

Design of RM Infantry Robot Based on ROS System

Yunxi Li

Hubei University of Technology, Wuhan, Hubei, 430068

Abstract: *Based on the original Robomaster infantry robot, this paper designs an infantry robot based on the ROS system, so that the ROS-based Jetson nano and PC can undertake more tasks. In this paper, an infantry robot control system based on the ROS software framework is designed; the robot prototype is built and tested, and the sensor module (PTZ IMU) is extended on the Jetson nano. The results show that the basic functions of the robot are realized and the design requirements are met.*

Keywords: ROS; Multisensor; Embedded systems; Attitude solution.

1. INTRODUCTION

With the widespread adoption of ROS as the practical standard for robot programming, there are also more and more developers and open fields. Marin Plaza Pablo. And other researchers in the research platform of autonomous vehicle based on ROS. The Robo Master Mech Master College Series is a robotics competition initiated by DJI Innovation. Infantry robots are an important member of the RM robot sequence. With the iteration and development of competitions and the emergence of new types of competitions, the performance requirements for infantry robots are increasing in both depth and breadth. Under existing conditions and new requirements, using onboard PCs to undertake more work is a suitable choice. Bohang et al. [1] proposed an active learning approach combined with hyperparameter optimization for improved image steganalysis, achieving enhanced detection accuracy. Industrial applications include Zhao et al.'s [2] deep learning-based optimization model for steel production scheduling and Yao et al.'s [3] drone-3D printing integration system for rapid post-disaster shelter construction. For sustainable urban development, Zhou et al. [4] optimized a garbage recognition model using ResNet-50 and weakly supervised CNN, while Lin et al. [5] explored intelligent exercise monitoring to improve executive function in children with ADHD. Peng et al. [6] further investigated the relationship between aerobic exercise intensity, executive function, and sleep quality. In logistics and robotics, Luo et al. [7] developed a path planning algorithm integrating Transformer and GCN networks for intelligent logistics management. Healthcare AI innovations include Shen et al.'s [8] LSTM-based system for anesthetic dose management in cancer surgery and Chew et al.'s [9] AI-driven financial risk assessment model for e-commerce platforms. Jin [10] introduced a PSO-enhanced ensemble model combining LightGBM, XGBoost, and DNNs for retail sales forecasting, while Guo et al. [11] proposed a hybrid ensemble method with focal loss to address imbalanced datasets. Finally, Weng et al. [12] presented a multi-task fusion framework with adaptive weighting for improved model alignment in intelligent robotics applications.

2. ROBOT SYSTEM DESIGN

2.1 Task analysis

The original infantry robot was equipped with STM32 sensors for data acquisition and processing (excluding vision), chassis and gimbal, and a mini PC for visual recognition and data transmission via USB to assist STM32 in judgment. This article is based on the original Robomaster infantry robot and designs an infantry robot based on ROS system, enabling NVIDIA Jetson nano and PC based on ROS to undertake more functions. In terms of functional implementation, the robot needs to complete the following functions, as shown in the framework in Figure 1.

- (1) Realize PC controlled omnidirectional movement of robots.
- (2) Small gyroscope motion can be achieved during movement.
- (3) Realize robot state recognition under multiple sensors.

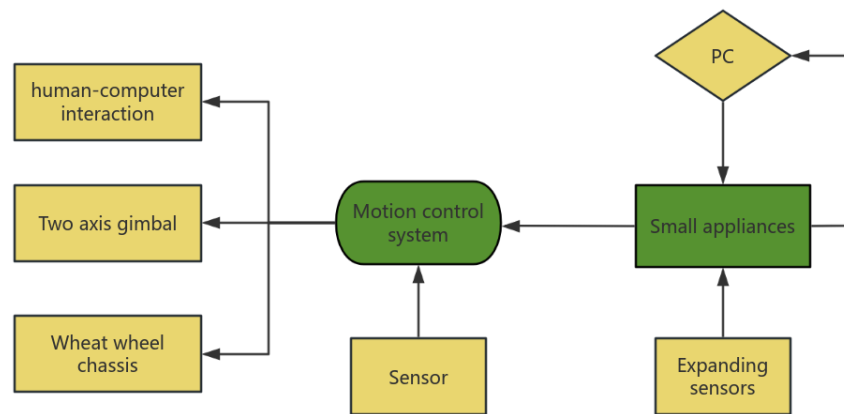


Figure 1: Robot Framework

2.2 Design and construction of robot hardware platform

2.2.1 Dual axis gimbal and Mecanum wheel chassis design

The dual axis of the dual axis gimbal refers to the Pitch axis and Yaw axis, which are controlled by two actuators that respectively control the yaw angle of the Yaw axis and the pitch angle of the Pitch axis. The prototype was built using two HWZ020 servos of different specifications to control the yaw axis heading angle and pitch axis pitch angle, respectively. Among them, the HWZ020 servo that controls the yaw angle of the Yaw axis has a maximum turning angle of 270°; The HWZ020 servo that controls the pitch axis pitch angle has a maximum angle of 180°.

