课 程 报 告

机器人软件工程

学　　院：自动化学院

专　　业：机器人工程

姓　　名：赵永强

学　　号：08117102

指导教师：谈英姿

2020 年 4 月

目 录

1 综合实验报告

1.1 机器人需求建模（含分析过程，并给出机器人行为模型）

- **目的：**主要是对《机器人软件工程》综合实验中对超市机器人的功能需求进行说明。
- **范围：**包含了对超市机器人的功能需求分析以及机器人的行为模型（状态图和顺序图）。
- **项目介绍：**如图 1 所示，超市中 A,B,C,D 四个货架上各有 12 个货窗，每个货窗里最多可以摆放 1 件货物，现每个货架上都有至少 3 个货窗缺少货物。要求需要设计并制作一台自主超市机器人。机器人从场地的红色“起点区”出发，完成货物的补货操作，最后回到起点。

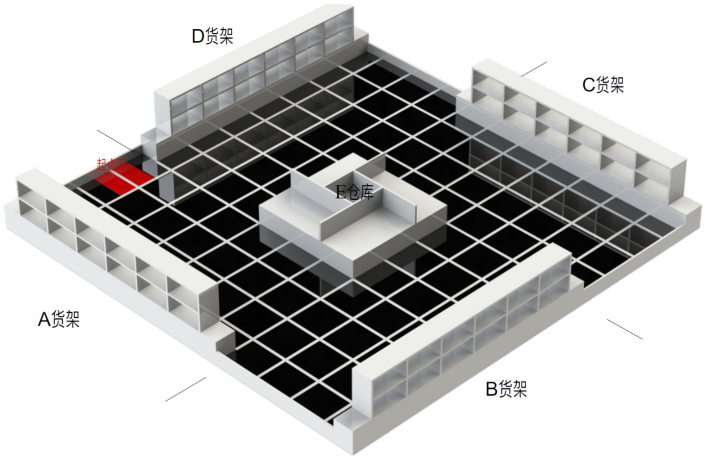

图 1: 超市机器人场地示意图

- **功能需求：**本项目是纯软件项目，故所实现的功能即为软件所具有的功能，即移动、抓取、视觉识别、自主定位以及避障等功能。
- **移动功能：**使机器人可以在超市中移动，通过轮子来实现。
- **抓取功能：**抓取获取，以实现取货、放货，通过机器臂和末端机械爪实现。
- **视觉识别：**识别货物和找寻空货窗，使用摄像机来实现。
- **自主定位：**确定机器人在超市中的位置，使用激光雷达实现
- **避障：**避障：当两台机器人同时补货时，以及当机器人靠近货架、墙壁等情况时，及时躲避，可以使用激光雷达实现。
- **标准符合性：**本节详细说明需求所采用的标准或规范的来源。如果项目采用了国际标准，应该说明国际标准及项目与标准的偏离情况。本系统需求采用国家标准 GB8567-88；国家标准 GB8567-88 ： 对所开发软件的功能、性能、用户界面及运行环境等作出详细的说明。它是在用户与开发人员双方对软件需求取得共同理解并达成协议的条件下编写的，也是实施开发工作的基础。
- **流程图：**状态流程图和顺序流程图如下所示

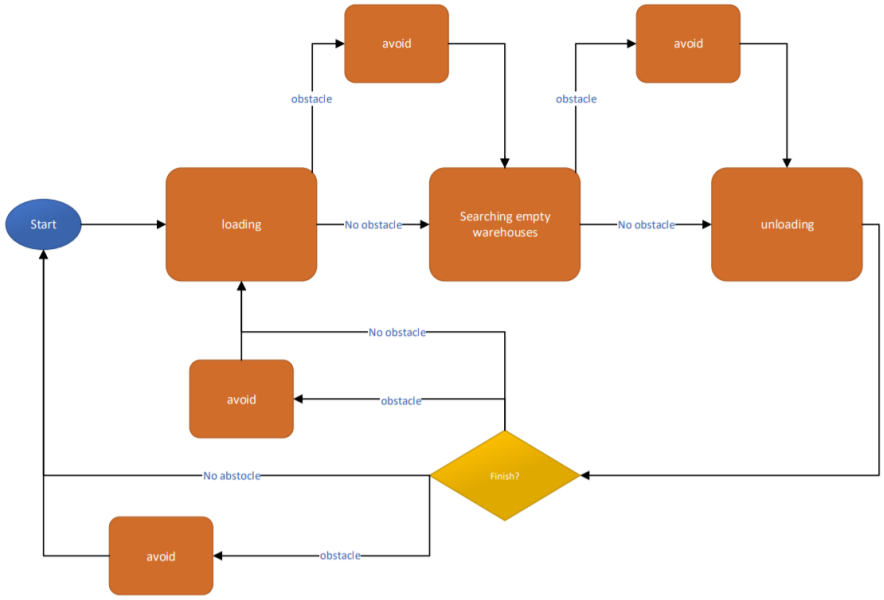

图 2: 状态流程图

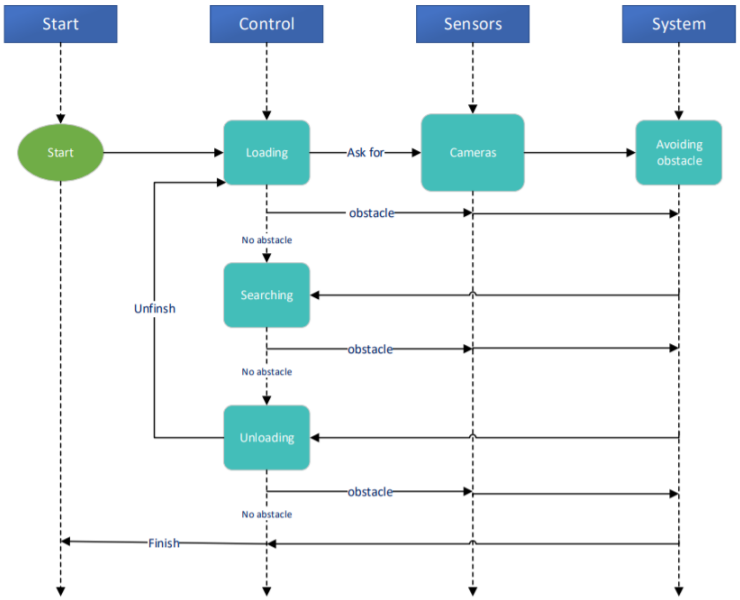

图 3: 顺序流程图

1.2 机器人模型设计

根据以上小节功能需求，设计如下机器人模型：

- **移动：**考虑到底盘的灵活性与整体承重性能，还有超市的平坦规则的地形，最后决定使用全向的麦克纳姆轮作为移动底盘，目的是使其更容易进行抓取。
- **机械臂：**通过机器臂带动末端执行器抓取货物。观察到仿真超市环境中，货仓区比较复杂，且内层还有货物，机器人在抓取的过程中，可能需要在到达作业点的过程中还需要避开其他的货物，因此本次仿真机器人选取了七个自由度机械臂。

