



# 电子信息系统综合设计I（论文）

ROS智能小车的运动控制与结点编程

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **观测点** | **A** | **B** | **C** | **D** | **得分** |
| 技术方案对比分析  （权重0.25） | 分析深入，技术方案性能突出。 | 分析和技术方案比较合理 | 分析和技术方案基本可行 | 分析一般，技术方案有欠缺 |  |
| 硬件设计与实现  （权重0.25） | 硬件设计完全符合专业规范，制作精良，稳定可靠。 | 硬件设计基本符合专业规范，制作比较规范，运行比较稳定。 | 硬件设计没有缺陷，制作一般，能正常运行。 | 硬件设计有缺陷，走线比较凌乱，基本能运行。 |  |
| 软件设计与实现  （权重0.25） | 软件功能完善，能实现题目全部要求。 | 软件功能实现完成了题目绝大部分要求。 | 软件功能实现有缺陷，但关键功能都有实现。 | 软件功能实现有缺陷，个别关键功能有缺漏。 |  |
| 系统测试与分析  （权重0.25） | 测试计划完整，测试结果丰富，  分析正确且深入透彻。 | 测试计划有缺漏，测试结果比较好，分析正确合理。 | 测试计划有缺漏，测试结果一般，分析基本正确。 | 测试计划有缺漏，测试结果和分析都有不少错漏。 |  |

**现场验收评分**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 考核点 | 是否能在ROS系统下控制智能车运动？ | 是否能在线调节智能车运动控制的PID参数？ | 是否加入新任务实现电磁传感器数据的读取和传输功能？ | 其他功能 |
| 得分 |  |  |  |  |

**学 院 信息工程学院**

**专业班级 电子信息工程(1)班**

**学 号**

**学生姓名**

**同组队员**

**2024年3月7日**

**摘 要**

**（以下只是字体格式示例，请自己完成具体摘要内容）**

ROS智能小车。。。。。。

**关键词：**智能小车，ROS，PID控制

**Abstract**

**（以下只是字体格式示例，请自己完成具体摘要内容）**

The control algorithm of the smart car…….

**Keywords:** Smart car, ROS, PID control

**目 录**

[电子信息系统综合设计I（论文） 1](#_Toc617108554)

[1 绪论 1](#_Toc1524032020)

[1.1综合设计题目要求 1](#_Toc742301622)

[1.2 论文主要内容和框架 1](#_Toc1467783376)

[2 ROS智能小车硬件组成与运动学建模 2](#_Toc81601306)

[2.1 ROS智能小车的硬件组成 3](#_Toc1457607254)

[2.1.1车架 3](#_Toc454609636)

[2.1.2 STM32运动控制板 3](#_Toc2012350202)

[2.1.3 车载Jetson Xavier平台 4](#_Toc293631017)

[2.1.3 车载多传感器 4](#_Toc524725286)

[2.2 运动学建模 4](#_Toc36402901)

[2.3 本章小结 5](#_Toc283801047)

[（每一章都另起一页，这里是插入了分页符，正式提交时请删掉本行文字） 6](#_Toc481276770)

[3 ROS智能小车主控程序 6](#_Toc378640496)

[3.1 FreeRTOS任务管理系统 6](#_Toc217318915)

[3.1.1 运动控制任务 6](#_Toc1189836129)

[3.1.2 人机交互任务 6](#_Toc1910076694)

[3.1.3 数据收发任务 6](#_Toc729584513)

[3.1.4 电磁循迹任务 6](#_Toc1610330012)

[3.2 运动控制 7](#_Toc1699708154)

[3.2.1 PWM控制原理 7](#_Toc2070113759)

[3.2.2 STM32 PWM外设配置 7](#_Toc739586044)

[3.2.3 舵机转向PID控制 7](#_Toc1539701686)

[3.2.2电机速度PID控制 7](#_Toc101940540)

[3.3 本章小结 8](#_Toc791365175)

[4 智能小车ROS节点编程 9](#_Toc1229768617)

[4.1 ROS系统特点 9](#_Toc2111077920)

[4.2 ROS系统节点编程 11](#_Toc1485156625)

[4.2.1 创建工作空间 11](#_Toc1122608997)

[4.2.2 创建包Package 13](#_Toc1497944784)

[4.2.3 创建结点Node 15](#_Toc1523235127)

[4.2.3 编译节点代码 17](#_Toc1739717551)

[4.3 CLion编写ROS结点程序 17](#_Toc874493156)

[4.3.1 CMake编译 17](#_Toc118053101)

[4.3.2 CLion部署功能 18](#_Toc1060017280)

[4.4 ROS系统订阅和发布电磁循迹数据 20](#_Toc956094462)

[4.4.1 修改串口数据解包代码 21](#_Toc1575660356)

[4.4.2 话题创建与消息发布 22](#_Toc1514626916)

[4.5 Ubuntu系统设置 23](#_Toc820961016)

[4.5.1 修改热点名称 23](#_Toc1869291373)

[4.5.2 设置风扇自启服务 23](#_Toc2039352203)

[4.6 本章小结 25](#_Toc857363917)

[5 系统测试与总结展望 26](#_Toc5608772)

[5.1 系统测试结果 26](#_Toc373145325)

[5.2 经验总结 26](#_Toc1236004413)

[5.3 展望 26](#_Toc222927687)

[参 考 文 献 27](#_Toc1562981454)

[附录A 智能小车程序 28](#_Toc998597460)

[附录B 电路原理图 28](#_Toc952512201)

[附录C 上位机程序 29](#_Toc1025827819)

（！！！！请不要直接修改本目录，正确做法应该是先修改正文标题，然后再点击本目录，左上角会弹出一个菜单“！更新目录”，点击它就会更新整个目录）

# 1 绪论

## 1.1综合设计题目要求

小四，宋体，1.5倍行距。

## 1.2 论文主要内容和框架

小四，宋体，1.5倍行距。

（每一章都另起一页，这里是插入了分页符）

# 2 ROS智能小车硬件组成与运动学建模

硬件组成。

**（下面两个图例仅供参考格式，包括编号，采用二级编号）**

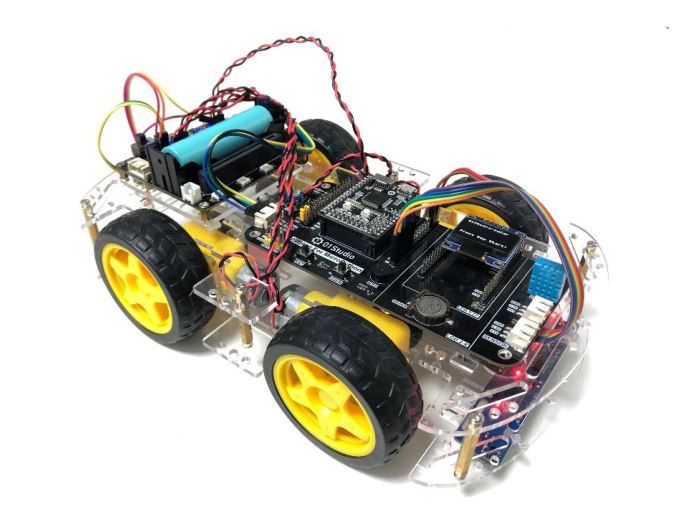


图2.1 智能小车实物图

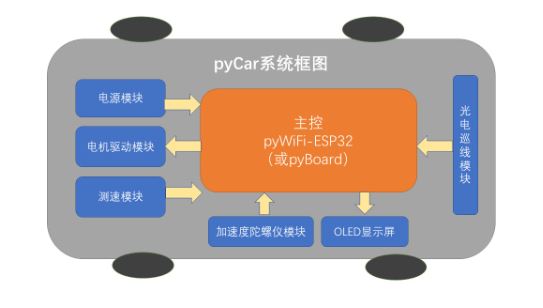


图2.2 智能小车系统框图

## 2.1 ROS智能小车的硬件组成

### 2.1.1车架

麦轮、三轮、阿克曼，带编码器电机，电机参数，舵机参数，电池参数等等。

小四，宋体，1.5倍行距。

### 2.1.2 STM32运动控制板

电源处理、蓝牙、电机驱动芯片、OLED、一键下载、USB转串口等等，请自行增加更多的运动控制板电路分析。

小四，宋体，1.5倍行距。

**！！！下面表格内容没有任何意义，仅供参考格式（提交时请删掉本行），不用表格的话直接删掉即可。**

**表4.1 智能小车元件清单**

|  |  |
| --- | --- |
| 模块名 | 型号 |
| 五路光电巡线模块 | TCRT5000l |
| 测速模块 | 红外对管 |
| 电子罗盘模块 | HMC5883L |
| WIFI模块 | pyWiFi-ESP32（主控板） |
| 三轴加速度+三轴  陀螺仪一体模块 | MPU6050 |
| 电机 | 3V-9V普通直流减速电机 |