Mecanum Wheel, also known as Ilon Wheel. The layout of the Mecanum wheel is mainly divided into X and O types, with Robomaster's assembly method used here for the X type. The Mecanum wheel chassis selected for prototype construction consists of four sets of Mecanum wheels. Each group includes one MG513P30-12V motor, one Hall encoder, and one 60mm Mecanum wheel.

2.2.2 Controller and Sensor Design

In addition to the chassis and gimbal, robots also need to serve as controllers for the upper and lower computers, as well as sensors for measuring various data to assist decision-making.

(1) Upper computer (main controller):

The basic architecture and usage methods of mainstream industrial control computers are similar, but there are differences in performance. NVIDIA Jetson Nano meets the design requirements, and this article chooses it as the main controller, mainly responsible for IMU acquisition, analysis of sensor data sent by STM32, and communication with the PC end.

(2) Lower computer (STM32):

The controller of the motion control system. According to the Robomate kit standard, it is composed of STM32F407VET6 microcontroller core board and expansion board, mainly used for the implementation of robot motion control system and the collection of related sensor data. At the same time, it has interfaces supporting PS2, OLED, serial communication, IMU data acquisition and other functional module access.

(3) IMU:

Inertial measurement unit is a device that measures the three-axis attitude angle (or angular velocity) and acceleration of an object. In order to perform robot posture calculation, there is one on each of the Mecanum wheel chassis and the dual axis gimbal. The IMU on the Mecanum wheel chassis is connected to STM32, and data is

collected by STM32; The IMU on the dual axis gimbal is directly connected to the main controller to conduct research on ROS based sensors for the main controller.

The chassis IMU is an ICM20948 integrated on the STM32 chassis motherboard. ICM20948 is a nine axis IMU launched by TDK. Select N100 module for gimbal IMU. The N100 module is a nine axis attitude sensor with a Type-C interface that supports communication with ROS.

2.3 Robot control system software design

This article is based on the ROS software framework to construct an infantry robot control system.

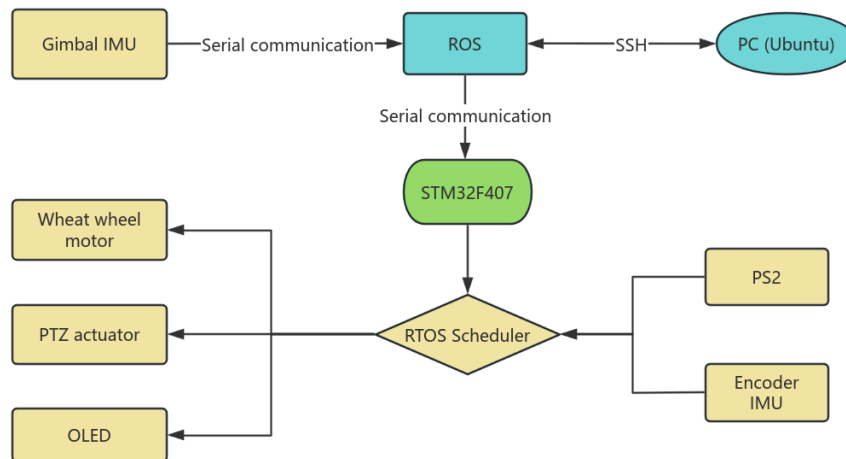


Figure 2: Software Design Framework

According to functional requirements, the infantry robot control system constructed is divided into three layers: PC end; Robot operating system based on industrial computer; The software design framework for the motion control system based on STM32 is shown in Figure 2.

2.4 Robot control strategy design

2.4.1 Attitude calculation algorithm

The posture and heading of the robot display the relative position relationship between the vehicle's own coordinates and the navigation coordinate system. The quaternion method is convenient for calculation and can more reasonably describe the rigid spatial motion of the carrier, and is widely used.

