**简易机器人**

**软件设计说明书**

**SDD Team110**

**1.6**

分工说明

|  |  |  |
| --- | --- | --- |
| 小组名称 | 稳稳的ROS | |
| 学号 | 姓名 | 本文档中主要承担的工作内容 |
| 17231131 | 宋卓煜 | 完成第六部分机器人避障移动部分，第7，8部分 |
| 17373476 | 张岑岳 | 完成文档的1,2,3,5部分 |
| 17373022 | 张佳豪 | 完成体系结构设计部分的内容。 |
| 16061149 | 季子涵 | 完成第六部分机器人检测抓取部分 |

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 | 4月19日 | 张岑岳 | 宋卓煜 | 完成文档1，2，3，5部分 |
| 1.1 | 4月19日 | 张佳豪 | 宋卓煜 | 完成文档第4部分 |
| 1.2 | 4月19日 | 宋卓煜 | 季子涵 | 完成文档第6.1~6.4 |
| 1.3 | 4月19日 | 季子涵 | 宋卓煜 | 完成文档第6.5~6.7 |
| 1.4 | 4月20日 | 宋卓煜 | 全体组员 | 完成文档7，8部分 |
| 1.5 | 4月26日 | 宋卓煜 | 张佳豪 | 修改文档概述，需求可追踪，图表编号 |
| 1.6 | 5月19日 | 张佳豪 | 宋卓煜 | 修改详细设计部分，修改第4部分 |

目录

[1范围 1](#_Toc40888767)

[1.1 项目概述 1](#_Toc40888768)

[1.1.1 项目背景 1](#_Toc40888769)

[1.1.2 功能和需求 1](#_Toc40888770)

[1.1.3 应用场景 1](#_Toc40888771)

[1.2 文档概述 2](#_Toc40888772)

[1.3 术语和缩略词 2](#_Toc40888773)

[1.4 引用文档 2](#_Toc40888774)

[2. 需求概述 3](#_Toc40888775)

[2.1 业务需求 3](#_Toc40888776)

[2.1.1 用户通过控制面板启动/关闭系统 3](#_Toc40888777)

[2.1.2 用户通过导航地图标点指引机器人的移动 4](#_Toc40888778)

[2.1.3 用户通过语音指令指挥机器人抓取运送 5](#_Toc40888779)

[2.2 数据需求 5](#_Toc40888780)

[2.3 功能需求 6](#_Toc40888781)

[2.4 非功能需求 7](#_Toc40888782)

[2.5 用户界面需求 8](#_Toc40888783)

[3. 数据库设计 10](#_Toc40888784)

[4. 体系结构设计 11](#_Toc40888785)

[4.1 总体结构 11](#_Toc40888786)

[4.2 软件体系结构 12](#_Toc40888787)

[4.3 硬件体系结构 15](#_Toc40888788)

[4.4 技术体系结构 16](#_Toc40888789)

[5. 接口设计 17](#_Toc40888790)

[6. 详细设计 17](#_Toc40888791)

[6.1 导航模块 17](#_Toc40888792)

[6.2 机器人运动模块 18](#_Toc40888793)

[6.3 建图模块 19](#_Toc40888794)

[6.4 物品检测与识别 20](#_Toc40888795)

[6.5 物品抓取 21](#_Toc40888796)

[6.6 语音识别模块 23](#_Toc40888797)

[7. 运行与开发环境 24](#_Toc40888798)

[7.1 运行环境 24](#_Toc40888799)

[7.2 软件环境 24](#_Toc40888800)

[8. 需求可追踪性说明 25](#_Toc40888801)

[8.1 功能需求 25](#_Toc40888802)

[8.1.1 地图建模 25](#_Toc40888803)

[8.1.2 避障移动 25](#_Toc40888804)

[8.1.3 检测抓取 25](#_Toc40888805)

[8.1.4 用户功能 26](#_Toc40888806)

[8.1.5 语音识别/播报 26](#_Toc40888807)

[8.2 非功能需求 26](#_Toc40888808)

[8.2.1 机器人避障效率 26](#_Toc40888809)

[8.2.2 机器人语音响应能力 26](#_Toc40888810)

[8.2.3 机器人抓取能力 27](#_Toc40888811)

[8.2.4 机器人运动特性 27](#_Toc40888812)

[8.2.5 机器人的有效负载能力 27](#_Toc40888813)

[8.2.6 系统健壮性与可用性 27](#_Toc40888814)

# 1范围

## 项目概述

### 项目背景

随着机器人领域的发展与相关技术的逐渐成熟，各个国家开始逐渐用机器人替代某些领域的人工劳动力，并取得了显著的效果，使得生产力进一步增加，生产效率也进一步提高。伴随着机器学习等领域的发展，机器人在识别检测方面，变得更加智能化，精准化。因此，机器人也越来越多地出现在我们的日常生活中，比如扫地机器人，餐厅送菜机器人，银行语音服务机器人等等。

### 功能和需求

本项目旨在开发一个拥有检测、识别并定位环境中的目标、动态接近目标物、抓取目标物三大功能的机器人。在检测识别目标物之前，机器人应首先学会主动接近特定地点的目标物。这也意味着机器人应当拥有导航功能，并且能够在前往导航地点时，对障碍物和行人进行避让。其次，机器人应可以通过语音指令，明白其抓取的目标是什么。因而，机器人需要拥有智能语音识别功能。之后，机器人要完成目标抓取功能，需要实现识别检测功能，以及动态抓取功能。最后，机器人在抓取目标物后，应当可以返回初始点或另一个目标点，去递送该目标物。

在实现上述功能的基础上，我们也对机器人的非功能性需求有所关注。首先，是识别能力上，这包含了机器人能够识别物品的种类，机器人对物品识别的精确度，机器人对语音指令识别的精确度。其次，在机器人的避障效率，这取决于机器人的避障策略，机器人的路径规划策略。最后是对机器人的抓取能力上，这包含了机器人能够抓取的物品种类，抓取物品后运送过程的稳定性等。

### 应用场景

在我们来看，本项目实现的具有检测识别与定位、动态移动、目标抓取多功能于一体的机器人具有非常广泛的应用场景。例如，在智能家居方面，该机器人可以充当扫地机器人、智能管家、智能家庭助理等；在工厂方面，该机器人可以充当物品搬运与摆放、零件装配等功能；在物流方面，该机器人可以实现全自动取货送货的功能；在服务行业，该机器人可以充当机器服务员，用于送餐等等。此外，在一些特殊场景，该机器人也有着意想不到的作用。例如，在消防方面，该机器人可以进入危险场景取出重要物品。

## 文档概述

本文档包含了对整个项目的软件设计分析，具体内容有数据库设计，体系结构设计，接口设计，对避障移动和检测抓取两个核心模块的具体设计，以及需求的可追踪性描述。主要用于描述整个项目开发的软件体系设计思想，为项目开发的具体实现提供指导。

## 术语和缩略词

表1-3-1 术语-缩略词对应表

|  |  |  |
| --- | --- | --- |
| 术语 | 英文全称 | 缩略词 |
| 简易机器人 | Simple Robot | SR |
| 物品抓取 | Object Grab | OG |
| 语音控制 | Speech Control | SC |

