目标检测及抓取型机器人

软件设计说明书

SDD 202

版本号 1.5

分工说明

小组名称	护肝队	
学号	姓名	本文档中主要承担的工作内容
17373505	曹博文	4、8；审核
17231002	曹玥	6
17373552	莫策	3、6
17373210	王波文	2、5
17373128	全庆隆	1、6、7；文档整体排版及组织

版本变更历史

版本	提交日期	主要编制人	审核人	版本说明
1.1	4.15	全庆隆	曹博文	完成软件设计说明第一版
1.2	4.20	全庆隆	曹博文	修改需求概述及基本用例图；修改体系结构设计；修改内部接口；修改详细设计；修改需求可追踪性说明
1.3	4.21	全庆隆	曹博文	增加数据库设计
1.4	5.1	全庆隆	曹博文	在第 4 章增加交互图
1.5	5.20	全庆隆	曹博文	修改数据库设计；增加用户界面设计相关内容；修改详细设计

目录

1. 范围

1.1 项目概述

本项目意在开发一个目标检测及抓取型机器人。

本项目的开发基于启智 ROS 机器人。启智 ROS 机器人是一款为 ROS 机器人算法开发量身打造的机器人硬件平台，拥有硬件里程计、激光测距雷达、立体视觉相机和语音输入输出阵列等一系列硬件设备，完美适配 ROS 的 TF、Navigation、Actionlib 和 Pluginlib 子系统，是深入学习 ROS 和高级机器人算法验证开发的理想平台。

本项目目标实现一系列功能性需求，其中包括：实现机器人的主动控制；静态或动态障碍物避障；机器人利用传感器实时建立环境地图；机器人根据地图和自身的位置信息实现动态路径规划及导航控制；检测、识别并定位环境中的特定目标，动态接近目标物；抓取目标物；语音交互；多目标检测。

本项目目标完成的非功能性需求包括：在 5 秒内对于正确的用户需求做出响应；对于错误的用户需求进行提示并正确处理；若出现错误可在有限时间内恢复；拥有简单的指令格式，易于使用和学习。

此目标检测及抓取型机器人可作为服务机器人在现实生活中有所应用，例如在餐饮业中用于为顾客送餐、在酒店服务中帮助带领顾客到指定房间、在仓库管理中搬运货物等，应用前景广泛。

1.2 文档概述

本文档是北京航空航天大学计算机学院 2020 年春季学期软件工程（嵌入式方向）课程中护肝队的软件设计说明书文档。本文档适用于基于 Ubuntu16.04 系统的 ROS 机器人操作系统开发，适用的硬件平台是启智 ROS 机器人，编写的软件的用途是在该硬件平台上实现一个具有目标检测和抓取功能的机器人，其功能主要包含机器人的主动控制，实时建立环境地图，静态/动态障碍物避障，路径规划和导航控制，单种/多种目标物的检测、识别和抓取，以及语音交互。说明书包括如下内容：需求概述、数据库设计、体系结构设计、接口设计、详细设计

以及需求可追踪性说明。与本文档相关配套的，还有如下文档：SDP 软件开发计划文档、SRS 软件需求规格说明文档、STD 软件测试说明文档。本文档初次撰写于 2020 年 4 月 15 日。

本项目的开发计划用于总体上指导 ROS 机器人软件项目顺利进行并得到通过最终评审的项目产品。本项目开发计划面向项目组的全体成员，项目周期为 3 个月。

1.3 术语和缩略词

缩略词	全称
ROS	Robot Operating System/机器人操作系统
URDF	Unified Robot Description Format/统一机器人描述格式
IMU	Inertial Measurement Unit/惯性测量单元
SLAM	Simultaneous Localization and Mapping/即时定位与地图构建
TOF 立体相机	Time of Flight/飞行时间技术

1.4 引用文档

文档格式要求按照我国 GB8567-2006 计算机软件文档编制规范进行。引用文档包括以下文件：

软件设计说明书 GB8567-2006 (SDD)

《启智 ROS 版_开发手册》

2. 需求概述

2.1 概述

本项目拟开发室内智能服务型机器人，在机器人管理员完成对机器人的配置，且手动控制机器人完成对房间的建图后，机器人从设置的起点开始运行。机器人可以跟随用户或自主去往指定地点，同时室内有预设的目标地点，放置着一些可以被机器人抓取的物品。用户发出语音指令或者在用户界面进行操作，机器

人接受指令并分析用户要求，之后机器人完成相应操作。

机器人要实现的基本功能有建图、自主导航、物品识别与抓取、动态避障及语音交互。在非功能需求方面，则需要保证机器人的高效性，系统的易用性、可靠性、可扩展性及可维护性。

2.2 基本用例

2.2.1 机器人建立环境地图

主要参与者：机器人管理员、机器人

目标：使机器人完成对房间的建图

前置条件：完成机器人的硬件及软件的启动

启动：机器人管理员发出建图指令

场景：

1. 机器人启动雷达开始建图
2. 机器人管理员控制机器人在房间中移动
3. 机器人在移动过程中完成对房间的建图
4. 机器人保存地图

优先级：高

何时可用：第一个增量

使用频率：低

次要参与者：机器人所携带的各类传感器

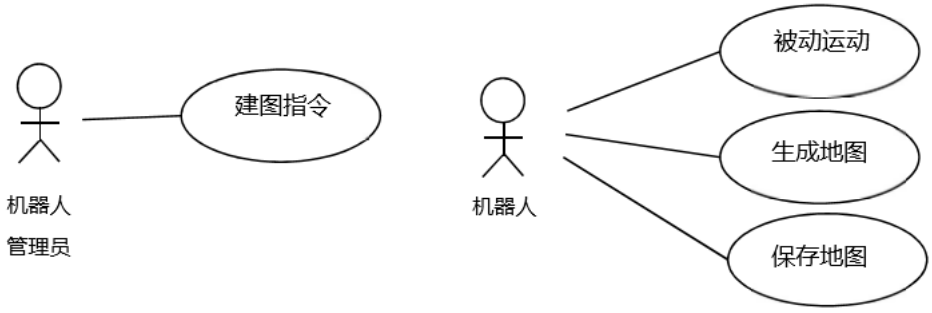

图 1 建立环境地图用例图

2.2.2 机器人的自主导航

主要参与者：用户、机器人

目标：在用户选定目的地后，机器人基于环境地图进行路径规划，而后基于主动控制开始运动，并在运动过程中动态避障、调整路径

前置条件：已进入导航模式

启动：用户选定目的地并确认启动机器人

场景：

1. 用户点击“选择目的地”
2. 系统弹出 RIVZ 界面显示地图
3. 用户在地图上标定目的地并点击确认
4. 用户点击“开始导航”，而后机器人开始运动
5. 机器人到达目的地后停止运动，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：机器人携带的各类传感器以及控制、连接组件

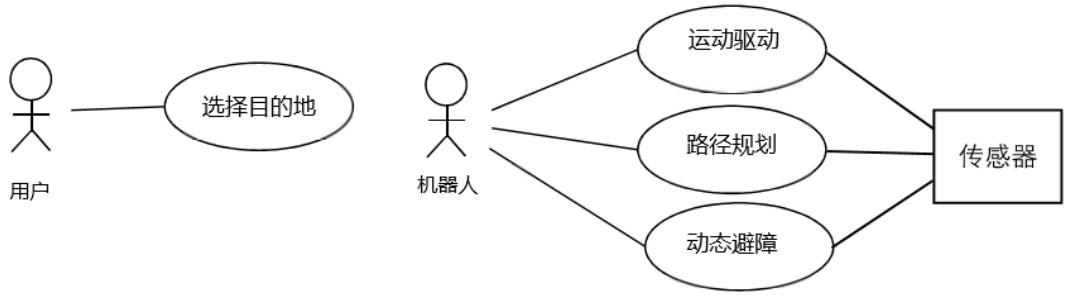

图 2 自主导航用例图

2.2.3 机器人的目标检测及抓取

主要参与者：机器人

目标：机器人可以准确识别视野中的单个/多个物体，并抓取目标物体

前置条件：已进入抓取模式

启动：用户确认启动机器人

场景：

1. 机器人识别出视野中物体
2. 机器人获取物体的位置、形状信息
3. 机器人对抓取行为进行规划
4. 机器人控制机械臂抓取物体
5. 机器人抓取完毕，并发出语音提示