（三线表，三条横线可见，其他隐藏，表标题居中，字体、字号请见上示例，提交时请删掉本备注）

### 2.1.3 车载Jetson Xavier平台

英伟达Jetson Xavier，小四，宋体，1.5倍行距。

### 2.1.3 车载多传感器

IMU芯片MPU6050， 高清摄像头，激光雷达，麦克风阵列等等，

小四，宋体，1.5倍行距。

## 2.2 运动学建模

请参考STM32运动底盘开发手册，第4节，机器人运动学分析

小四，宋体，1.5倍行距。

（3-1）

其中，k为采样序号；u(k)为第k个采样时刻的输出值；e(k)为第k个采样时刻的偏差值；Kp为比例系数；Ki=Kp∙T/Ti为积分系数；Kd=Kp∙Td/T为微分系数。

（公式编号、公式样式、公式参数说明，请见上述示范）

## 2.3 本章小结

# （每一章都另起一页，这里是插入了分页符，正式提交时请删掉本行文字）

# 3 ROS智能小车主控程序

## 3.1 FreeRTOS任务管理系统

主控程序总框图

小车STM32主控划分了多少个任务，每个任务分别实现什么功能

**（下面子标题仅供参考，请自行根据实际代码修改。）**

### 3.1.1 运动控制任务

小四，宋体，1.5倍行距。

### 3.1.2 人机交互任务

蓝牙、OLED屏等等

小四，宋体，1.5倍行距。

### 3.1.3 数据收发任务

与ROS系统数据交换IMU数据、里程计数据，。。。。。等等

小四，宋体，1.5倍行距。

### 3.1.4 电磁循迹任务

有做的请自行添加

小四，宋体，1.5倍行距。

## 3.2 运动控制

### 3.2.1 PWM控制原理

讲解PWM控制电机和舵机的基本原理。

小四，宋体，1.5倍行距。

### 3.2.2 STM32 PWM外设配置

如何计算和配置STM32的PWM脉宽、占空比、极性等，给出具体计算过程，以及配置代码，和代码解析。

小四，宋体，1.5倍行距[2]。

（参考文献引用编号在右上角，请见上述示范，大家还可以用软件“Note Express广东工业大学图书馆版”插入参考文献，这样后面的参考文献不用自己排版，更加规范）

### 3.2.3 舵机转向PID控制

（3-1）

其中，k为采样序号；u(k)为第k个采样时刻的输出值；e(k)为第k个采样时刻的偏差值；Kp为比例系数；Ki=Kp∙T/Ti为积分系数；Kd=Kp∙Td/T为微分系数。

（公式编号、公式样式、公式参数说明，请见上述示范）

### 3.2.2电机速度PID控制

小四，宋体，1.5倍行距。

## 3.3 本章小结

# 4 智能小车ROS节点编程

## **4.1 ROS系统特点**

在本次ROS小车课设中，套件使用Jetson nano作为上位机，在上面部署了Ubuntu18.04系统，并配置Melodic版本的ROS环境，实现诸如小车运动控制指令下发、巡线、跟随、建图、导航等功能。

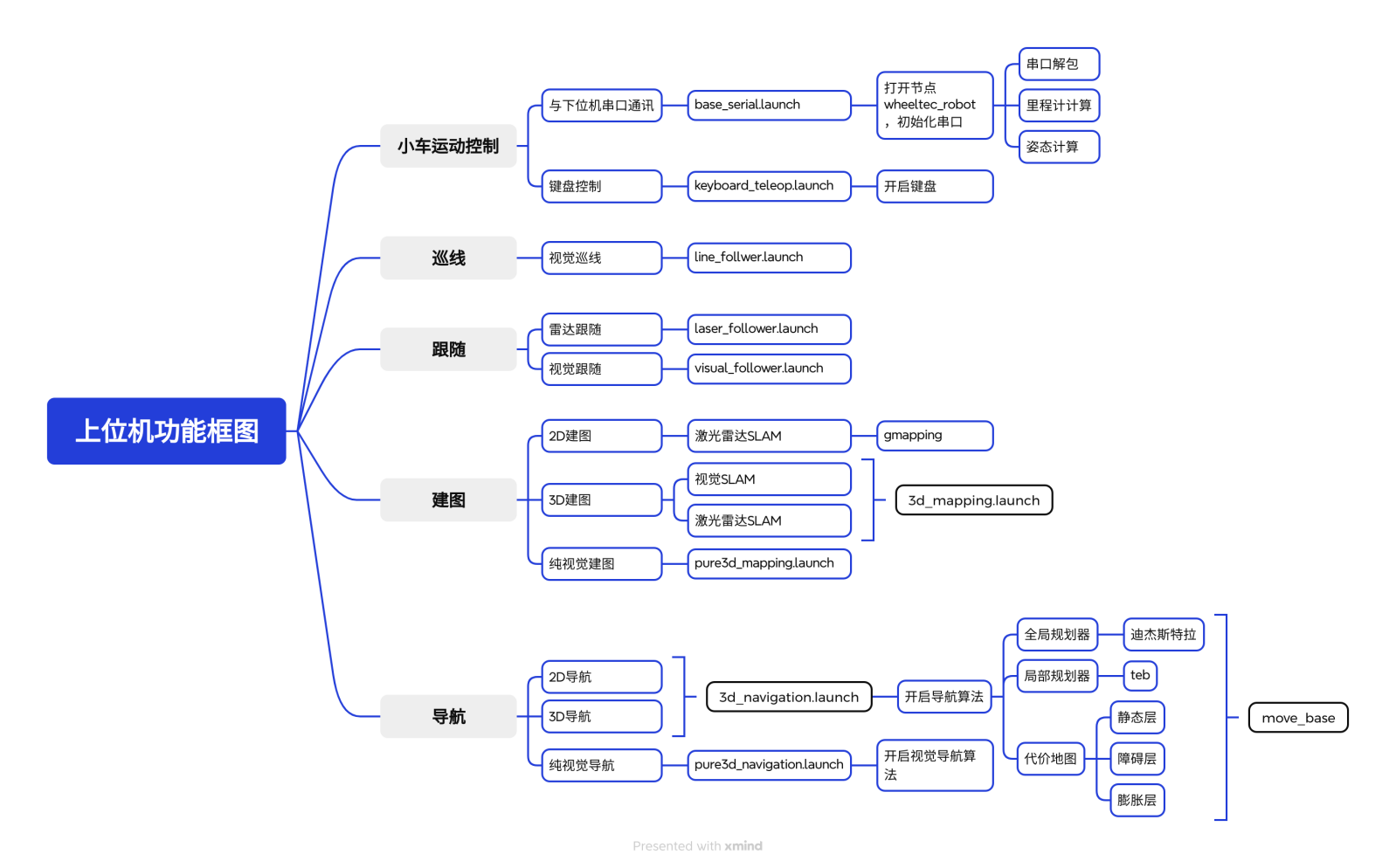
上位机功能框图如下：

图4.1 上位机功能框图

本次课设使用的ROS （Robot Operating System），是一个适用于机器人的开源的元操作系统，采用了模块化设计，将机器人软件分解成独立的功能单元，称为节点。这种架构使得软件开发更加灵活，便于代码的重用和团队协作。 ROS节点之间通过话题发布/订阅和服务调用等方式进行通信，实现松散耦合的交互。这种通信机制使得节点之间的数据传输更加灵活，能够适应不同的机器人架构。