（后续随时更新）

## 引用文档

1. 《启智ROS开发手册》V1.1.0 2018
2. 《SRS-开发计划模板》 2020版 2020-3，软件工程课程组

# 需求概述

## 业务需求

### 用户通过控制面板启动/关闭系统

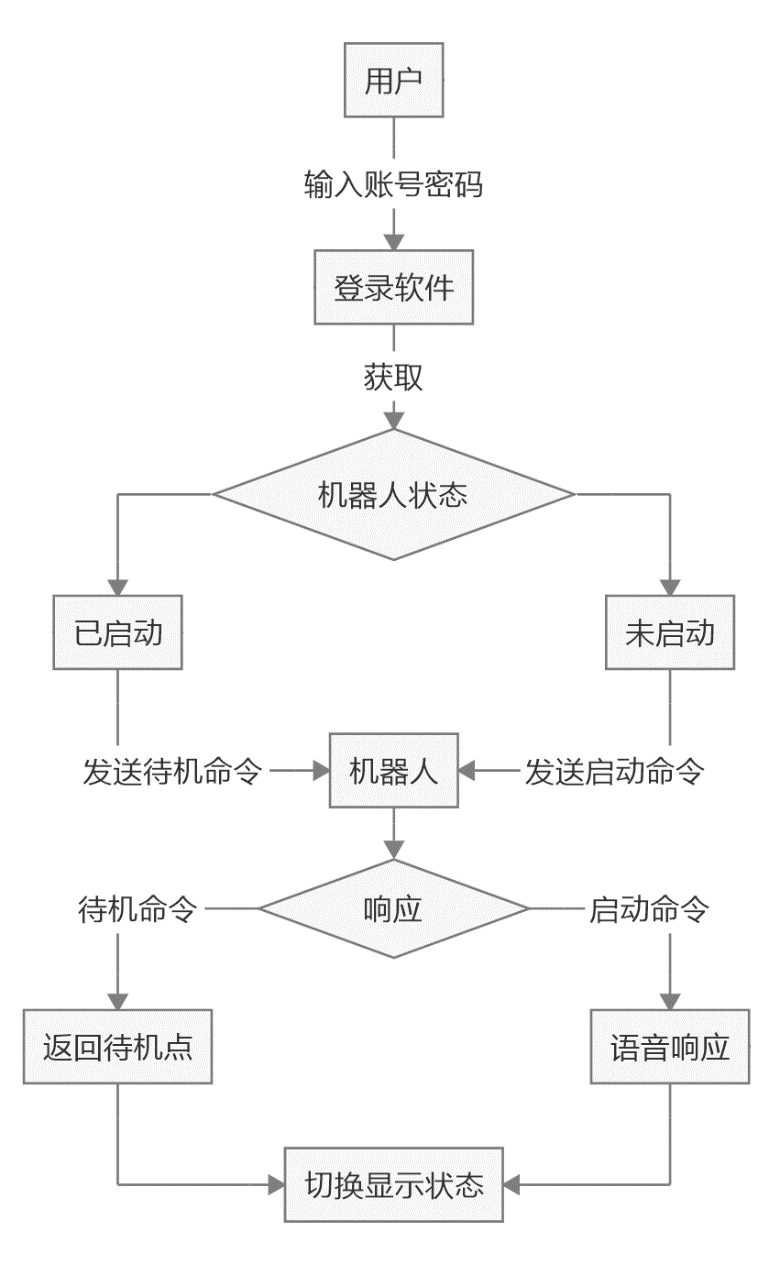


图2-1-1 用户启动/关闭系统流程图

### 用户通过导航地图标点指引机器人的移动

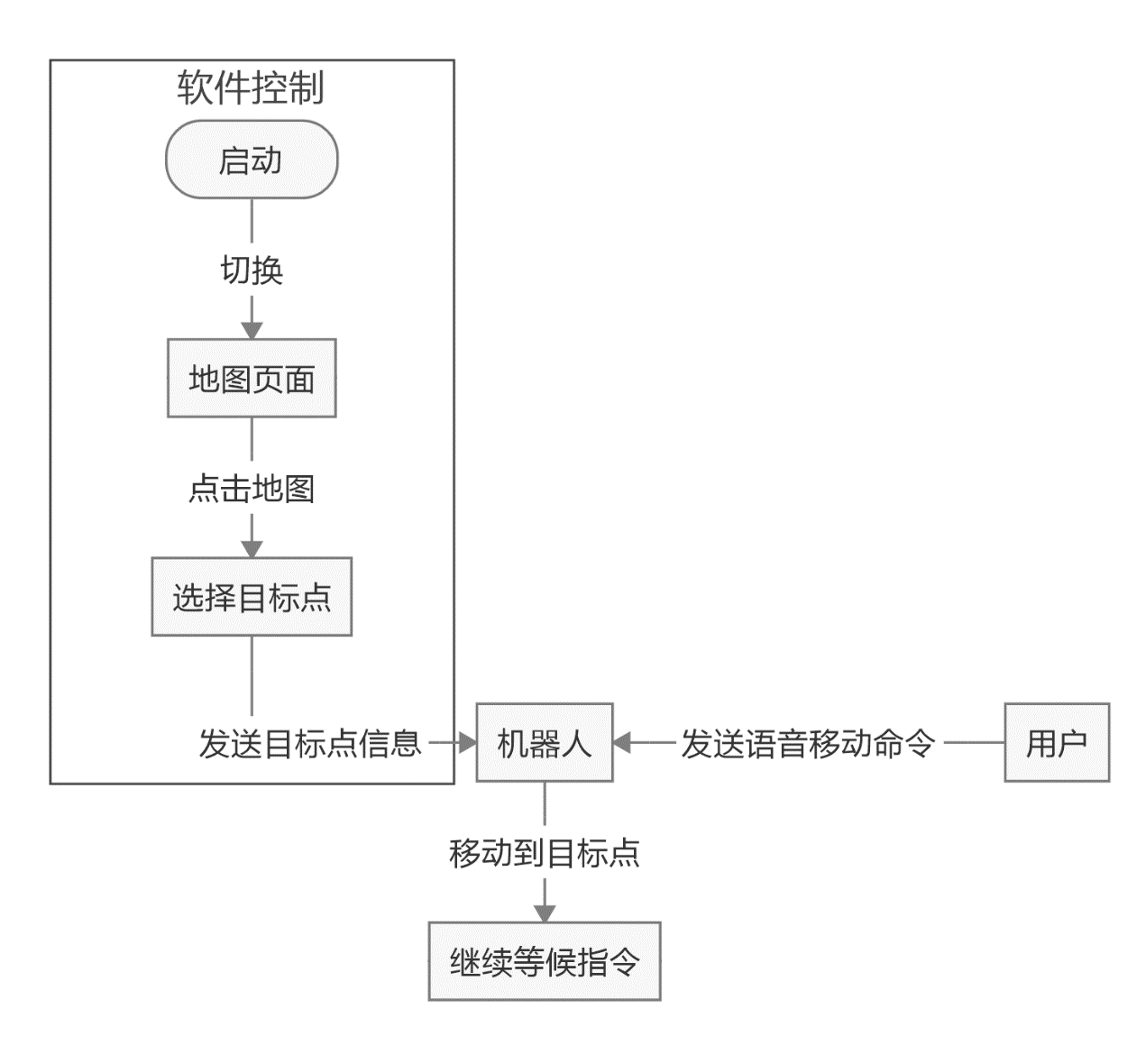


图2-1-2 用户指定机器人移动流程图

### 用户通过语音指令指挥机器人抓取运送

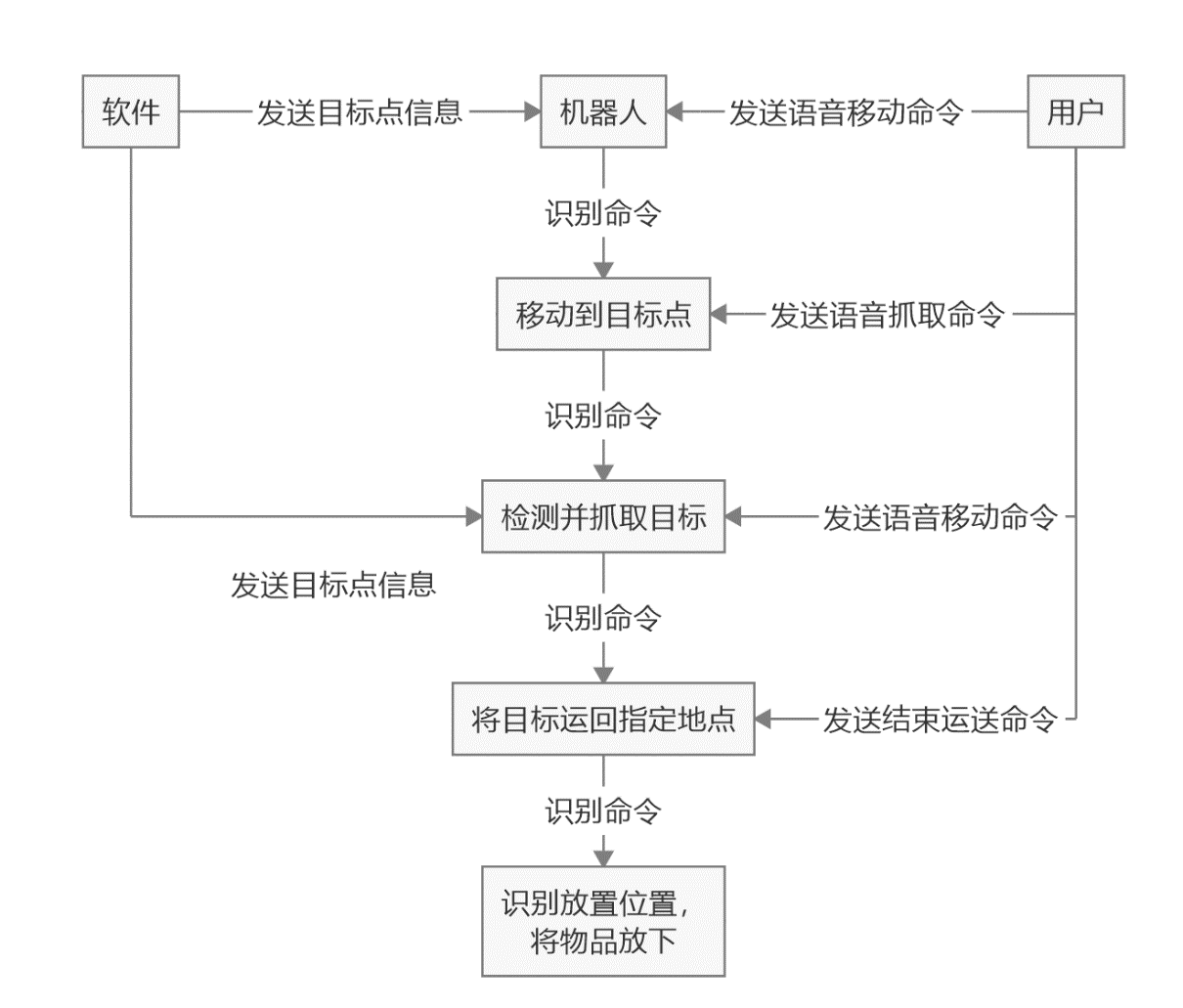


图2-1-3 用户控制机器人抓取流程图

## 数据需求

数据包含了用户发出的语音指令，用户通过app发出的指令，地图信息，目标物检测信息，路径信息，障碍信息，机器人发出的语音播报信息。

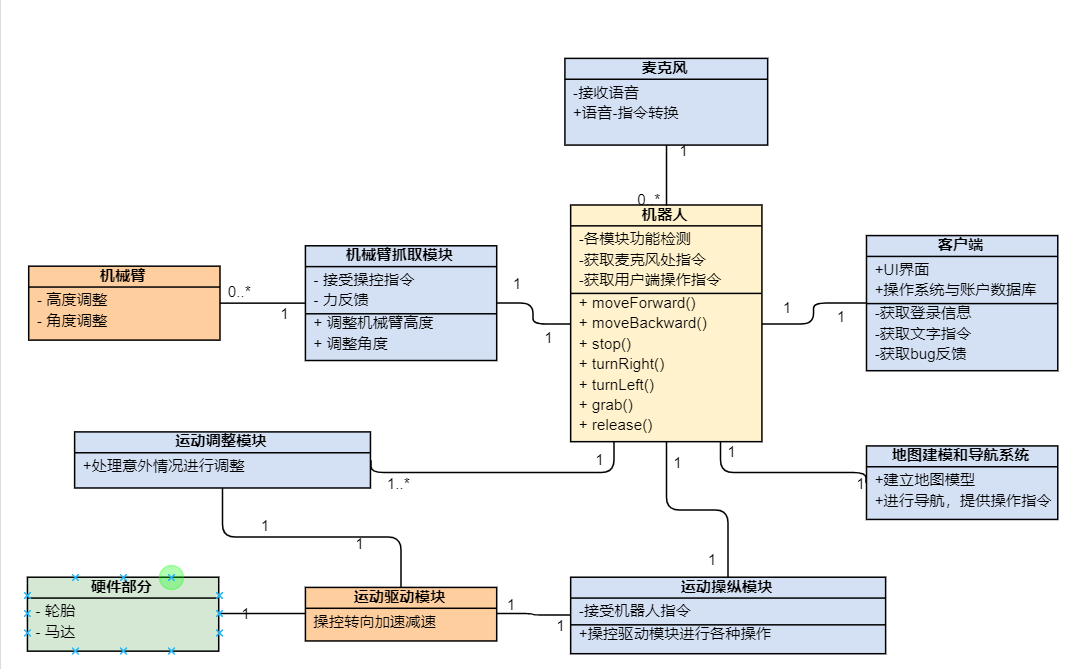


图2-2-1 机器人功能类图

## 功能需求

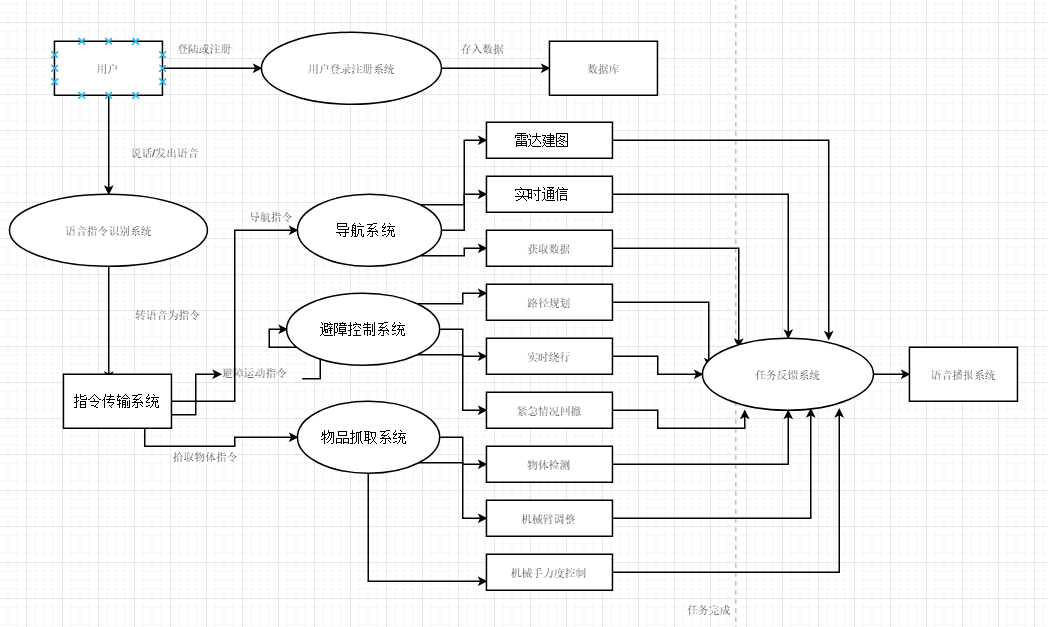


图2-3-1 机器人功能数据流图

在我们的系统中，最关键的两个功能模块为机器人避障移动和自主抓取。

