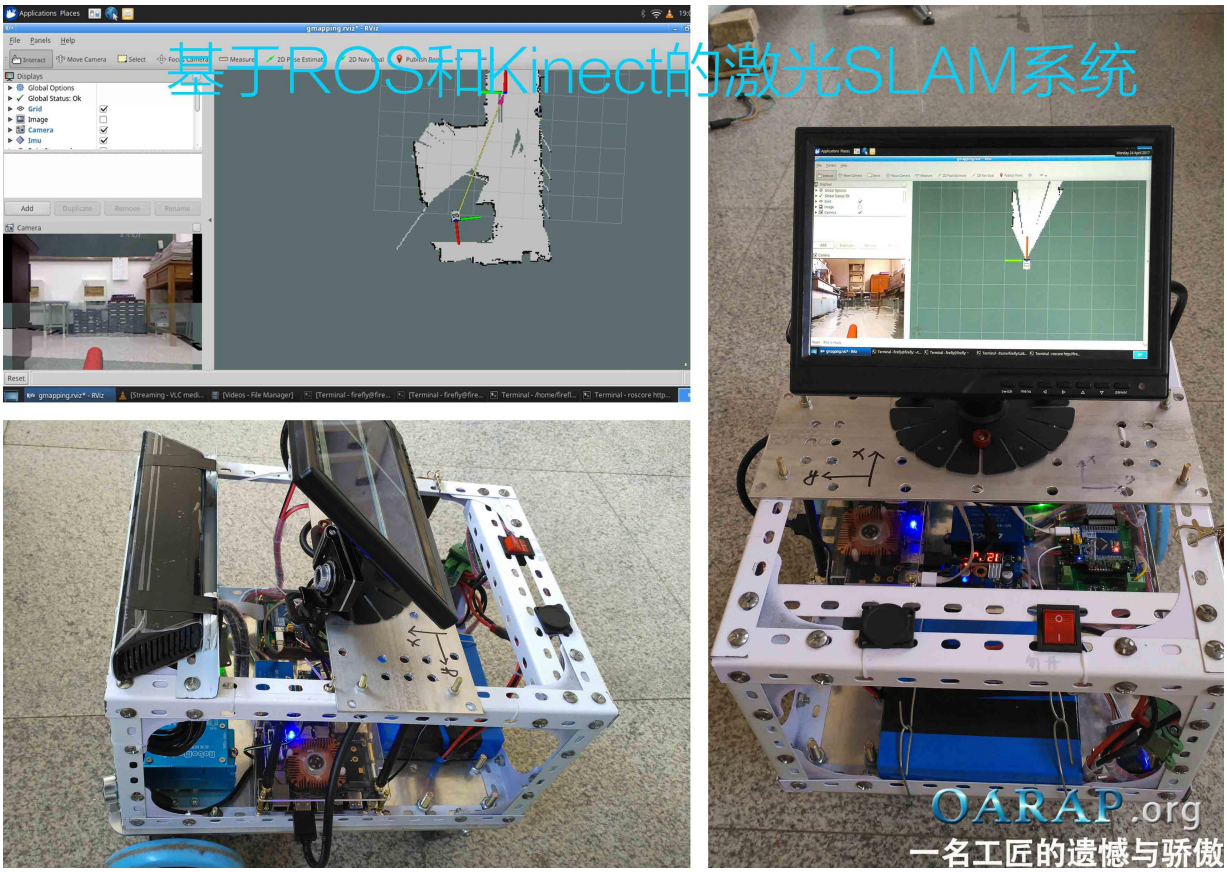
基于 ROS 和 RGB-D 传感器的 SLAM 智能机器人



博客文章：http://oarap.org/archives/1858

代码：https://github.com/oarap-org（正逐步整理）

视频展示：http://player.youku.com/player.php/sid/XMjcyMDE1Mjk3Ng==/v.swf

本文是作者的本科毕业论文，主要围绕现代智能机器人技术展开的实验和研究。权当抛砖引玉，希望本文的一点小尝试能给后人带来一些启发。作者学术水平有限，如有错误还望各位学者批评指正，作者邮箱：admin@oarap.org。

尽管在文末有一章致谢的内容，但在此我依然要对两个团队表达谢意：创客智造，该网站发布了许多机器人系列教程和指南，同时站长在设计、调试过程中提出大量的建设性意见让笔者受益匪浅；广东中山 Firefly 开源团队，该团队在笔者硬件资源不足的情况下免费提供了 RK-3399 开发板，同时提供了技术支持，才使设计工作得以继续。

本资料上传至互联网供学者免费下载，版权归 OARAP.org 所有。如需转载请注明所有者。

- 作者博客：http://oarap.org
- 项目集合：http://oarap.org/project
- 关于我：http://oarap.org/about-donate

©2017 一名工匠的遗憾与骄傲_OARAP.org

OARAP.org

一名工匠的遗憾与骄傲

One Artisan's Regret And Pride

中文题目：基于 ROS 和 RGB-D 传感器的 SLAM 智能机器人

外文题目：SLAM INTELLIGENT ROBOT BASED ON ROS AND RGB-D SENSORS

毕业设计（论文）共 72 页 图纸共 1 张

完成日期 2017 年 6 月　　答辩日期 2017 年 6 月

摘要

随着人工智能和控制技术的突飞猛进，机器人领域发生着日新月异的变化，机器人在人类生产生活中占据越来越重要的地位，智能机器人领域也逐渐被人们所熟知，诸如无人驾驶汽车、无人飞行器、AGV(智能搬运车)、扫地机器人、服务机器人等产品都逐渐走进人们的视野中。然而目前的智能机器人技术还十分不成熟，其中以定位技术为代表和包括地图构建、导航等技术在内的技术难题也逐渐浮现出来。

本文重点研究并实现了以下部分：第一部分：智能移动机器人的机械设计，从选材、结构、动力等方面介绍设计思路；第二部分：智能机器人理论分析，介绍了机器人运动学模型、航迹推算、SLAM、主动定位和路径规划的核心算法；第三部分：硬件设计：阐述了上下位机系统的硬件设计思路；第四部分：软件设计，阐述上下位机的软件系统设计思路，以及调试方法；第五部分，通用性设计，阐述了移动机器人底盘的软硬件通用性设计；第六部分，实验结果与分析，阐述了本设计中的智能机器人实验效果与实验分析。

从最初设计机械结构、组装底盘，到绘制 PCB、编写嵌入式程序，再到搭建上位机开发环境、编写上位机应用层算法，本设计实现了一个基于 ROS 的具有 SLAM、主动定位、路径规划等功能的移动机器人，包括一个可方便接入上位机的通用可扩展的移动底盘。

关键词：SLAM；定位；导航；机器人；ROS；Kinect

Abstract

Along with the rapid development of AI and control technology, the rapid changes in robot field, robots are more and more important role in human production and living, intelligent robot area is gradually known by people, such as driverless cars, unmanned aerial vehicles, AGV, sweep robot, service robot, and other products have gradually entered people's field of vision.However, intelligent robot technology is still very immature, which represented by positioning technology and including built figure, the technology of technical problems also gradually emerged.

This paper studies and implements the following parts: the first part: the mechanical design of the intelligent mobile robot, from material selection, structure, dynamic design were introduced from the aspects such as;The analysis of the second part: the theory of intelligent robot, the robot kinematics model, the core algorithm of DR, SLAM, active localization and path planning;The third part: the hardware design, this paper expounds the upper computer and lower computer system hardware design;The fourth part: the software design, this paper introduces the software system of upper computer and lower computer design, and debugging methods;The fifth part, the general design, expounds the hardware and software design of commonality of mobile robot chassis;The sixth part, the experimental results and analysis, this paper expounds the design of this intelligent robot experimental results and experimental analysis.

From the original design of the mechanical structure, the assembly chassis, draw the PCB, embedded programming, build development environment on the upper computer,write the algorithms for the upper computer application layer, this design implements a SLAM, active intelligent functions such as localization, path planning of mobile robot based on ROS, including a convenient access to PC common extensible mobile chassis.

Key words: SLAM; localization; navigation; robotics; ROS; Kinect

目录

摘要..... I

Abstract..... II

1 绪论.....1

1.1 选题的目的和意义..... 1

1.2 国内外研究现状..... 1

1.3 研究的内容及预期..... 2

2 机械设计.....3

2.1 机器人底盘结构..... 3

2.2 机械结构.....4

2.3 动力机构.....5

2.3.1 传动方式.....5

2.3.2 电动机.....6

3 理论分析.....8

3.1 机器人运动学模型..... 8

3.2 航迹推算.....9

3.3 SLAM.....10

3.4 路径规划.....12

4 硬件设计.....13

4.1 硬件系统概述..... 13

4.2 下位机典型硬件设计..... 14

4.2.1 微控制器.....14
4.2.2 蓝牙模块.....16
4.2.3 旋转编码器模块.....17
4.2.4 超声波模块.....18
4.2.5 惯性测量模块.....20
4.2.6 RS485 收发器.....20
4.2.7 电机驱动器.....21
4.2.8 无线接收模块.....23
4.2.9 电池电压采样.....24
4.2.10 稳压器.....24
4.2.11 电池.....25
4.3 PCB 设计.....26
4.4 上位机典型硬件设计.....28
4.4.1 计算平台.....28
4.4.2 传感器.....30
5 软件设计.....32
5.1 软件设计概述.....32
5.2 下位机典型软件设计.....33
5.2.1 下位机主程序.....33
5.2.2 通信模块软件.....34
5.2.3 惯性单元和驱动器.....35
5.2.4 多通道超声波.....38

5.2.5 无线接收模块 40
5.2.6 ADC 单元 40
5.3 上位机典型软件设计 41
5.3.1 操作系统 41
5.3.2 导航功能包集 44
5.3.3 基控制器 45
5.3.4 位姿滤波器 48
5.3.5 3D 模型描述 49
5.3.6 激光数据 50
5.3.7 SLAM 与导航 51
5.3.8 典型启动文件 54
5.4 通信格式设计 54
6 通用性设计 56
7 实验过程及分析 57
7.1 调试工具 57
7.1.1 下位机 57
7.1.2 上位机 57
7.2 SLAM 59
7.3 导航 61
8 技术经济分析 63
9 结论 64
致谢 65

参考文献 66
附录 A 通信格式定义.....67
附录 B 底盘尺寸图.....68
附录 C TF 树..... 69
附录 D SLAM 过程中节点关系..... 70
附录 E 导航过程中节点关系.....71
附录 F 下位机电路原理图..... 72

1 绪论

1.1 选题的目的和意义

随着科技发展水平的日益增长和生产生活自动化程度的提高，近年来仓储系统、家居系统、自动驾驶机器人、无人机、AR/VR、智能硬件等领域的无人化、智能化程度越来越高，人类社会对智能化移动机器人的需求越来越高。移动机器人应用广泛，在工业、医疗、生活、军事等方向都有机器人发挥着重要的作用。

而移动机器人这些场景中都需要解决的问题主要有以下三个：机器人要知道我在哪？——定位；周围环境是什么样？——建图；我要怎么去目标地点？——导航。可以看为是感知的“内外之分”，智能机器人一方面要知道自身状态（即位置），另一方面也要知道外在环境（即建图），当然解决这两个问题的方法有很多，比如可以使用 GNSS 系统定位，可以在地板铺设引导线、在墙壁贴上识别二维码、在桌子上放置无线定位设备等[1]，常把这些方法称为直接方法，受噪声影响较小，但很明显这些方法的缺点是需要提前对环境加以改造，而且机器人无法在室内搜索到 GNSS 系统的信号。

更多时候需要机器人到了陌生的环境自己就能解决这些问题，因此利用机器人的内部传感器，如激光传感器、视觉传感器、IMU 传感器、里程计等来定位、建图、导航等就显得十分有必要了。最近几年的同步建图定位技术成为了主流的机器人导航定位方案，因此展开对移动机器人 SLAM、导航、主动定位技术的深入研究将有着多方面的重要意义。

1.2 国内外研究现状

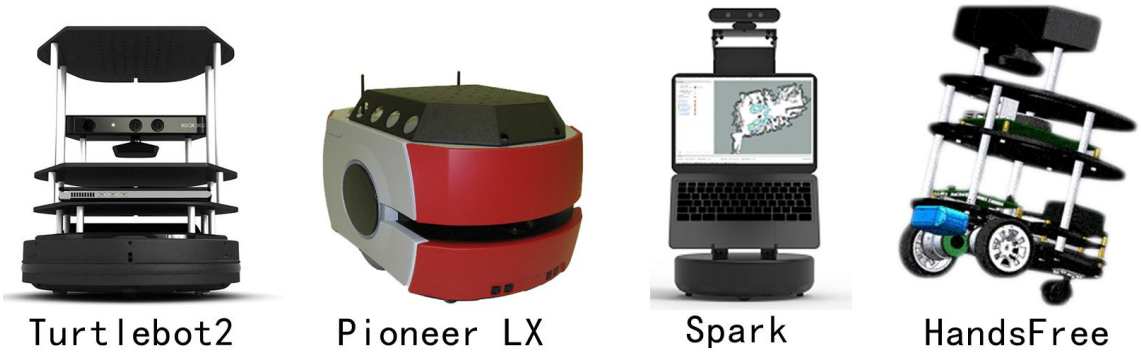


图 1-1 常见智能机器人底盘

Fig.1-1 Common intelligent robot chassis

近年来智能机器人发展迅速，如图 1-1 所示，许多机器人公司开始推出智能轮式机器人开发底盘，国外主要 Turtlebot2、Pioneer LX，其中 Turtlebot 是全球最受学者欢迎的移动轮式机器人底盘，国内的 NXROBO 公司刚刚开始发售（2017 年 6 月）的 Spark 机器人、HandsFree 团队的 mini 移动机器人等。昂贵的价格和复杂的购买流程让很多学者们望而却步，而且一些优秀的底层软硬件并非全部开源，所以成熟的通信机制和嵌入式控制器以及机械结构的设计都还有必要进一步挖掘。

而在软件层面，SLAM 问题自从 1988 年提出，至今为止已经近三十年。早期 SLAM 研究侧重于使用滤波器理论，最小化运动体位姿和地图的路标点的噪声。21 世纪之后，学者们开始借鉴 SfM(Structure from Motion)中的方式，以优化理论为基础求解 SLAM 问题。根据传感器种类和安装方式的不同，SLAM 的实现方式和难度会有很大差异。按传感器来分，SLAM 主要分为激光、视觉两大类，激光传感器可以直接获得相对于环境的直接距离信息，从而实现直接相对定位，对于激光传感器的绝对定位及轨迹优化可以在相对定位的基础上进行；视觉传感器很难直接获得相对于环境的直接距离信息，而必须通过两帧或多帧图像来估计自身的位姿变化，再通过累积位姿变化计算当前位置。在近年来智能机器人领域关注的人数快速增多，目前已经有数种开源 SLAM 算法可供学习和参考[2]。其中激光 SLAM 研究时间久，理论相对比较成熟。视觉方案目前尚处于实验室研究阶段，市面上几乎看不到实际产品应用。近年来移动计算和移动互联网的快速发展，让移动端的低成本、高性能的计算和通信网络成为现实，让以前迫于计算性能和通信成本的智能机器人有了广阔的发展空间。

1.3 研究的内容及预期

本论文的主要研究内容，智能轮式机器人是一个集周边环境感知、动态决策和路径规划、行为控制与动作执行等多功能于一体的综合性系统，通常由机械结构、控制系统、传感器、电源、计算平台等组成，其研究领域涉及图像处理、机器视觉、传感器技术、人工智能、通信技术、机械设计、数学分析等多种学科理论与技术，本文将主要围绕移动机器人的机械结构、嵌入式软硬件系统、下位机与上位机间通信格式、机器人建图定位路径规划等方面来研究。

本设计预期功能：通用可移动底盘：包括可裁减、可扩展的通用下位机软硬件系统，以及可裁剪、可扩展的通用的通信格式；未知环境中实现移动机器人同步定位与建图；已知栅格地图的情况下主动定位；对去向某个目标位置能够进行路径规划并支持壁障。

2 机械设计

2.1 机器人底盘结构

表 2-1 常见移动机器人底盘对比

Tab.2-1 Common mobile robot chassis comparison

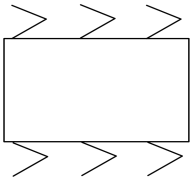
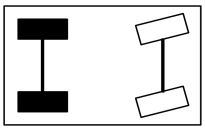
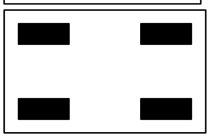
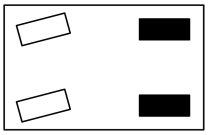
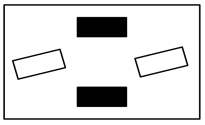
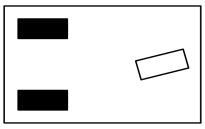
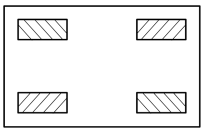
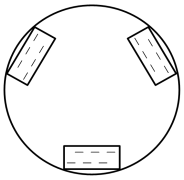
底盘模型	类型	描述	稳定性	机动性	可控性
	腿足式机器人	多足	高	高	高
	轮式机器人	履带式	较高	较高	较低
	轮式机器人	两个驱动轮，舵转向结构，前后均有差速器	较高	中	较低
	轮式机器人	四轮独立驱动，无差速器	较高	中	较低
	轮式机器人	两个独立驱动轮，后面两个小脚轮，无差速器	较高	中	中
	轮式机器人	两个独立驱动轮，前后各一个小脚轮，无差速器	较低	较高	中
	轮式机器人	两个独立驱动轮轮，前面一个小脚轮，无差速器	中	中	中
	轮式机器人	四个麦克纳姆轮，全向底盘	低	高	低
	轮式机器人	三个互成 60 度的全向轮，全向底盘	低	高	低

表 2-1 即为常见的移动机器人移动底盘模型的对比，其中底盘模型里：黑色填充代

表动力轮，透明填充代表无动力轮，矩形代表标准轮，圆形代表小脚轮，斜条纹代表麦克纳姆轮，纵条纹代表全向轮。

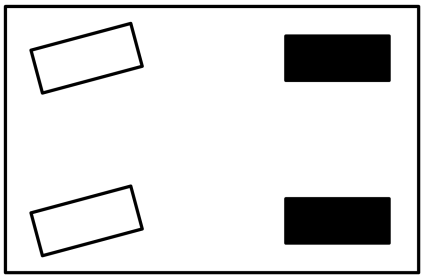


图 2-1 本设计中机器人底盘运动模型

Fig.2-1 In the design of the robot motion model of the chassis

图 2-1 中所示的模型为本设计使用的方案，底盘共连接四个轮子，其中前面为两个为独立驱动轮，后面两个为独立的小脚轮，前面两个轮子通过差速的方式改变机器人的运动方式和状态。差速运动的底盘数学模型要比全向底盘、多足底盘、带差速器的底盘要简单，实际的可控性好过由全向轮或麦克纳姆轮构成的全向底盘，机动性相比带差速器底盘要好，全向的底盘对地面光泽度要求较高，而履带式和腿足式机器人优秀的越野能力又无用武之地，所以本设计选用了前面两个独立驱动轮，后面两个小脚轮这种方案[3]。

2.2 机械结构

为保证刚度的情况下减轻重量，本设计中的轮式机器人底盘使用 4mm 厚，面积为 415mm*315mm 的铝合金板，支架使用了高强度轻质镂空的角钢，其上面的大面积规则镂空孔洞可以更方便调节高度、增减模块等，提高了移动机器人机械上的可扩展性。

本设计中的机器人可分为上下两，上层有电源开关、充电接口、显示器等便于开发者操作和查看，甚至还可以放置一把键盘，在调试程序的时候跟方便快捷。将机器人分层设计后，缩小了机器人的占地面积，灵活性和舒适度上都相比单层式结构有了提升。

设计中的驱动轮外径 150mm，轮宽 24mm，胎面是优质 PU 橡胶，保证了在与地面接触时足够的摩擦力。万向轮外径 100mm，轮宽 32mm，胎面材质为橡胶，具有防滑、耐磨的特性。

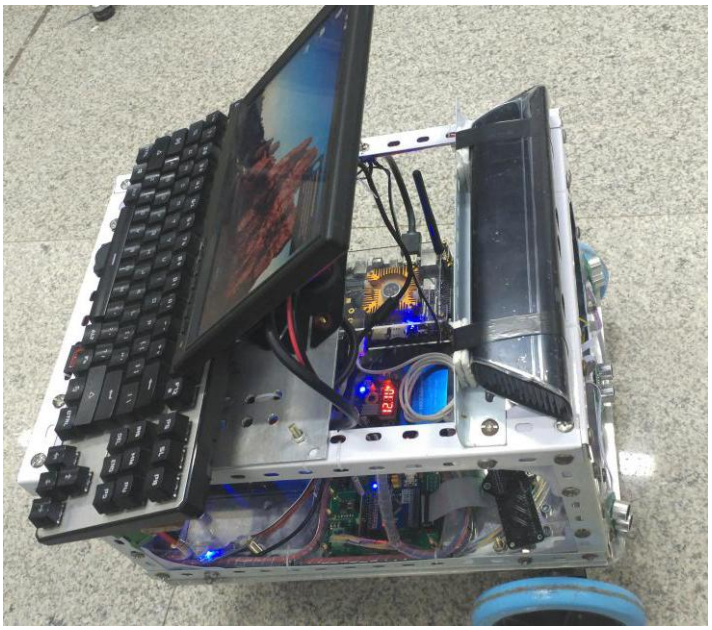


图 2-2 机器人实物图

Fig.2-2 The robot picture of real products

如图 2-2 所示为本设计中移动机器人实物图，底板尺寸图详见文末附录 B。

2.3 动力机构

2.3.1 传动方式

表 2-2 常见传动方式对比

Tab.2-2 Common drive for contrast

	电机直连	齿轮	同步带
承重	小	大	大
移动速度	高	中	高
安装结构	小	要求预留间隙	大
价格	低	高	中
制造	较难	容易	选购选准件
定位精度	高	中	低
磨损	无需调整	需补偿	需调整
缺点	承重对电机影响大	加工精度差、磨损大	传动长度有限