Runge Kutta method is a high-precision single step quaternion real-time update algorithm. The first-order Runge Kutta method approximates the slope of time period T using the slope of the initial endpoints. Applied to quaternions, the input parameter is the value of a three-axis gyroscope, and the output updates the quaternion, as shown in Equation 1.

$$\begin{pmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{pmatrix}_{1+\Delta t} = \begin{pmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{pmatrix} + \frac{\Delta t}{2} \begin{pmatrix} -\omega_x \cdot q_1 - \omega_y \cdot q_2 - \omega_z \cdot q_3 \\ +\omega_x \cdot q_0 - \omega_y \cdot q_3 + \omega_z \cdot q_2 \\ +\omega_x \cdot q_3 + \omega_y \cdot q_0 - \omega_z \cdot q_1 \\ -\omega_x \cdot q_2 + \omega_y \cdot q_1 + \omega_z \cdot q_0 \end{pmatrix} \quad (1)$$

2.4.2 Small gyroscope algorithm

A small gyroscope travels, as the name suggests, while rotating, it moves in all directions according to a certain coordinate system. When the robot starts to move with a small gyroscope, it obviously cannot achieve the desired effect by moving according to the relative coordinate system of the chassis. In RoboMaster, infantry robots are

composed of two main parts: a gimbal and a chassis. Therefore, according to actual needs, choose to move in all directions according to the relative coordinate system of the gimbal.

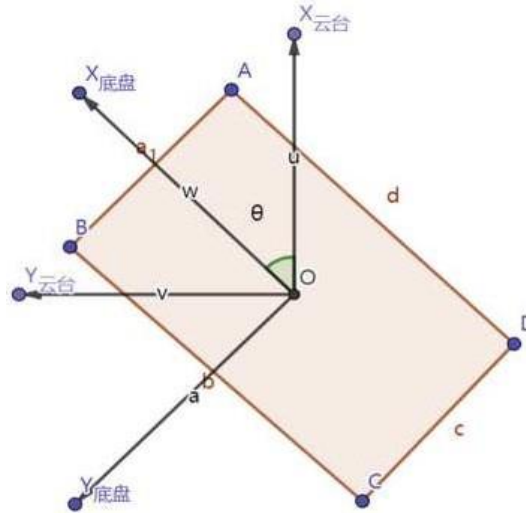


Figure 3: Schematic diagram of chassis relative coordinate system and pan tilt relative coordinate system

As shown in Figure 3, calculate the angle θ between the chassis relative coordinate system and the gimbal relative coordinate system, and convert the robot's motion speed (magnitude and direction) from the gimbal relative coordinate system to the chassis relative coordinate system, as shown in Equations 2 and 3.

$$x_{\text{底盘}} = y_{\text{云台}} \sin \theta + x_{\text{云台}} \cos \theta \quad (2)$$

$$y_{\text{底盘}} = y_{\text{云台}} \cos \theta - x_{\text{云台}} \sin \theta \quad (3)$$

Decompose the speed of the chassis relative coordinate system robot into wheel speed, and the overall effect should be manifested as movement in the pan tilt relative coordinate system.

2.4.3 PID algorithm

The PID controller uses proportional constants to improve the system's regulation accuracy, differential constants to accelerate response speed, and integral constants to eliminate steady-state errors based on system errors, in order to calculate the required control variables for control. Because computer control is sampling control. Therefore, the implementation of PID control law must use numerical approximation methods. By simulating the discretization of PID into a differential equation, positional PID (Equation 4) and incremental PID (Equation 5) are obtained, where $e(k)$ is the input, $e(k-1)$ is the previous input, out is the output, and sum : $e(k)$ cumulative value.

$$out_{\text{位置}} = P \times e(k) + I \times sum + D \times [e(k) - e(k-1)] \quad (4)$$

$$out_{\text{位置}} = P \times [e(k) - e(k-1)] + I \times e(k) + D \times [e(k) - 2e(k-1) + e(k-2)] \quad (5)$$

3. ROBOT SOFTWARE DESIGN AND IMPLEMENTATION

3.1 STM 32 Program framework based on Free RTOS

The STM 32 program is written based on Free RTOS. The most important advantages of Free RTOS for design purposes are its strong readability and portability.

This article designs 7 tasks based on the required functions: "Balance task", "ICM 20948_task", "show_task", "led_task", "PSTWO_task", "DATA_task", "Control_task". They are respectively robot chassis motion control task, chassis IMU data reading task, OLED display screen reading task, LED flashing task, PS2 handle control task, sensor data transmission task, and pan tilt motion control task. Meanwhile, the commands sent by ROS through

serial port 3 are processed by the corresponding interrupt response program, as shown in the program framework diagram in Figure 4.