机器人避障移动，主要是在机器人导航模块的主导下，先根据地图和起止点信息，规划出线路。然后，在机器人沿路径途中，边移动边检测障碍物。如果检测到障碍物，则将障碍物信息返回至导航模块，由导航模块调用路径规划模块重新规划出新的路径，从而完成避障。

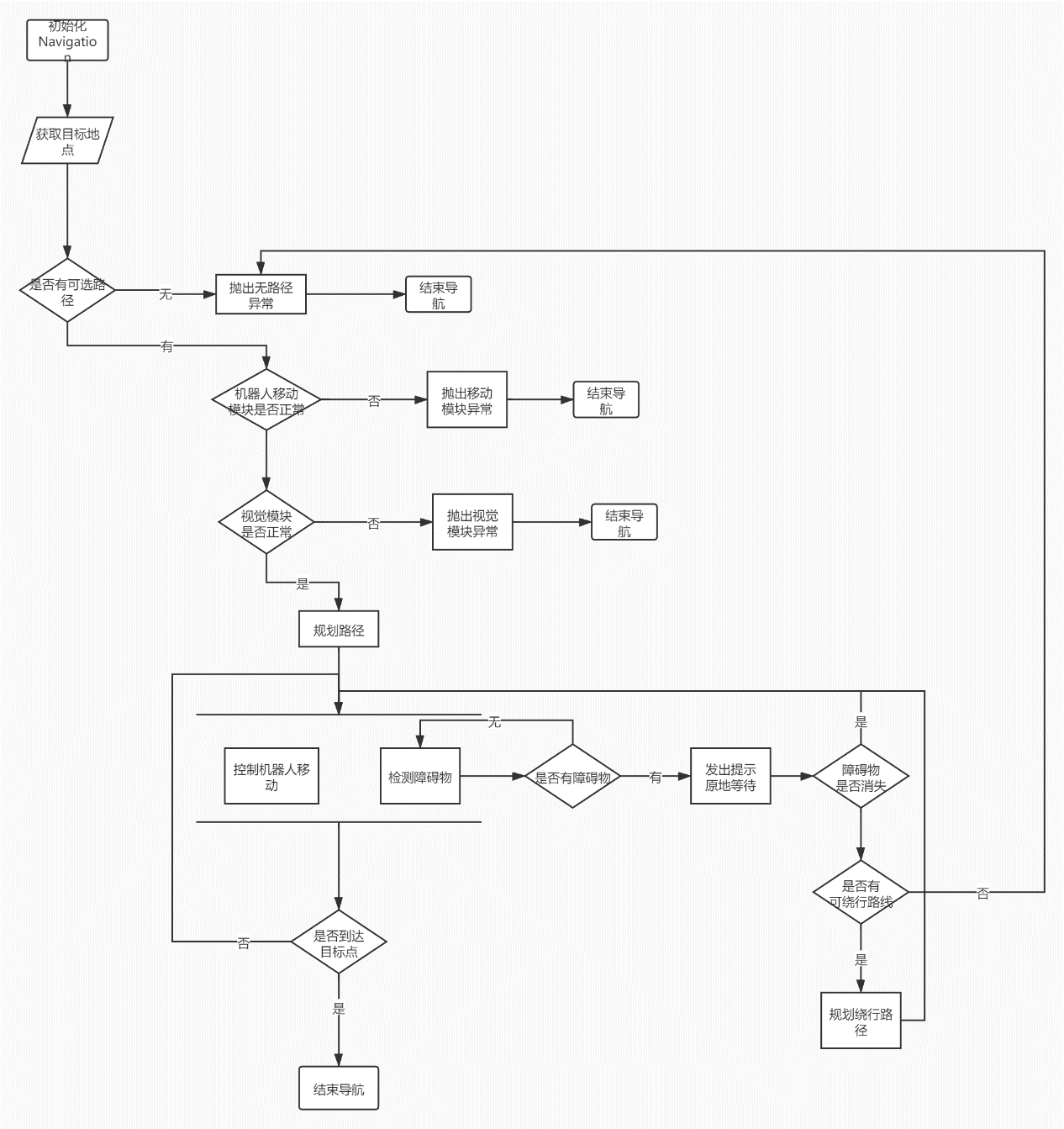


图2-3-2 机器人避障移动流程图

## 非功能需求

此处我们需要对机器人的避障效率，避障之后与原轨道之间的偏差，机器人语音的响应能力，抓取能力，运动特性，负载能力等进行综合评估

## 用户界面需求

表2-5-1 用户界面需求描述表

|  |  |  |
| --- | --- | --- |
| 登录页面 | 包含登录信息以及注册选项，登录使用用户名和密码，大概布局如右图中所示。 |  |
| 个人页面 | 登录成功后会进入用户个人页面。大概布局如右图所示，显示个人用户名。下面会有机器人页面选项，地图页面选项，用户意见反馈页面选项，以及联系我们选项。 |  |
| 操作页面 | 选择机器人页面选项后所显示的大概布局如右图所示。页面主要显示机器人当前状态，图中包含两个按钮分别代表启动机器人按钮和关闭机器人按钮。 |  |
| 地图显示页面 | 选择地图页面后所显示的大概布局如右图所示，地图显示页面里会显示已经建立好的地图数据，而后可以如图中绿色标识点一样，可以选择对地图中某点进行标点，而后显示。 |  |
| 用户反馈页面 | 选择用户意见反馈页面后所显示的大概布局如右图所示，可以选择反馈类型，而后填写具体情况，有所填写后便可以选择最下方的确认提交按钮进行提交。 |  |

