**嵌入式软件工程课程项目总结报告**

1. **项目名称**

简易机器人

1. **项目概述**

**2.1目标：**

通过用户界面的操作和语音指令，实现机器人手动控制运行，自动简易避障运行，紧急制动，自动抓取物品功能。

**2.2功能需求：**

用户界面操作：用户通过手机用户界面手动控制机器人运动进行建图或者调整位置；

添加导航点：通过语音指令添加导航点；

跟随功能：通过用户界面发出指令机器人开始跟随用户，通过语音指令命令机器人停止跟随；

自动前往目标点抓取物品功能：用户通过语音指令命令机器人开始前往目标点抓取桌面物品并自动返回原点将物品交给用户；

简易自动避障运行：自动运行途中遇见障碍物自动一直运动，等待障碍物消失后继续运行。

自动路径规划：根据预设导航点和建好地图规划可达路径

**2.3非功能需求：**

系统运行性能：系统保证机器人运行高效，APP控制高效

系统可靠性：系统需保证保证起始点与目标点可达，系统需要有紧急暂停功能，系统要有异常处理功能

系统易用性：用户操作界面简明，用户可以轻松使用，利用App控制机器人所有功能

系统安全性：保证系统运行环境必须满足说明书中的安全条件，机器人运行速度不能过高，要设计紧急制动功能防止与障碍物相碰

系统可扩展性：机器人设备上机器人基本功能可以进行扩展，App对机器人的控制功能可以进行扩展

**2.4完成的功能：**

用户界面控制，语音控制，定点物品抓取，利用手机App手动建图，简易避障功能，跟随功能，自动前往目标点抓取功能。完成了设计阶段提出的所有功能需求。

1. **需求分析阶段总结**

**3.1分析策略：**

根据可能的应用场景分析可能的功能需求及非功能需求。

然后根据机器人已有的基础功能调整。

**3.2面临的困难：**

对机器人的具体情况如机器人已有的基础功能不了解，部分功能需求分析为不必要工作；

对部分功能的实现难度估计不足，导致后续开发出现困难。

**3.3后续修改：**

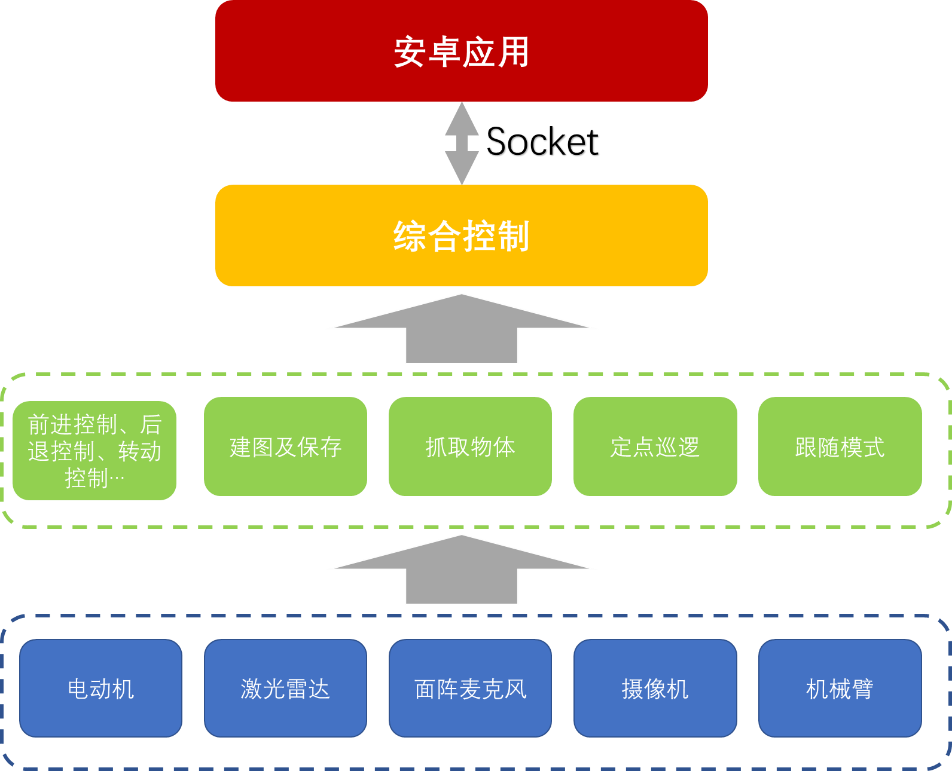
某些功能是机器人已经实现的基础功能，删除这部分功能需求；

部分功能实现难度过大，简化这部分功能需求。

1. **设计阶段总结**

**4.1系统架构概述**

本系统采用安卓应用和语音识别两种方式接收用户的命令，并作出相应的动作，完成抓取物体、定点巡逻等功能。系统可以分为四层，如图所示，



用户通过安卓应用对机器人下达命令。机器人端的综合控制程序收到命令后，调用相应的功能模块执行相应的功能。功能模块为ROS launch文件，每一个launch文件执行一个对应的动作，这些ROS launch文件通过编写的C++代码实现对电动机、激光雷达等硬件的控制，完成相应的功能。