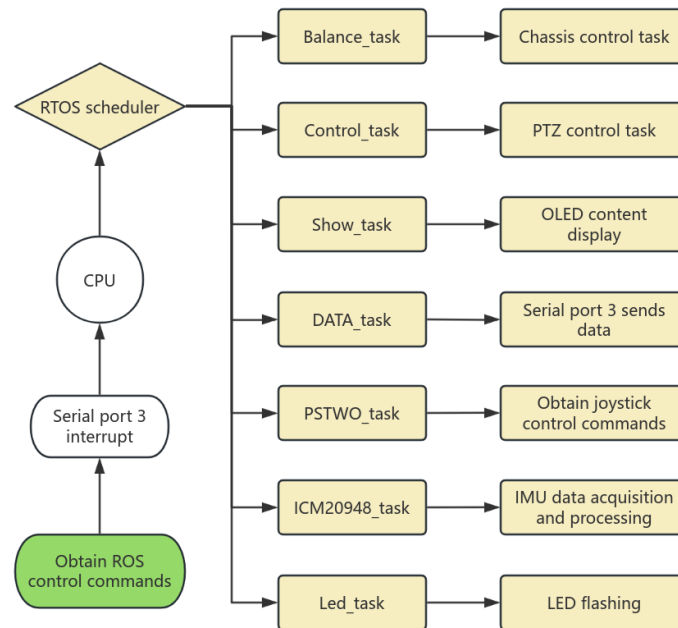


Figure 4: STM32 program framework based on FreeRTOS

3.2 Implementation of robot motion control program

3.2.1 Chassis kinematics implementation

As per the program framework mentioned earlier, the 'Balance Task' task is responsible for controlling the motion of the chassis. In the program, the three-axis target velocities are obtained through PS2 or PC keyboard input, and the kinematic inverse solution is performed to obtain four wheel target velocities. The kinematic implementation logic is shown in Figure 5.

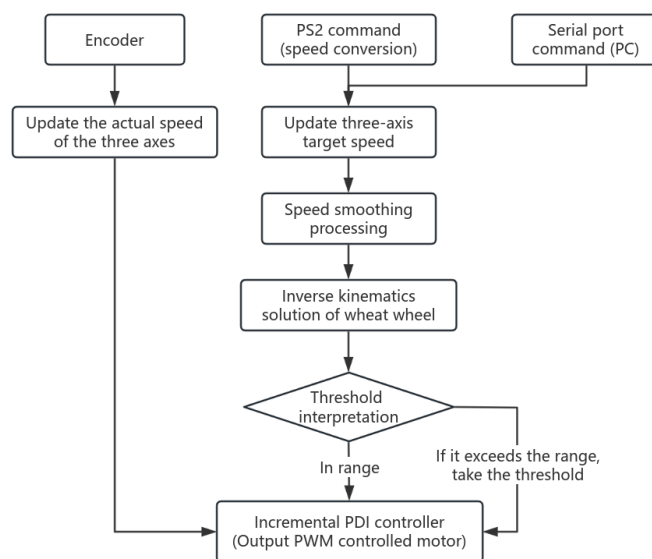


Figure 5: Kinematic Implementation Logic Diagram

There is a difference between the target speed and the actual speed, so the PI incremental PID controller in section 2.4.3 is used to make the actual output speed of the motor approach the target value.

3.2.2 Implementation of pan tilt control

Both PS2 control and upper computer serial port control achieve servo control by setting and changing the target angle of the servo. The target angle needs to be converted into the target PWM value to control the servo. The implementation process of pan tilt control is shown in Figure 6. In order to better control the effect, a position based PID controller is needed here to make the actual output speed of the motor approach the target value.