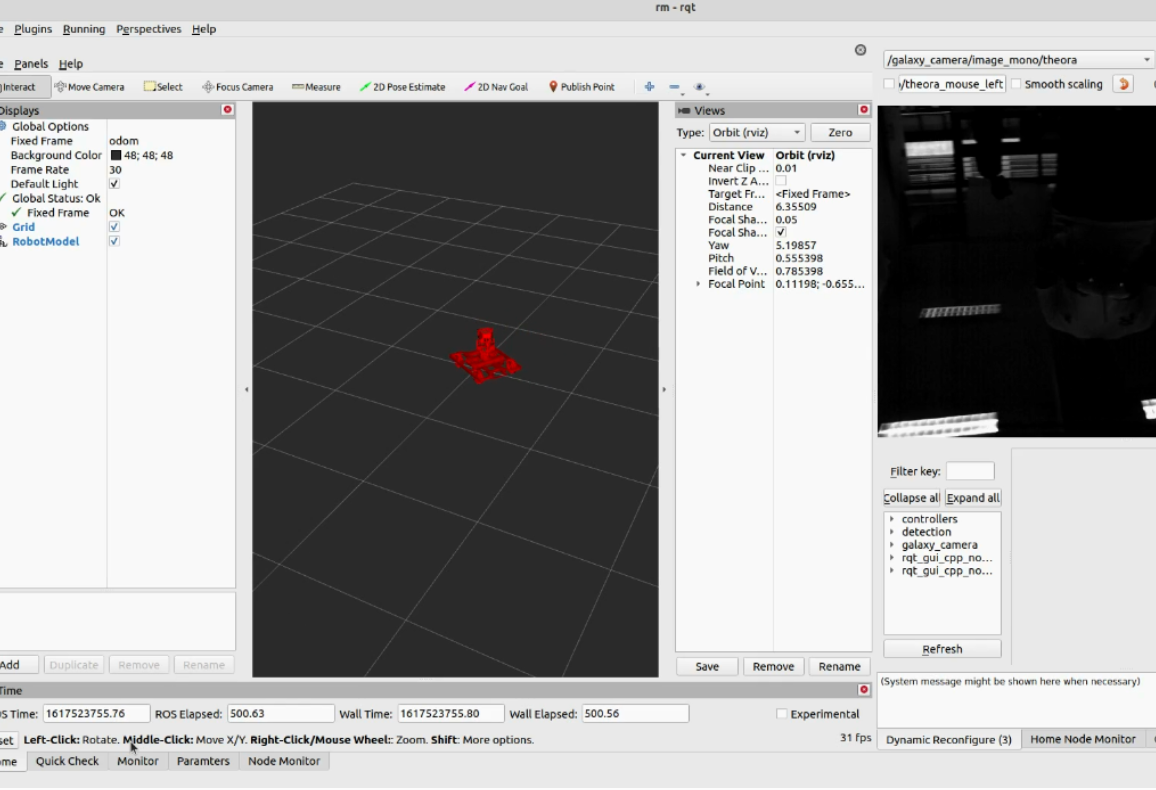
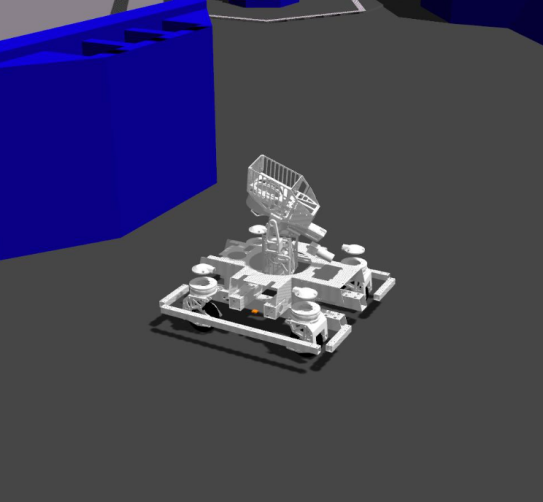
除此之外，ROS还提供了丰富的工具来简化开发、调试和可视化机器人软件。例如，RViz用于三维可视化、RQT提供了图形化界面工具、Gazebo用于仿真等等。这些工具为机器人调试和模型建立提供了极大便利。  


图4.2 Gazebo（左） RViz（右）

有一个很巧妙的比喻是，ROS之于机器人，就好比应用商店之于安卓系统，这意味着ROS提供了许多功能库，涵盖了机器人开发中常用的功能，包括运动控制、感知、SLAM、导航等，可以使开发者们避免“重复造轮子”的过程。本次课设使用的套件的大部分代码便是源于使用ROS统一了规范的开源包，如navigation，move\_base，gmapping等。

综上所述，ROS在机器人开发领域有着不可动摇的地位，同时也是机器人开发绝好的入门级系统，有着极高的学习价值，接下来我将展开并详细论述。

## 4.2 ROS系统节点编程

以wheeltec\_robot小车初始化节点为例，这个节点被定义在turn\_on\_wheeltec\_robot包中，承担了串口通信数据解包、IMU数据转换、里程计计算、将解包计算的数据发布到话题等功能。

节点程序架构图如下：

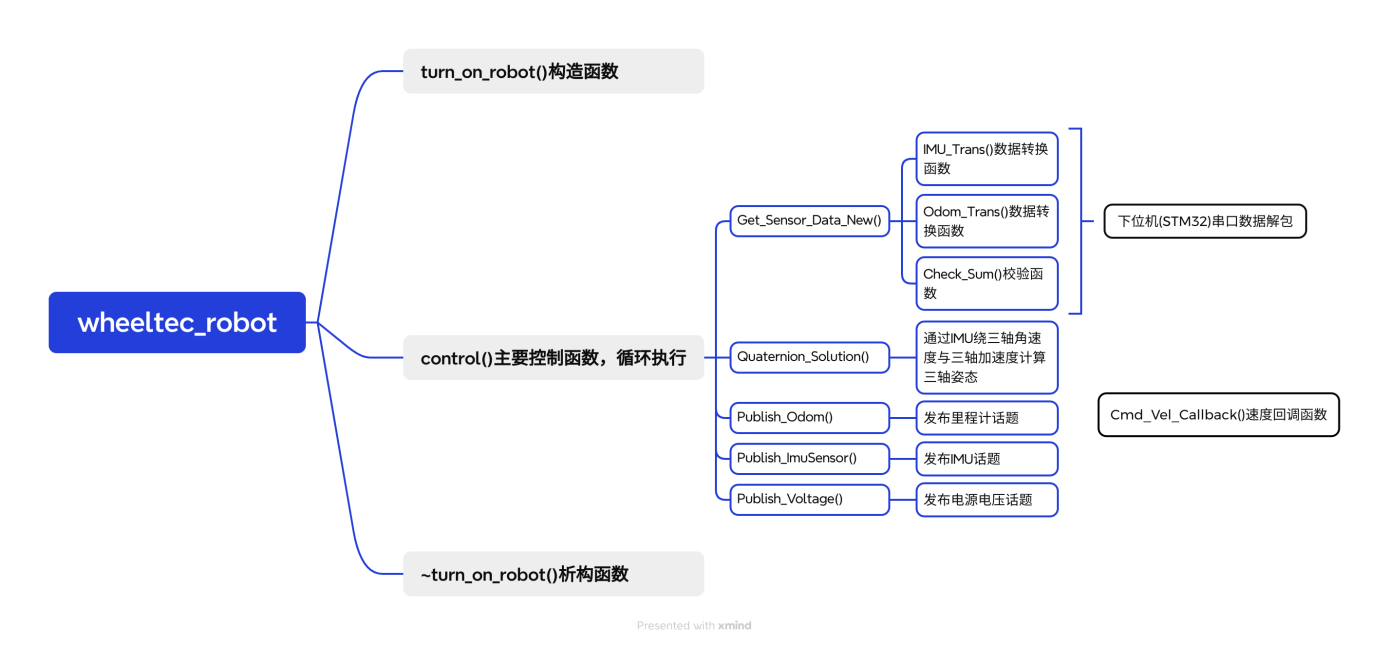


图4.3 wheeltec\_robot节点程序架构图

接下来我将展示如何创建这样一个节点。

### 4.2.1 创建工作空间

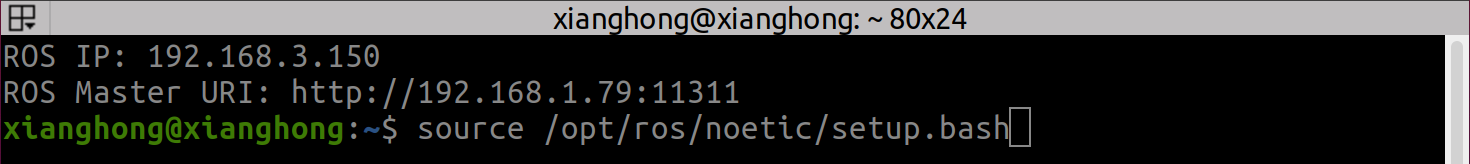
ROS工作空间一般直接创建在/home目录下。首先要先source配置的ROS环境，这里以noetic版本为例（课设使用的是melodic版本，将noetic改为melodic即可），这样才能正常使用catkin对工作空间进行编译。