如表 2-2 所示为常见传动方式的对比分析，结合表中多种因素主要包括精度和承重

方面，本设计中的传动系统使用的同步带传动方式。

2.3.2 电动机

表 2-3 有刷直流电机与无刷直流电机

Tab.2-3 Brush dc motor and brushless dc motor

	性能	寿命	控制器	成本
有刷直流电机	一般	一般	简单	低
无刷直流电机	高	长	复杂	高

如表 2-3 所示，直流电机虽然在性能和寿命上不及无刷电机，但是由于其控制器和控制算法相对简单，成本也相对较低，故本设计使用了有刷的直流电机。

本设计中驱动轮的外径为 150mm，根据经验在日常的生产生活中的机器人几乎不会超过 1m/s，在采用同步带传动的情况下，电机转速的计算公式如公式(1-1)：

$$\mathrm{n}=\frac{v}{\pi \mathrm{d}}=\frac{1}{0.15\pi}=2.123rps=127.39rpm \tag{1-1}$$

所以需要电机的转速在 127.39rpm 以上。

另外本设计的移动机器人整车重量大概在 10KG 左右，经查阅资料，橡胶轮与大理石地面间的滚动摩擦系数为大概为 0.2 左右，在机器人的最高移动速度为 1m/s 的情况下，所以滚动摩擦力计算如公式(1-2)：

$$M=F_{\mathrm{n}}\times\sigma=\mathrm{m}\times g\times\sigma=10\times9.8\times0.2=19.6N \tag{1-2}$$

只要保证整车的牵引力大于滚动摩擦力：

$$F_{牵}\geq M \tag{1-3}$$

整车功率计算方法如公式(1-4)：

$$P=F_{牵}/v=19.6/1=19.6W \tag{1-4}$$

所以单个电机功率至少要 9.8w 以上。

根据以上的计算结果，本设计中的电动机选用了日本 CITIZEN 公司生产的 RB35P-30NE-OEK36D 型号的有刷直流减速电机，参数表 2-4 所示：

表 2-4 RB35P-30NE-OEK36D 型电机性能参数

Tab.2-4 RB35P-30NE-OEK36D type motor performance parameters

电压	DC 24V
功率	12W
转速	240 转 / 分钟
机身直径	36mm，(长 44mm)
减速箱直径	37mm，(长 22.5mm)
出轴直径	6mm，(长 19mm)
机身总长	95mm

根据上述公式和参数可以推算出小车最快的前进速度能达到 1.884m/s，最多也能驱动 12.24kg 重量的车体，所以该电机在速度和功率冗余上都满足设计需求。

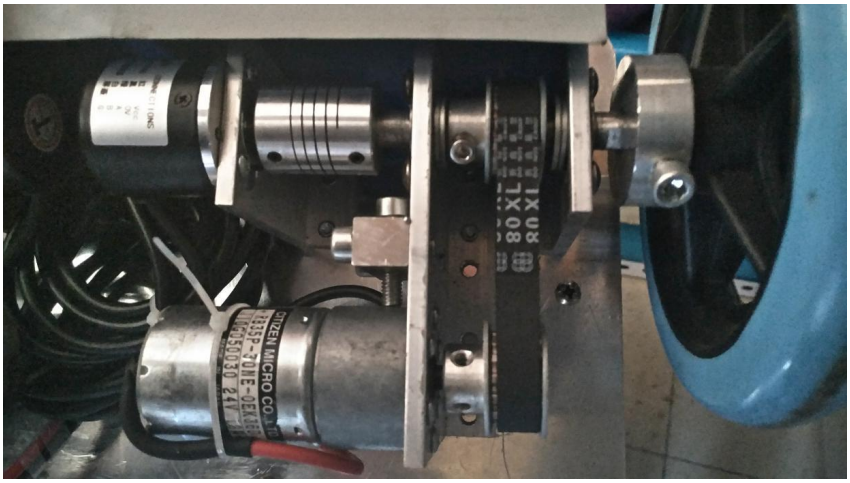


图 2-3 动力机构实物图

Fig.2-3 Power unit physical figure

图 2-3 为本设计中底盘的动力机构实物图，最上方为动力机构的保护板，减少异物对动力机构的伤害，左上方为旋转编码器，左下方为 RB35P-30NE-OEK36D 型直流有刷电动机，右侧为机器人的驱动轮，皆使用法兰固定在底盘的铝合金板上，编码器通过联轴器和轮轴相连，同时轮轴上固定有一个同步轮，电机轴上也装有一个同步轮，二者通过同步带相连接。

3 理论分析

3.1 机器人运动学模型

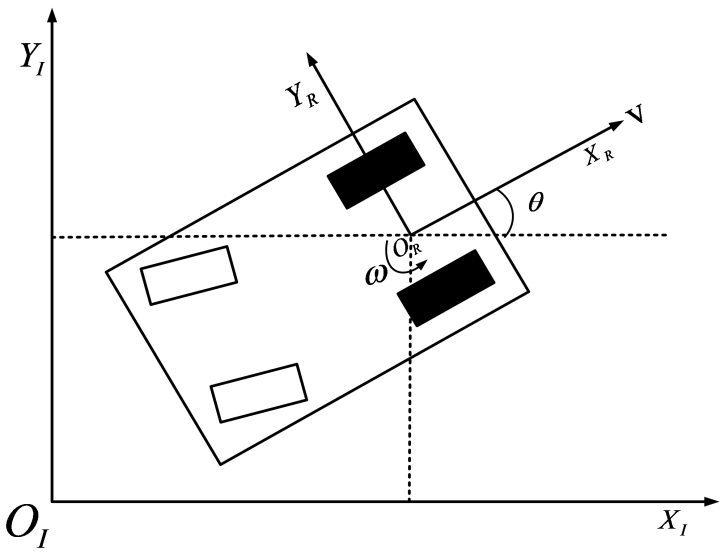


图 3-1 移动机器人运动模型

Fig.3-1 Motion models of mobile robot

已知机器人的位置可以通过正交旋转矩阵表示，如图 3-1 所示，θ 为机器人的姿态角，即机器人纵轴方向相对于 X 轴的夹角，$R(\theta)$ 为正交旋转矩阵，ξ_R 为沿全局参考框架的运动，ξ_I 为沿局部参考框架的运动，ξ_R 与 ξ_I 之间的关系如公式(3-1)所示[4]：

$$\xi_R = R(\theta)\xi_I = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \mathrm{x} \\ y \\ \theta \end{bmatrix} \tag{3-1}$$

对于由两轮差动的移动机器人的特性可以得到局部参考框架下的移动机器人的运动方程，如公式(3-2)所示，其中 l 为两驱动轮间的长度，r 为驱动轮的半径，φ_1 和 φ_2 分别表示左右两个驱动轮的旋转角速度。

$$\xi_I = \begin{bmatrix} \mathrm{x} \\ y \\ \theta \end{bmatrix} = f(l, r, \theta, \varphi_1, \varphi_2) \tag{3-2}$$

如公式(3-3)和公式(3-4)所示，在全局参考框架中差速驱动的机器人，两个驱动轮在

x_R 上的贡献分别为：

$$x_{R1} = \frac{1}{2} r\varphi_1 \tag{3-3}$$

$$x_{R2} = \frac{1}{2} r\varphi_2 \tag{3-4}$$

在本设计中的机器人没有一个驱动轮可以提供侧向运动,所以 $y_R = 0$；

两个驱动轮在 θ 上的贡献分别如公式(3-5)和公式(3-6)所示：

$$\omega_1 = \frac{r\varphi_1}{2l} \tag{3-5}$$

$$\omega_1 = -\frac{r\varphi_1}{2l} \tag{3-6}$$

联合方程，得到本设计中移动机器人的运动学模型如公式(3-7)：

$$\xi_I = R(\theta)^{-1}\xi_R = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{r\varphi_1}{2} + \frac{r\varphi_2}{2} \\ 0 \\ \frac{r\varphi_1}{2l} - \frac{r\varphi_2}{2l} \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0.0375(\varphi_1 + \varphi_2) \\ 0 \\ \frac{1}{12}(\varphi_1 - \varphi_2) \end{bmatrix} \tag{3-7}$$

3.2 航迹推算

航迹推算利用载体上一时刻的位置，根据航向和速度信息，推算得到当前时刻的位置，即根据实测的机器人行驶距离和航向计算其位置和行驶轨迹[5]，一般受外界环境影响（路面平坦、轮胎摩擦力足够大的情况下），但是由于其本身的测量误差是随时间累积的，所以单独工作时不能长时间保证高精度。这种方法在可以作为机器人运动里程信息的短时参考，由于误差累计的原因无法作为长时参考（视觉里程计的回环检测功能会在最终进行全局优化，能一定程度上解决这个问题），航迹推算方法可以认为是位移向量的不断累加，在每一个采样周期内，机器人的位姿估计取决于以前的计算周期，所以导致了误差的不断累积。

根据上一节中已经得到的机器人运动学模型，可以通过每一时刻轮子的变化情况，计算出机器人的运动轨迹以及当前的位姿描述，暂且将这种方法称为轮速计导航。

惯性导航主要依赖于 MEMS 元件，根据陀螺仪、加速度计、的变化推算机器人的运动轨迹。

相比之下，如果基于视觉传感器的视觉里程计中的回环检测就能一定情况下解决累计误差的问题。

表 3-1 不同的航迹推算算法比较

Tab.3-1 Compare different DR algorithm

	轮速计导航	视觉里程计	惯性导航
计算复杂度	简单	复杂	一般
稳定性	高	一般	一般
累计误差	大	小	大
回环检测	不能	可以	无
缺点	轮子与地面之间如果打滑就会导致所有的里程信息出错	光线和环境相似度会影响视觉里程计的计算结果	对传感器和外部电磁环境的要求较高

如表 3-1 所示为轮速计导航、视觉里程计、惯性导航算法间的差异比较，经过多方面的对比，为了弥补由轮速导航方法在转动角度和角速度计算不准确的缺陷，本设计加入惯性运动数据共同做扩展卡尔曼滤波（robot_pose_ekf ，navigation 集合包中），使用了基于机器人运动学模型的轮速计数据和惯性导航数据融合的组合定位方法生成里程信息，提供了更为精准的里程信息。

3.3 SLAM

SLAM（Simultaneous Localization and Mapping，同步定位与地图构建）是指搭载特定传感器的主题，在没有环境先验信息的情况下，于运动过程中建立环境的模型，同时估计自己的运功。常见的开源激光 slam 算法如表 3-2 所示。

表 3-2 不同的 SLAM 算法比较

Tab.3-2 Compare different SLAM algorithm

	hector_slam	gmapping	TinySLAM
低特征场景建图	差	较好	差
计算复杂度	高	较低	较低
回环检测	有	有	无
缺点	传感器要求高	要求地面够平坦	稳定性差

hector_slam 算法是利用已经获得的地图对激光束点阵进行优化，估计激光点在地图

的表示，和占据网格的概率，其中扫描匹配利用的是高斯牛顿的方法进行求解。找到激光点集映射到已有地图的刚体转换(x,y,theta)，为避免局部最小而非全局最优的(类似于多峰值模型的，局部梯度最小，但非全局最优)出现，地图采用多分辨率的形式。

gmapping 采用的是 RBPF 的方法，是一种粒子滤波(利用统计特性描述物理表达式下的结果)的方法，粒子滤波的方法一般需要大量的粒子来获取好的结果，但这必会引入计算的复杂度;粒子是一个依据过程的观测逐渐更新权重与收敛的过程，这种重采样的过程必然会代入粒子耗散问题, 大权重粒子显著，小权重粒子会消失(有可能正确的粒子模拟可能在中间的阶段表现权重小而消失). 自适应重采样技术引入减少了粒子耗散问题 ，计算粒子分布的时候不单单仅依靠机器人的运动(里程计)，同时将当前观测考虑进去, 减少了机器人位置在粒子滤波步骤中的不确定性[6]。

Tinyslam（也称 coreslam）基于蒙特卡洛定位算法的简单 SLAM 实现方法，将 SLAM 简化为距离计算和地图更新的两个部分, 第一部分，每次扫描输入，使用简单的粒子滤波算法计算距离，粒子滤波的匹配器用于激光与地图的匹配，每个滤波器粒子代表机器人可能的位姿描述和对应的概率权重，这些都依赖于之前的迭代计算，选择好最好的假设分布，即低权重粒子消失，新粒子生成，第二部分，将扫描得到的线加入地图中，当障碍出现时，围绕障碍点绘制调整点集，而非仅一个孤立点。

hector_slam 是个性能非常好的算法，而且不需要里程计数据，但是由于 hector_slam 通过最小二乘法匹配扫描点，且依赖高精度的激光雷达数据，使用扫描角很小且噪声较大的 kinect V1 传感器通过 hector_slam 建图效果很差。gmapping 算法目前激光 2Dslam 用得最广的方法，由于其运算量较小所以在更适合应用在移动机器人领域，只是对地面的平整性要求较高。Tinyslam 算法的代码十分轻量，运算压力也相对较小，但没有回环检测（尽管开源爱好者也正在为此努力），一旦计算出现较大的误差就会一直在错误的位置上建图，而无法矫正这一偏差。

蒙特拉罗定位算法又称为粒子滤波定位方法，是贝叶斯滤波在非线性、非高斯的系统中的实现，其基本思想使用过使用一组离散分布的随机采样来模拟并估计机器人位姿的概率分布情况，每个样本都有自己的权重数值，权重表示了对位姿估计的可信度。

相比自适应蒙特卡洛算法，标准蒙特卡洛算法算法的精度更高，原因是他使用了更多的例子，但这势必就会带来更多的运算量，在移动机器人上的运算能力要求也更高，而自适应蒙特卡洛算法在损失较小精度的情况下，减少了重采样的次数和采样的粒子个

数，提高算法的运行效率。通过使用自适应的 KLD 方法来更新粒子，AMCL 算法在已知地图数据的基础上根据订阅到激光传感器的扫描特征，使用粒子滤波的方法获取最佳定位点，该点称为 Mp (point on map)，以此实现主动定位，跟踪机器人的位姿[7]。

3.4 路径规划

机器人路径规划算法中，全局路径规划器使用了 A* 方法，A*算法是一种高效的路径搜索算法，采用启发函数来估计地图上机器人当前的位置到目标位置之间的距离，并以此选择最优的方向进行搜索，如果失败会选择其他路径继续搜索直到得到最优路径。

局部路径实时规划是利用 base_local_planner 包实现的，该包使用 DWA（Dynamic Window approaches，规划推理和动态窗口）算法，计算机器人每个周期内应该行驶的速度和角度（dx，dy，dtheta velocities）。DWA 算法中先离散采样机器人控制空间（dx,dy,dtheta），再对于每个采样速度，从机器人当前的状态，进行模拟预测，如果采样速度应用于一段时间内，将会出现什么情况，然后用合并一些特征的度量标准来，从模拟预测中，评价每个轨迹结果，如障碍物接近目标，接近全局路径和速度。舍弃不合适的路径（有障碍物碰撞的）最后挑选得分最高的轨迹，并且发布相关的速度给移动平台。

4 硬件设计

4.1 硬件系统概述

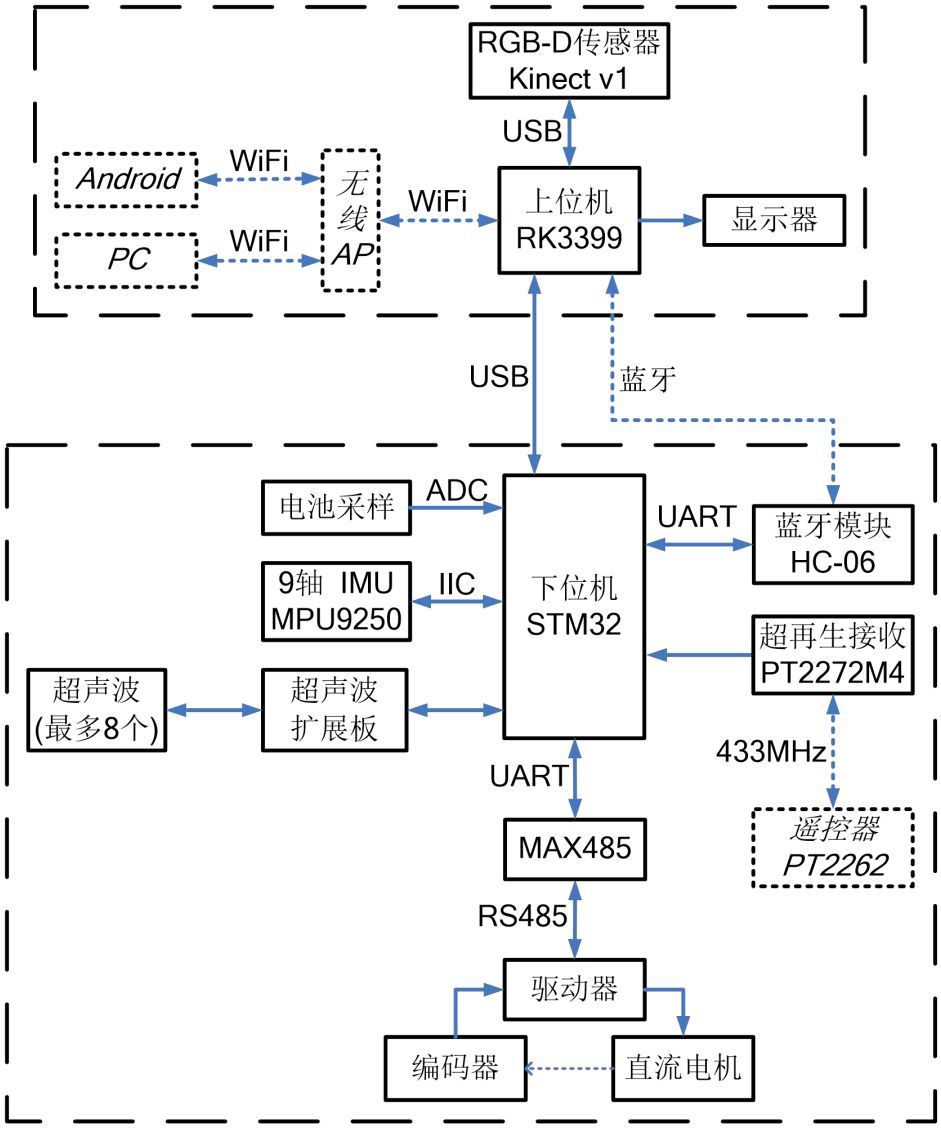


图 4-1 硬件系统框图

Fig.4-1 Hardware system diagram

如图 4-1 所示，本设计的硬件系统主要分为上位机系统和下位机系统两部分，上位机需要安装 Ubuntu 系统，在 Ubuntu 系统上搭载 ROS[①]，而且还要负责上层算法（建图、定位、导航）、人机交互、数据存取等任务，所以需要一款低功耗高性能的 ARM 单板计算机——Firefly-RK3399 单板计算机；本设计中使用的是微软公司生产的一款 RGB-D

① ROS：一种开源的机器人操作系统，Robot Operating System 的缩写。

传感器作为机器人的眼睛，该传感器可以生成周围环境的点云图像。下位机主要负责控制命令的执行和基础传感器驱动等任务，所以下位机选用了 ST 公司生产的基于 ARM-Cortex M3 内核的嵌入式处理器 STM32F103；为保证整个底盘的通用性，适配更多的计算平台主机，收发数据的端口不仅支持免布线的无线蓝牙连接和 TTL 电平的串口，而且还支持 USB 虚拟串口的有线连接方式，同时下位机系统中还有包括超声波、惯性测量、RS485 收发器、超再生接收等单元。

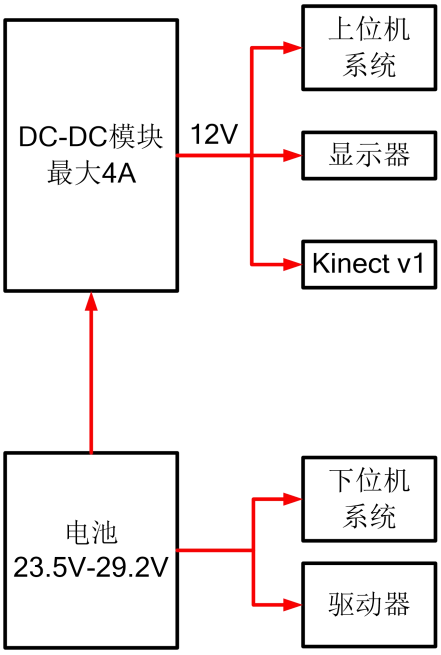


图 4-2 系统供电框图

Fig.4-2 System power supply block diagram

如表 4-2 所示为本设计的系统供电框图，描述了系统的供电框架，电池和驱动器之间不经过稳压电路而是选择直连，并且避免过度大起动电流对稳压电路造成伤害，这样步进保证电机的驱动能力和供电效率；电池直连下位机系统可以直接检测到电池状态，电池电量过低情况下会报警告知用户准备保存工作文件。上位机系统、显示器以及 Kinect V1 均为 12V 供电，由大功率的 DC-DC 模块稳压后来提供电源。

4.2 下位机典型硬件设计

4.2.1 微控制器

本设计的下位机主控制器使用微控制器，微控制器是下位机系统的核心单元，目前

市面上的微控制器种类繁多，差异化明显，如表 4-1 所示为常见微控制器的特点比较。

表 4-1 不同的 MCU 比较

Tab.4-1 Compare different MCU

	STM32F1	STM32F4	Arduino	AT89S51
价格	较高	高	中	低
性能	较高	高	中	低
扩展性	较高	高	中	低
开源代码	较多	较多	多	中
片上资源	较多	多	中	少

经过对表 4-1 中的数据对比，本设计中的主控芯片选用了 STM32F103RCT6，该芯片是一个意法半导体生产的基于 ARM-Cortex M3 架构的 32 位高性能嵌入式处理器，该芯片内置 CAN、I2C、IrDA、LIN、SPI、UART/USART、USB、IIC 等通信接口、3 组 16 通道的 12 位 ADC 单元、DMA 单元、RTC 单元、WDT 单元等，256KB 的 FLASH 程序存储容量以及 48KB 的 RAM 容量，该芯片丰富的片上外设、高可扩展性和性价比作为移动机器人的下位机主控制器是非常适合的[8]。