- **末端执行器：**考量到货物的体积，正面和侧面的面积大小，最后选择了一个具有一个自由度的，接触面积比较大的二指机械手爪。
- **定位：**本次仿真实验使用了带有全局坐标信息的 GPS 定位设备，随时可获取机器人相对仿真世界的位置，从而控制机器人的移动和抓取。
- **机器视觉：**经商量，为了更加方便、精确地抓取物品，最后采用了彩色深度摄像机，在识别物品的同时，判别物体和机器人手爪之间的距离。
- **避障：**根据超市的环境，避障决定采用激光雷达，安装在机器人前端进行 180 度扫描。

1.3 基于 Webots 的机器人模型搭建

根据以上设计分析，决定通过改造 Kuka 的 proto 实现所需要的功能。

- **抓取部分：**通过机器臂带动末端执行器抓取货物。观察到仿真超市环境中，货仓区比较复杂，且内层还有货物，机器人在抓取的过程中，可能需要在到达作业点的过程中还需要避开其他的货物，因此本次仿真机器人选取了七个自由度机械臂。

```
Arm5Mesh {
SliderJoint {
  jointParameters JointParameters {
    axis 0 1 0
  }
  device [
    LinearMotor {
      name "finger1"
      maxPosition 0.5
      maxForce 200
    }
    PositionSensor {
      name "finger1sensor"
    }
  ]}}
```

- **麦克纳姆轮：**在 youbot 机器人底盘的基础上，修改了麦轮的部分参数，例如与地面的摩擦力、颜色等等。

```
DEF WHEEL1 InteriorWheel {
  translation 0.228 -0.158 -0.055
  anchor 0.228 -0.158 -0.055
  name "wheel1"
  sensorName "wheel1sensor"
}
DEF WHEEL2 ExteriorWheel {
  translation 0.228 0.158 -0.055
  anchor 0.228 0.158 -0.055
  name "wheel2"
  sensorName "wheel2sensor"
}
DEF WHEEL3 ExteriorWheel {
  translation -0.228 -0.158 -0.055
  anchor -0.228 -0.158 -0.055
  name "wheel3"
  sensorName "wheel3sensor"
}
```

```
DEF WHEEL4 InteriorWheel {
  translation -0.228 0.158 -0.055
  anchor -0.228 0.158 -0.055
  name "wheel4"
  sensorName "wheel4sensor"
}
```

- **摄像头和 GPS 等：**将 robot 节点转化为 base 节点后，在机器人自带的拓展节点中增加了摄像头和 GPS，如图 4。

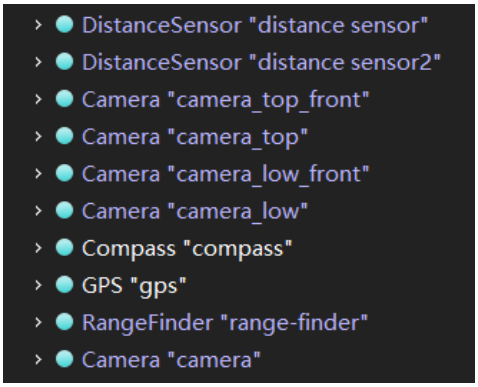

图 4: 设备图

- **整体模型：**通过合并以上功能，总体机器人模型如图 5：

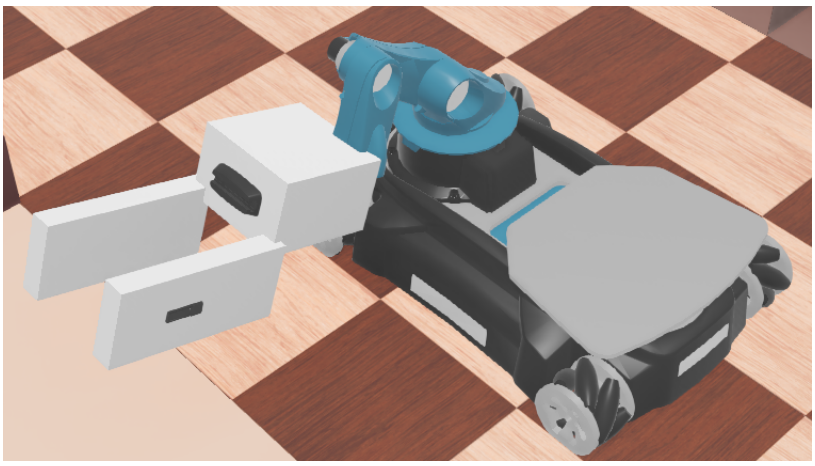

图 5: 机器人模型图

1.4 机器人控制器设计与实现

本次控制器设计采取了两种不同的方案，区别在于，方案一不依赖于货柜的标定而方案二依赖于货柜的标定，且方案二有优化路径的策略。

1.4.1 方案一

- **取货物：**通过深度摄像头识别物品，并测出机器人与物体之间的相对位置和距离，当物体在摄像头中的面积和距离摄像头的距离以及在摄像头中的相对位置符合一定阈值时，进行抓取操作

- **寻找货柜：**将两个 Kinect 相机安置于机器人右侧，上下各一个，对准上下层的货柜，在逆时针巡逻的过程中，扫描货柜中的货物，如发现了有与先在同类货物，且货柜中，此同类货物的前或后方的货架为空，则将货物放置在对应的货柜中，否则一直巡逻。
- **优缺点分析：**此控制器的优点在于，无需对货柜的位置进行标定，从而即使超市某些货物更换货柜，不会影响机器人的工作效果。缺点在于本法是依靠探测同类物品来进行货物放置的，因此，如出现全空货柜的情况，本法将不适用

