UNIVERSITY
OF TURKU

# Design and Implementation of Indoor Disinfection Robot System

Robotics and Autonomous Systems
Master's Degree Programme in Information and Communication Technology
Department of Computing, Faculty of Technology
Master of Science in Technology Thesis

Author:Qiang Li

Supervisors: Tomi Westerlund ; Zhuo Zou

May 2023

**Master of Science in Technology Thesis**
**Department of Computing, Faculty of Technology**
**University of Turku**

**Subject**: Robotics and Autonomous Systems
**Programme:** Master’s Degree Programme in Information and Communication Technology
**Author**: Qiang Li
**Title**: Design and Implementation of Indoor Disinfection Robot System
**Number of pages**: 88 pages
**Date**: May 2023

**Abstract**.

After the outbreak of COVID-19 virus, disinfection has become one of the important means of epidemic prevention. Traditional manual disinfection can easily cause cross infection problems. Using robots to complete disinfection work can reduce people's social contact and block the spread of viruses. This thesis implements an engineering prototype of a indoor disinfection robot from the perspective of product development, with the amin of using robots to replace manual disinfection operations.

The thesis uses disinfection module, control module and navigation module to compose the hardware of the robot. The disinfection module uses ultrasonic atomizers, UV-C ultraviolet disinfection lamps, and air purifiers to disinfect and disinfect the ground and air respectively. The control module is responsible for the movement and obstacle avoidance of the robot. The navigation module uses Raspberry Pi and LiDAR to achieve real-time robot positioning and two-dimensional plane mapping.

In terms of robot software,we have done the following work: (1) Based on the ROS framework, we have implemented functions such as SLAM mapping, location positioning, and odometer data calibration.(2) Customize communication protocols to manage peripheral devices such as UV-C lights, ultrasonic atomizers, air purifiers, and motors on the control board. (3) Develop an Android mobile app that utilizes ROSBridge's lightweight communication architecture to achieve cross platform data exchange between mobile devices and navigation boards, as well as network connectivity and interaction between mobile phones and robots

Finally, this thesis implements an engineering prototype of a household disinfection robot from the perspective of product development.

**Keywords**: disinfection robot; Simultaneous Localization And Mapping; robot operating system

## Table of contents

# Chapter 1  Introduction

## 1.1. Background

We have entered the era of robots. The rapid development of robotics makes robots everywhere in our lives. It plays an essential role in medical, transportation, education, industrial manufacturing, and other industries. With the development of Internet technology and artificial intelligence technology, robots have become more and more intelligent. It allows robots to help humans complete some tasks in many fields. Especially in the medical area, robots can help medical staff to serve patients better. Nursing robots are improving patient care during isolation. Robots can clean and prepare patient rooms independently, limiting person-to-person contact in infectious disease wards [1].

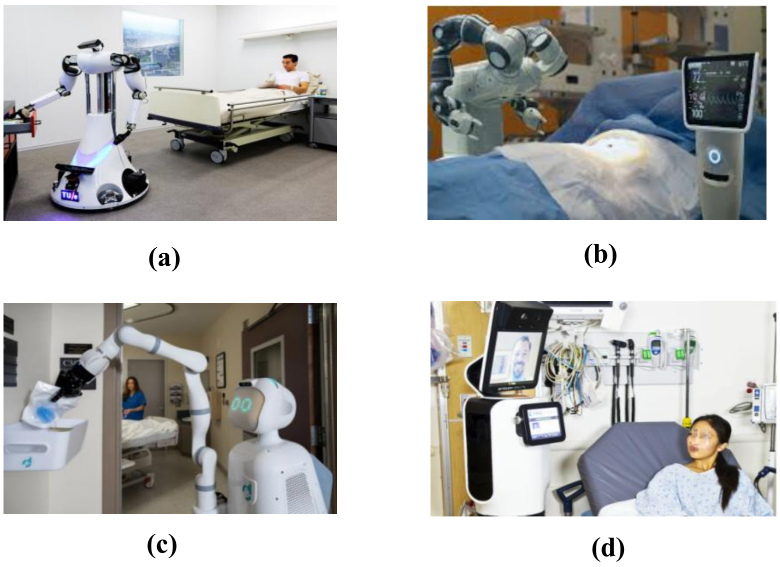


Figure 1- 1  Application of robots in the medical field [1]

Artificial intelligence makes robots recognize medicines, reducing the time it takes for medical staff to distribute medicines ( Figure 1-1 (a) ). With improvements in robot technology, the research conducted on the interaction between humans and robots has increased [2]. Human-robot interaction (HRI) and human-robot collaboration (HRC) are receiving more and more attention. A robotic arm with low latency and high precision allows doctors to do surgery for patients thousands of miles away. It means that when a patient needs help, remote robotics will enable experts to be there immediately, which solved the problem of the lack of senior medical and surgical experts in poor areas ( Figure 1-1 (b) & (c) ). Besides, the research of high-precision, low-cost lidar, and robot vision algorithms has brought new vitality to mobile robots. Localization of a mobile robot in an unknown environment while building a map simultaneously is known as Simultaneous Localization and Mapping (SLAM) [3]. In the 21st century, SLAM plays an essential role in the robotics industry as a core element that contributes to every modern robot, especially in the field of autonomous mobile robots and driverless vehicles [4]. Robots with SLAM functions can autonomously perform obstacle avoidance navigation and positioning like humans. It allows the robot to complete tasks such as voluntary garbage cleaning, crop irrigation, and community patrols without human intervention.

The COVID-19 has caused considerable losses to people's lives and property. Robots have played an irreplaceable role in this epidemic. A big challenge for COVID-19 is its high contagiousness and infection fatality rate. Robots provide a perfect solution to this situation since the virus can't replicate itself inside a robot, and the use of robots heavily reduces person-to-person contact [5]. The ongoing COVID-19 pandemic has already provided numerous robotics instances that are used to assist agencies with detecting and caring for infected populations and preventing further transmission [7]. Robots also enable industry, educational institutions, and individuals to cope with the unique consequences of sheltering-in-place restrictions and workplace absenteeism [8]. Under such circumstances, robotics, artificial intelligence, and human-computer interaction have received more and more attention. They play an essential role in helping hospitals, airports, transportation systems, scenic spots, hotels, restaurants, and communities to resist the spread of the COVID-19. Managers use humanoid robots, self-driving cars, drones, and other intelligent robots to avoid direct contact between people and reduce the risk of potential infection in various ways. These measures include transporting supplies, cleaning and disinfecting public spaces, measuring or checking body temperature, providing safety guarantees, comforting and entertaining patients, etc. It is widely confirmed that diagnosis and screening critical for containing the spread of the pandemic [13]. Many researchers have developed different diagnostic techniques to increase the efficiency and accuracy of COVID-19 testing (Figure1-2(a)). Oropharyngeal (OP) swabbing is widely used for viral nucleic acid detection. A robotic sampling system, which can be remotely controlled, was developed. The robot has a force control mode during the sampling. It can reduce the sampling time by 50% while achieving a sampling success rate of 95% compared to manual sampling [16].

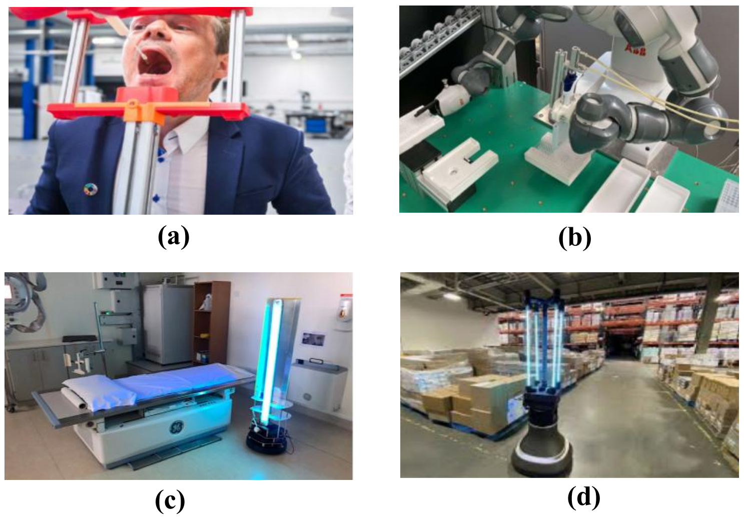


Figure 1- 2 Application of robots in the COVID-19. (a) Lifeline Robotics' throat swabbing robot collecting a sample [18]. (b) ABB's collaborative robot, YuMi [11], was used in an application designed at the Politecnico di Milano, in partnership with IEO to support hospitals in serological testing [12]. (c) Irish hospitals are testing this robot, developed by Akara Robotics, for coronavirus disinfection of radiology examination rooms [20].(d) MIT UV robot disinfecting food bank [17].

Since scientists have not yet developed a mature vaccine to deal with the COVID-19, we can only reduce the new crown's spread by blocking its transmission route. So, disinfection is required in hospitals, shopping malls, and homes. Different types of disinfection robots have been developed during this pandemic to fight against the novel coronavirus crisis. The MIT UV robot (Figure1-2(d)), which operates in food banks, consists of a commercial base from AVA robotics and a customized UV light controller. The robot includes four UV-C lamps, and it takes 30 minutes to cover 370 square meters with a nominal speed of 0.1m/s [19]. Similarly, Violet from Akara Robotics Ltd. is a disinfection robot that can sense people and turn off its transmitter when needed. The violet-The mobile platform called "turtlebot" is equipped with a UV-C lamp on the top. It occupies a small area and can be easily moved in a hospital ward's narrow corners with multiple medical equipment [20]. Compared with the traditional manual disinfection operation method, the disinfection robot has obvious advantages: the disinfection effect is better, the efficiency is higher, and time and labor costs are saved. The disinfection robot dramatically reduces the pressure on medical staff and effectively solves the problem of shortage of medical resources. At the same time, it also protects medical staff from virus infection. Home is a place where people always stay for a long time. It is necessary to sterilize it regularly. However, most disinfection robots on the market are large and expensive. Not suitable for household disinfection.

In view of the above situation, this thesis designs and implements a low-cost home disinfection robot prototype. The robot is designed with the idea of rapid development based on the platform so that it can be quickly applied to the prevention and treatment of COVID-19. The robot is equipped with a UV-C lamp, disinfectant, and air purifier device to disinfect the home environment. An Android APP is developed to control the robot.

## 1.2. Related Work

### 1.2.1. Disinfection Method

A common way to sterilize bacteria and disinfect freshwater is using ultraviolet (UV) technology. There are three types of UV light; they are UV-A (315 to 400nm), UV-B (280 to 315nm), and UV-C (100 to 280nm). UV-A and UV-B rays are the cause of human sunburn. However, ultraviolet-C rays are more intense and are usually blocked by the earth's ozone layer. The UV lamps used for disinfection are generally UV-C type; they can emit 254 nanometers of light at a speed of 20 joules per square meter per second (within a distance of 1 meter) and are guaranteed to kill 99.99% of bacteria[17].UV-C irradiation is very successful for virus inactivation and bacteria disinfection by reducing contamination on high-touch surfaces. It uses short-wavelength ultraviolet C (UV-C) light to kill or inactivate microorganisms by destroying nucleic acids and disrupting their DNA, leaving them unable to perform vital cellular functions [25]. Figure 1-3 represents UV radiation's function when using type C rays, which is very harmful to people. Its wavelength is very short, which makes it more effective in eradicating pathogens [26].

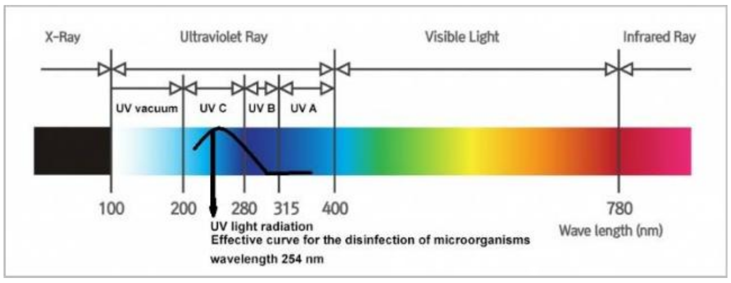
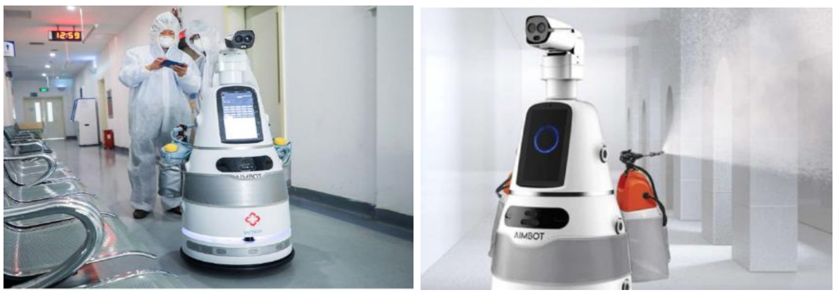


Figure 1- 3 Ultraviolet sterilization process [26]

Disinfectant is the most widely used method of disinfection in real life. Disinfectants are generally oxidizing or toxic. Some of these chemicals can break down the cell wall through enzymes' action to cause pathogens' death. Others can destroy the pathogen's DNA so that it can't carry out protein synthesis and thus can't replicate. Some disinfectants denature pathogen proteins and make them lose their destructive effects [27]. Because the disinfectant can diffuse into the air and the molecular heat movement allows the disinfectant to act on the entire disinfected area. So using this method can sterilize some inaccessible places. When the robot has the ability to atomize liquids, it can perform liquid-based disinfection tasks. Aimbot robot from UBTECH has also been adapted with liquid spray agents to disinfect hospitals and indoor areas, such as hospitals and healthcare rooms (Figure 1-4) [17]. In Dubai, drones are being used to disinfect the whole city [28]. In America, the company EagleHawk is performing a test for disinfecting large open areas, such as football fields [29]. The OMI Drone disinfection system disperses a mist of HEPA-approved agents that kills COVID-19 and 99.9% of bacteria within 60 seconds of contact [30].



(a) (b)

Figure 1- 4 A spray liquid disinfection robot [17] (a) Robot moving in hospital.(b) Robot spraying liquid agent.

Filtration uses a medium to trap solid substances in a suspended liquid or gas, while the liquid or gas passes through the medium to be clarified or purified. People have used filtration technology in production and life for thousands of years. The use of filter materials to filter out viruses and bacteria is a new approach that has emerged with the development of new materials in recent decades. It has also been a research hotspot in recent years. As people's awareness of environmental protection has increased and the requirements for quality of life have increased, purification materials for filtration and sterilization have also been rapidly developed and widely used [32]. The principle of filtration is mechanical sieve filtration and adsorption retention. Because the filter element's pore size can be made small (nano-level), and the diameter of bacterial pathogens is usually between 0.1 micrometers and several micrometers. Therefore, most pathogens with larger diameters can be removed by using a filter element. With the advancement of science and technology, people have improved traditional filter materials by adding antibacterial agents, developed new production processes, and created new antibacterial filter materials that can filter bacteria and have antibacterial effects, effectively resisting microorganisms.

### 1.2.2. Disinfection Robot

At present, the domestic disinfection robot market is still immature. Only a few companies on the market develop and sell disinfection robot products, most of which are commercial disinfection robots. The main application scenarios are large conference rooms, comprehensive shopping malls, schools, restaurants, and other public places. The main disinfection methods are ultraviolet radiation and spraying disinfectant.

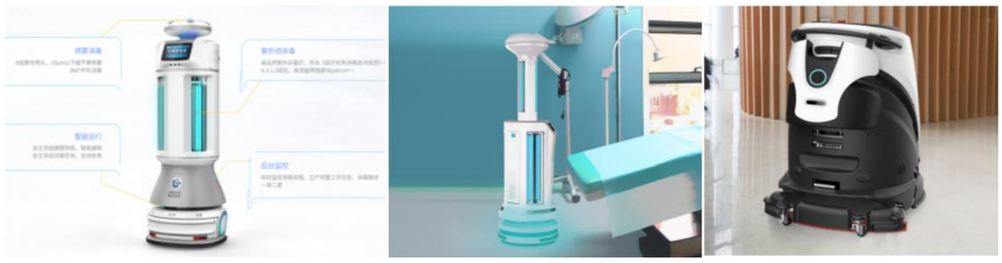

(a) (b) (c)

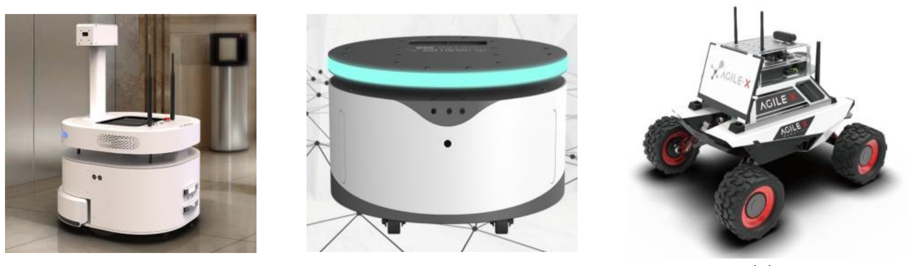
Figure 1-5 Disinfection robot (a) Qinglang [33] (b) TMiRobot [34] (c)Ecobot [35]

The Qinglang in Figure 1-5(a) is an unmanned disinfection robot released by Qinglang company. The product uses two types of sensors, lidar and vision, to collect surrounding environmental data as the primary input data of the SLAM algorithm. Due to the use of a variety of sensor data fusion navigation solutions, this product is an excellent product in mapping and positioning [33]. Figure 1-5 (b) TMiRobot is a robot specially used in the operating room. TMiRobot uses three disinfection and sterilization methods: ultraviolet light, ultra-dry mist hydrogen peroxide, and air filtration. Since most operating room patients are seriously ill patients, it is easy to cause infection and other problems if they are not disinfected and sterilized.

TMiRobot can strengthen the standardized management of disinfection, automatically calculate the disinfection time according to the space area, and automatically perform the 360° angle disinfection around the disinfection target without dead ends [34]. Figure 1-5 (c) Ecobot Scrubber is an intelligent scrubber device released by Gaussian Robotic, which integrates space disinfection, scrubbing and drying, dust pushing, sterilization, and ash reduction. Ecobot efficiently integrates the ultrasonic disinfection liquid physical and chemical spray system with the wet mopping floor cleaning function to create medical-grade environmental hygiene and safety for hotels, shopping malls, office buildings, hospitals, factory logistics workshops, and other scenes [35].

### 1.2.3. Robot Hardware Platform

The robot is a mechanical device that works automatically. It is mainly composed of a robotic body, memory or program functions, and core parts. It can accept human commands, run pre-arranged programs, or act according to principles and programs formulated with artificial intelligence technology. Suppose you want to develop a marketable product in the robotics field. In that case, you need to build a complete set of systems that can realize your ideas: including hardware devices, control systems, interfaces, and testing tools that allow the robot to run and serve as a test platform. It directly leads to a very high degree of difficulty in robot development. In order to reduce the difficulty of robot development, more and more hardware companies like to make their products into a universal platform like a software framework and provide them to other companies for secondary development.



**(a)** **(b)** **(c)**

Figure 1- 6 (a) SLAMTEC Athena [36](b) Water [37] (c) SCOUT MINI [38]

There are many robot development platforms on the market. These robot platforms all provide their hardware and software SDK for developers to use. Robot platforms of different companies have different characteristics. Figure 1-6(a) is SLAMTEC's latest general-purpose mobile robot development platform Athena. Athena universal robot development platform has built-in high-performance autonomous positioning and navigation system kit SLAM Cube, which has perfect positioning and navigation and path planning capabilities. Its body is small, and it has strong adaptability to compact environments. It can be equipped with different applications to meet the secondary development needs [35]. Figure 1-6 (b) Water is an indoor mobile universal robot chassis released by Yunji Technology. The robot chassis can pass unimpeded on smooth stone ground, floors, and carpets. Water has super obstacle jumping ability and obstacle avoidance ability. It is stable, safe, and widely used [37]. Figure1-6(c) is an educational robot platform SCOUT MINI manufactured by Songling Robot (Shenzhen) Co., Ltd. Based on the Songling Robot ROS ecosystem, this package integrates high-performance industrial control, high-precision LiDAR, and multiple sensors. It can realize applications such as mobile robot motion control, communication, navigation, and mapping [38]. SCOUT MINI has a lightweight and portable industrial design with a sense of science and technology. It provides us with an excellent experimental platform for rapid ROS secondary development in multiple directions such as education and scientific research, product pre-research, product demonstration, etc.

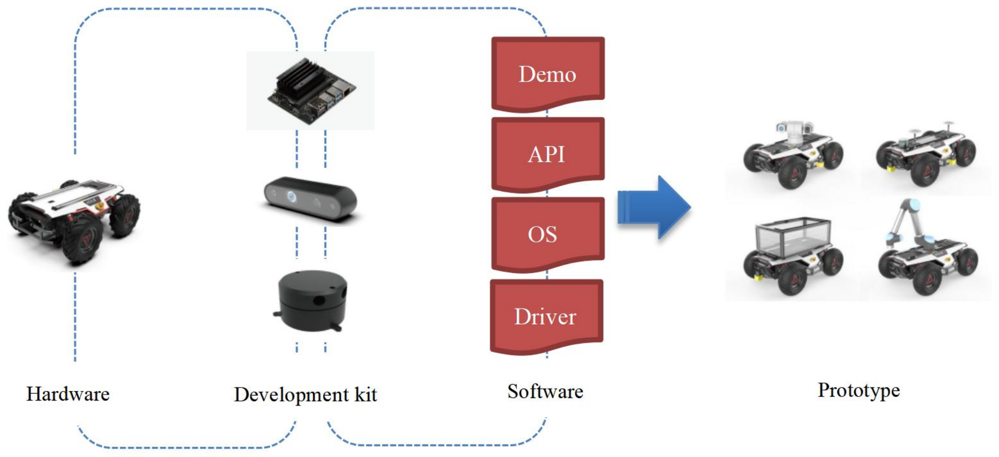


Figure 1-7 Process of the platform based on the secondary

Secondary development of the robot platform has high cohesion and low coupling characteristics. High cohesion means that each module in the robot platform concentrates on completing one specific task. Low coupling refers to the low degree of mutual influence between various modules. When a module is broken, it will not affect the regular operation of other modules. The secondary development based on the robot platform is like Lego blocks. The modules with different functions can be combined according to their own needs to complete the robot's overall development. Simultaneously, the robot platform usually provides a software framework or a demo version of the application program to help engineers quickly build robot prototypes in the early stage. Figure 1-7 shows the process of secondary development based on the platform. The chassis hardware platform is SCOUT2.0 of Songling Robot Co., Ltd. The company provides a wealth of development kits. The picture shows the system board, lidar, and RGBD camera module provided by Songling. Besides, the company's software framework and demo program can quickly drive the above modules to form an organic system to realize the robot prototypes.

### 1.2.4. Robot Software Framework

One of the main challenges when developing a complex robotic system is designing and implementing a software architecture, which is essential for the interaction and coordination of hardware and control modules. Previous robots were simple and had few functions. There are not many modules that need to be controlled internally. Therefore, the previous robots did not use the operating system but directly used the loop cycle to control each module. As robots become more and more complex, the use of loop control methods becomes impractical. On the one hand, the robot needs to perform multiple tasks simultaneously, which requires the robot to have multi-threading capabilities. On the other hand, without the support of an operating system, there is no uniform standard for the robot software system. Many code blocks with the same function cannot be reused, and the portability of the code is insufficient. So, it is necessary to have a software middleware that can abstract the robot's complex hardware (such as actuators and sensors) and provide a simple, standardized API to control the system. It should be modular, easy to use, robust, reliable, easy to maintain, efficient, and flexible, and should support multi-threading.

