**【简易机器人】**

**软件设计说明书**

**【SDD201】**

**【V1.0】**

分工说明

|  |  |  |
| --- | --- | --- |
| 小组名称 |  | |
| 学号 | 姓名 | 本文档中主要承担的工作内容 |
| 16061094 | 李贞子 | 体系结构、详细设计的导航部分，主要需求书写，全文修订 |
| 16061092 | 张璐 | 数据库设计、文档概要书写，关键字、文档整理 |
| 16231256 | 李天宇 | 接口设计、项目概述书写 |
| 16061108 | 张佳琳 | 体系结构、详细设计的自由避障部分，软件环境书写 |
| 16061117 | 王润泽 | 体系结构、详细设计的识别、抓取部分，需求跟踪书写 |

版本变更历史

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 版本 | 提交日期 | 主要编制人 | 审核人 | 版本说明 |
| V1.0 | 2019.4.21 | 李贞子 | 张璐 | 设计文档的第一个版本 |
| V1.1 | 2019.6.9 | 李贞子 | 王润泽 | 设计文档的最终版本 |
| V1.2 | 2019.6.9 | 李天宇 | 李天宇 | 修改接口说明 |
| V2.0 | 2019.6.9 | 王润泽 | 王润泽 | 修改避障和导航部分 |
|  |  |  |  |  |
|  |  |  |  |  |

**目录**

[1. 范围 1](#_Toc6782444)

[1.1 项目概述 1](#_Toc6782445)

[1.1.1 开发背景 1](#_Toc6782446)

[1.1.2 功能与需求 1](#_Toc6782447)

[1.1.3 应用场景 2](#_Toc6782448)

[1.2 文档概述 2](#_Toc6782449)

[1.3 术语和缩略词 3](#_Toc6782450)

[1.4 引用文档 4](#_Toc6782451)

[2. 需求概述 5](#_Toc6782452)

[2.1 功能说明 5](#_Toc6782453)

[2.1.1 系统用例图 5](#_Toc6782454)

[2.1.2 简要说明 5](#_Toc6782455)

[2.2 系统参与者 6](#_Toc6782456)

[2.3 用例模型 6](#_Toc6782457)

[2.3.1 安装控件 6](#_Toc6782458)

[2.3.2 启动/关闭机器人 7](#_Toc6782459)

[2.3.3 选择机器人任务模式 8](#_Toc6782460)

[2.3.4 控制机器人自由避障行走 8](#_Toc6782461)

[2.3.5 建立地图 9](#_Toc6782462)

[2.3.6 控制机器人规划路径 10](#_Toc6782463)

[2.3.7 控制机器人识别、抓取特定物体 11](#_Toc6782464)

[2.3.8 遇到错误条件 12](#_Toc6782465)

[2.3.9 调试/重新配置系统相关特性并维护硬件 12](#_Toc6782466)

[3. 体系结构设计 13](#_Toc6782467)

[3.1 总体结构 13](#_Toc6782468)

[3.1.1 构件图 13](#_Toc6782469)

[3.1.2 类图 14](#_Toc6782470)

[3.1.3 部署视图 15](#_Toc6782471)

[3.2 关键问题及解决方案 15](#_Toc6782472)

[3.2.1 自由行走时的避障问题 15](#_Toc6782473)

[3.2.2 机器人测位 17](#_Toc6782474)

[3.2.3 障碍物识别 17](#_Toc6782475)

[3.2.4 局部路径规划 17](#_Toc6782476)

[3.2.5 抓取参数调节 17](#_Toc6782477)

[4. 接口设计 18](#_Toc6782478)

[4.1 系统用户界面 18](#_Toc6782479)

[4.1.1 关机 18](#_Toc6782480)

[4.1.2 点击“随机行走模式” 19](#_Toc6782481)

[4.1.3 点击“自动寻路模式” 19](#_Toc6782482)

[4.1.4 点击“自动抓取模式” 19](#_Toc6782483)

[4.2 系统软硬件外部接口 19](#_Toc6782484)

[4.2.1 激光雷达 19](#_Toc6782485)

[4.2.2 kinetic相机 20](#_Toc6782486)

[4.3 系统软硬件内部接口 20](#_Toc6782487)

[4.3.1 行走部分 20](#_Toc6782488)

[4.3.2 地图构建部分 20](#_Toc6782489)

[4.3.3 抓取部分 20](#_Toc6782490)

[4.3.4 功能使用部分 21](#_Toc6782491)

[5. 数据库设计 21](#_Toc6782492)

[5.1 ER图 21](#_Toc6782493)

[5.2 数据库关系模式 21](#_Toc6782494)

[5.2.1 数据表及数据项介绍 22](#_Toc6782495)

[5.2.2 数据表之间关系 24](#_Toc6782496)

[5.3 关系模式的规范化 24](#_Toc6782497)

[5.3.1 用户使用关系 25](#_Toc6782498)

[5.3.2 功能实现关系 25](#_Toc6782499)

[5.3.3 测距关系 25](#_Toc6782500)

[5.3.4 行走控制关系 25](#_Toc6782501)

[5.3.5 物体识别关系 25](#_Toc6782502)

[5.3.6 物体抓取关系 25](#_Toc6782503)

[6. 详细设计 26](#_Toc6782504)

[6.1 运动模块 26](#_Toc6782505)

[6.2 避障模块 27](#_Toc6782506)

[输入：运行时长 27](#_Toc6782507)

[6.3 构建地图模块 28](#_Toc6782508)

[6.4 路径规划模块 29](#_Toc6782509)

[6.5 目标识别模块 30](#_Toc6782510)

[6.6 抓取模块 31](#_Toc6782511)

[7. 运行与开发环境 32](#_Toc6782512)

[7.1 运行环境 32](#_Toc6782513)

[7.2 软件环境 33](#_Toc6782514)

[8. 需求可追踪性说明 33](#_Toc6782515)

[8.1 功能需求 33](#_Toc6782516)

[8.2 非功能需求 33](#_Toc6782517)

# 范围

## 项目概述

### 开发背景

正如我们所知，机器人产生的主要背景是人力资源的珍贵。人们开始使用机器人代替人工来完成一些相应的工作。

根据目前的主流观点，目前的机器人主要分成三代：第一代机器人，即再现型机器人。这类机器人主要表现为将预存指令反复操作，举例来说就是工厂流水线的装配机器人等。

第二代机器人，带有感觉的机器人。这类机器人能够识别外界的某种信号，比如声信号、力信号、光信号等。

第三代机器人，即智能机器人，也是目前我们所设想的最高级的阶段。这类机器人拥有近似人类或者智能的思维。人们不需要教它们具体怎么做，只需要告诉它们任务目标，它们就能很好地完成自己的任务。

目前国内外正经历人工智能（AI）的热潮，这也是智慧机器人的核心。目前在传统机器人方面，我国相较于其他国家略为落后，但也正在奋起直追。全世界范围内均对各种机器人有大量需求。这不仅是因为机器人相对于人类行为更稳定的缘故，更是有一些特种任务、具有风险的任务更加适合机器人来完成。

本项目主要目的是开发一款能够自动控制的机器人。该机器人具有自动避障并随机行走、根据选定目的地进行路径规划与执行、目标物体识别并抓取的功能。根据这些功能我们把这个机器人归为第二代机器人。

### 功能与需求

项目中主要功能包括三大部分：自动避障与随机行走、路径规划与执行、目标物体识别和抓取。自由避障要求物体对静态、动态障碍物都可以躲避，运行流畅，按照设定速度和时间运行。导航功能则要求机器人可以根据建立好的地图，按照用户选择的航点走到指定地点。抓取物体时物品通常放在桌面或者货架上，利用YOLO算法对物体进行识别。以上三个功能呈递进关系上升，每一功能为下一功能打下基础。

### 应用场景

本组开发的抓取机器人将在具体生活中有所应用。例如家用扫地机器人的实例中，自动避障、路径规划将承担主要责任。另外有仓库管理机器人的实例，可以体现出物品识别抓取的作用。这些场景中能够体现出机器人减少人工成本从而降低总成本的优势。

另外根据机器人的特性，完成的机器人在一些特种场景，例如火场任务、排雷任务等都会大有建树。

## 文档概述

本文档为设计说明文档，主要进行项目的具体设计分析。其中包括一下一些部分：

①文档总述与项目概况：主要描述整个项目的概况与本文档的概况。

②需求概述：根据用户需求、业务需求，对本项目功能部分进行细化、分解，形成功能需求的描述。功能需求将细化到小模块的形式，方便程序的构造和实现。

③体系结构设计：主要采用系统体系结构图，UML类图等给出系统的总体结构，并对途中发内容进行说明，并分析系统总体结构中所设计的关键问题及解决方案。

④接口设计：接口是类，子系统或构件提供的操作的集合，通过分析得到系统用户界面、系统的软硬件外部接口和内部接口的设计。

⑤数据库设计：主要通过数据表及其之间的关系，以及对数据项的分析，关系数据模式的规范化，得到最终的数据库设计。

⑥详细设计：对于系统的每个关键模块，逐个给出每个模块的类图及活动图，得到系统的详细设计。

⑦运行与开发环境：主要包含整个机器人系统的运行物理环境（即机器人应当放置在什么样的空间内）、机器人系统的软件运行环境（应当在什么样的操作系统下进行什么样的操作可以运行此机器人、用户界面）和开发机器人使用的开发环境。

