**【项目名称】**

**软件设计说明书**

**SDD107**

**V1.2**

分工说明

|  |  |  |
| --- | --- | --- |
| 小组名称 | 冲鸭robot | |
| 学号 | 姓名 | 本文档中主要承担的工作内容 |
| 17373126 | 刘萱 | 章节1、4、5、6.1、7、8的编写及文档的汇总 |
| 17373138 | 杨嘉成 | 章节4编写 |
| 17373462 | 李宗淦 | 章节2、6.2编写 |
| 17373084 | 兰岸 | 章节3、6.3编写 |

版本变更历史

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 版本 | 提交日期 | 主要编制人 | 审核人 | 版本说明 |
| 1.0 | 2020.4.20 | 刘萱 | 杨嘉成、李宗淦、兰岸 | 初始版本 |
| 1.1 | 2020.4.22 | 兰岸 | 李宗淦、刘萱 | 修改数据库 |
| 1.2 | 2020.4.23 | 杨嘉成 | 刘萱 | 修改类图 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

# 1 范围

## 项目概述

如今, 机器人作为新时代科技的象征, 各种机器人已步入千家万户, 课程组也为我们准备了嵌入式的课程设计。本项目在课程组提供的软硬件资源的基础傻瓜，设计并开发一个可以在餐厅引导客人入座的ros机器人。

主要功能：

1. 可以对陌生的环境进行建图
2. 可以避开障碍物，如行人，桌子等
3. 机器人可以自动规划路径到达目的地
4. 可以实时显示机器人的摄像头画面

非功能性需求：

1. 友好的用户界面

2. 语音识别

3. 系统稳定运行

## 文档概述

本文是ROS机器人开发的设计说明书。

本文档包含以下八个章节：

1. 范围 概括性描述本项目的内容与本文档的用途，解释相关术语并列举引用文档。
2. 需求概述 从用户视角介绍系统的业务环境、用户、承载的主要业务流程。详细描述系统完整的功能与相关应用场景用例，并给出用例图进行描述。
3. 体系结构设计 描述总体结构和关键问题及解决方案。描述软件体系结构、硬件体系结构、技术体系结构、支撑体系（部署和实施方案）结构等各个方面的设计细节，并使用一系列图表进行说明。
4. 接口设计 说明系统用户界面、系统的软硬件外部接口和内部接口的设计。
5. 数据库说明 以ER图进行描述。
6. 运行与开发环境 明确系统开发与运行的硬件环境与软件环境，并介绍用户界面的设计方案。
7. 详细设计 对于系统的每个关键模块逐个给出一个模块的详细设计方案。包括该模块的类图及其顺序图等内容。
8. 需求可追踪行说明 说明SRS文档中功能同本文档总体结构部分的对应关系，说明设计如何满足需求各项功能和非功能性需求。

## 术语和缩略词

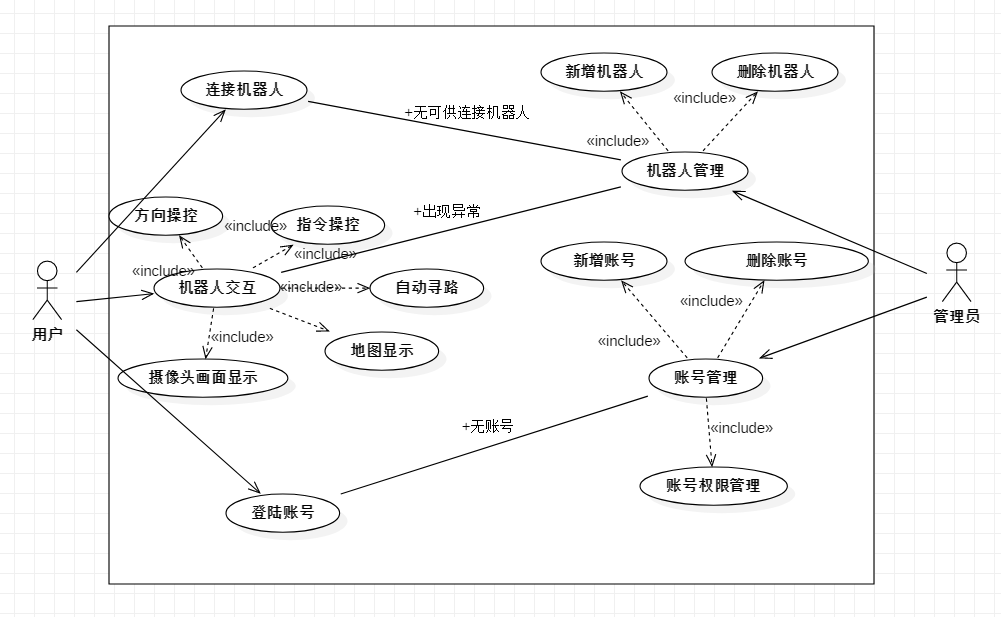
|  |  |
| --- | --- |
| 术语/缩略词 | 解释/全称 |
| ROS | Robot Operating System机器人操作系统 |
| Ubuntu | 以桌面应用为主的Linux操作系统 |
| Ubuntu 16.04 LTS | Ubuntu系统的16.04版本, 为长期支持版 |
| E-R图 | 实体-联系图，提供了表示实体类属性和联系的方法，用来描述现实世界的概念模型 |