People often like to call the robot framework an operating system. Strictly speaking, the robot framework is not an operating system. The operating system is a system that manages hardware and software resources. It provides a calling interface for upper-level application software to implement operations on hardware resources. ROS is a middleware running based on the operating system. It provides the services we expect from the operating system, including hardware abstraction, low-level device control, standard functions implementation, message-passing between processes, and function package management. It also provides tools and libraries for acquiring, building, writing, and running code between multiple computers. In other words, this type of framework includes a hardware abstraction similar to an operating system, but it is not a traditional operating system. It has features that can be used for heterogeneous hardware. Also, it is a robot software platform that provides a variety of development environments dedicated to developing applications for robots. There are many advantages to using the development platform: (1) Improve the program's reusability. Focus on the part you want to develop, and you can download the relevant function package to use the remaining functions. (2) Provide an efficient communication mechanism. There is no need to build a communication mechanism by yourself. (3) Provide development tools. Provide debugging-related tools-2D drawing and 3D visualization tools. So, you don't need to prepare the development tools required for robot development by yourself. You can use it directly. (4) Active developer community. So far, the relatively closed robotics circles and robotics industry have moved towards attaching importance to cooperation because of the robotic platform's advantages.

Software frameworks in the robotics field include OROCOS, XBotCore, OpenRT-M, etc. They all have their own characteristics. Among them, ROS is the most used one in robot development. It is easy to use and powerful. It is especially suitable for complex scenarios with multiple nodes and tasks. Therefore, since the birth of ROS, it has been popular in academia and industry. It has been widely used in robots, such as robotic arms, mobile chassis, unmanned aerial vehicles, and unmanned aerial vehicles. ROS is a framework suitable for robot programming. The framework couples the initially loose parts together and provides a communication framework for them. Figure 1-8 shows the development of an IoT robot that collects data using the ROS framework. Because the modules between ROS can carry out distributed communication, it is very convenient to deploy ROS on the entire system and uses RosBridge to realize data transmission.

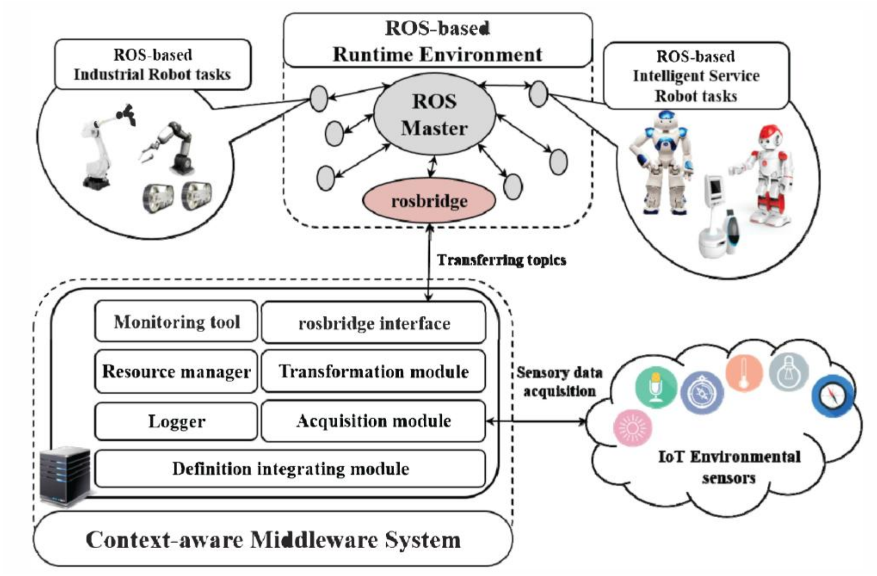

ROS-based Runtime Environment
ROS-based Industrial Robot tasks
ROS-based Intelligent Service Robot tasks
ROS Master
rosbridge
Transferring topics
Monitoring tool
rosbridge interface
Resource manager
Transformation module
Logger
Acquisition module
Definition integrating module
Sensory data acquisition
IoT Environmental sensors
Context-aware Middleware System

Figure 1- 8 A system architecture for providing robot tasks with sensory data acquisition [45]

## 1.3. Thesis Work and Contributions

According to the previous background introduction, under the influence of COVID-19, people urgently need the help of robots to fight the virus. Before developing a vaccine, blocking the route of transmission of the virus was the most effective method. Therefore, disinfection is one of the current effective ways for COVID-19 prevention. Although some robot products on the market can achieve disinfection functions, their size and operating noise are relatively large. Moreover, the price is high, and the power consumption is high. The main application scenarios of these robots are prominent public places. However, disinfection is also required in the home environment, which is the area with the longest human activities. However, there are few low-power and cheap disinfection robots that serve the home environment on the market.

Because of the above situation, this thesis uses the idea of platform-based development and refers to the existing robot chassis for development and design. We used the three disinfection methods introduced in the background: UV-C, disinfectant, and air purification, to realize a prototype of a household disinfection robot. This robot platform is jointly developed with my classmate Zhuoran Chao. He is responsible for designing the control board and developing the embedded software system [52]. This thesis mainly focuses on designing the robot's mechanical structure, the selection and integration of Commercial-Off-The-Shelf (COTS) components, navigation board's development, and the SLAM AND algorithm.

The main work and contributions of this thesis include:

**Design and implement a disinfection robot hardware for rapid prototype:** the hardware is mainly divided into the lower computer robot chassis and the upper computer SLAM navigation board. The upper computer uses Raspberry Pi as the navigation board. The lower computer sensors include the ultrasonic ranging module, infrared ranging module, and IMU inertial navigation module. The data collected by these sensors are sent to the upper computer through the serial port of the microprocessor on the bottom control board. The lower computer directly controls the UV-C lamp, Ultrasonic atomization, and air filter disinfection modules.

**Design and implement disinfection robot software system:** deploy ROS robot development platform. Write the corresponding launch file and Node file. In ROS, the code of each module is written as a Node file using python. Each Node only completes a specific function. And use the topic communication method that comes with ROS to realize the communication between each Node. Develop APP for Android client: Use Android APP as the primary means of human-computer interaction. Write the client socket and use the local area network to realize the communication connection between the APP client and the robot. Write a login Activity and main Activity for users to operate the disinfection robot.

**Realize and evaluate the SLAM function of the disinfection robot:** learn the principles of SLAM. Use open-source SLAM algorithms (for example, hector, gmapping, cartographer) in ROS to write corresponding data processing Node to form map data and coordinate data. This thesis tests the power consumption using different SLAM algorithms. The SLAM algorithm requires plenty of computing power resources. Different SLAM algorithms use different mathematical principles, resulting in various calculations. This thesis uses four algorithms to perform SLAM operations separately in the same environment. Determining the four algorithms' hardware consumption rate and power consumption to select the most suitable SLAM algorithm for the disinfection robot.

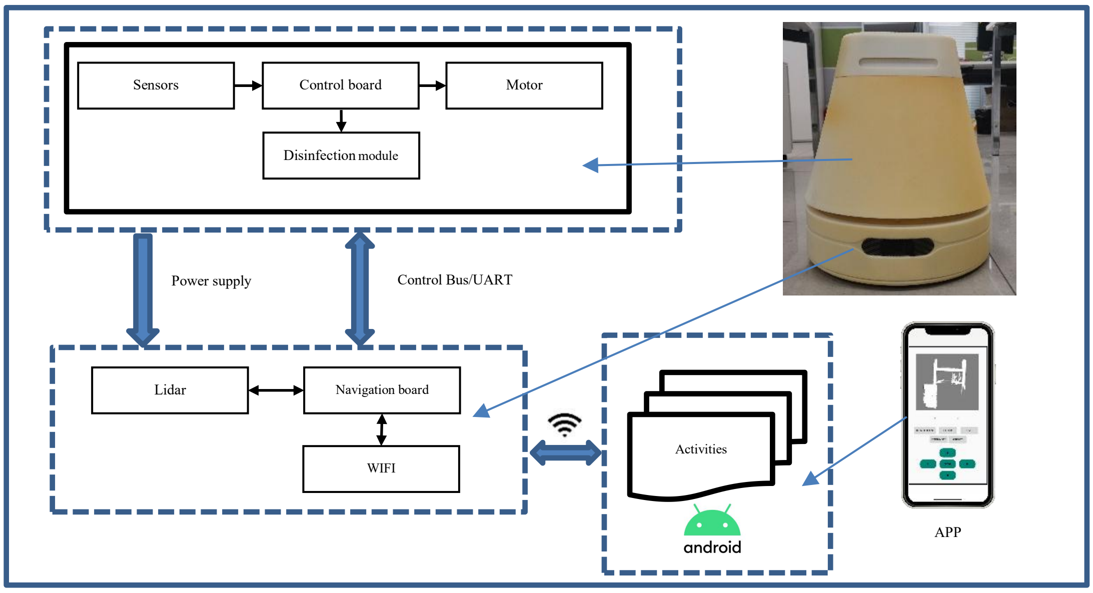


Figure 1-9 Illustration of Disinfection Robot System Architecture.

Finally, an indoor disinfection robot prototype was successfully built through the above work. Figure 1-9 shows the architecture of the entire disinfection robot. The prototype robot has three mainstream sterilization methods on the market to solve the problem of household sterilization.

## 1.4. Thesis Structure

Chapter 1 is the introduction that introduces the widespread application of robots in daily human life. It describes the application of robots in the medical industry under the influence of the COVID-19 virus and presents the research background and significance of disinfection robots.

Chapter 2 is system design. It analyzes the disinfection robot at the system level according to market demand. It formulates system development principles and sets various indicators of the robot. Finally, it introduces the overall architecture of the disinfection robot.

Chapter 3 is hardware prototyping. It mainly introduces the construction of the robot hardware platform. The motion model of the robot is proposed to determine the driving mode of the disinfection robot. It determines the disinfection method and selects the appropriate disinfection module to complete the entire hardware platform's construction.

Chapter 4 is the software system of the disinfection robot. It discusses the overall software design and introduces some essential nodes in the disinfection robot. It also introduces the communication methods between the disinfection robot boards and between the robot and the client. Finally, this chapter refers to the Android SDK provided by SLAMTEC and develops a simple APP to realize remote control in the same LAN environment.

Chapter 5 is the test and results. It tests the essential functions of the disinfection robot. It introduces the SLAM theory and introduces some mainstream SLAM open source algorithms. The hardware resource utilization of these algorithms was compared, which provides a basis for selecting the SLAM algorithm of the disinfection robot.

Chapter 6 is the conclusion and outlook. Mainly analyze the shortcomings of the robot. Put forward the measures that should be improved in the future according to the deficiencies.

# Chapter 2 System Design

This chapter first analyzes functional requirements for existing disinfection robots based on market demand, then formulates system design principles and system design goals according to market demand. Finally, this chapter proposes the system architecture of the disinfection robot.

## 2.1. Function Requirement Analysis

**Movable:** as a home edition disinfection robot, it must first have a move function. It requires the robot to have fundamental obstacle avoidance capabilities. In order to prevent multiple disinfection of the same area in the same disinfection task, the disinfection robot should have the ability to distinguish between disinfected and undisinfected areas.

**Size:** the size of the robot cannot be too large. The area of the home is not as large as in public places. Too much volume will occupy a larger space in the home. Some houses are tiny in size and cannot fit large-sized robots. The larger the size of the robot, the more inconvenient it is to move. Besides, it is difficult for a large-sized robot to enter a small space for disinfection.

**Disinfection method:** the disinfection method of the disinfection robot must be safe and effective. Commonly used disinfection methods include spraying disinfectant, ultraviolet sterilization, and air filtration. The disinfection robot should have at least the above-mentioned essential disinfection functions.

**Interaction:** a robot must include some human-computer interaction interfaces. There should be buttons on the robot that can operate the robot's functions for the users. There must be a corresponding display module for the robot's operating status to display its operating information. Because the robot is movable, we must have a remote control method to facilitate the user's human-computer interaction.

Network communication: in the era of such advanced communication, robots should also have the ability to communicate with the outside world. If the robot can access the

Internet, it can provide users with real-time dynamic information and more decadent operation options.

Development time: since COVID-19 has already caused a severe impact. The demand for disinfection robots is imminent. So, the robot development time is limited; the sooner, the better.

## 2.2. System Design Principles

### 2.2.1. Platform-Based Development Methodology

The development of electronic science and technology has brought various electronic devices into people's daily life. Electronic products can meet people's daily needs. In today's increasingly fierce competition in the data communication market, the dynamic market needs to shorten the product life cycle and increase product categories. Customers pay attention to delivery time, quality, and brand and pay attention to product prices. The development time and development cost of robot products are essential factors for the success of electronic products. Robot products need to go through the concept, planning, development, and verification stages from design to mass production [40].

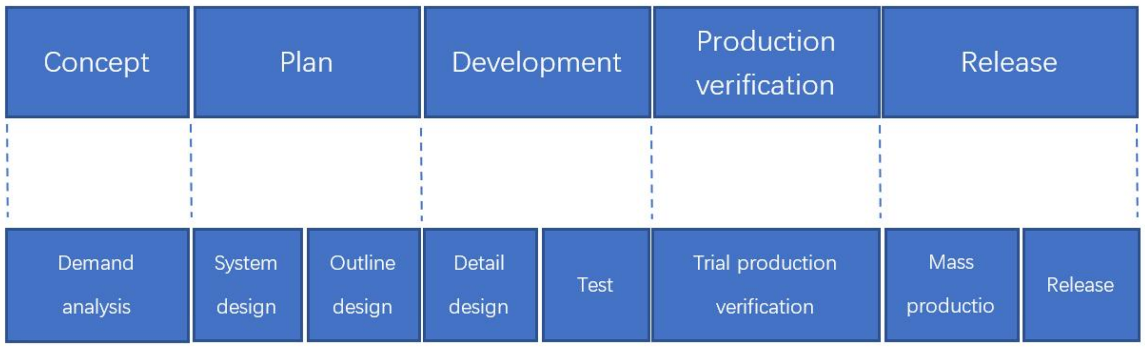


Figure 2- 1 IPD stage division

The electronic products' development stage is divided into several stages: demand analysis, system design, shape design, detailed design, testing, verification, mass production, and release(Figure 2-1). The most significant impact on the product development cycle and product cost is demand analysis, contour design, and detailed design in these stages. Suppose we want to develop a marketable product in the field of robotics. In that case, we need to build a complete system to realize your ideas: including hardware devices, control systems, interfaces, and allow the robot to run and act as a testing tool for the robot test platform. It directly leads to a high degree of difficulty in robot development. In order to reduce the difficulty of robot development, more and more hardware companies like to make their products into general-purpose platforms such as software frameworks and provide them to other companies for secondary development. Since COVID-19 is still spreading, we don't have enough time to develop a robot from scratch. It will not only bring high development costs but also lead to too long development time.

In view of the above situation, the current mainstream development method changes to a platform-based development method. For example， the designs of the single powertrains can feature certain commonalities among each other (e.g., they might all use the same power electronics unit). Such commonalities potentially reduce the development, production, and supply chain costs of the shared components [40]. The platform-based development method is similar to the framework in software development. A software framework is a semi-finished product. It usually refers to a software component specification to achieve a particular industry-standard or complete a specific essential task. It also refers to a software product that provides basic functions required by the specification to perform a particular software component specification. Developers do not need to build their own hardware from the lowest hardware circuit level when they can use a platform. A hardware platform that meets the product's requirements can be constructed quickly by analyzing its functional requirements and selecting the corresponding hardware development platform when designing the requirements. At the same time, the platform is generally equipped with a corresponding software framework. Engineers no longer need to write the underlying driver code by themselves. The platform provides some basic functions, which significantly improve the developer's research and development efficiency. Also, due to the use of platform development, the platform generally has passed many security verifications. The platform itself has reached the security standards of many countries. It provides convenience for the company's safety certification, which accelerates the product launch.

Thus, this thesis designs the hardware circuit by referring to the existing robot products on the market. The software uses the ROS robot development framework to realize the functions of the robot quickly. Shorten the overall development time of the robot.

### 2.2.2. Design Consideration

Through the functional analysis of the disinfection robot and market demand, the disinfection robot developed needs to comply with the following principles:

**Safe and reliable:** a robot is a movable object. It must be safe enough. It means that corresponding measures should be taken, or safety-certified devices should be used in the design where potential safety hazards are likely to occur. For example, the robot speed cannot be set too high. High-speed robots are more prone to collision accidents due to their heavyweight.

**Simple for using:** A good product must have a clear and straightforward operation logic. Because the robot is a complex system, there are many configurable parameters and functions. We should design a simple and easy-to-use user operation interface for users from the perspective of the customer.

**Low cost:** The disinfection robot is a home edition consumer product. The price cannot be too high. It requires a compromise between performance and price. We should select appropriate devices to reduce their production costs.

**Fast realization:** affected by the COVID-19, the development time cost must be shortened. Using the idea of rapid development based on the platform requires the use of existing results. The current hardware platform and software demo of the platform are reused in the development of the prototype. We need to put the focus of development on the debugging of function realization.

### 2.2.3. Design Specifications

The design specifications of the disinfection robot are shown in Table 2-1. The external structure of the robot is a cylindrical shape. The diameter of the bottom surface is about 37cm. This size is similar to the sweeping robots on the market. The size is appropriate, allowing the robot to move freely in complex home environments. The height of the robot is about 45cm. This size is chosen because if the robot's height is too high, the robot's center of gravity will be unstable. Easy to roll over when moving. Moreover, the size of some furniture is not large. Too high will hinder the movement of the robot in the home.

The moving speed of the robot is designed to be 0.25m/s. The robot at this speed is safe. When the robot hits an obstacle at this speed, neither the robot nor the hit object will be damaged. In addition, it takes a specific time for the UV-C lamp to sterilize, which requires the robot not to move too fast.

The external interface of the robot must be reserved. For users, they need to control and set the robot through buttons and software interfaces. Also, in the development stage, in order to facilitate debugging, we also reserve a UART interface for our debugging.

The working time of the robot determines how many areas the robot can disinfect at one time. The general housing area is 100 square meters. At a speed of 0.25m/s, it takes at least 2 hours to disinfect all rooms. Therefore, the continuous working time of the robot must be at least 2 hours.

**Table 2-1** Design goals of the disinfection robot system

| Item | index | parameter |
|---|---|---|
| Size and quality | The maximum diameter | <370 mm |
| | Height | <450 mm |
| | Weight | <5kg |
| Sports performance | Walking speed | 0.25m/s |
| | Maximum walking speed | 0.35m/s |
| Interface | Hardware interface | UART/keyboard |
| | Software interface | Android SDK |
| Power consumption | Continuous working time | >2h |
| and power supply | Power supply | 14.6V batteries |
| Working environment | Temperature | -10℃ ~ 40℃ |
| requirements | Humidity | 30% ~ 70% |

The working environment of the robot is indoor. The temperature is generally -10°C to 40°C, the typical household temperature. The humidity is about 30% to 70%. It needs to be considered when choosing the electronic components of the robot.

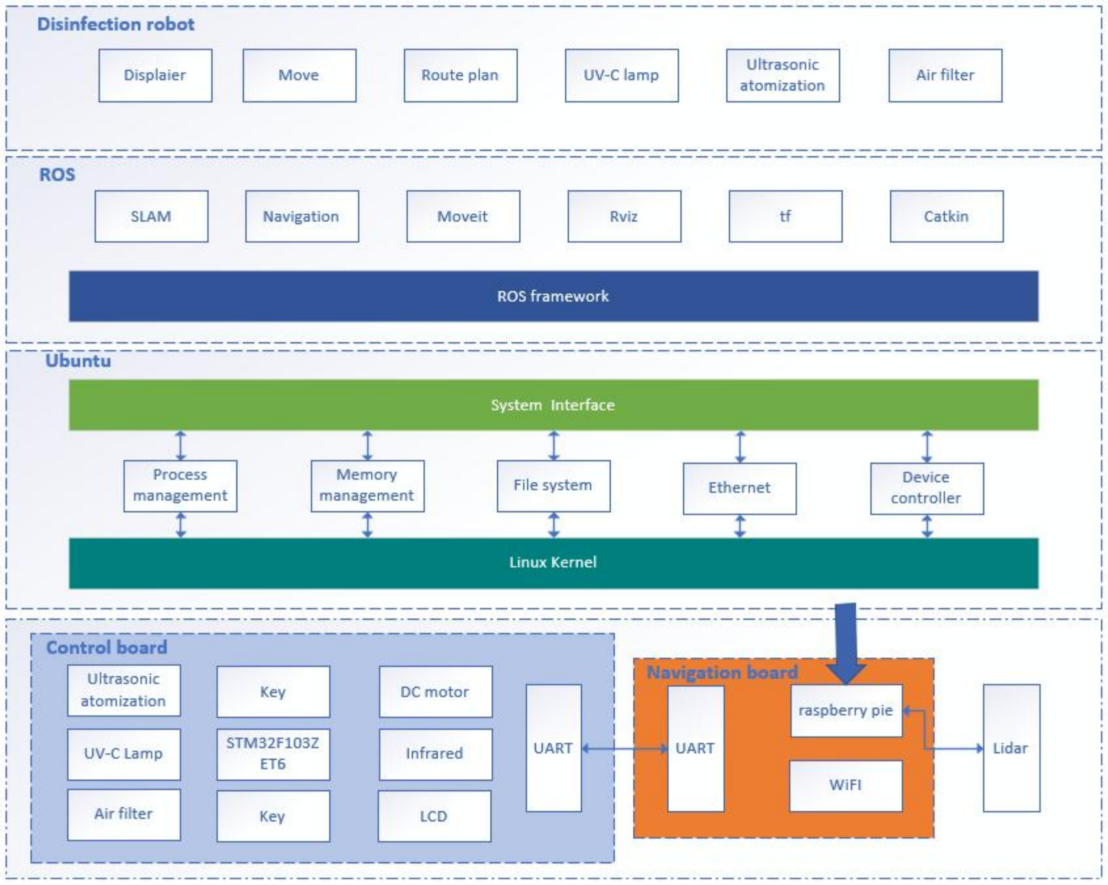


Figure 2- 2 System architecture of disinfection

## 2.3. System Architecture

The overall system architecture of the disinfection robot is shown in Figure 1-8. The robot hardware consists of three parts: the control board, the navigation board, and the radar module. The software system is divided into two parts: embedded software and upper computer software. The embedded software system runs on the control board. The Ubuntu system runs on the navigation board. Ubuntu is a released version of the Linux operating system. ROS needs the Linux kernel to provide a running environment. ROS uses the navigation board's hardware resources as the middleware in the software system by calling various operating systems' interfaces. Finally, the robot functions are developed using the ROS framework's function library to realize different disinfection robot functions.

**Control board:** The control board is installed at the bottom of the robot. It is an embedded development board. Its primary function is to collect data from the chassis sensors. The data is sent to the navigation board in the format of Control_Bus through the serial port of MCU. The control board also needs to output a PWM signal through the IO port to control the DC motor's speed and steering. The board has a power management circuit. The input DC voltage is stepped down to supply power to each IC chip on the board. The disinfection module is also directly controlled by the MCU.

**Navigation board:** the navigation board is Raspberry Pi. The reason for using the Raspberry Pi is that the SLAM algorithm requires powerful processing capabilities. The underlying control board does not have enough hardware resources to support the SLAM algorithm. Besides, considering that the robot needs Ethernet communication, multi-threading, and other requirements. So, we must use the operating system to complete the tasks of the disinfection robot. Raspberry Pi is a desktop-level development board. It can run various distributions of Linux operating systems smoothly.

**Lidar:** lidar is the primary sensor for disinfection robots to perceive the outside world. It is installed at the center of the chassis 15cm from the ground. It must be placed horizontally. Otherwise, the map created will have significant errors. The principle of lidar ranging is complicated, and a specific algorithm must process its raw data before it can be used. Generally speaking, the lidar on the market has its own data processing chip, and the measured data will be sent to the navigation board in the form of a serial port. The robot plans to use the mature laser radar module as the primary sensor of the robot.

**Operation system:** the operating system provides the software foundation for the entire robot. Using the operating system can save us from writing the Raspberry Pi hardware program driver. At the same time, the operating system provides us with multi-threaded task processing functions. We can quickly implement serial communication, WiFi communication, and other functions with the operating system's help. ROS can call the operating system's interface to provide more convenient functions for upper-level applications based on the operating system.