图4.4 source配置的ROS环境

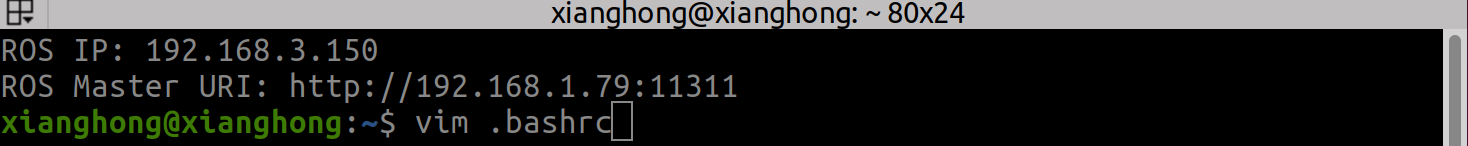
一般习惯在配置完ROS环境后就将这段指令写进.bashrc中，这样每次打开终端都会自动加载ROS环境，不用重复输入。

图4.5 使用vim打开.bashrc

将该指令复制粘贴进去即可。

执行完上述操作后，就可以创建新的工作空间目录了。执行以下指令：

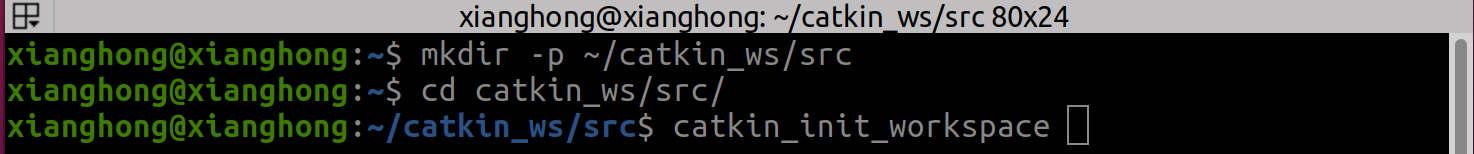


图4.6 创建新的工作空间目录

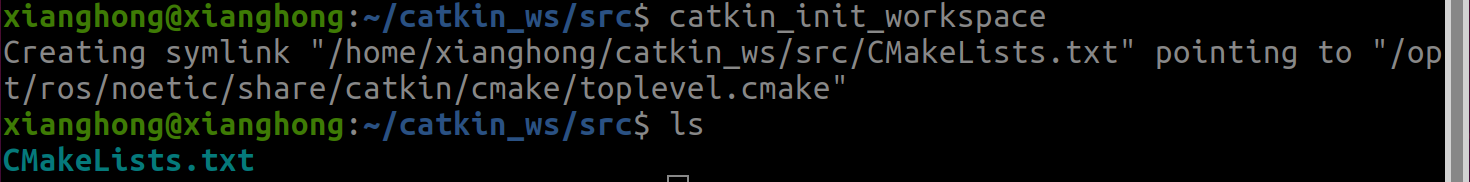
执行完上述指令后，你会发现在当前目录下（也就是src目录），生成了一个CMakeLists.txt文件。无论你使用哪种IDE（例如CLion和VSCode），都可以在“Open File or Project”页面下双击src目录下的CMakeLists.txt文件来 打开工作空间。在之后的开发中，直接将包添加在src目录下即可。  


图4.7 生成CMakeList.txt

### 4.2.2 创建包Package

执行以下指令，创建名为turn\_on\_wheeltec\_robot的Package。

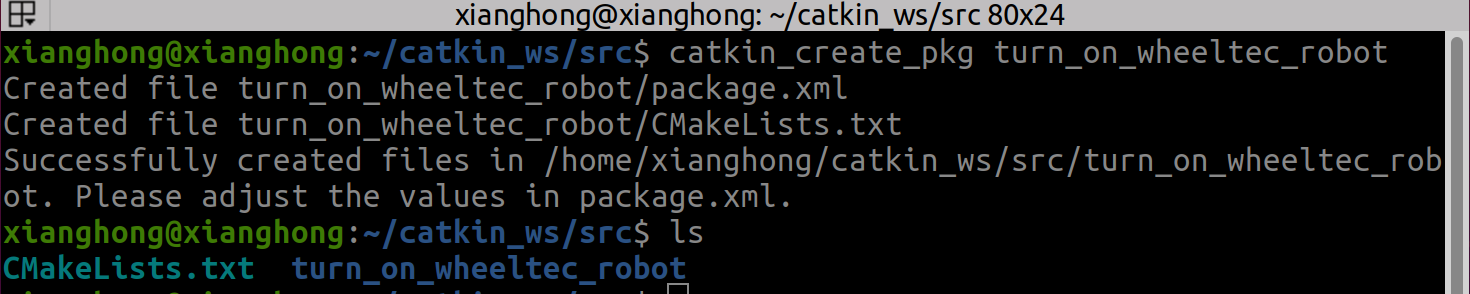


图4.8 创建Package

可以看到，新创建的Package中生成了package.xml和CMakeList.txt两个文件。package.xml是ROS软件包的元数据文件，其中包含了关于软件包的各种信息和配置，包括软件包的名称、版本、作者、许可证信息、依赖项、软件包发布的消息类型以及订阅的消息类型、编译选项、安装路径等。CMakeLists.txt则是用CMake语言编写的构建系统描述文件，用于配置和管理ROS软件包的构建过程，包括编译器标志、包含目录、链接库、软件包的编译目标、解析软件包的依赖项、软件包构建完成后安装的规则等。

创建好Package后，可以试着编译该工作空间下的所有包，注意要在工作空间目录下（即catkin\_ws）。推荐使用catkin build指令。若想指定单个包进行编译，在catkin build后加上包名即可。

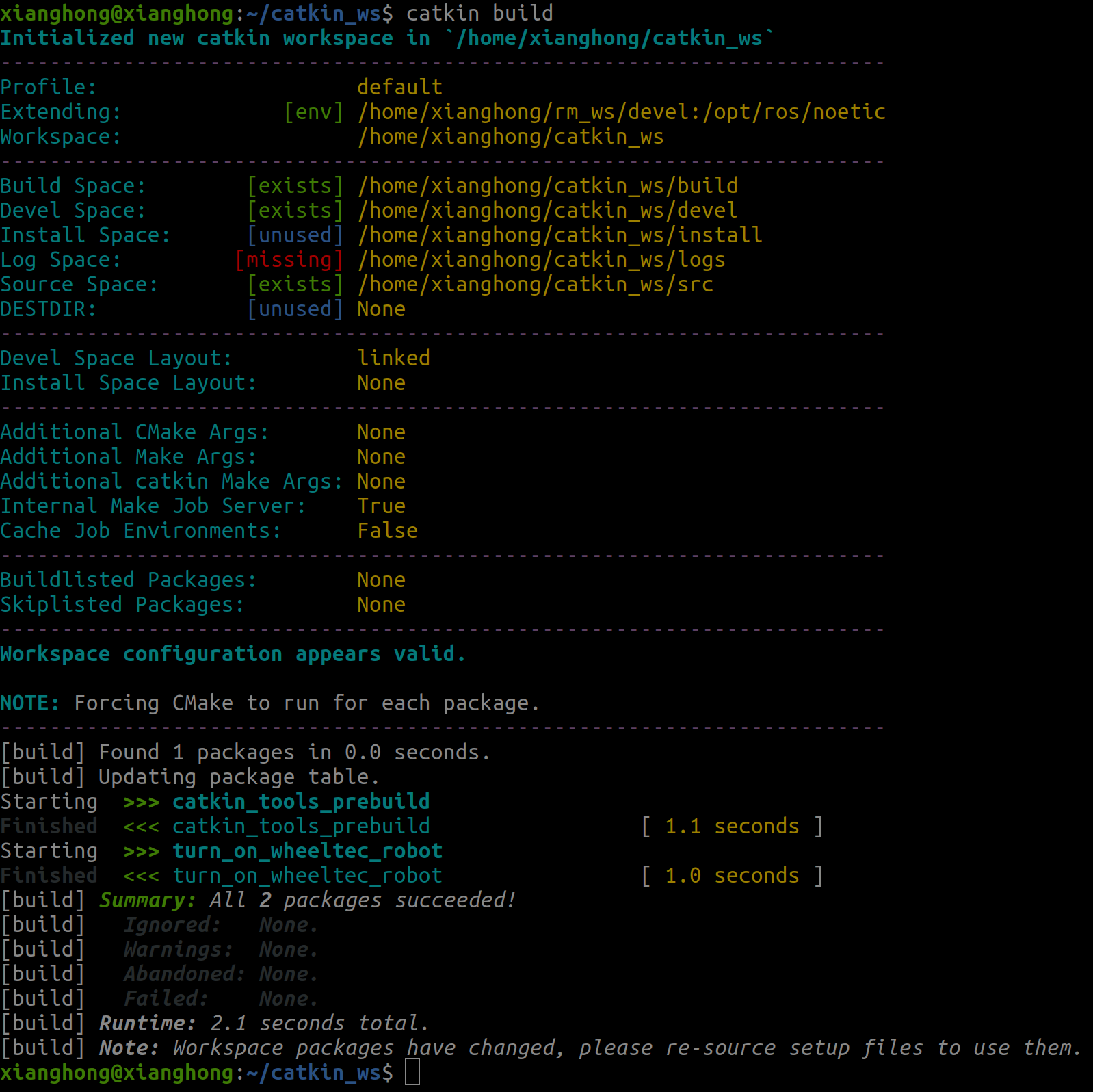


图4.9 编译工作空间下的所有包

编译成功后，会在工作空间下生成build、devel、logs文件夹。build文件夹用于存储编译过程中生成的中间文件和临时文件，包括编译产生的目标文件、库文件、可执行文件等。devel文件夹是一个软链接（symbolic link）目录，指向编译生成的ROS软件包的可执行文件、库文件等。logs文件夹用于存储构建过程中的日志文件，包括编译日志、错误日志等。这些日志文件记录了构建过程中的详细信息，有助于开发者诊断和解决构建中的问题。