## 引用文档

|  |  |  |  |
| --- | --- | --- | --- |
| 编号 | 标题 | 版本 | 发行日期 |
| 1 | 启智ROS机器人开发手册 | V1.1.0 | 2018.11.09 |
| 2 | SRS-107组软件需求规格说明书 | V1.0 | -- |
| 3 | 软件设计说明GB8567-2006 (SDD) | -- | 2006 |

# 需求概述

## 用例图

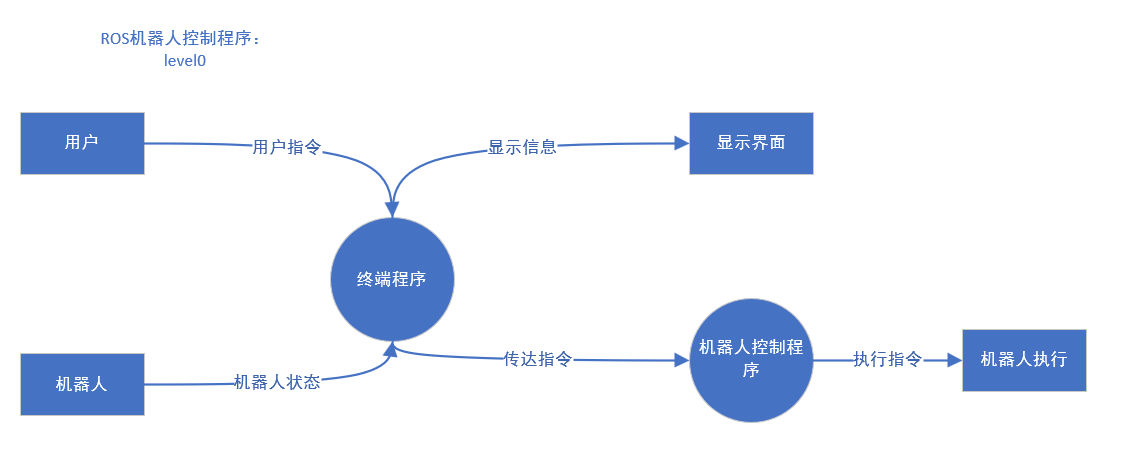


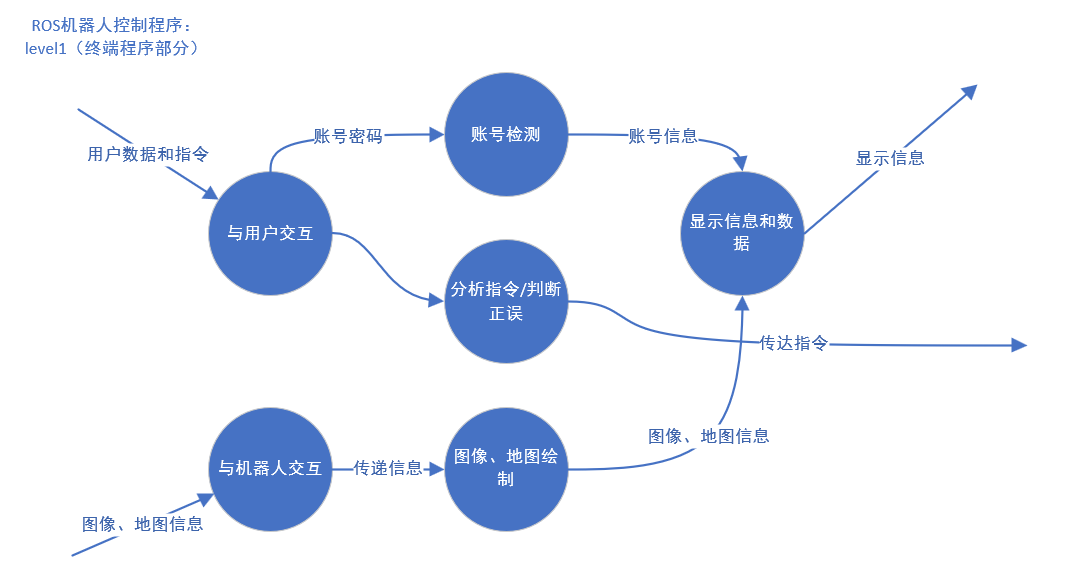
## 用例模板

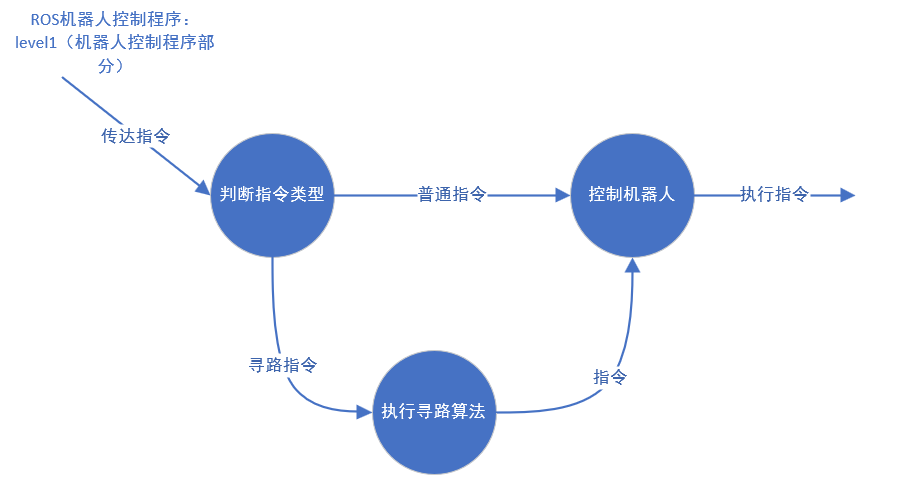
|  |  |
| --- | --- |
| 主要参与者 | 用户 |
| 目标 | 实现机器人的控制和自动寻路 |
| 前置条件 | 用户具有合法的账号和密码 |
| 启动 | 用户决定控制机器人 |
| 场景 | 1. 用户登陆App或者网页 2. 用户输入账号和密码 3. 系统显示可供选择的机器人 4. 用户选择机器人 5. 系统显示机器人控制界面 6. 用户控制机器人行走 7. 系统显示摄像头画面、绘制地图 8. 用户选择“自动寻路”按钮，指定目标地址 9. 机器人完成自动寻路 10. 用户选择“命令交互”按钮 11. 用户输入文本命令 12. 系统分析命令执行正确的命令 |
| 异常情况 | 1. 账号或者密码不正确：场景中的2， 2. 用户没有账号密码，交给管理员处理 3. 无可选机器人，交给管理员处理 |
| 可选方案 | 用户输入语音指令 |
| 次要参与者 | ROS机器人、管理员 |