**ROS:** ROS is a framework suitable for robot programming. This framework couples the initially loose parts together and provides them with a communication framework.

Although ROS is called an operating system, it is not an operating system in the usual sense like Windows and macOS. It just connects the operating system and the ROS application you develop. It can be regarded as middleware. In the ROS environment, the robot's perception, decision-making, and control algorithms can be better organized and run. In the disinfection robot, ROS provides us with modules such as SLAM, moveit, tf, catkin, and Rviz. Significantly reduce the time and difficulty of robot development.

**Disinfection robot:** we use the catkin compilation tool to write the function nodes of the disinfection robot. For example, proxy server node, SLAM node, command input node, etc. Choose the appropriate communication mechanism to realize the communication between these nodes. When each node completes its own task, the entire robot can operate normally to achieve the disinfection function.

## 2.4. Summary

This chapter mainly analyzes the market demand for robots. According to the market's needs, robots' requirements are formulated in terms of size, disinfection methods, safety, interactivity, and development time. The design specifications of the disinfection robot are also described in detail in Table 2-1. Finally, this chapter designed the disinfection robot's overall architecture and introduced each part of the architecture's functions.

# Chapter 3 Hardware Prototyping

This chapter mainly introduces the construction of the robot hardware platform. Due to the limitation of development time, this thesis uses platform-based development ideas. We refer to SLAMTEC's Slamware mini solution to design and implement the disinfection robot hardware platform's construction. This chapter first introduces the overall hardware architecture of the disinfection robot. My classmate Zhuoran Chao designed the embedded control board. The detailed circuits and PCB design can refer to his thesis [52]. In this thesis, we focus on components selection and integration of electrical and mechanical systems. Then we add a disinfection module to our own bottom control board to form the central part of the robot. The upper computer uses the Raspberry Pi as the navigation board to realize positioning and mapping functions. The two boards use serial ports for data transmission. All the above hardware platforms are integrated to form a disinfection robot prototype.

## 3.1. System-Level Hardware Architecture

Using slamware mini directly as the robot chassis can speed up the development progress. But its price is too high. The cost is too high for a consumer version of the home robot. If the platform is used for mass production in the future, consumers will not accept it. Therefore, this thesis uses the slamware mini platform as a reference for secondary development. In the early stages, plastic structures were made by hand molding. The driving wheels are placed on the left and right sides of the disinfection robot chassis to realize the robot's movement function. The steering function is realized by the difference in the speed of the left and right wheels. For the control board's hardware circuit, we refer to the open-source PCB board of slamware mini and design the control board for the disinfection robot by ourselves. The control board is directly connected with IMU, infrared sensor, ultrasonic sensor to collect surrounding environment data. The control board is also directly connected to the three disinfection modules to control their working status. The disinfection robot disinfects the ground through the UVC ultraviolet lamp installed on the chassis. Use an ultrasonic atomizing sheet to atomize the disinfectant into the air. The centrifugal fan draws air into the internal space, which can filter out PM2.5

and VOC harmful gases to achieve the purpose of air disinfection. Slamware mini uses a core board called slamecore as the host computer. We analyze the slamcore's primary function. It collects and processes lidar data and realizes the SLAM function. The core board is also responsible for communicating with the outside world through the wireless network. Therefore, a desktop-level embedded development board needs to be used instead. This thesis uses Raspberry Pi instead of slamcore. Figure 3-1 shows the system-level structure of the disinfection robot.

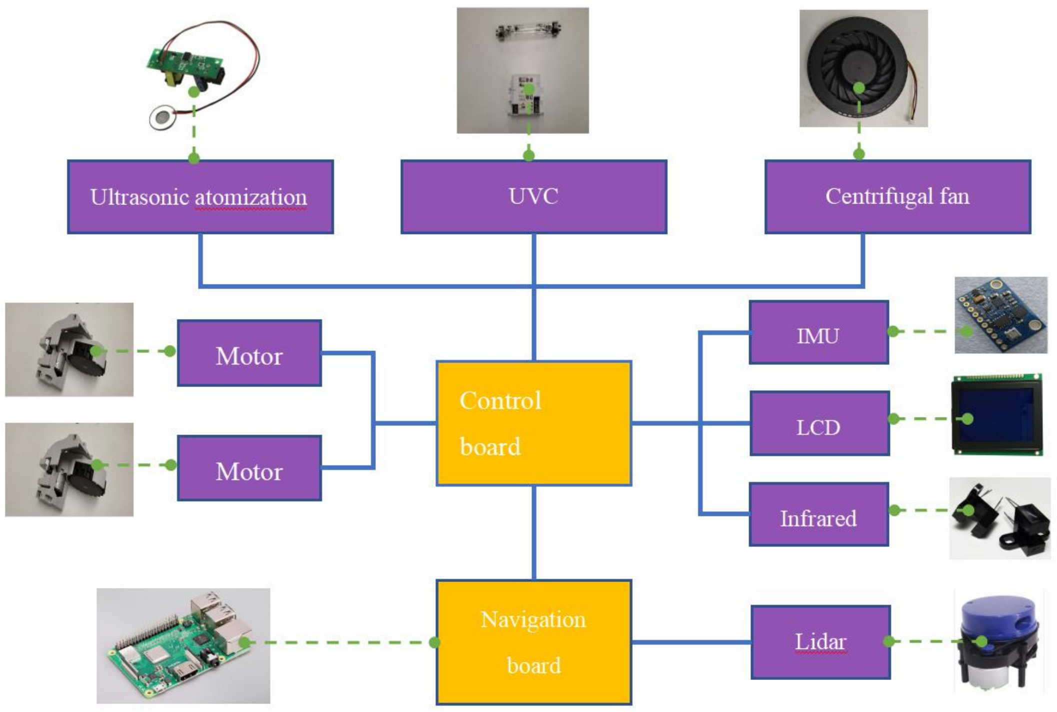


Figure 3- 1 Hardware architecture of disinfection robot

## 3.2. Robot Driving Type

Different robot chassis have different driving methods. It can be roughly divided into three types: front-wheel steering + rear-wheel drive, two-wheel drive + universal wheel, and four-wheel drive. Each of these three methods has its own characteristics. The following is a brief analysis of these three driving methods. We need to choose a suitable driving method to meet our needs.

### Front-wheel steering + rear-wheel drive

The front-wheel steering + rear-wheel-drive wheeled robot chassis mainly uses electric cylinders, worm gears, and other forms to achieve front-wheel steering. The motor of the rear wheel with a reducer provides power to the robot. This driving method has the advantages of low cost and simple control. But the turning radius is large, and the use is relatively inflexible. Figure 3-2 shows this drive model.

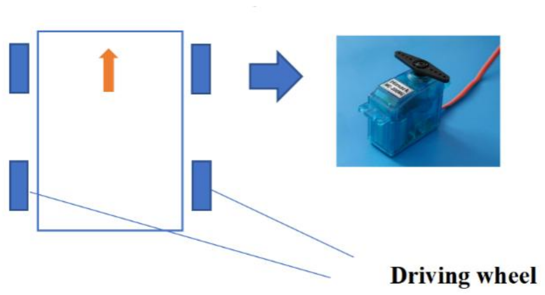


Figure 3- 2 Front wheel steering + rear wheel drive

### Two-wheel drive + universal wheel

Figure 3-3 shows the two-wheel-drive + universal wheel drive mode. In this way, the universal wheels and driving wheels can be arranged in different forms according to the robot's requirements for the design center of gravity and turning radius. Its advantages are that the structure and motor control are relatively simple, the robot is more flexible, and the control algorithm is simple.

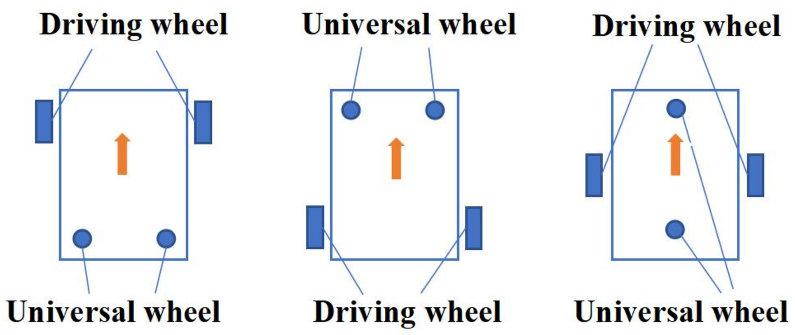


Figure 3- 3 Two-wheel drive + universal wheel

### Four-wheel drive

Figure 3-4 shows the four-wheel-drive model. The chassis driven in this way has a stronger ability to walk in a straight line, and the driving force is also comparative.

However, the cost is too high, and the motor control is more complicated. In order to prevent the robot from slipping, a more exemplary structural design is required.

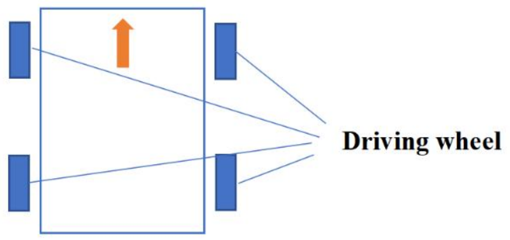


Figure 3-4 Four-wheel drive

The design load of the disinfection robot in this thesis is about 6kg. Under this load condition, a strong driving capability is not required. We pay more attention to the steering flexibility of the robot and the complexity of the control algorithm. Based on the above considerations, we choose two-wheel drive + universal wheel drive mode in Figure 3-3. This method can well meet our steering flexibility requirements. Since the height of the disinfection robot is about 45 cm, the use of two universal wheels can ensure the disinfection robot's stability during walking.

## 3.3. Robot Mathematical Motion Model

Since the robot generally moves on the ground, this thesis discusses the robot's motion model on a two-dimensional plane. Figure 3-5 is a schematic diagram of a two-dimensional planar robot motion model. $O_r$ is the local coordinate system of the robot itself. Ow is the world coordinate system. The angle $\theta$ is the angle between the X-axis of the robot itself and the X-axis of the world coordinate system, called the direction angle. The size of the angle indicates the direction of the robot at this time.

$$X_k = \begin{bmatrix} x_k \\ y_k \\ \theta_k \end{bmatrix} \tag{3-1}$$

The pose description of the entire robot in the world coordinate system is represented by formula 3-1. $X_k$ is the pose at the current moment k. Its two-dimensional coordinate in the world coordinate system is $\theta_k$ . $(x_{k,}y_k)$ is the direction angle of the current robot movement.

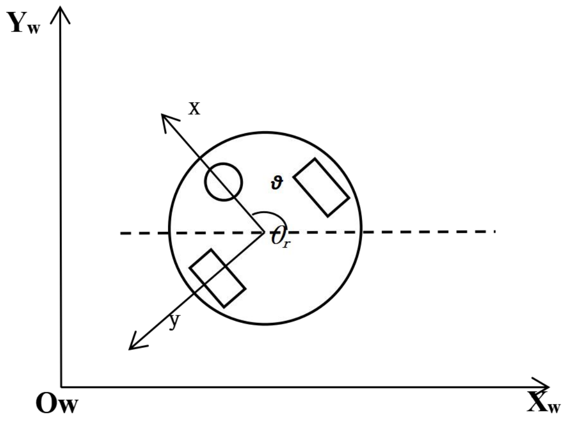


Figure 3- 5 Figure 3-5 two-dimensional planar robot motion model

## 3.4. Control Board and Peripheral Modules

The whole robot hardware contains many modules. This thesis mainly focuses on the selection of components and the design of the overall system architecture.

### 3.4.1. Control Board

Figure 3-6 is the PCB diagram of the control board. The Control board uses STM32F103 as the microprocessor. Since the disinfection robot does not need to run an operating system, it does not require high processor performance. Therefore, the control board uses STM32F103 as a microprocessor. Above the control board are the infrared sensor and ultrasonic sensor interfaces. These two interfaces are directly connected to the microprocessor IO port to collect environmental data for the robot. On the upper right is a WIFI module. The module model is ESP8266. This module is commonly used as a wireless transmission module for crop networking. This module is installed on our control board for pre-debugging to communicate with the site wirelessly without going through the Raspberry Pi. Simultaneously, the design also provides a data transmission alternative for the subsequent iteration of the product.

Starting from the right are a series of slots for peripheral devices. Air filter fan, UV-C lamp, and DC motor modules require additional drive circuits. A bridge rectifier circuit is installed on the control board to provide the motor's large current required. The UV-C lamp needs a rectifier to provide the turn-on voltage. The battery directly powers the rectifier. The air filter fan's control port is connected to the microprocessor, and the power management chip directly provides the power port. At the bottom on the right is the level detection module slot of the disinfectant. This module uses the IIC bus to communicate with the microprocessor. Next to it is an LCD interface. This interface uses the UART on the microprocessor for communication. The button board interface is reserved under the control board, convenient for the product to lead out all the buttons later. At the bottom left of the control board is a DC-DC power management circuit. This part of the circuit converts the 14.8v input voltage from the battery into 3.3V and 5V for the microprocessor and other modules. The control board's left side is an Ethernet interface directly connected to the Slamcore interface in the control board center. The purpose of this is to be compatible with the upper computer development board Slamcore. It is convenient for us to debug the robot. There is also a lidar interface on the left, which is also directly connected to the slamcore so that the slamcore development board can directly read the lidar data. The bottom left of the control board is the battery slot. It is the power supply for the entire robot.

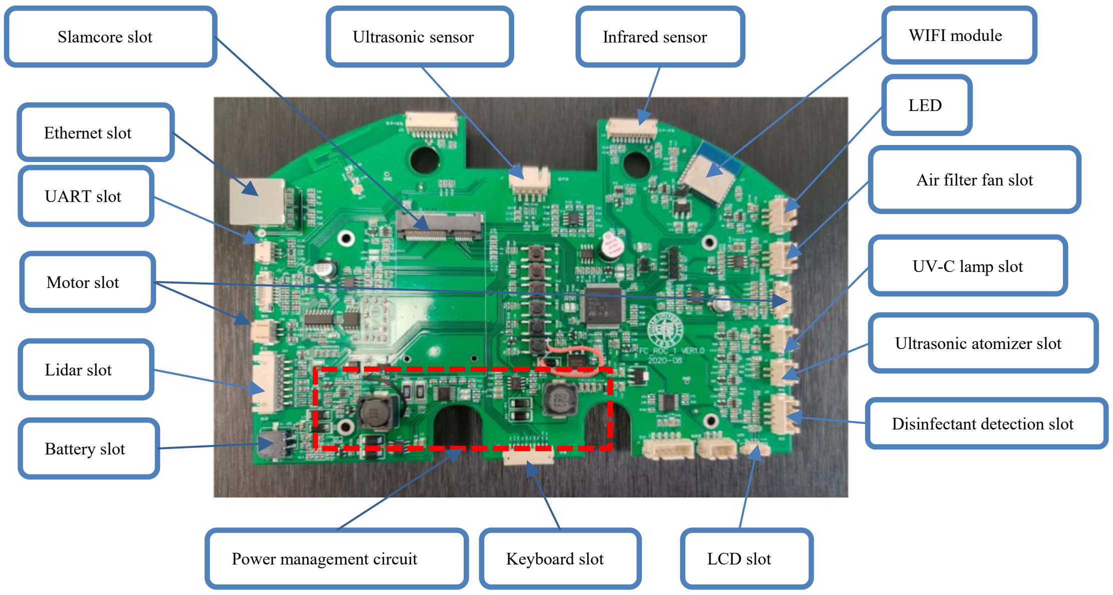
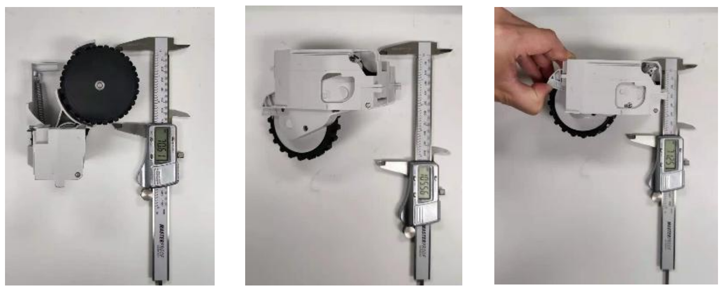


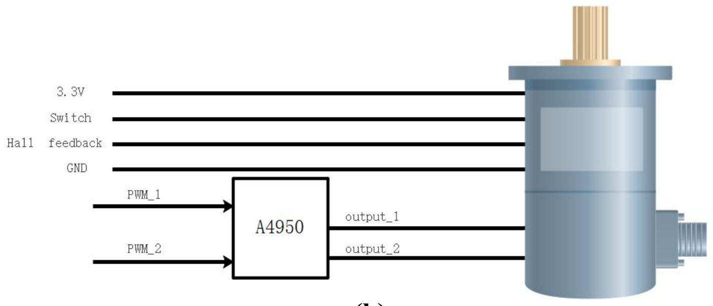
Figure 3- 6 Figure 3-6 PCB board of the control board

## 3.4.2. DC Motor

The movement of the robot needs to be driven by a motor. There are many types of motors available on the market: DC motors, brushless motors, stepper motors, etc. The weight of the robot is about 6kg. We don't need a high speed. However, a relatively large torque is required. And the motor control algorithm should be as simple as possible. It is most appropriate to consider the use of a DC motor. The general DC motor cannot provide enough torque to drive the robot to move. A better solution is to use a DC motor plus a reduction gear to increase the motor's torque. Figure 3-7(a) shows the robot's driving motor module, a DC brushless motor module provided by the Zhaowei company. The rated working power of the motor module is 12V, 0.5A. The rated power without load is 6w. The motor provides corresponding torque through the reduction gearbox to drive the movement of the entire robot. A Hall element is installed on the motor to detect the current motor speed. The Hall element's output is directly connected to the IO port of the STM32F103 microprocessor, and the motor speed data is obtained by counting pulses.



**(a)**



**(b)**

Figure 3- 7 (a)DC motor module (b) Electrical connection of the motor module

The chip of the motor drive is A4950. There is an H-bridge drive circuit inside the chip. The chip's power supply is converted into a 12V supply by the battery through DC-DC direct current power conversion. The motor speed control input is two-channel PWM signals. Two PWM signals can control the motor speed to realize walking, braking, reversing, and steering. The motor module contains a suspension switch, which can be used to detect terrain. The corresponding electrical signal is output through the Hall chip in the module. The electrical connection of the entire module is shown in Figure 3-7(b).

### 3.4.3. Ultrasonic Atomization Module

The ultrasonic atomizer converts energy between electrical energy and mechanical energy. Electrical signals cause objects to produce mechanical movement. The mechanical vibration of the object produces vibration waves of different frequencies. The ultrasonic frequency is 20-100 kHz. The ultrasonic atomizer uses ultrasonic waves' high-frequency vibration to generate tiny droplets on the liquid's surface to atomize the liquid.The atomized droplets can be diffused into the air through the micropores on the vibrator plate.

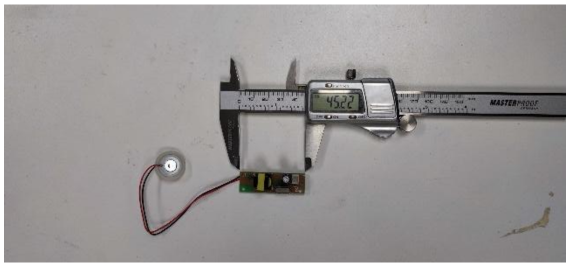
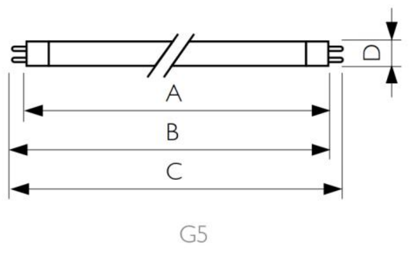


Figure 3- 8 Ultrasonic atomization module

Figure 3-8 shows the ultrasonic atomization module used in this disinfection robot. The module is a 5V power supply. A power supply interface is provided on the module. The module's powering will form a positive feedback resonant circuit through the module's capacitance and resistance to make the circuit generate 100KHz vibration. The electric signal is transmitted to the vibrator to cause it to vibrate. Thousands of micropores with a diameter of 3um-6um are distributed on the vibrator plate. Contact the non-porous side with the water surface or a wet cotton swab to produce dry mist pthesiss. Dry mist pthesiss can diffuse freely into the air. Using a DC power supply to measure its power is about 0.8W, and the working current is about 150mA. Therefore, the module can be directly connected to the IO port of the microprocessor STM32F103 without additional drive circuits.

### 3.4.4. UVC Ultraviolet Lamp

Ultraviolet rays (UV) are invisible to the naked eyes. Ultraviolet rays can be subdivided into three categories, namely UVA, UVB, and UVC. UVC can destroy the DNA of viruses, bacteria, and spores, making microorganisms non-pathogenic. Ultraviolet radiation can be applied to water and air purification treatment. In the disinfection process, it can inhibit the activities of microorganisms without using chemicals. Radiation can work on all microorganisms, including bacteria, viruses, fungi, and protozoa (such as Cryptosporidium and Giardia). This method will not produce harmful by-products to health after disinfection. It also has the advantages of low cost, simple operation, convenient maintenance; no excessive problems will occur.



**(a)**

**(b)**

Figure 3- 9 UVC tube (a) Tube size (b) Tube and adapter

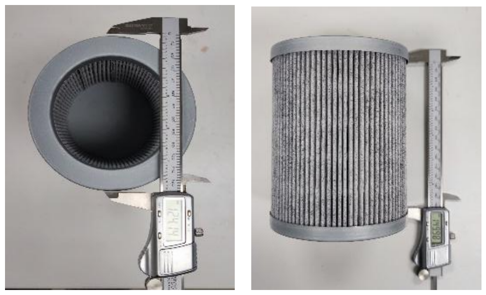
The disinfection robot in this thesis uses the UVC ultraviolet lamp produced by Philips. The specific parameters of the lamp can be found in Table 3-1. The tube can emit ultraviolet shortwave (UVC) radiation with a peak value of 253.7 nanometers. Considering that the irradiation area is only 30cm$^2$, the irradiation power of the 4W ultraviolet lamp is enough to kill the bacteria on the ground. Besides, 4W is a good low-power option for disinfection robots. Figure 3-9(a) shows the size and physical object of the UVC lamp tube installed at the bottom of the disinfection robot. The size of the lamp tube is (A): 136mm; (B): 140mm; (C): 150mm; (D) 16mm. The lamp is small in size and can be easily installed on the bottom of the robot's chassis. Generally speaking, UV lamps require a relatively high starting voltage. The power supply of the entire robot is a 16.8V lithium battery. The voltage of the battery cannot start the lamp. An adapter is needed to provide a starting voltage when the lamp is starting. This voltage is sufficient to break down the mercury vapor in the ultraviolet lamp tube, causing it to ionize and ignite the UVC lamp to enter the working state. The white module in Figure 3-9(b) is the power adapter for the lamp. The adapter only needs 12V input voltage, and 0.8A input current can start the UVC lamp normally.

**Table 3-1** Philips TUV4W lamp parameters [43]

| Item | Specification/Condition |
|---|---|
| Lamp holder | G5 |
| Power (W) | 4 |
| Lamp voltage (V) | 29 |
| UVC100(W) | 0.9 |
| Lamp current (A) | 0.17 |
| Lumen decay rate (%) | 20 |

### 3.4.5. Air Filter

The air purification module comprises two parts: HEPA (High-efficiency particulate air filter) and a centrifugal fan. The HEPA standard filter has an effective rate of 99.998%


(a)

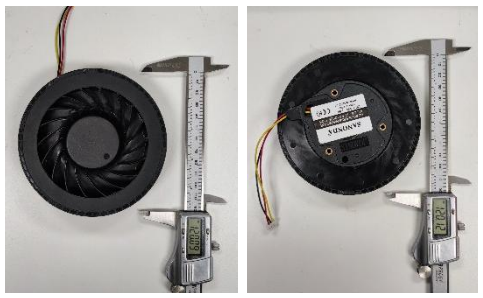

(b)

Figure 3- 10 (a) HEAP filter (b) Centrifugal fan

for 0.1 microns and 0.3 microns. The HEPA net characteristic is that the air can pass, but the fine pthesiss cannot pass. Its removal efficiency of pthesiss below 0.3 microns (1/200 of the hair diameter) can reach 99.7%. The filter is the most effective filter medium for smoke, dust, bacteria, and other pollutants. It is the most efficient filter material recognized internationally. It has been widely used in high-clean places such as operating rooms, animal laboratories, crystal experiments, and aviation [42].