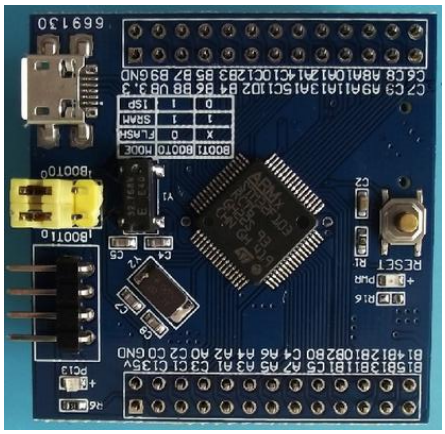


图 4-3 STM32F103RCT6 最小系统实物图

Fig.4-3 STM32F103RCT6 minimum system physical figure

为加快开发进度和提高系统稳定性，避免重复“造轮子”的情况，本设计使用了模块化的 STM32 最小系统核心板，其实物图片如图 4-3 所示。

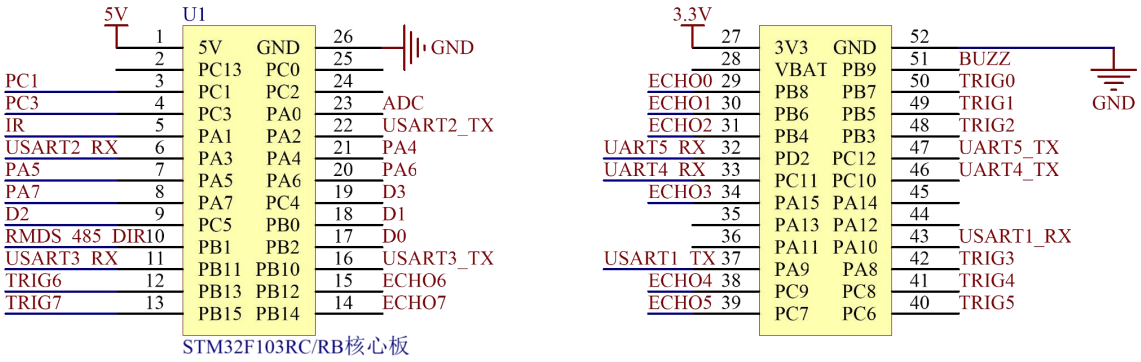


图 4-4 STM32F103RCT6 核心板接线图

Fig.4-4 STM32F103RCT6 Core board wiring circuit

如图 4-4 所示为 STM32 核心板的引出引脚接线图。

4.2.2 蓝牙模块

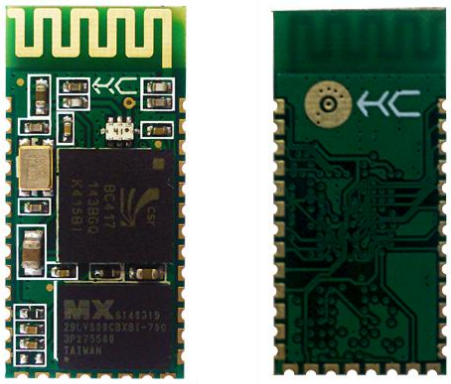


图 4-5 HC-06 蓝牙模块实物图

Fig.4-5 HC-06 bluetooth module physical figure

本设计中的蓝牙通信单元使用广州汇承生产的 HC-06 蓝牙模组，其实物如图 4-5 所示，该模块支持 AT 指令，支持串口透传，同时为邮票孔封装，便于嵌入到下位机系统中，具体参数如表 4-2：

表 4-2 HC-06 型蓝牙模块性能参数

Tab.4-2 HC-06 bluetooth module performance parameters

接口	串行 TTL
发射功率	4dBm（最大）
空中波特率	2Mbps
电流	40mA
接受灵敏度	-85dBM@2Mbps
参考距离	10m

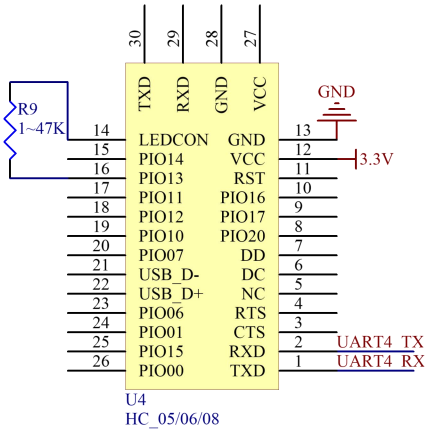


图 4-6 HC-06 蓝牙模块电路

Fig.4-6 HC-06 bluetooth module circuit

如图 4-6 所示为蓝牙模块的电路图，蓝牙模块的串行数据口交叉接在 STM32F103 的串口 4 引脚上，R9 为调节模块 LED 亮度的电阻，不同阻值的电阻 LED 亮度不同。

4.2.3 旋转编码器模块

图 4-7 编码器实物图

Fig.4-7 Encoder physical figure

如图 4-7 所示为本设计中使用的编码器为欧姆龙公司生产的 lpd3806-400bm-g5-24c 型号编码器，该传感器性能优良、输出稳定，具体参数如表 4-3 所示：

表 4-3 编码器性能参数

Tab.4-3 The encoder performance parameters

外径	38mm
轴径	6mm
分辨率	400
电压	最高 24V
输出方式	AB 两相正交输出
信号电平	开漏输出

4.2.4 超声波模块

图 4-8 US-015 型超声波模块实物图

Fig.4-8 US-015 type ultrasonic module physical figure

如图 4-8 所示为本设计使用的 US-015 型超声波模块，该超声波测距模块的分辨率高于 1mm，最高可达 0.5mm，重复测量一致性好，测距稳定可靠。US-015 超声波测距模块可实现 2cm ~400cm 的非接触测距功能，供电电压为 5 V，工作电流为 2.2mA，兼容 CMOS 或 TTL 电平通信。

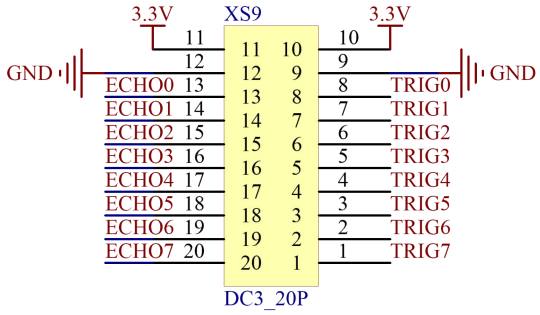


图 4-9 主控制板超声波插座电路

Fig.4-9 The main control board ultrasonic socket circuit

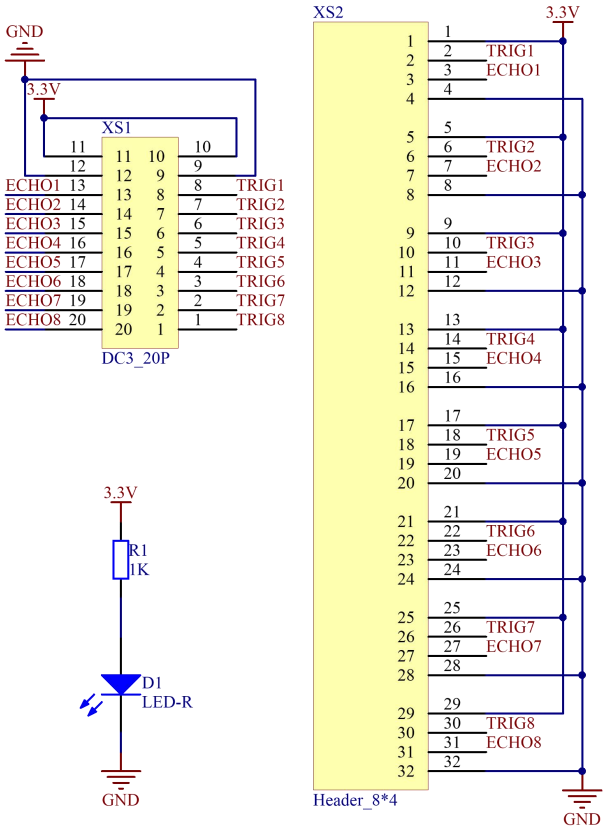


图 4-10 超声波转接板电路

Fig.4-10 Ultrasonic adapter board circuit

如图 4-9 和图 4-10 所示分别为主控制板的超声波牛角座电路和超声波转接板两部分电路，两部分通过 20P 带有防呆插头的排线相连接，为提高平台的通用性和设计，本论文中设计了一个超声波转接装置，在下位机主控板上面提供一组 20 建议牛角插座，然后通过 20P 排线连接到超声波转接板上面，超声波模块只需连接在转接板上接口，这样的设计简化了超声波和主控芯片之间的连接，同时还提高了超声波的可扩展性（最多支

持 8 个超声波接入）。如果不使用转接板，8 各超声波模块可能需要 32 根线和主控制板连接，这样会导致不仅布线繁琐而且容易接错。

4.2.5 惯性测量模块

MPU9250 是一个 QFN 封装的复合芯片（MCM），它由 2 部分组成。一组是三轴加速度计和三轴陀螺仪，另一组是 AKM 公司的 AK8963 三轴磁力针。所以，MPU9250 是一款 9 轴运动追踪装置，由于其使用了 I2C 的通信方案，所以可直接输出 9 轴的全部数据。一体化的设计，运动性的融合，始终校准功能。具有三个 16 位加速度 AD 输出，三个 16 位陀螺仪 AD 输出，三个 6 位磁力计 AD 输出，精密的慢速和快速运动跟踪，提供给用户全量程的可编程陀螺仪参数选择（±250，±500，±1000，and±2000°/(dps)), 可编程的加速度参数选择±2g,±4g，±8g，±16g，以及最大磁力针可达到±4800uT。

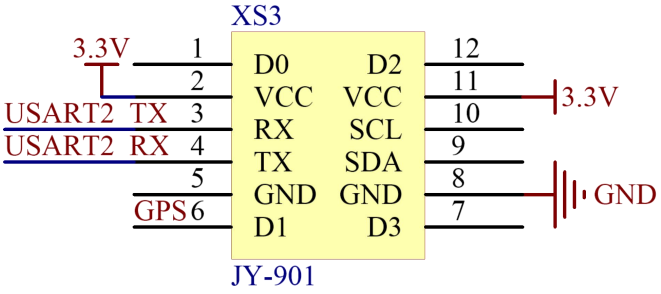


图 4-11 惯性测量模块电路

Fig.4-11 Inertial measurement module circuit

如图 4-11 所示为模块为惯性模块的电路原理图，模块内部自带电压稳定电路，工作电压 3v~6v，支持串口和 IIC 两种通信接口，串口速率范围在 400bps~921600bps 可调，IIC 接口支持全速 400K 速率，最高 200Hz 数据输出速率，输出速率可调节。模块采用邮票的 PCB 设计，方便嵌入到用户的 PCB 板中。

4.2.6 RS485 收发器

RS485 收发器使用 Maxin 公司生产 MAX485 芯片，该芯片是一个标准的 RS485 收发器，只能进行半双工的通讯，内含一个输出驱动器和一个信号接收器。MAX485 具有低功耗设计，静态电流仅为 300uA。MAX485 具有三态输出特性，在使用 MAX485 时，总线最多可以同时连接 32 个 MAX485 芯片。通讯波特率最高可达到 2.5Mbps。

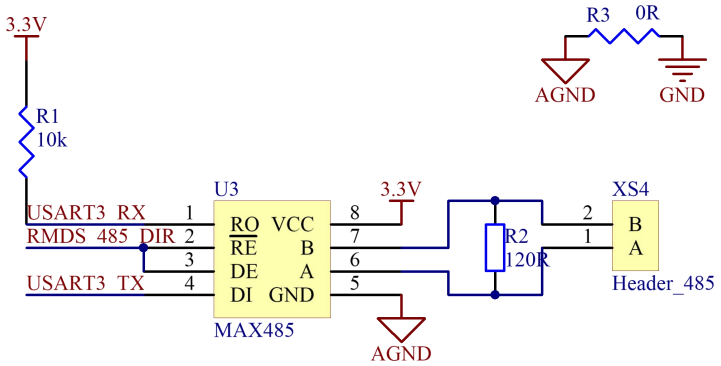


图 4-12 MAX485 电路

Fig.4-12 MAX485 circuit

如图 4-12 所示为 MAX485 的电路原理图，主要负责主控芯片和 RMDS 驱动器间通信时 RS485 和 CMOS 的电平转换任务，在 UART 端不仅有一组串口交叉连接到主控芯片上，还有一个 RS485 数据流向控制引脚，是因为 MAX485 只支持半双工模式，导致 RS485 总线工作在半双工状态，所以 RS485 总线的数据流向需要由主控芯片来控制。

4.2.7 电机驱动器

图 4-13 RMDS-106+型驱动器实物图

Fig.4-13 RMDS-106+ type driver physical figure

如图 4-13 所示为 RMDS-106+型电机驱动器实物图，该驱动器内置独立的 MCU，可以和用户的控制器进行双向数据交互，内置可调的软件 PID 控制器，可闭环控制电机的电流、速度、位置或开环控制，可支持供电电源电压 33V 以下电流小于 10A 的有刷电机。

表 4-4 RMDS-106+型驱动器性能参数

Tab.4-4 RMDS-106+ type driver performance parameters

电压支持	7-33v
电压输出效率	0.99
瞬态电压抑制	33V
最大短时电流	20A
待机电流	17mA
接口支持	CAN、RS232、RS485

如表 4-4 所示为 RMDS-106+驱动器参数，根据之前的计算该驱动器电压和功率指标皆满足设计需求。

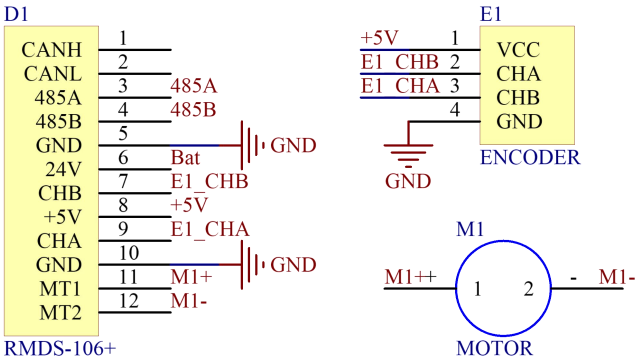


图 4-14 RMDS-106+型驱动器电路

Fig.4-14 RMDS-106+ type driver circuit

如图 4-14 所示为 RMDS-106+驱动器与电动机、编码器的连接电路原理图，编码器和电机连接在驱动器，驱动器通过 RS485 总线与微控制器 STM32 进行数据交互。

4.2.8 无线接收模块

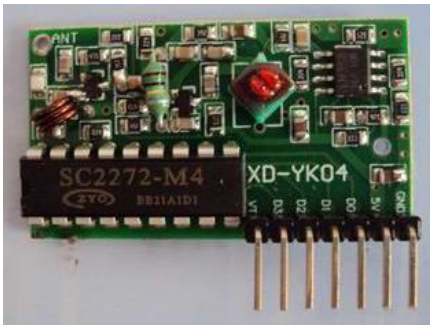


图 4-15 PT2272-M4 型无线模块实物图

Fig.4-15 PT2272-M4 type wireless module physical figure

如图 4-15 所示为基于 PT2272M4 芯片的超再生接收模块，该模块内部采用 LC 振荡电路，内含放大整形电路，输出的数据信号为解码后的 TTL 电平信号，接入到本设计系统中较为方便，支持调整频段可以保证多设备之间不冲突，并且性价比较高。

表 4-5 PT2272-M4 无线模块性能参数

Tab.4-5 PT2272-M4 type wireless module performance parameters

静态电流	4.5MA
调制方式	调幅（OOK）
工作温度	-10℃~+70℃
接收灵敏度	-105dBm
工作频率	266-433MHZ 频率段可任选
工作方式	点动（按住一直输出，抬手即停）
输出电平	TTL

如表 4-5 所示为 PT2272-M4 无线模块的参数，配合一个基于 PT2262 芯片的遥控器，就可以实现无线遥控、紧急制动的功能。

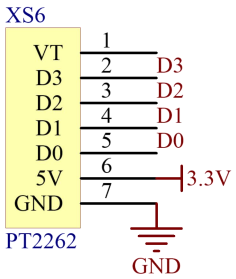


图 4-16 PT2272-M4 型无线模块电路

Fig.4-16 PT2272-M4 type wireless module circuit

如图 4-16 所示为无线接收模块的原理图，4 个输出引脚分别对应遥控器的 ABCD 四个按键，将 4 个与无线模块连接的引脚设置为外部触发模式，即可根据引脚电平的变化得知遥控按键是否被按下。

4.2.9 电池电压采样

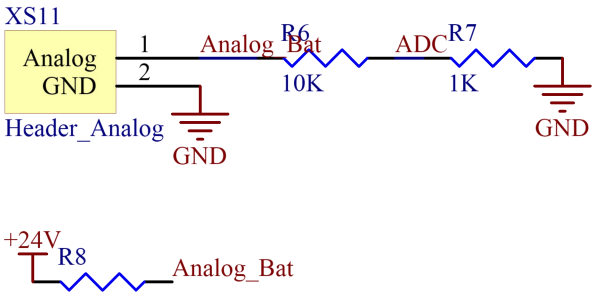


图 4-17 电压采集电路

Fig.4-17 Voltage acquisition circuit

如图 4-17 所示为电池分压电路，由于电池的电压较高无法直接接入单片机的 ADC 单元，所以设计两个电阻分压，分压后的电压和电池的实际电压成线性关系，主控芯片读取内置 10 位 ADC 结果寄存器数值为 n，通过公式(4-1)计算当前电池电压：

$$V_{Bat} = V_{Sample} \times \mathrm{k} = \frac{\mathrm{n}}{2^{12}} \times \frac{R6 + R7}{R7} = \frac{11 \times \mathrm{n}}{4095} \tag{4-1}$$

4.2.10 稳压器

Firefly-RK3399 单板计算机、显示器和 Kinect 都需要 12V 供电，总功率最大能达到 40W 以上，而本设计中的电池为直接输出，并没有稳压措施。所以需要使用一种稳压器来将电池和上位机系统连接起来。

表 4-6 不同的稳压器比较

Tab.4-6 Compare different voltage regulator

	线性稳压器	开关稳压器
效率	压差越大效率越低	大多 90%左右
输出功率	小（且和散热器设计有关）	大
电磁干扰	小	大（且和外壳设计有关）
纹波	小	一般较大
发热	大	小

如表 4-6 所示为可以得出本设计对大功率需求较高，而由于上位机系统中 Firefly-RK3399 单板计算机、显示器和 Kinect 的内部都有进一步的稳压措施，对于纹波的要求也不是特别高，所以开关稳压是更为适合本设计。

表 4-7 DC-DC 模块性能参数

Tab.4-7 DC-DC module performance parameters

整流方式	非同步整流
输入电压	4.5-60V
输出电压	1.25-30V（连续可调）
输出电流	最大 4A
转换效率	92%
开关频率	50KHz
负载调整率	±0.5%
电压调整率	±0.5%
输出纹波	小于 10mV(输入 48V 输出 5V 3A 时)

本设计选用了一款大功率的 DC-DC 稳压器，参数如表 4-7 所示，效率和功率以及纹波等主要参数都满足上位机系统的要求。

4.2.11 电池

本设计中机器人全速运行、上下位机全速运行时最大的功率达 80w，保守估计稳压单元的功率为 90%，则需要电池最大的输出功率为 88w，在这种情况下如果续航两个小

时，那么消耗能量可以根据公式(4-2)计算。

$$W = P \times \mathrm{t} = 88 \times 7200 = 633600J = 176W \cdot h \tag{4-2}$$

所以需要容量在 176$W \cdot h$以上的电池，根据电机的特性，需要选择输出电压为 24V 左右的电池。

表 4-8 电池的性能参数

Tab.4-8 Performance of the battery

电压	24V（23.5-29.3V）
最大电流	10A
容量	12ah/240Wh
体积	150mm*140mm*40mm

本设计中所使用的电池参数如表 4-8 所示，电压、电流和容量指标皆满足设计需求，体积也恰好适合机器人底盘。

4.3 PCB 设计

RS485 通信中的电平信号使用差模方式的模拟信号传输，而嵌入式控制器主要为数字电路，如果不对两部分电路采取一定的措施，把模拟地和数字地直接大面积相连，那么在工作时候数字信号的高次谐波通过接地回路电流串入模拟电路，模拟信号为高频或强电信号时，也会通过接地回路影响到数字电路中电平逻辑表达的准确性，所以采用了数字地模拟地之间单点接地的措施，单点接地处使用 0 欧电阻跨接，0 欧电阻电流路径狭窄，可以限制噪声电流通过，能够有效地限制环路电流，而且 0 欧电阻对所有频率的噪声都有衰减作用，这样的设计保证数字与模拟电路之间的互相干扰降到最低。

电源线都经过加粗到 30mil 或 50mil，保证电源大电流的通过性。

板载了大功率、低纹波的 DC-DC 稳压电源模块，使得宽电压输入范围得以实现，考虑到散热问题，在电源模块下的 PCB 上设计了一个过孔矩阵，能让稳压模块发出的热量尽快散开。在 PCB 设计中还采用了多边形覆铜的设计，其意义在于减小地线阻抗来提高抗干扰能力、降低压降来提高电源效率、与地线相连来减小环路面积。

为减小地环的面积来减小接地阻抗，抑制地线上的噪声，在保证机械强度的情况下，加入了一些地线过孔；也在 PCB 边缘增加地墙，有利于 EMI 抑制，同时为减小过孔自身的寄生电容，在控制成本的前提下将过孔缩小到 15mil。但是由于过孔在传输线上表

现为阻抗不连续的断点，会造成信号的反射，所以在设计过程中极力避免信号线中出现过孔的情况。

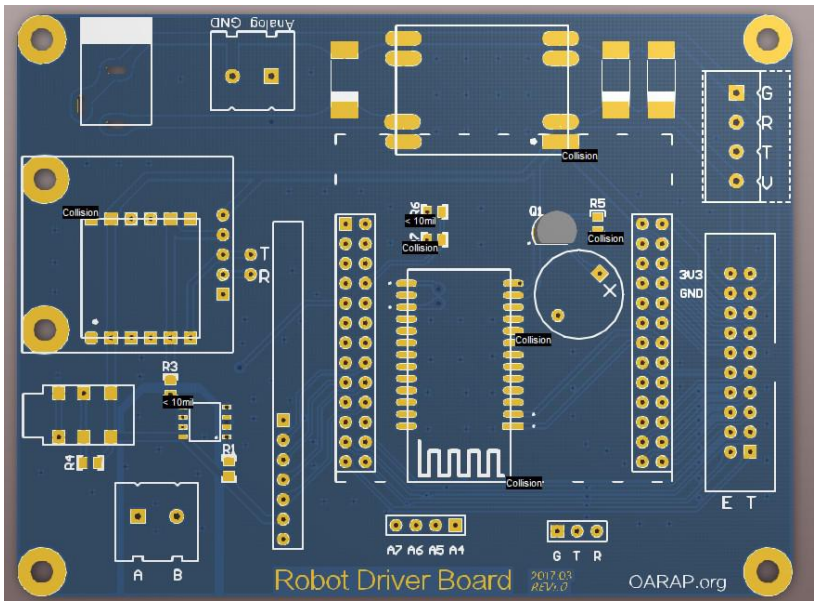


图 4-18 下位机主控板 PCB 预览图

Fig.4-18 Lower machine main control board PCB preview

如图 4-18 所示，在布局上，各功能单元间布局紧凑，元件在整个版面上分布均匀、疏密一致，将关键元器件、易受干扰的元器件、发热量大的元器件、电磁干扰强的元器件、异型元器件等都独成一个单元，避免单元间发生干扰或不兼容的情况。