优先级：中

何时何用：第三个增量

使用频率： 中

次要参与者：机器人携带的各类传感器、机械臂、控制和连接组件

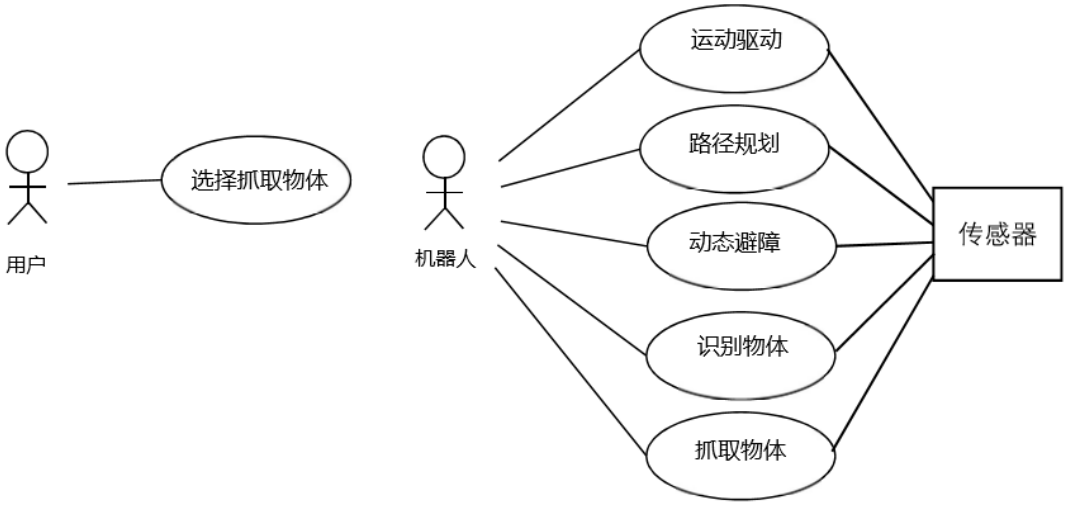

图 3 目标检测及抓取用例图

2.2.4 机器人的语音交互

主要参与者：用户、机器人

目标：机器人能够识别用户的语音指令并执行相应的任务

前置条件：用户说出语音指令

启动：用户确认启动机器人

场景：

1. 机器人识别用户说出的语音指令
2. 机器人解析该指令并做出行为规划
3. 机器人执行相应的任务

4. 机器人完成任务，发出语音提示

优先级：低

何时可用：第四个增量

使用频率：中

次要参与者：语音识别模块、机器人携带的各类传感器、控制和连接组件

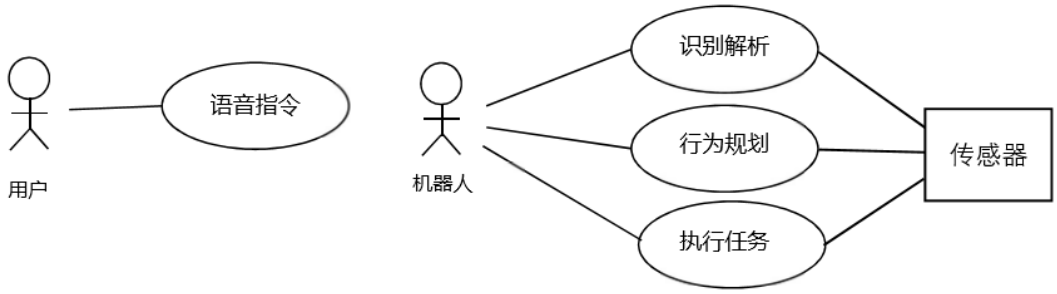

图 4 语音交互用例图

3. 数据库设计

3.1 ER 图

本项目的数据库 ER 图如下：

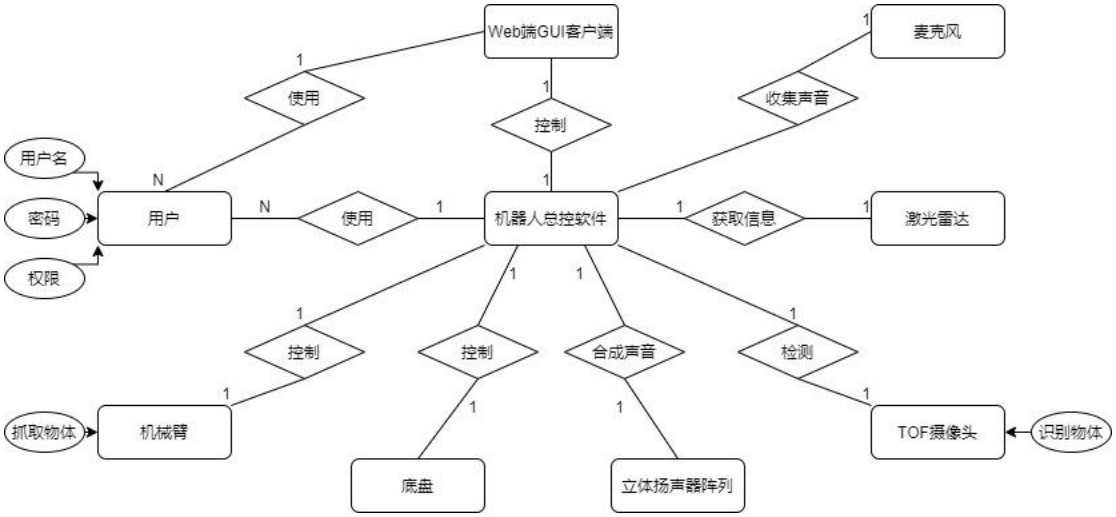

图 5 ER 图

3.2 数据库结构表

数据表名	属性名	属性类型
用户	用户名（KEY）	字符串
	密码	字符串
	用户权限	整数
位置	编号（KEY）	整数
	位置名称	字符串
	创建时间	时间
	坐标	元组
	描述	字符串
抓取物品	编号（KEY）	整数
	物品标签	字符串
	创建时间	时间
	三维坐标	元组
	图像	图像
运行日志	编号（KEY）	整数
	创建时间	时间
	事件信息	字符串

3.3 数据库关系模式

3.3.1 用户表

该关系的主码是用户名。密码和用户权限完全依赖于主码，因此是 3NF.

名称	字段名	数据类型	备注说明
用户名	UserName	字符串	用户的唯一标识
密码	Password	字符串	需要加密
用户权限	Authority	整数	0 表示开发权限，1 表示管理权限，2 是普通用户权限

数据类型：

Create table UserTable (

Identifier INT UNSIGNED AUTO_INCREMENT,

UserName CHAR(99),

Password CHAR(99),

Authority INT

);

数据存入时间：表中包含开发人员管理员的数据信息；web 端用户完成注册则向表中添加数据；

数据的用处：控制不同权限用户的操作权限，保障机器人运行的安全性。

3.3.2 地图位置关系表

主键是地点的编号，对用户透明，由系统给出。

名称	字段名	数据类型	备注说明
编号	Identifier	整数	地点的唯一标识
位置名称	PlaceName	字符串	用户标定的地点名称
创建时间	Time	时间	创建该地点的时间（UTC 时间）
坐标	Position	元组	(x, y) 坐标
描述	Description	字符串	用户对该地点的描述

数据类型:

Create table MapPlaceNameTable (

Identifier INT UNSIGNED AUTO_INCREMENT,

placeName char(10) NOT NULL,

time TIMESTAMP,

position_X float(24),

position_Y float(24),

Description Text,

);

数据存入时间：用户在地图中添加标记地点时将数据信息存入表中；

数据的用处： 存储用户定义的地点位置，方便机器人的使用。

3.3.3 抓取物品表

主键是抓取的物品的编号，对用户透明，由系统给出。

名称	字段名	数据类型	备注说明
编号	Identifier	整数	物品的唯一标识
物品标签	Label	字符串	识别该物体得到的名称
创建时间	Time	时间	抓取该物品的时间（UTC 时间）
三维坐标	Position	元组	抓取该物品的 (x, y, z) 坐标
图像	Image	图像	摄像头采集到的该物品的图像信息

数据类型：

```
Create table CatchTable (
Identifier INT UNSIGNED AUTO_INCREMENT,
Label char(99) NOT NULL,
time TIMESTAMP,
position_X float(24),
position_Y float(24),
position_Z float(24),
Image char(99),
);
```

数据存入时间：机器人完成抓取的全部动作并且抓取成功时，向表中插入相应数据；

数据的用处：记录抓取到的物体，方便开发人员和管理人员调试运行及故障处理。

3.3.4 运行日志表

名称	字段名	数据类型	备注说明
编号	Identifier	整数	日志信息的唯一标识
创建时间	Time	时间	创建本条日志信息的时间（UTC 时间）
事件信息	Info	字符串	程序自动生成的日志信息

数据类型：

Create table RunningLogTable (

Indentifier INT UNSIGNED AUTO_INCREMENT,

Time TIMESTAMP,

Info TEXT

);

数据存入时间：用户注册、用户对机器人的操作、机器人响应等事件的日志信息；

数据的用处：记录机器人的运行状态；方便开发人员维护与排除故障。

4. 体系结构设计

4.1 总体结构

本节将从实现视图，以构件结构形式给出系统的总体结构。

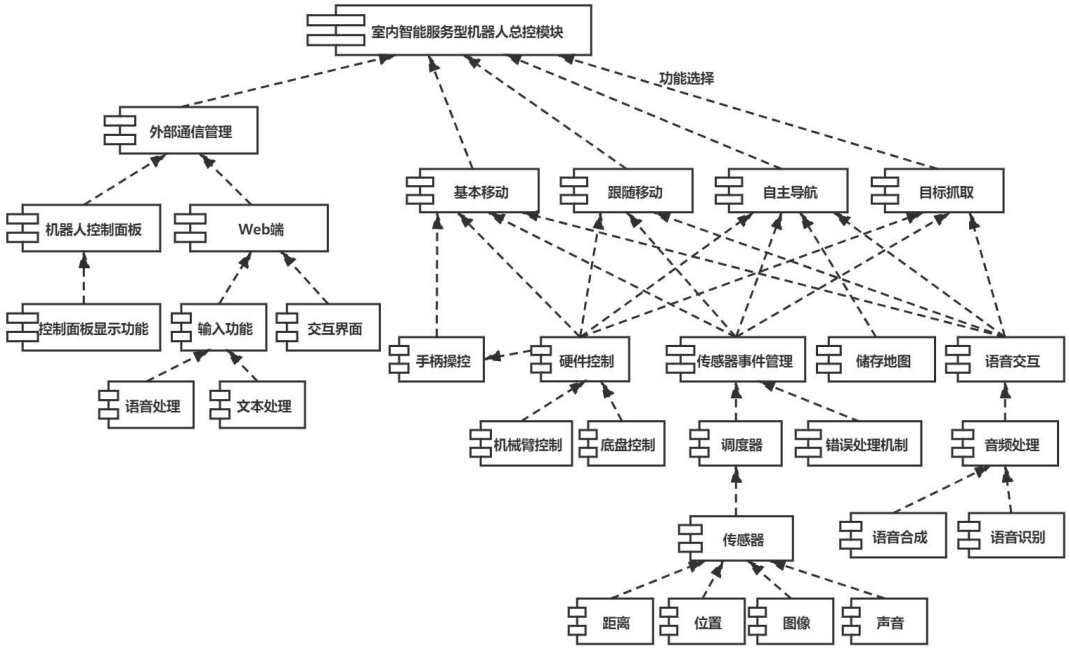

图 6 系统构件图

系统的核心单元为外部通信管理模块，机器人总控模块，以及基本移动、跟随移动、自主导航、目标抓取四种功能的控制模块。

外部通信管理模块获取来自机器人控制面板和 Web 端的指令，向机器人总控

模块发送消息。机器人控制面板需具备显示功能；Web 端需要设置交互界面并具备处理语音和文本输入的功能。

机器人总控模块监听外部通信管理模块，解析指令，按需激活功能控制模块。

基本移动、跟随移动、自主导航、目标抓取四种功能的控制模块调用硬件控制模块和传感器事件管理模块完成任务，基于语音交互模块完成语音的输入和输出。硬件控制模块需具备控制机械臂和底盘的功能，同时可以接受手柄的操控；传感器事件管理模块通过调度器来接收各传感器的状态及信息、与事件相关联，同时具备必要的错误处理机制。此外，地图作为重要信息存储在自主导航模块中。