**4.2设计到需求的追踪关系**

在需求分析中，确定的需求主要有建图、目标检测与抓取和定点巡逻。对于每一项需求，我们使用C++代码实现特定的功能，并使用一个ROS launch文件封装为一个单独的模块，随后使用综合控制软件功能模块进行调用，实现了所有的需求。

**4.3设计阶段发现的问题**

在设计阶段进行时，我们缺乏对于机器人的具体情况以及已有软件系统的认识，导致我们最初设计的重心放在了如何与底层硬件交互的方向上。在进入开发阶段后，对机器人的具体情况有了了解，将设计的中心转移到了如何利用现有的模块，即在现有模块的基础上进行包装。

1. **迭代1阶段总结**

**5.1概述**

第一次迭代主要完成任务为完成机器人的基本功能模块，包含两大部分：运动部分和抓取部分。运动部分向下拆分为初始建图，接收APP的指令移动，遇障急停，设置导航点，规划路径并前往导航点；抓取部分向下拆分为平面识别，物体识别，物体抓取，放开物体。第一次迭代主要任务为可以通过手动在机载电脑上操作，实现两个基本功能的下分小功能。

**5.2任务分配**

|  |  |
| --- | --- |
| **成员名称** | **具体任务** |
| 李嘉业 | 平面识别，物体识别以及物体抓取功能的实现；  物体抓取功能参数的调整； |
| 母江涛 | 通过调取雷达数据，进行机器人遇障急停的尝试；  进行每个模块所需的测试； |
| 王润安 | 调配机载平台环境；  通过机载平台实现机器人四个方向的移动；  机器人导航点的设置； |
| 张弩 | 调配机载平台环境；  通过机载平台实现机器人四个方向的移动；  机器人导航点的设置； |

**5.3代码提交**

Github

历次尝试与弃置代码中：

python控制；添加导航点尝试（弃置）；物体识别与抓取模块（弃置）；脚本控制尝试（弃置）；自动避障尝试；test；team-108；

Team108代码-终板中：

Src；

为本次迭代提交内容；

**5.4测试**

**测试存在些许问题的样例用\*标出，会在测试问题部分详述**

|  |  |  |
| --- | --- | --- |
| **测试用例编号** | **测试用例目的** | **测试结果** |
| T108TC01 | 测试房间地图的生成与保存 | 成功生成地图文件，且已移至指定路径； |
| \*T108TC06 | 测试机器人移动功能，包括四个方向的移动，两个方向的旋转； | 功能正常，机器人可以正常进行四个方向的移动和两个方向的旋转； |
| \*T108TC07 | 测试机器人识别平台 | 机器人在特定情况下可识别平台的存在； |
| \*T108TC08 | 测试机器人识别物体 | 机器人在特定情况下可识别物体的三相坐标大小及位置； |
| \*T108TC09 | 测试机器人抓取物体 | 在正确识别物体的前提下，机器人可正确抓取物体 |

**5.5问题管理**

本阶段项目出现的问题主要通过Github中issue平台进行管理。问题发布情况：

抓取不准确；

原始定点无法使用，需要自行设计；

尝试选择物体抓取；

提供的物体抓取程序到底抓的是那个物体；

逻辑调用命令行；

**5.6评审问题与改进措施**

|  |  |
| --- | --- |
| **问题** | **措施** |
| 看板功能未建立 | 评审当天立即建立看板系统，并且将之前的工作加入看板系统中； |
| Issue体系未建立 | 评审当天立即建立issue系统，并且将之前的工作加入issue系统中； |
| 单元化功能测试不完善 | 立即进行完善的单元化功能测试 |

1. **迭代2阶段总结**

**6.1概述**

第二次迭代主要任务为完成APP对机器人各项功能的控制将第一次迭代测试阶段所发现的问题解决。具体任务有：APP与机器人的通信；APP控制机器人四个方向的移动以及两个方向的旋转；APP控制机器人设置导航点；APP控制机器人开始抓取；