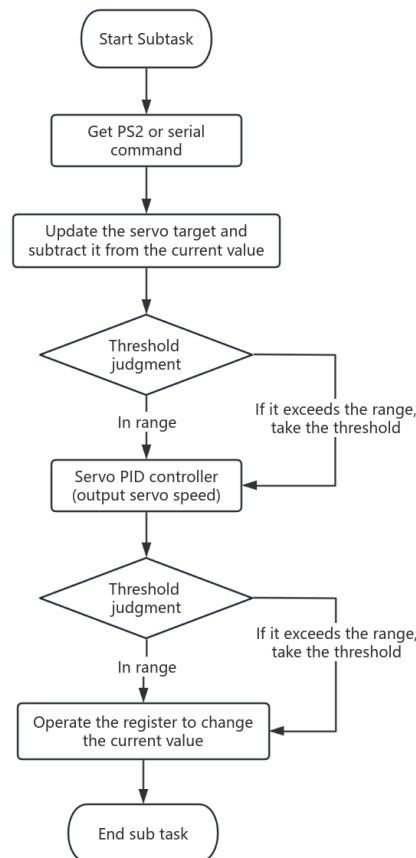


Figure 6: Implementation flowchart of pan tilt control

3.3 Attitude calculation program design

3.3.1 Chassis IMU attitude calculation design

The raw data of ICM20948 needs to undergo zero drift elimination. The calculation of heading angle uses Runge Kutta method. After initialization, first execute the gyroscope to remove zero drift, and then read the raw angular velocity data. Process to obtain three-axis angular velocity in radians. Substitute the formula to update the quaternion, take the modulus and normalize the updated quaternion, and use matrix knowledge to calculate the attitude angle.

3.3.2 gimbal IMU attitude data reading

The default communication method used by the N100 inertial navigation module is serial communication. According to the communication protocol, the required IMU data can be read through ROS as the attitude of the gimbal. Due to the multiple types and varying lengths of data sent by the N100 inertial navigation module, it is

necessary to classify and receive the data. Determine whether the received data is valid based on the header and footer of the received data frame, and then determine the type of data based on the instruction type and data length.

3.4 Implementation of small gyroscope movement

The algorithm implementation of the small gyroscope's movement is divided into two parts, the gimbal and the chassis. The implementation effect should be that the chassis continuously rotates while being able to move in all directions according to the gimbal's relative coordinate system.

Chassis part: According to the principle of the small gyroscope algorithm, the angle α and β between the relative coordinate system of the pan tilt and the relative coordinate system of the chassis and the reference coordinate system can be obtained through the data of the two IMUs of the pan tilt and the chassis. Furthermore, the difference between the two angles α and β can be obtained, that is, the angle θ between the relative coordinate system of the chassis and the relative coordinate system of the pan tilt. According to the derivation results in section 2.4.2, the chassis behaves as moving in the relative coordinate system of the gimbal.

Gimbal section: When moving in the gimbal relative coordinate system, the angle α between the gimbal relative coordinate system and the reference coordinate system should remain unchanged without any command to change the yaw axis heading angle of the gimbal. ROS calculates the attitude angle from the N100 module and sends it to STM32 through the serial port. STM32 outputs the servo PWM value control based on the difference in yaw axis heading angle of N100, and uses a PID controller to stabilize the value of angle α .

In hardware selection, the maximum rotation angle of the servo controlled by the bottom of the gimbal of the robot prototype is 270° , which cannot ensure that the yaw axis heading angle of the gimbal remains unchanged when the wheel chassis rotates automatically. Therefore, alternative implementation solutions are proposed. The relative coordinate system of the chassis at the initial power on position of the robot is used as the reference coordinate system for the movement of the small gyroscope. At this point, θ is the yaw axis heading angle ω calculated by the chassis IMU. Using the heading angle ω calculated by the chassis IMU as the entry parameter, as shown in Figure 7, executing the small gyroscope program should result in the robot rotating while being able to move in all directions in the chassis relative coordinate system at its initial position.

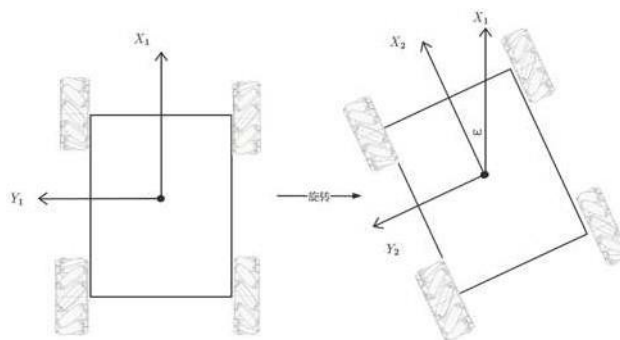


Figure 7: Implementation of Quaternion Solving Program

3.5 Robot communication design and implementation of human computer interaction in the lower computer

3.5.1 Implementation of human-machine interaction in the lower computer

The human-machine interaction of the chassis control system consists of two parts: OLED display and PS2 remote control. In the RTOS scheduler, there are two tasks: "show_task" and "PSTWO_task". By using OLED display and PS2 remote control, the logic shown in Figure 8 is implemented to receive PS2 control commands without ROS commands.