1.4.2 方案二

- **取货物：**通过深度摄像头识别物品，并测出机器人与物体之间的相对位置和距离，当物体在摄像头中的面积和距离摄像头的距离以及在摄像头中的相对位置符合一定阈值时，进行抓取操作
- **寻找货柜：**将两个彩色摄像机安置于机器人右侧，上下各一个，对准上下层的货柜，在逆时针巡逻的过程中，扫描货柜以及其中的货物。在取得货物后，有两种情况，一是货物就在对应的货柜旁边，此时机器人将倒退回货柜的起点处；二是对应的货柜不在取得的货物旁边，此时机器人继续巡逻去寻找对应货柜。在找到对应的货柜之后，机器人右侧的相机检测第一个格子是否为空，如果是空的则放置货物，否则前进一个格子的距离，对第二个进行检测，方法同第一个格子，以此类推，直到货物放置成功。
- **优缺点分析：**此控制器的优点在于，对于货柜上出现的各种情况都可以适应自如，不会出现第一种方案的全空货柜放置失败的情况。缺点在于需要对各个货柜的起点位置进行标定，且需要对各种商品和其对应的货柜之间的位置关系进行一一处理，增加了代码量。

1.4.3 避障实现

避障实现主要利用激光雷达进行实时检测，根据不同情况，有三种不同的策略。

- **前方有机器人且待取或待放的物体就在前方机器人的位置处：**此时即使超车也无法取或放物品，或者逆时针方向行驶很大可能更加耗时，所以策略为等待

图 6: 等待

- **前方有机器人且待取或待放的物体在该机器人前方较近处：**这时候选择加速，从后超车去放置物品，从而很大层度上节省了时间。

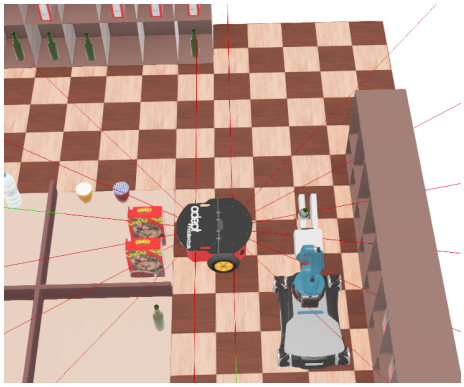

图 7: 超车

- **前方有机器人且待取或待放的物体在该机器人前方较远处：**策略为后退，逆时针走到需要取或放的物品处，是否后退的阈值取决于判断自身逆时针走到目标点时，对方机器人尚未到达该点。如未到达则后退，否则超车。

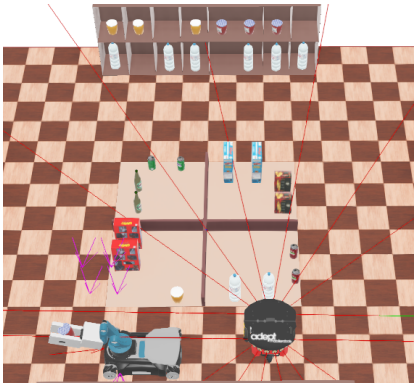

图 8: 后退

2 机器人软件开发

2.1 机器人软件开发特点

- 机器人的软件通常是嵌入式，具有并发性、实时性，分布式，数据密集型特点，并且必须满足特定要求，例如安全性，可靠性和容错性；
- 在机器人技术中，巨大的代码库（库，中间件等）共存而不具有互操作性，并且每个工具都各自独特性；
- 虽然重用现有和成熟的软件构件可以减少开发时间和成本，提高稳健性，但到目前为止还不大可行；
- 由于开放式的真实环境，在资源分配方面，设计时最优性和运行时最优性之间总是存在偏差。因此，需要在运行时动态资源分配；
- 运行时，系统配置需要根据当前情况进行更改，包括优先分配资源到活动、构件激活以及构件之间连接的更改；
- 运行时，机器人必须分析并决定最合适的配置。例如，如果当前处理器负载不允许以最高质量级别运行导航构件，则构件应配置为较低级别的导航质量。

2.2 超市机器人软件开发及其质量管理

2.2.1 软件开发

本次超市机器人软件开发是完全基于 webots 的仿真开发。在开发过程中，机器人被逐步定义和丰富，直到最终变得可执行。

在系统设计时根据各种技术问题进行了子系统划分，将系统分解为机械系统，传感器执行器，控制和算法软件系统。并且在开发过程中，重用了部分传感器和代码，减少了开发时间和成本，提高了稳健性。

2.2.2 质量管理

为机器人软件产品的质量提高可视性，明确哪些地方有质量问题，便于改进，提高产品质量。

在项目进行中，通过 github 管理版本，如下图所示

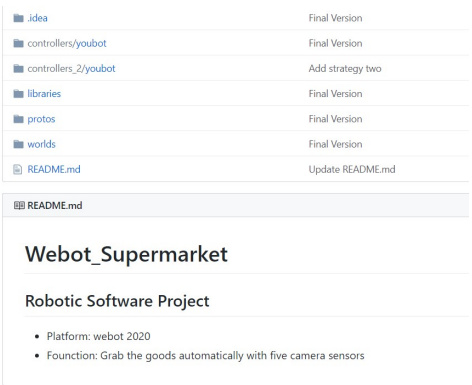

图 9: github 管理

- **功能性：**在货物随机以及半空货架的情况下，方案一和方案二均实现了取货、放货、视觉识别、自主定位以及动态避障的功能。
- **易用性：**控制程序有效性高、功能指示较为容易辨认、运行状态以及反馈信息清晰。在控制器中，各个模块均可以有效的完成相应的功能，且通过输出功能对机器人的每种状态都清晰的打印在输出栏。
- **可维护性：**因其功能较为单一，只是实现了超市机器人的取货和放货等功能，所以维护起来比较方便。如果出现意外修改代码等情况会造成程序一定程度上的失效。
- **可靠性：**软件界面容错能力较好，机器人控制程序几乎未出现崩溃或者是任务意外中断的现象。总的来说，可靠性较好。
- **效率：**经过测试，方案一和方案二均可以高效的完成取货和放货的功能。对比两种方案，方案一又比方案二更高效一些。程序调用时未出现资源空间不足或者是卡顿、未响应等情况。
- **可移植性：**因为 webots 的跨平台特性，本系统可以在 windows 以及 Linux 平台上使用，以及通过修改代码，可以实现多种语言，如 C、C++、Python、Matlab 以及 ROS 等。