4.2 软件体系结构

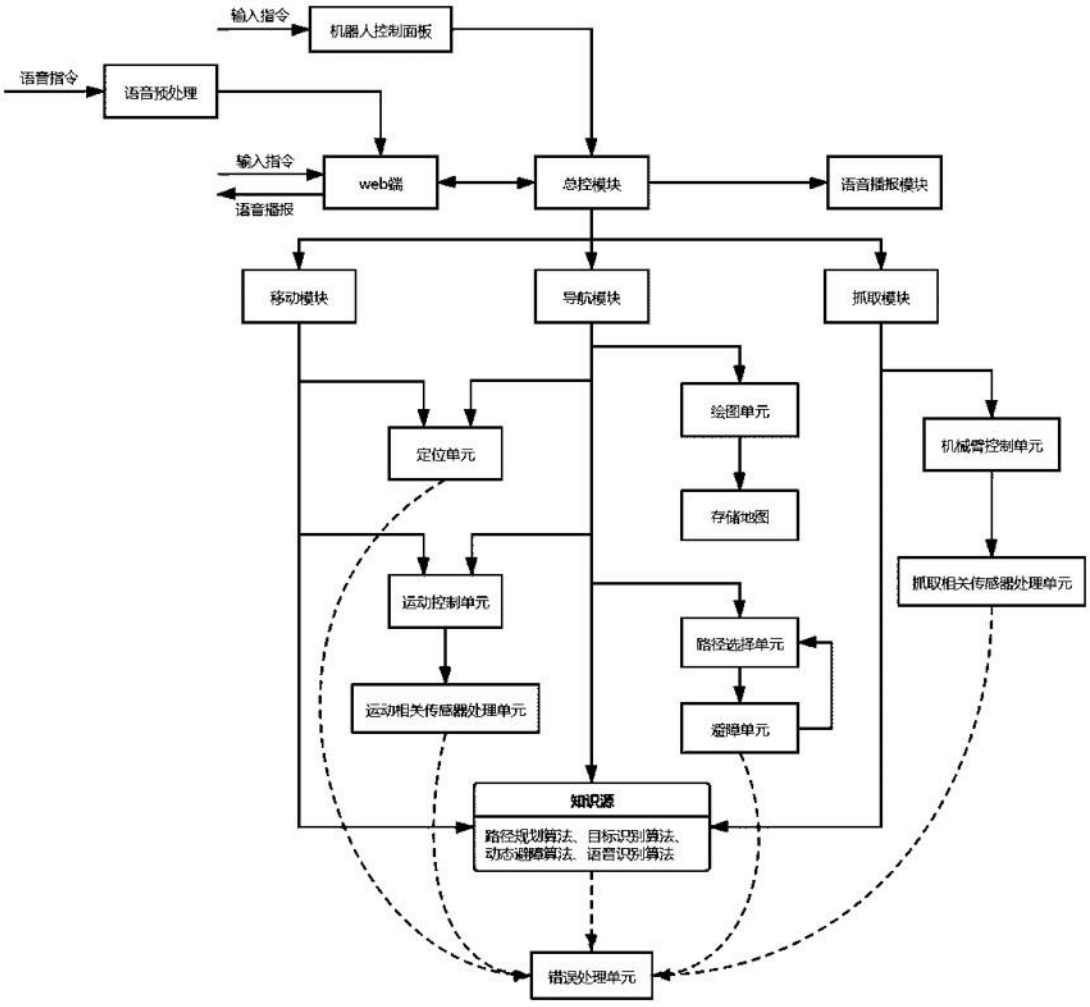

图 7 软件体系结构图

软件体系结构采用混合结构风格。

在 web 端和总控模块间采用分层结构模式（客户/服务器风格），提高了室内智能服务型机器人的软件复用性和可扩展性。

在总控模块和移动、导航、抓取模块间采用以数据为中心的结构模式（仓库系统风格）。基于知识源中的路径规划算法、目标识别算法和动态避障算法，总控模块调度移动模块、导航模块、抓取模块执行任务，提供基本移动、跟随移动、自主导航、目标抓取四项功能。

在移动、导航、抓取模块内部采用调用/返回结构（主程序结构）。其中，移动模块依照总控模块传来的模式信息，通过调用运动控制单元和定位单元提供基本移动、跟随移动两项功能；导航模块基于总控模块传来的目标位置信息、内部存储的地图信息以及知识源中的路径规划算法和动态避障算法，调用定位单元、运动控制单元以及路径选择单元提供自主导航功能；抓取模块基于总控模块传来的目标物体信息和知识源中的目标识别算法，调用机械臂控制单元，提供目标抓取功能。

在定位单元、运动相关传感器处理单元、避障单元和抓取相关传感器处理单元中均加入了错误处理机制。当调用知识源中算法出错时，也会调用错误处理单元。

4.3　硬件体系结构

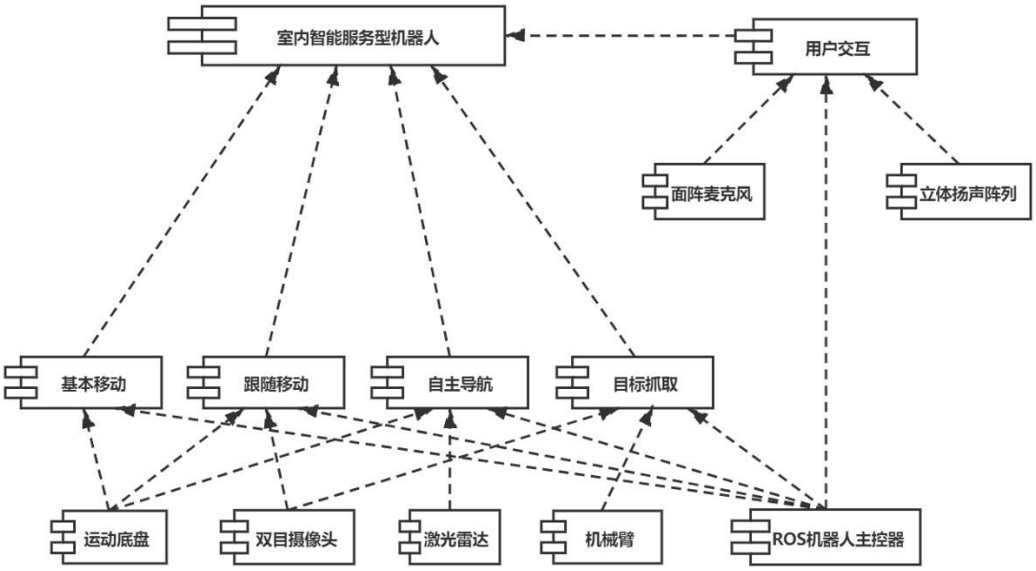

图 8　硬件体系结构图

本节主要分析了室内智能服务型机器人提供的各项功能的硬件依赖关系。其中，基本移动功能依赖于运动底盘和 ROS 机器人主控器；跟随移动功能依赖于运动底盘、双目摄像头和 ROS 机器人主控器；自主导航功能依赖于运动底盘、激光雷达和 ROS 机器人主控器;目标抓取功能依赖于双目摄像头、机械臂和 ROS 机器人主控器；用户交互功能依赖于 ROS 机器人主控器、面阵麦克风和立体扬声阵列。