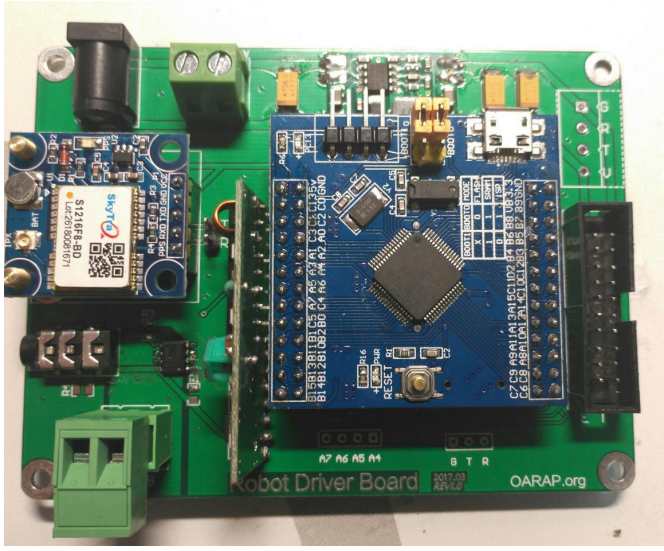


图 4-19 下位机主控板实物图

Fig.4-19 Lower machine main control board physical figure

如表 4-19 所示，PCB 上各硬件模块的接口大都使用了防呆设计，如超声波扩展板连接的 20P 简易牛角座、RS485 通信接口、电源插口、预留串口的插座、预留的 GPS 接口等。在 STM32F103 核心板下方的空间放置了蓝牙模块、蜂鸣器电路以及电池电压采集电路，无线超再生接收模块采用立式安装的方法，缩减了下位机系统所需的空间也降低了 PCB 打印成本，而且 PCB 面积越小其机械强度也越高。

4.4 上位机典型硬件设计

4.4.1 计算平台

表 4-9 不同的计算平台比较

Tab.4-9 Comparison of different computing platforms

	Intel-NUC 6i7kyk	树莓派-3B+	Firefly-RK3399	Nvidia-Jetson TK1
架构	X86	ARM	ARM	ARM
价格	4000 左右	300 左右	1000 左右	2000 左右
最高功耗	100W	10w	24w	50w
性能	高	低	中	较高
兼容性	高	中	较高	高
体积	中	小	中	大

如表 4-9 所示为市面上常见的计算平台特性对照表。在设计初期由于对运算能力的要求估算不准确，导致当时错误的选用了树莓派 3B+作为整个机器人的大脑，在调试过程发现其 USB 的带宽不足，导致读取点云数据稍微有些吃力，并且是经常出现内存和 CPU 占满导致死机的情况，于是在后来将运算平台换为开源的高性能 ARM 主板 Firefly-RK3399 单板计算机，高主频、多核心和大内存提供了更强的计算能力，其高性能的图形图像处理器，对本设计的建图定位等功能的实现也十分有帮助。

表 4-10 Firefly-RK3399 单板计算机性能参数

Tab.4-10 Firefly-RK3399 Single board computer performance parameters

主控芯片	Rockchip RK3399 (28 纳米 HKMG 制程）
处理器核心	6 核 ARM® 64 位处理器，主频高达 2.0GHz，big.LITTLE 大小核架构，双核 Cortex-A72(大核)+四核 Cortex-A53(小核)
图形处理器	ARM Mali-T860 MP4 四核 GPU，支持 OpenGL ES1.1/2.0/3.0/3.1, OpenVG1.1, OpenCL, DX11，支持 AFBC（帧缓冲压缩）
视频处理器	支持 4K VP9 and 4K 10bits H265/H264 视频解码，高达 60fps，1080P 多格式视频解码 (WMV, MPEG-1/2/4, VP8)，1080P 视频编码，支持 H.264，VP8 格式，视频后期处理器：反交错、去噪、边缘/细节/色彩优化
电源管理	RK808 PMU 芯片
内存	2GB/4GB 双通道 DDR3
存储器	16GB/32GB 高速 eMMC，MicroSD (TF) Card Slot，PCIE M.2 NGFF (B-KEY) 接口 ，可用于扩展 SSD
无线网络	板载 WiFi 模块（AP6356S）：2.4GHz/5GHz 双频 WiFi，支持 802.11a/b/g/n/ac 协议 、2x2 MIMO 技术，Bluetooth 4.1（支持 BLE）
以太网	10/100/1000Mbps 以太网 (Realtek RTL8211E)
显示	1 x HDMI 2.0 (Type-A), 支持 4K@60 帧输出，1 x DP 1.2 (DisplayPort) , 支持 4K@60 帧输出，1 x MIPI , 支持双通道 2560x1600@60 帧输出，1 x eDP 1.3 (4 lanes with 10.8Gbps)

图 4-20 Firefly-RK3399 单板计算机实物图

Fig.4-20 Firefly-RK3399 Single board computer real figure

如图 4-20 和表 4-10 分别为本设计上位机计算平台的实物图和硬件参数。

4.4.2 传感器

表 4-11 不同的传感器比较

Tab.4-11 Comparison of different sensors

	RPLIDAR-A1	Kinect V1	罗技 C270	罗技 C270 *2
类型	激光雷达	RGB-D	图像	图像（双目）
稳定性	高	中	低	中
价格	1000 左右	400 左右	100 左右	200 左右
角度	360 度	57.3 度	75 度	75 度左右 (取决于安装)
运算量	少	中	较多	多
测量半径	8m	0.8-4m	不受限	不受限
计算深度	能	能	不能	能

如表 4-11 所示为市面常见的机器人“眼睛”方案，经过多方对比，本设计使用了较为成熟且成本不高的微软 Kinect 传感器。

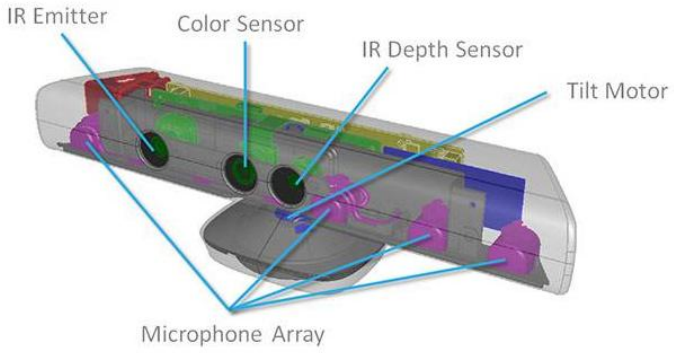


图 4-21 Kinect 硬件功能

Fig.4-21 Kinect hardware function

作为机器人的“眼睛”，如图 4-21 所示，Kinect 有三个镜头，中间的镜头是 RGB 彩色摄影机，用来采集彩色图像。左右两边镜头则分别为红外线发射器和红外线 CMOS 摄影机所构成的 3D 结构光深度感应器，用来采集深度数据（场景中物体到摄像头的距离），Kinect 采用了“光编码”的方式，读取投射的红外线光谱，通过反射斑点（称为散斑）的变形来取得纵向的深度的信息。彩色摄像头最大支持 1280*960 分辨率成像，红外摄像头最大支持 640*480 成像，最高输出速度为 30fps。Kinect 还搭配了追焦技术，底座马达会随着对焦物体移动跟着转动。Kinect 也内建阵列式麦克风，由四个麦克风同时收音，比对后消除杂音，并通过其采集声音进行语音识别和声源定位[9]。

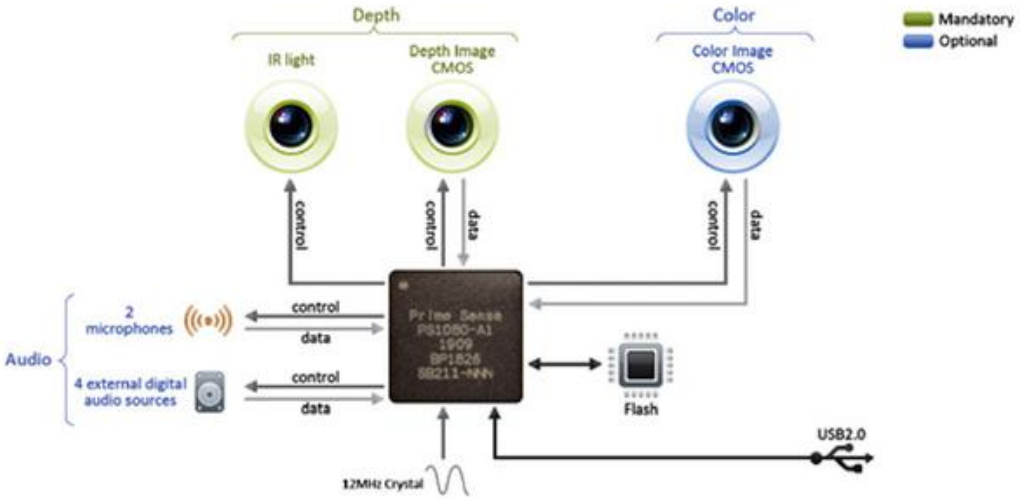


图 4-22 Kinect 内部结构图

Fig.4-22 Kinect internal structure diagram

如图 4-22 所示为 Kinect 传感器的内部结构图。

5 软件设计

5.1 软件设计概述

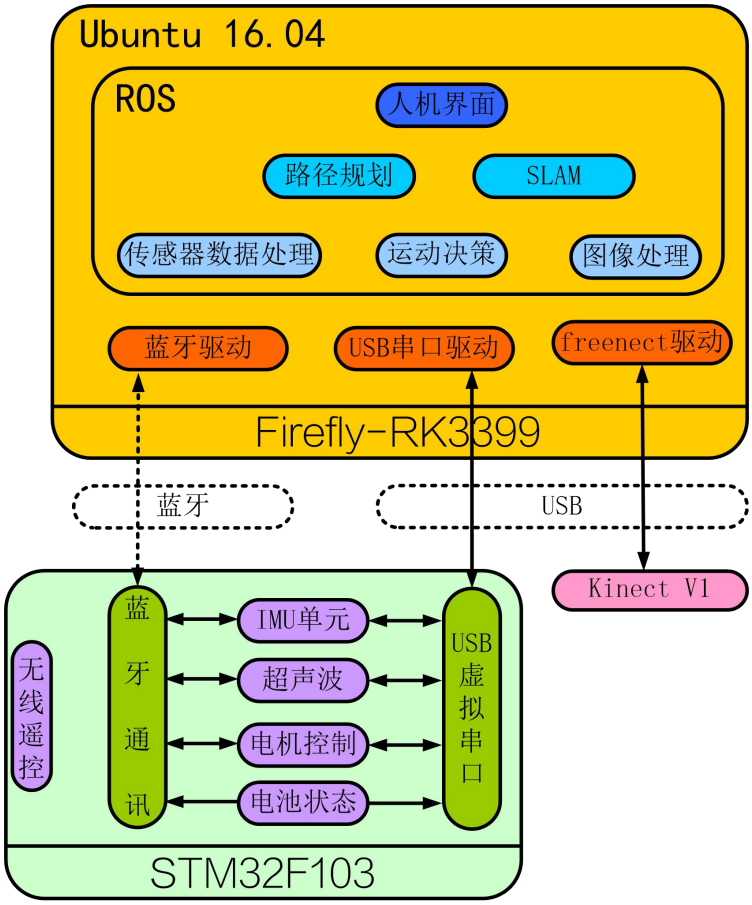


图 5-1 软件系统框图

Fig.5-1 The software system diagram

如图 5-1 所示为系统的软件框架图，下位机部分主要是由惯性单元驱动程序、超声波驱动程序、驱动器驱动程序、电池状态监测程序、无线遥控处理程序、针对通信格式解包打包程序等组成，完成执行上位机发出的动作指令同时反馈一些传感数据到上位机的工作。上位机部分 Ubuntu 提供了基础的 Kinect 传感器驱动和串口、蓝牙驱动，在 Ubuntu 上面搭载的 ROS，在 ROS 社区有很多学者将研究成果和遵循统一接口规范的程序开源，这极大降低了机器人软件的开发成本，本文在前人的基础上，对相关程序作出部分改进和删减，就实现了运动决策、图像处理、传感器数据处理、路径规划、SLAM 等算法，通过 ROS 以节点形式有机组织起来，具体的执行和传感数据将通过 USB、蓝牙或 TTL 串口与单片机进行交互[10]。

5.2 下位机典型软件设计

5.2.1 下位机主程序

下位机是基于 ST 官方提供的标准驱动库编写的代码，主要实现对上位机之间交互的数据进行打包和解包，对周围器件的驱动。风格模块化编程，以“高内聚 低耦合 可扩展”为软件的设计风格。统一接口，规范命名。

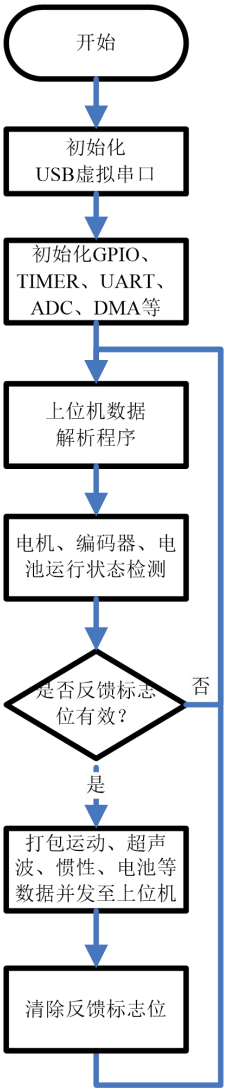


图 5-2 主程序流程图

Fig.5-2 Main program flowchart

如图 5-2 所示为下位机系统的主程序流程图，在开机时初始化所有的外设单元，在主循环中监测上位机数据并解析，根据反馈标志位的变化（反馈标志位会每隔 25ms 在定时器中被置高一次）来决定是否向上位机反馈数据。

5.2.2 通信模块软件

蓝牙模块使用串口方式通信，工作在透传模式下，会直接将 STM32 主控制器发送的串行数据以字节的形式直接转发给已连接的蓝牙主机，而蓝牙主机发来的数据也会直接通过串口发送个 STM32 主控制器。

USB 虚拟串口使用 ST 官方提供的虚拟串口的程序，可以在电脑端（Windows、Linux、mac OS）中实现一个虚拟 USB HID 串口设备，而不需要任何硬件电平转换电路。

USB 虚拟串口、TTL 串口与蓝牙模块的处理流程相同，在代码上也实现了复用。主要分为解包和打包两部分：

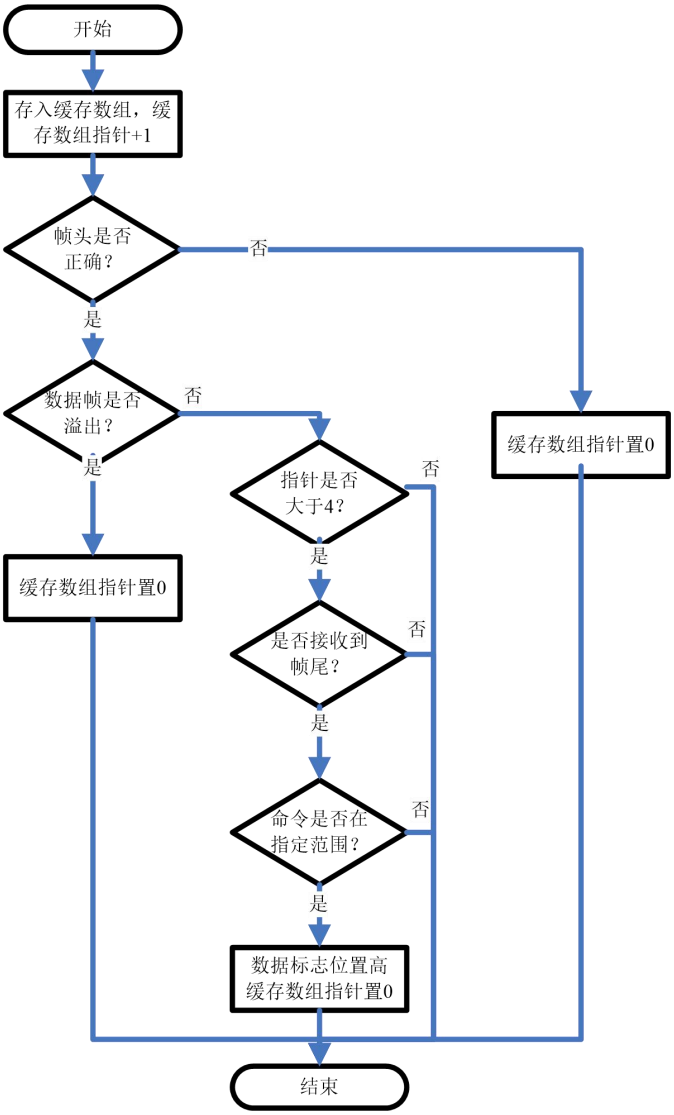


图 5-3 解包程序流程图

Fig.5-3 Unpack function flow chart

如图 5-3 所示为串口数据解包函数的流程图，在对应的中断中调用该解包程序。

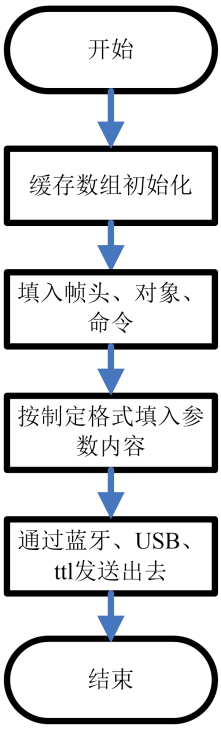


图 5-4 打包函数流程图

Fig.5-4 Packaging function flow chart

如图 5-4 所示，串口数据打包函数中，将数据按照通信格式的定义存入到缓存中并通过蓝牙、USB、TTL 串口发送出去。

5.2.3 惯性单元和驱动器

惯性模块使用 UART 协议，提供三轴加速度、三轴角速度、三轴磁力计数据以及经过姿态解算后的欧拉角数据的输出。RMDS 驱动器的参数可以通过电脑端的配套软件直接设置，包括编码器分辨率、编码器输出模式、PID 参数、控制周期等，在嵌入式控制器中可以实现编码器数值和当前电流的读取、电机速度的控制和读取等。

二者在嵌入式软件的设计思路上相似，故放在一个单元介绍，惯性模块和 RMDS 驱动器都有自己内部的通信格式定义，官方手册中也对通信格式定义进行了详细的阐述，惯性模块的通信格式的数据帧内容：

表 5-1 部分惯性模块 JY-901 通信格式

Tab.5-1 Part of the IMU module JY-901 communication format

长度	1 字节	1 字节	2 字节	2 字节	2 字节	2 字节	1 字节
功能	帧头	对象	变量 1	变量 2	变量 3	温度	校验和
加速度	0x55	0x51	AxL AxH	AyL AyH	AzL AzH	TL TH	SUM
角速度	0x55	0x52	wxL wxH	wyL wyH	wzL wzH	TL TH	SUM
欧拉角	0x55	0x53	RollL RollH	PitchL PitchH	YawL YawH	TL TH	SUM
磁力计	0x55	0x54	HxL HxH	HyL HyH	HzL HzH	TL TH	SUM

表 5-1 为惯性模块的通信格式定义，每个数据帧长度 11 个字节，其中校验和 SUM 等于前十个字节相加的低八位数值。

表 5-2 部分 RMDS 驱动器通信格式

Tab.5-2 Part of the RMDS driver communication format

长度	1 字节	1 字节	2 字节	2 字节	2 字节	2 字节
功能	帧头	ID	变量 1	变量 2	变量 3	变量 4
速度写入	0x48	0x14 或 0x24	PWM_H PWM_L	Vel_H Vel_L	0x55 0x55	0x55 0x55
参数读取	0x48	0x1A 或 0x2A	0x01 0x55	0x55 0x55	0x55 0x55	0x55 0x55
参数反馈	0x48	0x1B 或 0x2B	Cur_H Cur_L	Vel_H Vel_L	Pos_HH Pos_HL	Pos_LH Pos_LL

表 5-2 为 RMDS 驱动器的通信格式定义，数据帧长度共 9 个字节，其中 ID 中的高 4 位代表驱动器号（本设计中为 1 和 2），低四位代表操作的对象。

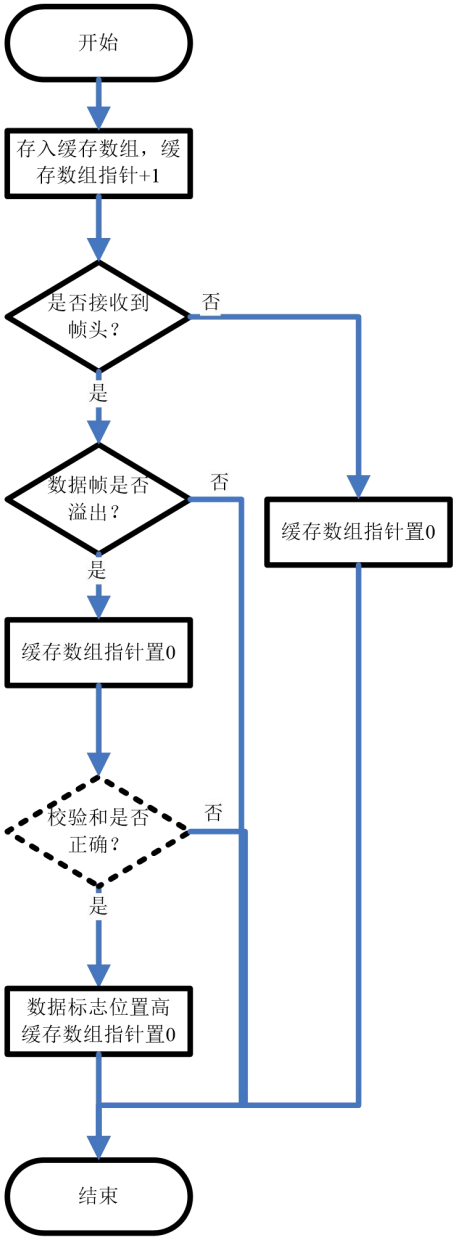


图 5-5 数据解包流程图

Fig.5-5 The flow chart of data unpack

图 5-5 为 RMDS 驱动器和 JY-901 惯性模块的数据解包程序流程图②，该程序分别在各自模块对应的串口中断中实现。数据发送程序较为较为简单，只需按照手册中格式的定义将数据逐次填入到发送缓冲区中并发送，其思路可参考图 5-3，本文在此不再赘述。