3 项目管理

3.1 人员管理

领域	问题
团队	本次项目中，我们团队共有两人，均为机器人工程专业大三本科生，分工是赵永强负责传感器部分，毕志海同学负责机械部分，我们共同编写控制器
目标	我们的目标是完全实现项目要求，即完成取货、放货、视觉识别、自主定位以及动态避障等功能，目前已经全部完成，潜在风险是控制器稳定性还有待提高
沟通	我们成员之前通过 QQ 以及腾讯会议等渠道进行交流
合作	成员之间的合作非常顺畅
风险	团队之间最大的风险和问题是由于完全是线上交流的形式，故而有时候不能及时交流或者是交流的不够透彻
期望	我们希望通过各自的努力，共同完成项目，实现要求的功能

图 10: 人员管理表

3.2 项目估计

- **规模：**本系统是以仿真平台为基础，以代码为工具实现的，故规模用代码行来体现，估计代码行为 1KLOC。
- **工作量：**本项目共 2 人，预计工作 2 星期，故工作量为 1 人月。
- **成本：**因为本项目完全是仿真实现，故成本主要是时间成本。

3.3 进度管理

活动		时间（天）	责任人	开工日数				
				2	4	6	8	10
详细设计	计划	4	赵/毕					
	实际	4	赵/毕					
编程	计划	4	赵/毕					
	实际	2	赵/毕					
单元测试	计划	2	赵/毕					
	实际	4	赵/毕					

图 11: 进度管理表

4 总结体会

首先，对于本次课程大作业，我们团队从需求建模、机器人建模以及机器人控制等方面完整的完成了整个项目，并结合了本课程所学的软件设计、软件管理以及项目管理等内容。前期主要是设计过程，通过和队友的线上讨论确定了项目所需要实现的功能以及机器人的构型。之后通过 webots 仿真平台，基于 youbot 机器人开发出了更加符合要求的超市机器人。并通过对整个项目的分析确定了机器人巡检方式、机械臂抓取和放置规则、视觉识别、自主定位以及动态避障方式。通过对项目的模块划分，我负责传感器的设计，队友负责机械臂的设计。在每个人负责的模块完成之后，我们进行控制器的编写，我们首先是采用了“倒退”的策略，虽然对于货柜上出现的各种情况都可以适应自如，但是需要对各个货柜的起点位置进行标定，且需要对各种商品和其对应的货柜之间的位置关系进行一一处理，代码量大大增加，于是我们又思考仅仅采用相机识别，大大减少了代码量，且无需标定，但缺点在于本法是依靠探测同类物品来进行货物放置的，因此，如出现全空货柜的情况，本法将不适用。最终，我们完成了整个项目的功能。

其次，对于本门课程，我们学习了关于软件的设计和管理以及项目管理等知识，并延伸到了与我们专业的相关的机器人软件设计。总的来说，对于本专业学生来说是十分有用的一门课程。大部分的同学都是从大一左右开始接触软件开发设计的，也做了不少的项目，但是在项目进行中，大家基本上都是“随心所欲”的，缺少规范，虽然最终都可以把项目完成的不错，但往往使得项目很“丑陋”。通过本课程的学习，我们可以很好的规范开发过程。除了理论知识外，我们还学习了与本专业息息相关的各种内容，如 ROS、webots 等，这些都是我们机器人学习道路上的好帮手。在过程中，还通过 webots 完成了了各种实验和挑战实验，这些都增加了我们对机器人的控制方法。

最后，对于本课程的提议就是，本课程的很多内容是基于《人月神话》还有其他一些书的，仅仅通过课堂上讲述，大家可能会存在听不进去的现象，如果可以让大家直接阅读原书，然后进行研讨，可以很好的让大家理解，提升大家的知识。