4.4 软件整体类图

软件整体类图如下：

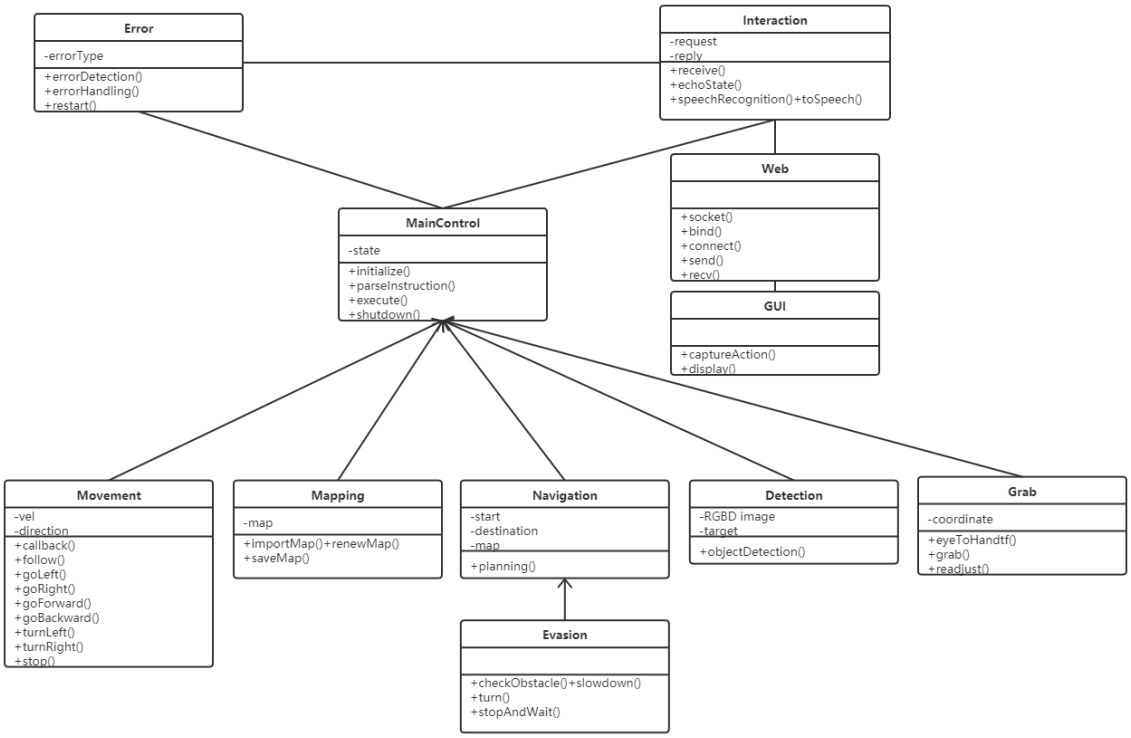

图 9 软件整体类图

类之间协作关系如下图：

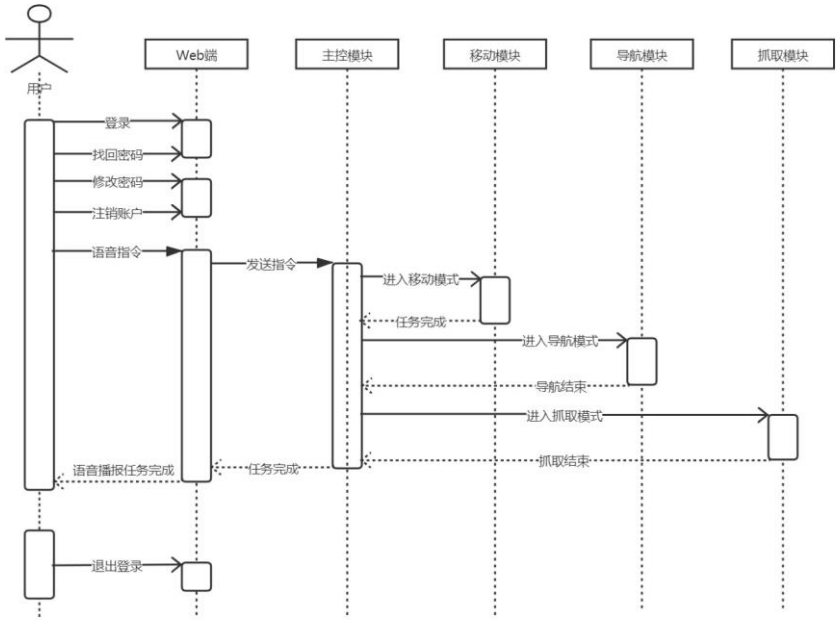

图 10 总体交互图

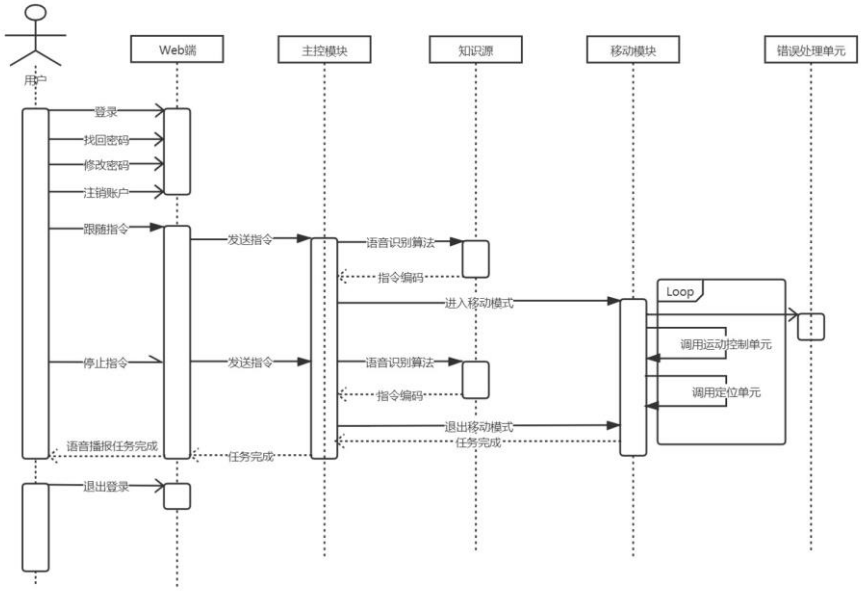

图 11 移动模式交互图

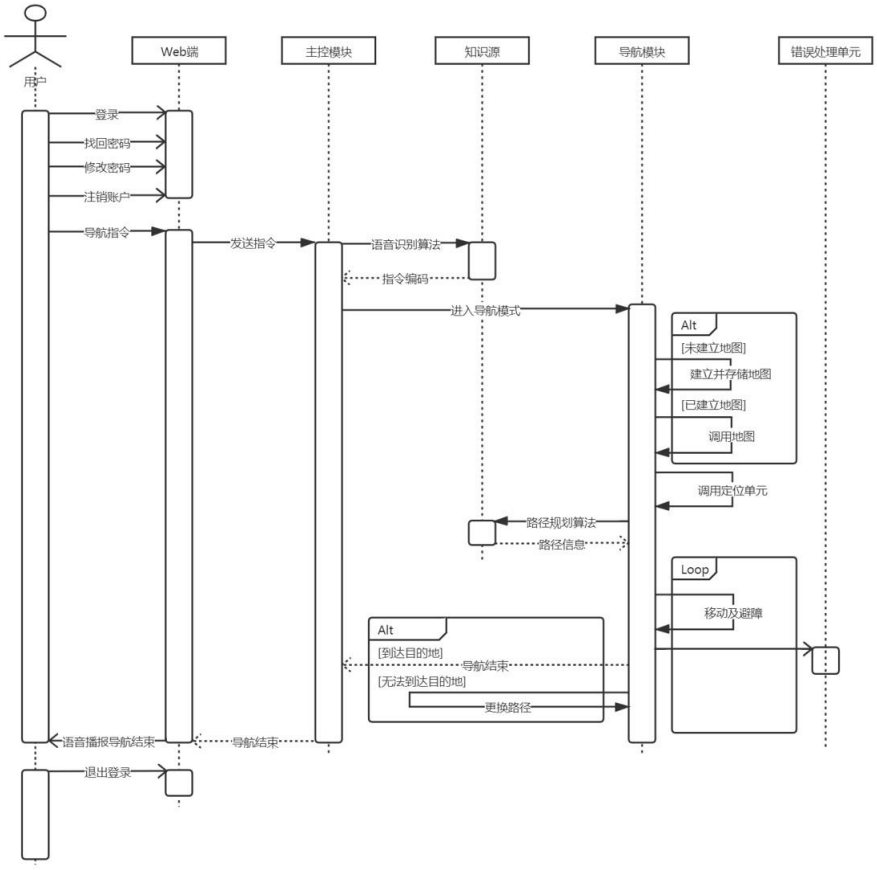

图 12 导航模式交互图

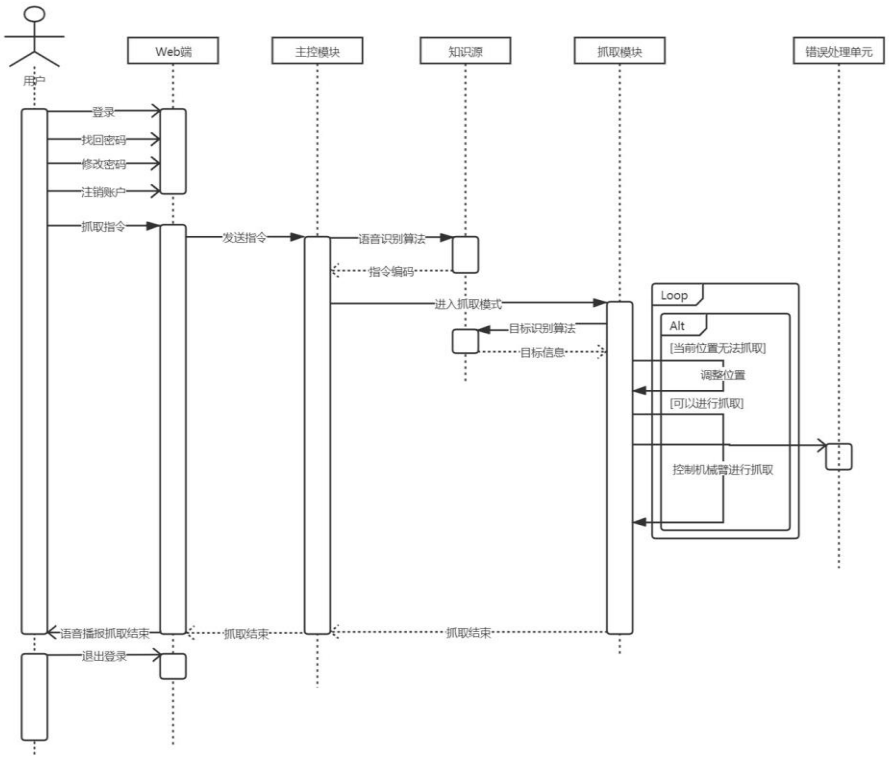

图 13 抓取模式交互图

5. 接口设计

5.1 用户界面

5.1.1 界面分析

1. 用户分析：通过界面与系统交互的人为机器人使用者或管理员，对系统的熟悉程度不同。
2. 任务分析：用户承担选择功能和操作机器人的任务。
3. 内容分析：界面需要显示的内容包括窗口、菜单、对话框，完成系统的交互和对用户操作的引导。
4. 环境分析：使用手机或电脑进行远程访问，完成对机器人的控制。