② 考虑到驱动器和惯性模块的解包程序思路相似，为减少冗余叙述，故将其画在同一张流程图中，虚线框判断部分表示是惯性模块独有的数据校验程序，在 RMDS 驱动器数据解析程序中无此步骤。

5.2.4 多通道超声波

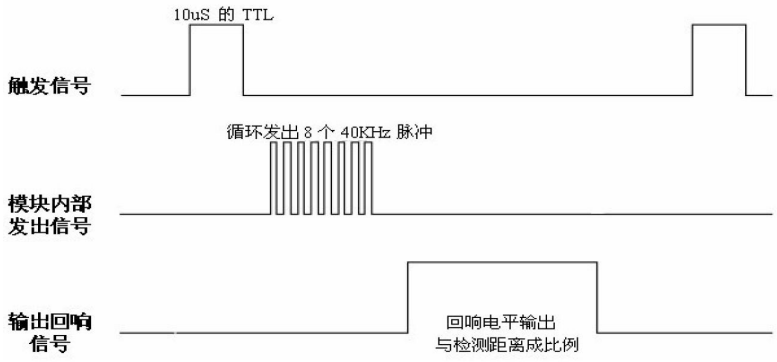


图 5-6 超声波模块时序图

Fig.5-6 Ultrasonic module sequence diagram

根据图 5-6 可知，超声波模块的工作需要单片机发送一个 10us 的触发脉冲，然后超声波模块的发射头会发出 8 个 40kHz 脉冲，发送过后超声波模块的回波引脚会从低电平跳变成高电平并一直保持，直到接收到回波为止。

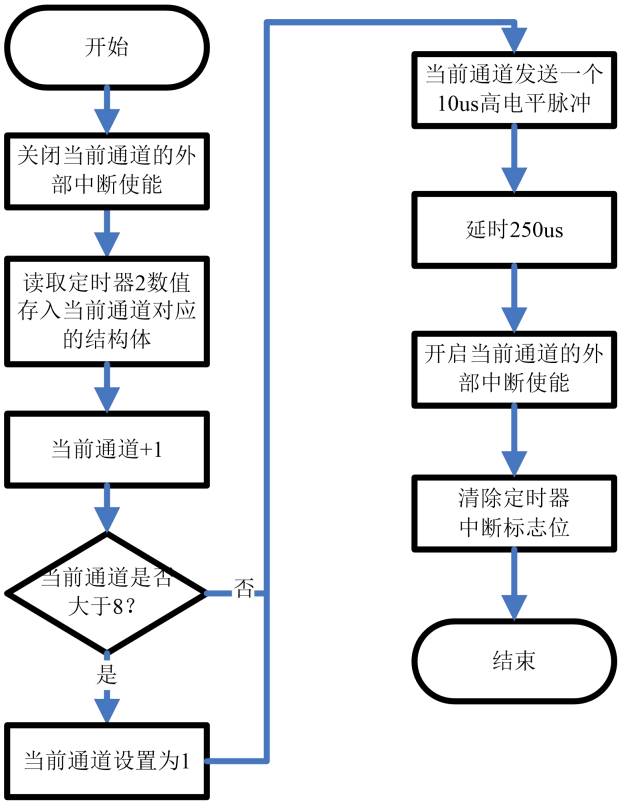


图 5-7 多超声波调度定时器 4 中断流程图

Fig.5-7 Ultrasonic scheduling flow chart of the timer interrupt

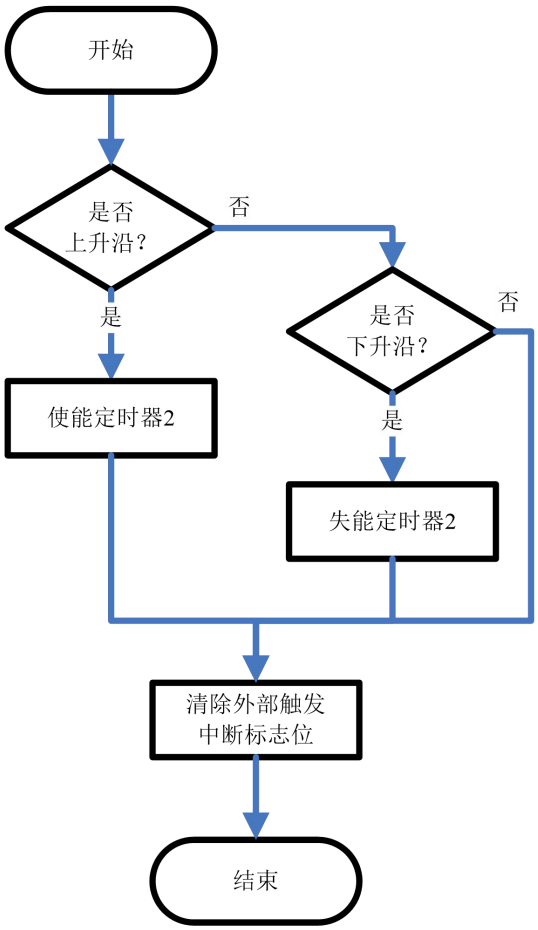


图 5-8 多超声波外部中断流程图

Fig.5-8 Ultrasonic flow chart of the external interrupt

如图 5-7 所示为本设计中多通道超声波测距程序流程图，图 5-8 为超声波回波输出引脚所对应的外部中断引脚的中断函数流程图。为防止多超声波通道间相互干扰，本设计提出了一种时分复用的模式，宏观上实现并行探测，微观上每个时刻只探测一个通道的串行探测，其思路如下，使用定时器 4 中断触发通道 1 发送脉冲，通道 1 反馈脉冲上升沿触发外部中断开始定时器 2 计数，通道 1 反馈脉冲下降沿触发外部中断并停止定时器 2 计数的模式，在下一次进入定时器 4 的时候可以读取通道 1 的反馈脉冲的高电平时间并关闭通道 1 开始触发通道 2 发送脉冲，如此循环往复，而定时器 2 对应的时间就是超声波从发送到返回到接收头的时间，据此可以计算得出与障碍物之间的距离。

通过设置定时器 4 使得相邻通道间的检测间隔为 30ms，而由于信号衰减的原因超声波的探测距离不超过 5m，也就是超声波往返的传输距离不超过 10m（传输距离超过 10m 的超声波信号已经微弱到无法被接收头探测到），而超声波传输 10m 的距离要花费

29.4ms，意味着只要我们两通道检测时间间隔大于 29.4ms 即可保证通道间互不干扰。

5.2.5 无线接收模块

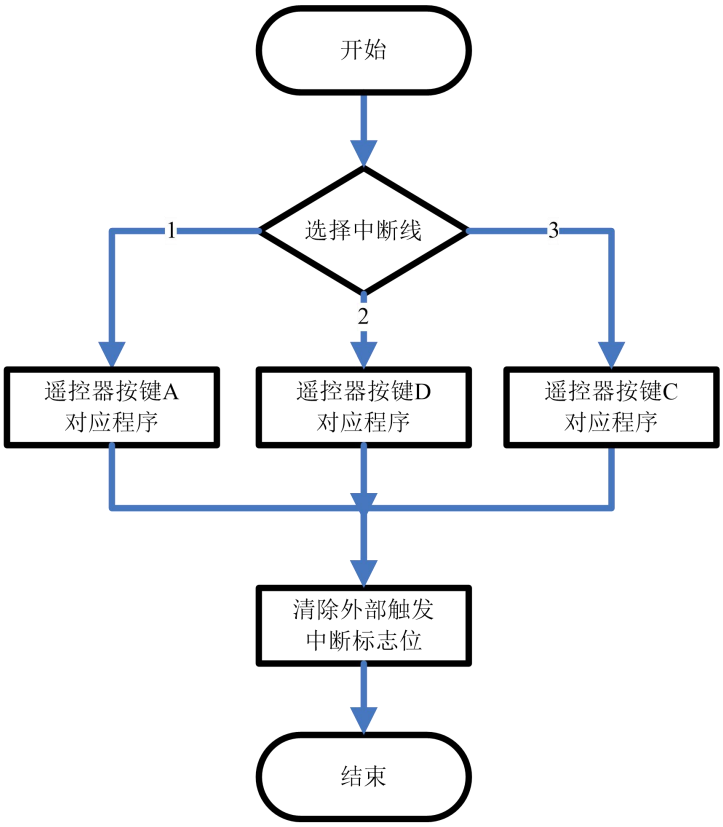


图 5-9 无线接收模块软件流程图

Fig.5-9 The wireless receiving module software flow chart

如图 5-9 所示为无线接收模块的软件程序流程图，本设计中将模块与主控制器相连接的四个引脚设置为输入模式和使能外部中断，并配置好对应的中断线，这样当无线模块的引脚电平发生跳变之后，主控制器 STM32 直接进入外部中断，根据中断线的不同来判断发生中断的引脚，从而判断用户在遥控器端按下的按键。

5.2.6 ADC 单元

使用 STM32 内置的 10 位 ADC 外设，通过初始化设置 DMA 的总线基地址、内存基地址、传输长度、地址方向等变量后，可以实现在运行过程中不需要 CPU 任何干预，DMA 单元实时将 ADC 结果寄存器中的数值传输到指定内存位置，程序中只需要每次读取预先设置好的变量即可得到当前实时 ADC 数值。

5.3 上位机典型软件设计

使用 ROS 的开发机器人时，一般要安装一个基操作系统作为 ROS 的寄生环境，同时提供人机交互和调试的功能。安装 ROS 后的开发流程中，主要需要创建功能包、定义节点、定义消息等几个步骤。

5.3.1 操作系统

1） Ubuntu 系统

在上位机的操作系统层级，使用 Ubuntu 16.04 作为基操作系统，Ubuntu 是基于 Debian 发行版和 GNOME 桌面环境，以桌面应用为主的 Linux 操作系统，Ubuntu 的理念是为用户提供一个最新且又稳定，主要以自由软件建构而成的操作系统。Ubuntu 计划强调可达性和国际化，以便能为尽可能多的人所用。Ubuntu 系统具有庞大的社区力量与资源支持，用户可以方便地从社区获得帮助。

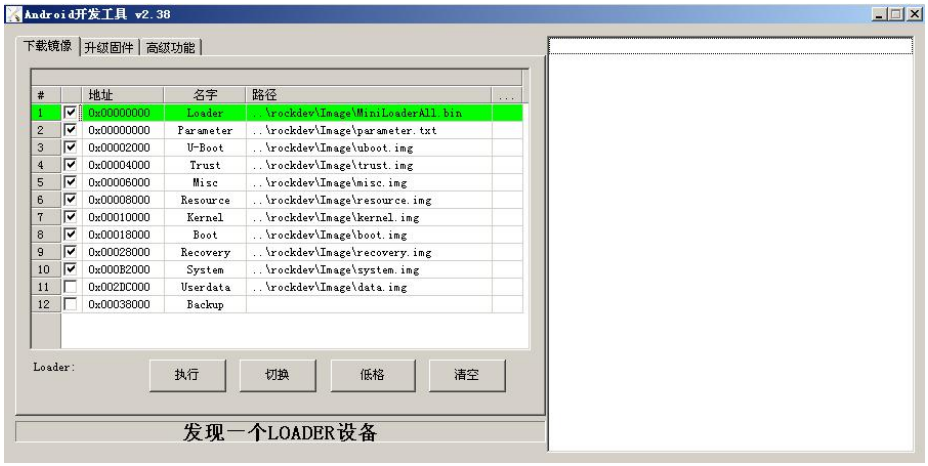


图 5-10 RK-3399 官方固件烧录工具

Fig.5-10 RK-3399 official firmware writing tools

如图 5-10 所示为瑞芯微公司官方提供的运行在 PC 机上的固件烧录工具，通过 USB 连接 PC 机和 Firefly-RK3399 单板计算机后即可使用该软件烧录固件，同时也可以该软件升级单板计算机中的 Ubuntu 固件，本设计中烧录的固件为 firefly 团队制作的 Ubuntu 派生版 Xubuntu（轻量级发行版，基于 Ubuntu 16.04，Xfce4 作为默认桌面环境）。

2） ROS 简介

ROS 是专为机器人软件开发所设计出来的一套电脑操作系统架构。它是一个开源的元级操作系统（后操作系统），提供类似于操作系统的服务，包括硬件抽象描述、底层

驱动程序管理、共用功能的执行、程序间消息传递、程序发行包管理，它也提供一些工具和库用于获取、建立、编写和执行多机融合的程序，以节点的形式组织多个负责不同任务的代码，以话题的形式在节点间基于 TCP/IP 协议传输进行数据的发送和接收。提供基于订阅发布模式的一套松散耦合的程序管理机制。提高机器人研发领域的代码复用率。运行架构是一种使用 ROS 通信模块实现模块间 P2P 的松耦合的网络连接的处理架构，它执行若干种类型的通讯，包括基于服务的同步 RPC（远程过程调用）通讯、基于 Topic 的异步数据流通讯，还有参数服务器上的数据存储。ROS 是一种分布式处理框架（又名 Nodes）。这使可执行文件能被单独设计，并且在运行时松散耦合。这些过程可以封装到数据包（Packages）和堆栈（Stacks）中，以便于共享和分发。ROS 还支持代码库的联合系统。使得协作亦能被分发。这种从文件系统级别到社区一级的设计让独立地决定发展和实施工作成为可能。上述所有功能都能由 ROS 的基础工具实现。ROS 目前只能在基于 Unix 的平台上运行。ROS 的软件主要在 Ubuntu 和 Mac OS X 系统上测试，同时 ROS 社区仍持续支持 Fedora，Gentoo，Arch Linux 和其它 Linux 平台。与此同时（2017 年）Microsoft Windows 端口的 ROS 已经实现，但并未完全开发完成。

3） ROS 架构：

ROS 系统的架构主要被设计和划分成三部分，每一部分都代表一个层级：计算图级（Computaion Graph Levell），文件系统级（FileSystem Level），开源社区级（Community Level）。

a）计算图级：

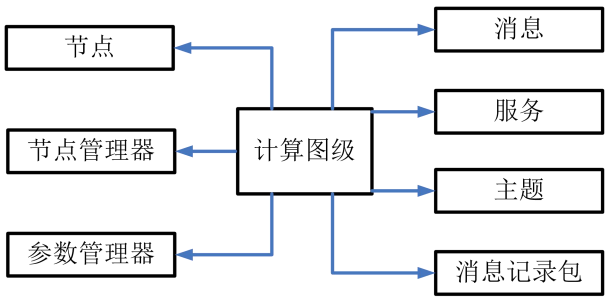


图 5-11 ROS 计算图级

Fig.5-11 ROS Calculation chart level

如图 5-11 所示为 ROS 的计算图，计算图级是 ROS 处理数据的一种点对点的网络形式。程序运行时，所有进程以及他们所进行的数据处理，将会通过一种点对点的网

络形式表现出来。这一级主要包括几个重要概念：节点（node）、消息（message）、主题（topic）、服务（service）。

b）文件级：

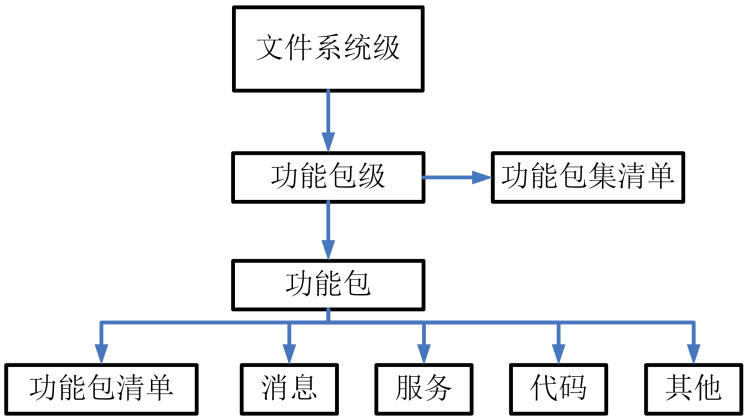


图 5-12 ROS 文件级

Fig.5-12 ROS file level

如图 5-12，ROS 文件系统级指的是在硬盘上面查看的 ROS 源代码的组织形式。ROS 中有无数的节点、消息、服务、工具和库文件，需要有效的结构去管理这些代码。包括功能包（Package）,功能包清单（Manifest）,功能包集（Stack）,功能包集清单（Stack manifest）,服务类型（Service/srv type）。

c）社区级

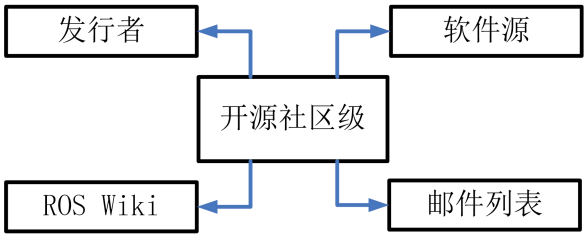


图 5-13 ROS 开源社区级

Fig.5-13 ROS opensource community level

如图 5-13 所示，ROS 开源社区的概念主要关于 ROS 资源，能够独立的网络社区分享软件和知识。这些资源包括：发行者（Distribution）,软件源（Repositorie）, ROS Wiki, 邮件列表（Mailing list）[11]。

4） ROS 安装

本设计中参考 ROS 官方指南③的说明，通过在线安装的方式在 Ubuntu 系统上安装了 Kinetic 版本的 ROS。

5.3.2 导航功能包集

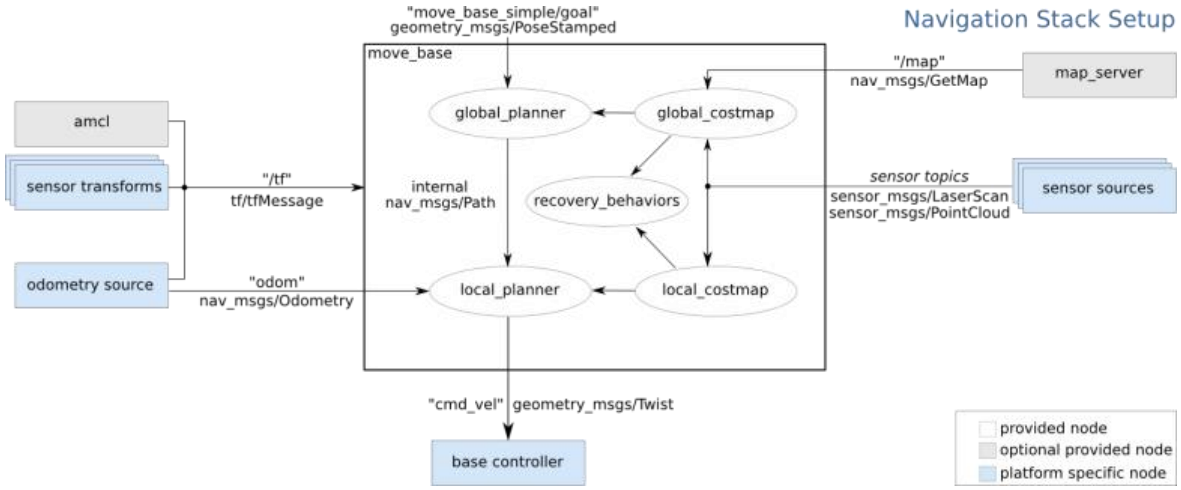


图 5-14 导航功能包集配置

Fig.5-14 Navigation feature stack set configuration

如图 5-14 所示为 ROS 官方提供的导航功能包集运行时的软件架构图，使用 ROS 的导航功能包集需要满足以下条件和限制：导航功能包集只能处理双轮差分驱动和完全驱动的机器人；需要机器人发布关于所有关节的传感器位置关系；机器人必须能发布线速度和角速度消息；机器人必须有平面激光雷达完成地图构建和定位[12]。

③ http://wiki.ros.org/kinetic/Installation/Ubuntu

表 5-3 机器人主要节点

Tab.5-3 The robot main node

节点	功能
/base_controller	基控制器，负责与机器人底盘（即下位机）之间的数据交互
/robot_pose_ekf	基于扩展卡尔曼滤波的位姿滤波器，整合轮速计航迹推算数据和惯性数据
/camera	基于 freenect 驱动的 Kinect 的数据获取与发布，包括点云数据与 tf 关系
/pointcloud_to_laserscan	虚拟 2D 激光数据发布节点，将 Kinect 的点云数据转化为 2D 激光数据
/slam_gmapping	基于 gmapping 的 2D 激光 slam 算法，得到栅格地图
/amcl	自适应蒙特卡洛定位，用于估计机器人当前位姿
/move_base	导航节点，主要包括全局和局部路径规划器与代价地图
/tf	发布机器人各连接的实时位姿及依赖关系等
/map_server	地图服务器，用来加载和保存全局地图使用
/rviz	机器人状态可视化工具节点