附录 A　主要代码

- **方案一**:

```
if(TURN_FLAG==1 && TURN_COUNT >200)
   {
      base_reset();
      passive_wait(0.5);
      base_turn_right();
      while(abs(ANGEL_new -angle[angle_index])>0.5 )
      {
        step();
        double *north1 = wb_compass_get_values(compass);
        ANGEL_new = get_bearing_in_degrees(north1);
      }
      STOP = STOP + 1;
      base_reset();
      angle_index = angle_index + 1;
      if(angle_index==4)
        angle_index = 0;

      TURN_COUNT = 0;
      TURN_FLAG = 0;
   }
   int number_of_objects = wb_camera_recognition_get_number_of_objects(camera);
   int width = wb_range_finder_get_width(range_finder);
   const float *image = wb_range_finder_get_range_image(range_finder);
   const WbCameraRecognitionObject *objects = wb_camera_recognition_get_objects(camera);

   //选取距离最近的物体信息输出
   for (i = 0; i < number_of_objects; ++i) {
       float depth = wb_range_finder_image_get_depth(image,
           width,objects[i].position_on_image[0],objects[i].position_on_image[1]);
       if(NEAREST > depth && strcmp(objects[i].model,"cabinet")!=0 ){
           NEAREST = depth;
           NEAREST_INDEX = i;
       }
   }
   OBJECT_AREA = objects[NEAREST_INDEX].size_on_image[0] *
        objects[NEAREST_INDEX].size_on_image[1];

   //根据id号、物体在图像X轴相对位置、物体在图像中的面积抓取物体
   ID = objects[NEAREST_INDEX].id;
   Mystrcpy1(Model ,objects[NEAREST_INDEX].model);
   IMAGE_X = objects[NEAREST_INDEX].position_on_image[0];

   if( BUSY == 0 && IMAGE_X <= 67 && IMAGE_X >= 63 && NEAREST<0.32 &&\
   ( (strcmp(Model,my_models[0])==0 && OBJECT_AREA<5200 && OBJECT_AREA>2900   && box ==0)
       ||\
     (strcmp(Model,my_models[1])==0 && OBJECT_AREA<5200 && OBJECT_AREA>3500 && boxr == 0)
         || \
     (strcmp(Model,my_models[2])==0 && OBJECT_AREA<5200 && OBJECT_AREA>3500 && boxb == 0)
         || \
     (strcmp(Model,my_models[3])==0  && OBJECT_AREA>1700 && canr == 0)  || \
     (strcmp(Model,my_models[4])==0  && OBJECT_AREA>1700 && cang == 0)  || \
     (strcmp(Model,my_models[5])==0 && OBJECT_AREA>1500 && jar == 0)  ||\
     (strcmp(Model,my_models[6])==0 && OBJECT_AREA>2200 && beer == 0)  ||\
     (strcmp(Model,my_models[7])==0 && OBJECT_AREA>3800 && water == 0))  )
   {
```

```
  Mystrcpy1(Model_target,objects[NEAREST_INDEX].model);
  printf("target %s\n",Model_target);
  Grip_ready();
  BUSY = 1; //目标机器人已经抓有物体，等待放置
  base_forwards();
  passive_wait(0.8);
  STOP = 0;
}
else if(angle_index == 0 && STOP >=4 )
 {
     base_reset();
     arm_reset();
     passive_wait(3);
     base_strafe_right();
     passive_wait(2);
     base_backwards();
     passive_wait(2.5);
     base_reset();
     while(1)
       step();
 }
else{
  while(distance/max_distance < 0.99 || distance2/max_distance2 < 0.99)
  {
   step();
   base_reset();
   if(distance/max_distance > 0.99 && distance2/max_distance2 > 0.99)
   {
      passive_wait(1);
      break;
   }
  }
  base_forwards();
}

if(BUSY == 1){
  STOP = 0;
  for (i = 0; i < number_of_objects_topf; ++i) {
    if(strcmp(objects_top_front[i].model, Model_target) == 0)
      OTF = 1;
  }
  for (i = 0; i < number_of_objects_top; ++i) {
    if(strcmp(objects_top[i].model, Model_target) == 0)
      OT = 1;
  }
  for (i = 0; i < number_of_objects_lowf; ++i) {
    if(strcmp(objects_low_front[i].model, Model_target) == 0)
      OLF = 1;
  }
  for (i = 0; i < number_of_objects_low; ++i) {
    if(strcmp(objects_low[i].model, Model_target) == 0)
      OL = 1;
  }

  if(OTF == 1 && OT == 0)
  {
    OTF = 0, OT = 0, OLF = 0, OL= 0;
    base_reset();
    passive_wait(0.5);
```

```
  base_forwards();
  passive_wait(0.1);
  if(strcmp("cereal boxb", Model_target) == 0)
  {
    passive_wait(0.8);
    base_reset();
    Detect();
    if(BLANK ==0){
      Low_SecondFloor_ready();
      BUSY = 0;
    }
  }
  else if(strcmp("biscuit box", Model_target) == 0 )
  {
     passive_wait(0.8);
     base_reset();
     Detect();
     if(BLANK == 0){
       Low_SecondFloor_ready();
       BUSY = 0;
     }
  }

  else if(strcmp("jar", Model_target) == 0 )
  {
     passive_wait(0.7);
     base_reset();
     Detect();
     if(BLANK == 0){
       Top_SecondFloor_ready();
       BUSY = 0;
     }
  }
  else if(strcmp("cereal boxr", Model_target) == 0 )
  {
     passive_wait(0.9);
     base_reset();
     Detect();
     if(BLANK == 0){
       Top_SecondFloor_ready();
       BUSY = 0;
     }
  }
  else
  {
     passive_wait(0.85);
     base_reset();
     Detect();
     if(BLANK == 0){
       Top_SecondFloor_ready();
       BUSY = 0;
     }
  }
}

if(OTF == 0 && OT == 1)
{
  OTF = 0, OT = 0, OLF = 0, OL= 0;
  base_reset();
```

```
  passive_wait(0.5);
  base_forwards();
  passive_wait(0.8);
  if(strcmp("biscuit box", Model_target) == 0 )
  {

     passive_wait(0.8);
     base_reset();
     Detect();
     if(BLANK == 0){
       Low_SecondFloor_ready();
       BUSY = 0;
     }
  }
  else if(strcmp("cereal boxb", Model_target) == 0 )
  {
     passive_wait(0.8);
     base_reset();
     Detect();
     if(BLANK == 0){
       Low_SecondFloor_ready();
       BUSY = 0;
     }
  }
  else if(strcmp("jar", Model_target) == 0 )
  {
      passive_wait(0.8);
      base_reset();
      Detect();
      if(BLANK == 0){
        Top_SecondFloor_ready();
        BUSY = 0;
      }
  }
  else if(strcmp("cereal boxr", Model_target) == 0)
  {
      passive_wait(0.65);
      base_reset();
      Detect();
      if(BLANK == 0){
        Top_SecondFloor_ready();
        BUSY = 0;
      }
  }
  else{
     passive_wait(0.5);
     base_reset();
     Detect();
     if(BLANK == 0){
       Top_SecondFloor_ready();
       BUSY = 0;
     }
  }
}

if(OLF == 1 && OL == 0)
{
  base_reset();
  OTF = 0, OT = 0, OLF = 0, OL= 0;
```

```
      base_reset();
      passive_wait(0.5);
      base_forwards();
      if(strcmp("beer bottle", Model_target) == 0)
      {
        passive_wait(0.8);
        base_reset();
        Detect();
        if(BLANK == 0){
          FirstFloor_ready();
          BUSY = 0;
        }
      }
      else
      {
        passive_wait(0.9);
        base_reset();
        Detect();
        if(BLANK == 0){
          FirstFloor_ready();
          BUSY = 0;
        }
      }
     }
    if(OLF == 0 && OL == 1 && strcmp("cereal boxb", Model_target) != 0)
    {
      base_reset();
      OTF = 0, OT = 0, OLF = 0, OL= 0;
      base_reset();
      passive_wait(0.5);
      base_forwards();
      if(strcmp("water bottle", Model_target) == 0)
          passive_wait(0.8);
      else
          passive_wait(1.1);

      passive_wait(0.8);
      base_reset();
      Detect();
      if(BLANK == 0){
        FirstFloor_ready();
        BUSY = 0;
      }
    }
    OTF = 0, OT = 0, OLF = 0, OL= 0;
  }
```