# 数据库设计

系统数据库分为以下几个层面分别进行处理：

用户层：包括注册的用户，管理员，对应的账号密码，以及可以访问的权限，反馈问题记录，更新日志等

传输层：输入语音和转换的文本指令的数据库，每次返回结果（成功或者故障）的记录

功能层：地图建模的缓存，每次运动路线，抓取物品的记录

# 体系结构设计

## 总体结构



图4-1-1 系统层次结构图

从总体来看，整个系统一共分为了五个层次。最上一层为用户层，主要起到了与用户或者管理员进行交互的作用，提供了账户注册/登录、机器人启动/关闭、语音指令接收、意见反馈、结果返回、地图构建等功能。

第二层传输层是一个中间层，起到了用户层和功能层之间的过渡作用。传输层通过解析用户层接收到的语音，生成对应的指令对功能层进行调用，获得返回结果，并根据返回结果进行语音播报。如果返回结果为异常，则还要进行异常处理。如果返回结果正常，则将结果返回给用户层。

第三层为功能层，该层通过调用数据层的数据及接口，实现各种功能，包括地图建模、路径规划、避障移动、物品识别和抓取、用户数据修改、结果返回、异常返回等。

第四层为数据层，该层读取硬件层的硬件，并储存各种必要的信息，例如储存硬件信息、用户数据、图像数据、路径信息、抓取物品信息等各种数据。

最下层为硬件层，主要包括激光雷达、机械臂、麦克风、摄像头、移动底盘、俯仰调节装置等各种机器人硬件。

## 软件体系结构

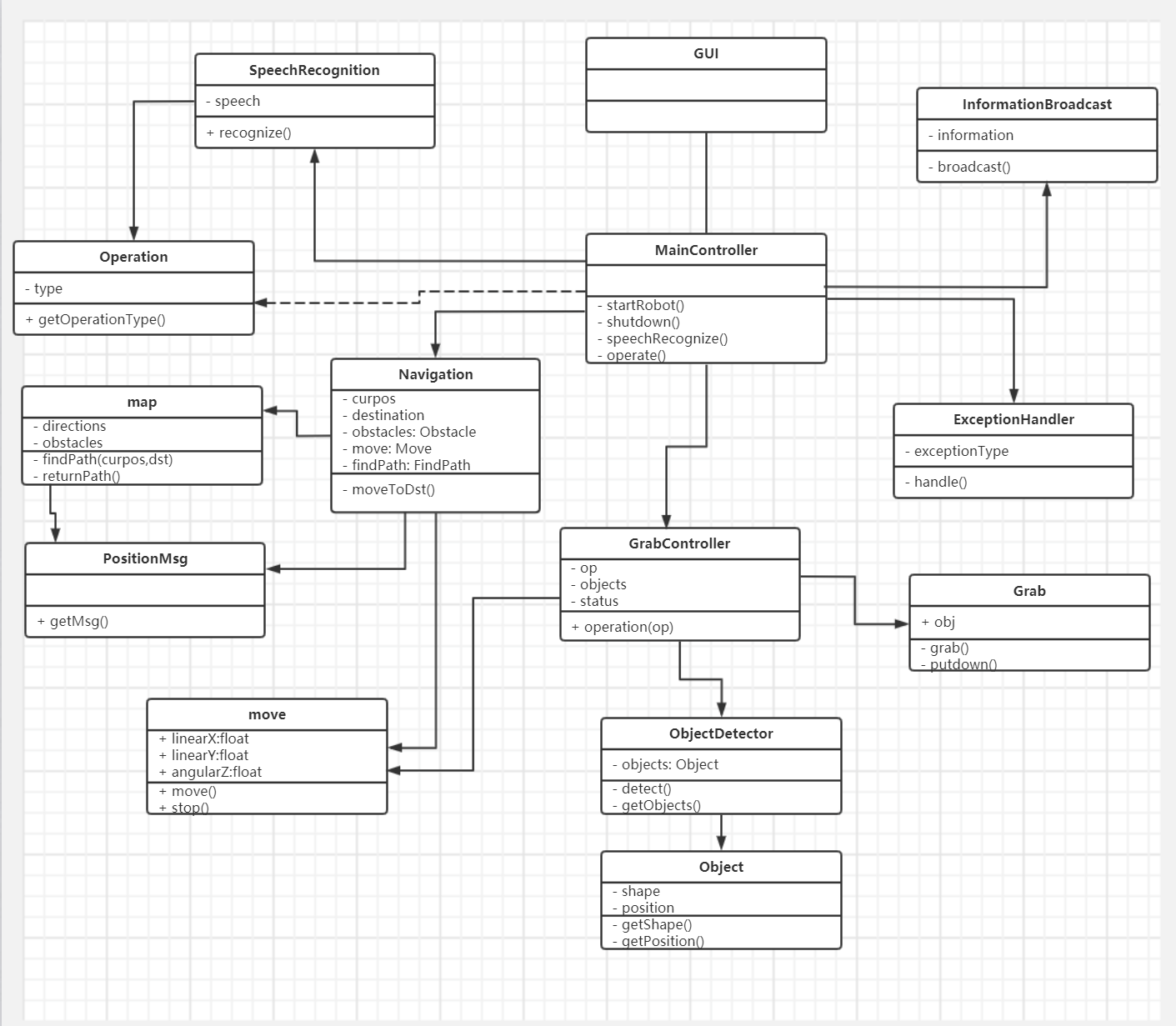


图4-2-1 软件类图

软件类图如上图所示，具体各个类的属性和方法在详细设计部分给出，这里只给出了简单的方法和属性。各个类的主要功能为：

GUI：提供用户图形界面。

SpeechRecognition: 语音识别类，负责处理语音文件，生成相应的指令操作。

InformationBroadcast：信息广播类，负责将机器人运动信息通过语音合成广播出来，该类与SpeechRecognition类共同实现了语音交互。

Operation：操作类，由SpeechRecognition类实例化，MainController的方法使用该类作为参数，从而实现对整个系统的控制。

MainController：主控制类，用来控制整个系统的运行。

Navigation：导航类，该类记录了机器人当前位置以及机器人目标地点位置的信息，通过这些信息调用path类的findPath方法，实现路径规划，得到路径信息，再根据路径信息调用Move类，实现机器人的移动。

Move：移动类，使用机器人的物理接口，控制机器人移动。

Path：路径类，实现了findPath方法，该方法通过调用ObstacleDetector类，得到障碍物信息，再根据障碍物信息以及机器人当前位置、目标位置等信息，使用路径规划算法，得到路径。

ObstacleDetector：障碍物检测类，通过调用激光雷达接口，扫描障碍物，并记录下距离机器人一定范围内的障碍物位置、形状等信息，将这些信息实例化为Obstacle类。

Obstacle：障碍物类，记录了障碍物的位置、形状等信息，一个Obstacle对象表示一个障碍物。

GrabController：抓取控制类，再得到抓取指令后，该类先调用ObjectDetector类的detect()方法，获取物品位置、形状等信息，在调用Grab类的grab()方法，将物品抓起来，当将物品移动到指定位置后，再调用Grab类的putdown()方法，将物品放下，从而实现物品的抓取控制。

