1: 3-14.

[2] Mushuang Liu, Ilya Kolmanovsky, H. Eric Tseng, et al. Potential game-based decision-making for autonomous driving[J]. IEEE Transactions on Intelligent Transportation Systems, 2023, 24(8): 1-14.

[3] Vikram Raja, Dhruv Talwar, Akilesh Manchikanti & Sunil Jha. Autonomous navigation for mobile robots with sensor fusion technology[J]. Industry 4.0 and Advanced Manufacturing, 2022, 13-23.

[4] Wang, Zengxi, Zhang, Qingyu, Jia, Tong; Zhang, Sulin. Research on control algorithm of a automatic driving robot based on improved model predictive control[J]. Journal of Physics: Conference Series, 2021, 1920(1): 012116.

[5] Jinbiao Zhu, Dongshu Wang, Jikai Si. Flexible behavioral decision making of mobile robot in dynamic

environment[J]. IEEE Transactions on Cognitive and Developmental Systems, 2023, 15(1): 134-149.

[6] Xin Xiong, Runan Huang, Cuixian He. Research on intelligent path planning technology of logistics robots based on Giraph architecture[J]. International Journal of Computing Science and Mathematics, 2022, 16(3): 252-264.

[7] Zhang Lei, Li, Dezhong. Research on mobile robot target recognition and obstacle avoidance based on vision (Article)[J]. Journal of Internet Technology, 2018, 19(6): 1879-1892.

[8] Hekm, Zhang X. Y., Ren S Q, et al. Spatial pyramid pooling in deep convolutional networks for visual recognition[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2015, .37(9): 1904-1916.

[9] Giorgio Grisetti, Cyrill Stachniss, Wolfram Burgard. Improved techniques for grid mapping with rao-blackwellized particle filters. [J]. IEEE Trans. Robotics, 2007, .23(1): 34-46.

[10] Qilie Cai; Qiang Wang; Yulong Zhang, Zhibo He, Yuhong Zhang. LWDNet-A lightweight water-obstacles detection network for unmanned surface vehicles[J]. Robotics and Autonomous Systems, 2023, 166: 104453.

[11] Liu Yuzhe. Research and implementation of mobile robot obstacle avoidance algorithm based on machine vision[D]. Chongqing University of Posts and Telecommunications, 2021 (in Chinese).

[12] Qingyong, Liu. Research on visual recognition method of unmanned driving based on deep learning[J]. Journal of Physics: Conference Series, 2021, 2010(1): 012001.

[13] ZHU Chunhao. Design and implementation of SLAM System for mobile robot based on ROS[D]. Beijing: Beijing Jiaotong University, 2020 (in Chinese).

[14] Yan Han, Wang Wei, Cui Jinhua, Dai Didi, Wang Rujia. Research on SLAM_gmapping optimization based on AF and AS algorithm[J]. Journal of Jiangsu Institute of Technology,2022, 28(2): 93-101.

[15] Wang Linlin, Chen Yunfang. Diversity based on entropy: a novel evaluation criterion in multi-objective optimization algorithm[J]. International Journal of Intelligent Systems and Applications, 2017, 10: 113-124.

[16] Mohammad Zohaib, Muhammad Ahsan, Mudassir Khan, Jamshed Iqbal. A featureless approach for object detection and tracking in dynamic environments[J]. PloS one, 2023, 18(1): e0280476.

[17] Sithmini Gunasekara, Dilshan Gunarathna, Maheshi B. Dissanayake, Supavadee Aramith, Wazir Muhammad, "Deep Learning Based Autonomous Real-Time Traffic Sign Recognition System for Advanced Driver Assistance", International Journal of Image, Graphics and Signal Processing, 2022, 14(6): 70-83.

[18] B.V. Arun Kumar, G V Marutheswar, "Multi Quadrant Operation of Brushless Direct Current Motor Drive with PI and Fuzzy Logic Controllers", International Journal of Information Engineering and Electronic Business, 2019, 11(3): 25-32.