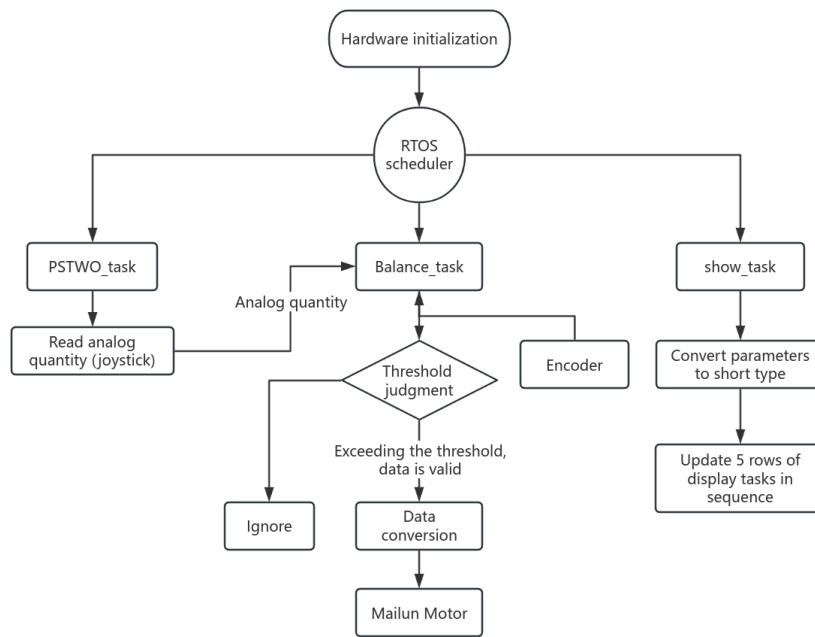


Figure 8: Logic diagram of human-computer interaction implementation

3.5.2 STM32 and upper computer communication

The communication between ROS and STM32 controller is achieved through serial port, and the communication protocol includes: STM32 controller sends data to ROS, and ROS sends data to STM32 controller. STM32 sends data to ROS using the 'data_task' task. The data frame sent includes three-axis velocity, angular velocity, acceleration, as well as information such as frame header, frame footer, and checksum included in the communication protocol. STM32 uses interrupt reception to receive data, including enable control flags, robot three-axis target speeds, and data check bits.

ROS runs on Jetson Nano, where STM32 collects information from the chassis IMU and encoder, and then transmits sensor data and sends control commands through serial port 3 and Jetson Nano. ROS sends the target three-axis motion speed and gimbal servo angle to STM32 to control the robot's motion: first, create a keyboard_teleop node, publish topics related to three-axis speed control commands, and then subscribe to the wheeltec_robot node and send them to STM32 through serial communication.

3.5.3 Communication between Nano and PC

Secure Shell (SSH) is a secure network protocol built on the application layer that can be used for remote login. This article uses a free and open-source implementation of SSH in Ubuntu, which makes it easy to remotely log in to Jetson Nano and operate robots. For example, the three-axis target speed sent by the ROS system can be input through the keyboard on the PC side.

4. IMPLEMENTATION AND ANALYSIS OF ROBOT FUNCTIONS

The actual prototype of the infantry robot built is shown in Figure 9.

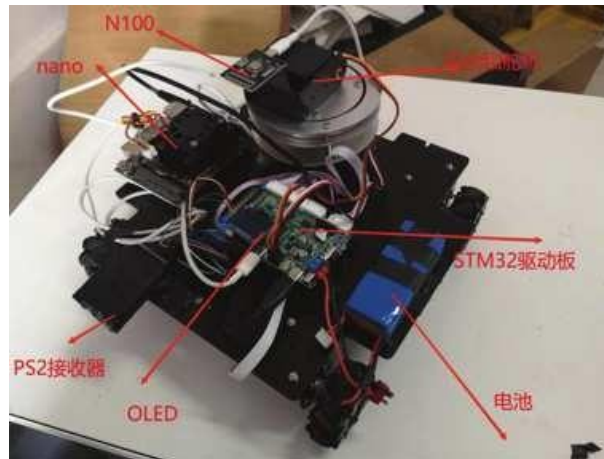


Figure 9: Physical image of the robot

4.1 Functional testing of infantry robots

4.1.1 Communication testing between Nano and PC

The Ubuntu system on the PC and Jetson nano communicate remotely through SSH login. SSH provides two levels of authentication methods: the first level (password based security authentication): logging into a remote host using an account and password. All transmitted data will be encrypted. The format of the password login command is: `ssh client username - Y @ server IP address`. Enter the above command, and the command prompt will be changed to the prompt of the remote host, proving that the operation after executing the command is Jetson nano.