- **方案二**:

```
int main(int argc, char **argv) {
  wb_robot_init();
  base_init();
  arm_init();
  gripper_init();
  passive_wait(2.0);
  int TURN_COUNT = 0,TURN_FLAG = 0;
  int pc = 0;
  int i, j,OBJECT_AREA,ID,IMAGE_X,NEAREST_INDEX = 0;
```

```
float NEAREST;
double ANGEL,ANGEL_new = 0;
char MODEL;
double angle[4] = {270,0,90,180};
int angle_index = 0;
int id = 0;
int cabinet_num=0;
int STOP = 0;//取放货物的个数
int
    id1_flag=0,id2_flag=0,id3_flag=0,id4_flag=0,id5_flag=0,id6_flag=0,id7_flag=0,id8_flag=0,id9_flag=0;
int
    grip_num1=0,grip_num2=0,grip_num3=0,grip_num4=0,grip_num5=0,grip_num6=0,grip_num7=0,grip_num8=0,grip_num9=
const char* my_models[9] = {"biscuit box","cereal box1","honey jar","water bottle1",\
"beer bottle","cereal box_r1","jam jar1","can1","can_g"};
wb_keyboard_enable(TIME_STEP);

//摄像头
camera = wb_robot_get_device("camera");
wb_camera_enable(camera,TIME_STEP);
wb_camera_recognition_enable(camera,TIME_STEP);

camera_top = wb_robot_get_device("camera_top");
wb_camera_enable(camera_top,TIME_STEP);
wb_camera_recognition_enable(camera_top,TIME_STEP);
camera_low = wb_robot_get_device("camera_low");
wb_camera_enable(camera_low,TIME_STEP);
wb_camera_recognition_enable(camera_low,TIME_STEP);

//距离传感器
range_finder = wb_robot_get_device("range-finder");
wb_range_finder_enable(range_finder,TIME_STEP);

//GPS
WbDeviceTag gps = wb_robot_get_device("gps");
wb_gps_enable(gps, TIME_STEP);

//Compass
WbDeviceTag compass = wb_robot_get_device("compass");
wb_compass_enable(compass, TIME_STEP);

//Distance sensor
WbDeviceTag distance_sensor = wb_robot_get_device("distance sensor");
wb_distance_sensor_enable(distance_sensor, TIME_STEP);
//main loop
while (true) {
  step();
  NEAREST_INDEX = 0;
  NEAREST = 10;
  TURN_COUNT = TURN_COUNT+1;
  const double *gps_values = wb_gps_get_values(gps);
  double *north = wb_compass_get_values(compass);
  double distance = wb_distance_sensor_get_value(distance_sensor);
  double max_distance = wb_distance_sensor_get_max_value(distance_sensor);
  ANGEL = get_bearing_in_degrees(north);
  ANGEL_new = ANGEL;
  TURN_FLAG = IS_turn_point(gps_values[0], gps_values[2]);

  printf("the distance from the obstacle: %f \n",distance/max_distance);
  if(distance/max_distance < 0.5)
```

```
{
   base_reset();
   passive_wait(5);
}

if(BUSY == 0&&STOP == 4)
 {

     if (fabs(gps_values[0]-3.35)<0.05&&fabs(gps_values[2]-0.3)<0.05)
        {
          printf("%s \n","Go back to the starting point!");
          //arm_reset();
          //base_reset();
          //passive_wait(10);
          base_reset();
          arm_reset();
          passive_wait(3);
          base_strafe_right();
          passive_wait(1.5);
          base_backwards();
          passive_wait(2.5);
          base_reset();
          while(1)
             step();
        }
 }

if(TURN_FLAG==1 && TURN_COUNT >200)
{
   base_reset();
   passive_wait(0.5);
   base_turn_right();
   while(abs(ANGEL_new -angle[angle_index])>0.5 )
   {
     step();
     double *north1 = wb_compass_get_values(compass);
     ANGEL_new = get_bearing_in_degrees(north1);
     //printf("Angel %.3f\n", ANGEL_new);
   }
   //passive_wait(3.841);
   //STOP = STOP + 1;
   base_reset();

   angle_index = angle_index + 1;
   if(angle_index==4)
     angle_index = 0;
   TURN_COUNT = 0;
   TURN_FLAG = 0;
}

// printf("Using the GPS device: %.3f %.3f %.3f\n", gps_values[0], gps_values[2], ANGEL);

 int number_of_objects = wb_camera_recognition_get_number_of_objects(camera);
 int width = wb_range_finder_get_width(range_finder);
 const float *image = wb_range_finder_get_range_image(range_finder);
 const WbCameraRecognitionObject *objects = wb_camera_recognition_get_objects(camera);
 int number_of_objects_top = wb_camera_recognition_get_number_of_objects(camera_top);
 int number_of_objects_low = wb_camera_recognition_get_number_of_objects(camera_low);
 const WbCameraRecognitionObject *objects_top =
```

```
    wb_camera_recognition_get_objects(camera_top);
const WbCameraRecognitionObject *objects_low =
    wb_camera_recognition_get_objects(camera_low);
//printf("number_of_objects_low: %d \n",number_of_objects_low);

//选取距离最近的物体信息输出
for (i = 0; i < number_of_objects; ++i) {
    float depth = wb_range_finder_image_get_depth(image,
        width,objects[i].position_on_image[0],objects[i].position_on_image[1]);
    if(NEAREST > depth){
        NEAREST = depth;
        NEAREST_INDEX = i;
    }
}
OBJECT_AREA = objects[NEAREST_INDEX].size_on_image[0] *
    objects[NEAREST_INDEX].size_on_image[1];
printf("model: %s id: %d   x: %d   y: %d  distance:
    %f\n",objects[NEAREST_INDEX].model,objects[NEAREST_INDEX].id,objects[NEAREST_INDEX].position_on_image[0]
printf("Area of the object in image %d\n", OBJECT_AREA);

//根据id号、物体在图像X轴相对位置、物体在图像中的面积抓取物体
ID = objects[NEAREST_INDEX].id;
IMAGE_X = objects[NEAREST_INDEX].position_on_image[0];
const char* Model = objects[NEAREST_INDEX].model;
//判断分别为                          biscuit box
                                                    cereal box
                                        honey jar                       water
    bottle                                beer   bottle
if( BUSY == 0 && IMAGE_X <= 67 && IMAGE_X >= 63 && NEAREST < 0.32&&\
( (strcmp(Model,my_models[0])==0 && OBJECT_AREA<4500 && OBJECT_AREA>2900&&grip_num1==0)
    ||\
  (strcmp(Model,my_models[1])==0 && OBJECT_AREA<5000 && OBJECT_AREA>3500&&grip_num2==0)
      ||\
  (strcmp(Model,my_models[2])==0 && OBJECT_AREA>1500&&grip_num3==0)  ||\
  (strcmp(Model,my_models[3])==0 && OBJECT_AREA>3800&&grip_num4==0)  ||\
  (strcmp(Model,my_models[4])==0 && OBJECT_AREA>2200&&grip_num5==0)  ||\
  (strcmp(Model,my_models[5])==0 && OBJECT_AREA<5000 && OBJECT_AREA>3500&&grip_num6==0)
      ||\
  (strcmp(Model,my_models[6])==0 && OBJECT_AREA>1500&&grip_num7==0)  ||\