Grab：抓取类，直接使用机器人硬件接口，实现物品的抓取和放置。

ObjectDetector：物品检测类，通过调用立体相机硬件，实现物品检测模块，获取物品的位置、形状信息，并将这些信息实例化为Object对象。

Object：物品类，包括物品的形状、位置信息等。一个Object对象表示一个物品。

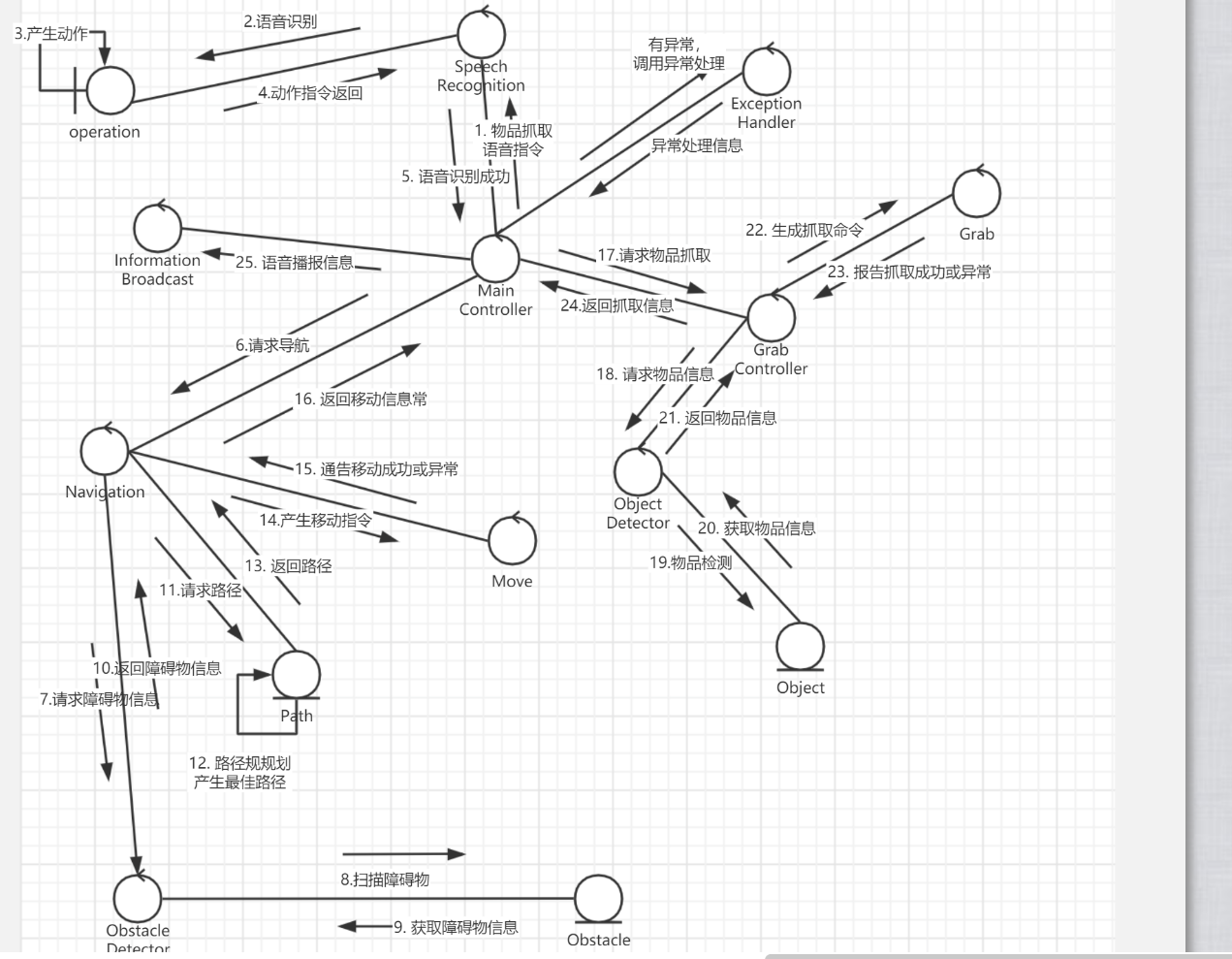


图4-2-2 类间协作图

上图阐述了整个系统实现物品抓取功能时，各类之间的协作关系。首先，MainController类得到用户发出的语音指令和目标地点信息后，将其发给SpeechRecognition类，SpeechRecognition类通过语音识别，生成一系列Operation操作指令并返回。MainController类收到语音识别成功的消息并得到指令后，调用Navigation类，发送目标位置信息，控制机器人向目标地点移动。Navigation类向Path类请求路径，根据目标地点进行路径规划，产生路径并将其返回给Navigation类。Navigation类得到路径信息后，调用Move类，根据路径信息将机器人移动至指定位置。在机器人移动的同时，调用ObstacleDetector类，对障碍物进行检测，如果检测到障碍物，则进行相应的避障处理。当机器人到了指定位置后，返回移动信息给MainController类。接着，MainController类再根据指令向GrabController请求物品抓取。GrabController类收到请求后，调用ObjectDetector类检测物品，获取物品信息并实例化为Object对象，再调用Grab类的grab方法，抓取物品。抓取成功后，grab方法会返回抓取信息，GrabController再将这些信息发送给MainController。最后，由MainController调用InformationBroadcast类，将信息播报出来告诉用户抓取成功还是失败。

## 硬件体系结构

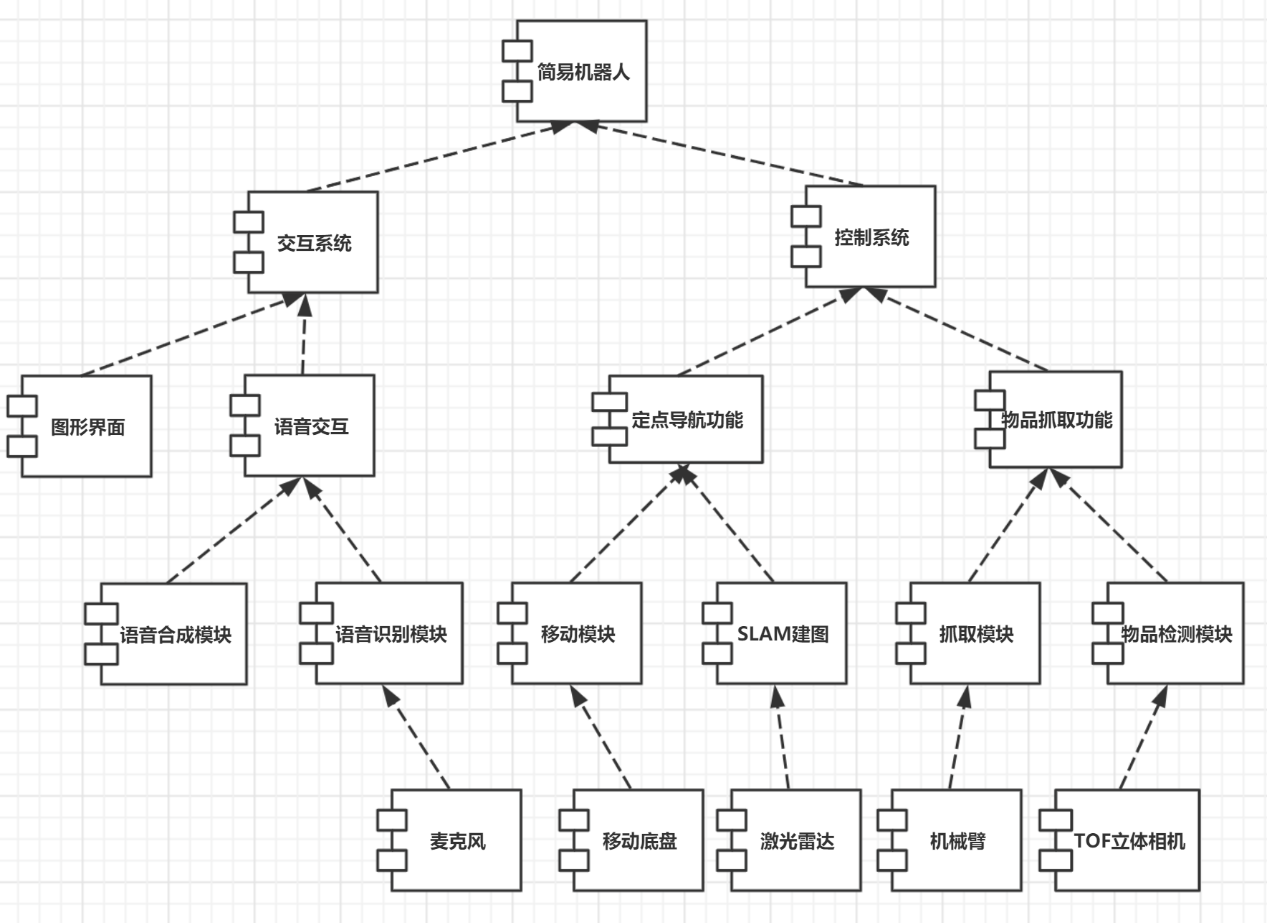


图4-3-1 硬件构件图

系统主要使用的硬件构件有：麦克风、移动底盘、激光雷达、机械臂、立体相机等，其中，麦克风用于语音识别模块，语音识别模块与语音合成模块共同实现了语音交互功能，在加上图形界面，则构成了交互系统。移动底盘用于移动模块，激光雷达用于SLAM建图，SLAM建图和移动模块共同实现定点导航功能，而物品抓取功能由使用机械臂的抓取模块以及使用立体相机的物品检测模块构成，物品抓取功能以及定点导航功能都由控制系统所控制。控制系统和交互系统一同构成了本项目的简易机器人。

## 技术体系结构

本项目设计的ROS机器人会用到以下几个主要技术：IMU姿态传感、、SLAM环境建图、物品识别、语音识别和交互技术。

表4-4-1 技术介绍表

|  |  |
| --- | --- |
| 技术名称 | 简要介绍 |
| IMU姿态传感技术 | 通过ROS机器人的IMU单元实时获取机器人的滚转、倾斜和朝向信息。 |
| SLAM环境建图 | 通过激光雷达，实时扫描机器人周围的障碍物分布情况，借助HectorSLAM和Gmapping算法，创建环境地图。 |
| 物品识别 | 通过立体相机，获得每个物品的外形、尺寸和三维空间坐标 |
| 语音识别和合成技术 | 使用ROS自带的科大讯飞语音识别和语音合成引擎。 |