**6.2任务分配**

|  |  |
| --- | --- |
| **成员名称** | **具体任务** |
| 李嘉业 | 解决并完善第一次迭代中出现的问题；  进行相应的单元测试；  APP控制机器人抓取； |
| 母江涛 | 继续进行遇障急停的尝试； |
| 王润安 | APP与机器人通信；  APP控制机器人移动与旋转；  APP控制机器人设置导航点； |
| 张弩 | APP与机器人通信；  APP控制机器人移动与旋转；  APP控制机器人设置导航点； |

**6.3代码提交**

Github历次尝试与弃置代码中：

无线通信测试1.0，2.0，3.0；APP与server端代码（终板）

**6.4测试**

|  |  |  |
| --- | --- | --- |
| **测试用例编号** | **测试用例目的** | **测试结果** |
| \*T108TC10 | 测试APP控制机器人移动功能，包括四个方向的移动，两个方向的旋转； | 功能正常，机器人可以正常进行四个方向的移动和两个方向的旋转； |
| T108TC11 | 测试语音控制机器人设置导航点 | 机器人成功设置导航点； |
| \*T108TC12 | APP控制测试机器人抓取物体 | 在正确识别物体的前提下，机器人可正确抓取物体； |

**6.5问题管理**

本阶段的issue有：

原始定点无法使用，需要自行设计；

关于Topic（话题）的问题；

完成建图后，返回建图起始点；

将机器人获取的信息传输给APP；

不同用户情景下的脚本代码；

导出检测到的物体图片；

**6.6评审问题与改进措施**

|  |  |
| --- | --- |
| **问题** | **措施** |
| 看板中内容未按阶段划分； | 每个看板问题添加阶段性标签； |

1. **迭代3阶段总结**

**7.1概述**

第三次迭代主要的任务有：将之前完成的基本功能组合形成可供用户使用的流程；对用户使用流程以及之前的基本功能进行系统化测试；APP前端界面的完善。具体任务有：跟随模式的搭建；手动模式中抓取流程的搭建；手动模式中巡航模式的搭建；APP前端界面完善；对各模块的测试。

**7.2任务分配**

|  |  |
| --- | --- |
| **成员名称** | **具体任务** |
| 李嘉业 | 手动模式中抓取流程的搭建；  手动模式中巡航模式的搭建；  抓取流程的测试； |
| 母江涛 | 跟随模式的搭建； |
| 王润安 | 手动模式中抓取流程的搭建；  手动模式中巡航模式的搭建；  巡航流程的测试； |
| 张弩 | 手动模式中抓取流程的搭建；  手动模式中巡航模式的搭建；  APP前端的设计与改进；  单元测试； |

**7.3代码提交**

除之前提交代码外的其他代码，包括所有终板代码。

**7.4测试**

|  |  |  |
| --- | --- | --- |
| **测试用例编号** | **测试用例目的** | **测试结果** |
| T108TC03 | 抓取物体，机器人抓取物体，并带着物体回到出发点，把物体松开。 | 得到应有输出和效果 |
| T108TC104-1 | 机器人根据设定的导航点进行定点巡逻。 | 得到应有的输出和效果 |
| T108TC104-2 | 机器人停止运动，并在移开障碍物后继续运动。 | 成功完成避障急停 |
| T108TC05 | 对机器人的跟随流程进行测试 | 跟随状态良好 |