接下来就可以在创建的包中进行一系列开发了。

### 4.2.3 创建结点Node

在创建的包中添加src、include目录：

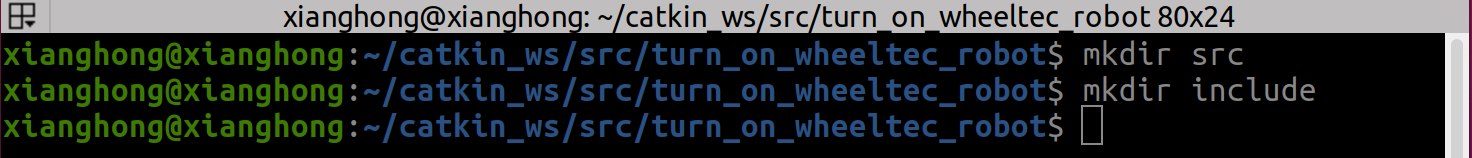
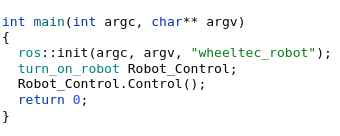


图4.10 添加目录

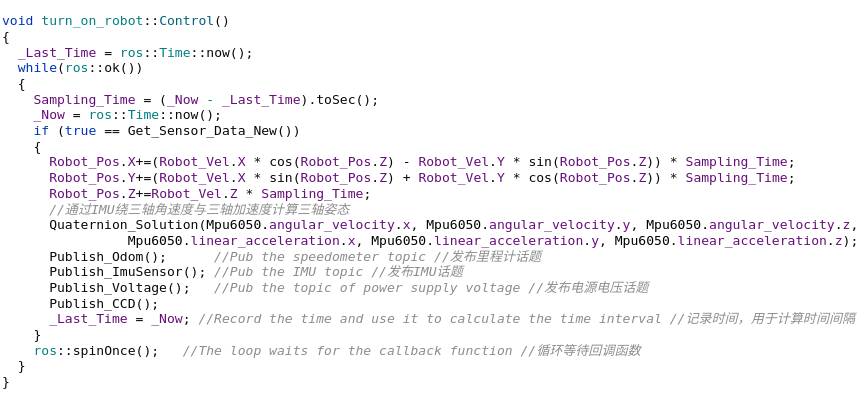
在CLion中打开工作空间，向src中添加cpp文件，向include中添加头文件。以wheeltec\_robot.h头文件为例，需要在头文件中引用ros头文件、消息类型头文件等，并按需求写好类，其中包括构造函数、析构函数、对象声明等。值得注意的是，每个头文件都需要在开头的地方加#pragma once，防止重复编译（课设代码使用的是#ifndef，不如#pragma once简便）。



在cpp文件中引用定义好的同名头文件。以wheeltec\_robot.cpp为例，每个节点都有一个main函数，在里面进行ROS初始化、对象实例化、执行主循环等。



其中，节点名称就是由ros\_init函数中输入的名称定义的。



ROS节点的主循环结构非常简单，while(ros::ok())是一个while循环，它的条件是ros::ok()。ros::ok()函数返回一个布尔值，表示ROS节点是否仍处于运行状态。只要ROS节点处于运行状态，该循环将继续执行。在循环的每次迭代中，调用ros::spinOnce()函数。这个函数的作用是处理所有的回调函数。在ROS中，消息和服务的处理是通过回调函数实现的，这些回调函数会在接收到相应的消息或服务请求时被调用。ros::spinOnce()函数的调用让节点有机会处理这些回调函数，以确保及时响应接收到的消息或服务请求。整个结构的作用是让节点保持运行状态并等待事件的发生，同时确保及时处理接收到的消息或服务请求。

### 4.2.3 编译节点代码

如果正确按上述说明创建工作空间、创建包并设置节点，就可以执行编译了。来到工作空间目录下，执行catkin build turn\_on\_wheeltec\_robot指令。若是节点代码编写有误，或是缺少依赖，编译就无法正常执行，终端会出现相应错误提示。除了终端的错误报告，IDE中同样会用红线将错误代码标出，方便排查问题。

## 4.3 CLion编写ROS结点程序

CLion专门针对C和C++开发，因此非常适合作为开发ROS的IDE进行使用。ROS使用CMake来管理和构建软件包，而CLion具有内置的CMake支持。因此可以在CLion中打开ROS软件包所在的工作空间，并使用CMakeLists.txt文件来配置和构建ROS软件包。

在CLion中选择Open Project，双击工程的CMakeList即可打开工程。

### 4.3.1 CMake编译

CLion自带CMake工程重载功能。一般在写完代码后，要先通过CLion内部的CMake检查，再到终端进行编译。



图4.11 在CLion上检查CMake

如图所示，现在CLion页面下方选择CMake选项卡，再在左侧点击Reload CMake Project。如果CMake检查通过，会在最后显示[Finished]字样。