根据 ROS 官方给出的导航功能包集配置如图 5-14，本设计主要设计和使用的节点如表 5-3 所示，节点间订阅关系图详见附录 D 和附录 E。

5.3.3 基控制器

基控制器主要负责和下位机之间通信数据的解包和打包，实现下位机和 ROS 之间的数据交互。

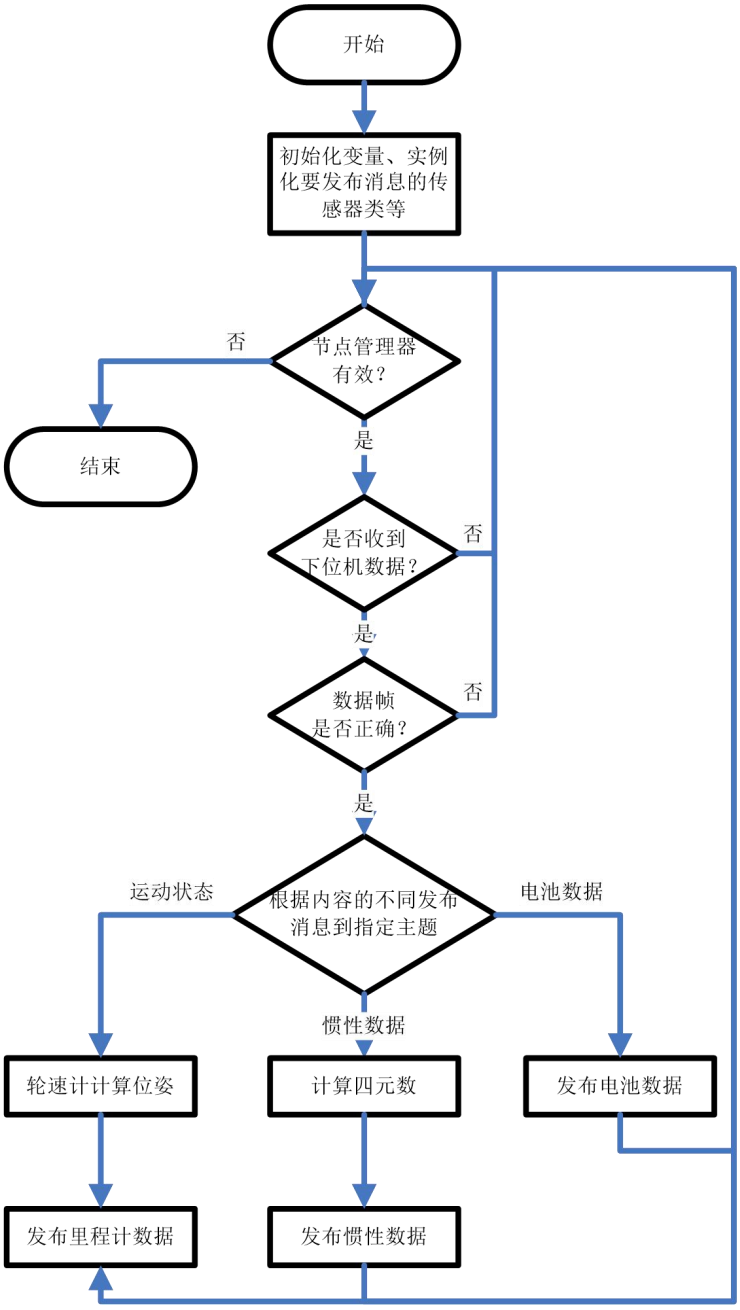


图 5-15 主程序流程图

Fig.5-15 Main program flowchart

如图 5-15 所示为基控制器的主程序流程图，在初始化后，会进入主循环中，任务管理器是指 ROS_Core④，只有该进程有效运行，其他的功能包才能运行，节点间才能进行通信。主循环中解析并计算下位机发来的数据，根据对象的不同，分别发布里程计消息

④ ROS_Core：ROS 核心进程，提供基础的节点管理、话题订阅、消息传递等。

到 odom_raw⑤话题，解析并计算下位机发来的 IMU 数据，以四元数的形式发布到 imu 话题，还解析电池、超声波数据包并发送到对应的话题。

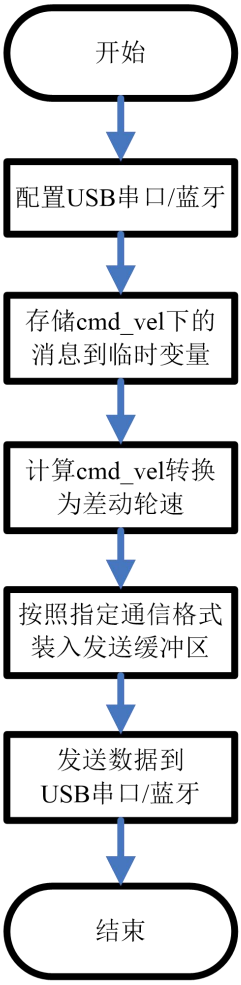


图 5-16 cmd_vel 话题回调函数流程图

Fig.5-16 cmd_vel topic callback function flow chart

如图 5-16 所示为基控制器中 cmd_vel⑥回调函数的软件流程图，当 cmd_vel 话题下的消息有更新时，系统会自动调用该程序，由于本设计中的机器人在理想状态下是在一个 2D 的平面运动，而且由第三章的分析可知，本设计的机器人没有横向移动的能力，只需要前进速度和转动速度就可以描述机器人的运动，所以程序中将 cmd_vel 中 x 轴方向的线速度和绕 z 轴角速度带入公式(3-7)中，即解算出两个驱动轮的轮速，然后通过 USB 串口或蓝牙发送至下位机。

在实验过程中发现，由于机械上的偏差，机器人的参数总是近似的，这种近似就会

⑤ odom_raw：里程计的原始数据话题，包括三维位姿和当前三维速度、角速度信息。
⑥ cmd_vel：机器人目标速度描述话题，包括三维的速度与角速度。

带来实际运动数据与期望不符的情况（如让机器人前进 1m，实际只前进 0.9m），这时候应对基控制器的线速度与角速度部分进行校准，重复让机器人前进固定的距离或旋转固定的角度，认为实际运动与预期之间的偏差是线性的（最好是通过多次的重复试验，观察偏差的分布情况），于是可以计算出一个线性的系数，这个系数极为运动机构校正系数。

5.3.4 位姿滤波器

在第三章中分析航迹推算的时候，提到由于差速驱动轮根据轮速计算机器人的角速度导致精度差的问题，所以使用了扩展卡尔曼滤波器，将惯性数据和轮速计算的位姿数据进行融合，得到更为精准的机器人位姿估计。

本设计中使用了开源的 robot_pose_ekf 功能包，该功能包订阅 odom、imu、vo(Visual Odometry，视觉里程计)类型的话题，内部滤波后发布整合后的位姿数据和 odom 到 base_link 的坐标变换到 tf 话题下。

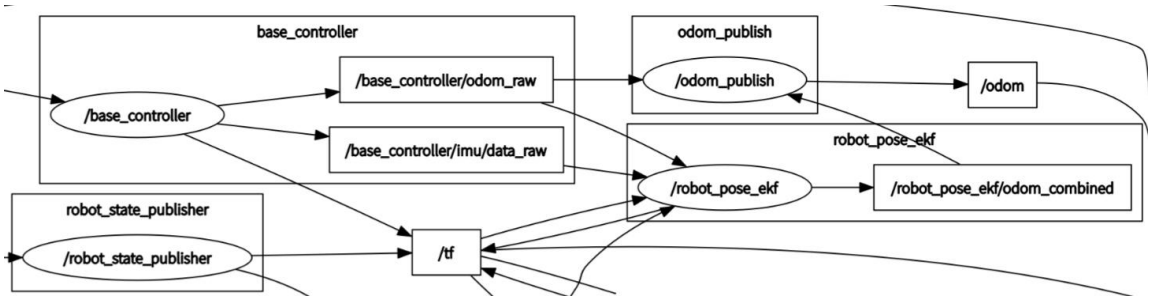


图 5-17 基控制器、位姿滤波器关系图

Fig.5-17 Base_controller and Robot_pose_ekf relation graph

本设计中编写了一个具有里程计消息重发布功能的功能包，将原始的轮速计推算的位姿信息和铝滤波后的运动状态信息整合再发布到 odom 话题下。如图 5-17 为基控制器和位姿滤波器、里程计消息重发布节点之间的发布订阅关系。

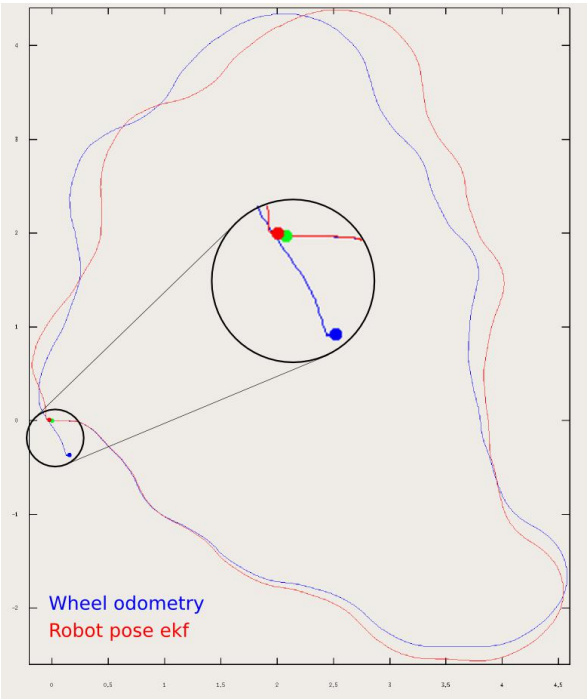


图 5-18 使用位姿滤波器前后的轨迹图

Fig.5-18 Before and after using robot_pose_ekf trajectory chart

如图 5-18 为位姿滤波器作者提供的测试结果图，所示为使用滤波器前后机器人运动一周回到原点的位姿估计偏差大小，在图中放大位置可以看到使用整合里程计数据与惯性数据的位姿滤波器后，机器人位姿估计与实际之间的偏差非常小，而单独使用里程计数据的位姿估计相对偏差较大。

5.3.5 3D 模型描述

机器人 3D 模型主要用于仿真机器人或帮助开发者简化调试过程，在 ROS 中可以通过 URDF 文件来实现对机器人的 3D 模型描述。

Urdf 是基于 xml 格式的，描述机器人几何结构的基本字段：连接（link）和关节（joint），连接描述元素的基本特性，包括形状、体积、颜色、材质纹理，关节描述了两个元素的依赖关系，包括运动类型（转动、固定、旋转、平面运动等）、相对位置、相对角度等。

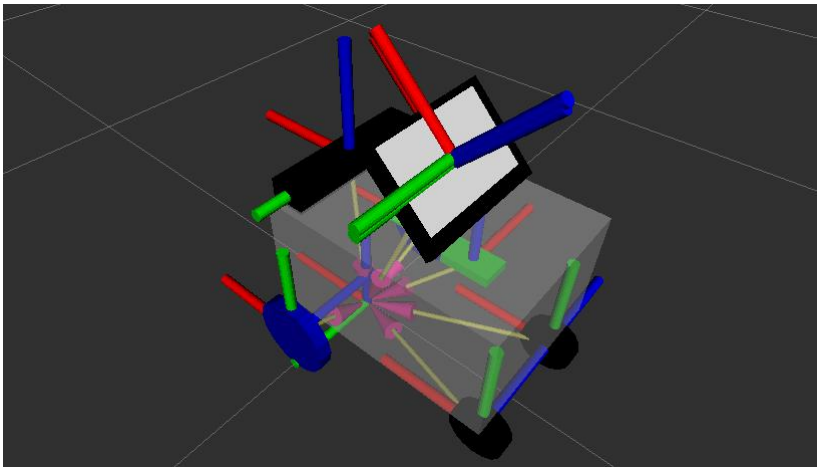


图 5-19 3D 机器人模型

Fig.5-19 3D robot model

如图 5-19 所示为本设计中的机器人 3D 模型，其中主要包括底盘、轮子、下位机、Kinect、显示器等元素的 3D 描述。

5.3.6 激光数据

本设计中在 ROS 上安装开源的 freenect 驱动来获取 Kinect 传感器数据。

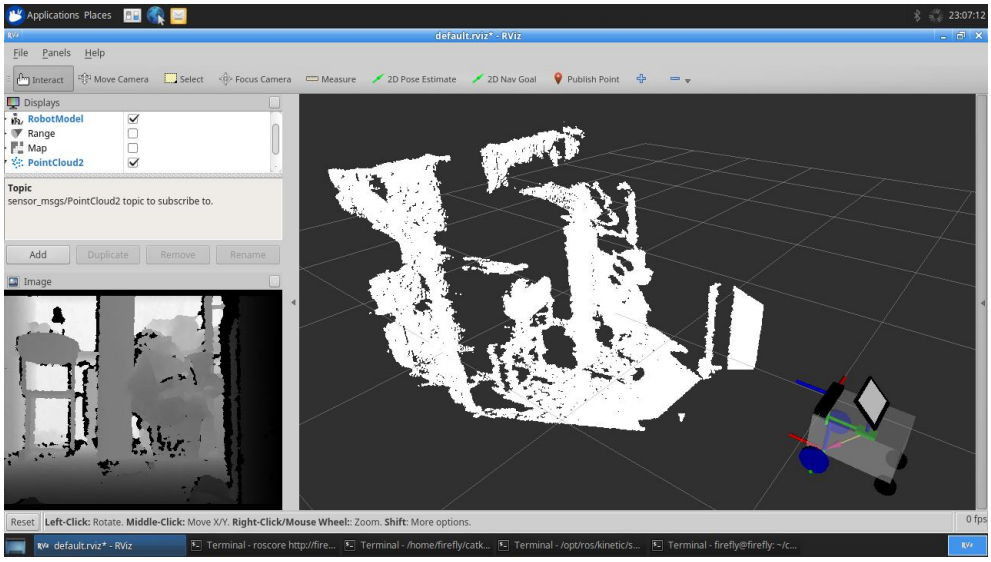


图 5-20 Kinect 点云图像

Fig.5-20 Kinect point cloud image

如图 5-20 所示在 ROS 中使用 rqt_rviz 显示 Kinect V1 获取的点云图像，其中左下角为当前深度图像，同时 freenect 还提供发布相机上各部件的 tf 关系。

由于 Kinect 并不能直接输出激光数据，只能输出点云数据，于是通过开源的

pointcloud_to_laserscan 功能包将订阅 Kinect 的 3D 点云数据转换成 2D 的激光数据并发布在 scan 话题下，该功能包将竖直方向每一列点云中离 Kinect 相机最近的像素作为 2D 激光中的障碍物距离，将所有列中的最小值投影到机器人坐标系所在平面。

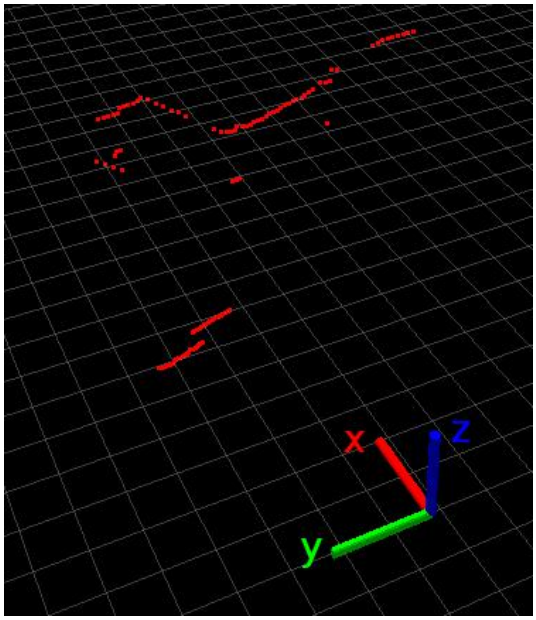


图 5-21 Kinect 2D 激光数据图像

Fig.5-21 Kinect 2D laserscan diagram

如图 5-21 所示，将 Kinect 点云数据经过转化后得到的 2D 激光数据，表示障碍物与相机间的连续距离关系数据。

5.3.7 SLAM 与导航

该部分主要借鉴成熟的开源算法功能包，通过设置部分针对机器人特性的参数实现 SLAM 与导航等功能，极大的降低了开发的难度。

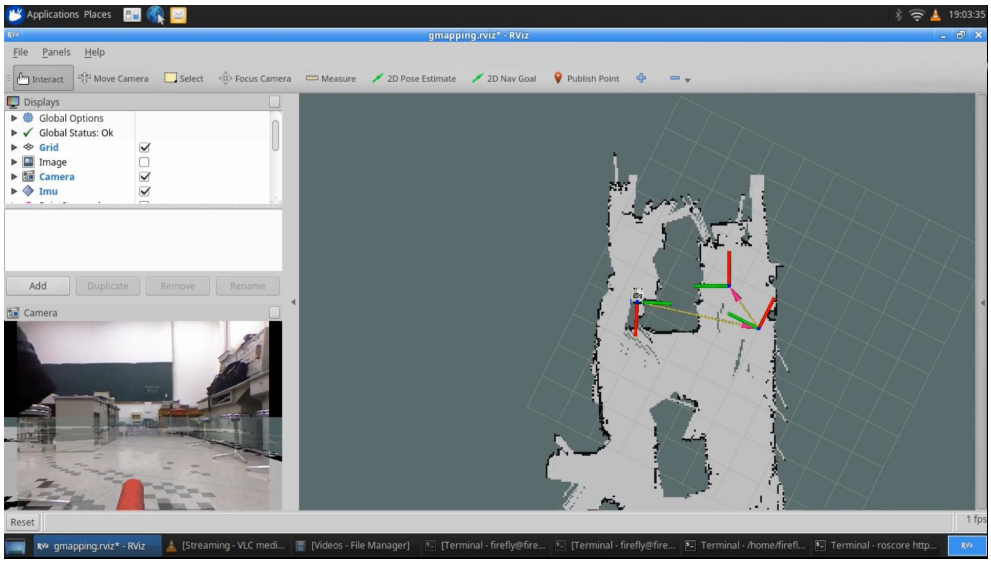


图 5-22 使用 gmapping 算法建图

Fig.5-22 Mapping by using gmapping algorithm

如图 5-22 所示为使用 slam_gmapping 开源功能包的建图情况，该节点需要/tf（坐标变换信息）和/scan 话题（激光扫描数据），然后输出到/map 话题（即栅格地图）。左下角为 Kinect 所看到的 RGB 图像与地图、tf 等融合后的实时图像。

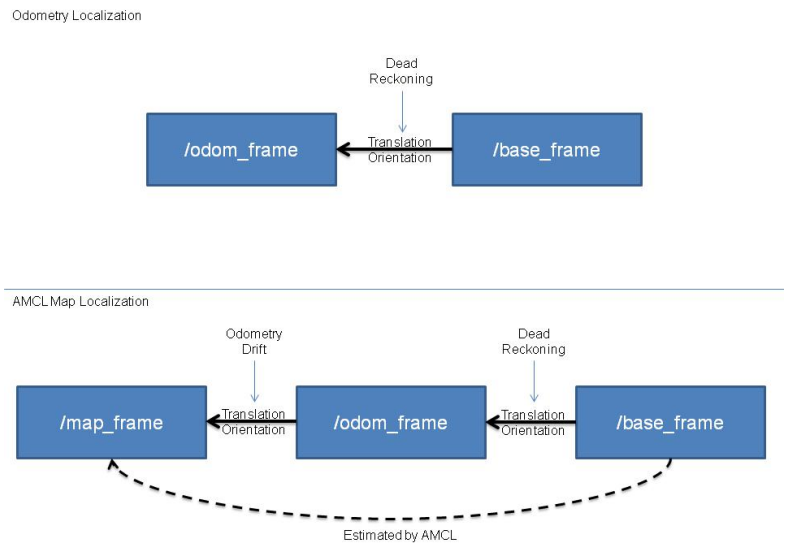


图 5-23 里程计与 AMCL

Fig.5-23 Odometry and AMCL

在 odom 坐标系中机器人位姿可以任意变化，没有任何限制。这种特性就使得 odom 坐标系无法作为长期的全局参考。但是 odom 坐标系中的机器人位姿姿态能够保证是连

续的，这意味着在 odom 坐标系中机器人的位姿的变化是平滑而且连续的。如图 5-23 所示，在本设计中 odom 与机器人底盘之间的位姿变化是基于轮速计和惯性模块的测量计算得到的。

map 坐标系是一个世界固定坐标系，其 Z 轴指向上方。相对于 map 坐标系机器人的位姿，不应该随时间显著移动。而且 map 坐标是不连续的，这意味着在 map 坐标系中移动平台的姿态可以随时发生离散的跳变。但是基于外部传感器的不断检测和重新计算世界坐标系中机器人的位姿，可以避免像轮速计定位这种基于内部传感器的方法带来累计误差。如图 5-23 所示本设计中 map 与机器人之间的位姿关系是基于 Kinect 传感器得到的激光数据使用 AMCL 来计算的，AMCL 的粒子滤波的主动定位算法功能包，用于在已知地图的情况下快速估计机器人的当前位姿[13]。

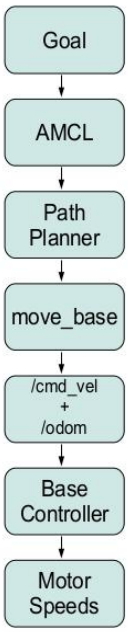


图 5-24 导航控制层级图

Fig.5-24 Navigation control level diagram

如图 5-24 所示为机器人导航功能的层级关系，描述了从设置一个目标位置后到电机执行速度变化的流程。move_base 是 ROS 提供的导航避障功能包，该软件根据你发布的传感器信息做出路径规划以及机器人的速度和转向控制，而如图 5-13 所示，move_base 包内部集成了全局和局部的路径规划器与代价地图[14]。

表 5-4 导航参数配置文件

Tab.5-4 The navigation parameter configuration file

base_local_planner_params	基本局部路径规划器参数
costmap_common_params	代价地图公共参数
global_costmap_params	全局代价地图参数
local_costmap_params	局部代价地图参数

如表 5-4 所示为本设计中导航功能包的参数配置文件说明，move_base 使用前需要配置一些参数：运行成本、机器人半径、到达目标位置的距离、机器人移动的速度、最大探测距离限制等，这些参数都在这些 yaml 格式的配置文件中。通过这种描述文件的方式可以更方便明了的设置导航参数，同时也便于机器人间代码的移植。

5.3.8 典型启动文件

当机器人系统复杂起来时，会有数十个甚至上百个节点，那么在启动这些节点时如果每一个节点都需要手动输入命令来启动将是十分费时的，所以 ROS 提供了一种可以批量启动节点、设置节点启动参数的 launch 描述文件。

其中典型的 slam 启动文件中启动了如下节点：地图服务器、键盘遥控器、位姿滤波器、基控制器、slam_gmapping、rviz 可视化界面、点云转 2D 激光数据、move_base 节点等。相比之下导航启动文件中仅 gmapping 节点被 amcl 节点所替代，其余部分基本相同。

5.4 通信格式设计

为了使数据交互稳定和规范，使底盘更具通用性,本设计中提出了一种上下位机之间的可扩展和裁剪的通信协议，格式如下：

表 5-5 通信格式定义

Tab.5-5 The communication protocol format

长度	1 字节	1 字节	1 字节	1 字节~28 字节	1 字节
功能	帧头	对象	指令	参数段	帧尾
内容	0XA5	0X01 蜂鸣器 0x02 运动 0x03 IMU 0X04 欧拉角 0X05 超声波 0X06 电池 0X07 红外* 0X08 GPS*	0X80 写入 0x81 读取 0x82 反馈	取决于不同的对象和指令	0XAA