Figure 3-10 (a) shows the physical picture of the filter inside the disinfection robot. In order to adapt to the size of the centrifugal fan, the disinfection robot in this thesis uses the HEAP filter customized by Guangdong Haibei Purification Technology Co., Ltd. The height of the filter is 15cm, the outer diameter is 12cm. A plastic cover seals the bottom. It is ensured that the air is sucked into the filter screen by the fan and filtered through the cylindrical filter element, and then blown out. The filter absorbs the impurities in the air to purify the air.

**Table 3-2** Centrifugal fan specifications [44]

| Item | Specification/Condition |
|---|---|
| Mode1 No. | KFFB12025H12B |
| Operating Voltage | DC 7V ~ 13.2 V |
| Rated .voltage | DC12V |
| Starting Voltage | DC ≤7V on/off |
| Rated current | 1.3A(Max:1.5A) |
| Power consumption | 15.6W(Max:18W) |
| Speed | 3200±10%RPM |

The centrifugal fan use KFFB12025H12B model produced by Guangzhou Guangfengtongda Electronics Co., Ltd. Figure 3-10(b) is a physical object, and its specific parameters are given in Table 3-2. The three wires from the module terminals are red: VCC; black: GND; yellow: PWM. The rated working voltage of the centrifugal fan is 12V. It can be powered by the DC-DC output on the bottom control board. PWM controls its speed, and the signal is directly given by the IO port of the STM32F103 microprocessor.

One of the more critical indicators of centrifugal fans is their PQ curve. In the same fan with different airflow, measuring its pressure value can draw a PQ curve, and this curve can be used to describe the fan's characteristics. It is called the fan characteristic curve [32]. Figure 3-11 is the PQ curve of the fan this time. Because the filter screen will produce bigger resistance to the air. If the pressure generated by the fan is insufficient, the air cannot pass through the filter, and the purpose of purifying the air cannot be achieved. Sufficient wind pressure must be generated to allow air to pass through a filter with a significant drag coefficient. From Figure 3-11, it can be seen that the centrifugal fan generates a large static pressure under the premise of ensuring the air volume, and at the same time, the noise is low. The fan meets the needs of home edition air purifiers.

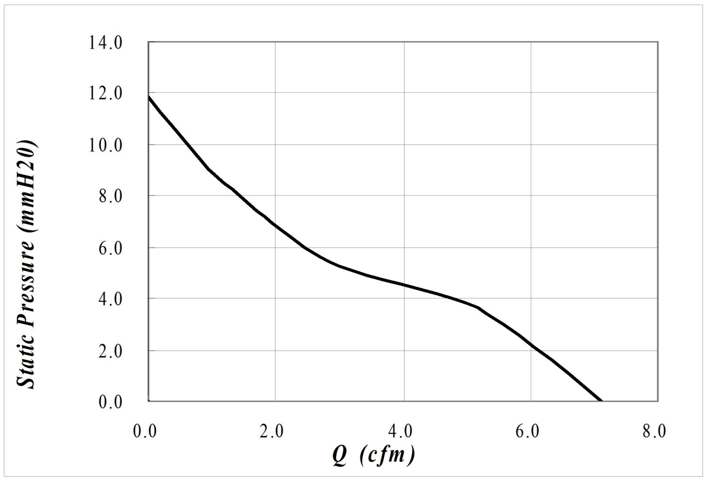


Figure 3- 11 PQ curve of centrifugal fan [44]

## 3.5. Navigation Board

The prototype of the disinfection robot needs to use the ROS development platform under the Ubuntu system. Ubuntu is a computer operating system distribution based on the Linux operating system. Running Ubuntu requires a more powerful hardware platform, and computing SLAM algorithms require higher processor performance for computer hardware. In addition, the disinfection robot needs to be operated on the Internet, and the core board should be a development board with a wireless communication function. In order to shorten the development cycle of the prototype, the disinfection robot uses Raspberry Pi 3B+ as the navigation board. Raspberry Pi is a development kit commonly used by electronics hobbyists. It is only the size of a credit card but basically has all the functions of a computer. With abundant hardware interfaces, it can install any ARM platform operating system.

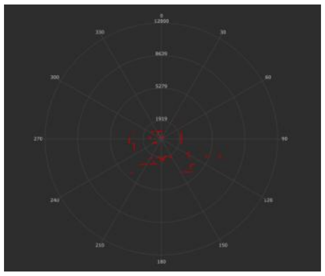
Selecting Raspberry Pi 3B+ as navigation bord due to it has a 100M wired Ethernet interface. In the pre-debugging stage, the computer and the navigation board can be communicated with each other conveniently via a wired method. More importantly, the robot is a freely moving object. In order to control the robot on the mobile phone, the robot must be able to communicate wirelessly. WiFi is the most common communication method at home. Raspberry Pi 3B+ contains an 802.11 b/g/n wireless network module. It is convenient to send commands to the robot through WiFi.

## 3.6. Lidar System

Lidar is essentially a ranging sensor. The medium used for ranging is light, and light travels in a straight line in the air. Therefore, the distance measured each time is the distance between a tiny target and the lidar. Each small target can be regarded as a point. Points can be used to form lines and surfaces that represent environmental information. The primary input in laser SLAM is point cloud data from lidar. Figure 3-12(a) shows the point cloud image output by the 2D lidar. The red dot in the figure is the distance between the target object and the lidar itself. Lidar itself has a positive direction. And all points can be expressed in the polar coordinate system while using coordinates.

According to the scanning dimension, lidar can be divided into 2D and 3D. According to the ranging principle, lidar can be divided into TOF and Triangular ranging. Among them, 2D lidar is also called single-line lidar. As the name implies, it can only transmit and receive one laser beam at the same time. In order to scan the surrounding environment of the robot itself in 360 degrees, a DC motor will be used on the lidar to drive the transmitter to rotate to achieve two-dimensional planar point scanning. This type of lidar is generally used in two-dimensional scenes. Figure 3-12(b) shows the A1M8-R5 single-line lidar released by SLAMTEC, which is also the lidar used in this experimental platform. Its ranging principle uses the triangular ranging method. Corresponding to 2D lidar is 3D lidar which is also called multi-line lidar. This type of lidar can emit multiple rays at the same time. It can be considered that multiple single-line lidars are working at the same time to scan the space. Currently, on the market, there are mainly 4-line, 8-line, 16-line, 24-line, 36-line, 64-line, and 128-line multi-line lidars. The environmental data scanned by the multi-line can be used to restore the three-dimensional structure of the space.

The main performance indicators of lidar mainly include measuring distance, scanning angle, range resolution, angular resolution, sampling frequency, scanning frequency, anti-interference ability, etc. Since the disinfection robot's main SLAM requirements are navigation and establishing a 2D indoor plane map, there is no need to reconstruct the room's entire surrounding environment. So, the lidar used in this experimental platform is a 2D single-line lidar.


(a)

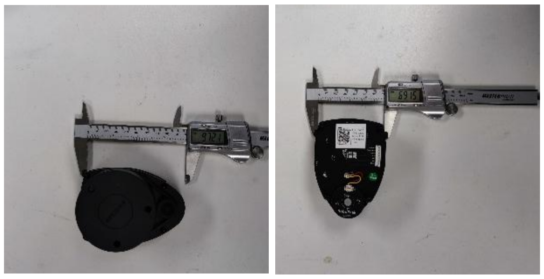

(b)

Figure 3- 12 (a) 2D lidar point cloud (b) A1M8-R5 single-line lidar

Considering that the primary working environment of the disinfection robot is indoors. The indoor area is generally not very large. There are no special requirements for the range and accuracy of the lidar. Integrating the disinfection robot's product positioning and the SLAM requirements, this experimental platform uses SLAMTEC's model A1M8-R5 lidar as SLAM's primary measurement sensor. A1M8-R5 is a low-cost two-dimensional lidar solution developed by SLAMTEC. It can achieve a 360-degree omnidirectional laser ranging scan within a 12-meter radius of a two-dimensional plane. Generate plane point cloud map information in space. This cloud maps information can be used in practical applications such as map surveying and mapping, robot positioning and navigation, and object/environment modeling. Under the condition that the sampling period is set to 1450 points sampling/week, the lidar scan frequency can reach 5.5hz, and the scan frequency can reach up to 10hz. A1M8-R5 adopts the low-cost laser triangulation ranging system developed by SLAMTEC, which performs well in various indoor environments and outdoor environments without sunlight.A1M8-R5 is mainly divided into the laser ranging core and the mechanical part that makes the laser ranging core rotate at high speed. After each subsystem is powered separately, the ranging core will start to rotate clockwise to scan. Users can obtain the scanning and ranging data of

A1M8-R5 through the communication interface (serial port/USB, etc.) of A1M8-R5. A1M8-R5 adopts laser triangulation ranging technology and cooperates with the high-speed visual acquisition and processing mechanism developed by SLAMTEC, which can perform distance measurement up to 8000 times per second. During each ranging process, the A1M8-R5 will emit a modulated infrared laser signal. The A1M8-R5's vision acquisition system will accept the laser signal's reflection after it hits the target object. After real-time calculation by the DSP processor embedded in the A1M8-R5, the distance between the illuminated target object and the A1M8-R5 and the current angle data will be output from the serial port.

## 3.7. Integration of Disinfection Robot

The disinfection robot's appearance and internal structure were handed over to Shenzhen Siteou Technology Co., Ltd. for design and production. The disinfection robot contains many modules. How to reasonably place all modules in a limited space requires careful consideration. Figure 3-13 is a three-dimensional structure diagram of the disinfection robot made by solid work software. The figure 3-13 shows its internal structure.

The whole disinfection robot can be divided into three parts:

**Bottom part:** two DC motors are installed on the chassis as the disinfection robot's power source. The universal wheels are installed outside the bottom to ensure the smooth operation and steering of the robot. A 14.8V battery is placed on the bottom of the chassis as a power source. The battery capacity is 10.4. By measuring the power of the entire disinfection robot around 50W. So theoretically, the working time of the robot is 2.8 hours. Using this battery can ensure the robot work for 2.5 hours. A UV-C lamp is placed at the bottom of the chassis. His starter is directly connected to the palace port of the battery. The control terminal is connected to the control board. The control board is installed on the ground to facilitate the electrical connection between the control board and other functional modules. The lidar is placed above the control board, about 15 cm above the ground. The connection between the robot chassis and the upper part uses eight pillars to ensure that the laser beams are not blocked.

**Middle part:** this part is to install a large air filter element and a centrifugal fan. When the centrifugal fan is running, it will draw out the air in the filter element's middle. Reduce the pressure inside the filter element. Under atmospheric pressure, external air will be forced into the filter element to achieve the purpose of filtering air.

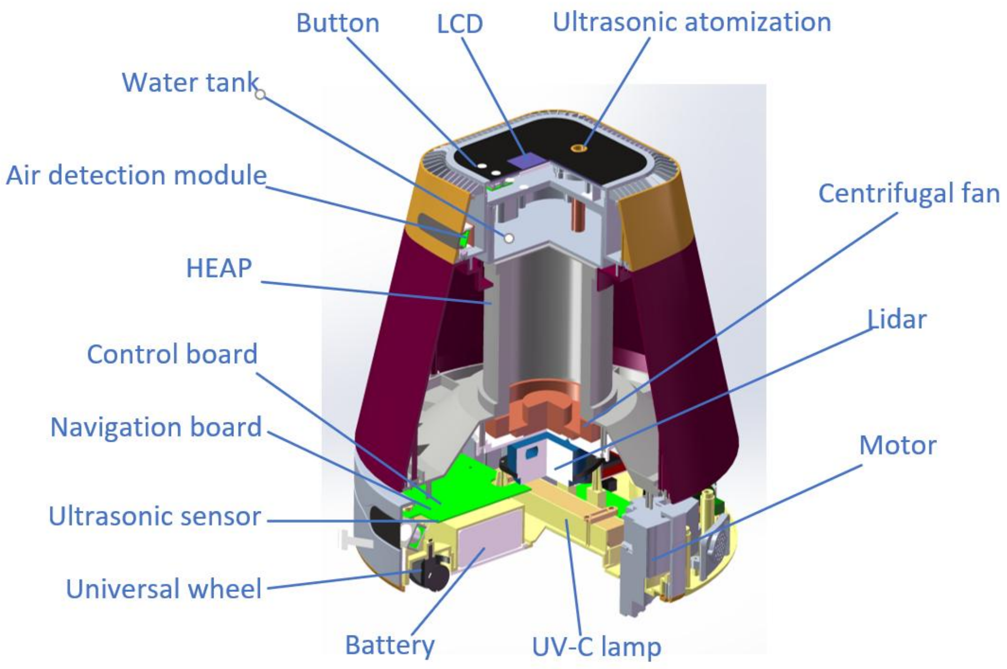


Figure 3- 13 Disinfection robot solidwork three-dimensional diagram

**Upper part:** this part contains the button module, LCD module, and ultrasonic atomization module. The buttons use reliable and straightforward mechanical buttons. The buttons' essential functions are to control the startup and shutdown of the robot and turn on and off the disinfection module. The display screen can display the current

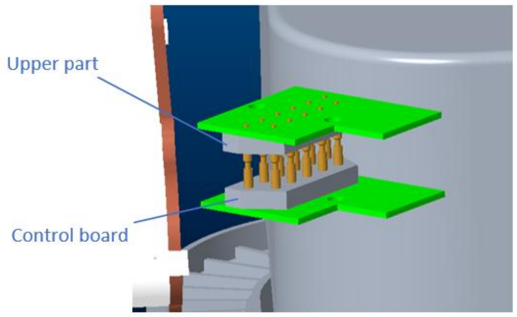


Figure 3- 14 Separate interface for upper and lower parts working information of the robot in real-time. The ultrasonic module is placed on the top. The disinfectant is delivered to the atomizing sheet through a cotton swab. The water tank containing the disinfectant is also placed in the upper part to facilitate timely cleaning and replacement of the water tank.

Each part needs a power supply and control. We lead the power supply port, serial port, IIC port, and button port from the bottom control board through external wires. Because the upper and middle parts of the robot are detachable. We need to adopt a separable contact method to solve the wire connection problem for the connection between the middle and upper parts. Figure 3-14 illustrates how the upper and lower parts of the robot are connected.

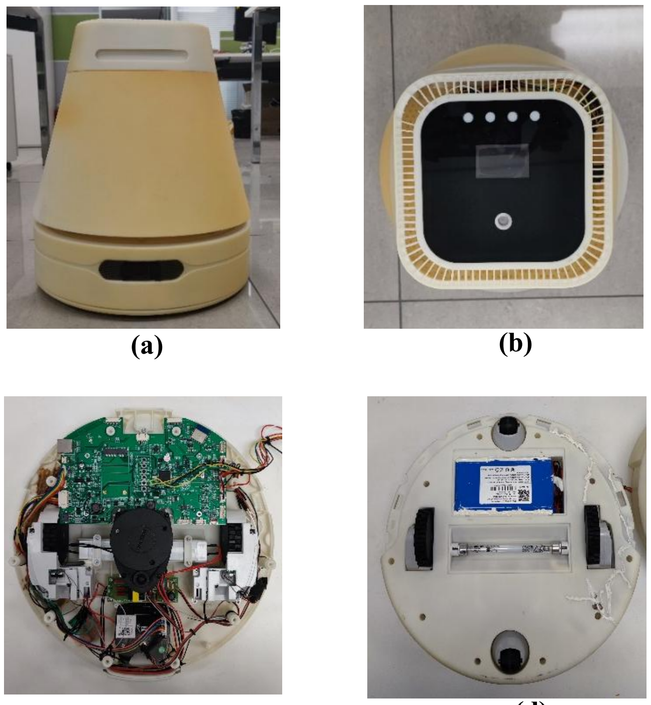


Figure 3- 15 Prototype of disinfection robot

Figure 3-15 shows the prototype of the disinfection robot. The prototype uses Raspberry Pi as the host computer. Can run Linux operating system. Under Linux operating system, ROS robot development platform can be quickly deployed to realize SLAM. As the primary data input of the SLAM sensor in the picture, the lidar is directly connected to the SLAM core board Raspberry Pi. It transmits the collected lidar data to the SLAM core board through the serial port. The core board is mainly responsible for processing lidar data and running SLAM algorithms to map the surrounding environment and calculate its own posture. Finally, the SLAM core board is connected to the control board. Use the custom Control_Bus communication protocol to send the motor control commands and other data to the control board through the serial port. The robot provides two control methods for the disinfection module. One is that the mobile phone sends instructions to the control board through the navigation board. The buttons directly control the other on the top of the robot.

## 3.8. Summary

This chapter mainly introduces the construction of the disinfection robot hardware platform. The robot is mainly divided into control board, navigation board, disinfection module, sensor module, and power supply part. This chapter analyzes the disinfection robot's functional requirements and selects the appropriate robot development platform and components. Design the structure with reference to the existing disinfection robots on the market. Finally, all the modules are combined to form a robot prototype.

# Chapter 4 Software System of Disinfection Robot

This chapter mainly introduces the disinfection robot software system. The software system is divided into three parts: control board embedded software system, navigation board ROS operating system, and Android App client. The main tasks are: (1) The host computer software on the Navigation board uses the ROS framework to complete the robot node functions' programming and the communication between the nodes. (2) Activity design of Android App client. The goal is that customers can connect to the robot chassis and perform some basic operations under the same local area network. (3) Design the communication protocol between the upper computer and the lower computer. Realize the communication between the host computer and Android. My classmate Zhuoran Chao completes the embedded software of the disinfection robot. This thesis will not introduce it in detail in this thesis. The detailed embedded part can refer to my classmate's thesis [52].

## 4.1. Software System Architecture

The overall structure of the disinfection machine software consists of three parts. The main tasks of the underlying embedded software are: control sensors to collect environmental data around the robot, send chassis information to the upper computer, and receive commands from the upper computer to execute the corresponding walking according to the command, switch the disinfection module, and respond to the robot buttons. The upper computer is a robot operating system composed of Ubuntu and ROS. The difference between ROS and operating system and their relationship has been explained in detail in Chapter 2. In ROS, the function of the robot is realized through the smallest unit of a node. We subdivide each function of the robot into a node. There are many nodes in the navigation board in Figure 4-1. These nodes each complete their own tasks and communicate through communication methods provided by ROS. Finally, it is necessary to use the rosbridge provided by ROS to realize the communication between ROS and other platforms. In this thesis, the data transmission between the robot and Android is realized through the API interface of rosbridge. The underlying transmission method between the two adopts the TCO/IP protocol and realizes the wireless data transmission between the robot and the mobile phone APP in the form of WIFI transmission.

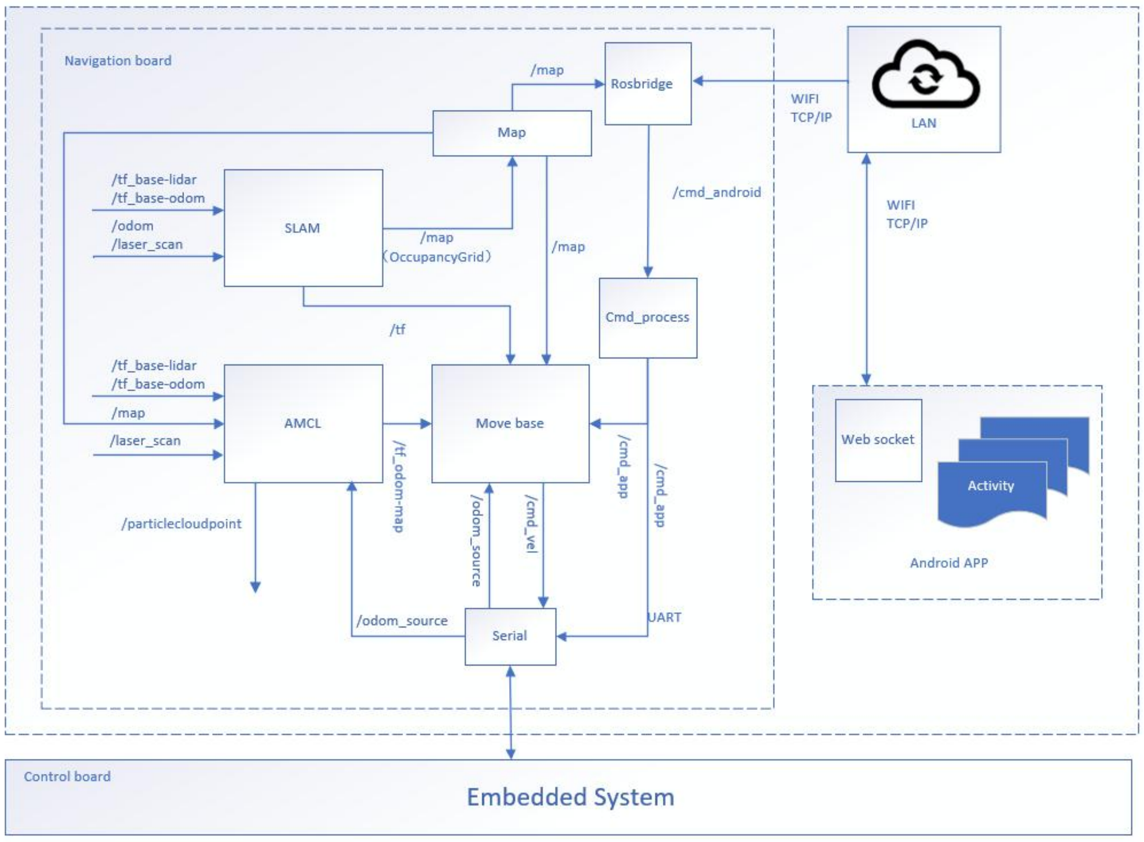


Figure 4-1 Schematic diagram of the overall software

## 4.2. ROS Functions in Disinfection Robot

### 4.2.1. Node and Master

In the world of ROS, the smallest process unit is a node. There can be multiple executable files in a software package. The executable file becomes a process after running. This process is called a node in ROS. A node is an executable file (usually an executable file generated by C++ compilation, a Python script) executed and loaded into memory from a program point of view. From a functional point of view, usually, a node is responsible for a single function of the robot. Because the robot's functional modules are very complex, we often do not concentrate all the functions on one node. We will adopt a distributed approach.

Due to the robot has many components and huge functions, many nodes are often run in actual operation state. These nodes are responsible for functions such as perception of the world, control of movement, decision-making, and calculation. Proper deployment and management of these nodes are problems that need to be solved. The node manager master provided by ROS is equivalent to the management center in the entire network communication architecture, managing each node. The node is first registered with the master, and then the master will incorporate the node into the entire ROS program. The communication between nodes is also "wired" by the master first, and then peer-to-peer communication can be carried out in pairs. When the ROS program starts, the first step is to start the master, and the node manager will start the node in turn. Figure 4-2 shows us the relationship between node and master.