|  |  |
| --- | --- |
| 主要参与者 | 管理员 |
| 目标 | 管理用户账号、机器人、处理异常状况 |
| 前置条件 | 有可供连接的机器人 |
| 场景 | 1. 管理员选择管理账号 2. 管理员输入密码 3. 管理员选择新增或删除账号 4. 管理员选择新增机器人按钮 5. 系统显示当前可供连接的机器人 |
| 可选方案 | 为每个用户赋予不同的权限 |
| 次要参与者 | ROS机器人、用户 |

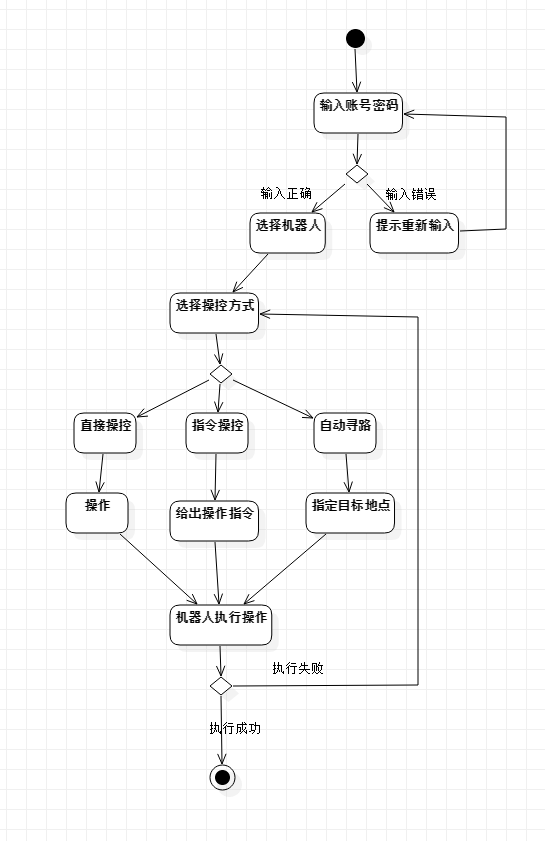
## 数据流图



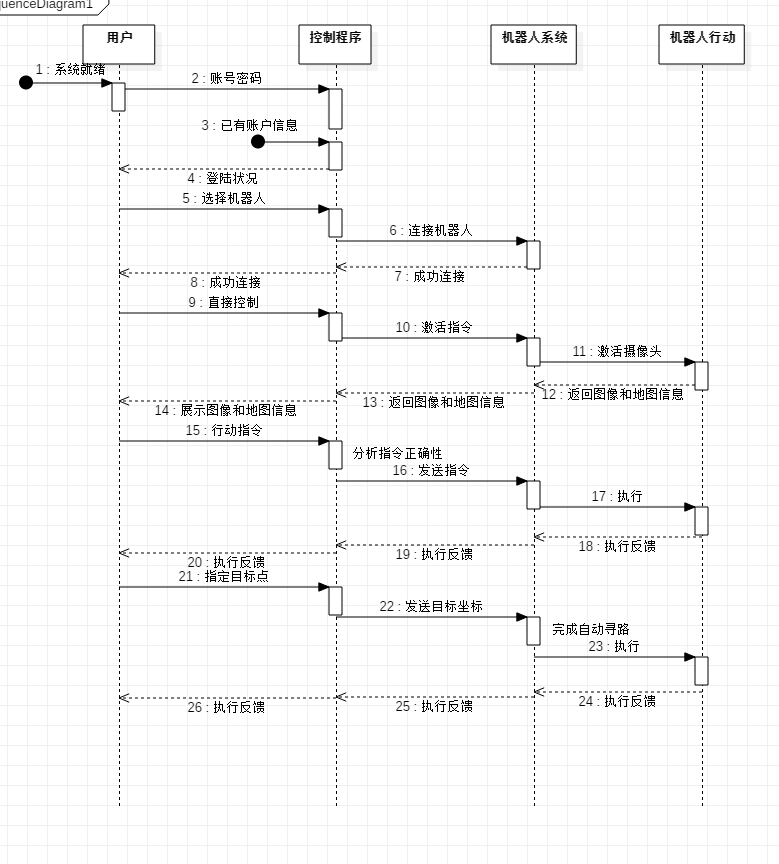




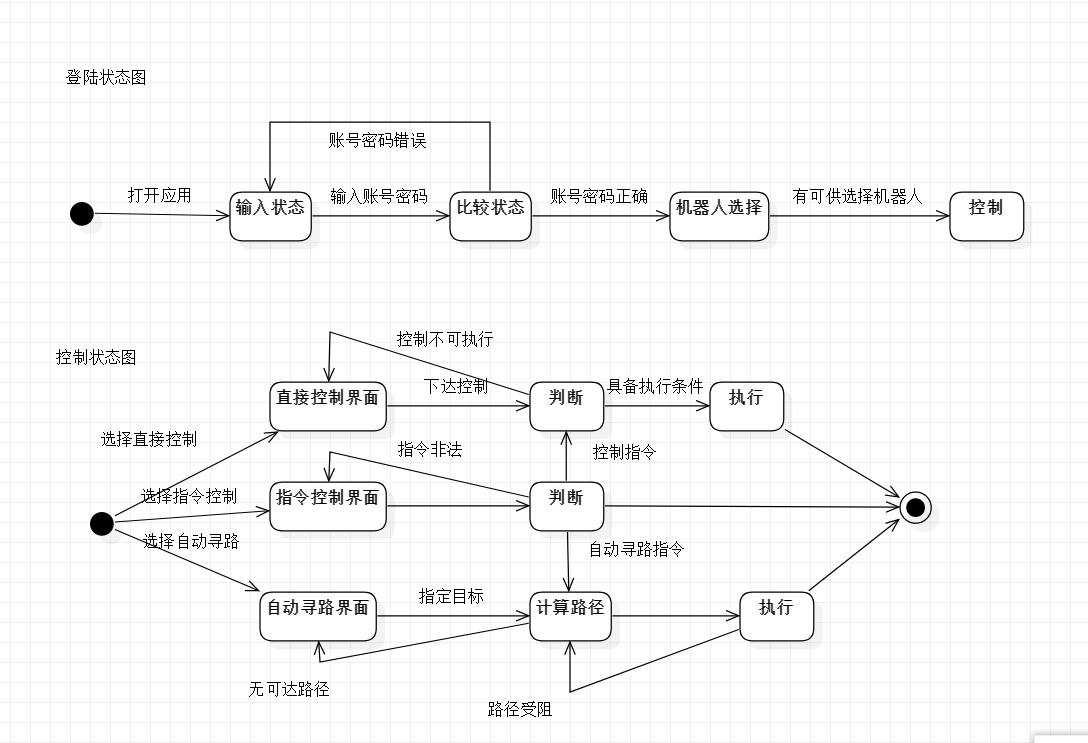
## 活动图



## 时序图



## 状态图

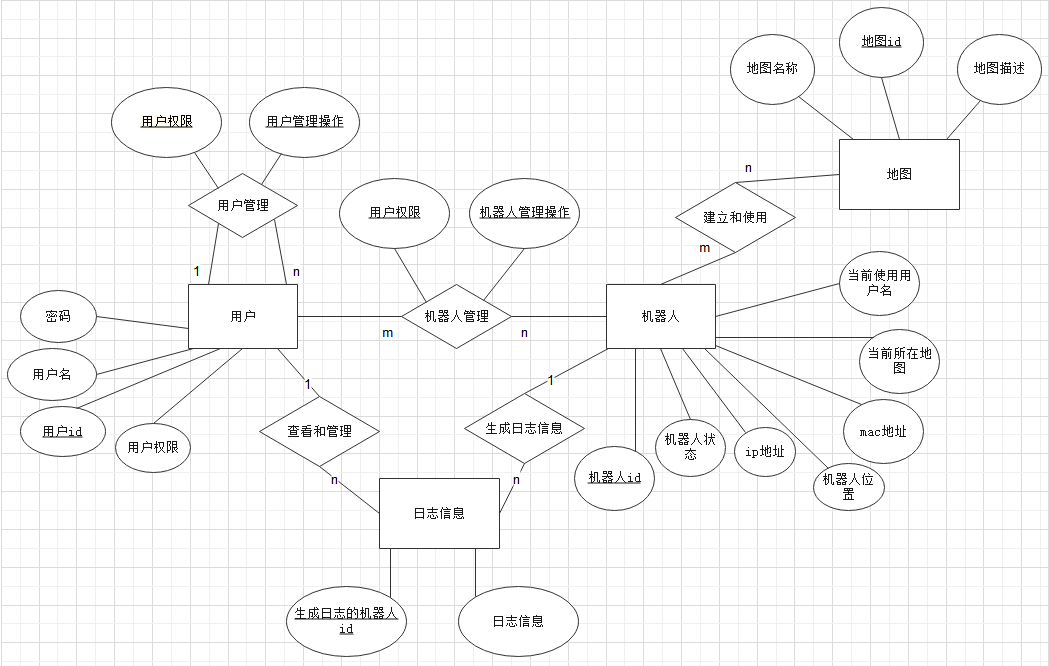


# 数据库设计

## 数据库表

|  |  |  |
| --- | --- | --- |
| **数据表名** | 属性名 | 属性类型 |
| **用户（包含管理员）** | 用户id | Int(16) KEY |
| 用户名 | Char(32) |
| 密码 | Char(64) |
| 用户权限 | Int(16) |