如表 5-5 所示为本设计中上下位机的通信格式定义，数据帧的帧头 0xA5 和帧尾 0XAA 的二进制表示分别是 10100101 和 10101010，两个字节传输过程中在硬件电平上的变化都是十分频繁的，如果硬件电路有问题就会首先在这两个字节上表现出来，基本可以认为是非常容易传输出错的两个字节（但并不限于这两个，还包括 0X5A、0x55），所以用这两个字节作为帧头和帧尾的标识不仅可以方便接收方对数据定位，还可以用来做简单的正确性校验。

对象字节用来描述本数据帧是针对那个部分，指令字节用来描述本数据帧的属性，参数段的内容分别对应本次操作的具体数据内容；指令字节表示了本数据帧的性质，读、写或反馈；参数段部分是要传递的具体数据内容，其长度和大小取决于对象和指令；完整的通信格式参数定义详见文末附录 A。

6 通用性设计

为使本设计中的移动机器人更方便裁减和扩展，设计之初考虑一部分的通用性设计。

机械设计方面，预留了较多的空间，可以加装更多传感器、执行器，甚至安装机械臂，并且机器人支架多孔洞的角钢可以更加方便的固定这些设备。

下位机硬件方面，STM32 产品线丰富，多种配置的芯片可根据实际需求供升级或降级选择，可以方便的移植到其他性能不同的 STM32 平台上；预留 GPS 接口和固定位置，加装 GPS 模块后可以让系统应用于室外智能机器人领域；预留红外接收头接口，主要可以用来实现自动充电的光源寻找，充电插座的精准定位；超声波数量可以自定义，最多可支持同时接入 8 路超声波；主控板嵌入高性能 DC-DC 模组，实现宽电压范围输入，且方便检测电池电压；同时支持 USB 虚拟串口、蓝牙、TTL 三种方式的通信，适配多种类的计算平台接口；预留多组 USART 接口可以用来控制 Dynamixel 系列舵机实现机械臂设计和通用 GPIO 接口。

下位机软件方面，高内聚低耦合的软件框架，添加或删除部分功能模块都不容易因为函数间耦合引起工程问题；可扩展可裁减通信格式，后续可以任意增减通信格式的内容，降低跨平台移植软硬件的开发成本。

上位机的计算平台接口丰富，可以接入多种周边设备（如 USB 设备、声音设备、视频设备、存储设备、LTE 网络等），支持双摄接口，甚至可以尝试做双目视觉导航；

上位机基操作系统使用 Ubuntu，可以运行多种开源的开发工具或程序，同时搭载的 ROS“高内聚地耦合”的软件组织方式使得功能与功能之间的接口十分统一，只要在遵循接口标准前提下的代码都可以快速移植开发，让机器人功能包软件更具通用性；而且 ROS 支持分布式计算，在协同开发项目或多计算机协同工作的情况下，ROS 的通用接口使得这一问题不需要刻意解决。

7 实验过程及分析

7.1 调试工具

7.1.1 下位机

表 7-1 下位机调试工具

Tab.7-1 Lower machine debugging tools

工具	功能
IAR	嵌入式处理器的 IDE，支持编辑、编译、调试等功能。
Logic	电平逻辑分析软件，配合硬件逻辑分析仪的控制，可以将前端的硬件逻辑分析仪采集到的信号电平变化进行采集、存储、分析等操作，还可以用来分析通信协议内容。
Hatek-6022BE	青岛汉泰公司生产的一套虚拟示波器软硬件系统，包括 PC 端运行的示波器软件和前端的信号采集器。

如表 7-1 所示为下位的主要调试工具，皆运行在 PC 端，部分需要借助外扩硬件实现。

7.1.2 上位机

1）在 PC 端，主要实现对终端的远程命令和文件操作：

SecreCRT：提供 PC 端以命令行的方式通过 ssh 连接登录远程终端的工具，直接对远程终端进行命令操作；

FlashFXP：提供给 PC 端与远程终端实现基于 ftp 协议的文件传输的工具；

2）在计算平台端，主要实现对机器人程序的开发和调试工作：

Comcute：在 Firefly-RK3399 单板计算机中用来发送串口数据和查看串口发来的数据的工具软件；

Eclipse：支持多种语言的集成开发环境，支持众多插件，使得 Eclipse 拥有较佳的灵活性；

Catkin：ROS 官方的软件包编译系统，结合了 Cmake 宏和 Python 脚本提供了 Cmake 的正常流程，但是增加了自动发现包等功能；

表 7-2 rqt_gui 典型插件

Tab.7-2 rqt_gui typical plugins

插件	功能
rqt_rviz	使用 openGL 渲染的 3D 可视化的图形调试工具，可以查看点云、坐标、地图、里程数据、imu 数据、激光数据、图像数据、机器人 3D 模型等。
rqt_plot	曲线绘制工具，可视化时域上某变量的变化过程。
rosbag	记录机器人各节点间的对话过程与内容，在回放数据的时对机器人各节点的数据仔细分析，大大缩短了调试时间。
rqt_graph	动态绘制 ROS 绘画的当前状态，图形化表示发布者-订阅者在主题与节点之间的连接网络。
rqt_reconfigure	图形化界面的动态配置节点参数，方便调试。
rqt_console	日志记录工具，支持日志不同层级输出，支持可视化工具过滤和查看。

rqt_gui 是由 ros 官方提供基于 qt 的图形化调试界面的集合，如表 7-2 所示为 rqt_gui 包括的典型插件。

3）在安卓端，主要实现对机器人的无线遥控和数据查看的工作：

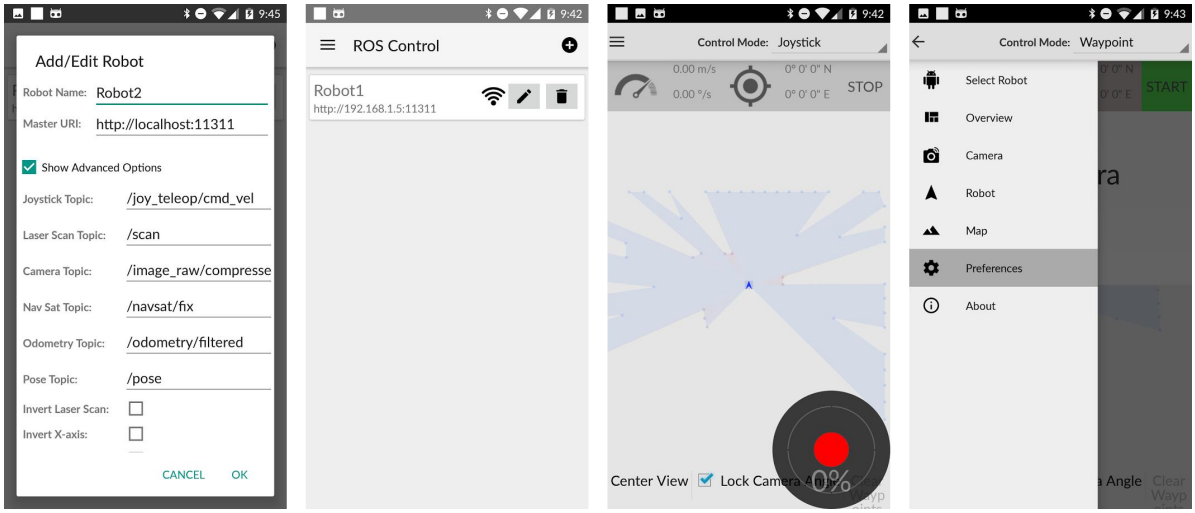


图 7-1 ROS Control 软件运行界面截图

Fig.7-1 ROS Control software running interface screenshots

如图 7-1 所示为安卓端调试工具 ROS Control，由开源爱好者们提供的 Android 客户端，支持地图查看、深度或彩色图像显示、速度方向控制等功能，只需从 Github 下载他

们的源码后使用谷歌公司的 Android Studio 即可进行编辑和开发，编译生成 APK 的安装文件[15]。

7.2 SLAM

在启动 gmapping 的 launch 文件后，对应的节点诸如基控制器、位姿滤波器、Kinect 驱动、点云转换激光、可视化界面等被启动，此时使用 Android 客户端对应的 ROS Control 软件通过无线局域网络连接至计算平台，作为一个分布式节点接入至 ROS 中，通过虚拟摇杆控制机器人的运动，此时机器人在行走过程中会即时估计自己的位姿并构建增量式地图，该过程即称为 SLAM。

在设计初期尝试过使用 hector_slam 算法，该算法在设计之初是依赖高精度的激光雷达的，很快发现由于 Kinect 的可视角度太小，噪点过多的问题，导致建立的地图几乎无法使用，于是将 SLAM 算法更换为 gmapping，效果改善明显。

图 7-2 实验环境实拍

Fig.7-2 The experiment environment to film

如图 7-2 为辽宁工程技术大学静远 116 实验室环境实拍，图中白色线条描述了场地的基本轮廓。

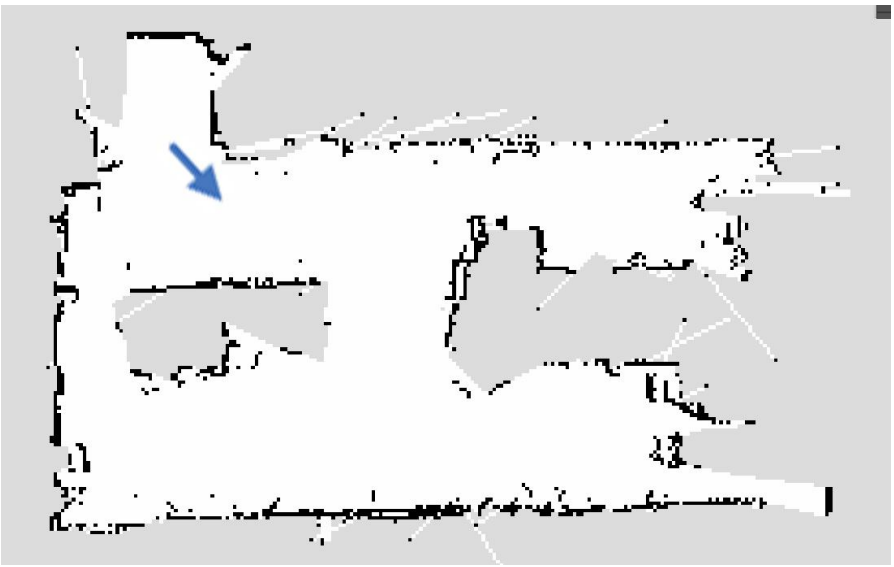


图 7-3 机器人构建的地图

Fig.7-3 Robot build the map

图 7-3 为手动控制机器人使用 Kinect 作为传感器，在实验室内遍历一周后构建的 2D 栅格地图。图中灰色区域表示未知，白色区域表示已知通行区，黑色区域则是探测到的固定障碍物，其中图 7-3 左上角箭头的起始位置和方向即为图 7-2 的拍摄位置与角度。

可以看到机器人的全局地图与实际的实验环境轮廓基本符合，但是在角落或细节描述上还有缺陷，主要原因是轮速计导航的精度不够高，Kinect 的可视角度小和噪点多造成的，如果提高轮速计的精度和使用高精度、宽视角的激光雷达将有效解决这个问题。

7.3 导航

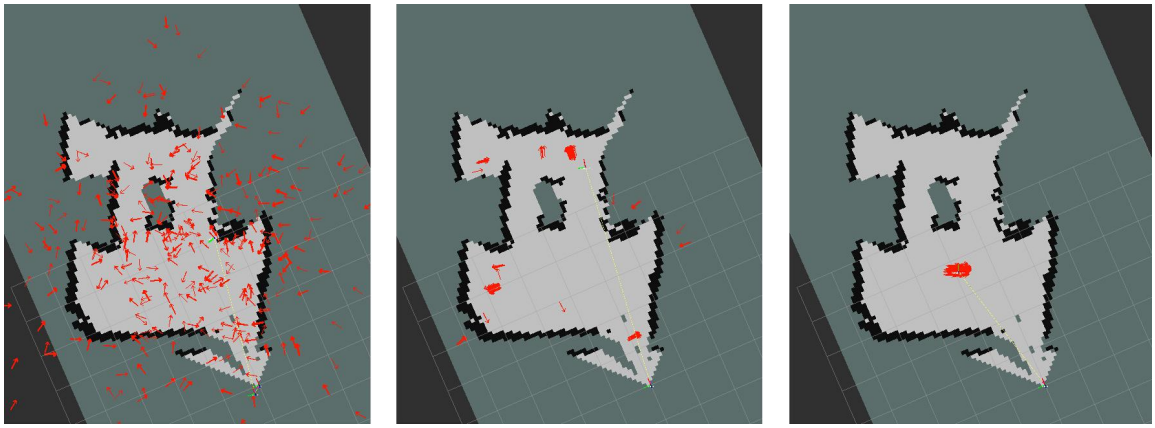


图 7-4 AMCL 粒子可视化

Fig.7-4 AMCL particle visualization

在已知地图的环境中，使用 AMCL 对机器人位姿进行估计，如图 7-4 所示为使用 AMCL 算法估计机器人位姿时粒子状态的可视化，图中的红色小箭头即为用于位姿估计的离散分布的粒子的可视化表示，箭头越集中表示估计的位姿和实际位姿的偏差越小，左图为初始化一批随机的粒子，中间图片表示机器人在运动过程中粒子趋于某几个值，右图表示经过一段时间的运动后，所有的粒子位置和方向几乎相同，此时得到了较为准确的机器人位姿描述。

在实验过程中发现，粒子数量的多少对位姿估计的准确性以及定位时间影响明显，但由于机器人的运算能力有限，经过多次尝试，当初始粒子数量在 200 左右时，计算平台可以即可保证定位速度、定位准确性且不会卡顿。当初始粒子数量继续增加的时候运算量的增加让运算平台显得有些吃力，对于定位的效果帮助并不大。

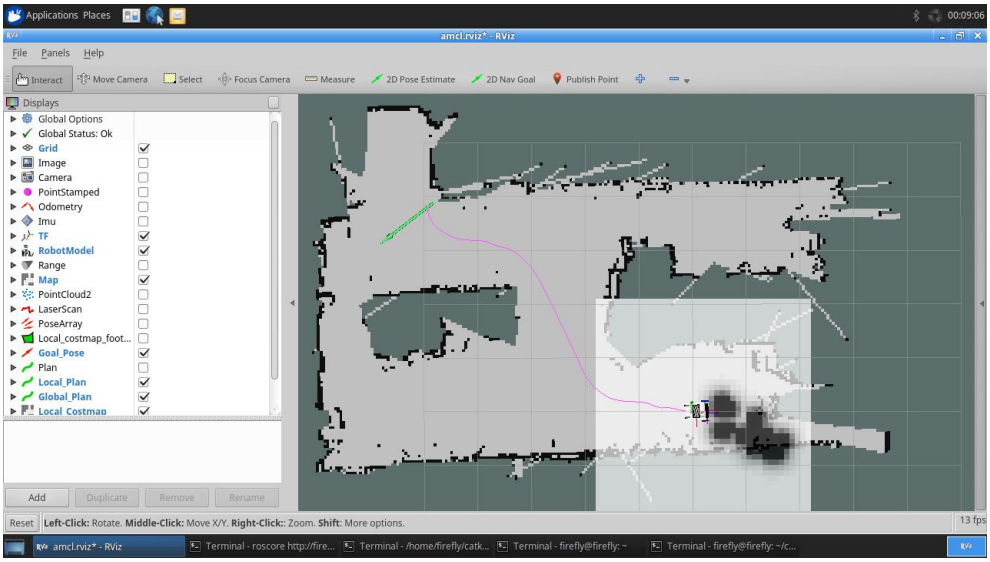


图 7-5 机器人路径规划图

Fig.7-5 Robot path planning diagrams

如图 7-5 为机器人的路径规划时的截图，左上角箭头为人为给定的机器人目标位姿，机器人根据当前的已知地图和位置，已经计算出一个最优路径。同时在右下角的灰色区域内表示为机器人的局部代价地图，用于局部导航，根据传感器实时更新。

在实验过程中发现机器人对于突然出现在眼前的障碍物躲避能力不强，原因在于 Kinect 的深度传感器只支持 0.4m~4m 的探测距离，所以机器人前方 0.4m 的范围基本是盲区，另外一个原因是路径规划器的规划频率，但是如果路径搜索频率过高，会导致计算量的大幅度提高，并且如果路径频繁变化会导致机器人不易抵达目标位置，所以在快速达到目标位置和快速响应临时避障之间要相互权衡。

8 技术经济分析

移动机器人的底盘使用轻质铝合金板，保证刚度前提下降低了整车的重量，相比之下铝合金板的加工难度也小于钢板、铁板等；机器人支架使用镂空角钢，成本低廉，可以任意剪裁，像积木一样自由拼装，而且角钢上的孔洞方便在未来可以扩展设计。

近年来 STM32 的市场占有率逐步上升是有目共睹的，凭借自身的高性能、低价格、多兼容、低功耗、高稳定、快开发的优势挤压了 8 位机和 16 位机的市场，足以证明 STM32 是目前（2017 年）最适合嵌入式开发的微处理器之一。

在计算平台上，选择了性价比较高的开源 ARM 单板计算机，在保证运算性能的前提下尽可能降低了硬件成本，而在设计过程中也能证明 Firefly-RK3399 单板计算机是可以完成本设计中提到的智能机器人任务，而机器人的眼睛采用了相对“妥协”的方案，使用了相比较为成熟的微软 Kinect，尽管在可视角度与噪点控制上比激光雷达稍逊一筹，但它有着驱动程序稳定、可参考代码丰富、性价比高等优点，也已经满足本设计的需要。

在下位机软件设计上，使用 ST 官方提供的标准驱动函数库，让开发者和底层硬件、寄存器等隔离开来，这样可以极大的加快开发进度。

在上位机软件上使用开源的 Ubuntu 和 ROS 程序，Ubuntu 具有类 Linux 系统的稳定性和安全性，同时还具有类 Windows 系统的友好操作界面，ROS 甚至是支持商用的机器人操作系统，提高了代码的复用率，缩短了机器人开发周期。

9 结论

本设计主要完成底盘、电机、传动等单元的组装和调试，对机器人运动学模型以及航迹推算、SLAM、定位、路径规划等算法进行对比分析；设计了包括 STM32 主控制器和超声波、惯性测量、电机驱动器、RS485 收发器、蓝牙模块等多种外围器件组合构成的通用的下位机系统，以及 Firefly-RK3399 和 Kinect 等组成的上位机系统；上位机中 Ubuntu 作为母系统提供框架和人机界面，ROS 作为次级操作系统负责机器人功能包的调试、通信等功能，并通过 ROS 实现了 SLAM、自主定位、路径规划等预期的功能；但在将这些功能稍作调整后应用在无人驾驶、服务机器人、AGV、扫地机器人等产品之前，系统的稳定性和可重复性还有待提高。同时本设计中还存在以下不足：下位机如果运行嵌入式操作系统可以进一步提高系统的通用性；如果 PID 闭环在 STM32 中完成，可使用价格更低的不带主控驱动器；编码器接在电机的减速后的输出轴上，如果和变速前的电机轴连接将提高精度；通信协议还有待改善，加入长度校验、数据完整性校验等功能；机械结构在体积上还有进一步减小的可能；

在智能机器人算法领域上，仍让有很多问题等待解决，诸如：人脸识别、室内 3D 重建、八叉树地图、vSLAM 等等，这些关键技术对于智能机器人的发展是至关重要的，笔者水平和知识面有限，本论文权当抛砖引玉。

致谢

感谢我的毕业设计导师王焱老师，在毕业设计过程中离不开王老师的全力支持和耐心帮助，在设计和撰写论文过程中提出很多优秀的建议，才使得本设计得以顺利完成。同时感谢老师、同学、室友给予的支持和帮助，这并不是我一个人的设计，更像是一群人合作的设计，我更像这个设计的主要负责人。

感谢创客智造网站在设计、调试过程中提出大量的建设性意见；感谢 Firefly 团队在硬件资源不足的情况下免费提供开发工具使设计工作得以继续；感谢北京海风智能科技股份有限公司在实习期间让我接触到移动机器人嵌入式开发工作，为本设计做了扎实的铺垫；感谢开源团队的无私贡献，包括但不限于以下：Github、ROS、Ubuntu、Openslam、Firefly、泡泡机器人、STM 等等，篇幅有限无法将开源世界里热爱奉献的团队一一列出，再次感谢以上各团队和组织的帮助和支持。

感谢计算机、数学、电子、机械、互联网等领域前辈的奉献，才让我们有机会站在巨人的肩膀上迈一步；感谢这个时代和政策，能让我有机会深入研究热爱的领域，能有机会在某个领域实现自我价值，本文权当抛砖引玉，希望本文的一点小尝试能给后人带来一些启发。

每当我人生中面临重要抉择时，我的父母一直是我最大的支持者，无论是学习方向、报考院校还是择业方向、发展计划等等，每一次我的父母都十分尊重我的想法，并且会提出可行性建议和提供经济支持，这是我一生中最大的幸运与慰藉。

四年时间飞逝，转眼即将踏出校门，感慨万千，当想起“永远年轻，永远热泪盈眶”那一刻，我便明白，这并不是结束，是另一种开始。

参考文献

[1] 高翔 张涛.视觉 SLAM 十四讲：从理论到实践[M].北京:电子工业出版社,2017.

[2] 权美香, 朴松昊, 李国. 视觉 SLAM 综述[J]. 智能系统学报, 2016, 11(6): 768-776.

[3] （德）托马斯·布劳恩（Thomas Braunl）著.嵌入式机器人学 基于嵌入式系统的移动机器人设计与应用=mobile robot design and applications with embedded systems[M].西安交通大学出版社,2012.

[4] （美）R. 西格沃特（Roland Siegwart），（美）I. R. 诺巴克什（Illah R. Nourbakhsh）著,李人厚译.自主移动机器人导论[M].西安交通大学出版社,2006.

[5] 陈慧岩,熊光明,龚建伟.无人驾驶汽车概论[M].北京理工大学出版社,2014.

[6] 李智. 基于 Kinect 的同时定位与地图构建的研究[D]. 北京交通大学, 2016.

[7] 郑梓均. 基于 ROS 系统的简易服务机器人关键技术的研究[D]. 江南大学, 2016.

[8] 刘火良, 杨森. STM32 库开发实战指南[J]. 机械工业出版社, 2013.

[9] 贺白羽, 蒋蓁. 基于体感传感器的室内机器人定位研究[J]. 机电一体化, 2012 (8): 13-17.