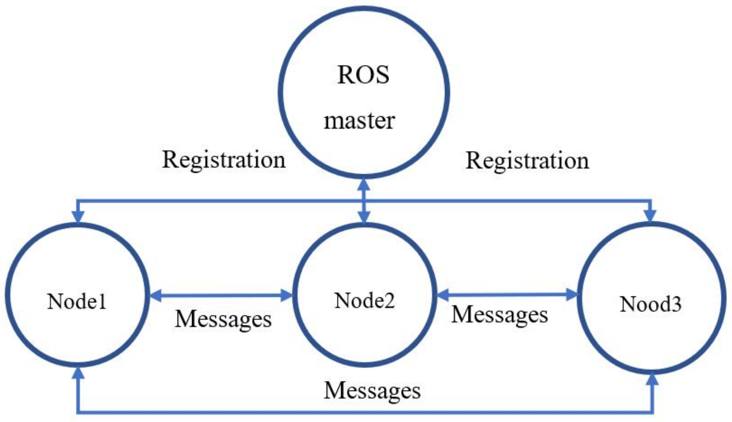


Figure 4-2 The relationship between master and node

### 4.2.2. Coordinate Transform

In the construction of the robot environment, the coordinate system is a crucial concept. On a very complex robot, there will be many different sensors, laser, lidar, radar, camera, etc. These devices have their own relative coordinate system. When the robot's lidar obtains a set of data about the environmental object's coordinate position relative to the robot chassis, this coordinate is only relative to the robot lidar sensor. It is not directly applicable to the execution of the robot chassis. Then the object is relative to the lidar and the chassis. The coordinate transformation between the two is TF. The coordinate transformation includes the transformation of position and posture. The tf in ROS is a software package that allows users to record multiple coordinate systems at any time. TF maintains the relationship between the coordinate systems in the cache tree structure and allows the user to convert points or vectors between any two coordinate systems at any desired point in time.

Disinfect the robot maintainer a tf tree. There are four coordinate systems on this tree. They are base_footprint, map, odom, bass_laser_link. Footprint is the chassis coordinate system defined by us. He and the map coordinate system coincide when the robot first starts to run. There is a conversion relationship between the map coordinate system and base_footprint when the robot leaves the origin. Through this conversion relationship, the position of the robot at this time can be obtained. As long as there is a path between the two coordinate systems in tf, the conversion relationship between the two coordinate systems can be calculated by this tree.

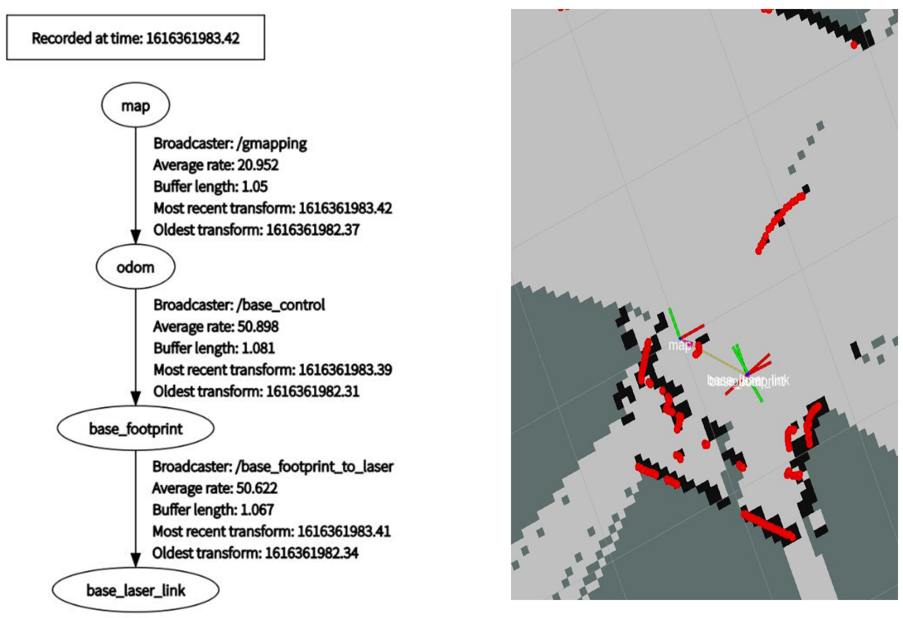

Recorded at time: 1616361983.42
map
Broadcaster: /gmapping
Average rate: 20.952
Buffer length: 1.05
Most recent transform: 1616361983.42
Oldest transform: 1616361982.37
odom
Broadcaster: /base_control
Average rate: 50.898
Buffer length: 1.081
Most recent transform: 1616361983.39
Oldest transform: 1616361982.31
base_footprint
Broadcaster: /base_footprint_to_laser
Average rate: 50.622
Buffer length: 1.067
Most recent transform: 1616361983.41
Oldest transform: 1616361982.34
base_laser_link
map

Figure 4-3 Disinfection robot tf tree

### 4.2.3. SLAM Node

One crucial part of the disinfection robot is the navigation algorithm. Realizing the external environment's perception, locating its position in the environment, and planning a point-to-point path are the essential functions that robots should have today. It requires the robot to be able to perceive changes in the external environment in real-time. It can establish a map of the surrounding environment and determine its own position in the current map while moving, allowing the robot to know which areas have been visited. Currently, SLAM technology occupies a significant position in trackless navigation motion robots. The ability to accurately build a map of the surrounding environment and accurately estimate its own pose is the cornerstone of all service robots and the premise of many advanced functions. Figure 4-4 illustrates the entire SLAM process. The figure mainly contains the following five parts.

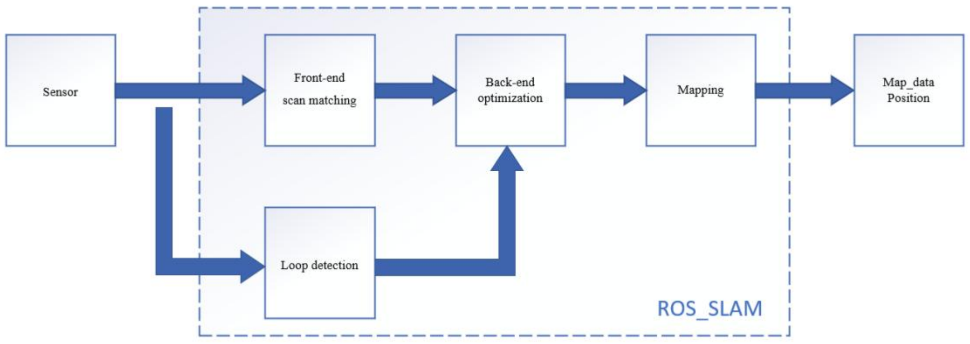


Figure 4-4 SLAM flow chart

The sensor is the input end of the entire SLAM. This part contains various sensors. In visual SLAM, the camera (monocular, binocular, RGB-D) is used as the odometer's data source. Laser SLAM mainly uses lidar to realize odometer. Theoretically speaking, SLAM can also be performed using the only lidar as a sensor. However, due to the external environment's complexity, precise mapping and positioning cannot be achieved only by laser. Therefore, mainstream SLAM solutions usually add IMU and wheel odometer and use multi-sensor data fusion solutions to implement SLAM.

Laser SLAM needs to match the data of two frames to obtain the coordinate transformation relationship in the two image frames to estimate the robot's current posture. Therefore, front-end scan matching is also called a front-end odometer. The front-end scan can now be regarded as a short-term odometer.

However, various noises are mixed into the sensor data. Since each pose estimation uses the previous frame of data as a reference, errors caused by noise will accumulate over time. This accumulation for a long time without correction will cause significant errors in the entire pose estimation and mapping. Make the entire pose estimation and mapping unavailable. Back-end optimization is used to solve this problem. There are many options for back-end optimization. The common ones are filter solutions based on the extended Kalman filter and nonlinear optimization solutions based on graph optimization.

There is an error in the position estimation. The error due to external noise or the robot's own processing will cause the robot to produce different position estimates in the same position. At this time, the loop detection is used to make the robot realize that it is back to the same place, and this information is passed to the back-end optimization so that the accumulated error can be reduced to obtain a more precise positioning and a globally consistent map can be built.

Mapping is an operation that could construct a description of the surrounding environment. The map can be expressed in many ways. Common ones are 2D grid graph, 2D topology graph, 3D point cloud graph, and 3D grid graph. In actual applications, appropriate forms should be selected to build maps for different applications.

ROS provides us with many excellent open-source SLAM algorithms. We can install various slam packages directly in Ubuntu. Use the "rosrun" command to run these nodes directly. What we need to do is to provide these nodes with the topic data they need. We need to provide this algorithm node with lidar data and the coordinate conversion relationship between lidar robot chassis tf_base-lidar, wheel odometer, and the coordinate conversion relationship between chassis wheel odometer tf_base-odom. Figure 4-5 is a typical SLAM node. The slam node performs a gmapping algorithm on the input data to obtain the transformation between map_frame and odom_frame, the pose's estimation, and the map data.

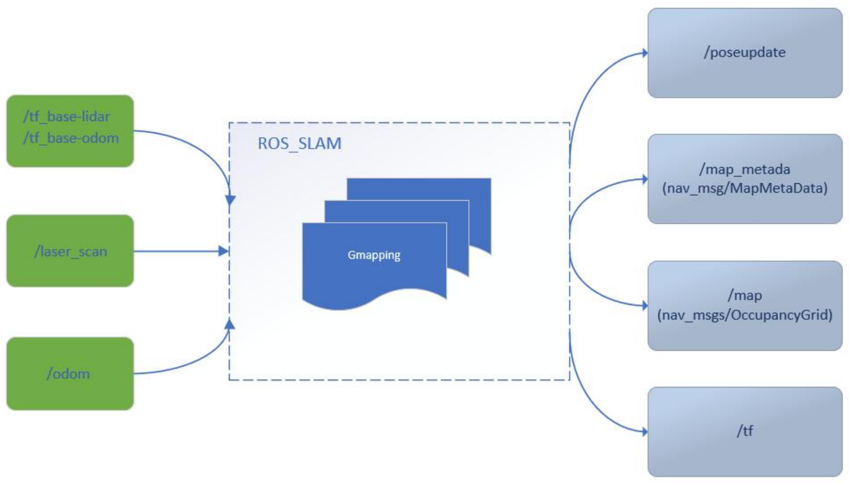


Figure 4-5 slam message flow diagram

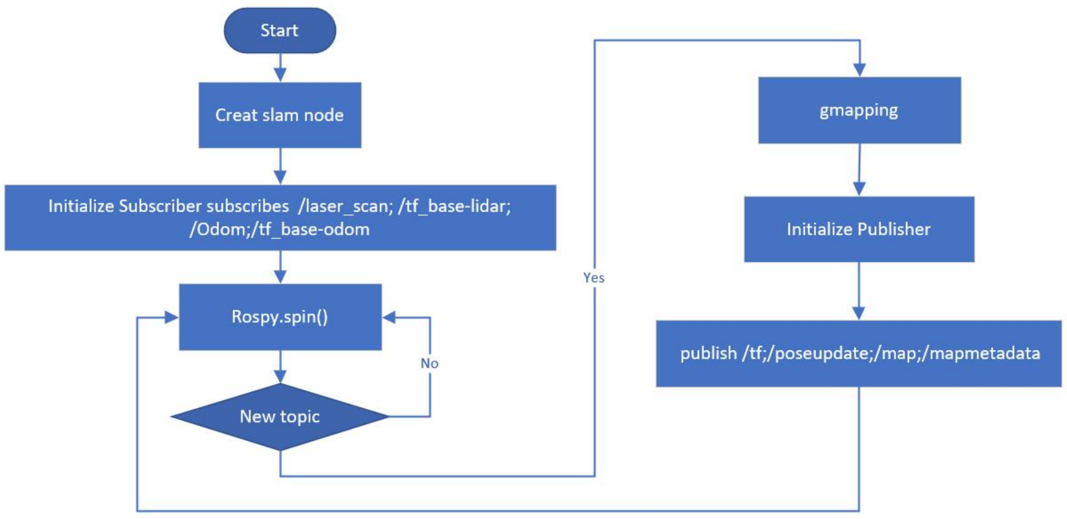

Start
Creat slam node
Initialize Subscriber subscribes /laser_scan; /tf_base-lidar; /Odom;/tf_base-odom
Rospy.spin()
New topic
No
Yes
gmapping
Initialize Publisher
publish /tf;/poseupdate;/map;/mapmetadata

Figure 4-6 Block diagram of slam node

Figure 4-6 is a block diagram of the program flow of the slam node. The slam node needs to be able to subscribe and publish messages. The slam node is first created when the node is executed. Then initialize the subscriber to subscribe to the topics required by gmapping. Then enter the blocking state. When topic data is received, it will enter the callback function to call the gmapping algorithm to process the data. Finally, the results of the processing are published through the publisher. The program returns to the blocking state.

### 4.2.4. AMCL Node

The positioning of the robot refers to the coordinates of the robot in the world coordinate system. It reflects where the robot is sitting at the moment on the map. The wheel odometer can calculate the coordinates of the robot. However, there are errors in the data measured by the wheel odometer. When the wheels slip, the wheel odometer data will be inconsistent with the accurate displacement data. So we need to use other methods to locate the robot. Adaptive Mentcarto Localization (AMCL), Monte Carlo adaptive positioning is a commonly used positioning algorithm that locates by comparing the detected obstacles with a known map. The AMCL node in the ROS system of the disinfection robot adopts an adaptive (or kld sampling) Monte Carlo positioning method and uses a pthesis filter to track a robot's posture against an existing map.

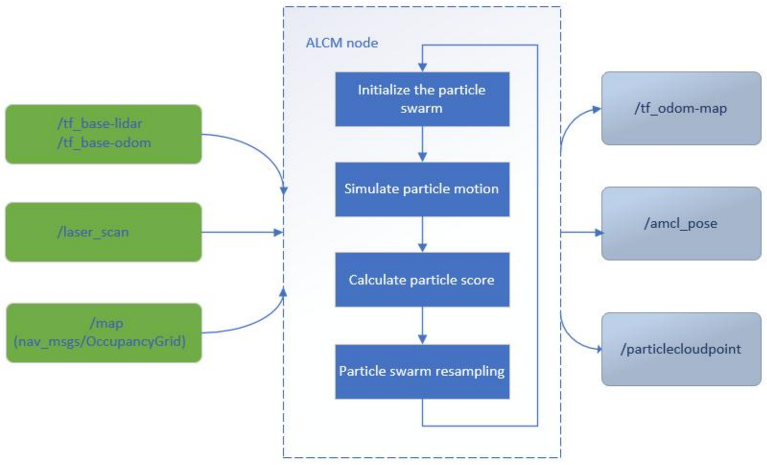


Figure 4-7 AMCL message flow diagram

In ROS, we need to write an AMCL node to publish the robot's pose topic for use by the base move node. Figure 4-7 shows the data flow diagram of the AMCL node. The input data contains lidar scan data and the coordinate conversion relationship between lidar robot chassis tf_base-lidar; wheel odometer data and the coordinate conversion relationship between chassis and wheel odometer tf_base-odom; map data. AMCL node uses these data to use pthesis filters to obtain more accurate positioning of the robot. And the point cloud data of the possible position of the robot. At the same time, AMCL positioning will correct the odometer error. The correction method is to add the odometer error between map_frame and odom_frame, and between odom_frame and base_frame is the odometer measurement value. This measurement value will not be corrected. Finally, a coordinate conversion from odometer to map-frame is output. This tf can be used as path planning.

The specific content of the AMCL node is explained in detail in Figure 4-8. The pthesis filtering principle is to distribute some pthesiss evenly in the map space at first and then move the pthesiss by obtaining the robot's motion. For example, if the robot moves one meter forward, all the pthesiss will also move forward one meter, regardless of the pthesis's current position. Use each pthesis's location to simulate

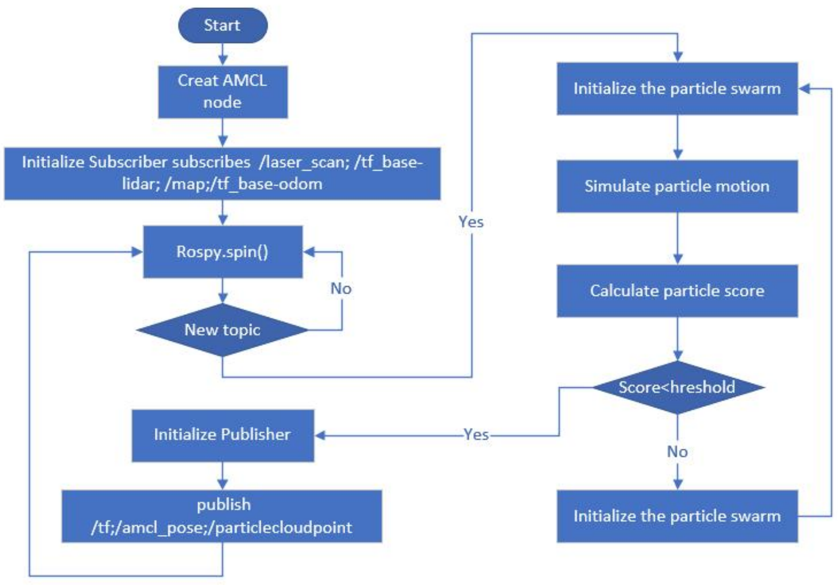

Start
Creat AMCL node
Initialize Subscriber subscribes /laser_scan; /tf_base-lidar; /map;/tf_base-odom
Rospy.spin()
No
New topic
Yes
Initialize the particle swarm
Simulate particle motion
Calculate particle score
Score<hreshold
Yes
No
Initialize Publisher
publish /tf;/amcl_pose;/particlecloudpoint
Initialize the particle swarm

Figure 4-8 Block diagram of AMCL node

sensor information and compare it with the observed sensor information (usually a laser), thereby assigning a score to each pthesis. Afterward, the pthesiss are regenerated according to the scores, and the pthesiss with higher scores are given higher weights. After this iteration, all the pthesiss will slowly converge together, and the exact position will be calculated.

AMCL algorithm steps are shown in Figure 4-8. It is mainly divided into four steps: (1) Initialize the pthesis swarm. Since the initial position has been given an accurate value, we only need to set the population size as the initial population. In the actual application of slam, if the initial position is uncertain, the random sampling method (random function) of Gaussian distribution can initialize the pthesis swarm. (2) Simulate pthesis motion. Use Kalman filter to model robot motion. It is also possible not to model the robot's motion and directly assume that the robot has a limited range of motion within the interval of two sampling time points and then use the random function to generate the motion's possible positions randomly. (3) Calculate the pthesis score. For each laser ray, obtain the match between the sensor reading and the map. And record the score. (4) Pthesis swarm resampling. After the scoring is over, we will find that some pthesiss have poor scores. That is, they seriously deviate from the possible positions. We need to discard these pthesiss. Some pthesiss have very high scores and are very close (for example, pthesiss whose sensor readings are more than 80% consistent with the map), and we need to keep them all. It is pthesis swarm resampling. We directly discard the pthesiss with too low scores in actual operation, copy the pthesiss with high scores, and keep the number of pthesis swarms basically unchanged after resampling. Repeat the above steps until the position converges to the threshold that we set to output the position result.

### 4.2.5. Move Base Node

Navigation is one of the most basic functions of the robot. ROS provides us with a complete set of navigation solutions, including global and local path planning, cost maps, abnormal behavior recovery, map servers, etc. These open-source toolkits significantly reduce the workload. Any set of mobile robot hardware platforms can be quickly deployed through this set of solutions. Among them, the move base is a significant path planning node in navigation.

Figure 4-9 shows the hierarchical diagram of the move base node. the move base node is at the center of the navigation function, which can be understood as a powerful path planner. We need to start this node in the navigation task and provide it with data to plan the path and speed. Move base can achieve path planning because it contains many plug-ins: global_planner, local_planner, global_costmap, local_costmap, recovery_behaviors. These plugins are used for some more subtle tasks: global planning, local planning, global map, local map, recovery behavior. And each plug-in is a package placed in the Navigation Stack. A lot of work has been done for us in ROS. What we need is to provide move base with the data it needs. These data include tf tree, odometer data, grid map, radar scan data, target coordinates. Through the move base node calculation, the speed of the left and right motors is obtained. The move base publishes the speed value in the form of a topic. This topic will be subscribed by the serial port node and sent to the bottom control board.

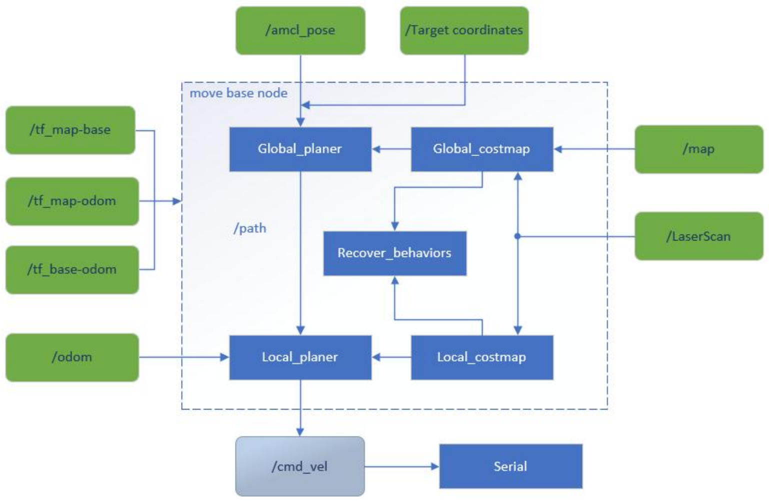


Figure 4-9 Block diagram of move base

## 4.3. Nodes Communication

The ROS communication architecture includes the processing of various data, processes, and transmission of messages. There are four communication methods in ROS: topic, service, parameter server, and action library. Each communication method has its own characteristics. This section introduces the communication methods and related concepts of the communication architecture and introduces the communication process between nodes in the disinfection robot.

### 4.3.1. Topic

Topic is the most commonly used communication method in ROS. For real-time and periodic messages, using the topic to transmit is the best choice. Topic is a point-to-point one-way communication method. The "point" here refers to a node. In other words, information can be transmitted between nodes through the topic. The topic needs to go through the following steps of the initialization process: (1) publisher node and the subscriber node register in the node manager. (2) The publisher publishes the topic. (3) The subscriber subscribes to the topic under the command of the master. The communication between sub-pubs is established, and the whole process is one-way.

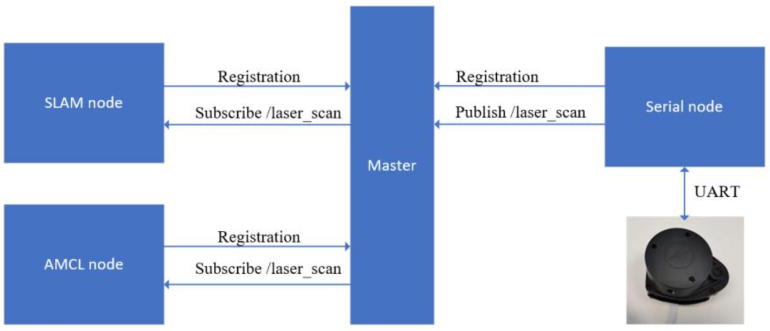


Figure 4-10 Diagram of transmitting lidar data by topic communication

Figure 4-10 shows the process of using the Topic communication mode in ROS to transmit lidar data in this disinfection robot. The serial port node is the node responsible for collecting lidar data. At startup, it will publish a theme named "/laser_scan" to the master. Other nodes can subscribe to this topic. In Figure 4-10, both "SLAM node" and "AMCL node" are subscribed to this topic. After the introduction of the Master, the two nodes can establish communication. As can be seen from the figure, Topic is an asynchronous communication method. After the serial port node publishes each message, it will continue to execute the following action without any feedback. The serial node does not need to know the message's status and its processing method. The same "SLAM node" and "AMCL node" received the message through the subject. They don't care about who sent it. Many messages in the disinfection robot are transmitted in the form of topics. Their transmission structure is consistent with the lidar data transmission structure. This thesis will not repeat them here.

### 4.3.2. Service

Topic is a one-way asynchronous communication method in ROS. However, sometimes one-way communication cannot meet the communication requirements. For example, in the disinfection robot, the map data is not periodically required. The amount of map data is enormous, and frequent conferences using topic communication will consume many unnecessary system resources, resulting in low efficiency and high power consumption. In this case, another request-query communication model is needed. ROS provides us with a communication mechanism called service.