技术构件图如图所示：

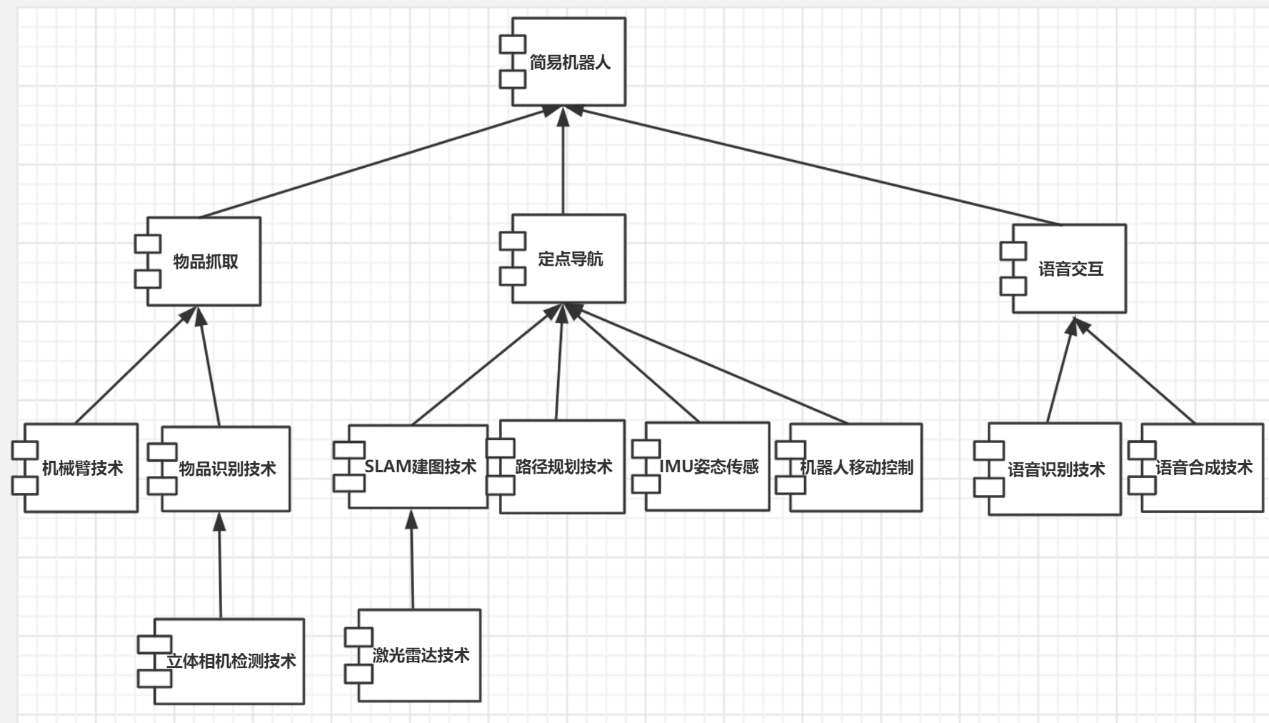


图4-4-1 技术构件图

简易机器人具有三大功能：物品抓取、定点导航和语音交互，要实现这三个功能，需要多种技术的支持。物品抓取功能需要机械臂控制技术、物品识别技术，而物品识别技术是基于立体相机检测来实现的；定点导航功能需要SLAM建图技术、路径规划技术、IMU姿态传感和机器人移动控制技术，其中SLAM建图技术又基于激光雷达的检测；语音交互功能则需要语音识别和合成技术。

# 接口设计

首先先声明一点：对于管理员，所有的接口都是外部接口，必要的时候需要对任一接口可以进行访问并调试

系统用户界面

外部接口：语音输入，用户注册/登录，意见反馈输入栏，语音播报

内部接口：语音识别接口，指令转换接口

系统软硬件

外部接口：指令接收和反馈接口，从下级接收到消息的接口（譬如数据层需要获得机器人各硬件的信息并且进行整合），向上层传递必要信息的接口（譬如将地图建模的信息传递给导航系统）

内部接口：仅供本层使用的功能（譬如硬件层驱动机器正常运动的过程，这并不需要传递给上层）

# 详细设计

## 导航模块

**功能：**

该模块通过输入目的航点，结合路障信息进行路径规划，生成一条导航路径。

**输入：**

目的航点的坐标位置。

**输出：**

路径信息。（由一系列航点组成）

**类图：**

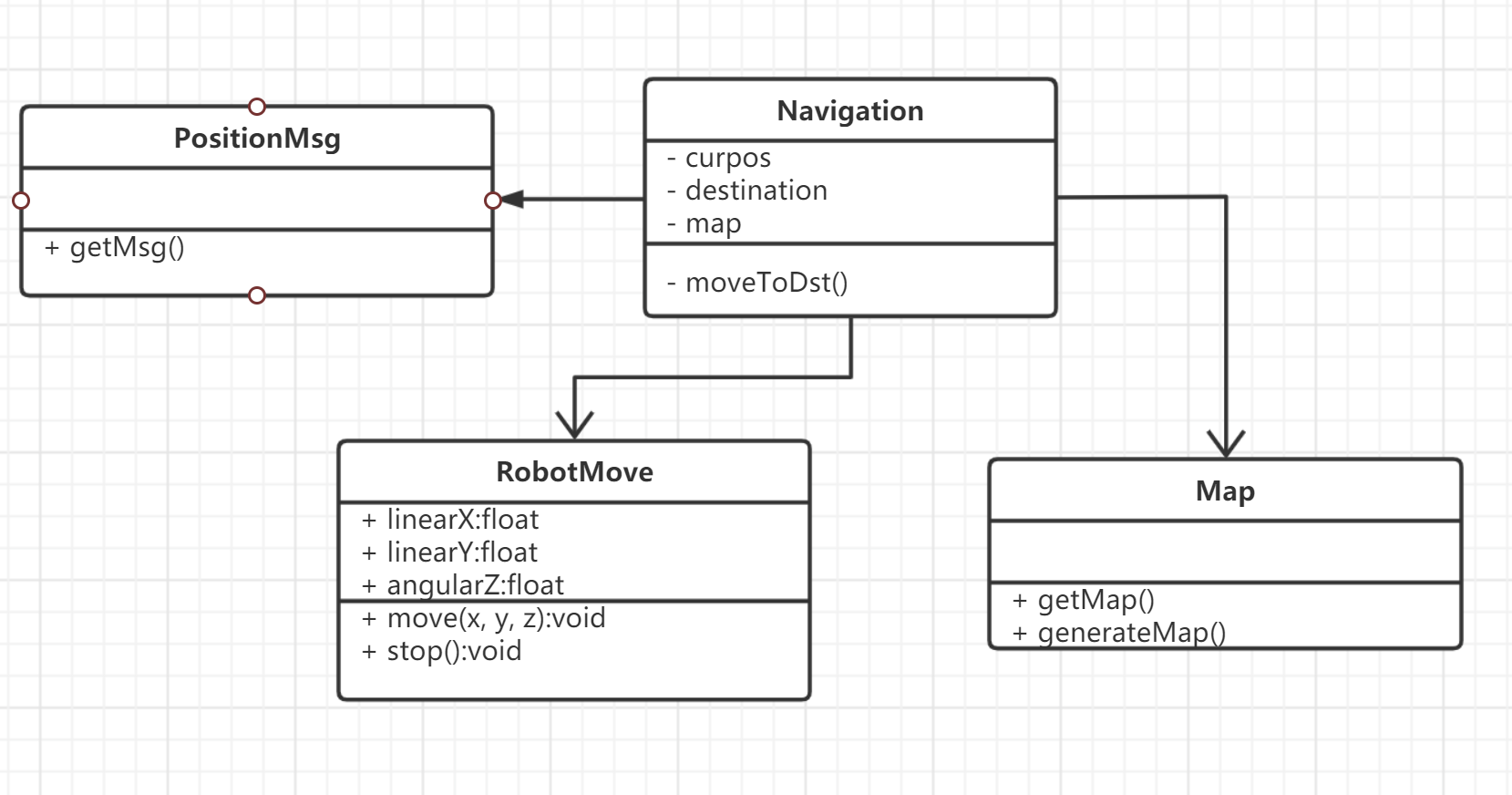


图6-1-1 导航类图

Navigation类为该模块的总控类，通过调用Map类的getMap()方法获得SLAM建图模块建立的地图信息，再根据输入的目标航点以及通过PositionMsg类的getMsg()方法得到的amcl自定位等相关信息，生成一条可到达目的地的路径，同时通过控制运动模块的RobotMove类，控制机器人的移动。从而实现机器人的导航。

## 机器人运动模块

**功能：**

该模块负责机器人的移动，实现机器人的基本运动功能，包括机器人的平移和旋转。在该模块中，通过给定x，y轴的线速度和z轴角速度，实现对机器人的运动控制。

**输入：**

x,y轴方向的线速度（单位：米/秒），以及关于z轴旋转的角速度（单位：度/秒）。

**输出：**

返回成功信息或异常

**类图：**

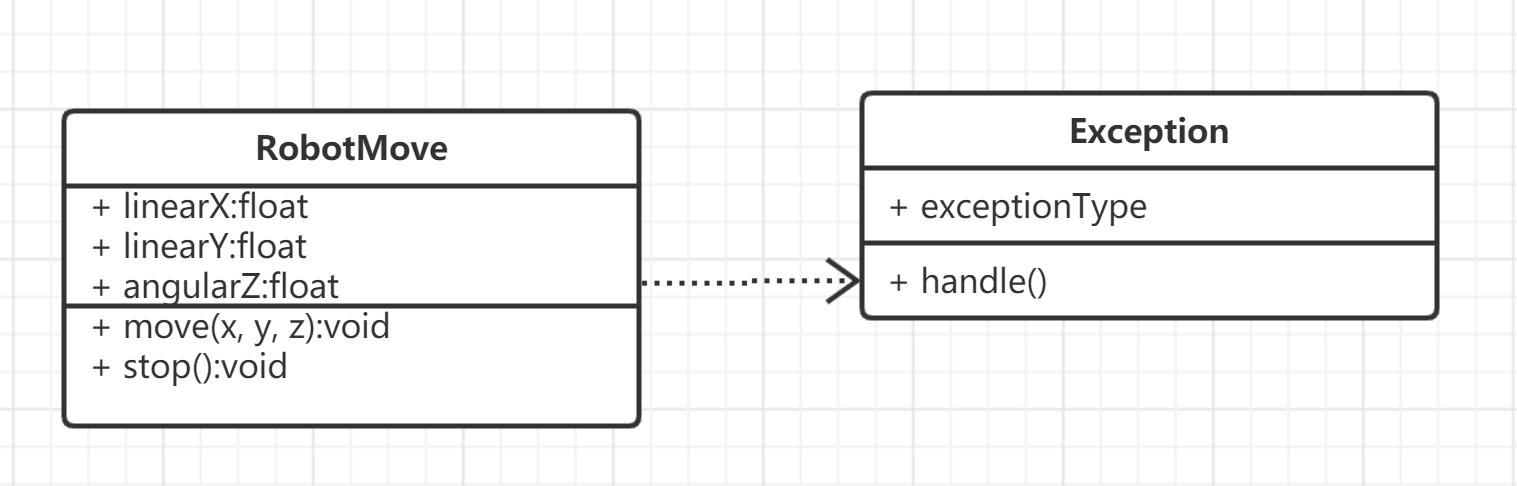


图6-2-1 机器人移动类图

## 建图模块

**功能：**

该模块实现对场景地图的构建。机器人在场景中移动一周，在移动过程中通过对场景进行检测，通过amcl自定位实时获取自身位置，实现对场景的地图建模。

**输出：**

生成的地图文件(map.pgm与map.yaml)