⑧需求可追踪性说明：主要通过对系统功能性需求和非功能性需求的分析得到系统的需求可追踪性说明。

## 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。 |
| YOLO | 滑动窗口的目标检测算法 |
| ER图 | Entity-relationship model，实体联系图。 |
| USB-HUB | Universal Serial Bus - Hub 通用串行总线集线器。 |
| 激光雷达 | 思岚（SLAMTEC） RPLIDAR A2。测距范围： 0.15 米-12 米；扫描角度： 360°；  测距分辨率：＜实际距离的 1%；角度分辨率： 0.9°；扫描频率： 10Hz。 |
| 控制面板 | 配置机器人系统的接口。 |
| 底盘控制器 | 启智控制器内部运行了启智 ROS 机器人专用固件，负责 PC 机于机器人之间的数据交互。 |
| Kinect2传感器 | Kinect2 视觉传感器装备在ROS 版的头部，可以在 Rviz 中查看其数据形态。 |
| 机械臂 | 启智 ROS 安装有一个用于抓取桌面上物品的机械臂，该机械臂提供两个控制量：上升高度和手爪的闭合宽度。 |
| 三维点云数据 | Kinect2传感器能够得到的数据，可用于物体的识别。 |
| 主码 | 被数据库设计者选中的，用来在同一实体集中区分不同实体的候选码。 |
| 主属性 | 包含在任一候选关键字中的属性称主属性。 |
| 非主属性 | 不包含在主码中的属性称为非主属性。 |
| 传递函数依赖 | 在关系模式R(U)中，设X，Y，Z是U的不同的属性子集，如果X确定Y、Y确定Z，且有X不包含Y，Y不确定X，（X∪Y）∩Z=空集合，则称Z传递函数依赖于X。传递函数依赖会导致数据冗余和异常。 |
| 部分函数依赖 | 设R(U)是属性集U上的关系模式，X，Y是U的子集。 若对于R(U)的任意一个可能的关系r，r 中不可能存在两个元组在X上的属性值相等，而在Y上的属性值不等，则称“X函数确定Y”或“Y函数依赖于X”，记作X→Y。 |

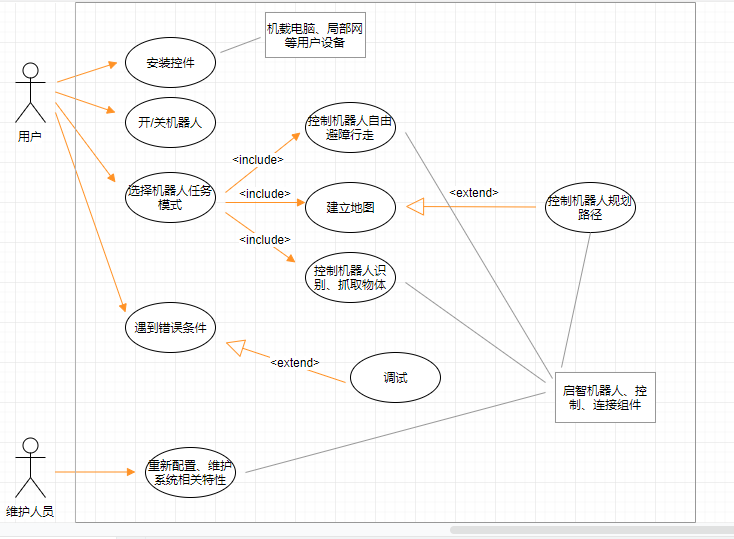
## 引用文档

|  |  |  |  |
| --- | --- | --- | --- |
| 编号 | 标题 | 版本 | 发行日期 |
| 1 | 启智ROS机器人开发手册 | V1.1.0 | -- |
| 2 | 【简易机器人】项目开发计划文档 | V1.0 | -- |
| 3 | 数据库在软件开发中的地位是怎样的？  <https://www.zhihu.com/question/31606584> | -- | -- |
| 4 | 【简易机器人】项目需求规格说明书 | V3.0 | -- |
| 5 | 2019软件工程嵌入式开发软件设计PPT | -- | -- |
| 6 | 2019软件工程嵌入式开发构件极设计PPT | -- | -- |
| 7 | ROS机器人编程 | -- | 2017.12.22 |
| 8 | ROS局部路径规划--DWA学习  <https://www.cnblogs.com/kuangxionghui/p/8484803.html> | -- | -- |
| 9 | 数据库系统概论 | V5.0 | 2014.9 |

# 需求概述

## 功能说明

### 系统用例图



**图1 系统用例图**

### 简要说明

简易机器人系统可以实现三种任务模式。实现模式之前需要用户通过USB接口安装控制组件，包括机载电脑、局部网等。接下来需要用户打开机器人控制开关，机载电脑此时进入系统功能选择界面。用户在该界面可以选择三种任务模式中的任意一种。

第一种任务模式为控制机器人自由避障行走，机器人在用户给定的初速度情况下，避障自由行走，行走时间由用户设定；第二种任务模式为控制机器人规划路径，该任务模式要求用户先利用手柄推动机器人建立环境地图数据，建立好地图之后，用户在UI标定航点，选择目标地点之后，机器人开始规划最短路径并行走；第三中任务模式为控制机器人识别、抓取目标物体，该模式要求用户先放置好几个物体（用例中使用瓶子作为目标物体），然后标定备选物体的航点，机器人识别出目标物体之后并行走至该位置，机器人开始抓取目标物体。三种任务完成任意一种之后，系统重新进入功能选择界面，此时用户可以继续选择任意一种任务模式，或者选择退出简易机器人系统。

当系统出现错误时，用户可以联系维护人员进行调试，维护人员将解决问题并对系统的相关特性重新进行配置和维护。

## 系统参与者

**用户:** 用户负责配置机器人运行所需的环境，包括连接网络以及将机器人与机载电脑通过USB接口相连接；用户使用机器人之前需要打开机器人的电源，不同场景下可选择打开急停按钮；用户通过机载电脑选择机器人的运行模式，控制机器人完成自由避障行走，规划路径，识别、抓取目标物体三种任务中的任意一种；机器人运行结束后用户需要关闭机器人。出现错误的情况下，用户可以联系维护人员对机器人系统重新进行调试和配置。

**系统维护人员:** 系统维护员负责在机器人软件系统出现错误后及时响应用户需求，选择调试软、硬件，修复漏洞。

**启智机器人、控制及连接组件：**机载电脑控制机器人完成移动、转向、抓取等任务，机器人通过雷达等传感器向机载电脑反馈数据，协同合作完成用户指令。

## 用例模型

### 安装控件

**主要参与者：**用户

**目标：**正确设置保证机器人正常运行的部件基础

**前置条件：**用户有相关组件（局部网络、机载电脑、USB接口等）和安装指导（说明手册）

**启动：**用户想要使用机器人

**场景：**

1. 用户连接机载电脑USB和机器人USB-HUB
2. 用户将U盘、手柄等自选设备通过USB与机器人控制器连接

**优先级：**高

**何时可用：**存在于任何一个增量

**使用频率：**高

**次要参与者：**启智机器人、控制部件

### 启动/关闭机器人

**主要参与者：**用户

**目标：**启动：联通机器人电源，使机器人处于准备运行的状态；关闭：切断机器人电源，使机器人停止工作

**前置条件：**必要控件已安装完毕，机器人软硬件状态良好，机器人正常运行时的环境已设置完毕，电池电量充足

**启动：**用户想要使用机器人

**场景：**

1. 使用之前，用户按下开机按钮
2. 使用完毕，用户按下关机按钮

**异常情况：**

机器人电池电量不足——出现点：场景中的1，此时需要更换电量充足的电池，再执行场景中的1

**优先级：**高

**何时可用：**存在于任何一个增量

**使用频率：**高

**次要参与者：**机器人控制面板

### 选择机器人任务模式

**主要参与者：**用户

**目标：**用户选择需要机器人完成的任务模式，用以控制机器人完成指定任务

**前置条件：**完整配置机器人系统所需组件，机器人硬件功能良好，开启机器 人开关，将机器人放置于合适的环境，系统显示功能选择界面

**启动：**用户打开机器人电源开关，连接好机载电脑

**场景：**

1. 系统显示三种任务模式按钮和“退出”按钮

2. 用户根据具体需求点击任一按钮

**可选方案**：

用户点击“退出”按钮——出现点：场景中的2，用户此时退出系统

**优先级：**高

**何时何用：**所有增量之前

**使用频率：**高

**次要参与者：**启智机器人、控制、连接组件

### 控制机器人自由避障行走

**主要参与者：**用户

**目标：**控制机器人按照设定速度在预定环境内自由行走指定时间，并可自动避障

**前置条件：**已进入功能选择界面

**启动：**用户选择避障行走模式

**场景：**

1. 系统显示初始速度和运行时间设定窗口以及“确定”按钮

2. 用户输入指定速度以及运行时间

3. 用户点击“确定”按钮

4. 系统显示“运行开始”提示语句

5. 机器人开始行走

6. 机器人遇到障碍可以避开

7. 机器人在指定时间结束后停止运动

8. 用户重新进入选择模式或者退出

**异常情况**：

用户未设置初始参数——出现点：场景中的2， 用户点击“确认”按钮后，系统使用默认的时间和速度参数

**优先级：**中

**何时何用：**第一个增量

**使用频率：** 中

**次要参与者：**启智机器人、控制、连接组件

### 建立地图

**主要参与者：**用户

**目标：**建立当前环境实时地图，为机器人规划路径提供数据

**前置条件：**已进入功能选择界面

**启动：**用户选择路径规划模式

**场景：**

1. 用户按下“建立地图”按钮

2. 系统弹出RVIZ可视界面

3. 用户利用手柄推机器人建立地图

4. 用户点击“保存地图”按钮

**优先级：**中

**何时何用：**第二个增量

**使用频率：** 中

**次要参与者：**启智机器人、控制、连接组件

### 控制机器人规划路径

**主要参与者：**用户

**目标：**建立当前环境实时地图，为机器人规划路径提供数据

**前置条件：**已进入功能选择界面

**启动：**用户选择路径规划模式

**场景：**

1. 用户点击“标定航点按钮”
2. 系统弹出RIVZ界面显示地图
3. 用户在刚刚建立好的地图上表标定航点
4. 用户点击下拉菜单选择目标航点
5. 用户点击“GO！”机器人行走至目标地点
6. 机器人到达目标地点后停止运动，并发出语音提示