| **机器人** | 机器人id | Int(16) KEY |
| 机器人状态 | Int(16) |
| Ip地址 | Int(16) |
| Mac地址 | Int(16) |
| 当前所在地图id | Int(16) |
| 当前用户id | Int(16) |
| 当前位置横坐标 | Int(16) |
| 当前位置纵坐标 | Int(16) |
| **地图** | 地图id | Int(16) KEY |
| 地图名称 | Char(32) |
| 地图描述 | Char(64) |
| **用户管理** | 用户权限 | Int(16) KEY |
| 用户管理操作 | Int(16) KEY |
| **机器人管理** | 用户权限 | Int(16) KEY |
| 机器人管理操作 | Int(16) KEY |
| **日志信息** | 机器人id | Int(16) KEY |
| 日志时间 | Char(16) |
| 日志信息 | Char(512) |

## E-R图



**图 3.2-1 E-R图**

# 体系结构设计

## 总体结构

### 软件体系结构

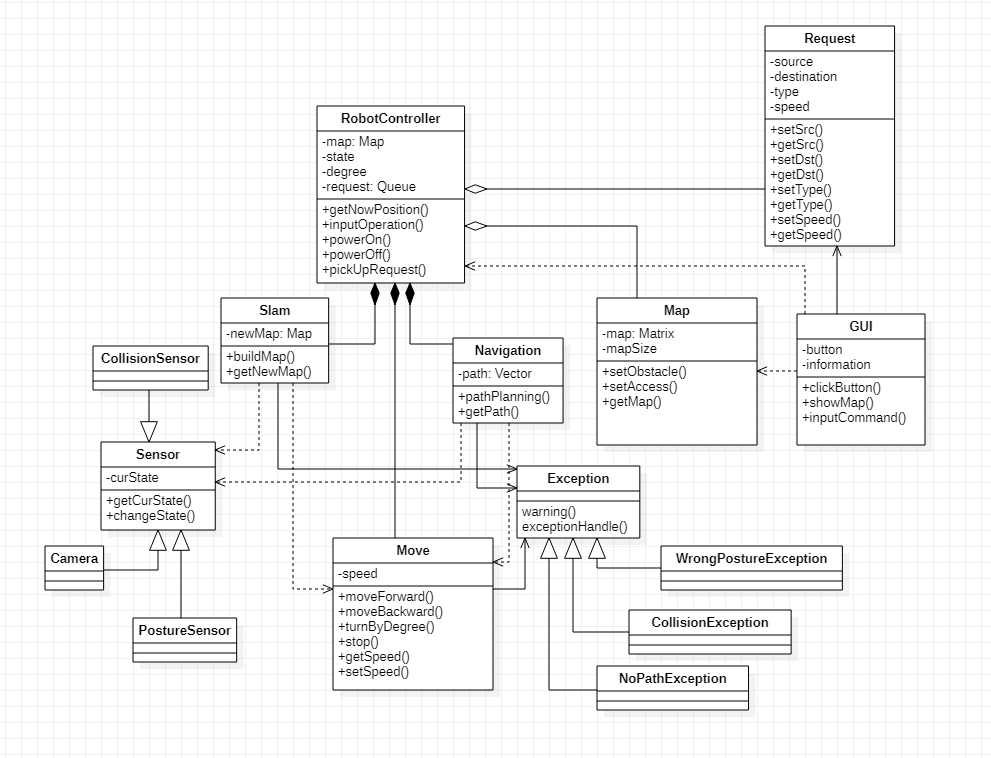


图4.1-1 软件体系结构类图

其中Exception类和Sensor分别有3个子类, 为继承关系。