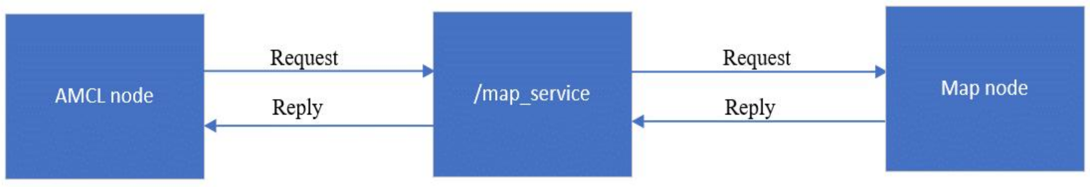


Figure 4-11 Diagram of transmitting map data by service communication

The process of AMCL node using service communication to obtain map data is shown in Figure 4-11: map_server (responder) provides a service interface called /map_service. We generally use string type to refer to the name of the service. AMCL node initiated a request to Map node and got feedback after processing. Process Service is a synchronous communication method. The so-called synchronization means that the AMCL node will wait for the reply in place after publishing the request. The AMCL node will not continue to execute until the Map node has processed the request and completed the reply. While the AMCL node is waiting, the communication is in a blocked state. Such a communication model has no frequent message transfer, no conflicts, high system resource occupation, and services executed only when requests are accepted, which is efficient and straightforward.

### 4.3.3. Parameter Server

The robot's parameters must be set when the robot works (such as sensor parameters, algorithm parameters). Some parameters (such as the robot's contour, the sensor's height) can be set when the robot is started. In contrast, some parameters need to be changed dynamically (especially during debugging). ROS provides a parameter server to meet this demand. We can set the parameter server parameters and get them from the parameter server when we need them. Unlike topic and service communication methods, the parameter server can be a particular "communication method." The particular point is that the parameter server is the place where the node stores the parameters, is used to configure the parameters, and the parameters are shared globally. The parameter server uses Internet transmission and runs in the node manager to realize the entire communication process.

The parameter server maintains a data dictionary in which various parameters and configurations are stored. A dictionary is a key-value pair. Each key is not repeated, and each key corresponds to a value. It can also be said that a dictionary is a mapping relationship. In actual project applications, we often put some rarely used parameters and configurations into the dictionary in the parameter server because of the dictionary's static mapping characteristics. Data reading and writing will be convenient and efficient. Table 4-1 shows some robot parameters set in the disinfection robot. These parameters can be loaded directly at runtime using the ROS command or manually set at runtime.

### 4.3.4. Action

Similar to the service communication mechanism, actionlib is also a communication method of request-response mechanism. Actionlib mainly makes up for a shortcoming of service communication. When the robot performs a long-term task, if the service communication method is used, then the publisher will not receive the feedback reply for a long time, causing the communication to be blocked. When the service communication cannot complete the task well, the actionlib can be more suitable for realizing the long-term communication process. The actionlib communication process can be viewed at any time, and the request can also be terminated. This feature makes it has high efficiency in some special mechanisms.

## 4.4. Board-to-Board Communication

The control board and the navigation board need to exchange sensor data, motor speed data, disinfection module working status data, etc. This section mainly describes how to realize data communication between two boards.

### 4.4.1. Communication Method Selection

There are many communication methods between two boards, which can be roughly divided into wired communication and wireless communication. Wired communication refers to the existence of entire lines in the communication process. This method relies on lines for effective information transmission. Use wires to transmit signals from one communication transmitter to another communication receiver. Standard wired communications include UART, SPI, IIC, etc. And wireless communication refers to the transmission of information without any wires in the process of information transmission. In this way, the transmission tower is used to exchange information. Standard wireless communication methods include Bluetooth, WLAN, and Cellular Network. The advantage of wired communication: (1) It can keep the signal transmission speedwell. (2) The anti-interference ability is strong. (3) It does not produce radiation and is relatively safe and reliable. Disadvantages: There are line restrictions, limitations, and increased investment costs. The advantage of wireless communication: (1) Not limited by the line. (2) Facilitates people's daily life and work. (3)Expands the areas and regions where people work. Disadvantages: (1) Affected by electromagnetic factors. (2) The transmitted signal is susceptible to interference and instability. (3) Produce a large amount of radiation to cause harm to the human body.

Raspberry Pi 3B+ supports multiple communication methods such as Bluetooth, WiFi, serial port, IIC, SPI, GPRS, etc. There is no Bluetooth module on the bottom control board, but a WiFi module is reserved, and the microprocessor on the bottom control board has its own serial port. So, the available communication methods are WiFi and serial port. The use of WiFi requires an additional WiFi module on the bottom plate, which will increase the robot's cost and increase the burden on the underlying embedded software. And the distance between the bottom control board and the navigation board is not far, and the two boards are installed in a fixed position with the robot—no need to use wireless communication. Therefore, the wired communication method should be more economical and reliable. As the most commonly used and simplest communication method in wired communication, the UART has simple, reliable, and low resource consumption characteristics. Moreover, the communication between the two boards does not require a high communication rate, and the UART can meet its communication rate requirements. The UART configuration used in disinfection robots is: baud rate 115200, 8 data bits, no parity, 1 stop bit.

### 4.4.2. Communication Protocol Design

UART is a communication method and a carrier. In order to exchange data more efficiently, we also need to develop a communication protocol based on UART. The communication protocol is the rules and agreements that the two entities must follow to complete the communication or service. The protocol defines the data unit's format, the information and meaning that the information unit should contain, the connection method, and the timing of information sending and receiving. Ensure that the data is smoothly transmitted to the determined terminal.

Figure 4-12 shows the specific content of this communication protocol. The picture is a frame of data for this communication protocol. It consists of the frame header, frame length, function code, data segment, and check code. The frame header is a fixed value: 0xAA. The function code represents the meaning of the data segment in this frame of data. The definition of some specific function codes is given in table 4-3. The function codes sent by the control board are even numbers, and the function codes received by the control boards are odd numbers. The check method is CRC-8/MAXIM.

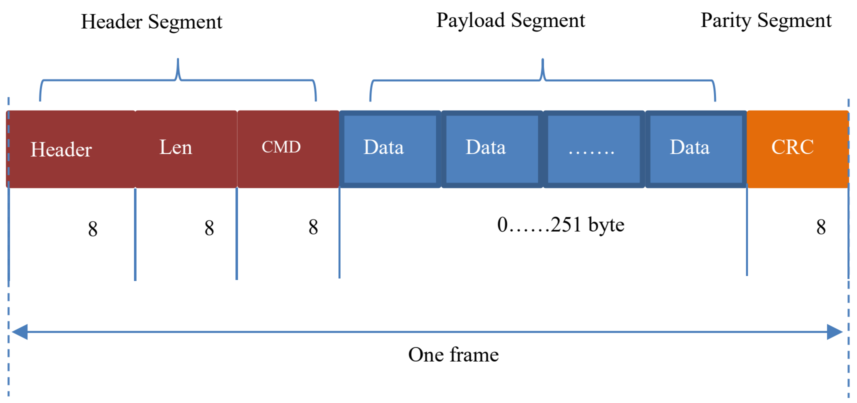


Figure 4-12 Communication protocol frame format

**Table 4-3** Command code and Specifications

| CMD | Specifications |
|---|---|
| 0x01 | The core board sends a speed control instruction to the control board. The data length is 6Byte, and the data is X-axis direction speed*1000(int16_t) + Y-axis direction speed*1000(int16_t) + Z-axis angular speed*1000(int16_t) |
| 0x02 | The control board replies to the speed control command of the core board, the data length is 1 Byte. There will be a return only when the speed setting fails, and there will be no response under normal circumstances. |
| 0x03 | The core board sends a speed query command to the control board, and the data length is 0 Byte |
| 0x04 | The current speed is reported on the control board, the data length is 6Byte, and the data is X-axis direction speed*1000(int16_t) + Y-axis direction speed*1000(int16_t) + Z-axis angular speed*1000(int16_t) |

| | |
|---|---|
| 0x05 | The core board queries the control board for IMU data, the data length is 0Byte |
| 0x06 | The control board reports IMU data, the data length is 6Byte, and the data is Pitch*1000(int16_t) + Roll*1000(int16_t) + Yaw*1000(int16_t) |
| 0x07 | The navigation board obtains the odometer information from the control board |
| 0x08 | The speed and heading information is reported on the control board, the data length is 6Byte, and the data is linear velocity*1000, angle*100, angular velocity*1000 |
| 0x09 | The core board sends a restart command to the control board, no reply |

### 4.4.3. Implementation of The Board-to-Board Communication

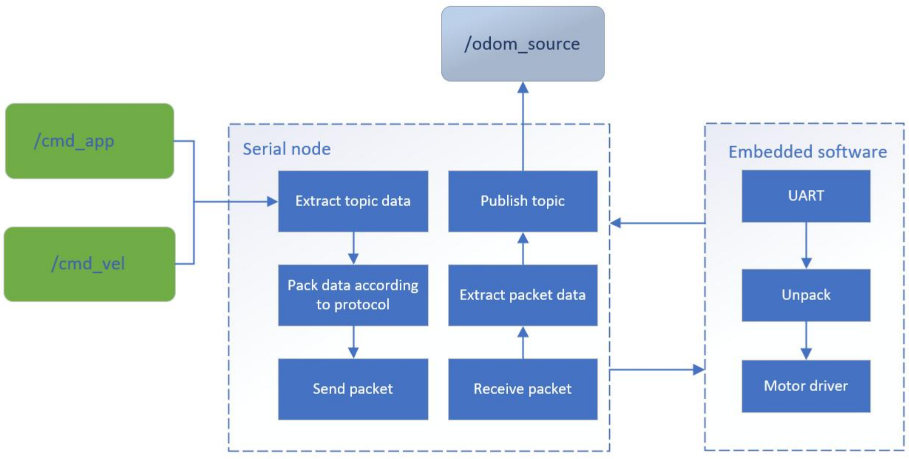


Figure 4-13 Schematic diagram of inter-board communication

There is a serial node in the disinfection robot software system responsible for data transmission with the bottom control board. ROS provides serial port libraries in multiple languages. We use the provided python library to write our serial node code. The serial port node obtains the disinfection module control instruction sent by the smartphone by subscribing to the cmd_app topic published by the cmd_process node to obtain the disinfection module's work instruction. Get the motor speed value by subscribing to the cmd_vel topic of the "Mobile Basic Node" node. The above two topics are packaged into data packets according to the communication protocol in the previous section and sent to the lower computer's control board using the serial port of the Raspberry Pi. The serial port node will publish the odometer information received from the bottom control board through the theme. The name of the theme is called "/odom_source". The communication process is shown in Figure 4-13.

The serial port node needs to perform four tasks: reading serial data, publishing topics, subscribing to topics, and sending serial data. Use two threads to complete the above operations. Figure 4-14 shows the entire process. In the serial port node, we first create a node, which will subscribe to the two topics "/cmd_app" and "/cmd_vel". Then the node will create a thread B. Execute in the main thread to obtain topic data. After the topic data is packaged according to the defined communication protocol, the serial port is called to send. Thread B is responsible for reading serial port data. When there is data in the serial buffer, and the serial port is available, the serial buffer data will be read. And analyze the data according to the protocol. Finally, the parsed data will be published in the form of topics.

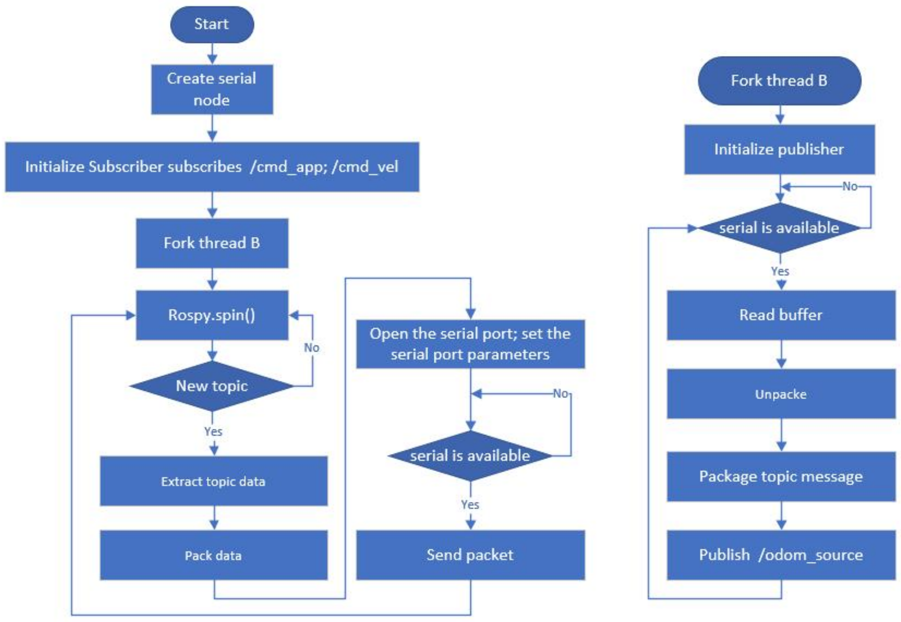


Figure 4-14 Schematic diagram of inter-board communication

## 4.5. Android App of Disinfection Robot

There are only four human-computer interaction buttons on the disinfection robot. The control settings of many disinfection robots cannot be set via buttons. And the disinfection robot is always moving when it is working. It is difficult for the user to operate the disinfection robot in a working state. Therefore, there must be a remote control port to control the robot. Now, almost every family has a smartphone. Choosing a smartphone APP to control the robot is both convenient and cost-effective. Android is currently the most widely used operating system for smartphones. Therefore, the APP of this disinfection robot uses AndroidStudio to develop.

### 4.5.1. Functional Requirements Analysis

**Communication:** APP is an application to control the robot. It first needs to solve the communication problem between APP and robot. The early prototype does not consider connecting to the external network. Therefore, in the early stage, only by obtaining the IP address of the robot under the same local area network can establish a TCP/IP connection with the robot and then send and receive data. Similarly, we also need to use a complete communication protocol based on the TCP/IP protocol to facilitate our various data transmission types to the robot.

**Identity verification:** when the robot connects to the local area network, it is likely to receive connection requests from other terminals. If no verification is done for the connection request. Then the robot will be in a state where anyone can connect and control. Therefore, there must be a corresponding verification mechanism for connection requests.

**Motion control and map and coordinate display:** the primary function of APP is to control robot movement. For two-dimensional plane movement, it is mainly divided into four actions: forward, backward, left turn, and right turn. The robot should also display the established map in real-time and mark the auto-generated coordinate position on the map.

### 4.5.2. Functional Design

#### Communication between ROS and Android app

ROS is a framework that runs on the Ubuntu operating system. It can use a socket to connect to other hosts. When a socket can be established between the robot and the mobile phone, the communication between the two can be realized. But because the language bindings officially supported by ROS are only C++ and Python. Therefore, if Android wants to communicate with ROS, the semi-official rosjava package must be used. And Rosjava is too big because it is a fully functional ROS binding like C++/Python . We can create a master node on the Java (android) platform. Other nodes (C++/Python) can connect to this master node for distributed communication. For mobile android clients, this is too complicated. Besides, the Gradle script of rosjava is too complicated and requires in-depth Gradle knowledge to integrate it into your own android project. We need a lightweight method to communicate with ROS.

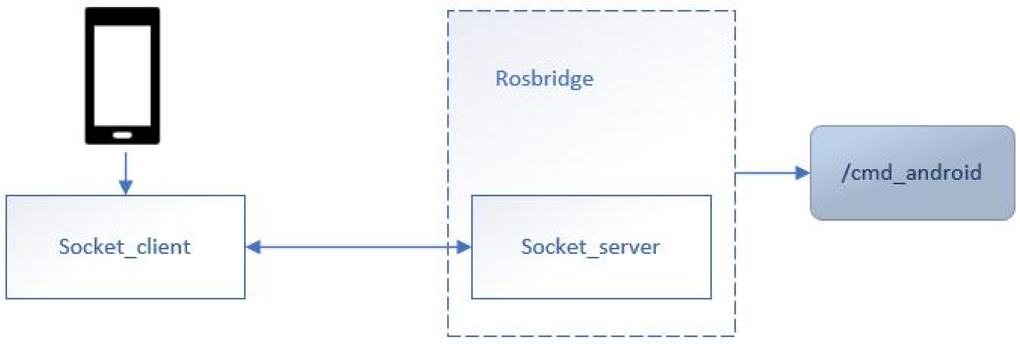


Figure 4-15 ROSbridge communication architecture

For mobile platforms or embedded systems, it is too resource-intensive to implement distributed communication with ROS in the form of nodes. Therefore, the use of C/S architecture may be more in line with the needs of mobile platforms. Therefore, ROS formally proposed the ROS bridging protocol. This protocol's basic idea is to change the distributed communication between nodes to C/S communication between client nodes and agent nodes. Then, the proxy node forwards the request to the server node.

In this way, the mobile terminal does not need to implement the entire ROS platform but only needs to perform socket communication with the agent node.

The entire communication architecture is shown in Figure 4-15. The robot in the picture can be seen as a server. The server provides services through socket sockets. Any terminal that can use sockets can communicate with the robot through sockets. It means that you can run WebSocket on the computer and control the robot. The mobile phone is now regarded as a client. Use the socket class to create a socket instance under the Android platform to communicate with the robot.

### Authentication

Identity verification is necessary to protect the robot from other malicious terminals. The development of early prototypes pays attention to timeliness. For authentication, only password verification is performed. The password is set in advance inside the robot. The whole process is shown in Figure 4-16. (a) As the client, the mobile phone first sends a TCP connection request. Send the account and password to the robot via TCP protocol. Let the APP block the current state and wait for the connection result. Enter MainActivity when the connection is successful. When the connection fails, it returns to the start state. Figure 4-16(b) The robot end is waiting for connection as a server. When there is a connection request, obtain the account and password in the requested connection, and match the account password with the password stored by itself. If the match is successful, the connection is established. Otherwise, the connection is rejected. The state of waiting for connection will be returned.

### Main Activity

The main interface can be divided into two parts: map display; motion control. Motion control is implemented using button monitoring events. AndroidStudio provides Button components. The component can be bound to click events. The corresponding action can be realized when the user triggers the button. The button component used this time is a single trigger button, and a trigger event is generated whenever the user clicks it.

The flow of the trigger is shown in Figure 4-17. When a key event is triggered, we first get the button ID. Match the ID through the Switch Case statement and execute the corresponding event code block. All the statements in the code block send messages to the robot. Different code blocks send different messages with the command code represented by this button. The robot executes the corresponding action after obtaining the command. The concrete realization needs to rely on the ROS bridge library.

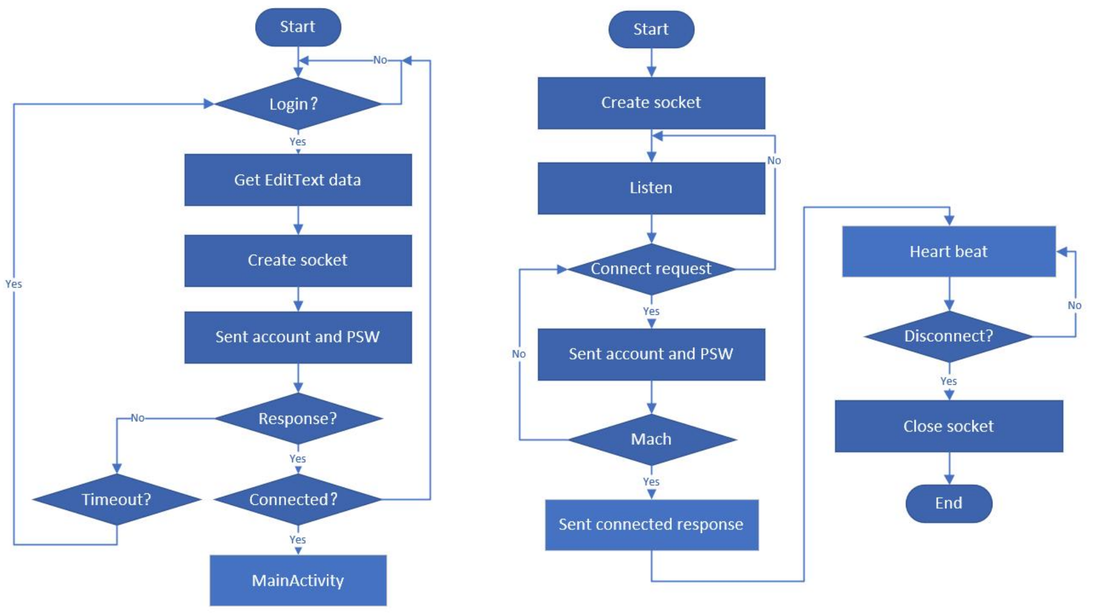


Figure 4-16 (a) Client  (b) Robot

Another critical function of the main interface is to display the map. The map display needs to obtain the map data collected by the robot. The map data needs to be continuously refreshed in Main Activity to display the map in real-time. The refresh operation of the map uses a double for loop statement to draw a two-dimensional coordinate system. The specific process is shown in Figure 4-18. First, get the width and height of the map. The bitmap can be regarded as a two-dimensional array. The operation of drawing is to mark the gray value of each two-dimensional array. Finally, it can be shown through the ImageView component.

## 4.5.3. UI Design

The two UI interfaces are relatively simple. But it can basically fulfill the needs of the prototype in the early stage. Figure 4-19(a) shows the login interface. Use the

EditText component in AndroidStudio to get the user's input information. Users can enter their account passwords in EditText. Click the LOGIN button to try to connect to the robot. When the connection is unsuccessful, an error message "connection failed" will pop up. Users can connect different robots under the same LAN by modifying the IP address. The username and password are stored inside the robot. You need to enter the correct account and password combination to enter the MainActivity interface.

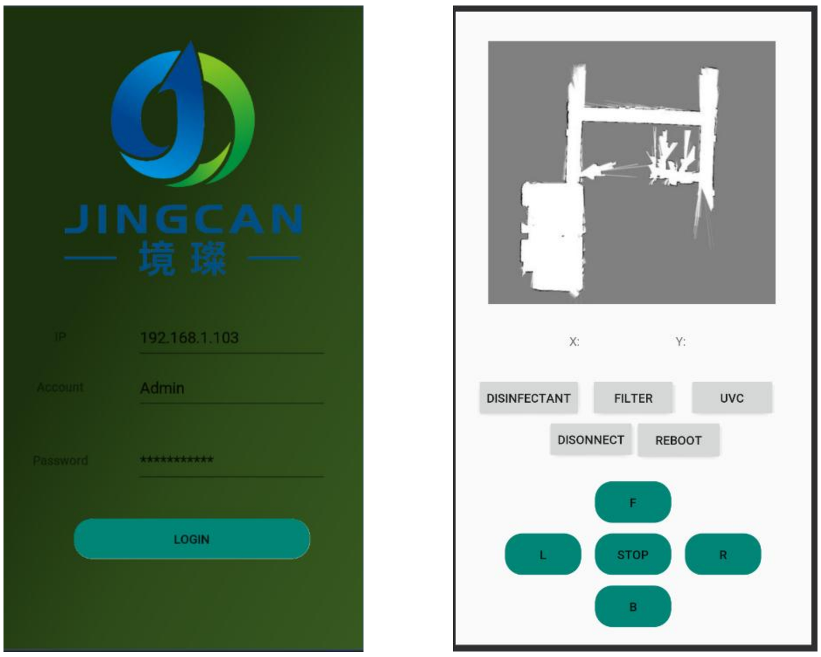

IP 192.168.1.103
Account Admin
Password ************

Figure 4-19 (a) Login Activity (b) Main Activity

Figure 4-19 (b) shows the Main Activity interface. The upper part of the interface is the map display area. This area will visualize the map data into a map. Users can intuitively see the SLAM mapping process. Below are five function buttons. They are disinfectant, filter, UVC, disconnect, reboot. A disinfectant is used to control the opening and closing of ultrasonic atomization. The filter is to control the opening and closing of the centrifugal fan in the air purifier. UVC is a switch that controls the ultraviolet lamp. The disconnect is used to disconnect from the robot and return to the login interface. Reboot is to restart the robot.