**优先级：**中

**何时何用：**第二个增量

**使用频率：** 中

**次要参与者：**启智机器人、控制、连接组件

### 控制机器人识别、抓取特定物体

**主要参与者：**用户

**目标：**机器人识别用户放置的物体，从起始位置到物体放置位置进行抓取

**前置条件：**进入识别抓取、模式

**启动：**用户选择识别、抓取模式

**场景：**

1. 用户点击“标记航点”按钮

2. 用户带着机器人依次标记航点

3. 标记完航点用户说出”stop following”

4. 机器人停止标记航点

5. 用户启动物体识别节点

6. 机器人识别物体

7. 机器人行走至目标位置

8. 机器人调整机械臂抓取物体

9. 机器人发出语音播报“抓取完毕”

10. 用户重新进入选择模式或者退出

**优先级：**中

**何时何用：**第三个增量

**使用频率：** 中

**次要参与者：**启智机器人、机械臂、控制、连接组件

### 遇到错误条件

**主要参与者：**用户

**目标：**通知系统管理员系统出现问题并尽快解决

**前置条件：**机器人出现异常行为时

**启动：**用户发现机器人在任何状态下出现有悖于程序设定的行为

**场景：**

1. 用户联系系统管理员

2. 用户向系统管理员详细描述机器人出现的异常行为以及系统提示的错误信息

**优先级：**低

**何时可用：**存在于任何一个增量

**使用频率：**低

**次要参与者：**系统管理员，启智机器人、机载电脑

### 调试/重新配置系统相关特性并维护硬件

**主要参与者：**系统管理员

**目标：**纠正机器人出现的异常行为并对系统不完善之处进行维护

**前置条件：**机器人出现异常情况且系统管理员收集到问题

**启动：**系统管理员接到用户反馈后

**场景：**

1. 系统管理员记录并分析用户描述情况，初步推测问题成因
2. 系统管理员实地检查机器人部件是否出现损坏、接触不良、电路过载等硬件问题
3. 系统管理员用管理员权限登陆系统，检查代码是否存在漏洞
4. 系统管理员根据用户描述还原使用场景，观察用户报告的问题是否依然存在，是则返回2,否则结束调试