4.1.2 Human Computer Interaction and Motion Control Testing

The OLED displays the target speed and actual speed of the four wheels, which can be modified to display the Z-axis angular velocity or yaw heading angle calculated by the chassis IMU module (ICM20948) according to actual needs.

The omnidirectional movement of the chassis and the rotation of the gimbal can be controlled by PS2 or PC. The priority of control commands sent by the PC through the upper computer is higher than that of PS2 (STM32 serial port 3 interrupt reception). Figure 10 shows the display interface for PC controlled chassis movement, which is performed on the logged in terminal, and the same applies to pan tilt control.

```

/home/xitefel/grad_ws/src/wheeltec_robot_rc/launch/keyboard_teleop.launch http://local...
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
ROS_MASTER_URI=http://localhost:11311

setting /run_id to 1c20bdf8-d5d5-11ec-9c67-74d83e4652fd
process[rosout-1]: started with pid [10787]
started core service [/rosout]
process[turtlebot_teleop_keyboard-2]: started with pid [10790]

Control Your Turtlebot!
-----
Moving around:
  u    l    o
  j    k    i
  m    ,    .

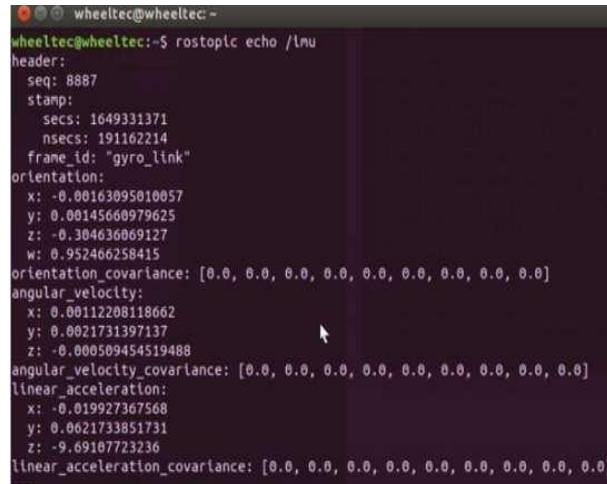
q/z : increase/decrease max speeds by 10%
w/x : increase/decrease only linear speed by 10%
e/c : increase/decrease only angular speed by 10%
space key, k : force stop
anything else : stop smoothly
b : switch to OnnMode/CommonMode
CTRL-C to quit

currently:  speed 0.2      turn 0.5

```

Figure 10: PC Control Display Interface

The partial N100 inertial navigation module parameters published in the command line terminal through ROS on the PC side are shown in Figure 11.



```
wheeltec@wheeltec:~$ rostopic echo /imu
header:
  seq: 8887
  stamp:
    secs: 1649331371
    nsecs: 191162214
  frame_id: "gyro_link"
orientation:
  x: -0.00163095010057
  y: 0.00145660979625
  z: -0.304636069127
  w: 0.952466258415
orientation_covariance: [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
angular_velocity:
  x: 0.00112208118662
  y: 0.0021731397137
  z: -0.000509454519488
angular_velocity_covariance: [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
linear_acceleration:
  x: -0.019927367568
  y: 0.0021733851731
  z: -9.69107723236
linear_acceleration_covariance: [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
```

Figure 11: PC terminal displays IMU data

4.2 Error analysis of small gyroscope travel test and attitude calculation

The test results indicate that the alternative solution can basically achieve small gyroscope operation. However, prolonged operation can result in data drift, causing the trajectory of the robot's small gyroscope to drift.

Record the yaw axis heading angle data of IMU attitude calculation over a period of time, with counterclockwise direction as positive. As shown in Table 1.

Table 1: PC terminal displays IMU data

Module	Number of times	Measuring angles (unit°)			
		0	90	180	270
N100	1	0.0	90.1	180.0	270.0
	2	0.1	90.1	180.2	269.9
ICM20948	1	0.0	89.2	180.4	269.9
	2	0.5	90.8	181.3	272.2
	3	3.4	94.4	183.5	275.5
	4	6.5	97.8	188.8	279.0
	5	10.7	101.1	193.6	284.9

Obviously, the IMU data of the gimbal IMU N100 meets the design requirements. There is a significant drift error in the attitude data of the chassis IMU. The reason analysis is as follows:

(1) The Runge Kutta method for calculating yaw axis heading angle essentially integrates the Z-axis angular velocity. Although it handles zero drift, it is difficult to eliminate the influence of drift.