5.1.2 界面设计

本项目根据需求用例和活动图提取界面对象和作用于其上的行为，进而设计界面，使得用户能够通过界面在满足系统约束时完成所有定义的任务，具体实现详见下节。

5.1.3 界面实现

本项目的界面实现选择基于 web 端的方式，使用 flask 及 bootstrap 框架，运用 Html、Python、JavaScript 语言完成。旨在为用户提供可视化界面，更加简单便捷的使用机器人的各项功能，完成与机器人的交互。

用户通过远程电脑上的 web 端控制机器人。控制需要有选择模式的界面、控制机器人进行移动的界面和让机器人完成指定操作的按钮。界面及按钮如下图所示：

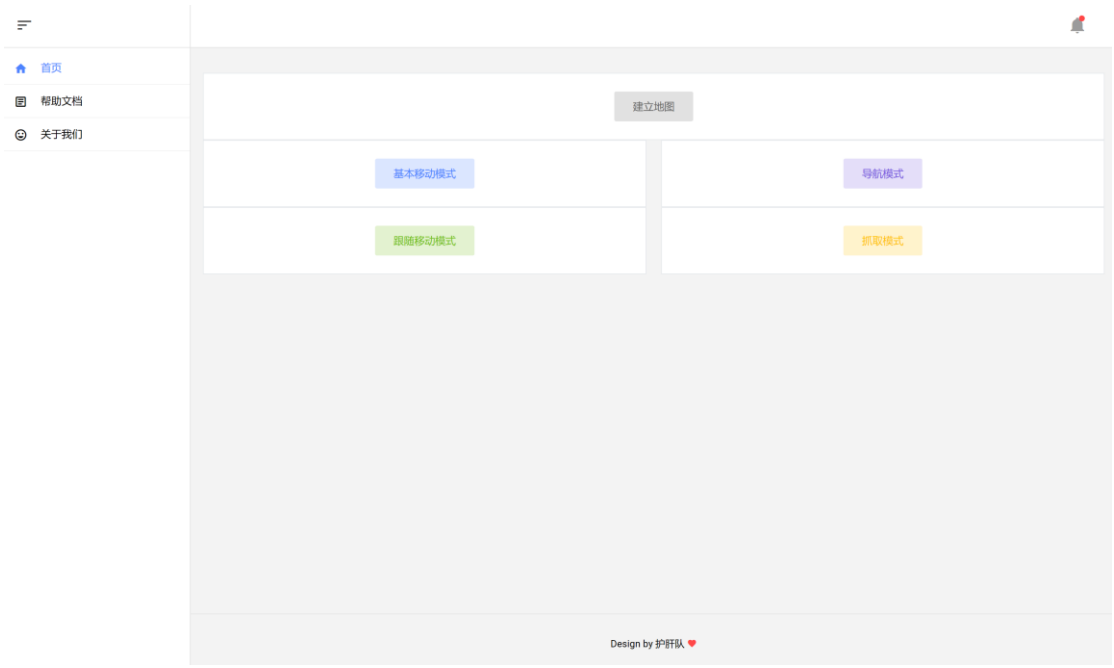

图 14 主界面

图 15 基本移动模式界面

图 16 跟随移动模式界面

图 17 导航模式界面

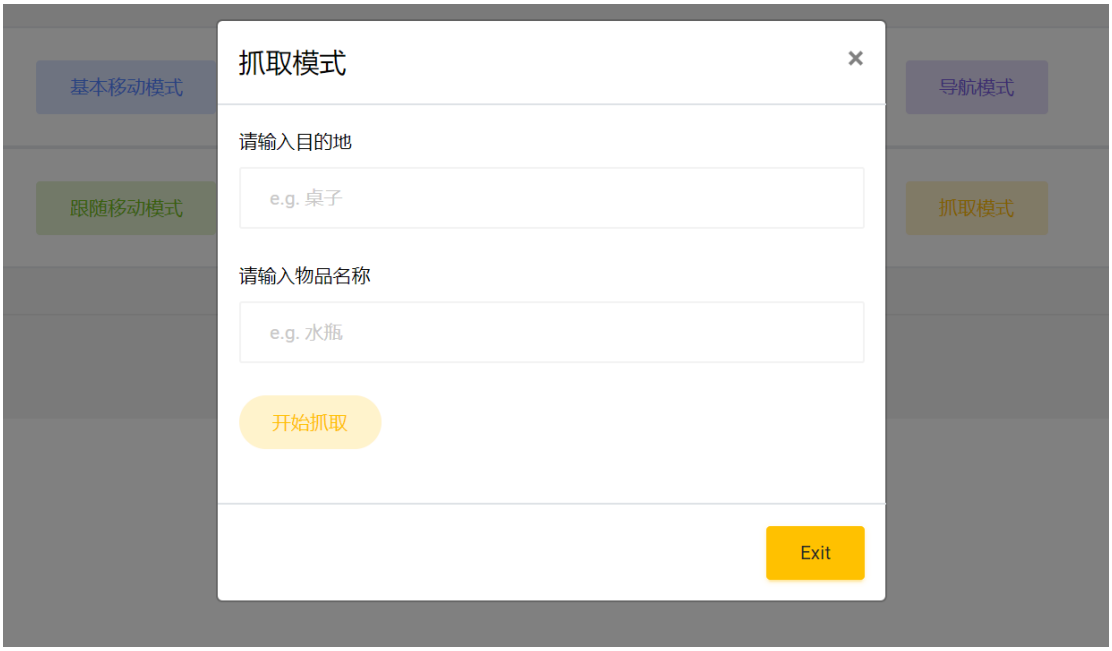

图 18 抓取模式界面

若 web 端与机器人主控电脑端成功建立 TCP 连接，则会跳转到主界面，主界面清晰的标明机器人的四种功能，用户点击按钮，会弹出相应功能的对话框界面。点击界面上的按钮，前端通过 ajax 将命令传回后端，通过 socket 技术后端将收到的指令用 TCP 连接传到机器人控制端的主控模块，机器人完成相应指令并反馈信息，通过 TCP 连接传回后端，后端将数据提交给前端，显示在页面上，从而实现交互。

5.1.4 界面评估

问卷采集得到的用户体验结果表明，本项目的用户界面简洁易懂，美观适用，

能够与软件功能相对应，并能给用户呈现反馈。经过简单介绍后用户可以通过用户界面操作机器人。此外界面布局、色彩设计风格也具有一致性。

5.2 外部接口

机器人顶部配备了一个 Kinect2 相机，一个可以在上下前后运动的机械臂，扬声器及麦克风阵列。位于机器人躯干部分外挂的平板电脑运行 ROS 操作系统， 通过 USB 接口与机器人底盘内的 USB-HUB 连接。USB-HUB 将计算机的 USB 接口扩展为多路。 扩展后的 USB 接口分别连接到启智控制器（以异步串口方式访问）、USB 转以太网接口、激光雷达、面板接口（用于用户自行连接设备，例如 U 盘、控制手柄）。