### 4.3.2 CLion部署功能

CLion自带的部署功能，可以方便地将代码部署到ROS小车上，只需要通过SSH协议连接。具体配置如下：

在CLion上方的Tools选项卡中选择Deployment，再选择Configuration。

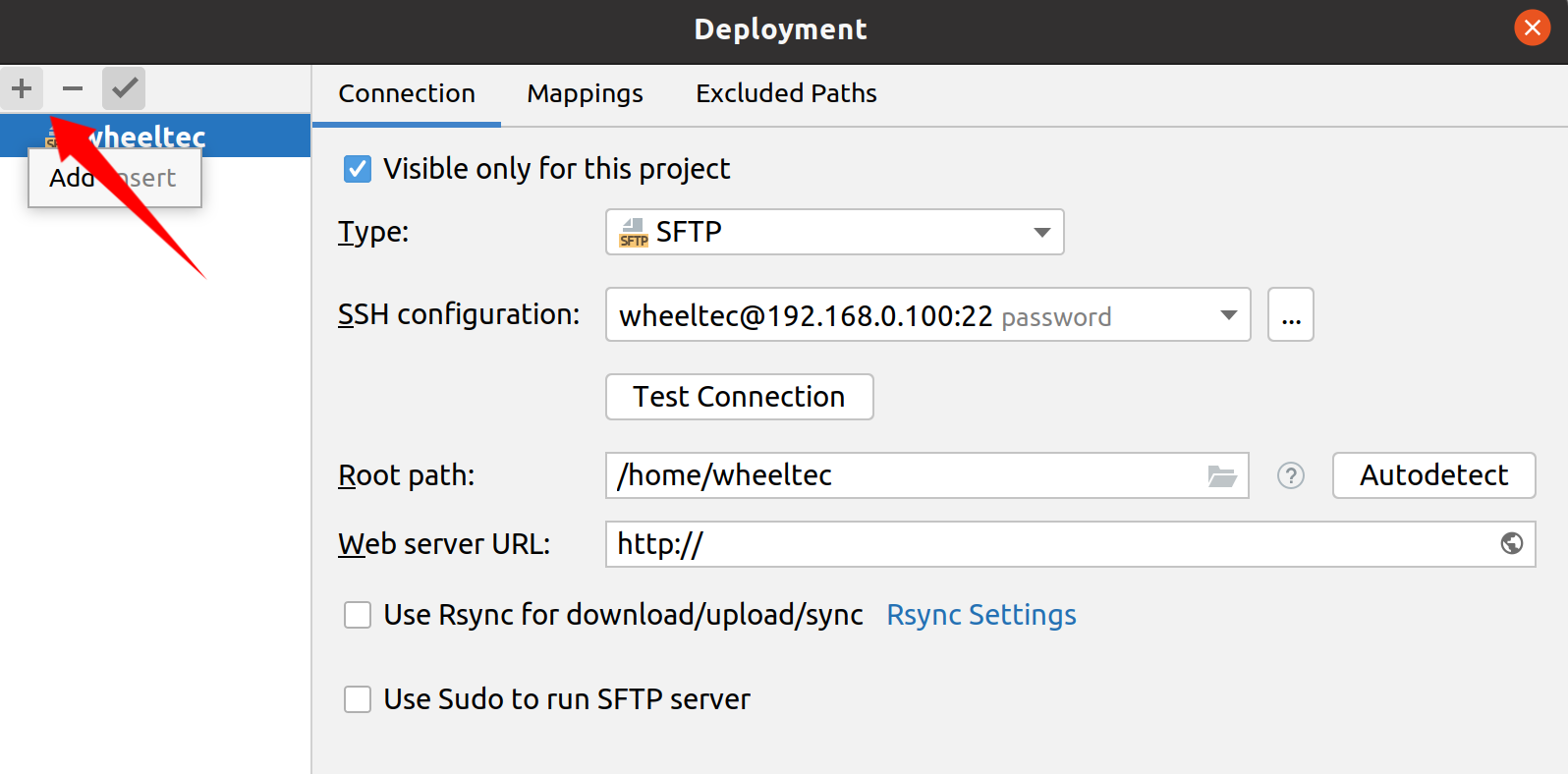


图4.12 部署设置卡

点击左上方+号，选择SFTP。将服务名设置为wheeltec（可以自定义）。

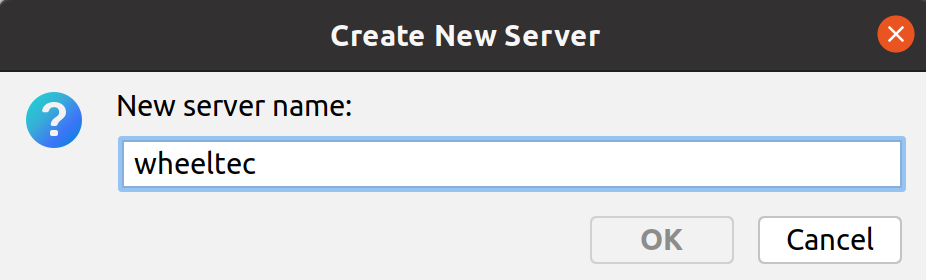


图4.13 设置服务名

点击SSH Configuration一行左侧的省略号，将要连接的WiFi共享对象的ip地址、用户名、密码填入，端口号默认22。

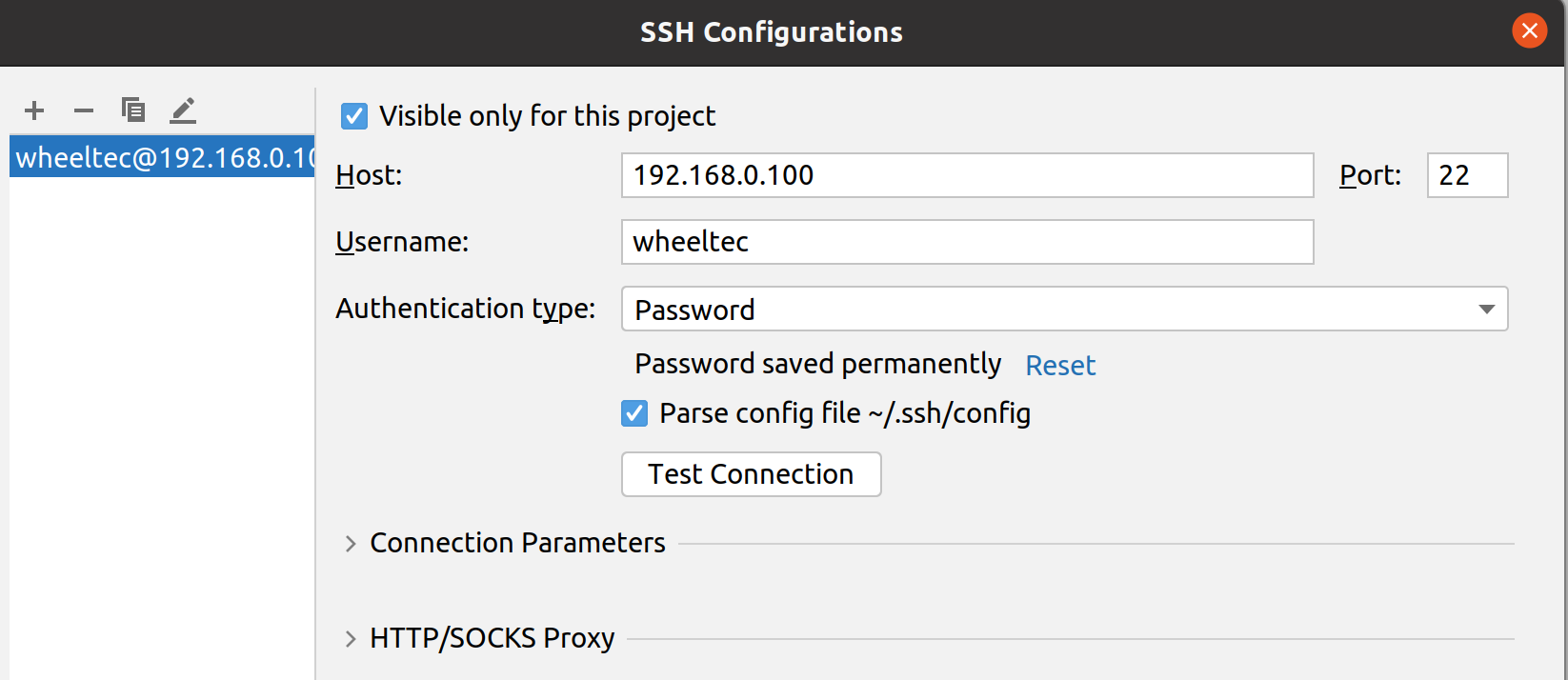


图4.14 SSH Configuration

确保本机和小车的WiFi连接正常，点击Test Connection。若配置无误，就会显示连接成功。

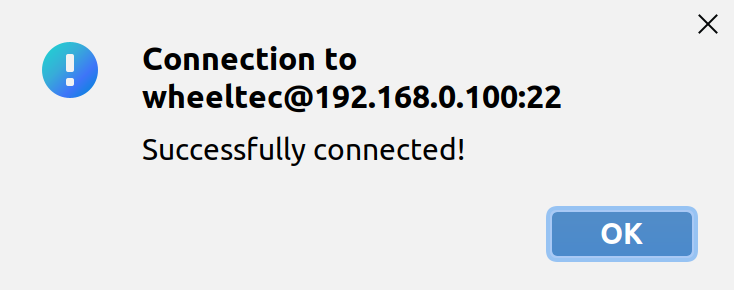


图4.15 SSH连接成功

SSH连接成功后，就可以配置文件夹映射关系了。

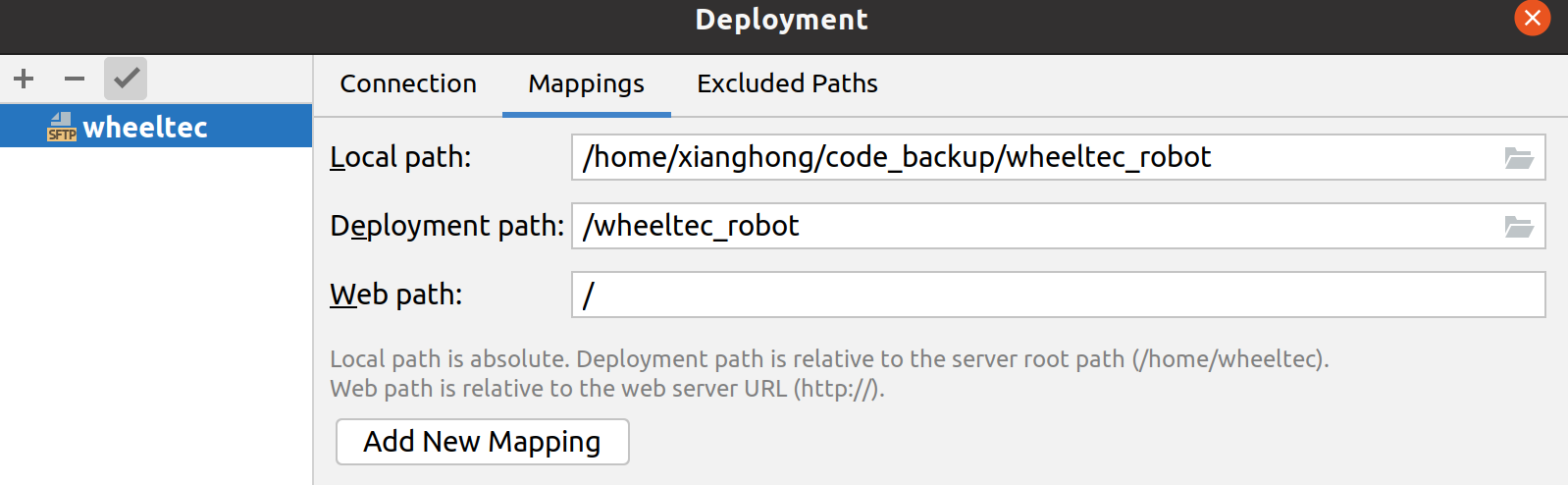


图4.16 配置文件夹映射关系

在Deployment页面选择Mappings。将本机代码所在文件夹与远程连接的系统中要替换的代码文件夹映射起来。

配置完成后，在代码页面右键，选择Deployment，点击Upload to wheeltec，即可将本机代码部署到小车上。

## 4.4 ROS系统订阅和发布电磁循迹数据

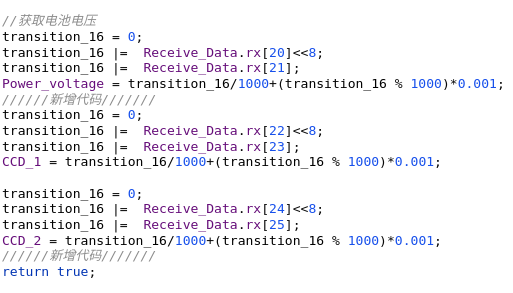
本拓展项的要点主要分成两部分：修改串口数据解包代码和创建话题、发布消息。经过上文的解析，会发现串口数据解包代码及话题发布代码都在wheeltec\_robot.cpp和wheeltec\_robot.h中，因此本拓展部分主要修改上述代码。