RobotController类的功能为控制机器人的行走, 因此其由Slam, Navigation, Move三个类组合而成, 而其需要包含Request和Map属性才能正常运行, 因此其与Request和Map的关系为聚合。

### 硬件体系结构

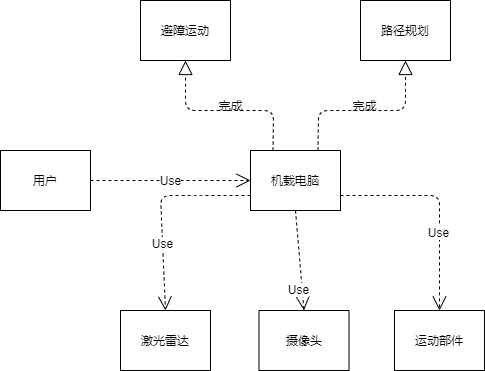


图4.1-2 硬件体系结构

机载电脑使用激光雷达、运动部件完成路径规划和运动；使用激光雷达和摄像头完成避障和地图建立。

### 部署和实施方案

系统的开发与运行都建立在ROS系统之上，源码储存与目标文件都符合ROS的文件管理规则。代码以C++为主，各种脚本辅助编译。在软件发行后，目录结构会去掉源码相关目录，仅保留目标可执行文件。

## 关键问题及解决方案

### 路径规划

路径规划是机器人的重点功能之一，该功能的实现涉及定位和导航两大方面。本设计在采用A\*算法，使用摄像头和激光雷达在已知地图中对机器人进行位置确定；在导航上采用ROS的move\_base包，将导航用到的地图、坐标和行为规划器连接。

需要注意的是，由于算法精度不足、地图校准不精确和机器人初始位置校对不准确等问题，机器人的位置可能会与实际位置存在一定误差，进而可能会出现机器人“远远躲避”障碍物、规划路径离实际障碍物较远，不能得到最优路径的情况。