软件外部接口主要为 web 端应用，界面设计如上一节所示。

5.3 内部接口

1. 无线传输接口：

socket() 创建 socket；

bind() 将本地地址与 socket 绑定；

connect() 建立连接；

send() 发送数据；

recv() 接收数据，写入本地绑定的 socket 中。

2. 导航控制接口：

sendGoal() 将导航信息传递给导航服务客户端；

waitForResult() 等待导航结果直到整个导航过程结束，或导航过程被其他原因中断。

3. 物体识别接口：

objectDetection() 调用 kinect2 摄像头进行物体识别。

4. 机械臂控制接口：

init() 定义一个机械臂控制消息对象并初始化；

publish() 将控制消息发布至机器人控制节点，实现对机械臂的控制。

6. 详细设计

本节中，将以类图的形式给出构件的局部数据组织和函数接口，以控制流图的形式介绍构件内控制流，并附上文字描述分析实现细节。

6.1 移动

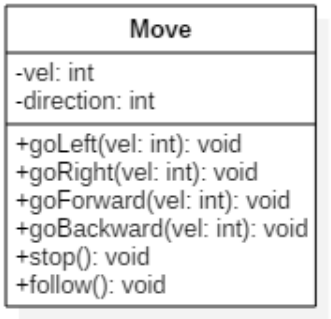

图 19 移动模块类图

属性说明：

1） vel：速度

2） direction：方向（0 代表向左，1 代表向右，2 代表向前，3 代表向后）

操作说明：

移动模块提供基本移动和跟随移动两种操作。

1） goLeft(vel: int)：自定义速度向左移动，直接改变启智机器人的速度和转角变量。

2） goRight(vel: int)：自定义速度向右移动，直接改变启智机器人的速度和转角变量。

3） goForward(vel: int)：自定义速度向前移动，直接改变启智机器人的速度和转角变量。

4） goBackward(vel: int)：自定义速度向后移动，直接改变启智机器人的速度和转角变量。

5） stop()：停止运动。

6） follow()：调用启智机器人的跟随服务，实现跟随运动。

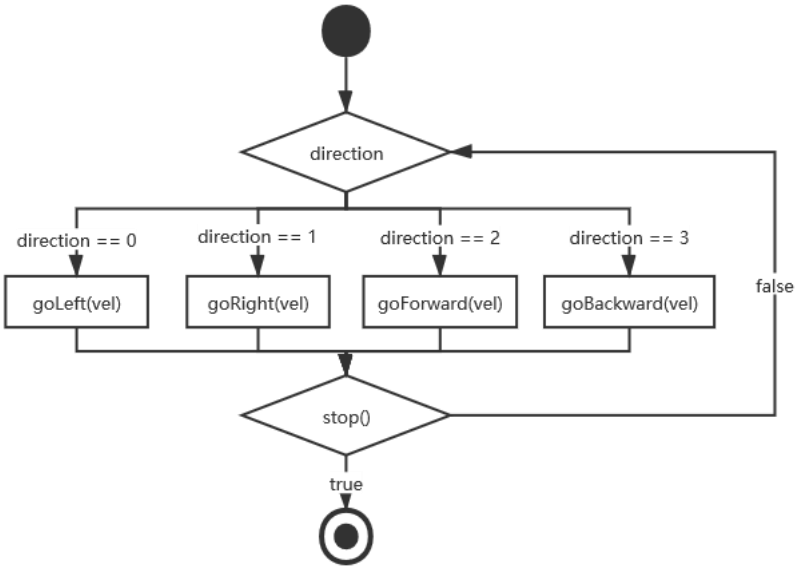

图 20 基本移动控制流程

基本移动控制流程：

根据 direction 决定运动的方向，若 direction==0，则调用 goLeft()函数向左移动；若 direction==1，则调用 goRight()函数向左移动；若 direction==2，则调用 goFoward()函数向前移动；若 direction==3，则调用 goBackward()函数向后移动。移动后，若收到停止信号，则退出移动模式，否则，可以选择新的模式继续移动。

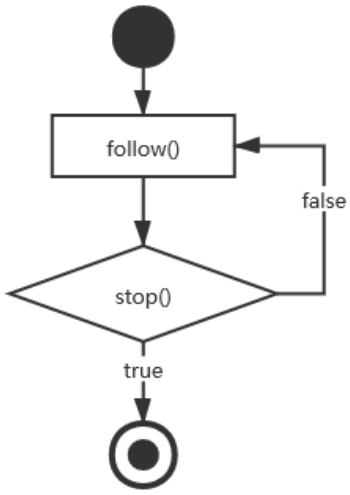

图 21 跟随移动控制流程

跟随移动控制流程：

模块使用启智机器人内置跟随模块，开启跟随模式后，调用启智 ROS 的跟随服务开始跟随。收到停止信号后，调用启智 ROS 的停止跟随服务停止跟随。

6.2 避障

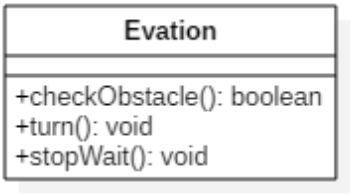

图 22 避障模块类图

操作说明：

1） checkObstacle()：通过调用 lidar_test.launch 检测是否有障碍物。
2） turn()：通过改变启智机器人的运动转角参数进行小角度避障。
3） stopWait()：改变启智机器人的运动各方向速度为 0 从而实现停在原地等待。

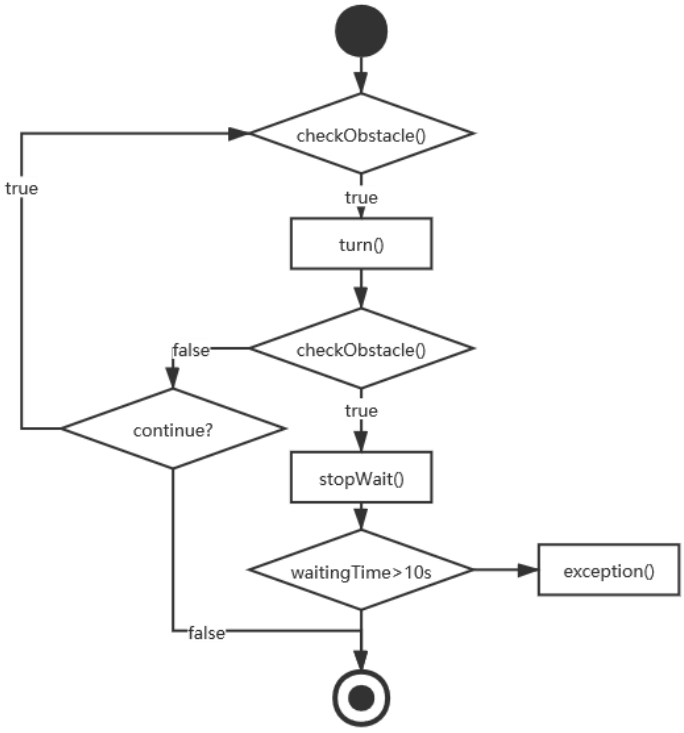

图 23 避障控制流程

控制流程：

先调用 checkObstacle()检测是否有障碍物，若有，则先调用 turn()进行小角度避障，避障后若调用 checkObstacle()检测成功避障，则继续检测，若调用 checkObstacle()检测避障失败，则调用 stopWait()停止等待。若停止等待时间超过 10s，则进入异常处理。

6.3 建立环境地图

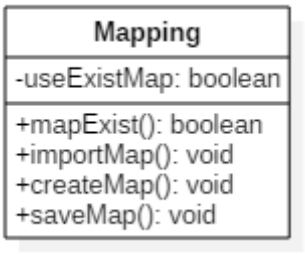

图 24 建图模块类图

属性说明：

1） useExistMap：是否使用已存在的地图

操作说明：

1） mapExist()：查找地图库，看是否有存好的地图。
2） importMap()：导入已建好的图。
3） createMap()：调用启智机器人 hector_mapping.launch，用手推动机器人进行移动建图。
4） saveMap()：调用启智机器人 map_saver 保存地图，保存完毕后，会在 Ubuntu 系统的“主文件目录”里，发现两个新文件，一个名为“map.pgm”，另一个名为“map.yaml”。 其中“map.pgm”为图片格式，双击可以查看图片内容，就是建好的地图图案。

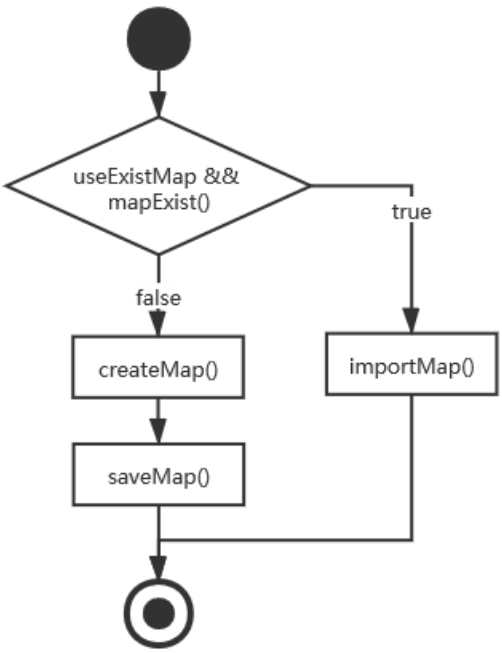

图 25 建图控制流程

控制流程：

若 useExistMap==true 并且 mapExist() == true，则使用 importMap()导入已存在的地图。若不使用已有地图或不存在已有地图，则先使用 createMap()新建地图，在使用 saveMap()保存地图。

6.4 导航

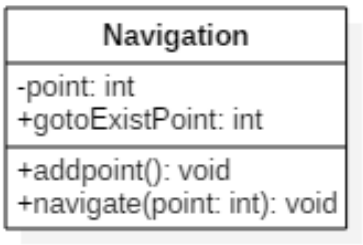

图 26 导航模块类图

属性说明：

1） gotoExistPoint：是否去已标注的点

2） point：点标号

操作说明：

1） addPoint()：加入标注点。

2） navigate(point)：调用启智机器人 nav.launch 进行导航。导航过程中需先回到预设初始点，在前往目标地点。

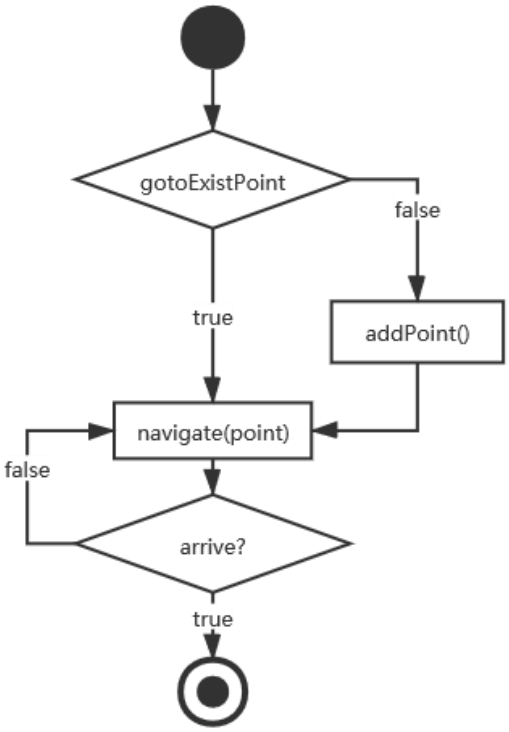

图 27 导航控制流程

控制流程：

若使用已有点，则直接调用 navigation()函数进行导航，若到达，则结束，若未到达，则重新导航。若不使用已有点，则调用 addPoint()添加新的导航点，然后调用 navigation()函数进行导航，若到达，则结束，若未到达，则重新导航。