## 4.6. Summary

This chapter mainly introduces the disinfection robot software system. It mainly introduces the functions of some primary nodes in ROS and the communication methods between nodes. Simultaneously, the API interface provided by ROS is used to realize the communication with the embedded bottom control board. For better users' experience and facilitate users' operation, this thesis also designed a simple Android App for robot operation. This thesis using ROSbridge to solve the communicatiosn problem between the robot and the Android App.

# Chapter 5 Tests and Results

This chapter mainly evaluates the positioning and mapping functions and disinfection modules of the disinfection robot. This experiment uses various open-source SLAM algorithms for positioning and mapping tests. Through experiments, the CPU usage rate and power consumption of different algorithms are analyzed to obtain the most suitable SLAM algorithm for disinfection robots.

## 5.1. Disinfection Module and Movement Test

Use ssh tool on the computer to connect to the robot and configure it to connect to the local area network through WIFI automatically. Fix the IP address of the robot under the local area network. Use the developed Android APP to input the robot's IP address to connect to the robot. When the connection is successful, the APP will automatically jump to MainActivity.

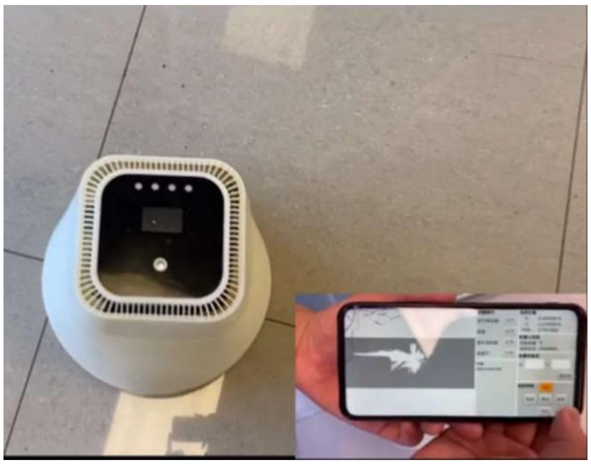
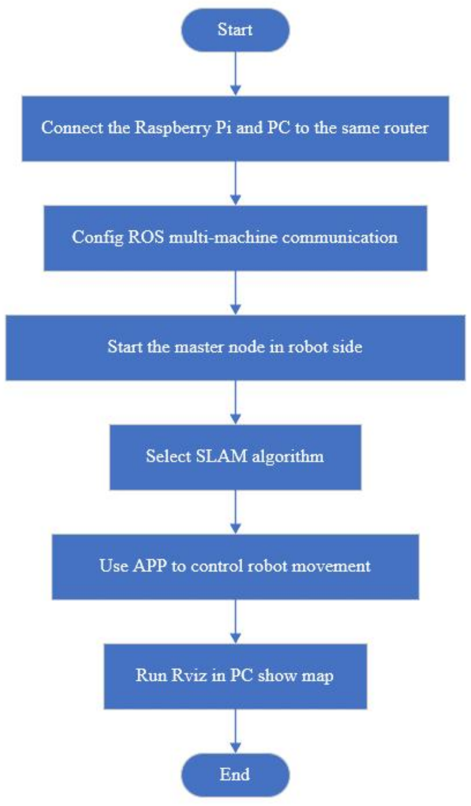


Figure 5-1 APP control test

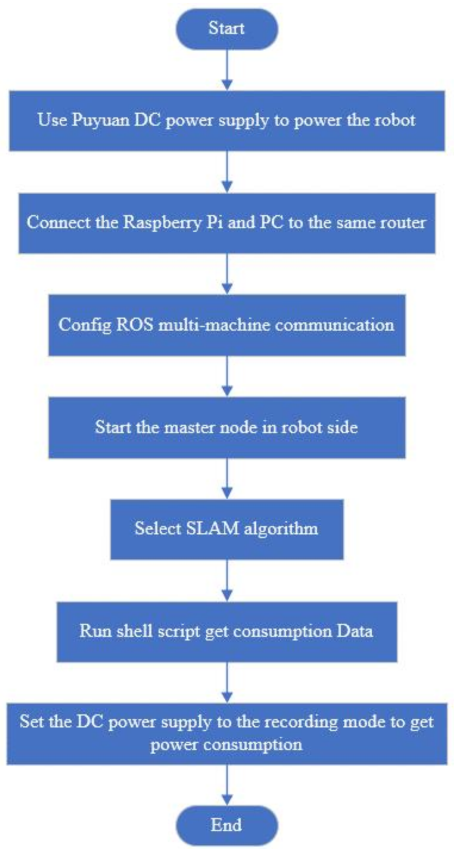
In MainActivity, click the forward, back, left, and right buttons on the screen to move the robot accordingly. After clicking the disinfection button, the disinfection robot turns on the centrifugal fan, UVC lamp, and ultrasonic atomizer for disinfection operation. A specific disinfection method can be turned off by clicking the corresponding button. Figure 5-1 shows the scene of the disinfection robot working during the experiment.

## 5.2. SLAM Test and Result Analysis

### 5.2.1. SLAM Test


Start
Connect the Raspberry Pi and PC to the same router
Config ROS multi-machine communication
Start the master node in robot side
Select SLAM algorithm
Use APP to control robot movement
Run Rviz in PC show map
End

**(a)**


Start
Use Puyuan DC power supply to power the robot
Connect the Raspberry Pi and PC to the same router
Config ROS multi-machine communication
Start the master node in robot side
Select SLAM algorithm
Run shell script get consumption Data
Set the DC power supply to the recording mode to get power consumption
End

**(b)**

Figure 5-2 (a) SLAM test steps (b) Resource consumption test steps

This thesis's experimental site is the meeting room on the 7th floor of Fudan University Jiangwan Campus's teaching building. In the experiment, a laptop is used to connect to the Raspberry Pi. Run the SLAM algorithm in ROS. The mobile phone controls the robot to walk into the meeting room and establish a meeting room map. The Rviz tool in ROS can display the robot's current position and the map created in real-time.Figure 5-2 describes the main steps of the SLAM test steps and resource consumption test steps.

## 5.2.2. Results and Analysis

### Gmapping

Gmapping is currently the most widely used SLAM framework in SLAM. Unlike cartographers, the mapping method mainly uses the pthesis filter method based on RBPF (Rao-Blackwellized Pthesis Filter) for position estimation. Gmapping integrates IMU and wheels odometer data. Therefore, the high scanning frequency of lidar is not required. It first calculates the trajectory of the robot through odometer data and sensor observation data. Then calculate the probability distribution of the map based on the robot's trajectory and the known observations. RBPF uses pthesis filtering to estimate the posterior probability. Since the trajectory is highly related to the environment map, each pthesis corresponds to a separate map in the robot trajectory and the environment. Finally, the pthesiss with the highest probability will be selected for mapping. However, since each pthesis in the pthesis filter has a graph, the calculation amount is relatively large when constructing a large graph. At the same time, resampling may also cause problems such as pthesis consumption or reduced pthesis diversity. Therefore, "mapping" is not suitable for map construction of large scenes, but the effect of "mapping" is better when building maps indoors.

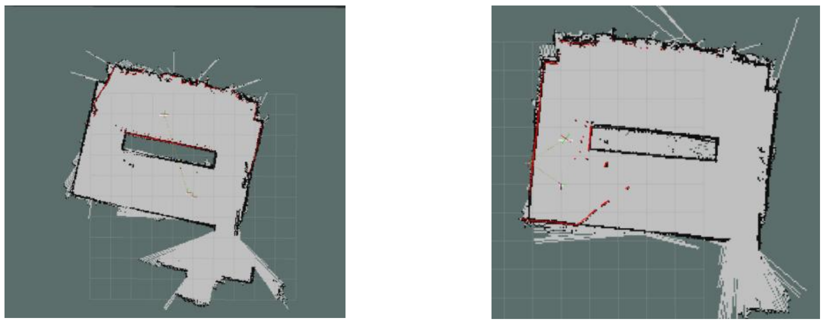
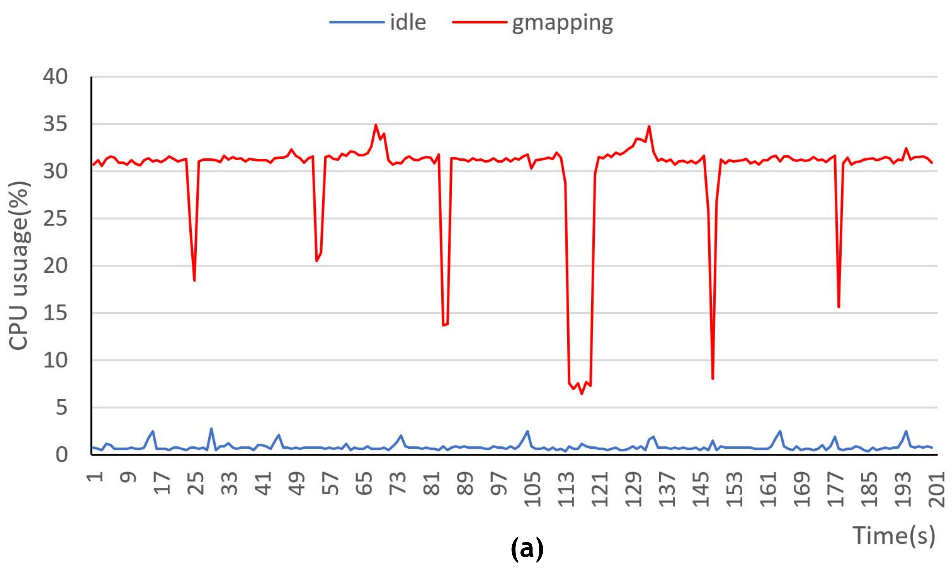


Figure 5-3 Gmapping mapping result

Figure 5-3 shows two images created using Gmapping. Among them (a) is the result when the wheels are running normally without slipping. (b) Apply a little oil on the wheels. The result was obtained by allowing the robot to slip while walking. The environment map can be successfully established in both experiments. The robot can mark autogenous coordinates in the created map. When the wheel slips, the map created is basically the same as the one that does not slip. The reason is that Gmapping uses the fusion method of lidar point cloud data and point cloud data for positioning and mapping. When a certain wheel odometer data is unreliable, the weight of the wheel odometer data can be reduced to reduce the impact on positioning and mapping.



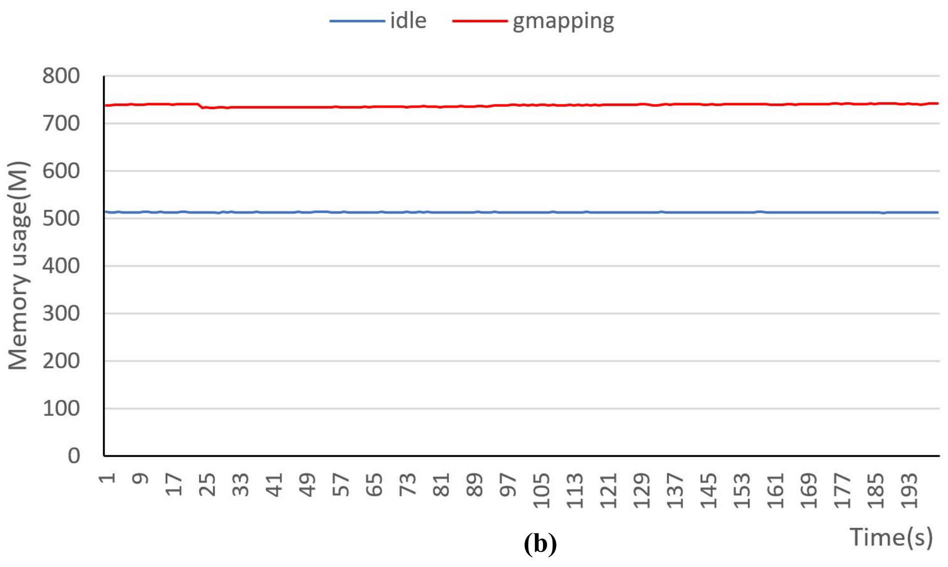


Figure 5-4 Gmapping hardware resource consumption. (a) Gmapping algorithm running CPU usage. (b) Gmapping algorithm running memory usage.

Figure 5-4 shows the CPU usage and memory usage of the Raspberry Pi when running gmapping. It can be seen from the figure that the CPU usage rate is maintained at about 32% when running gmapping. 30% more than when idle. The memory usage is about 740M, which is about 220M more than when idle. It can be seen that running the gmapping algorithm will add a lot of burden to the CPU.

### Hector SLAM

Hector SLAM is a SLAM algorithm that does not use any odometer as a sensor except lidar. Since only the lidar sensor data is used in the whole algorithm for positioning and mapping, the algorithm has higher requirements for lidar. Generally, using the Hector SLAM algorithm requires the lidar's scanning frequency to be higher than 40 frames per second. Since no other sensors are needed, Hector is often used in situations where the wheel odometer fails, such as earthquake relief and aerial drones. Hector is prone to mismatches when the robot turns quickly, and the built map is misaligned. The main reason is that the optimization algorithm easily falls into a local minimum. Therefore, when using Hector, the robot's moving speed should be as low as possible to ensure the establishment of high-precision maps. In order to avoid the Hector algorithm falling into local minimization, a multi-resolution map representation is used. The space robot's pose estimation in the three-dimensional space navigation needs to join the IMU and use EKF for filtering.

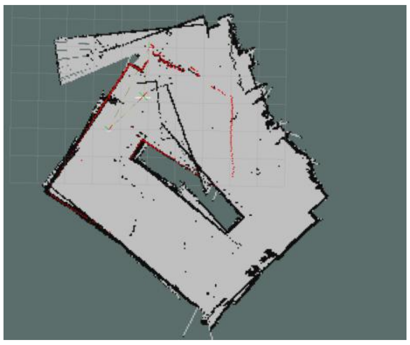
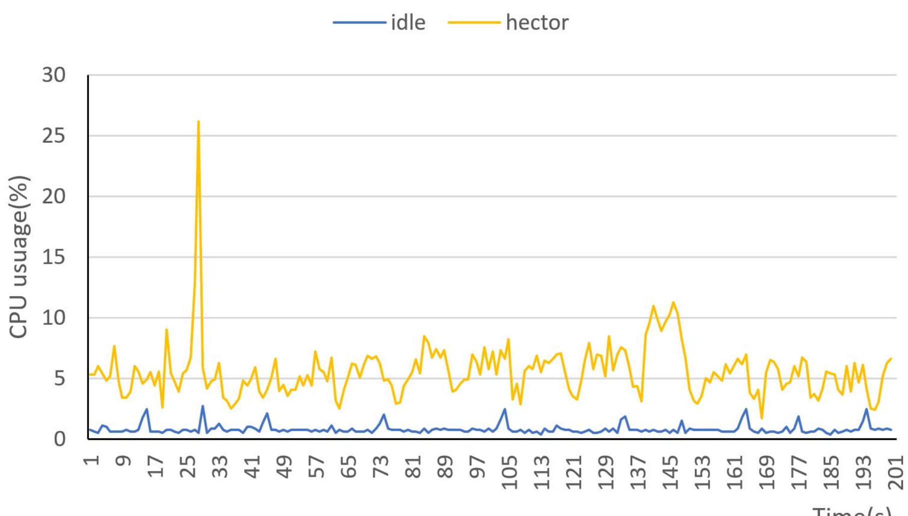
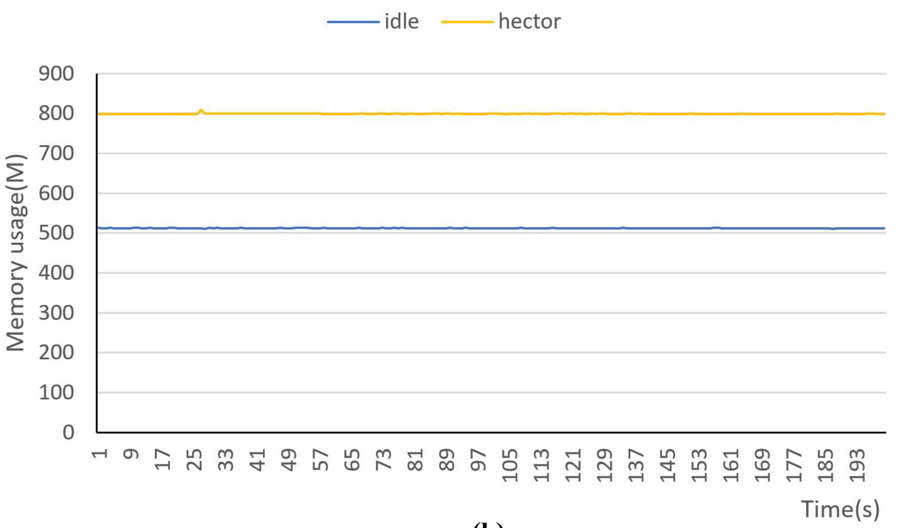


Figure 5-5 Hector mapping result

Figure 5-5 shows the map obtained using the hector algorithm. The map is quite different from the actual terrain at the corners. The surrounding environment is not well established. The reason is the low performance of the lidar used this time. Its scanning frequency is only 5Hz. When the robot has a large displacement at the corner in a short time, the lidar data becomes unreliable. Hector only uses lidar data and does not have wheel odometry as correction data. Therefore, the result of the deviation at the corner is formed.



(a)



(b)

Figure 5-6 Hector hardware resource consumption. (a) Hector algorithm running CPU usage. (b) Hector algorithm running memory usage.

Figure 5-6 shows the CPU usage and memory usage of the Raspberry Pi when running hector. It can be seen from the figure that the CPU usage rate fluctuates between 5%-10% when running hector. The average utilization rate is 5.57%. Only 4% more than when idle. The memory usage is about 800M, which is about 300M more than when idle. It can be seen that the average CPU usage rate of running the hector algorithm is low. But its memory usage is relatively high.

## Cartographer

Cartographer is an indoor SLAM algorithm based on graph optimization launched by Google. The algorithm provides an initial value for predicting the pose and map through the IMU, and the back end uses the map optimization method to make corrections to the prediction. Achieve the function of establishing high-precision positioning and high-precision mapping. The cartographer can be regarded as two independent but related subsystems, Local SLAM and Global SLAM. Local SLAM can be regarded as a front-end (local trajectory builder), and the task is to build a submap. Each submap is locally consistent but will Drift slowly. Global SLAM is the back end used for loop closure constraints. It works by matching scans (collected into nodes) and sub-maps. It can also be combined with data from other sensors to improve accuracy further. Determine the most consistent Global optimization solution. 3D SLAM will also use the direction of gravity provided by the accelerometer.

Figure 5-7 is the result of using the Cartographer algorithm. It is the same as Gmapping,

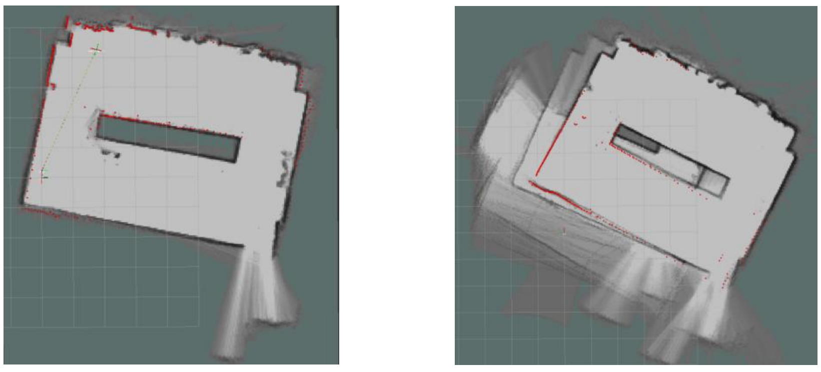
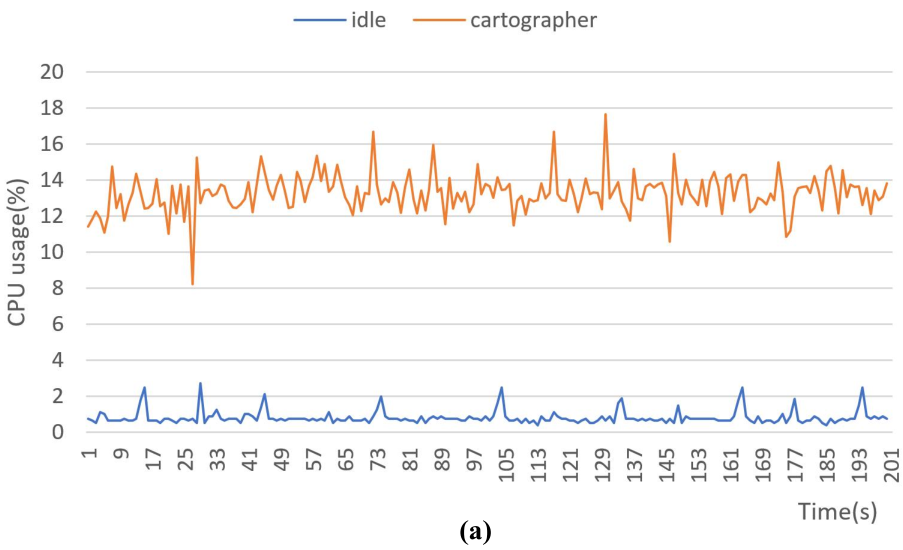
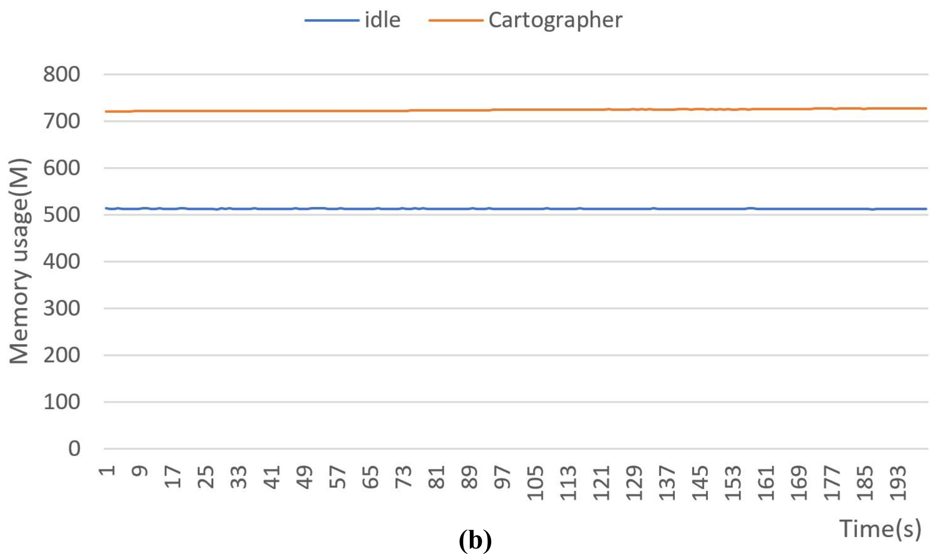

**(a)** **(b)**

Figure 5-7 Cartographer mapping result

where (a) is when the wheel does not slip during normal walking. (b) Put a little oil on the wheels. The result was obtained by allowing the robot to slip while walking. Unlike Gmapping, this slip has a more significant impact. From figure (a), we can see that the map can be successfully built when the wheels do not skid. Once slipping occurs, the map will have errors immediately. According to the mapping principle of the Cartographer, the reason for this error is that Cartographer mainly relies on loop detection to eliminate the drift error generated by the front end. However, the use of lasers for loop detection is complex and challenging. As a result, the submap cannot be inserted into the best position.



**(a)**



**(b)**

Figure 5-8 Cartographer hardware resource consumption. (a) Cartographer algorithm running CPU usage. (b) Cartographer algorithm running memory usage.