但由于场景设定，本设计将不会着重考虑前述误差。

# 接口设计

## 系统用户界面

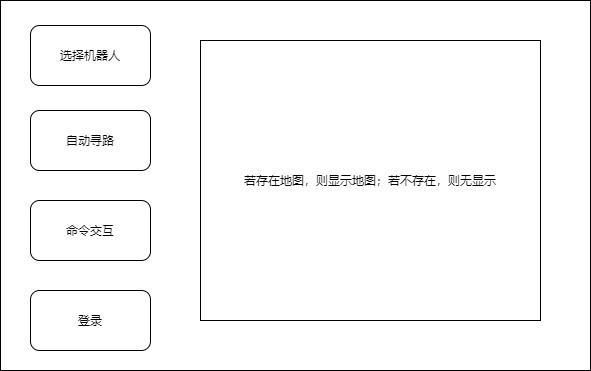


图 5.1-1 系统用户界面示意图

系统用户界面承担了用户与系统进行交互的一部分功能，在左侧的四个按钮分别可以将系统调至“选择机器人”、“自动寻路”、“命令交互”、“登录”模式。右侧的窗口在系统数据库中存在地图时显示地图，若无地图，则无显示。

用户在选取了具体模式后，控制系统将会显示相应的信息和各类控制组件，大致设计如下：

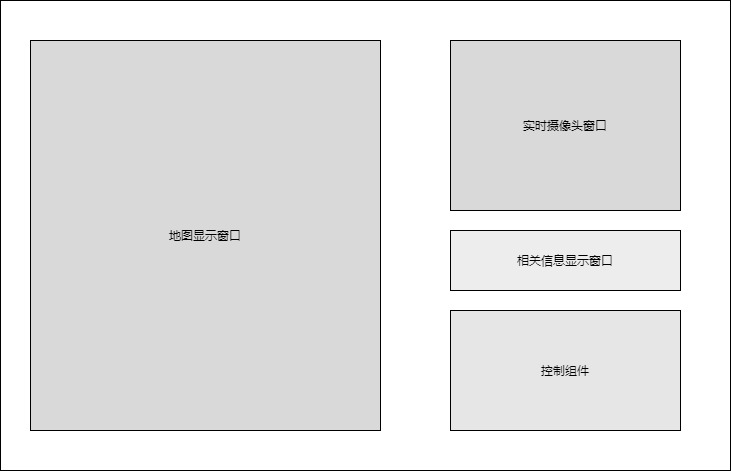


图 5.1-2 系统用户界面控制组件示意图

除了系统用户界面之外，用户也可以在使用过程中通过交互下达命令，进行地图指定目的地的选取等功能。

## 软硬件接口

# 详细设计

## 运动模块

【功能】

运动模块主要负责控制机器人实现基本的运动，主要是平移运动和原地旋转运动。运动模块可以接收位移向量作为输入参数，然后规划机器人的速度和运动时间开始运动。另外运动模块需要接收激光雷达的测距信息，从而对障碍物做出急停反应。

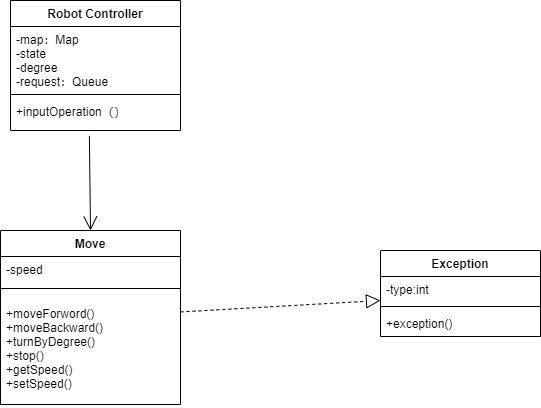
【输入】

位移向量，上层模块根据路径规划等需要，设置位移参数。

【输出】

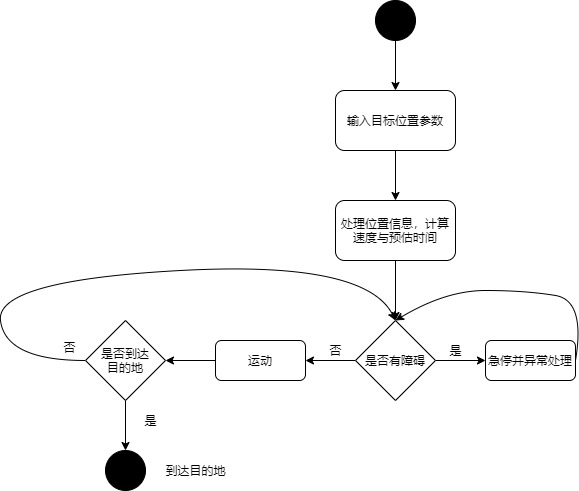
运动控制指令，直接调用ros的内置模块。

【设计】



**图 6.1-1 运动模块类图**

模块的主类是Move，它调用机器人控制系统的方法来操作机器人的运动。同时会检测运动路径上是否有障碍物，可以通过障碍物距离信息来判断，从而保证安全移动。另外还需要异常处理类Exception，对运动过程中的异常进行处理。