**优先级：**低

**何时可用：**任何一个增量

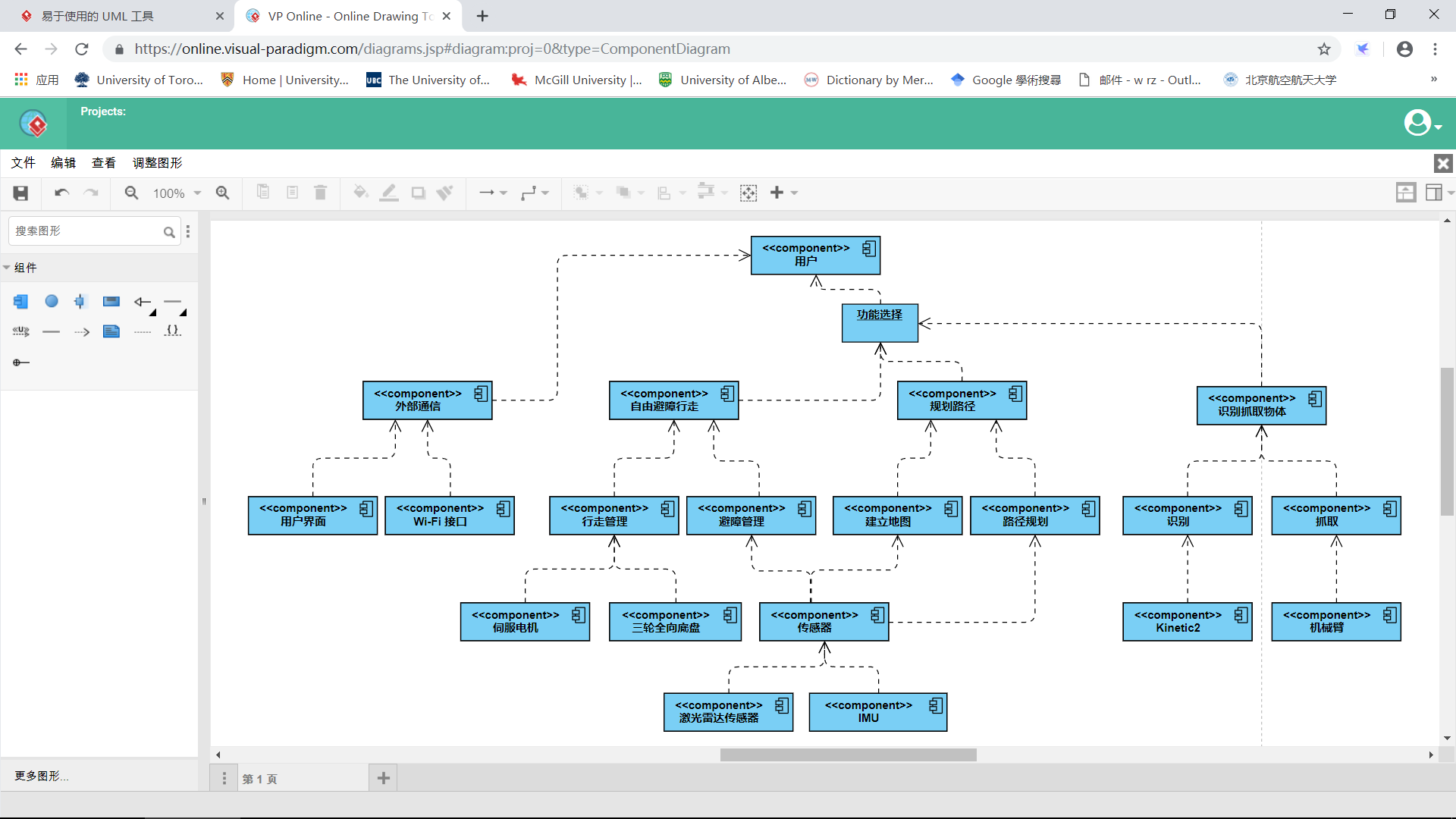
**使用频率：**低

**次要参与者：**用户、启智机器人、机载电脑

# 体系结构设计

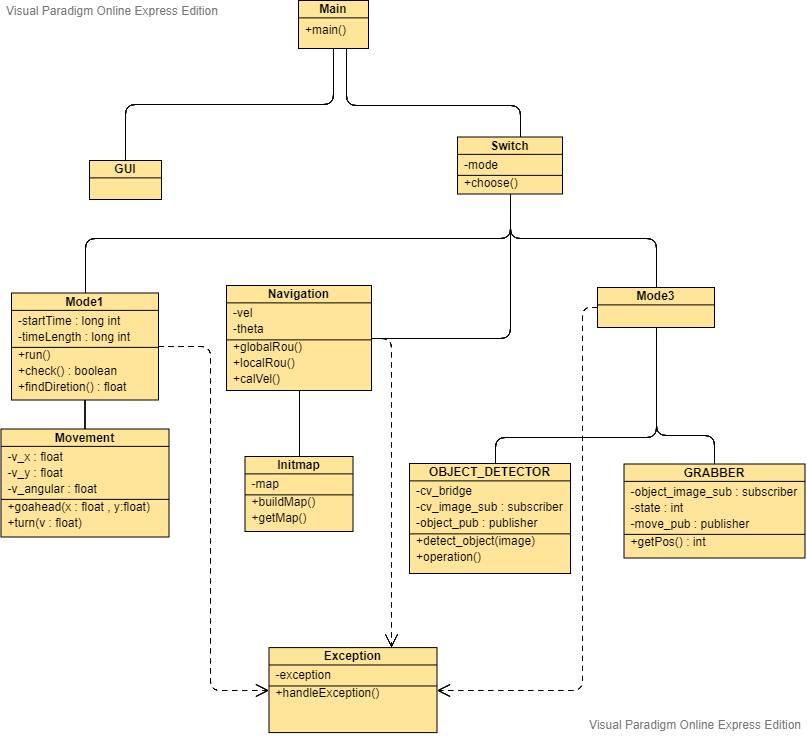
## 总体结构

### 构件图



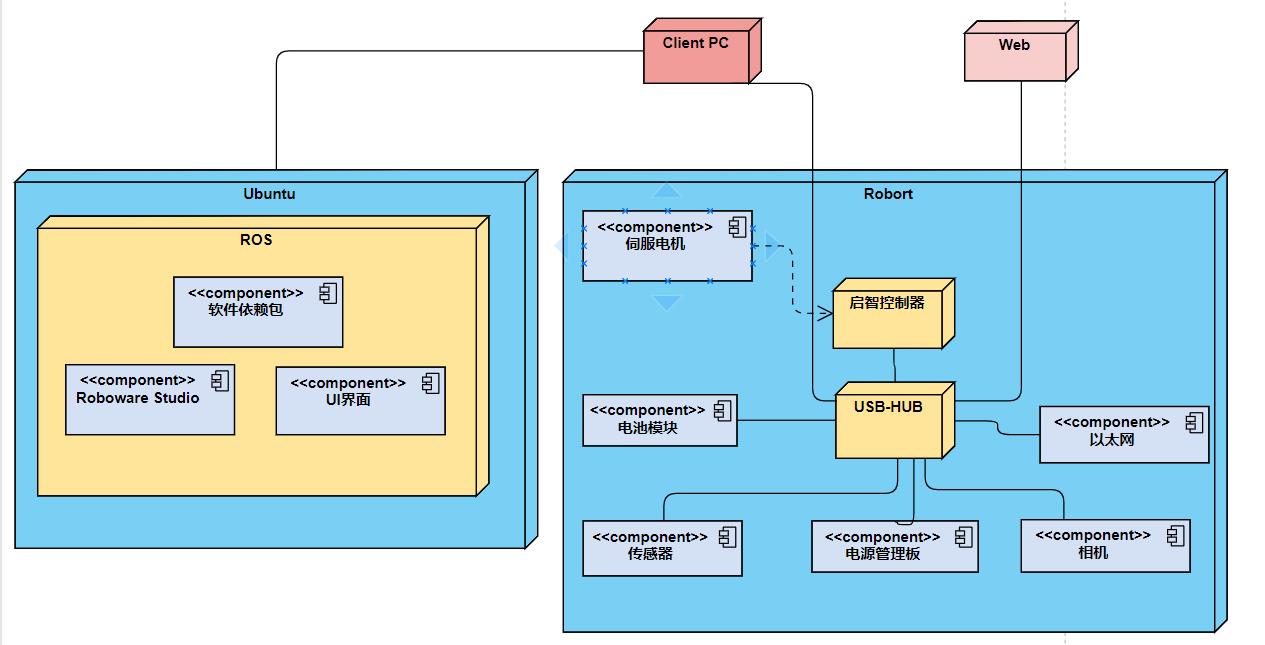
机器人用户通过用户界面进行功能选择，并利用Wi-Fi实现互联网通信功能。功能选择构件可以分为自由避障行走、规划路径和识别抓取物体。其中自由避障功能里机器人行走需要伺服电机构件产生电信号反馈双向姿态信息，也需要三轮全向底盘实现轮子的转动，以支持机器人机械行走。传感器主要分为激光雷达传感器和IMU，用以识别障碍物、提供机器人与障碍物的距离，实现避障、定位等功能，因此传感器构件将被用于功能一和功能二。功能三需要对物体进行识别和抓取，识别物体主要通过kinetic相机，机械臂构件用来抓取目标物体。

### 类图



软件体系结构主要由UML类图说明。Main类是整个系统的入口和总体控制。GUI类实现和用户的交互，帮助用户选择功能模式。Switch类用以接入具体的功能类。Model1是自由避障行走的类，用于实现机器人的避障模式，该类调用Movement类实现机器人的速度、姿态控制。Navogation类用于实现导航和路径规划功能，调用Initmap构建场景的实时地图。Model3用于实现机器人的抓取和识别物体功能，调用OBJECT\_DETECTOR和GRABBER类实现目标物体的检测和机械臂抓取。这些类需要依赖Exception类，用以实现异常信息捕捉和处理。

### 部署视图



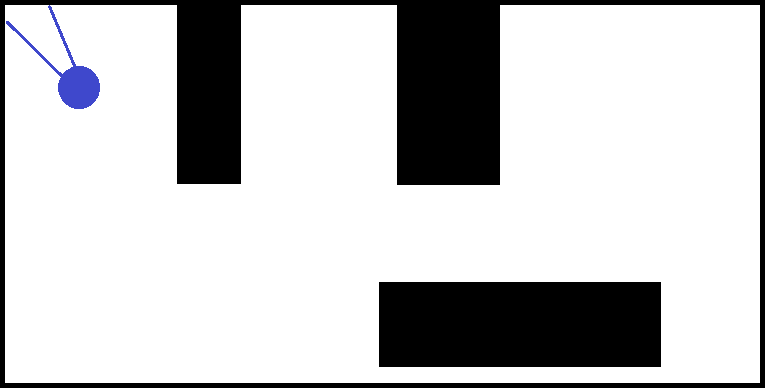
机器人体系主要由软件、PC、Wi-Fi网络和机器人硬件系统部署而成。软件部分架构于Ubuntu系统，ROS系统用来提供机器人运行环境，Roboware studio用来编写机器人系统的代码，UI界面则负责与用户的交互，这三者并列于ROS系统之上。PC端通过USB-HUB与启智机器人连接，扩展PC端USB为多路接口，连接Web和启智控制器，启智控制器控制伺服电机，用以完成机械控制和PC双向电信号交互。USB-HUB还连接了电池模块、传感器、电源管理板和相机。

## 关键问题及解决方案

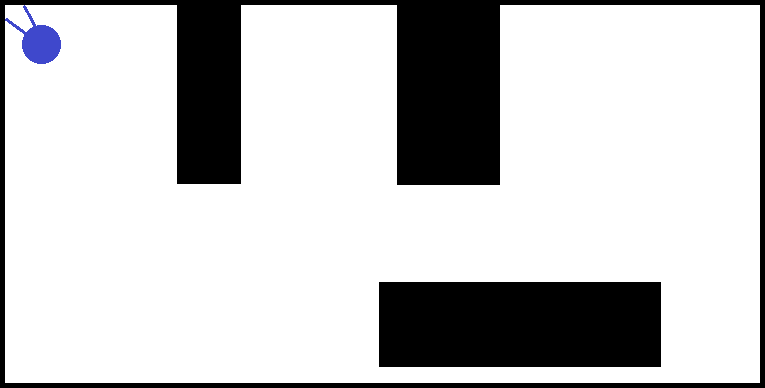
### 避障功能检测障碍物问题

当开启模式一即自由行走时，不断收集激光距离传感器的信息，以得知机器人周围的障碍物情况，获取障碍物信息后需要做出相应的姿态调整。关于障碍物的判定可以设计如下算法：

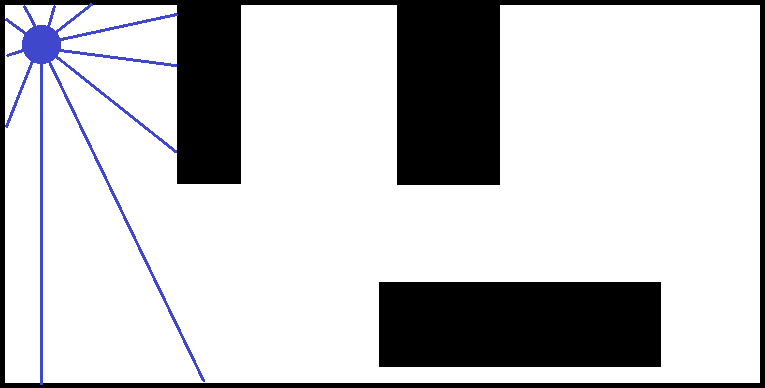
以障碍物前进方向90度夹角为搜索范围，收集该范围内的雷达信息，根据雷达的距离数组获取障碍物距离，然后遍历该90度范围内的距离，计算sin值作为距离机器人的直线距离，计算cos值作为和机器人前进通道内的宽度距离。如果直线距离即sin值<=0.4m，则判定该障碍物会与机器人碰撞，该距离大于机器人以最大速度1m/s行走的最大长度，但不至于过大而限制机器人的活动范围；如果宽度距离即cos值<=0.25m，则认为该障碍物位于机器人前进的通道范围内，0.25m是机器人的底盘半径。综合考虑一个距离的sin值和cos值，如果90度范围内有同时满足sin值<=0.4m，cos值<=0.25m的障碍物点存在，则认为该机器人前进范围内有障碍物，需要转向重新进行寻找。例如：



假设初始方向为图示方向。



向前行进，直到检测到前进方向有障碍物直线距离<=40cm并且宽度距离<=0.25m，位于机器人通道内。



准备转向。360°雷达搜索合适方向。

### 无障碍物通道搜索问题

避障模式中，如果机器人检测到目前通道范围内有障碍物，此时需要对没有障碍物的通道进行搜索，然后发布角速度信息，使得机器人可以继续避障前进。具体的解决方法如下：

根据判断障碍物时收集的雷达信息，在机器人360度范围内依据距离数组搜索无障碍物通道。以10度为一个角度分辨率进行搜索，以这个角度为中心，考察该角度左侧45度和右侧45度角度范围内是否有障碍物，判断障碍物的算法如3.2.1叙述，如果以该角度为中心的90度范围内没有障碍物，那么认为这个角度存在一个无障碍物通道，此时可以计算出该角度和机器人目前的角度之间的差值，可以得到时间*t*（*w*是机器人预设的角速度），*t*则为机器人需要转动的时间，在该时间范围内发布角度信息，使得速度控制模块可以接收到该信息，进而控制机器人行走；如果以该角度为中心的90度范围内有障碍物，则检测下一个角度。如果360度范围内都找不到合适的通道，那么机器人停止运行。

### 机器人测位

就像汽车导航过程中需要GPS定位一样，机器人按照规划的路径行驶过程中需要对自己的位置进行测定。由于室内定位的精确性和可行性不高，因此采用概率定位系统进行解决，概率定位指的是根据机器人车轮旋转量和IMU传感器等获取惯性信息来对机器人室内的位置进行粗略估计。

在这里我们使用amcl包来进行机器人定位，它实现了自适应（或KLD采样）蒙特卡罗定位方法，其使用粒子滤波器来针对已知的地图跟踪机器人的位姿。目前情况下，该节点仅能使用激光扫描数据和激光雷达地图来工作。它可以通过修改代码以扩展到其他传感器数据，amcl接收基于激光的地图，激光扫描和tf变换消息。

### 障碍物识别

为了进行实时避障需要对传感器的信息进行采集，这些传感器要求能够通过ROS发布sensor\_msgs/LaserScan格式的消息，也就是二维雷达信息。ROS内部有大量对激光雷达传感器等设备的驱动包，利用这些功能包可以获得传感器数据。

### 局部路径规划

机器人路径规划的过程中首先利用costmap规划出一条大致的路线，但是在具体的移动过程中需要规划出实时的局部路线，以此达到目的地。为了实现此功能，先将全局路径分割获得局部目标点，并进行坐标转换，再根据相关算法规划出局部路径。