**类图：**

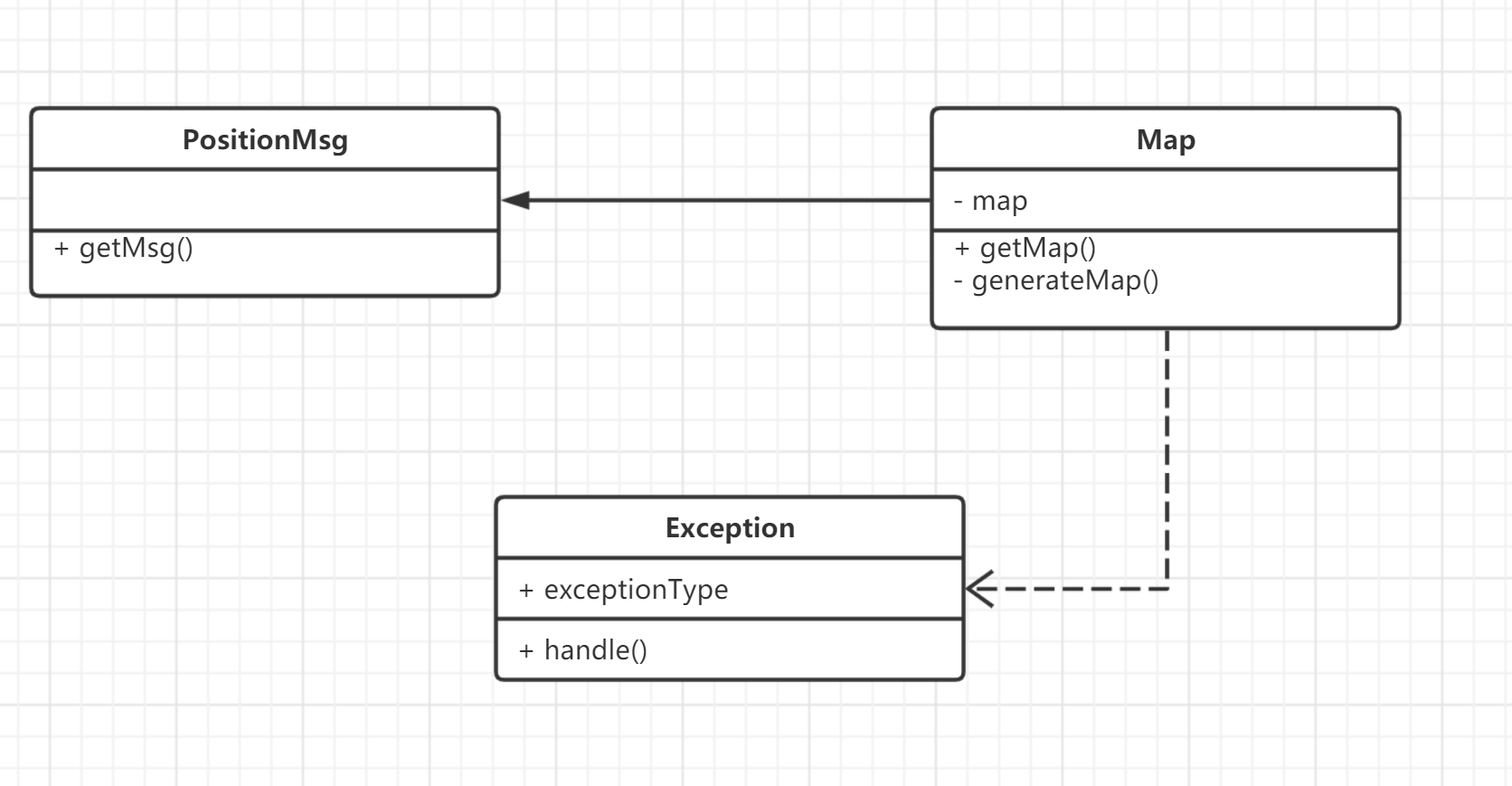


图6-3-1 建图模块类图

该模块主要的类为Map类，Map类通过generateMap()方法生成地图文件(map.pgm和map.yaml)，其他类通过调用getMap()方法获得生成的地图文件。 PositionMsg类通过get\_msg()方法获取从硬件得到的信息，如当前位置、方向等，帮助对地图进行建模。当建模过程中出现异常时，还要进行异常处理。

## 物品检测与识别

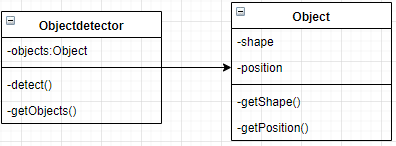


图6-4-1 目标物体识别与确定类图

目标物体的识别与确定主要是靠在机器人头部的Kinect2视觉传感器，利用传感器锁定目标物体，进行图像采集，并确定相对机器人而言，目标物体的位置以及机器人与目标物体间的距离。