6.5 目标检测

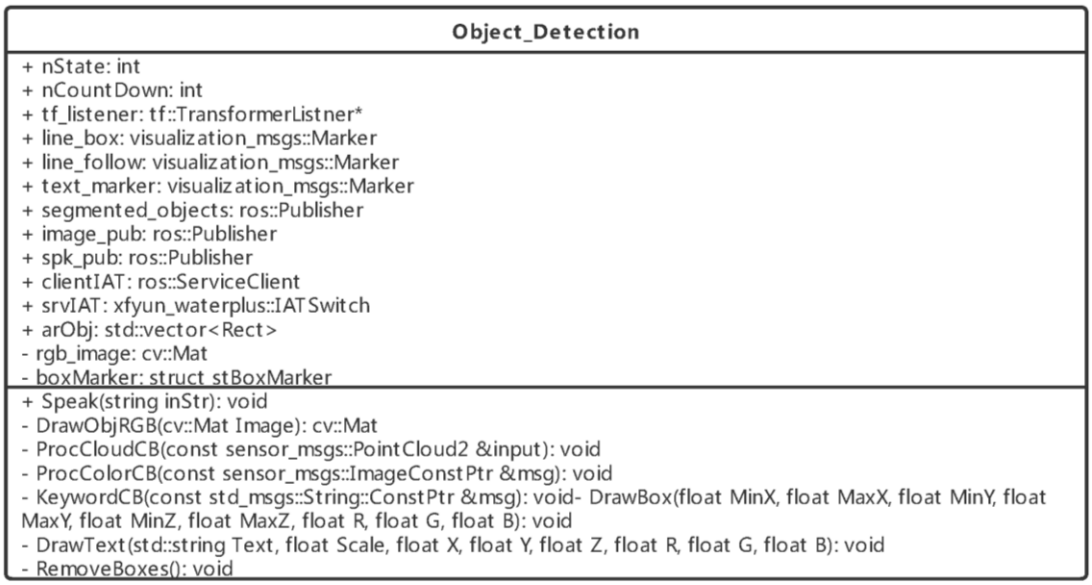

图 28 目标检测模块类图

属性说明：

1）nState 和 STATE_CountDown 是目标检测功能的状态变量，用以保证机器人在检测目标期间不会继续识别语音指令，以避免出错。
2）tf_listener：tf 包提供了 TransformListener 的实现，以帮助简化接收 tf 变换数据的任务，此处设置监听双目摄像头传来的 RGB-D 图像。
3）line_box：用于标注检测到的目标物体。
4）line_follow：用于清除目标物体的标注框。
5）text_marker：用于标注物品类别。
6）spk_pub、segmented_objects、image_pub：控制机器人语音播报和处理检测到的目标物体的发布节点。
7）clientIAT、serverIAT：用于实现 services/clients 通信机制，完成数据通信。
8）arObj：记录目标物体的标注框。
9）rgb_image：转换得到的 RGB 图像。
10）box_marker：用于标注检测到的目标物体的中间变量。

主要操作说明：

1）Speak(string inStr)：完成语音播报。
2）ProcCloudCB(const sensor_msgs::PointCloud2 &input)：对双目摄像头传来的

RGB-D 图像进行 PCL 点云分割并标注目标物体。

3）ProcColorCB(const sensor_msgs::ImageConstPtr &msg): 生成并保存目标物的 RGB 图像。

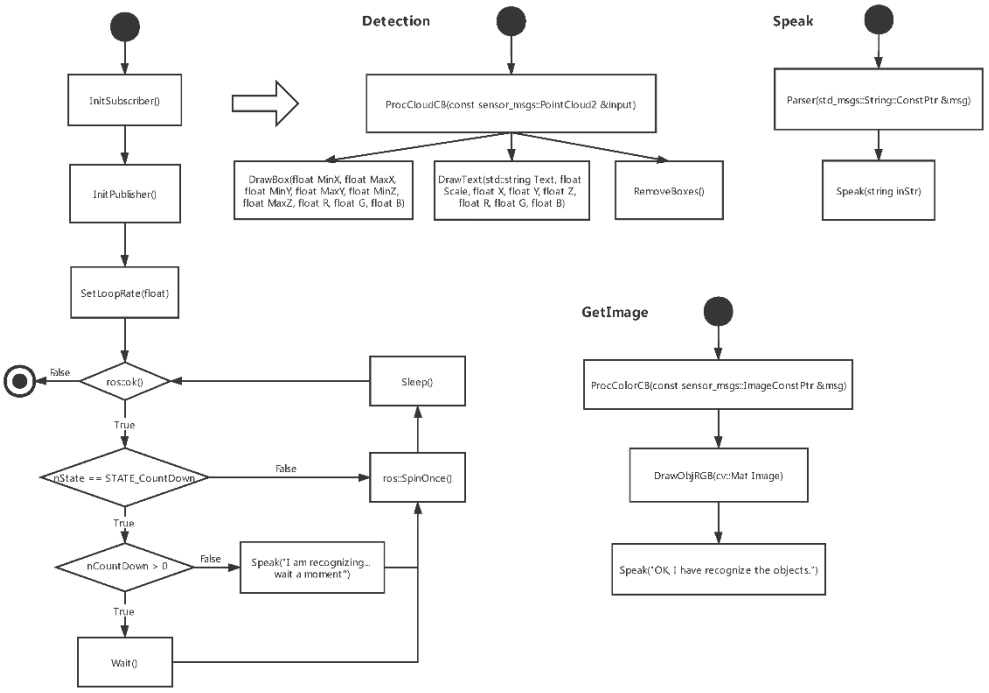

图 29 目标检测控制流程

控制流程：

- InitSubscriber()函数创建用于解析指令和处理 RGB-D 图像的订阅节点；InitPublisher ()创建用于控制机器人语音播报和处理检测到的目标物体的发布节点；SetLoopRate(float)函数设置发送消息的频率；nState 和 STATE_CountDown 是目标检测功能的状态变量，用以保证机器人在检测目标期间不会继续识别语音指令，以避免出错。

- 处理 RGB-D 图像主要依赖于 ProcCloudCB(const sensor_msgs::PointCloud2 &input)函数，其中实现了进行 PCL 点云分割和标注目标物体，其调用的 DrawBox(float MinX, float MaxX, float MinY, float MaxY, float MinZ, float MaxZ, float R, float G, float B)、DrawText(std::string Text, float Scale, float X, float Y, float Z, float R, float G, float B)、RemoveBoxes()分别实现框选目标物体、标注物品类别和取消框选的功能。

- 保存目标物的 RGB 图像主要依赖于 ProcColorCB(const sensor_msgs::ImageConstPtr &msg)函数，其调用的 DrawObjRGB(cv::Mat Image)

函数用于处理数据、生成像素值。

6.6 目标抓取

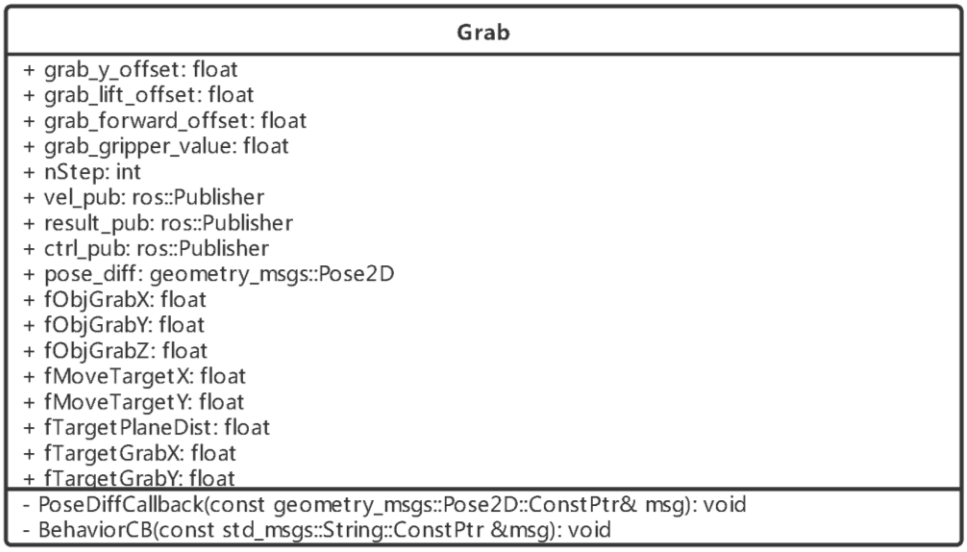

图 30 目标抓取模块类图

属性说明：

1）grab_y_offset：抓取前，对准物品，机器人的横向位移偏移量。
2）grab_lift_offset：手臂抬起高度的补偿偏移量。
3）grab_forward_offset：手臂抬起后，机器人向前抓取物品移动的位移偏移量。
4）grab_gripper_value：抓取物品时，手爪闭合后的手指间距。
5）nStep 是状态变量，用以控制机器人通过前后、左右移动及调节机械臂来锁定目标物体。
6）vel_pub、result_pub、ctrl_pub：控制机器人移动、语音播报和改变状态的发布节点。
7）pose_diff：机器人和目标物的距离。
8）fObjGrabX、fObjGrabY、fObjGrabZ：目标物体的三维坐标。
9）fMoveTargetX、fMoveTargetY：机器人为接近目标而需移动的距离。
10）fTargetPlaneDist：机器人与桌子之间的目标距离。
11）fTargetGrabX、fTargetGrabY：抓取时目标物品的 x、y 坐标。

操作说明：

1）PoseDiffCallback(const geometry_msgs::Pose2D::ConstPtr& msg)：计算机器人和目标物体的距离。
2）BehaviorCB(const std_msgs::String::ConstPtr &msg)：用于控制机器人的行为、切换状态。