这里利用[base\_local\_planner](http://wiki.ros.org/base_local_planner?distro=melodic)包，使用Trajectory Rollout 和Dynamic Window approaches算法采样机器人速度和角度，针对采样得出行驶路线，再利用一些评价标准（是否撞击障碍物、行驶时间）为路径打分，以此获得最优的局部路径。

### 物体识别问题

物体识别不单单是检测到物体就结束，还需要分辨这个物体是什么。这里就需要用到机器学习的方法，通过训练一个神经网络来让其能够根据kinect2相机传来的图像判断物体的的种类，并且达到较高的准确率。并且在识别时需要注意桌面上不能有其他物体，因为这种模型的判断并不是永远正确，受光线、背景颜色、物体颜色的变化影响较大，所以应该尽可能减少这些因素的干扰。

### 物体抓取调节

在抓取物体时应该要注意三个问题：

（1）抓取前应该设定好机械臂的参数，使得机械臂的升高量近似等于物体的高度。

（2）抓取时物体的三维点云信息会作为抓取的启动标志，所以应该保证机器人的kinect2相机姿态调整正确，距离桌子有一定距离，保证相机的视野良好，否则无法检测到物体。

（3）抓取前机器人需要检测到障碍物才会停下然后进行一系列动作，所以应当选用下方实心的桌子，保证让机器人“知道”已经到达桌子可以开始抓取了。

# 接口设计

## 系统用户界面

用户界面主要用于使机器人和用户进行交互，是部署在机载电脑上的界面。用户界面可以采用按钮的形式呈现，并且有输入框可以和用户进行互动。

### 点击“随机行走模式”

点击后进入新界面，内容如下：

输入时间、速度参数，并点击开始，机器人开始行走；。

### 点击“导航模式”

点击后进入新界面，内容如下：

点击“构建地图”，按照提示构建新地图；

点击“保存地图”，保存刚刚构建的地图；

点击“设置航点”，可以在Rviz界面中新建一些航点；

点击“保存航点”，以保存刚新建的航点；

点击“开始导航”，进入导航准备阶段，等待机器人发出“I am ready”语音后可以开始导航；

点击“航点”下拉菜单选择导航目的地，点击“GO！”开始导航。

### 点击“自动抓取模式”

点击后进入新界面，内容如下：

用户点击根据提示点击“预设目标地点”按钮；

用户点击“开始抓取”按钮；

机器人将行走并抓取目标物体。

## 系统软硬件外部接口

启智ROS机器人可以配备并使用多种传感器，通过传感器可以接收来自外界的信号。包括深度、颜色。我们主要用到以下几种传感器：激光雷达、kinetic像机等。

### 激光雷达

启智机器人提供测距范围0,15米-2米的测距雷达，扫描角度为360度。测距分辨率小于实际距离的1%，角度分辨率为0,.9度，频率为10Hz。装备在底座位置的激光雷达可以根据获取的深度信息分析出周围障碍物的分布。因为能获得较高精度的深度信息，所以我们使用激光雷达构建地图。

### kinetic相机

该相机可以用于构建周围一定范围内的三位模型。该模型可以被用于识别和抓取目标物体。机器人检测到目标物体后，根据其他传感器获取的信息（例如深度、高度、角度等），对机械臂的行为进行决策，以便抓取目标物体。

### 系统软硬件内部接口

系统使用面向对象的设计思路，分为多个类（模块）。不同的类有不同的功能。详细信息见第三部分的“类图”部分。面向对象的设计中，不同的类通过通信或者相互调用的方式进行工作，因此我们需要构建相应的接口，以便不同类的协作。下面将介绍主要的软件内部接口。

### 行走部分

Movement类有goAhead(float,float)和turn(float)接口，实现机器人的行走和转向功能。该类可以在主要功能类中被调用以实现控制机器人。

### 地图构建部分

InitMap类有buildMap()和getMap()两个主要对外接口，以实现构建和获取地图。该类的接口主要被用于第二功能以实现对于附近障碍物的识别和建模，方便功能实现；InitPoint有newPoint和savePoint两个接口，实现航点的构建与存储，在调用时按照顺序调用。

### 抓取部分

有OBJECT\_DETECTOR和GARBBER类，分别有对外接口detect\_object、operation、getPos等接口，实现探测、操作等功能。上述接口主要被第三功能的类调用以实现物品识别、抓取的任务。

### 功能使用部分

Model1是避障行走模块，将会调用goAhead、turn来控制机器人的行走并且周期性调用hasFrontObstacle功能检测前方障碍物情况；

Model2是导航模块，会调用buildMap、getMap、newPoint、savePoint来构建当前地图，并且根据用户的选择使用initWay并且调用goAhead、turn来控制机器人生成导航路径、根据导航路径行走。

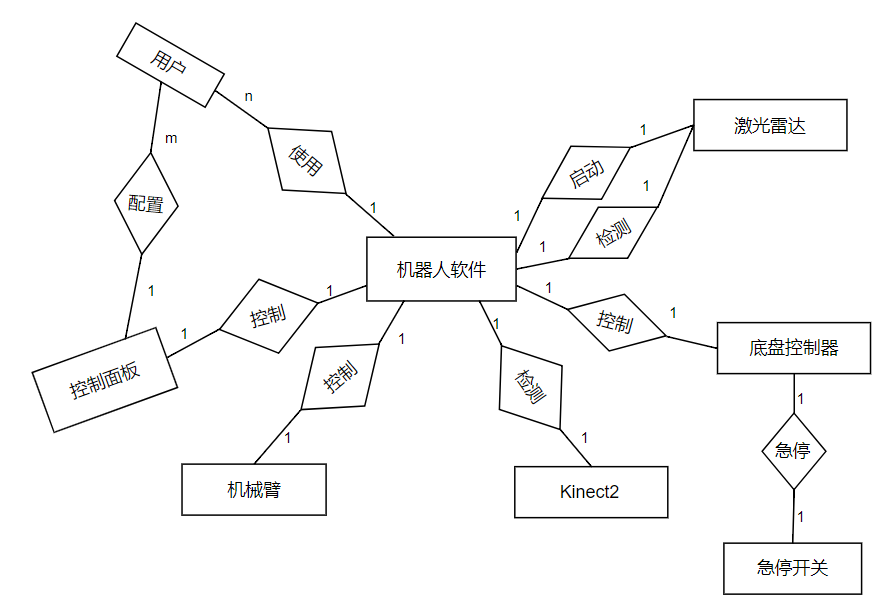
Model3是抓取模块，会调用自己的setPoint来标定预设点位，并且循环调用detect\_object直到检测到目标物体；检测到物体后将会调用goAhead、turn来将机器人运动到预设点位，调用operation来操作机器人抓取目标物体。

# 数据库设计

数据库是软件开发的根本与即使，是绝大多数信息系统的关键组件，针对于我们的软件开发，数据库设计至关重要。

在数据库设计中，首先对逻辑结构进行设计，得到了ER图，之后基于关系数据模式，对物理模型进行设计，创建了数据表结构、数据项以及关系等，并为了得到理想的数据结构，将数据模型规范到第三范式。

## ER图



## 数据库关系模式

采用关系模式，每个关系对应一个表，下面将对表结构进行介绍。

### 数据表及数据项介绍

|  |  |  |  |
| --- | --- | --- | --- |
| 用户使用关系 | | | |
| 用户ID | 用户密码 | 用户权限 | 可选功能 |
| 唯一标识用户的数据项，也是该关系的主码 | 用户可以设定密码，行使自己的权限 | 用户可以有不同的权限 | 可选功能有三个：  1.自由行走，避障  2.路径规划  3.识别抓取物体 |

|  |  |
| --- | --- |
| 功能实现关系 | |
| 功能项 | 使用设备 |
| 功能项是该关系的主码，不同功能项用于实现不同的功能 | 不同的功能项需要使用不同的设备，每个功能项的控制设备和接收设备都是使用设备 |

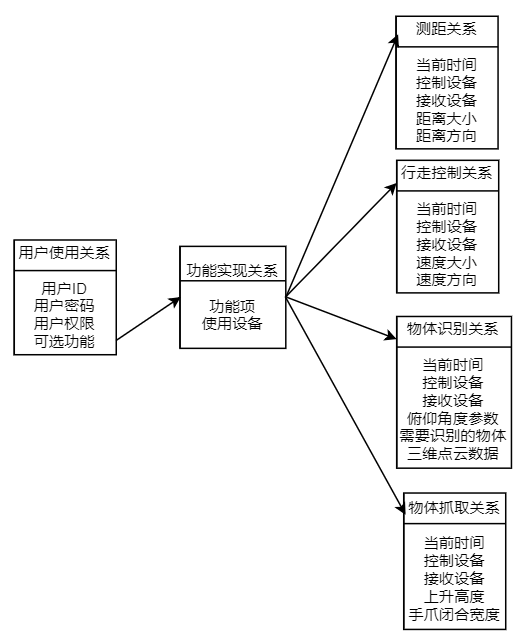
|  |  |  |  |
| --- | --- | --- | --- |
| 行走控制关系 | | | |
| 控制设备 | 接收设备 | 速度大小 | 速度方向 |
| 机器人软件 | 底盘传感器 | 机器人软件通过向地盘传感器传输速度大小数据，控制机器人行走速度 | 机器人软件通过向地盘传感器传输速度方向数据，控制机器人行走方向 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测距关系 | | | | |
| 当前时间 | 控制设备 | 接收设备 | 距离大小 | 距离方向 |
| 唯一标识测距关系数据项的属性，主属性 | 激光雷达传感器 | 机器人软件 | 激光雷达传感器可以测量机器人距离障碍物的距离大小，传输给机器人软件，用于数据的分析处理 | 激光雷达传感器可以测量机器人距离障碍物的距离方向，传输给机器人软件，用于数据的分析处理 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 物体识别关系 | | | | | |
| 当前时间 | 控制设备 | 接收设备 | 俯仰角度参数 | 需要识别的物体 | 三维点云数据 |
| 唯一标识测距关系数据项的属性，主属性 | Kinect2传感器 | 机器人软件 | Kinect2传感器工作需要的一些参数数据 | 需要利用Kinect2传感器识别的物体 | Kinect2传感器得到的三维点云数据，传输到机器人软件用于识别 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 物体抓取关系 | | | | |
| 当前时间 | 控制设备 | 接收设备 | 上升高度 | 手爪闭合宽度 |
| 唯一标识测距关系数据项的属性，主属性 | 机器人软件 | 机械臂 | 机械臂抓取物体需要的上升高度数据 | 机械臂抓取物体需要的手爪闭合宽度数据 |