  (strcmp(Model,my_models[7])==0  && OBJECT_AREA>1700&&grip_num8==0)  ||\
  (strcmp(Model,my_models[8])==0  && OBJECT_AREA>1700&&grip_num9==0)))
{
  for (int i =0;i<9;i++)
  {
      if (strcmp(Model,my_models[i])==0)
          id = i+1;
  }

  printf("%d \n",id);
  Grip_ready();
  BUSY = 1; //目标机器人已经抓有物体，等待放置
  STOP = STOP + 1;
  number_of_objects_top = wb_camera_recognition_get_number_of_objects(camera_top);
  number_of_objects_low = wb_camera_recognition_get_number_of_objects(camera_low);
  const WbCameraRecognitionObject *objects_top =
      wb_camera_recognition_get_objects(camera_top);
  const WbCameraRecognitionObject *objects_low =
      wb_camera_recognition_get_objects(camera_low);
  for (int j=0;j<number_of_objects_top;j++)
```

```
{
    if (strcmp(objects_top[j].model,"cabinet_a")==0)
       cabinet_num = 1;
    else if (strcmp(objects_top[j].model,"cabinet_b")==0)
       cabinet_num=2;
    else if (strcmp(objects_top[j].model,"cabinet_c")==0)
       cabinet_num=3;
    else if (strcmp(objects_top[j].model,"cabinet_d")==0)
       cabinet_num=4;
}
if ((id == 5 || id ==6) && cabinet_num == 2)
   {//printf("%s \n","11111111111111111111");
   while ((gps_values[2]-0.620)>0.005)//0.586
       {
          step();
          base_backwards();
       }
   number_of_objects_top = wb_camera_recognition_get_number_of_objects(camera_top);
   number_of_objects_low = wb_camera_recognition_get_number_of_objects(camera_low);
   const WbCameraRecognitionObject *objects_top =
        wb_camera_recognition_get_objects(camera_top);
   const WbCameraRecognitionObject *objects_low =
        wb_camera_recognition_get_objects(camera_low);
   printf("number_of_objects_top: %d \n",number_of_objects_top);
   }
else if ((id == 3||id == 4||id == 7) && cabinet_num == 4)
        {
         while ((2.451-gps_values[2])>0.005)
         {
            step();
            base_backwards();
         }
         number_of_objects_top =
              wb_camera_recognition_get_number_of_objects(camera_top);
         number_of_objects_low =
              wb_camera_recognition_get_number_of_objects(camera_low);
         const WbCameraRecognitionObject *objects_top =
              wb_camera_recognition_get_objects(camera_top);
         const WbCameraRecognitionObject *objects_low =
              wb_camera_recognition_get_objects(camera_low);
         printf("number_of_objects_top: %d \n",number_of_objects_top);
        }
else if ((id == 1||id == 8) && cabinet_num == 1)
        {
         while ((3.14-gps_values[0])>0.005)
         {
            step();
            base_backwards();
         }
         number_of_objects_top =
              wb_camera_recognition_get_number_of_objects(camera_top);
         number_of_objects_low =
              wb_camera_recognition_get_number_of_objects(camera_low);
         const WbCameraRecognitionObject *objects_top =
              wb_camera_recognition_get_objects(camera_top);
         const WbCameraRecognitionObject *objects_low =
              wb_camera_recognition_get_objects(camera_low);
         printf("number_of_objects_top: %d \n",number_of_objects_top);
        }
```

```
  else if ((id == 2||id == 9) && cabinet_num == 3)
          {
           while ((gps_values[0]-1.33)>0.005)
           {
              step();
              base_backwards();
           }
           number_of_objects_top =
               wb_camera_recognition_get_number_of_objects(camera_top);
           number_of_objects_low =
               wb_camera_recognition_get_number_of_objects(camera_low);
           const WbCameraRecognitionObject *objects_top =
               wb_camera_recognition_get_objects(camera_top);
           const WbCameraRecognitionObject *objects_low =
               wb_camera_recognition_get_objects(camera_low);
           printf("number_of_objects_top: %d \n",number_of_objects_top);
          }
}
else{
  if (id1_flag==0&&id2_flag==0&&id3_flag==0&&id4_flag==0&&id5_flag==0\
      &&id6_flag==0&&id7_flag==0&&id8_flag==0&&id9_flag==0)
      base_forwards();
}
printf("id: %d \n",id);
//放物品,A货架id=762,B货架id=963,C货架id=561,D货架id=1164
if(BUSY == 1){
  //STOP = 0;
  number_of_objects_top = wb_camera_recognition_get_number_of_objects(camera_top);
  number_of_objects_low = wb_camera_recognition_get_number_of_objects(camera_low);
  objects_top = wb_camera_recognition_get_objects(camera_top);
  objects_low = wb_camera_recognition_get_objects(camera_low);

  if(id == 1){
    int flag_1=0;
    for (int i=0;i<number_of_objects_top;++i)
    {
        if (strcmp(objects_top[i].model,"cabinet_a") == 0)
           flag_1 ++;
    }
    if (flag_1 != 0)
    {
      /*printf("number_of_objects_top: %d \n",number_of_objects_top);
      if (number_of_objects_top != 1)
      {
          //step();
          base_reset();
          //passive_wait(0);
          base_forwards();
          passive_wait(0);
          base_backwards();
          passive_wait(0.1);

          //number_of_objects_top =
              wb_camera_recognition_get_number_of_objects(camera_top);
      }
      if (number_of_objects_top == 1)
        {
           Low_SecondFloor_ready();
           BUSY = 0;
```

```
      }*/
    if (number_of_objects_top == 1)
    {
      if (id1_flag==1)
       {

         //passive_wait(1.2);
         //base_backwards();
         //base_forwards();
         //passive_wait(1.6);
       }
      Low_SecondFloor_ready();
      BUSY = 0;
      id1_flag=0;
      grip_num1=1;
    }
    else{
       id1_flag=1;
       //passive_wait(1.2);
       base_forwards();
       passive_wait(0.01);
       base_reset();
       //passive_wait(0.6);

    }
  }
}
else if (id == 2){
  int flag_2=0;
  for (int i=0;i<number_of_objects_top;++i)
  {
      if (strcmp(objects_top[i].model,"cabinet_c") == 0)