**7.5问题管理**

跟随测试；定点巡航；抓取测试；移动测试；建图测试；

参照指导书例程实现的前后左右运动可能无法停止；

使用OS命令行调用过程中由于非阻塞式执行导致结果不可控；

AMCL与MOVE\_BASE的调用方式；

紧急避障；

避障紧急停止；

基本样例测试；

**7.6评审出现问题与改进措施**

评审的方式为课上答辩，故没有类似于之前的问题。

1. **测试总结**

**8.1从设计单位角度出发的测试策略**

从设计单位的角度出发，对系统中的各个接口、函数进行单元测试，测试内容如表格所示

|  |  |
| --- | --- |
| **被测试的接口、函数名** | **概述** |
| getinstance() | 返回一个唯一的tcp服务线程实例 |
| connectSend(String str) | 新建一个socket，与服务器端建立连接，并发送字符串str，启动收听线程 |
| startServerReplyListener() | 新建一个线程，接收服务器发送的消息 |
| run() | 启动tcp服务线程，从UI进程接收消息并作出相应处理 |
| str2shell2(str) | 将输入的字符串转换为机器人的指令并执行 |
| tcplink() | 启动服务器，监听消息，并将收到的消息发送至移动端，表示确认 |
| minimal.launch | 启动相关节点，为控制移动做准备 |
| hector\_mapping.launch | 开始建图 |
| map\_saver | 保存构建完毕的地图文件 |
| go\_forward | 控制机器人以0.1m/s的速度前进1s |
| go\_backward | 控制机器人以0.1m/s的速度后退1s |
| go\_left | 控制机器人以0.1m/s的速度向左行走1s |
| go\_right | 控制机器人以0.1m/s的速度向右行走1s |
| turn\_left | 控制机器人向左转动 |
| turn\_right | 控制机器人向右转动 |
| script\_add1.launch | 机器人前往指定地点，收到语音命令后，抓取物体，并返回起点 |
| script\_add2.launch | 机器人受用户控制行走至目的地点，记录地点，重复两次，随后返回起点。然后前往地点一，地点二，返回起点 |
| shopping.launch | 启动跟随模式 |

首先针对各个函数、接口进行单元测试是十分必要的。如果没有单元测试而直接进行集成测试，一旦出现问题后，由于系统的复杂度比较高，定位问题比较困难，也不利于快速解决问题。进行单元测试，测试比较简单直接，能够发现各个函数实现的是否正确，便于快速确定出现问题的位置。在进行集成测试之前，确保每一个功能部件基本正确，为后续测试打好基础。

**8.2从需求覆盖角度出发的测试策略**

从需求覆盖的角度，对系统进行测试，主要的测试内容如表格所示

|  |  |
| --- | --- |
| **测试用例序号** | **测试对应的需求** |
| T108TC01 | 生成房间的地图 |
| T108TC02 | 完成机器人的移动功能，包括前进、后退、左移、右移、左转、右转 |
| T108TC03 | 机器人抓取物体，并带着物体回到出发点，将物体松开 |
| T108TC04-1 | 机器人根据设定的导航点进行定点巡逻 |
| T108TC04-2 | 机器人遇到障碍物后停止运动，并在移开障碍物后继续运动 |
| T108TC05 | 机器人拿到指定的物体并返回 |

针对提出的需求进行测试，将系统的各个部分整合起来，确保系统能够正常完成提出的各种需求。仅仅进行单元测试是不够的，将各个部分组合起来后可能出现一些问题，从需求覆盖的角度进行测试，能够模拟用户使用的过程，降低用户使用的过程中出现问题的概率。

所有测试综合起来，我们的安卓应用端代码覆盖率98%，分支覆盖率92%；服务器端代码覆盖率达到96%。

**8.3测试中发现的问题**

在测试过程中，发现的问题主要存在于物体抓取部分、运动控制部分以及跟随部分。

在物体抓取过程中，如果桌面上物体较多，机器人可能无法正确识别被抓取物体，导致抓取失败。此外，经测试发现机器人的物体识别算法在识别桌面的平面时，仅能识别前方带挡板的桌面，面对其他种类的桌子时无法正确识别桌面。

在运动控制的过程中，机器人在执行向右移动时会向右后方偏移，可能是机器人的轮轴出现磨损造成的。此外，机器人在执行前往一个点并返回的过程中，机器人返回的位置与实际的起点位置存在一个小的偏移，直线距离在0~4cm之间。在项目开发初期，可能出现机器人前进后无法停止的情况，在改进运动控制算法后，这个问题得到了解决。

在跟随模式下，如果用户行走的过快，机器人可能丢失跟随的目标而原地旋转。

1. **团队协同总结**

**9.1分工模式**

本团队采用按任务模块分工的方式。即在开发计划阶段就把总目标划分为几个并行的任务模块，再将每个任务模块分发给个人。在之后的阶段里会把该阶段的任务按照总任务的划分分类在拆分为同样并行的任务。这样在保证每位小组成员在每个阶段中都有任务的同时，是每个人都分配到与自己之前所完成任务相关的任务，可以减少不必要的准备工作。

以三次迭代的任务分工为例。组员A，B在第一次迭代阶段分配到控制机器人四个方向运动功能的实现。则在第二次迭代，他们会分配到APP控制机器人四个方向运动的任务，这样在熟悉第一次完成的功能接口的前提下，可以更方便的实现APP的控制功能。