### 数据表之间关系



## 关系模式的规范化

为了得到理想的数据结构，消除冗余，避免部分函数依赖与传递函数依赖，需要达到第三范式的要求。

下面将针对每个表进行分析，由第三范式的定义，若每一个非主属性既不传递依赖于码，又不部分依赖于码，则该关系达到了第三范式。

### 用户使用关系

针对于用户使用关系，主码是用户ID，用户密码，用户权限和可选功能都是完全依赖于主码，因此用户使用关系达到了第三范式的要求。

### 功能实现关系

针对于功能实现关系，主码是功能项，非主属性只有一个，就是使用设备，而使用设备完全依赖于功能项，因此，非主属性完全依赖于主码，是第三范式。

### 测距关系

针对于测距关系，当前时间是主码，控制设备和接收设备是固定的值，可以删除，删除之后，非主属性完全依赖于当前时间，能够达到第三范式的要求。

### 行走控制关系

针对于行走控制关系，当前时间是主码，控制设备和接收设备是固定的值，可以删除，删除之后，非主属性完全依赖于当前时间，能够达到第三范式的要求。

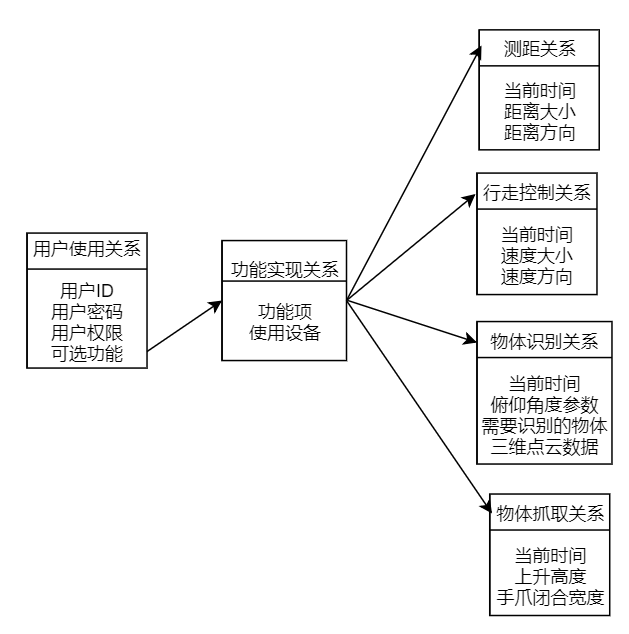
### 物体识别关系

针对于物体识别关系，当前时间是主码，控制设备和接收设备是固定的值，可以删除，删除之后，非主属性完全依赖于当前时间，能够达到第三范式的要求。

### 物体抓取关系

针对于物体抓取关系，当前时间是主码，控制设备和接收设备是固定的值，可以删除，删除之后，非主属性完全依赖于当前时间，能够达到第三范式的要求。

于是可以得到规范化之后的关系模式图：



# 详细设计

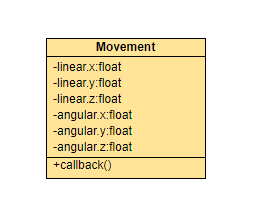
## 运动模块

**输入：**移动速度、旋转速度

**输出：**机器人运动控制消息（以一定比特率输出转化的速度信息，用来控制伺服电机）

**功能：**控制机器人基本的运动

**类图：**



有6个控制运动的运动的消息：linear.x, linear.y, linear.z , angular.x, angular.y angular.z。linear.x, linear.y, linear.z控制线速度移动，分别代表坐标轴方向，其中linear.x为机器人前进方向的速度；angular.x, angular.y, angular.z为角速度，控制机体的旋转。该模块有一订阅者，订阅”/cmd\_vel”话题，如果接收到速度消息，则调用callback函数向底层运动硬件发送转化过的速度消息，机器人可以运动。

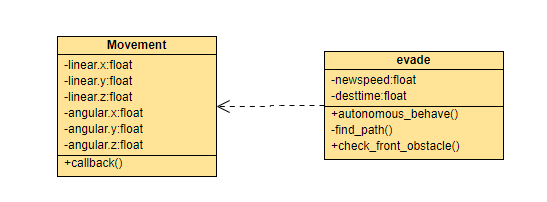
## 避障模块

## 输入：期望的线速度、时长

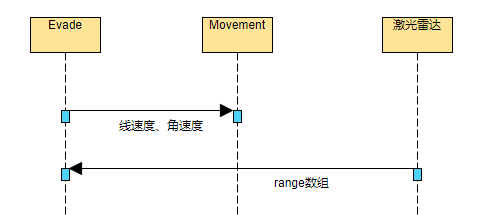
**输出：**机器人下一步的角速度、线速度

**功能：**实现功能一，自由行走及避障

**类图：**



**活动图：**



避障模块接收雷达的距离数组range信息，触发autonomous\_behave()回调函数对雷达信息进行处理，首先调用check\_obstacle()函数判断当前通道内是否有障碍物：如果有障碍物则继续调用find\_path()函数寻找无障碍物通道，如果找到则发布角速度信息和线速度信息，由Movement模块接收，如果没有找到通道则发布角速度、线速度均为0的信息使机器人停止运行；如果没有障碍物则发布当前线速度信息，角速度为0。

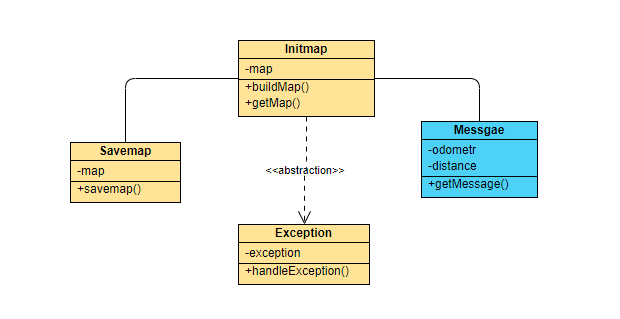
## 构建地图模块

**输入：**测位信息、传感器信息

**输出：**场景地图

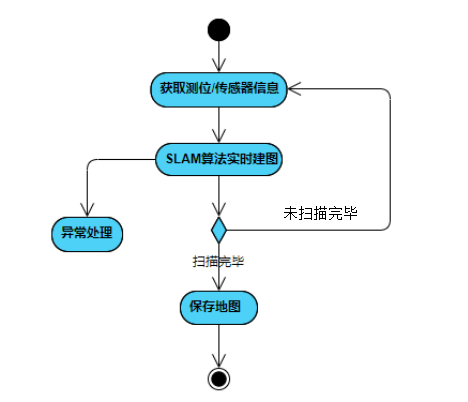
**功能：**用户使用之前，系统人员使用该类对场景进行实时地图构建，并保存地图

**类图：**

****

Initmap类调用bulidMap进行实时地图的构建，该函数使用gampping包根据激光数据（或者深度数据模拟的激光数据）构建地图，getmap函数可以返回构建好的地图。Savemap类将构建好的实时地图以map.pgm和map.yaml格式保存，以备后续使用。Message类提供机器人的测位信息和传感器信息来对机器人进行定位，以实现SLAM建图。如遇到异常情况，则调用Exception类处理。

**活动图：**

****

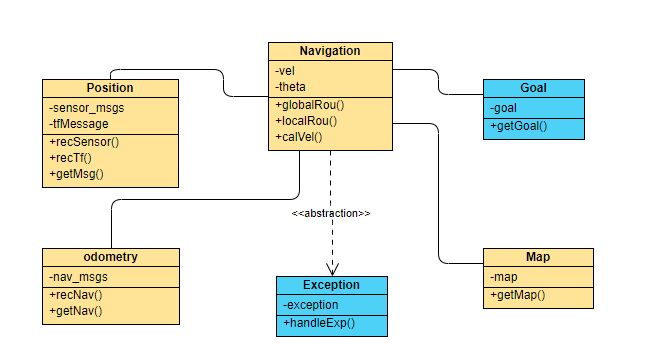
## 路径规划模块

**输入：**测位信息（速度+角度）、坐标变换关系、传感器信息、地图、目标坐标

**输出：**机器人速度（V）、角度（θ）

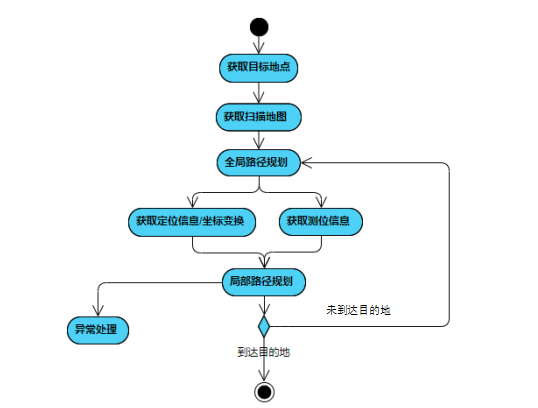
**功能**：该模块规划机器人路径，根据已有的部分信息规划出全局和局部路径，用以控制机器人以特定速度和方向到达目的地