         flag_2 ++;
  }
  if (flag_2 != 0)
  {
    printf("number_of_objects_top: %d \n",number_of_objects_top);
    if ((number_of_objects_top == 1||number_of_objects_top == 0)\
        &&number_of_objects_low != 0)
      {
         if (id2_flag==1)
         {
           //base_backwards();
           //passive_wait(0);
         }
        Low_SecondFloor_ready();
        BUSY = 0;
        id2_flag=0;
        grip_num2=1;
      }
      else{
         id2_flag=1;
         base_forwards();
         passive_wait(0.1);
         base_reset();
      }
  }
}
else if (id == 3){
```

```
  int flag_3=0;
  for (int i=0;i<number_of_objects_top;++i)
  {
      if (strcmp(objects_top[i].model,"cabinet_d") == 0)
         flag_3 ++;
  }
  if (flag_3 != 0)
  {
    printf("number_of_objects_top: %d \n",number_of_objects_top);
    if (number_of_objects_top == 1)
      {
         if (id3_flag==1)
         {
            //base_backwards();
            //passive_wait(0);
         }
         Top_SecondFloor_ready();
         BUSY = 0;
         id3_flag=0;
         grip_num3=1;
      }
      else{
         id3_flag=1;
         base_forwards();
         passive_wait(0.97);
         base_reset();
      }
  }
  }
else if (id == 4){
  int flag_4=0;
  int i;
  for (i=0;i<number_of_objects_low;++i)
  {
      if (strcmp(objects_low[i].model,"cabinet_d") == 0)
      {
         flag_4 ++;
         //break;
      }
  }
  if (flag_4 != 0)
  {
      //FirstFloor_ready();
      printf("number_of_objects_low: %d \n",number_of_objects_low);
      if ((number_of_objects_low == 1||number_of_objects_low == 0)&&
      number_of_objects_top != 0)
      {
         if (id4_flag==1)
         {
            //base_backwards();
            //passive_wait(0);
         }
         FirstFloor_ready();
         BUSY = 0;
         id4_flag=0;
         grip_num4=1;
      }
      else{
         id4_flag=1;
```

```
            base_forwards();
            passive_wait(1);
            base_reset();
        }
    }
  }
  else if (id == 5){
    int flag_5=0;
    for (int i=0;i<number_of_objects_low;++i)
    {
        if (strcmp(objects_low[i].model,"cabinet_b") == 0)
            flag_5 ++;
    }
    if (flag_5 != 0)
    {
      printf("number_of_objects_low: %d \n",number_of_objects_low);
      if (number_of_objects_low == 1)
        {
            if (id5_flag == 1)
            {
                //base_backwards();
                //passive_wait(0);
            }

            FirstFloor_ready();
            BUSY = 0;
            id5_flag=0;
            grip_num5=1;
        }
        else{
            id5_flag = 1;
            base_forwards();
            passive_wait(2);
            base_reset();
        }
    }
  }
 else if (id == 6){
    int flag_6=0;
    for (int i=0;i<number_of_objects_top;++i)
    {
        if (strcmp(objects_top[i].model,"cabinet_b") == 0)
            flag_6 ++;
    }
    if (flag_6 != 0)
    {
      printf("number_of_objects_top: %d \n",number_of_objects_top);
      if (number_of_objects_top == 1)
        {
            if (id6_flag==1)
            {
               //base_backwards();
               //passive_wait(0);
            }
            Top_SecondFloor_ready();
            BUSY = 0;
            id6_flag=0;
            grip_num6=1;
        }
```

```
        else{
           id6_flag=1;
           base_forwards();
           passive_wait(0.001);
           base_reset();
        }
    }
 }
else if (id == 7){
    int flag_7=0;
    for (int i=0;i<number_of_objects_top;++i)
    {
        if (strcmp(objects_top[i].model,"cabinet_d") == 0)
           flag_7 ++;
    }
    printf("flag_7:%d \n",flag_7);
    if (flag_7 != 0)
    {
      printf("number_of_objects_top: %d \n",number_of_objects_top);
      if (number_of_objects_top == 1)
        {
           if (id7_flag==1)
           {
              //base_backwards();
              //passive_wait(0);
           }
           Top_SecondFloor_ready();
           BUSY = 0;
           id7_flag=0;
           grip_num7=1;
        }
        else{
           id7_flag=1;
           base_forwards();
           passive_wait(0.97);
           base_reset();
        }
    }
 }
else if (id == 8){
    int flag_8=0;
    for (int i=0;i<number_of_objects_low;++i)
    {
        if (strcmp(objects_low[i].model,"cabinet_a") == 0)
           flag_8 ++;
    }
    if (flag_8 != 0)
    {
      printf("number_of_objects_low: %d \n",number_of_objects_low);
      if (number_of_objects_low == 1)
        {
           /*if (id8_flag==1)
           {
              base_backwards();
              passive_wait(0);
           }*/
           FirstFloor_ready();
           BUSY = 0;
           id8_flag=0;
```

```
                grip_num8=1;
            }
            else{
                id8_flag=1;
                base_forwards();
                passive_wait(2);//2
                base_reset();
            }
        }
      }
     else if (id == 9){
        int flag_9=0;
        for (int i=0;i<number_of_objects_low;++i)
        {
            if (strcmp(objects_low[i].model,"cabinet_c") == 0)
                flag_9 ++;
        }
        if (flag_9 != 0)
        {
          printf("number_of_objects_low: %d \n",number_of_objects_low);
          if (number_of_objects_low == 1)
            {
                if (id9_flag==1)
                {
                   //base_backwards();
                   //passive_wait(0);
                }
                FirstFloor_ready();
                BUSY = 0;
                id9_flag=0;
                grip_num9=1;
            }
            else{
                id9_flag=1;
                base_forwards();
                passive_wait(2);
                base_reset();
            }
        }
      }
    }
}
  wb_robot_cleanup();

  return 0;
}
```