**图 6.1-2 运动模块顺序图**

## 路径规划模块

【功能】

通过A\*算法规划路径，能够使机器人从当前点出发沿路径运动，避开障碍物，到达目标点。

【输入】

地图、目标点。

【输出】

从当前点到目标点的路径，机器人下个时刻的速度和方向向量。

【设计】

主类为PathAgent。通过getPath方法调用Astar类的PathFinding方法。在Astar类中，通过PathFing方法可以执行A\*寻路算法，需要信息为当前位置、地图信息、目标点位置。对于路径中的每一个位置，可以计算出距离起始点走过的路程以及距离目标点的曼哈顿距离，通过估价函数求出下一时刻的速度大小和方向向量。完成从当前点到目标点的路径规划。ExceptionHandler处理异常情况。

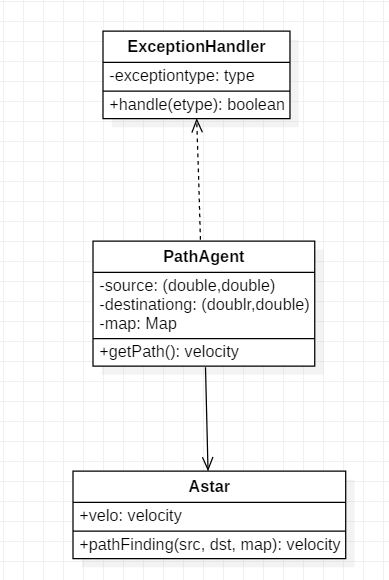


图6.2-1路径规划模块类图

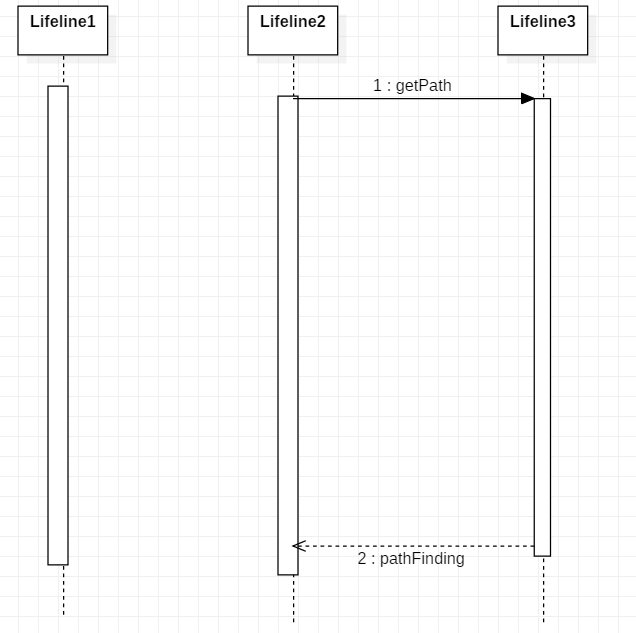


图6.2-2 路径规划模块时序图

## 建图模块

【功能】

建图实现即时定位与当前场景的地图构建。根据激光雷达的扫描结果，使用gmapping算法构建地图并保存地图。

【输入】

激光雷达扫描结果。

【输出】

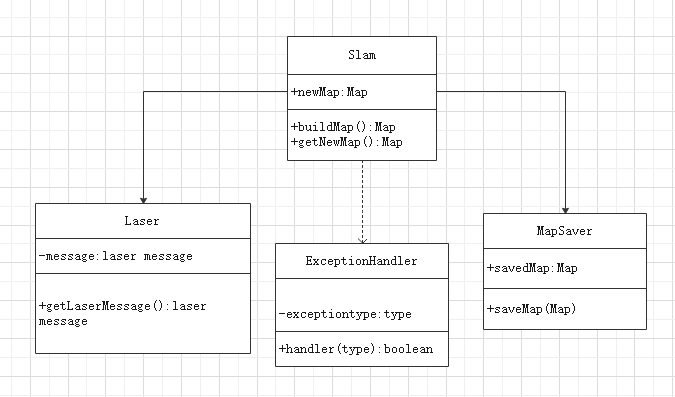
当前场景的地图，保存为“map.pgm”和“map.yaml”。其中PGM为可查看的图片格式。YAML文件描述了地图元数据，并命名了图像文件。图像文件对占用数据进行编码。YAML文件包含：

* **image** : YAML文件的相对路径
* **resolution** : 地图的分辨率，单位是m / 像素。
* **origin** : 地图中左下角像素的二维表示，为（x，y，yaw），偏航为逆时针旋转（偏航= 0表示无旋转）。
* **occupied\_thresh** : 像素值高于此阈值被认为是完全占用。
* **free\_thresh** : 像素值低于此阈值被认为是完全空闲。
* **negate** : 是否应该反转白/黑，自由/占用语义（阈值的解释不受影响）