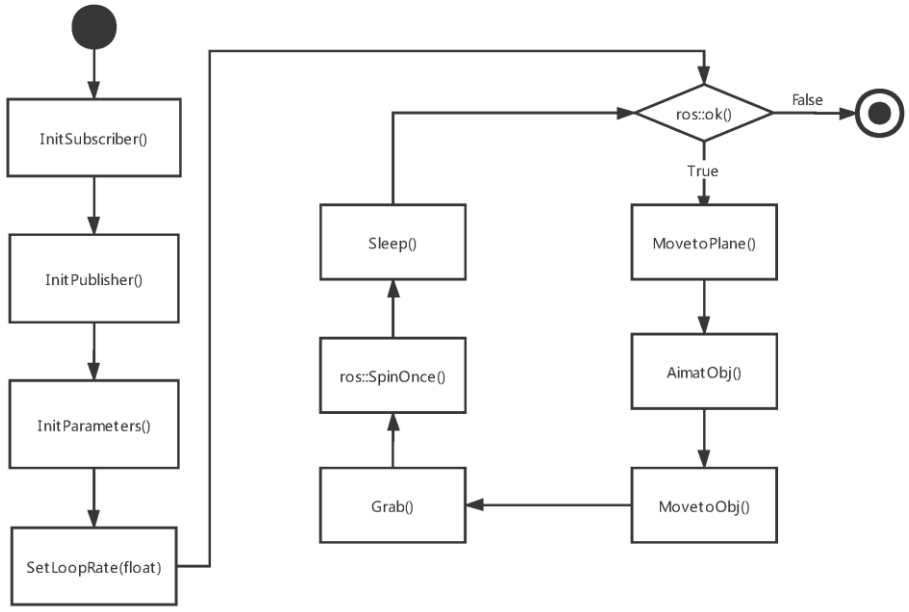

图 31 目标抓取控制流程

控制流程：

- 默认已通过目标识别模块获取到目标物体的三维坐标。

- InitSubscriber()函数创建用于解析指令、计算机器人/机械臂与目标物体距离以及控制机器人移动的订阅节点；InitPublisher ()创建用于控制机器人语音播报和移动的发布节点。

- InitParameters()用于初始化手臂抬起高度的补偿偏移量、抓取物品时手爪闭合后的手指间距等抓取参数；SetLoopRate(float)函数设置发送消息的频率。

- nStep 是状态变量，用以控制机器人通过前后、左右移动及调节机械臂来锁定目标物体，这三个状态分别对应 MovetoPlane()、AimatObj()和 MovetoObj()函数，同时控制机器人在抓取目标物体期间不再处理语音指令，以避免出错。

特殊说明：

基于本流程实现的功能是：机器人首先靠近桌面，而后根据机器人参数做摄像头与机械臂的坐标转换，使机械臂移动至目标处并抓取目标物体。

6.7 交互

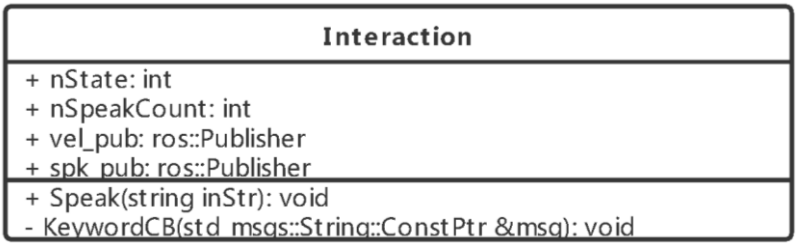

图 32 交互模块类图

属性说明：

1）State 和 STATE_SPEAK 是语音播报功能的状态变量，用以保证机器人在播报期间不会继续识别语音指令，以避免误识别。

2）vel_pub 和 spk_pub 是用于控制机器人语音播报和移动的发布节点。

操作说明：

1）Speak(string inStr)：完成语音播报。

2）KeywordCB(const std_msgs::String::ConstPtr &msg)：基于关键词来解析指令。

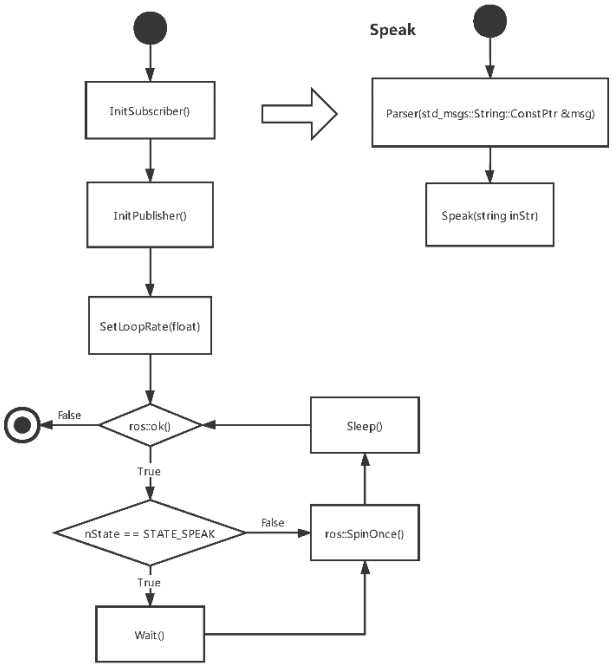

图 33 交互控制流程

控制流程：

- InitSubscriber()函数创建用于解析指令的订阅节点；InitPublisher ()创建用于控制机器人语音播报和移动的发布节点；SetLoopRate(float)函数设置发送消息的频率；nState 和 STATE_SPEAK 是语音播报功能的状态变量，用以保证机器人在播报期间不会继续识别语音指令，以避免误识别。

特殊说明：

支持的指令输入形式包括语音指令，Web 端远程指令和从控制面板获取的指令，机器人收到指令后对其进行解析并开始执行相应功能。执行过程中机器人会分别在收到指令后，开始移动时，到达目的地后，检测到目标后，抓取到目标后，故障时通过扬声器语音播报当前状态。

7. 运行与开发环境

7.1 运行环境

硬件环境：启智机器人（详情如下表）。

名称	数量	参数
主控器	1	Intel I3 处理器、4G 内存、128GSSD、触摸屏、键盘
激光雷达	1	360°无死角、最大距离 8 米
视觉传感器	1	Kinect 2
伺服电机模块	3	20W 伺服电机、内置驱动
轮子	3	3 个全向轮
电池	1	24V3.5AH 锂离子动力电池

软件环境：

ROS：kinetic 版本 ROS 系统，基于 Ubuntu 16.04。

软件包：启智机器人的源码包 wpb_home_bringup、wpb_home_behaviors、wbp_home_tutorials 、 wpbh_local_planner, 和 xfyun_waterplus 、waterplus_map_tools 等启智机器人扩展软件包。软件功能包括：URDF 模型描述、电机码盘里程计、IMU 姿态传感、三维立体视觉、SLAM 环境建图、自主定位导航、动态目标跟随、物品检测、人脸检测、传感器融合、语音识

别。

7.2 开发环境

硬件环境：

处理器：intel(R) Core(TM) i5-7400 CPU @ 3.00GHz 3.00GHz

内存(RAM)：8.00GB

系统类型：64 位操作系统，基于 x64 的处理器

软件环境：

操作系统：Ubuntu 16.04

IDE： RoboWare Studio

8. 需求可追踪性说明

8.1 功能性需求

本节将主要从设计的角度论述本文档中体系结构设计的内容和 SRS-软件需求规格说明书中功能/非功能需求的对应关系。

8.1.1 机器人建立环境地图

1. 设计方案：在导航模块中加入绘图单元，绘制地图并存储地图。
2. 模块：绘图单元。

8.1.2 机器人的自主导航

1. 设计方案：导航模块基于总控模块传来的位置信息、内部存储的地图信息以及知识源中的路径规划算法和动态避障算法，调用定位单元、运动控制单元和路径选择单元提供自主导航功能。
2. 模块：定位单元、运动控制单元、路径选择单元。

8.1.3 机器人的目标检测及抓取

1. 设计方案：抓取模块基于总控模块传来的目标物体信息和知识源中的目标识别算法，调用机械臂控制单元，提供目标抓取功能。
2. 模块：机械臂控制单元。

8.1.4 机器人的语音交互

1. 设计方案：Web 端发来的经过预处理的语音指令或机器人搭载的面阵麦克风收集到的语音指令会在总控模块中被编码为机器人的行动指令；机器人在完成任务后会发出语音提示，播报任务完成情况。二者结合，即完成了语音交互。
2. 模块：语音交互模块。

8.2 非功能需求

本节将主要从实现的角度论述本文档中体系结构设计的内容和 SRS-软件需求规格说明书中功能/非功能需求的对应关系。

8.2.1 性能需求

1. 优化避障算法及语音识别算法，提高系统实时响应速率。
2. 优化 SLAM 建图算法，提高建图准确率，确保绕场一周即可建立地图模型。
3. 优化路径选择算法，避免出现摆荡，确保机器人平稳移动。
4. 优化目标识别及抓取算法，提高物体识别准确率及抓取成功率。
5. 优化语音的预处理算法，扩大语料库，提高语音识别准确率。
6. 降低算法复杂度，避免系统长时间高负荷运转。

8.2.2 易用性需求

1. 用户界面简洁易用，操作简单易懂。
2. 系统在用户首次操作时给出操作指南，在用户长时间未选定模式时给出操作建议。

8.2.3 可靠性需求

1. 优化错误处理机制，使系统具备一定的处理错误操作的容错能力。
2. 加入数据保护机制及复位机制，使系统即使在发生严重故障时，也依然可以在重启后恢复正常状态。

8.2.4 可扩展性需求

1. 在 web 端和总控模块间采用分层结构模式（客户/服务器风格），提高了室内智能服务型机器人的可扩展性。
2. 系统内部模块化、层次化清晰，便于新增、修改和删除系统功能。

8.2.5 可维护性需求

系统支持维护检修，可以通过更换硬件和修复软件 bug 的方式解决故障。