而对于文档撰写部分，依旧是每位成员撰写与自己实现部分技术相关的文档，避免信息传达的麻烦，减少文档错误。

**9.2成员分工**

**名称按拼音排序，没有排名先后**

|  |  |
| --- | --- |
| **成员名称** | **具体任务** |
| 李嘉业 | 平面识别，物体识别以及物体抓取功能的实现；  物体抓取功能参数的调整；  解决并完善第一次迭代中出现的问题；  进行相应的单元测试；  APP控制机器人抓取；  手动模式中抓取流程的搭建；  手动模式中巡航模式的搭建；  抓取流程的测试； |
| 母江涛 | 通过调取雷达数据，进行机器人遇障急停的尝试；  进行每个模块所需的测试；  继续进行遇障急停的尝试；  跟随模式的搭建； |
| 王润安 | 调配机载平台环境；  通过机载平台实现机器人四个方向的移动；  机器人导航点的设置；  APP与机器人通信；  APP控制机器人移动与旋转；  APP控制机器人设置导航点；  手动模式中抓取流程的搭建；  手动模式中巡航模式的搭建；  巡航流程的测试； |
| 张弩 | 调配机载平台环境；  通过机载平台实现机器人四个方向的移动；  机器人导航点的设置；  APP与机器人通信；  APP控制机器人移动与旋转；  APP控制机器人设置导航点；  手动模式中抓取流程的搭建；  手动模式中巡航模式的搭建；  APP前端的设计与改进；  单元测试； |

**9.3协同效果评价**

在整个项目过程中，小组各成员都积极配合与合作，极大的降低了项目推进的复杂度。小组成员在任务分配过程时的积极参与确保了任务分配的合理性，并且使后续任务的推进与完成得以顺利进行。总体而言，本小组协同效果较佳。

**9.4反思**

尽管本小组在最后项目结束时取得了不错的协同效果，但是在项目推进的过程中也出现了一些问题，反思如下：

1. 初期的小组协同主要在线下进行，虽然有每次会议记录的支持，但是并没有达到课程要求的标准。在教师提出问题后改正。
2. 在项目大后期出现了版本管理的问题，出现一次严重影响项目进度的版本问题。本小组立即进行修改，在Github中建立历史文件与终板文件的分类。较好的解决了这个问题。
3. **感受与建议**

这次课程总体做下来还是感受蛮多的，下面的感受和建议来自于全组成员的讨论与总结。

**10.1感受**

**可能是因为课程初次设置机器人项目的原因，感觉课程组对机器人这方面技术层面的支持还是有些不太够。或者说有些滞后，小组没有从一开始就得到一手的信息，导致前部的设计和需求等和后期的实现有些脱节。增加了项目的困难性。**

在**设计之初面临的巨大挑战就是我们的设计与机器人之间有较大的差别。**我们设想中的机器人是什么都没有的一个框架，甚至需要我们自行去写底层的驱动。而我们看到的机器人是一个已经具备了一定功能的比较完备的机器人。这一定程度上减轻了我们编写程序的困难，但同时**增加的困难是我们不得不花时间去了解现有的一些模块的调用方式以及现有功能的实现程度。ROS系统过于庞大，**直至现在，完成了功能需求的我们，对一些其他我们没有用到的系统部分仍然是一知半解。

对系统了解程度不够这个弊病一直存在在整个开发周期中。例如我们一开始设想的物体识别是利用Kinect的摄像头去获取物体的照片，然后根据照片实现基于机器学习的物体识别，没想到底层只提供了照相的接口而并没有实现照相的算法。于是我们被迫放弃了这个方案。

在发现了这个问题之后我们转换思维，从现有的例程中去学习ROS系统，从现有的例程中去改造例程使之成为我们需要的功能。**我们后半程的开发过程中，我们逐渐将ROS视为一个黑盒子，试图直接在我们写的程序和机器人的表现中建立直接的联系。这就大大加大了我们调试的难度。**

在项目的前期开发过程中，机器人环境的配置也是一大难题。所有的文件不可能放入即用，需要书写大量的Make以及launch文件，无疑也为该项目的实现增加了难度。当然这也是嵌入式系统必须要克服的问题。

**当然这次课程着实让我感受到了管理一个长期且多人项目的困难性，繁多的版本控制，预估任务时间的困难性，以及任务预期和完成之间的时间差都在影响和威胁着项目的进行。**整个项目完成下来感觉对管理能力的一次提升。

**10.2建议**

1. 应在课程开始时就使各小组尽早的接触到机器人，使大家前期的设计与需求有目的性。可以按照真实情况撰写，而不是依靠设想。
2. 如果有条件的话，建议购置更多的机器人。过少的机器人不但会使各组进度缓慢，而且可能导致组间误修改的现象出现。
3. 可以将每一届的成果和技术总结起来，这样可以为下一届的同学节省处理底层技术的时间。给他们提供更大的创新空间。
4. 老师们是否可以把课上讲解的项目管理方法能与这种小型的课程团队情况结合起来，在传统管理理论的基础上提供更适于小团队管理的理论。