Figure 5-8 shows the CPU usage and memory usage of the Raspberry Pi when running cartographer. It can be seen from the figure that the CPU usage rate fluctuates between 12%-18% when running hector. The average utilization rate is 13.25%. It was 12.5% more than when it was idle. The memory usage is about 720M, which is about 220M more than when idle. It can be seen that the average CPU usage rate of running the cartographer algorithm is relatively low. And its memory usage is also low.

### Karto

Karto SLAM is a method based on graph optimization. It uses a highly optimized and non-iterative Cholesky matrix to decouple the sparse system as the solution. The graph optimization method uses the mean value of the graph to represent the map. Each node represents a position point of the robot trajectory and a sensor measurement data set. The connection of the arrow points represents the movement of the consecutive robot position points. When each new node is added, the map will be calculated and updated according to space's node arrows' constraints.

The ROS version of Karto SLAM, in which the Spare Pose Adjustment (SPA) is related to scan matching and closed-loop detection. The more landmarks, the greater the memory requirements. However, the graph optimization method has a more significant advantage than other drawing methods in a large environment. In some cases, KartoSLAM is more effective because it only contains a graph of points (robot pose) to find the position then calculate for map later.

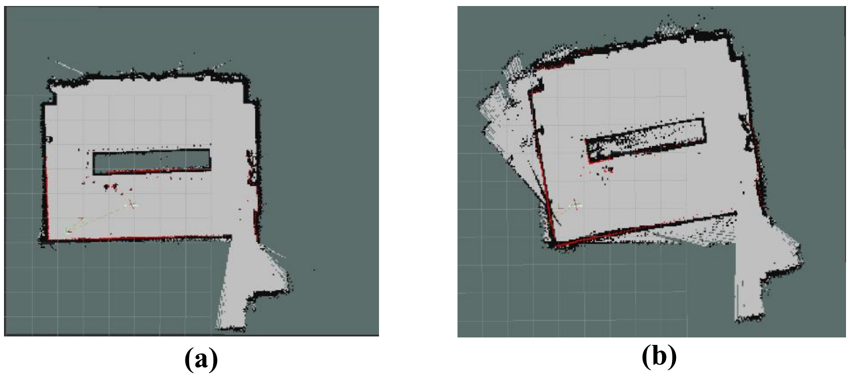
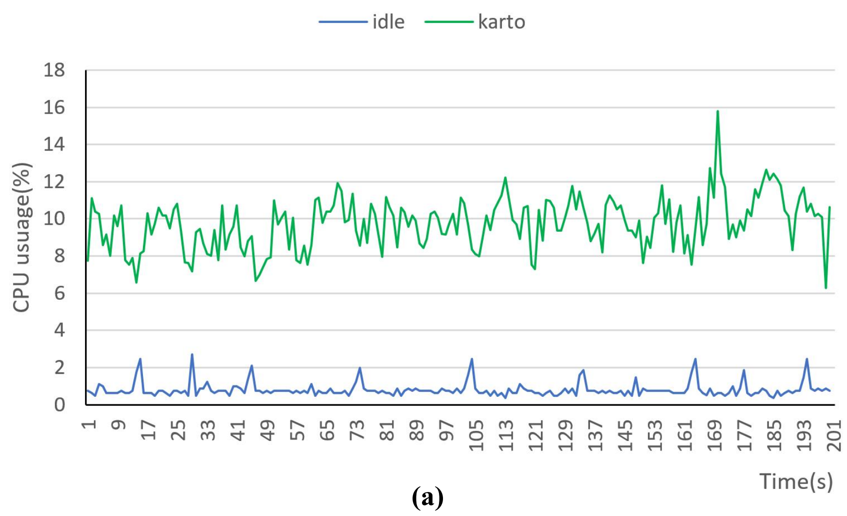
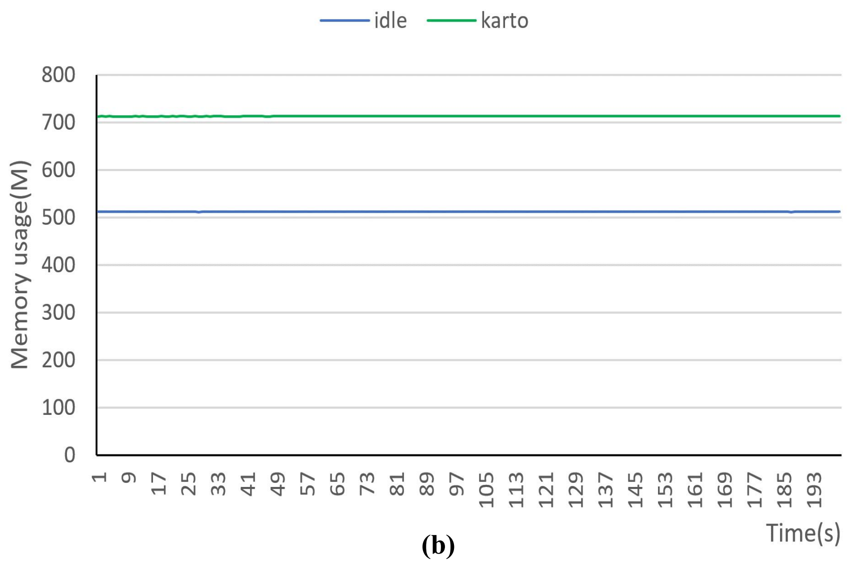

**(a)** **(b)**

Figure 5-9 Karto mapping result

Figure 5-9 is the result of using the karto algorithm. Same as "gmapping", where (a) is the result of the wheels not slipping during normal walking. (b) Put a little oil on the wheels. The result was obtained by allowing the robot to slide while walking. Compare the two results because the input data of the karto algorithm contains wheel odometry. So when the wheels slip, although the established map will drift. However, the establishment of the overall boundary of the interior is not affected.



(a)



(b)

Figure 5-10 Karto hardware resource consumption. (a) Karto algorithm running CPU usage. (b) Karto algorithm running memory usage.

Figure 5-10 shows the CPU usage and memory usage of the Raspberry Pi when running the karto algorithm. It can be seen from the figure that the CPU usage rate fluctuates between 6.2% and 16% when running hector. The average utilization rate is 9.75%. 8.9% more than when idle. The memory usage is about 713M, which is 200M more than when idle. It can be seen that the CPU usage of running the karto algorithm is less than 10%. And the memory overhead is not significant. It is an excellent algorithm. However, the CPU usage rate variance is significant, which means that the CPU usage rate fluctuates wildly when the algorithm is used. It sometimes has a specific impact on the real-time performance of the system.

The above four algorithms can build maps well except hector. Combining the above four algorithms, we averaged each algorithm's resource usage rate to form Table 5-1. The average power of each calculation operation is added to the table. The table shows that the lowest CPU usage rate is Hector. His average CPU usage rate is only 5.58%. But it is through increasing the memory usage rate in exchange for the CPU's low usage rate. The most used CPU is the gmapping algorithm. Its CPU usage rate is as high as 30%. High CPU usage means higher power consumption. So its power consumption is also the highest among all algorithms. Fortunately, its memory usage is at an average level. Synthesize the data in the table and the mapping capabilities of the four algorithms. Karto is the most suitable slam algorithm for disinfecting robots.

**Table 5-1** SLAM algorithm resource consumption

| Algorithm | Average CPU usage(%) | Average memory usage(M) | Average power(w) |
|---|---|---|---|
| idle | 0.7975 | 512.13 | 5.55 |
| cartographer | 13.2504 | 723.56 | 6.57 |
| karto | 9.7504 | 713.86 | 6.39 |
| hector | 5.5787 | 798.53 | 6.40 |
| gmapping | 30.0082 | 737.61 | 7.18 |

## 5.3. Summary

This chapter mainly tests the movement function and mapping function of the disinfection robot. Test the CPU usage and memory usage of the Raspberry Pi when only running the SLAM algorithm. Analyze each algorithm's resource consumption rate and finally choose karto SLAM as the positioning mapping algorithm for the disinfection robot.

# Chapter 6 Conclusion and Outlook

## 6.1. Conclusion

In order to meet market demand, this thesis quickly designed a prototype of a household disinfection robot engineering prototype based on the idea of secondary development on the platform. This thesis constructs a hardware platform for a disinfection robot. Choose cost-effective disinfection equipment in the market as a disinfection method to reduce the cost of robots. Using raspberry pie as a navigation board can handle massive SLAM data processing and control the entire robot's movement and disinfection operations. Use the STMF103 microprocessor as the bottom peripheral control center to directly control the movement of various disinfection modules and motors. This thesis utilizes the ROS framework to develop a robot software system. The communication between robots and Android terminals is achieved through the ROS bridging protocol. Develop Android applications to control robot movements and control disinfection modules on the local area network. This thesis implements the SLAM algorithm on disinfection robots and tests and evaluates four mainstream SLAM algorithms. By evaluating the results, Karto was selected as the robot localization and mapping algorithm.

The robot clearly implements the following functions: it is equipped with three disinfection modules. Ultrasonic atomization, UV lamp, and air purifier. All three modules are functioning properly. It can disinfect the ground and air. This robot has basic SLAM functions. A two-dimensional grid map can be created indoors. And mark your location on the map. Support uploading maps to the terminal for display. This robot can communicate with intelligent terminal devices on the Android platform. Proxy nodes forward terminal messages. Implement communication between terminals and ROS nodes. Robots can be controlled through Android applications.

## 6.2. Outlook

In this thesis, the disinfection robot's SLAM navigation function is realized by using an open-source algorithm. A self-designed control panel is used to control the disinfection module of the disinfection robot. The disinfection robot's basic functions are successfully realized so that the whole disinfection robot can typically solve the indoor disinfection problem. But there are still the following deficiencies.

## Deficiencies

The disinfection robot has not yet achieved automatic path planning function. Positioning and mapping can only be achieved through manual operation. In addition , the overall height of the disinfection robot is 45 centimeters, and the LiDAR is placed at a distance of 15 centimeters from the ground. Lidar can only scan aircraft at a distance of 15 centimeters from the ground. It creates a huge visual blind spot. For obstacles that are not on the scanning plane, the disinfection robot cannot recognize them. Therefore, the disinfection robot is likely to encounter obstacles at heights such as tables. Besides that, there is no corresponding control algorithm for the spraying control of disinfection dosage. Unable to execute the corresponding spraying mode based on the concentration of indoor disinfectant. Similarly, the dosage of ultraviolet radiation is not specified. The disinfection robot runs locally, and its data can only be uploaded to smartphones. It is not yet connected to the external network. The functions of robots can only be controlled through smartphones on the local area network.

## Future Work

I think the following measures can be incorporated into the subsequent work regarding the aforementioned defects.

(1) Add a 3D ToF camera to scan the three-dimensional environment. 3DToF can perform three-dimensional mapping of the surrounding environment. Reduce the visual blind zone of the robot so that it can perceive obstacles at high places and avoid collisions.

(2) Install a corresponding concentration sensor. We can use automatic control algorithms to control the spray module to control the concentration at a reasonable value.

(3) WiFi module is reserved on the control board. In theory, the working state of the robot can be remotely controlled through the server. Simultaneously, the server can also collect and analyze the operational data of the disinfection robot. Provide more intelligent services.

(4) Add mainstream path algorithms such as DWA and TEB to the robot which can allow the robot to perform disinfection operations without human manipulation.

## References

[1] D'Auria D, Persia F. Robots against the Coronavirus: the need for a new generation of robots to help global society[C]//2021 IEEE 15th International Conference on Semantic Computing (ICSC). IEEE, 2021: 421-424.

[2] Chan S H, Wu P T, Fu L C. Robust 2D indoor localization through laser SLAM and visual SLAM fusion[C]//2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC). IEEE, 2018: 1263-1268.

[3] Cadena C, Carlone L, Carrillo H, et al. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age[J]. IEEE Transactions on robotics, 2016, 32(6): 1309-1332.

[4] Quigley M, Conley K, Gerkey B, et al. ROS: an open-source Robot Operating System[C]//ICRA workshop on open source software. 2009, 3(3.2): 5.

[5] Zeng Z, Chen P J, Lew A A. From high-touch to high-tech: COVID-19 drives robotics adoption[J]. Tourism Geographies, 2020, 22(3): 724-734.

[6] Ngo D T, Pham H A. Towards a Framework for SLAM Performance Investigation on Mobile Robots[C]//2020 International Conference on Information and Communication Technology Convergence (ICTC). IEEE, 2020: 110-115.

[7] Yang G Z, Nelson B J, Murphy R R, et al. Combating COVID-19—The role of robotics in managing public health and infectious diseases[J]. 2020.

[8] Murphy R R, Gandudi V B M, Adams J. Applications of robots for COVID-19 response[J]. arXiv preprint arXiv:2008.06976, 2020.

[9] T. R. R. Staff. (2020) How Robotics Helps Diagnose COVID-19 at NewLab. Accessed:Nov.3,2020.[Online].Available:https://www.therobotreport.com/liquid-handling-robot-covid-19-new-lab/

[10] Hu Y, Li J, Chen Y, et al. Design and Control of a Highly Redundant Rigid-Flexible Coupling Robot to Assist the COVID-19 Oropharyngeal-Swab Sampling[J]. IEEE Robotics and Automation Letters, 2021.

[11] ABB_COBOT_YUMI. (2020) ABB’s Collaborative Robot-YuMi-Industrial Robots from ABBRobotics. Accessed: Nov.3, 2020. [Online]. Available: https://new.abb.com/products/robotics/industrial-robots/irb-14000-yumi

[12] ABB_YUMI_ARM. (2020) COVID-19: YuMi, a Collaborative Robot Helps Analyse a Higher Number of Serological Tests. Accessed: Nov.3,2020. [Online].Available:https://www.mynewsdesk.com/it/politecnico-di-milano/pressreleases/covid-19-yumi-a-collaborative-robot-helps-analyse-a-higher-number-of-serological-tests-3034336

[13] Salathé M, Althaus C L, Neher R, et al. COVID-19 epidemic in Switzerland: on the importance of testing, contact tracing and isolation[J]. Swiss medical weekly, 2020, 150(1112).

[14] Brotherhood L, Kircher P, Santos C, et al. An economic model of the Covid-19 epidemic: The importance of testing and age-specific policies[J]. 2020.

[15] Binnicker M J. Emergence of a novel coronavirus disease (COVID-19) and the importance of diagnostic testing: why partnership between clinical laboratories, public health agencies, and industry is essential to control the outbreak[J]. Clinical chemistry, 2020, 66(5): 664-666.

[16] Li S Q, Guo W L, Liu H, et al. Clinical application of an intelligent oropharyngeal swab robot: implication for the COVID-19 pandemic[J]. European Respiratory Journal, 2020, 56(2).

[17] Shen Y, Guo D, Long F, et al. Robots under COVID-19 Pandemic: A Comprehensive Survey[J]. IEEE Access, 2020.

[18] L. Robotics. Lifeline Robotics Swab Robot. Accessed: Nov.3,2020. [Online]. Available: https://www.lifelinerobotics.com/#careebo-llr-s1

[19] UV Light Might Keep the World Safe From the Coronavirus—And Whatever Comes Next–IEEE Spectrum. Accessed: Nov.4,2020. [Online].Available: https://spectrum.ieee.org/biomedical/devices/uv-light-might-keep-the-world-safe-from-the-coronavirusand-whatever-comes-next

[20] E. Ackerman, (2020)Akara Robotics Turns TurtleBot Into Autonomous UV Disinfecting Robot. IEEE Spectrum. Accessed: Nov.4, 2020. [online] Available:https://spectrum.ieee.org/automaton/robotics/medical-robots/akara-robotics-turtlebot-autonomous-uv-disinfecting-robot.

[21] Cao Y, Chen W, Li M, et al. Simulation Based Design of Deep Ultraviolet LED Array Module Used in Virus Disinfection[C]//2020 21st International Conference on Electronic Packaging Technology (ICEPT). IEEE, 2020: 1-4.

[22] Begić A. Application of service robots for disinfection in medical institutions[C]//International Symposium on Innovative and Interdisciplinary Applications of Advanced Technologies. Springer, Cham, 2017: 1056-1065.

[23] Casini B, Tuvo B, Cristina M L, et al. Evaluation of an ultraviolet C (UVC) light-emitting device for disinfection of high touch surfaces in hospital critical areas[J]. International journal of environmental research and public health, 2019, 16(19): 3572.

[24] Nottingham M, Peterson G, Doern C, et al. Ultraviolet-C light as a means of disinfecting anesthesia workstations[J]. American journal of infection control, 2017, 45(9): 1011-1013.

[25] McGinn C, Scott R, Donnelly N, et al. Exploring the Applicability of Robot-Assisted UV Disinfection in Radiology[J]. Frontiers in Robotics and AI, 2020, 7.

[26] Casini B, Tuvo B, Cristina M L, et al. Evaluation of an ultraviolet C (UVC) light-emitting device for

disinfection of high touch surfaces in hospital critical areas[J]. International journal of environmental research and public health, 2019, 16(19): 3572.

[27] Pang Hongbing, Ding Xinze, Wang Wanxu. Study on the influence of different disinfection methods on site DNA detection[J]. Chinese Journal of Forensic Medicine, 2021, 36(01): 78-81.

[28] R. Naji. (2020) Dubai Is Using Drones to Disinfect All The City. Accessed: Nov.4, 2020. [Online]. Available: https://www.the961.com/dubai-is-using-drones-to-disinfect-all-the-city

[29] F. Patterson. (2020). Buffalo Startup Repurposes Drones for Sanitation. [Online].Available:https://spectrumlocalnews.com/nys/buffalo/news/2020/05/05/buffalo-start up-re-purposes-drones-for-sanitation-to-fight-covid-19

[30] Drone Disinfection System Kills Coronavirus In Large Indoor Spaces. Accessed:Nov.4,2020.[Online].Available:https://facilityexecutive.com/2020/04/drone-disinfection-system-kills-coronavirus-in-large-indoor-spaces/

[31] J. Zimmer. (2020). Fighting COVID-19 With Disinfecting Drones and Thermal Sensors. Accessed: Nov.4, 2020. [Online]. Available: https:// www.engineering.com/DesignerEdge/DesignerEdgeThesiss/ThesisID/20324/Fighting-COVID-19-with-Disinfecting-Drones-and-Thermal-Sensors.aspx

[32] Ren Feng, Liu Taiqi. Research and application of purification materials for filtration and sterilization [J]. Chemical World, 2007(02): 121-124.

[33] Introduction to the disinfection robot of Qinglang Co., Ltd. Accessed: Nov.4, 2020. [Online]. Available: http://b.keenonrobot.com/xd.html?baidu_103204

[34] Isolation Ward Disinfection Solution. (2020) TMiRob Co., Ltd. Accessed: Nov.4, 2020. [Online]. Available: http://www.tmirob.com/solutions/11

[35] Commercial disinfection and cleaning robot [Aikebao] Ecobot Scrubber 50A o.(2020)Gaussian Robotic Co., Ltd. Accessed: Nov.4, 2020. [Online]. Available: http://www.gs-robot.com/products/ecobot/ecobotVirus

[36] Athena. (2020) General-purpose robot development platform. Accessed: Nov.4, 2020. [Online]. Available: http://www.slamtec.com/cn/Athena

[37] General Intelligent Mobile Platform. (2020) Yunji Technology Co., Ltd. Accessed:Nov.4,2020.[Online].Available:https://www.yunjichina.com.cn/a/52.html

[38] Agilex Ros Education Development Kit. (2020) Agilex Technology Co., Ltd. Accessed:Nov.4,2020.[Online].Available:https://translate.google.cn/?sl=auto&tl=en&text=Agilex%20Ros%E6%95%99%E8%82%B2%E5%BC%80%E5%8F%91%E5%A5%97%E8%A3%85%0A&op=translate

[39] Elkady A, Sobh T. Robotics middleware: A comprehensive literature survey and attribute-based bibliography[J]. Journal of Robotics, 2012, 2012.

[40] Gu Y, Liu J, Wu W. Integrated Product and Process Development Mode Based on Models Coupling[C]//2006 6th World Congress on Intelligent Control and Automation. IEEE, 2006, 2: 6744-6747.

[41] Jose A, Tollenaere M. Modular and platform methods for product family design: literature analysis[J]. Journal of Intelligent manufacturing, 2005, 16(3): 371-390.

[42] Cao G, Zhang Y. Microbial Removal Efficiency in situ Measurement of Exhaust Air HEPA Filters Employed in Cleanrooms[C]//2009 3rd International Conference on Bioinformatics and Biomedical Engineering. IEEE, 2009: 1-4.

[43] Royal Philips Electronics Co., Ltd. of the Netherlands. Germicidal lamp datasheet[Z]. Netherlands: Royal Philips Electronics Co., Ltd. of the Netherlands, May 2011.

[44] Guangzhou Guangfeng Tongda Electronics Co., Ltd. KFFFB12025 data sheet[Z]. Panyu District, Guangzhou City, Guangdong Province: Guangzhou Guangfeng Tongda Electronics Co., Ltd., 2018.

[45] Park Y, Choi J, Choi J. A system architecture to control robot through the acquisition of sensory data in IoT environments[C]//2016 13th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI). IEEE, 2016: 749-752.

[46] Muratore L, Laurenzi A, Hoffman E M, et al. Xbotcore: A real-time cross-robot software platform[C]//2017 First IEEE International Conference on Robotic Computing (IRC). IEEE, 2017: 77-80.

[47] HuC,HeDetal. A new ROS -based hybrid architecture for heterogeneousmulti-robot systems[C]//Control and Decision Conference (CCDC), 2015 27th Chinese.IEEE, 2015: 4721-4726.

[48] Dissanayake M. W. M. G, Newman P, Clark S, et al. A Solution to the Simultaneous Localization and Map Building (SLAM) Problem[J]. IEEE Transactions on Robotics and Automation, 2001, 17(3): 229-241.

[49] Ma Yan, Ju Hehua, Cui Pingyuan. Research on Localization and Mapping for Lunar Rover Based on RBPF-SLAM[A]. International Conference on Intelligent Human-Machine Systems and Cybernetics, 2009. IHMSC '09[C]. Hangzhou, Zhejiang:IEEE, 2009: 306-311.

[50] R. Casas, D. Cuartielles, A. Marco, et al. Hidden Issues in Deploying an Indoor Location System[J]. Pervasive Computing, IEEE, 2007,6(2):62-69.

[51] Khalajmehrabadi A, Gatsis N, Akopian D. Modern WLAN fingerprinting indoor positioning methods and deployment challenges[J]. IEEE Communications Surveys & Tutorials, 2017, 19(3): 1974-2002.

[52] Chao Zhuoran. A Design and Implementation of Domestic Disinfection Robot Based on 2D Lidar[D]. Fudan University: Fudan University Library, 2020.

[53] Dai Kai, Shen Qiren, Wu Guanpu, Zhao Yang, Ma Fangwu. Overview of SLAM and fusion positioning methods based on lidar[J]. Car Digest, 2021(02):1-8.

[54] Huang Rui, Zhang Yi. Highly adaptable lidar SLAM[J]. Journal of University of Electronic Science

and Technology of China, 2021, 50(01): 52-58.

[55] Lee D, Kim D, Lee S, et al. Experiments on localization of an AUV using graph-based SLAM[C]//2013 10th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI). IEEE, 2013: 526-527.

[56] Hidalgo F, Bräunl T. Review of underwater SLAM techniques[C]//2015 6th International Conference on Automation, Robotics and Applications (ICARA). IEEE, 2015: 306-311.

[57] Rahmati M, Karten S, Pompili D. Slam-based underwater adaptive sampling using autonomous vehicles[C]//OCEANS 2018 MTS/IEEE Charleston. IEEE, 2018: 1-7.

[58] Dissanayake M W M G, Newman P, Clark S, et al. A solution to the simultaneous localization and map building (SLAM) problem[J]. IEEE Transactions on robotics and automation, 2001, 17(3): 229-241.