值得注意的是，该ROS小车套件的代码编写没有遵循ROS C++ Style，例如函数名首字母小写，其余单词首字母大写、全局变量后加下划线等等。良好的开发规范有助于代码的简明直观。

### 4.4.1 修改串口数据解包代码

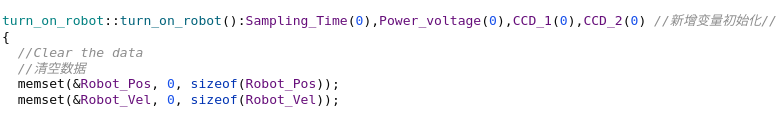
wheeltec\_robot.cpp

411-427行//解包CCD电压数据



一路CCD电压占用两个字节，此处仿照电池电压传输方式编写即可。

470行//类成员初始化



wheeltec\_robot.h

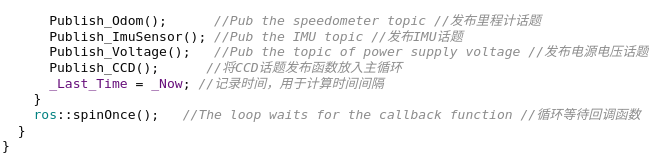
168-169行//定义全局变量类型



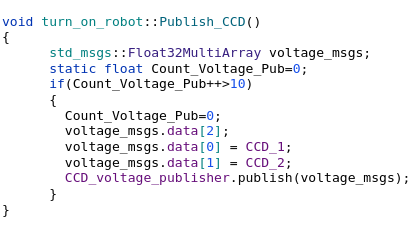
### 4.4.2 话题创建与消息发布

wheeltec\_robot.cpp

459行//将CCD消息发布函数放入主循环



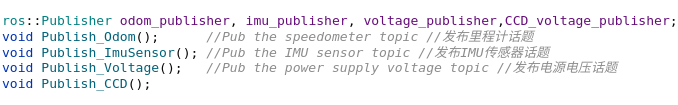
171-183行//话题发布函数内容



因为存储两路CCD电压的格式使用了double数组的消息类型，因此还要加上std\_msgs/Float32MultiArray.h头文件进行引用。

wheeltec\_robot.h

148-152行//定义发布者和话题发布函数



## 4.5 Ubuntu系统设置

### 4.5.1 修改热点名称

由于本课设所有小车用的是同一个系统镜像文件，因此WiFi配置也是一样的，由小车Jetson nano共享热点，当小车与使用者电脑同时连接到这个热点时，就能通过SSH协议达成数据传输。然而，所有小车的热点名称都是一致的，若同一课室内有多台小车，就会造成热点连接混乱。因此要修改热点名。