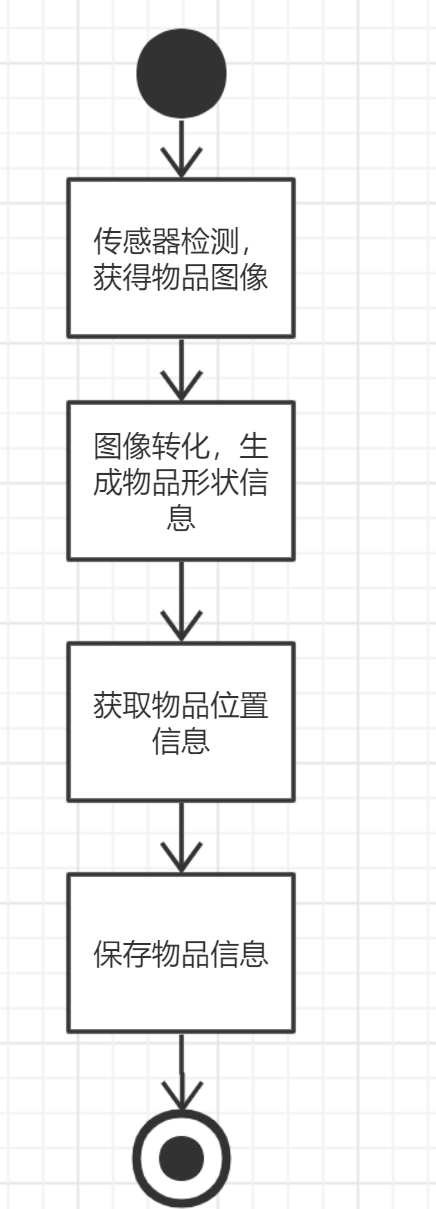


图6-4-2 物品检测流程图

根据机器人视觉传感器给机器人提供的目标物体的信息来确定目标物体的类型。如若未能识别到目标物体，则返回未识别到可抓取物体异常。

计算出目标物体相对机器人的位置信息，并返回，以帮助机器人调整抓取位置。

## 物品抓取

**功能：**

该模块通过物品识别模块获得物品的形状、位置等信息，到达物品地点，通过对机械臂进行操纵，实现对物品的抓取。

**输入：**

物品信息（位置、形状等）

**类图：**

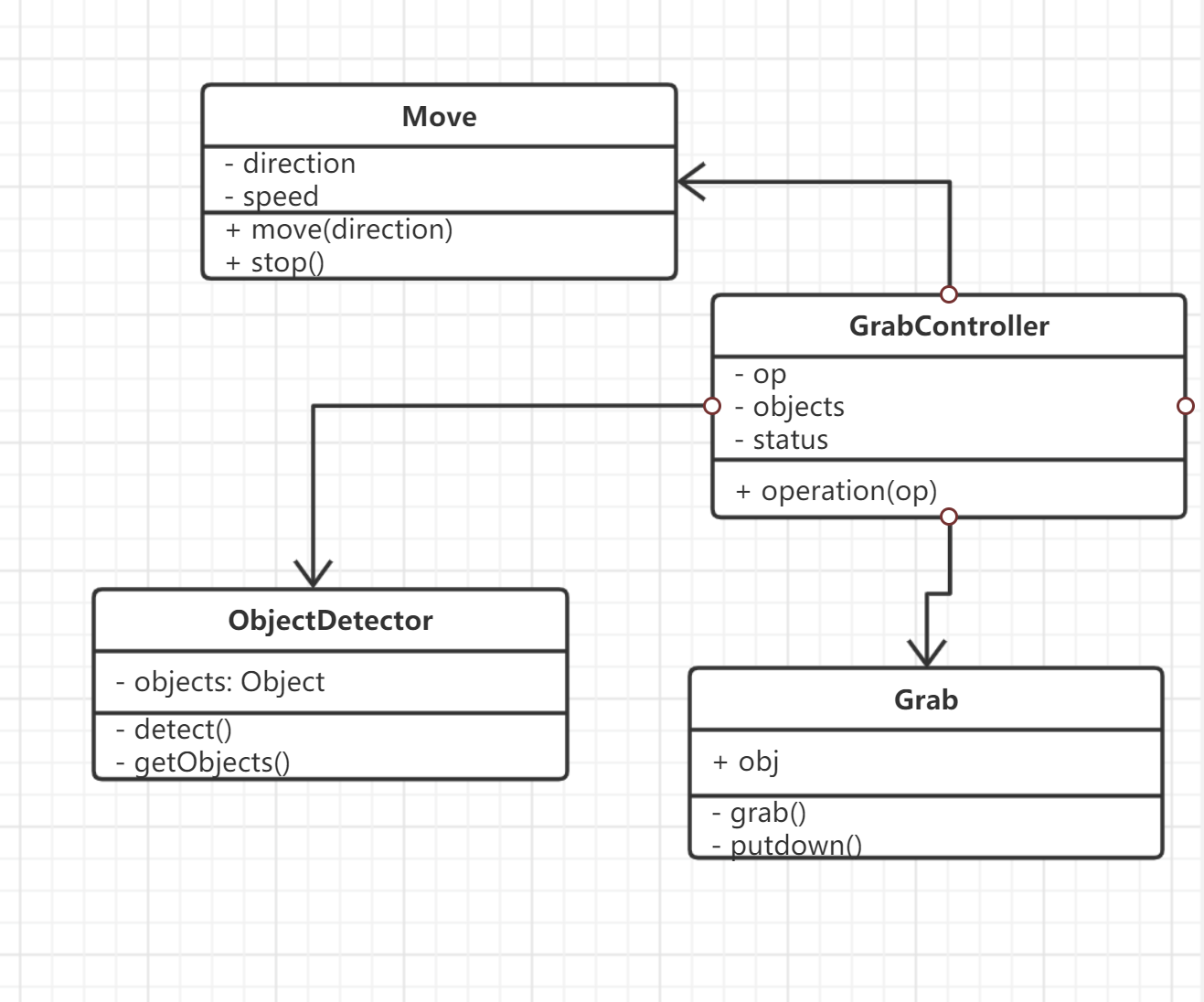
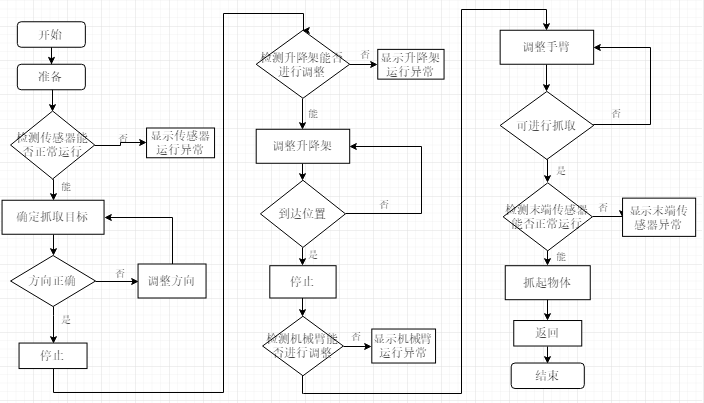


图6-5-1 物品抓取模块类图

该模块的主控类为GrabController, 当获得ObjectDetector类检测到的物品信息后，GrabController先调用运动模块中的Move类，到达物品位置，再调用抓取类Grab，对物品进行抓取。Grab类通过对机械臂进行控制，实现物品的抓取功能。

**流程图：**

 图6-5-2 机器人目标抓取流程图

首先，先对目标物体进行识别与确定，然后根据确定的目标物体进行方向以及高度的调整，其中高度的调整主要就是指对机械臂的升降架进行调整，而后对应目标物体大小对机械臂的开合度进行调整，到此就完成了抓取的准备工作。接下来就是抓起目标物体，此处应对机械臂末端的物理压力传感器进行检测，确保抓起物体同时不会毁坏目标物体。确定无误后即可完成目标物体的抓取。

## 语音识别模块

**功能：**

该通过对用户语音指令进行识别，提取语音中的关键词，从而生成相应控制指令。

**输入：**

用户语音指令。

**类图：**

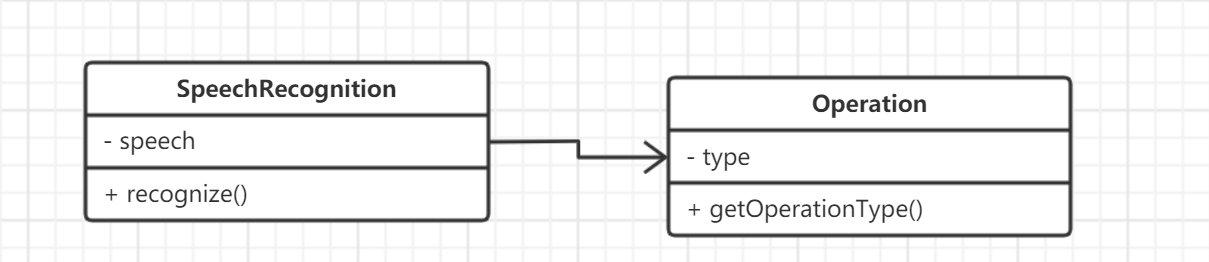


图6-6-1 语音模块类图

在语音模块中，SpeechRecognition类通过读取用户语音指令，提取指令中的关键词，再与事先设定的关键词进行匹配，从而生成相应的操作指令Operation。

# 运行与开发环境

## 运行环境

硬件：

启智ROS机器人

软件：

Ubuntu16.04版本Kinetic

## 软件环境

硬件：

至少三台符合以下配置要求的计算机：

内存4GB以上

硬盘40GB以上

启智ROS机器人一台

软件：

Linux系统或Windows系统下装有Linux的虚拟机，Kinetic开发环境，Java，Python，C++开发环境，Android Studio，RoboWare Studio等IDE。

# 需求可追踪性说明

## 功能需求

表8-1-1 功能需求追踪对应表

|  |  |
| --- | --- |
| 功能需求 | 所在体系结构层次 |
| 地图建模 | 功能层 |
| 避障移动 | 功能层 |
| 检测抓取 | 功能层 |
| 用户功能 | 用户层 |
| 语音识别 | 传输层 |
| 语音播报 | 传输层 |

### 地图建模

管理员在用户层发送地图建模指令。传输层接收管理员信息，识别指令后，调用功能层的地图建模功能。功能层，地图建模功能，从数据层获取激光雷达数据，完成SLAM建模。之后，将地图数据存储至数据库。传输层调用数据库访问功能，获取地图数据，并将其返回至用户层的app界面。app界面显示地图信息。

### 避障移动

用户在app地图界面选择目标点和朝向，传输层接收目标点信息，等待用户发出语音指令。在接收到正确语音指令后，调用功能层的导航模块，机器人进行向目标点的避障移动。功能层具体执行流程可参见详细设计部分。在到达目标点后，返回到达信息至传输层。传输层语音播报到达信息，通知用户。

### 检测抓取

用户发出语音抓取指令，传输层检测到相应指令后，分析指令信息，获得抓取目标物信息，调用功能层抓取模块，机器人进行检测抓取。功能层具体执行流程可参见详细设计部分。在抓取目标物后，返回抓取成功信息至传输层。传输层播报抓取成功信息，通知用户，等待下一步指令。放置流程大致相似，只是机器人执行放置方法。

### 用户功能

用户功能中，登录/注册，启动/关闭系统，地图标点，以及提交bug信息，都在用户层的app里完成。完成相应功能后，app后台发送相应信息至传输层，进而调用相关功能，实现用户所需要的功能。

### 语音识别/播报

传输层完成用户的语音识别/播报功能。它直接识别用户的语音，并将其转换为相应的指令，之后调用功能层的相关模块，完成功能的实现。在完成功能或出现异常情况时，播报相应的语音提示。

## 非功能需求

### 机器人避障效率

这点主要由功能层的避障移动模块决定。具体流程参考详细设计。其中，主要影响因素是路径规划类中的路径规划算法，以及，检测到障碍物后，绕路的避障算法。

### 机器人语音响应能力

这点主要由传输层的语音识别模块决定。一方面，受机器人自身麦克风的收音能力影响；另一方面，受语音识别模块里所调用的科大讯飞包的识别能力的影响。

### 机器人抓取能力

这点主要有功能的检测抓取模块决定。具体流程参见详细设计。该功能主要受两方面因素影响。一是，基于Kinect2视觉传感器获取的信息，检测算法的识别精确度；二是，机器人机械臂的抓取算法的效果。

### 机器人运动特性

这点主要由功能层的机器人运动模块的基础设定，以及机器人自身的电机功率决定。在功能层中，我们可以初设机器人的运动最大速度。

### 机器人的有效负载能力

这点主要由机器人机械臂的物理和机械特性，以及功能层的抓取模块选择的抓取姿势决定。

### 系统健壮性与可用性

这点主要通过各个层次建立相应的异常情况处理，并逐层次上报反馈。各层在发生异常情况时，可以停止当前流程，然后在本层处理或向上层抛出异常。在接收到下层的异常时，同样可以本层处理或继续上报。最终到传输层，可以发出语音播报指令，来提示用户异常情况的发生。