(2) The calculation of yaw heading angle using a nine axis IMU can only be done with a gyroscope and a magnetometer. The ICM20948 is integrated on the STM32 drive board, and the distance between it and the various modules of the robot is not large. The magnetometer is easily affected and calibrated, so the effectiveness of the three-axis magnetometer with ICM20948 cannot be guaranteed.

In RoboMaster, the implementation of the small gyroscope relies on the absolute encoder of the yaw axis motor of the gimbal and the IMU module on the gimbal. Infantry robots are allowed to be equipped with conductive slip rings and omnidirectional motors, so this issue in the small gyroscope test in this section can be avoided.

5. CONCLUSION

This article presents the design of an infantry robot based on ROS system, and builds a robot prototype to enable Jetson nano and PC based on Linux and ROS to undertake more tasks.

Currently, ROS has been widely used, with a significant increase in the number of developers and a wider range of design fields. Meanwhile, in RM's new competition format, taking automatic infantry as an example, robots autonomously make decisions without an operator, but can receive the relative positions of the two sides based on a small map provided by the radar station. So the inertial navigation module installed on the gimbal, which directly receives data from the onboard PC, can help the robot determine the navigation coordinate system. And connected to the onboard PC for more timely data reception, while saving STM32 computing resources.

Of course, with the advancement and integration of technology, better control solutions and superior sensors will surely emerge one after another.

REFERENCES

- [1] Bohang, L., Li, N., Yang, J. et al. Image steganalysis using active learning and hyperparameter optimization. *Sci Rep* 15, 7340 (2025). <https://doi.org/10.1038/s41598-025-92082-w>
- [2] Zhao, H., Chen, Y., Dang, B., & Jian, X. (2024). Research on Steel Production Scheduling Optimization Based on Deep Learning.
- [3] Yao, T., Jian, X., He, J., & Meng, Q. (2025). Drone-3D Printing Linkage for Rapid Construction of Sustainable Post-Disaster Temporary Shelters.
- [4] Zhou, Y., Wang, Z., Zheng, S., Zhou, L., Dai, L., Luo, H., ... & Sui, M. (2024). Optimization of automated garbage recognition model based on resnet-50 and weakly supervised cnn for sustainable urban development. *Alexandria Engineering Journal*, 108, 415-427.
- [5] Lin, L., Li, N., & Zhao, S. (2025). The effect of intelligent monitoring of physical exercise on executive function in children with ADHD. *Alexandria Engineering Journal*, 122, 355-363.
- [6] Peng, Y., Zhang, G., & Pang, H. (2025). Impact of Short-Duration Aerobic Exercise Intensity on Executive Function and Sleep. *arXiv preprint arXiv:2503.09077*.
- [7] Luo, H., Wei, J., Zhao, S., Liang, A., Xu, Z., & Jiang, R. (2024). Intelligent logistics management robot path planning algorithm integrating transformer and gcnn network. *IECE Transactions on Internet of Things*, 2(4), 95-112.
- [8] Shen, Z., Wang, Y., Hu, K., Wang, Z., & Lin, S. (2025). Exploration of Clinical Application of AI System Incorporating LSTM Algorithm for Management of Anesthetic Dose in Cancer Surgery. *Journal of Theory and Practice in Clinical Sciences*, 2, 17-28.
- [9] Chew, J., Shen, Z., Hu, K., Wang, Y., & Wang, Z. (2025). Artificial Intelligence Optimizes the Accounting Data Integration and Financial Risk Assessment Model of the E-commerce Platform. *International Journal of Management Science Research*, 8(2), 7-17.
- [10] Jin, T. (2025). Optimizing Retail Sales Forecasting Through a PSO-Enhanced Ensemble Model Integrating LightGBM, XGBoost, and Deep Neural Networks.
- [11] Guo, X., Cai, W., Cheng, Y., Chen, J., & Wang, L. (2025). A Hybrid Ensemble Method with Focal Loss for Improved Forecasting Accuracy on Imbalanced Datasets.
- [12] Weng, Y., Fan, Y., Wu, X., Wu, S., & Xu, J. (2024, November). A Multi-Layer Alignment and Adaptive Weighting Framework for Multi-Task Model Fusion. In *2024 International Conference on Intelligent Robotics and Automatic Control (IRAC)* (pp. 327-330). IEEE.