**类图**：



Navigation类主要根据输入的信息进行路径规划，vel和theta是路径规划之后需要计算出的机器人下一步速度与角度。globalRou用于规划全局路径，即根据costmap利用A\*等算法建立出大致路径；locaRou用于规划局部路径，机器人根据全局路径截取出局部路径并确定目标地点，之后根据姿态信息、定位信息算出多条局部路径，根据一定的评优准则选取最优的一条作为机器人下一步走的轨迹。规划路径过程中需要调用amcl包和激光雷达等传感器对机器人的位置做出估计，即sensor\_msgs，然后利用tf树得到机器人的坐标转换信息tfMessage，将全局坐标转换为以机器人为中心的坐标，这一部分功能由Position类实现。规划路径的过程中还需要根据测位信息nav\_msgs对机器人的速度和角度采样，根据这些信息对多条路径进行预测，测位信息主要由Odometry类实现。Map类提供建立好的场景地图，Goal类提供目标地点的xml数据。如出现异常情况，由Exception类捕捉并处理。

**活动图：**

****

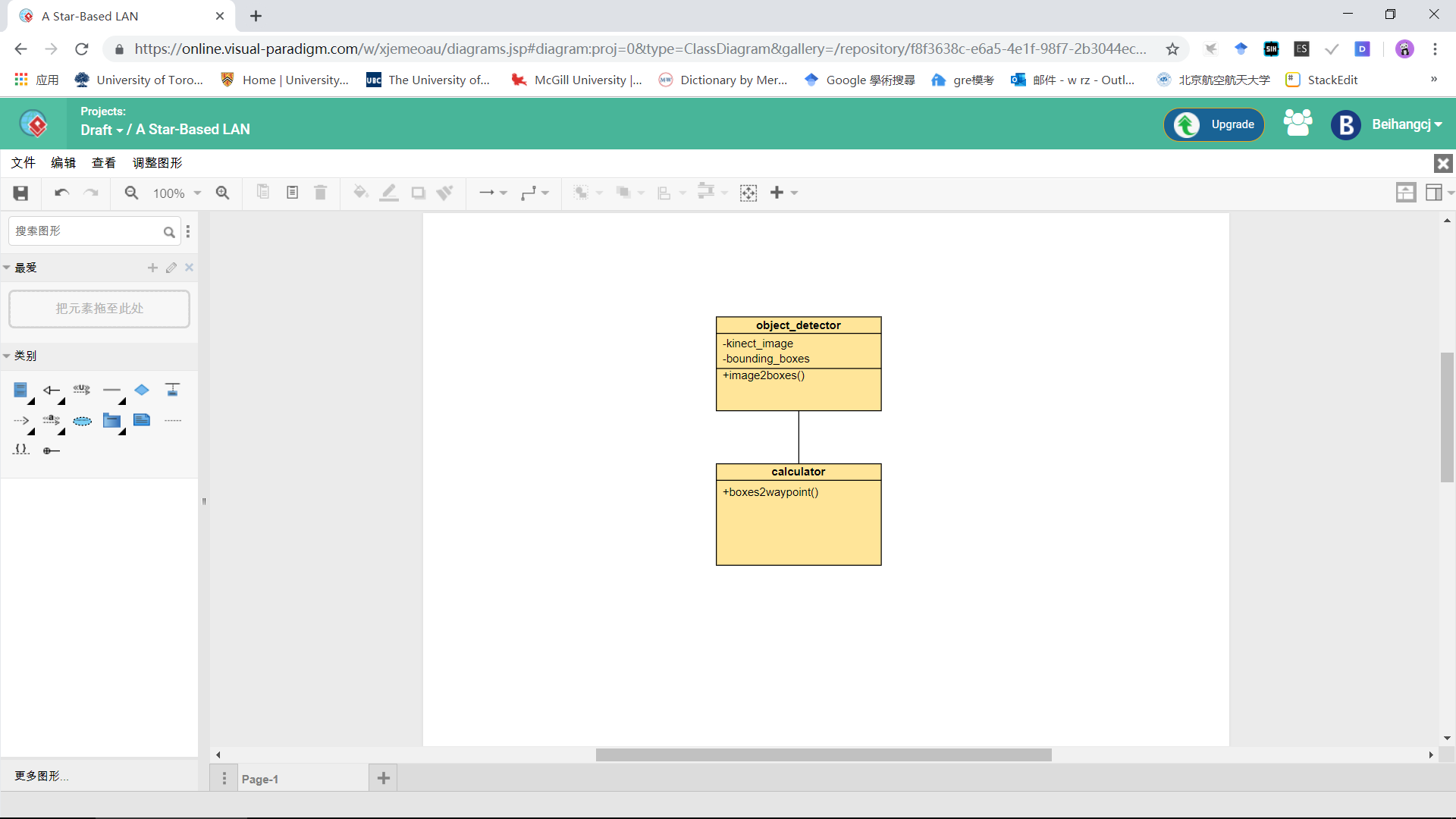
## 目标识别模块

**输入：**摄像头拍摄的真实世界的图像

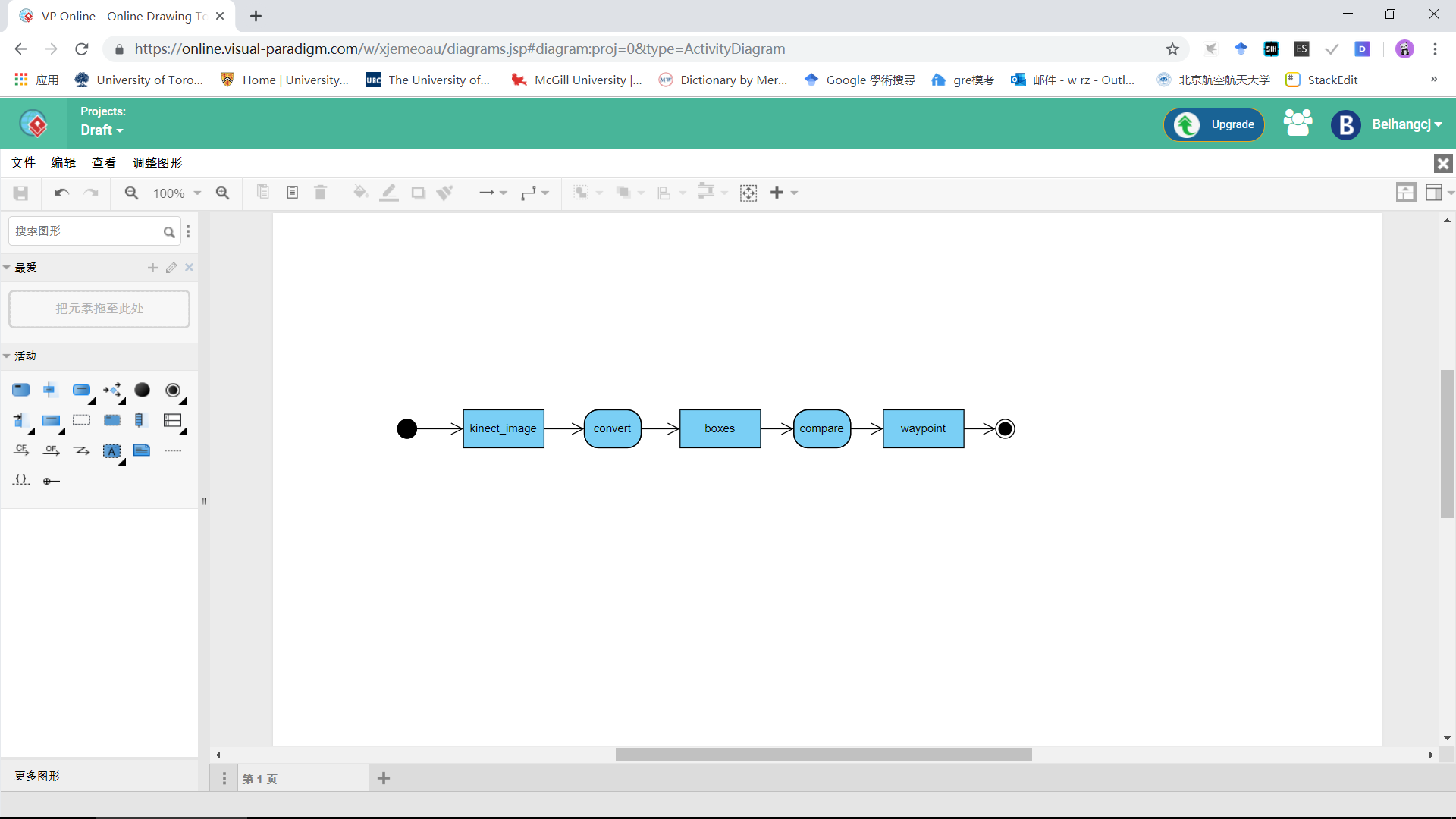
**输出：**物体对应的航点位置

**功能：**识别需要检测的物体，计算机器人要去的航点位置

**类图：**



**活动图：**



在进行程序处理前，需要预先对摄像头进行标定，这是由于摄像头这种精密仪器对光学器件的要求较高，由于摄像头内部与外部的一些原因，生成的物体图像往往会发生畸变，标定可以有效避免数据源造成的误差

该模块的主要以ROS的kinect2来捕获真实世界的图像作为输入， object\_detector类主要接收相机信息（/image\_color）将其转化为物体的位置和种类信息。再交给calculator模块，这个模块通过比较物体之间的相对位置，计算出要抓取物体对应的航点。