**【设计】**

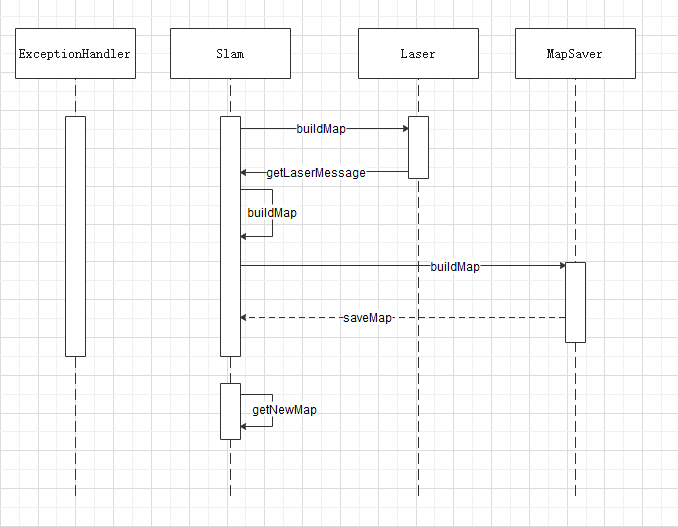
主类是Slam类。Slam的buildMap方法首先调用Laser类的getLaserMessage方法获取激光雷达数据，根据Laser类传递的数据通过Hector SLAM程序建图，将结果传递给Mapsaver类。Mapsaver使用ROS的map\_server包，将当前 SLAM 建好的图保存。Slam的getNewMap方法获取已保存的地图。ExceptionHandler处理异常情况。

类图如下：



**图 6.3-1 建图模块类图**

时序图：



**图 6.3-2 建图模块时序**

# 运行与开发环境

## 运行环境

|  |  |
| --- | --- |
| **硬件环境** | 功能 |
| 启智ROS机器人 | 主体，能实现自动寻路、避障等功能 |
| 计算机 | 机器人控制平台 |
| 安卓手机 | 机器人控制平台 |

|  |  |
| --- | --- |
| **软件环境** | 功能 |
| Ubuntu 16.04 LTS | 计算机操作系统 |
| Android 5.0以上操作系统 | 手机操作系统 |
| Chrome等浏览器 | 运行网页端机器人控制程序 |
| 机器人控制App | 控制机器人 |

## 开发环境

|  |  |
| --- | --- |
| **硬件环境** | 功能 |
| 计算机 | 开发的主体，完成代码等的编写 |
| 安卓手机 | App的调试 |
| 启智ROS机器人 | 机器人的调试 |

|  |  |
| --- | --- |
| 软件环境 | 功能 |
| Ubuntu 16.04 LTS | 计算机操作系统 |
| Android 5.0以上操作系统 | 手机操作系统 |
| Kinetic | 机器人代码开发 |
| RoboWare Studio | 机器人代码开发 |
| Android Studio | 安卓端控制App开发 |
| Dreamweaver等 | 网页端控制程序开发 |

# 需求可追踪性说明

## 功能性需求

### 地图初始化

针对SRS文档涉及的地图建立功能，本设计中借助机器人的激光雷达进行扫描、使用gmapping算法进行建图。

激光雷达会将扫描结果保存到特定文件，通过提取该文件并使用gmapping算法进行处理，我们就能得到准确可用的地图；随后经过使用激光雷达来对机器人位置进行校准之后，机器人就站在地图的相应位置。

### 路径规划

针对SRS文档涉及的路径规划功能，由于机器人的使用场景为餐厅引路，本设计中A\*算法进行路径规划。A\*是路径搜索中最受欢迎的选择，因为它相当灵活，并且能用于多种多样的情形之中。这是一种启发式搜索算法，启发式搜索就是在状态空间中的搜索对每一个搜索的位置进行评估，得到最好的位置，再从这个位置进行搜索直到目标。这样可以省略大量无谓的搜索路径，提高了效率。

### 主动控制

针对SRS文档设计的主动控制功能，本设计在用户界面提供了主动控制按钮、当前机器人摄像头画面的实时显示，以及当前机器人处于地图的具体位置，用户可以在控制界面对机器人进行主动的控制。

### 避障

针对SRS文档中设计的避障功能，本设计主要采用了机器人自带的摄像头及激光雷达进行避障。当机器人的激光雷达检测到前方有障碍物时会停止，然后调用规划算法，生成新的路径以规避障碍并前往目标地点。

### 用户交互

针对SRS文档中涉及的用户交互功能，本设计主要实现了界面交互。用户通过Web端和移动端可以对机器人进行管理和使用，通过按钮和聊天框交互，对机器人下达指令。同时，也可以在交互界面获得当前地图及路径的相关信息。

## 非功能性需求

### 性能属性

1. 从用户发出指令，中断分析指令传给机器人到机器人开始执行，不能让用户感觉到明显的时延（超过10秒）。
2. 功耗方面，要尽可能减少机器人的功耗，保证机器人的续航时间。
3. 要能够实时显示摄像头画面，能够绘制当前环境的地图，能够实现用户下达的指令并及时反馈，能够实现避障和自动寻路功能。

### 质量属性

1. 要尽可能做到对用户友好，方便用户操作，设计控制界面应该简洁易懂。反馈要及时。
2. 要做到多端的同步，安卓端和网页端实现功能应该大致相同。
3. 要具有一定的鲁棒性，当机器人出现问题时，要能够及时做出反应，并反馈给管理员解决问题。
4. 拥有一定的扩展能力，方便后续迭代加入语言识别、用户权限等功能。

## 用户界面需求

### 登陆界面

设计账号输入框和密码输入框，登陆按钮以及新增账号按钮

### 管理员后台界面

管理员密码输入框、账号输入框和密码输入框以及确认密码输入框，确认按钮，用户权限增加、减少按钮。

### 机器人选择界面

右上角设计新增机器人按钮，正中显示可供连接的机器人

### 控制界面

**直接控制界面：**显示机器人摄像头画面，设计方向键控制按钮**；**显示已经绘制的地图，能够指定目标地点

**命令交互界面：**以聊天界面的形式显示，下方设计编辑组件，供用户输入命令，并预留语音识别接口