[10] 李建勇, 刘雪梅, 李雪霞, 等. 基于 ROS 的开源移动机器人系统设计[J]. Journal of Mechanical & Electrical Engineering, 2017, 34(2).

[11] 张建伟, 张立伟, 胡颖, 等. 开源机器人操作系统——ROS[J]. 2012.

[12] MartinezA F E. ROS 机器人程序设计[J]. 2014.

[13] 李卫成, 汪地, 宗殿栋, 等. 基于 RGBD 的机器人室内 SLAM 与路径规划系统[J]. 工业控制计算机, 2016 (2016 年 03): 77-78, 81.

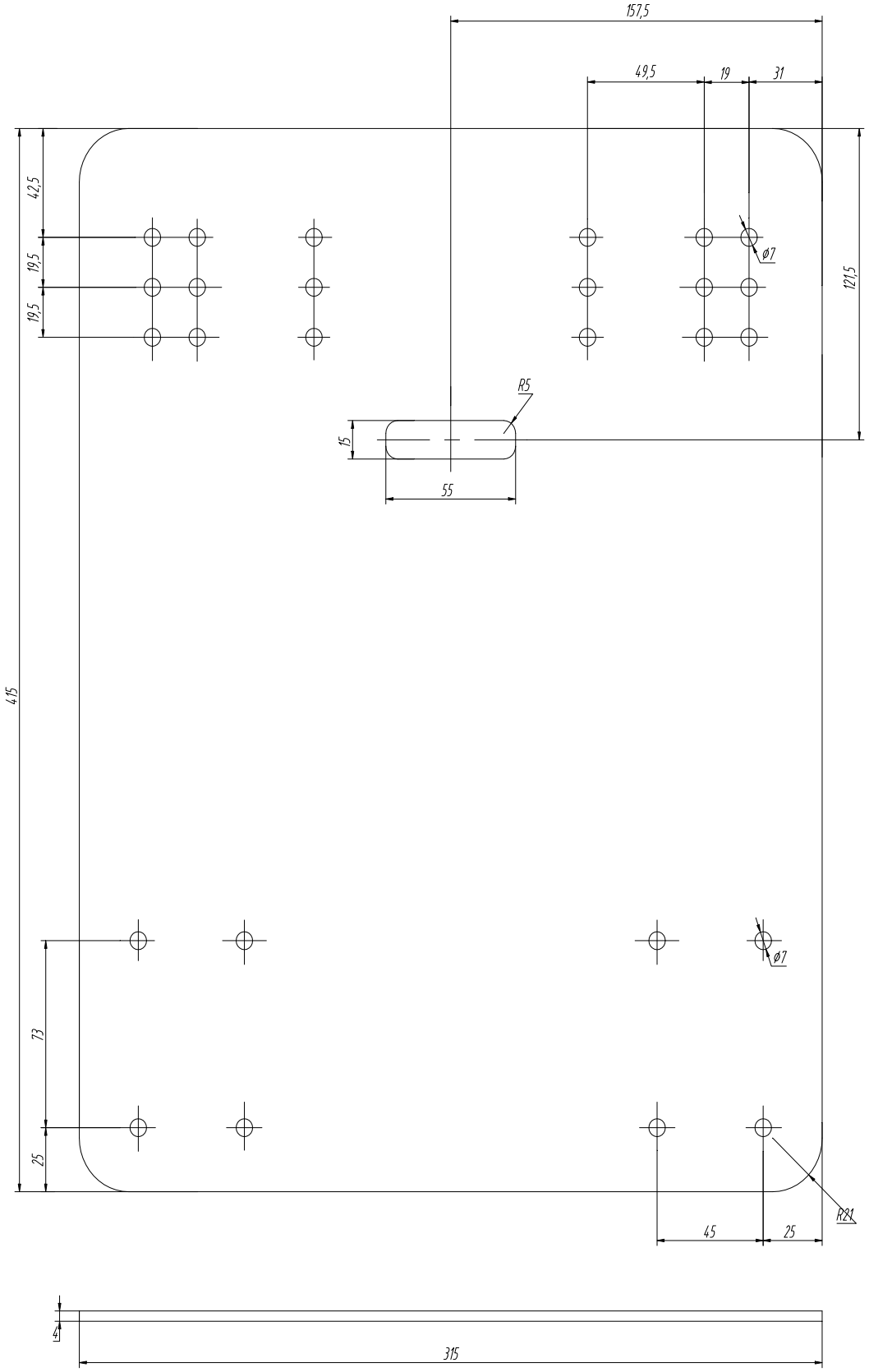
[14] Fairchild C, Harman T L. ROS Robotics By Example[M]. Packt Publishing Ltd, 2016.

[15] Lentin Joseph.ROS Robotic Projects[M].Birmingham:Packt Publishing imited, 2017.

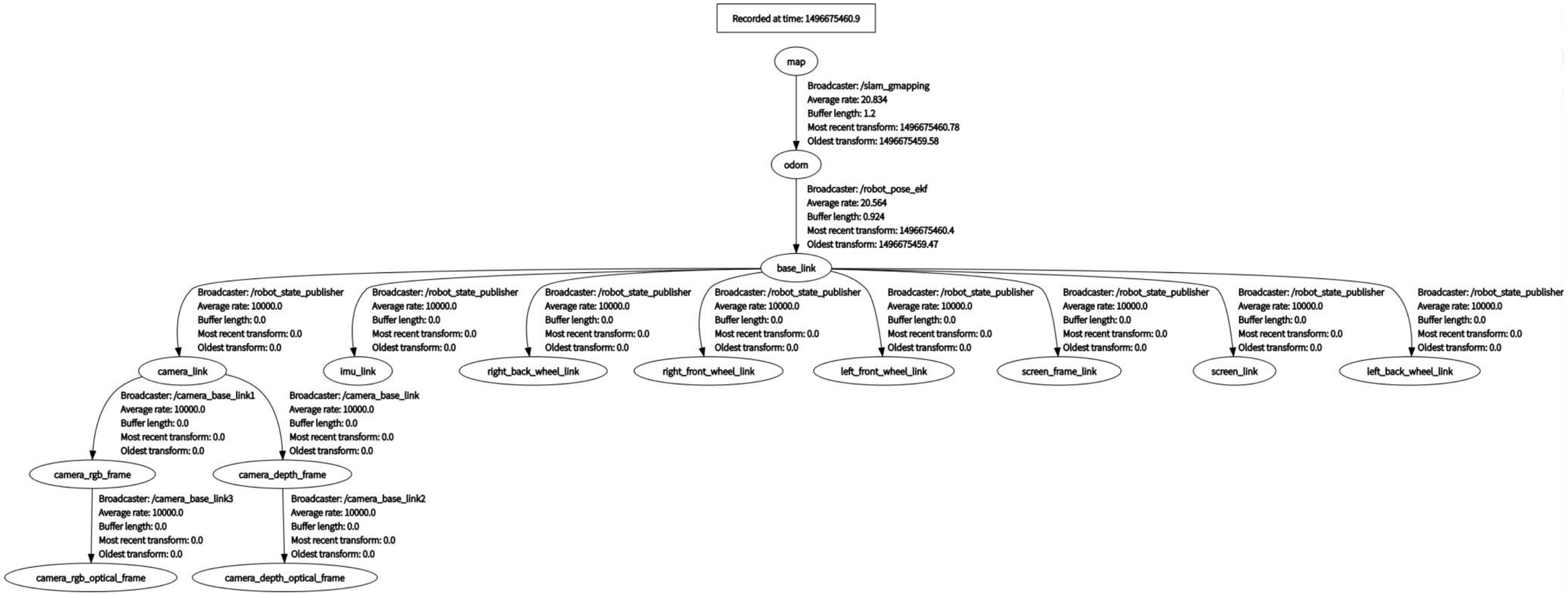
附录 A 通信格式定义

对象	指令	参数段	备注
0x01 蜂鸣器	0x80 写	时长 n（1 字节）	鸣叫时间=n*100ms
0X02 轮速	0x80 写	左轮速（2 字节） 右轮速（2 字节）	16 位整型数据 单位 mm/s 低字在前高字在后
	0x82 反馈	左轮速（2 字节） 右轮速（2 字节） 左脉冲数（4 字节） 右脉冲数（4 字节）	16 位整形数据 单位 mm/s 低字在前高字在后 32 位无符号整形数据 低字在前高字在后
0X03 IMU 数据	0x82 反馈	Acc_x（2 字节） Acc_y（2 字节） Acc_z（2 字节） Gyro_x（2 字节） Gyro _y（2 字节） Gyro _z（2 字节）	16 位整型数据 低字在前高字在后 16 位分辨率 Acc: - 2G~+2G Gyro: -2000°～ +2000°
0X04 欧拉角	0x82 反馈	q1（2 字节） q2（2 字节） q3（2 字节） q4（2 字节）	16 位整型数据 低字在前高字在后
0x05 超声波	0x82 反馈	距离 1（2 字节） 距离 2（2 字节） 距离 3（2 字节） . . . 距离 8（2 字节）	16 位无符号整型数据 单位 CM （509 表示超出检测范围或未介入传感器） 低字在前高字在后
0x06 电池	0x82 反馈	电压 m（2 字节）	16 位无符号整型数据 实际电压=m/100 v 低字在前高字在后
0X07 红外*	0x82 反馈	强度（2 字节）	16 位无符号整形数据 红外强度（最大 4095） 低字在前高字在后
0X08 GPS*	0x82 反馈	经度（4 字节） 纬度（4 字节） 高度（4 字节） 航向（4 字节） 速度（4 字节）	32 位无符号整型数据 低字在前高字在后

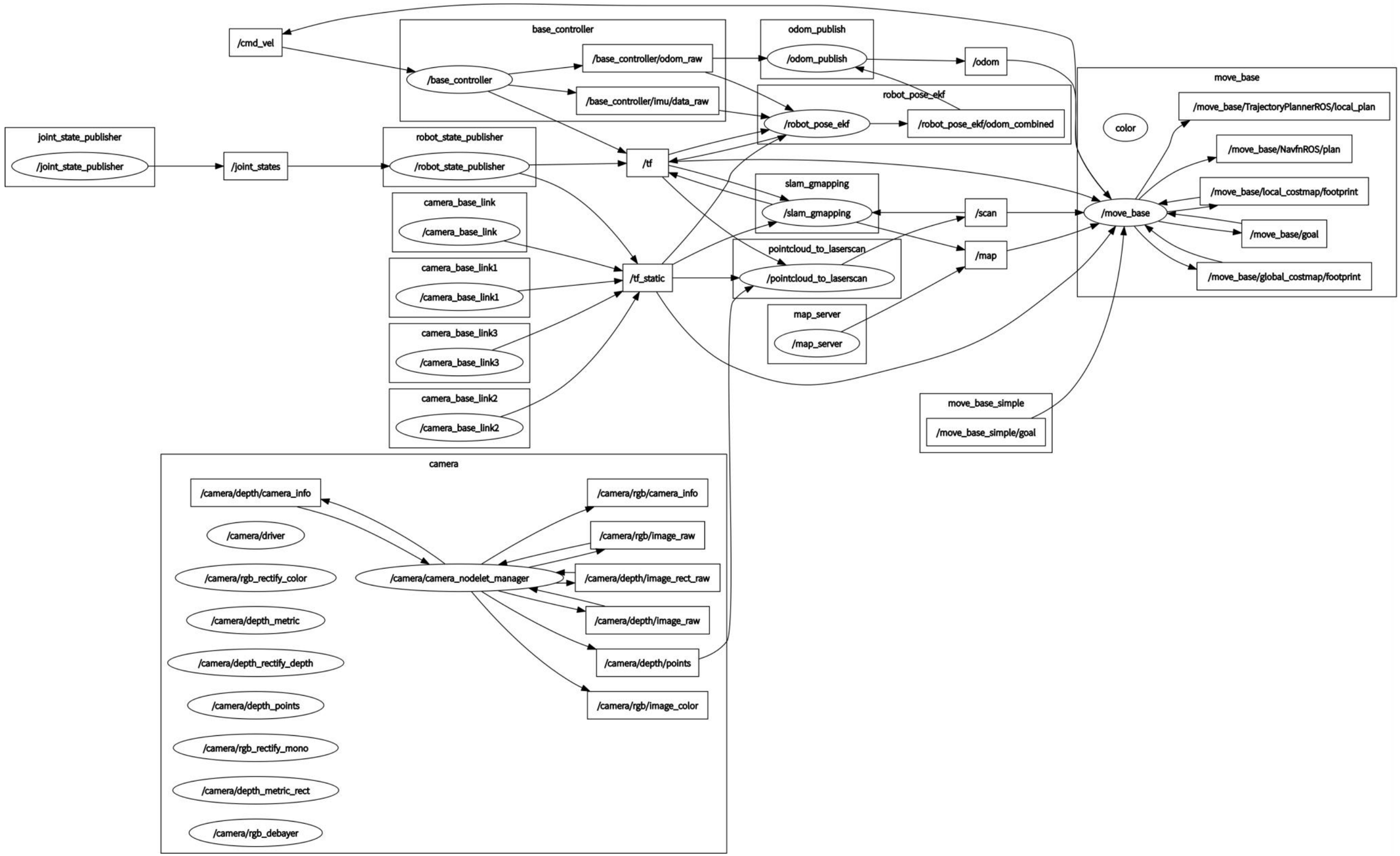
附录 B 底盘尺寸图


157,5
49,5
19
31
42,5
19,5
19,5
ø7
121,5
R5
15
55
415
ø7
73
25
R21
45
25
4
315

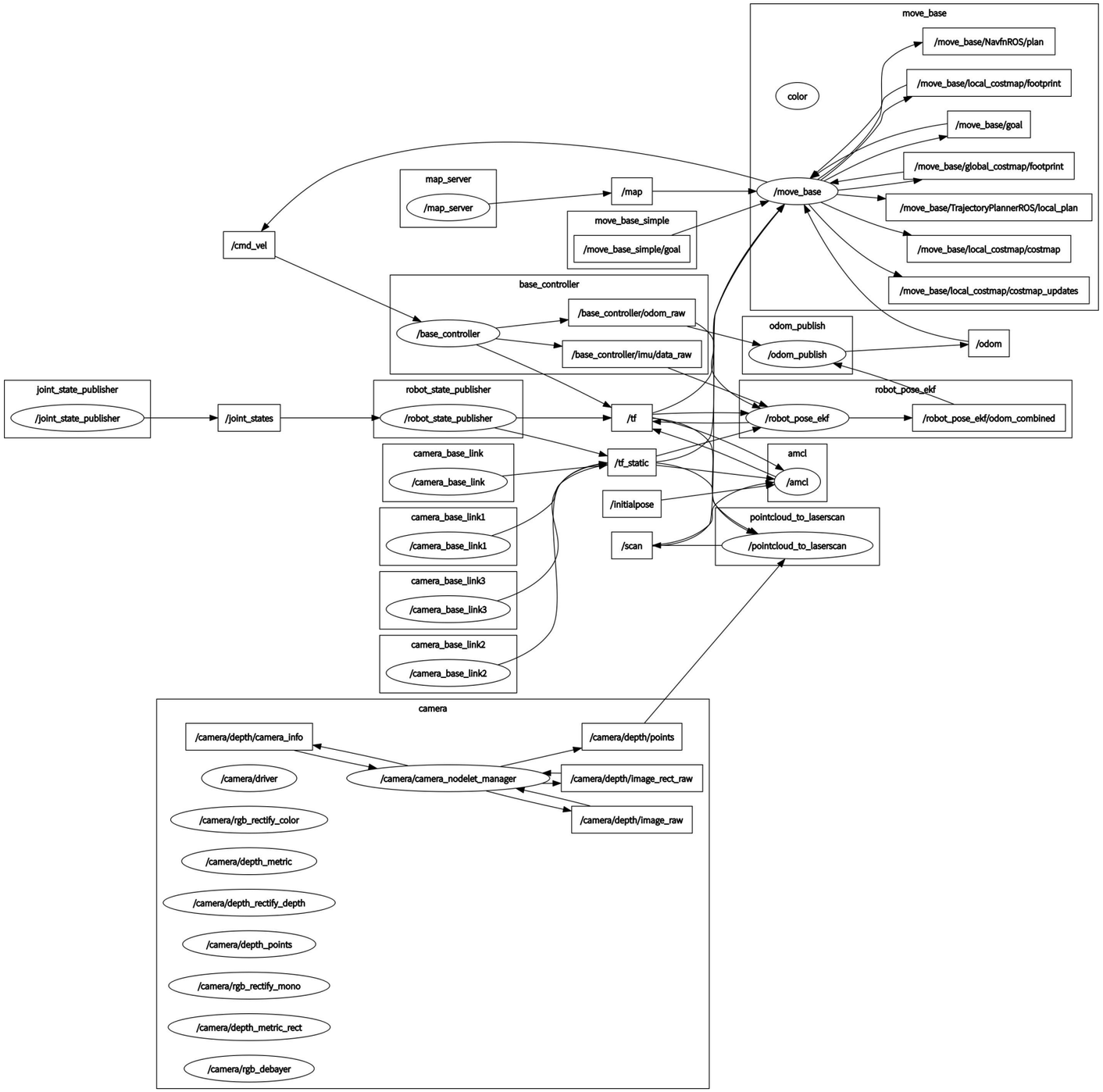
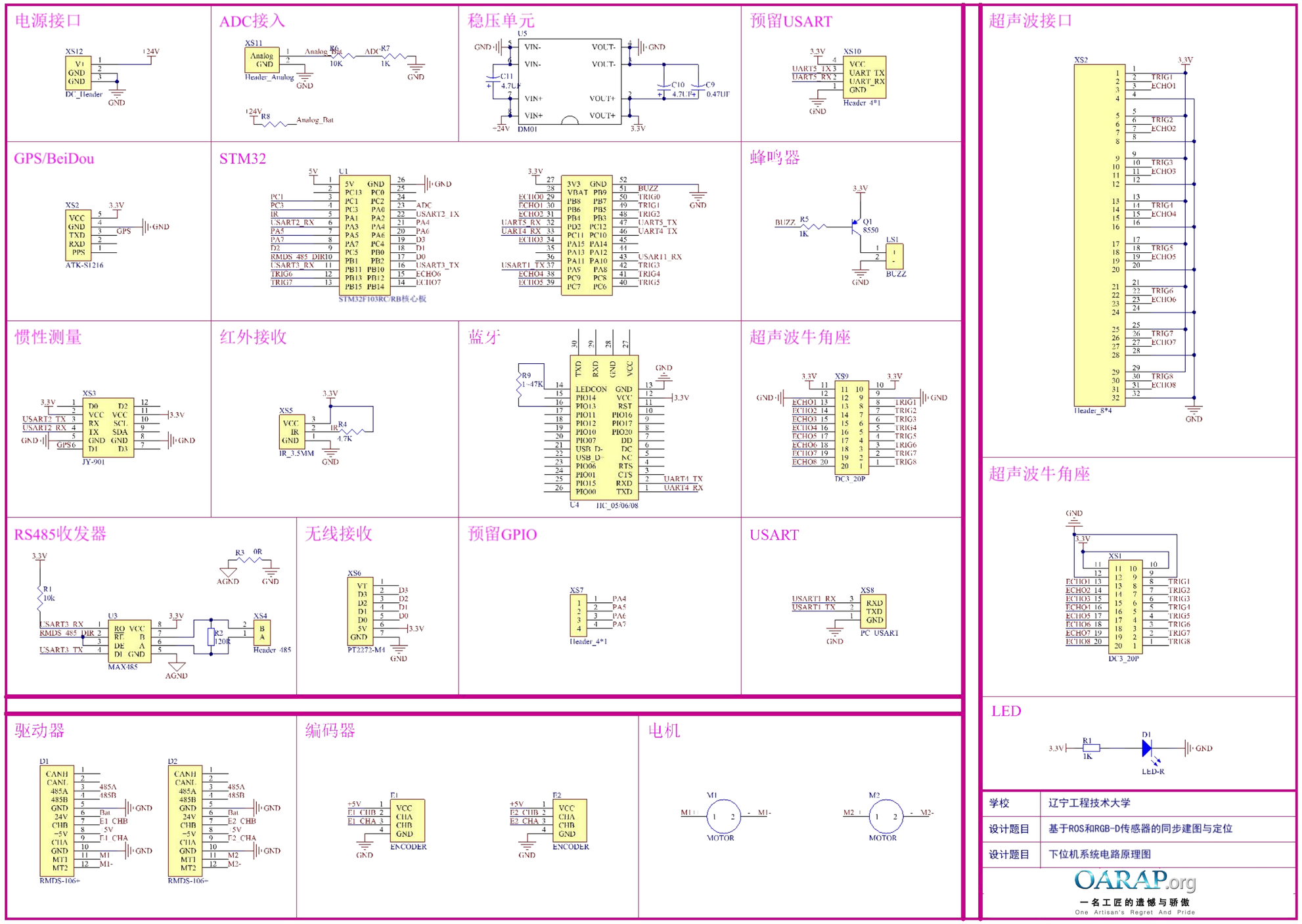
附录 C TF 树


Recorded at time: 1496675460.9
map
Broadcaster: /slam_gmapping
Average rate: 20.834
Buffer length: 1.2
Most recent transform: 1496675460.78
Oldest transform: 1496675459.58
odom
Broadcaster: /robot_pose_ekf
Average rate: 20.564
Buffer length: 0.924
Most recent transform: 1496675460.4
Oldest transform: 1496675459.47
base_link
Broadcaster: /robot_state_publisher
Average rate: 10000.0
Buffer length: 0.0
Most recent transform: 0.0
Oldest transform: 0.0
camera_link
imu_link
right_back_wheel_link
right_front_wheel_link
left_front_wheel_link
screen_frame_link
screen_link
left_back_wheel_link
Broadcaster: /camera_base_link1
Average rate: 10000.0
Buffer length: 0.0
Most recent transform: 0.0
Oldest transform: 0.0
Broadcaster: /camera_base_link
Average rate: 10000.0
Buffer length: 0.0
Most recent transform: 0.0
Oldest transform: 0.0
camera_rgb_frame
camera_depth_frame
Broadcaster: /camera_base_link3
Average rate: 10000.0
Buffer length: 0.0
Most recent transform: 0.0
Oldest transform: 0.0
Broadcaster: /camera_base_link2
Average rate: 10000.0
Buffer length: 0.0
Most recent transform: 0.0
Oldest transform: 0.0
camera_rgb_optical_frame
camera_depth_optical_frame

附录 D SLAM 过程中节点关系



附录 E 导航过程中节点关系



附录 F 下位机电路原理图