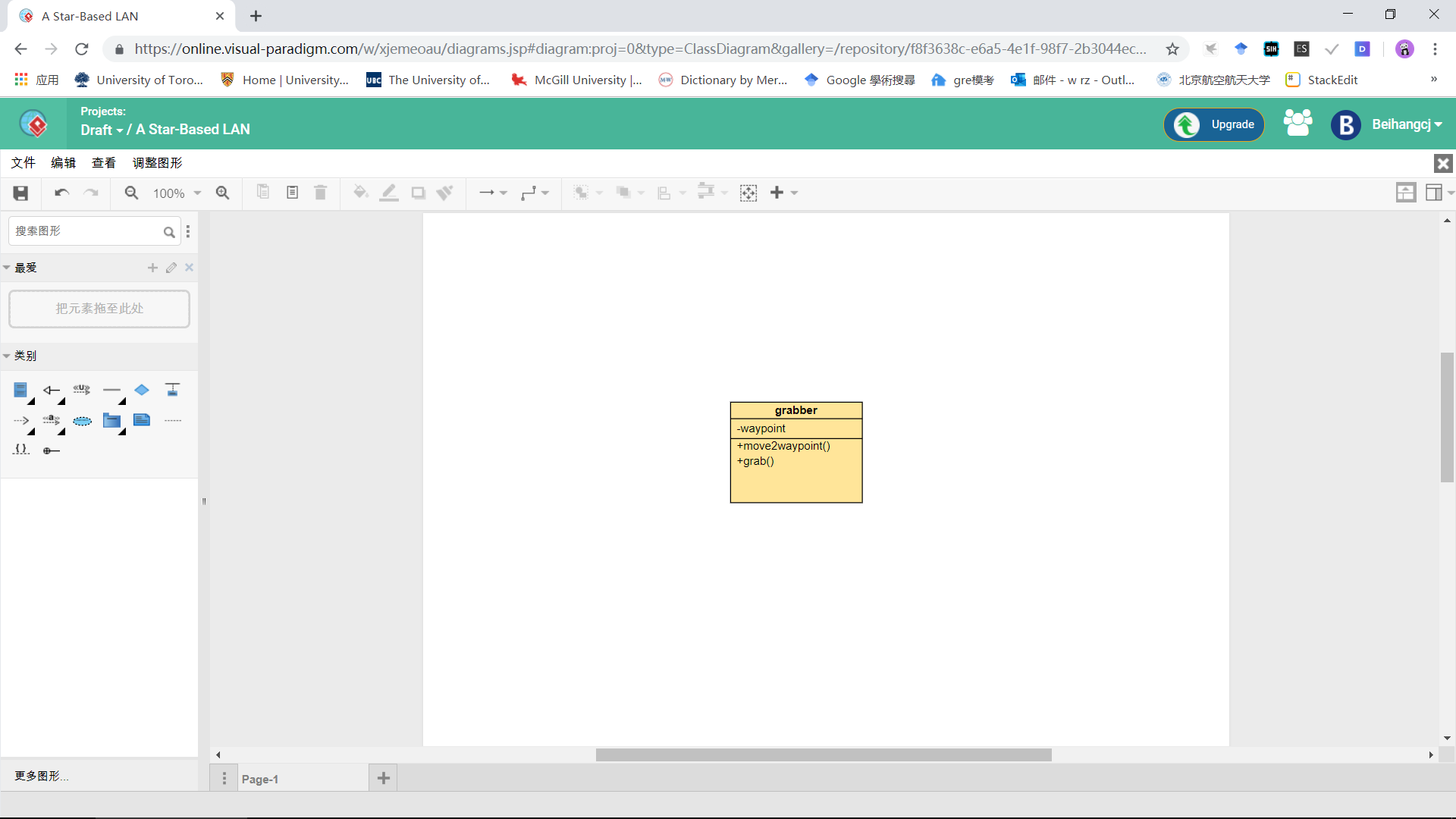
## 抓取模块

**输入：**由识别部分传来的航点

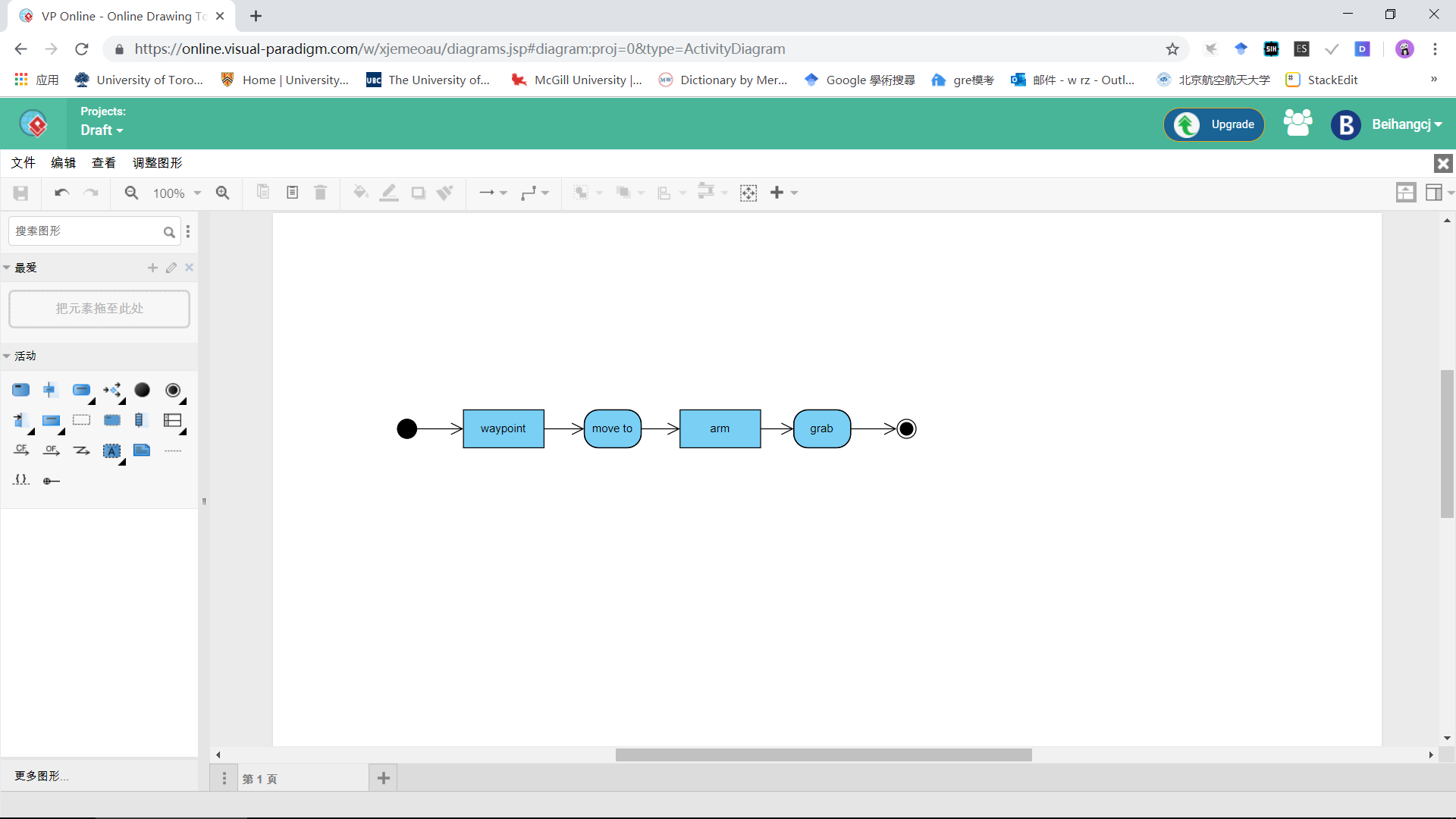
**输出：**无输出，结果是操作机械臂

**功能：**抓取目标物体

**类图：**



**活动图：**



该模块的输入为模块传来的目标航点，通过move2waypoint函数使机器人移动到航点位置，然后通过grab函数根据预设的参数调整机械臂的姿态，完成抓取任务。

# 运行与开发环境

## 运行环境

使用智启ROS机器人作为机器人，机器人自重30kg，承重能力约为10kg。

机器人应当运行在室内环境中，工作平面应当能承受40kg的重量且不应过软，否则可能导致机器人运行时卡住。地面建议使用商用地毯或瓷砖等材质。原则上机器人在水平面上工作，坡度不应大于15度，否则易发生倾覆。

机器人不具备防水措施，因此不应与任何液体接触，否则可能导致电路和机构损坏。

机器人工作温度为15℃到35℃，使用中应远离明火和其他热源。

使用运行在Ubuntu14.04操作系统上的ROS indigo软件进行控制。

## 软件环境

整个开发过程我们使用运行在Ubuntu14.04上的ROS indigo软件和RoboWare Studio进行开发调试。包括基础软件包、依赖软件包、Rviz可视化界面。

# 需求可追踪性说明

## 功能需求

1、自由行走且能否自主避障

主要依赖Movement行走模块来实现行走需求，Movement中的evade模块可以计算前方是否有障碍物，如果有的话就换一个方向重新行走，这实现了避障需求。

2、导航

主要依赖navigation类来规划行走和initmap类来创造地图。

3、识别并抓取物体

1）程序根据kinect相机得到的照片，能够识别物体的种类和其位置，这对应了识别物体的需求。

2）程序在通过一定算法得出航点后，机器人需要走向那个航点以完成抓取，这个对应了控制机器人规划路径的需求。

3）到达指定航点后，机器人可以抓取物体，这对应了抓取物体的需求。

## 非功能需求

1. 速度适宜。速度不能过快，否则会影响避障等功能的实现；速度不能过慢，否则执行任务效率低下。机器人应动态地调节速度，在较长距离无障碍的路径中，可适当地增加速度，一旦检测到障碍物时，应及时减速，如果距离过近可暂停。
2. 避免损伤。除了避免速度过快造成的损伤外，还要注意在接近地图安全边界时避免碰撞。另外，在进行物体抓取时，需要注意机械臂的姿态和力度，尽量在确保牢固的同时避免物体受到损坏。
3. 可维护性。在系统发生故障后能够排除故障予以修复。
4. 操作方便。要尽量易上手、易操作。