进入/etc/NetworkManager/system-connections目录。

输入sudo vim 'Wi-Fi 连接 1',输入用户密码dongguan，进入编辑界面。

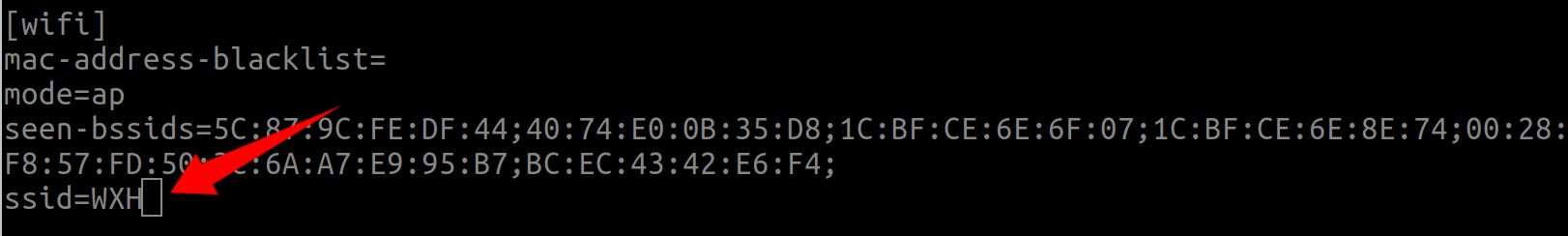


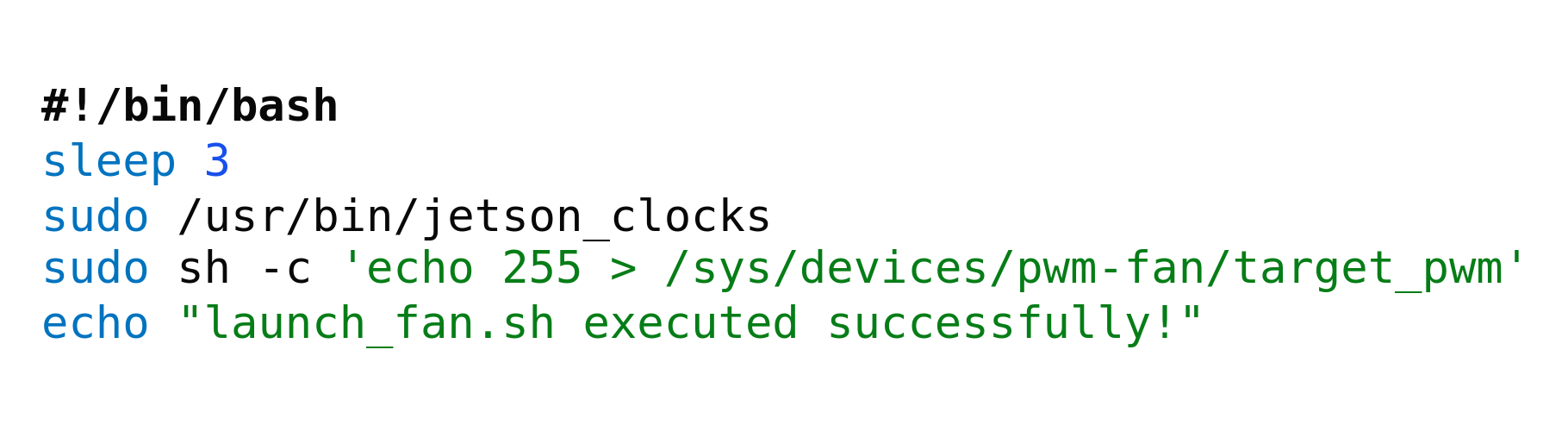
图4.17 修改热点名称

修改[wifi]一栏的ssid内容，即可修改热点名称，防止与其他小车冲突。

### 4.5.2 设置风扇自启服务

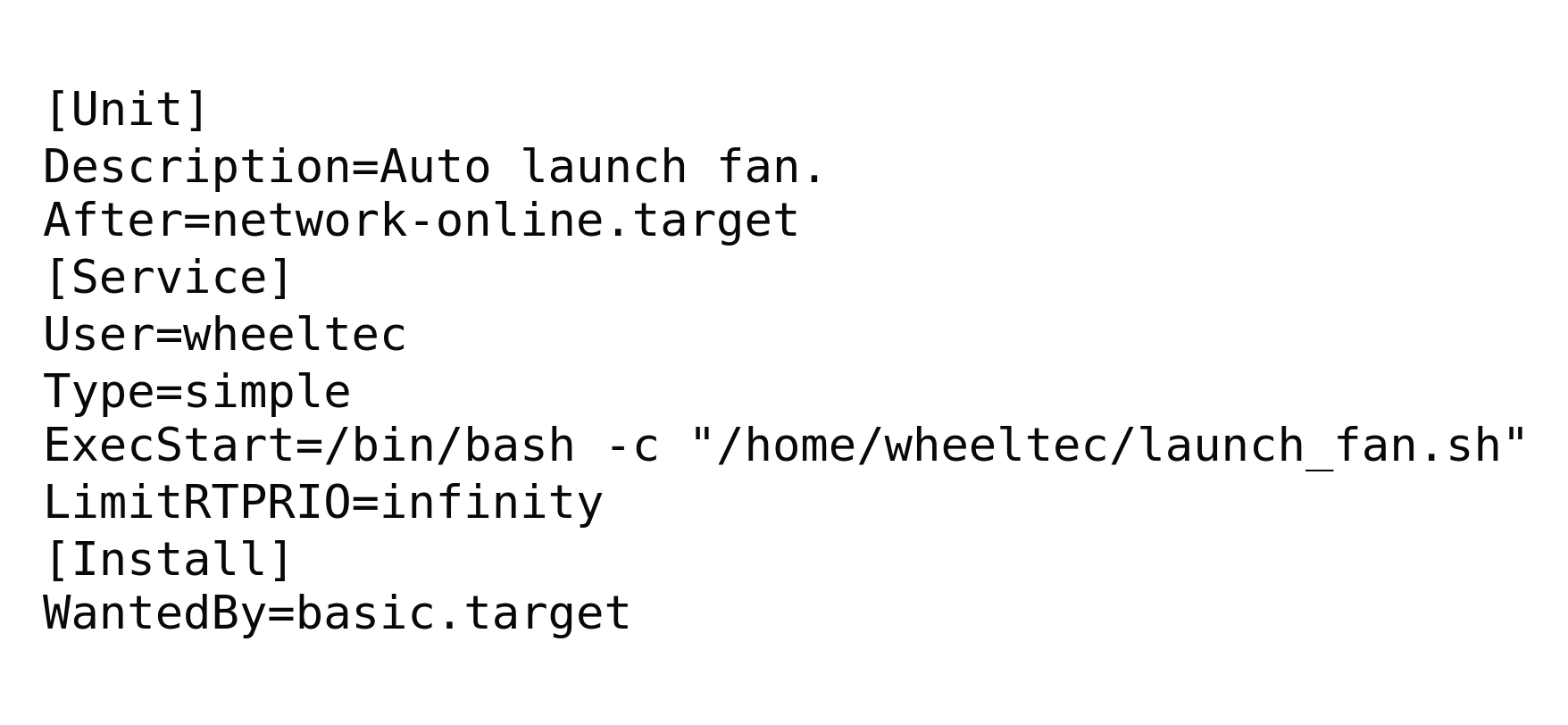
Jetson nano工作时会产生大量热量，需要风扇散热才能保证正常工作。为了使用方便，可以配置风扇自启服务。

在/home目录下创建launch\_fan.sh脚本。



使用sudo chmod 777 launch\_fan.sh指令为脚本赋予权限。

跳转到/lib/systemd/system/目录下，创建launch\_fan.service服务



使用sudo systemctl enable launch\_fan.service使能风扇自启服务。若配置正确，重启后即可生效。

还可以使用sudo systemctl status launch\_fan.service指令查看该服务的运行状态。

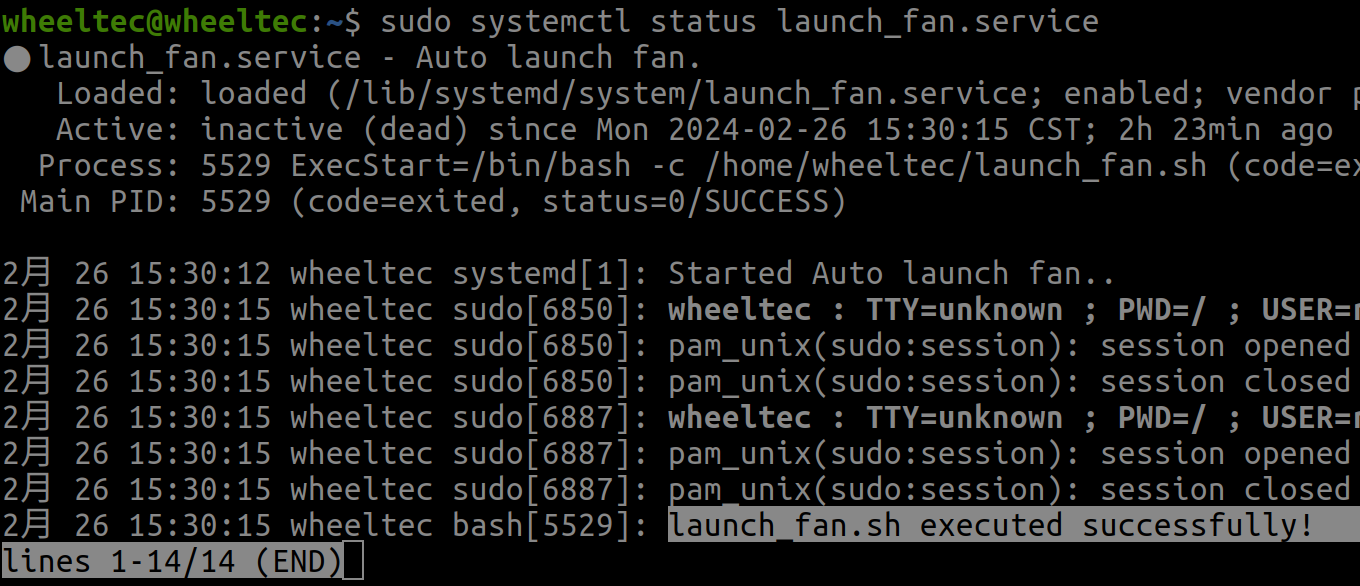


图4.18 查看服务状态

如图，风扇自启脚本返回launch\_fan.sh executed successfully!代表自启服务执行脚本成功。

## 4.6 本章小结

本章内容主要是关于ROS小车的上位机控制，涉及到ROS的节点编程、C++编程、话题定义与消息发布、串口数据解包、Linux命令行、Linux系统目录结构、CLion的ROS开发配置等。ROS作为机器人控制中比较常用的平台，还有许多功能与思想值得学习借鉴。

# 5 系统测试与总结展望

## 5.1 系统测试结果

你实现了哪些内容、进行了哪些测试，测试效果如何。最好图文并茂，数据详实。

## 5.2 经验总结

列举在综合设计开展过程中遇到的哪些困难，采用了怎样的解决方法。

（不建议写空洞的主观感受、套话，论文应该是客观陈述、书面语，不要主观感想、口语化，比如不要写“这个综合设计让我感触良多”之类的）

## 5.3 展望

你认为本综合设计的完成内容有什么缺漏的地方，可以从哪些方面来进一步完善。

对综合设计这一门课的设计内容，你觉得有哪些值得改进的地方，有哪些你觉得很有意义、有价值的内容可以补充进来。

# 参 考 文 献

**（以下所有参考文献只是格式示范用，请自己更新和添加）**

1. 李楠. PID控制参数现代设计技术的研究与应用[D]. 浙江工业大学, 2009.
2. Candes, E.J, Wakin, M.B. An Introduction To Compressive Sampling[J]. IEEE Signal Processing Magazine, 25(2):21-30.
3. 张燕, 陈华. HMC5883L电子罗盘的误差补偿系统设计[J]. 单片机与嵌入式系统应用, 2015(05):45-48.

# 附录A 智能小车程序

init\_speed =

# 附录B 电路原理图

# 附录C 上位机程序

filename = 'cc19.txt'; %解压后的数据

[left2,left1,center