

车载软件设计基础

松灵小车任务4



**2023至2024学年第 1 学期**

|  |  |  |
| --- | --- | --- |
| 学号 | 姓名 | 团队作用 |
| E2020196 | 侯加喜 | 代码编写、报告撰写 |
| E2020017 | 费川川 | 代码编写、报告撰写 |
| E2020105 | 陈荣 | 代码编写、视频讲解 |
|  |  |  |
| 任课教师 | 陈恒鑫 | |
| 成 绩 |  | |

|  |  |
| --- | --- |
| 任务书 | |
| 任务内容 | 实验分组进行，每组人数不超过2人。在松灵小车上编写ROS程序或者配置相关程序，使其能够在固定和略有变化的实际场景中能够实现自主导航，实际环境又老师统一指定。其具体任务内容有：  （1）采用gmapping方法进行SLAM建图；  （2）采用move\_base方法从固定起点到固定终点的小车自主导航；  （3）在环境实时变化的场景中，尽量保证更快更准确地进行小车自主导航；  （4）通过识别任务卡上的图标，理解车辆的终点，并完成（3）。 |
| 程序规范 | （1）所有程序代码采用C++编写，使用git进行源代码管理；  （2）类名、变量名、函数名应符合C++的命名规范，并在代码中前后保持一致；  （3）涉及面向对象的程序，例如自定义的类，应符合面向对象的设计原则；  （4）正确使用头文件和源文件，自定义的头文件应符合头文件的编写原则，例如用条件宏定义确保头文件不被多次引用、不在头文件中进行类和函数的实现（模板除外）；  （5）项目必须是ROS项目，符合ROS的项目的规范，正确编写CmakeLists.txt等文件；  （6）程序能够在松灵小车上运行。 |
| 报告要求 | （1）报告至少应该包括人员分工、需求分析、程序设计、程序效果展示、总结分析4个部分；  （2）人员分工介绍组员各自的工作情况；  （3）需求分析侧重描述程序所需要实现的功能，功能预期的效果；  （4）程序设计描述组成程序的模块、类、函数以及他们之间的相互关系，若有算法，可以描述算法流程；  （5）程序效果展示除了程序运行效果截图之外，应该有必要的文字说明；  （6）总结分析可以分析实现的效果与理想情况的差异，分析导致这些差异的原因，切忌不要写成心得体会；  （7）报告应该格式规范、排版整洁、少语病和错误。 |
| 作业提交 | （1）含有git仓库（有.git目录）的完整源代码；  （2）程序功能演示的讲解视频；  （3）任务报告。 |
| 评分标准 | 按照五级制打分，分为优秀、良好、中等、及格、不及格，各评分项占总成绩的比例为：  （1）任务完成情况占评分的60%；  （2）程序规范占评分的20%；  （3）报告占评分的20%。  评分老师根据各部分的完成情况，直接给出总成绩。 |

**一、人员分工**

1.侯加喜: 担任组长,协调各组员按时完成任务。组织小组会议讨论遇到的困难并及时解决。同时负责记录小车坐标和车身朝向的调整和编写识别任务卡图标代码，以及报告撰写

2.费川川: 采用move\_base方法从固定起点到固定终点的小车自主导航，以及报告撰写。

3.陈荣: 采用gmapping方法进行SLAM建图,讲解视频。

**二、需求分析**

1.采用gmapping方法进行SLAM建图；

建图: roslaunch scout\_bringup gmapping.launch

保存: rosrun map\_server map\_saver -f /home/agilex/task4/maps/demo

2.采用move\_base方法从固定起点到固定终点的小车自主导航；

roslaunch scout\_bringup navigation\_4wd.launch

3.在环境实时变化的场景中，尽量保证更快更准确地进行小车自主导航；

基于A\*算法进行自动导航

4.通过识别任务卡上的图标，理解车辆的终点，并完成（3）。

编写ImageConverter类,通过image\_recognition\_node节点进行文字识别

**三、程序设计**

1.CimageConverter.h

#include <image\_transport/image\_transport.h>

#include <cv\_bridge/cv\_bridge.h>

#include <tesseract/baseapi.h>

#include <geometry\_msgs/PoseStamped.h>

#include <move\_base\_msgs/MoveBaseAction.h>

#include <actionlib/client/simple\_action\_client.h>

#include <opencv2/imgproc/imgproc.hpp>

#include <opencv2/highgui/highgui.hpp>

#include <cctype>

#include <iostream>

#include <mutex>

#include <thread>

#include <stdlib.h>

typedef actionlib::SimpleActionClient<move\_base\_msgs::MoveBaseAction> MoveBaseClient;

class ImageConverter{

private:

ros::NodeHandle nh\_;

image\_transport::ImageTransport it\_;

image\_transport::Subscriber image\_sub\_color;

ros::Publisher pub = nh\_.advertise<geometry\_msgs::PoseStamped>("/move\_base\_simple/goal", 10);

MoveBaseClient ac{"move\_base", true};

public:

//构造函数: 将ROS节点句柄nh\_传递给image\_transport::ImageTransport对象it\_，在ROS系统中进行图像传输

ImageConverter() : it\_(nh\_) {

//订阅/camera/color/image\_raw的彩色图像消息回调函数指定为&ImageConverter::imageCb

image\_sub\_color = it\_.subscribe("/camera/color/image\_raw", 1, &ImageConverter::imageCb, this);

system("aplay /home/agilex/task4/sources/wav/Hello.wav");

return;

}

//析构函数

~ImageConverter() {

return;

}

void imageCb(const sensor\_msgs::ImageConstPtr& msg) {

cv\_bridge::CvImagePtr cv\_ptr;

try {

// 将ROS图像消息转为OpenCV图像对象

cv\_ptr = cv\_bridge::toCvCopy(msg, sensor\_msgs::image\_encodings::BGR8);

}

catch (cv\_bridge::Exception& e) {

ROS\_ERROR("cv\_bridge exception: %s", e.what());

return;

}

// 创建Tesseract OCR对象

tesseract::TessBaseAPI tess;

tess.Init(NULL, "eng", tesseract::OEM\_DEFAULT);

tess.SetPageSegMode(tesseract::PSM\_SINGLE\_CHAR);

// 进行OCR文字识别

tess.SetImage(cv\_ptr->image.data, cv\_ptr->image.cols, cv\_ptr->image.rows, 3, cv\_ptr->image.step);

char\* outText = tess.GetUTF8Text();

// 去除字符串中的空格或不可见字符

std::string cleanedText;

for (int i = 0; i < strlen(outText); i++) {

if (!std::isspace(outText[i])) {

cleanedText += outText[i];

}

}

// 输出识别结果

ROS\_INFO("OCR result:%s",outText);

// 如果识别结果是A、B、C或D，则发布导航目标点消息并进行导航

std::string result(cleanedText);

// std::unique\_lock<std::mutex> locker(m\_mutex, defer\_lock);

// locker.lock();

// std::cin >> result;

if (result == "A") {

ROS\_INFO("Navigating to point A \n");

system("aplay /home/agilex/task4/sources/wav/A.wav");

publishAndNavigate(-2.13, 6.15, -0.97, -2.6);

system("aplay /home/agilex/task4/sources/wav/RA.wav");

} else if (result == "B") {

ROS\_INFO("Navigating to point B \n");

system("aplay /home/agilex/task4/sources/wav/B.wav");

publishAndNavigate(-3.18, 4.10, -0.62, -1.2);

system("aplay /home/agilex/task4/sources/wav/RB.wav");

} else if (result == "C") {

ROS\_INFO("Navigating to point C \n");

system("aplay /home/agilex/task4/sources/wav/C.wav");

publishAndNavigate(0.79, 4.82, 0.70, 1.3);

system("aplay /home/agilex/task4/sources/wav/RC.wav");

} else if (result == "D") {

ROS\_INFO("Navigating to point D \n");

system("aplay /home/agilex/task4/sources/wav/D.wav");

publishAndNavigate(0.53, 1.24, 0.28, -0.5);

system("aplay /home/agilex/task4/sources/wav/RD.wav");

} else {

//如果识别结果是其他信息就做出相应提示

ROS\_WARN("OCR result is not A、B、C or D \n");

}

// 释放内存

delete[] outText;

// 显示图像

cv::imshow("OPENCV\_WINDOW\_COLOR", cv\_ptr->image);

cv::waitKey(3);

return;

}

void publishAndNavigate(double x, double y, double az, double aw) {

// 将识别到的字母对应的坐标发送给导航目标点

// 创建一个PoseStamped消息，用于发送导航目标点信息

geometry\_msgs::PoseStamped goal\_msg;

// 设置导航目标点的header信息

goal\_msg.header.seq = 0;

goal\_msg.header.stamp = ros::Time::now();

goal\_msg.header.frame\_id = "map";

// 设置导航目标点的位置信息

goal\_msg.pose.position.x = x;

goal\_msg.pose.position.y = y;

goal\_msg.pose.position.z = 0.0;

// 设置导航目标点的朝向信息(默认朝向)

goal\_msg.pose.orientation.x = 0.0;

goal\_msg.pose.orientation.y = 0.0;

goal\_msg.pose.orientation.z = az;

goal\_msg.pose.orientation.w = aw;

// 发布导航目标点消息

ROS\_INFO("Publish navigation target point message");

pub.publish(goal\_msg);

// 创建并设置一个导航目标消息

move\_base\_msgs::MoveBaseGoal goal;

goal.target\_pose = goal\_msg;

// 发送导航目标点消息

ROS\_INFO("Send navigation target point message");

ac.sendGoal(goal);

// 等待导航完成

ac.waitForResult();

// 输出导航结果

if (ac.getState() == actionlib::SimpleClientGoalState::SUCCEEDED) {

ROS\_INFO("Navigation succeeded");

} else {

ROS\_WARN("Navigation failed");

}

std::cin.get(); // 导航途中不识别

return;

}

};

2.image\_recognition\_node.cpp

#include <ros/ros.h>

#include "../include/CImageConverter.h"

int main(int argc, char\*\* argv) {

ros::init(argc, argv, "image\_converter");

ros::NodeHandle nh;

ImageConverter ic;

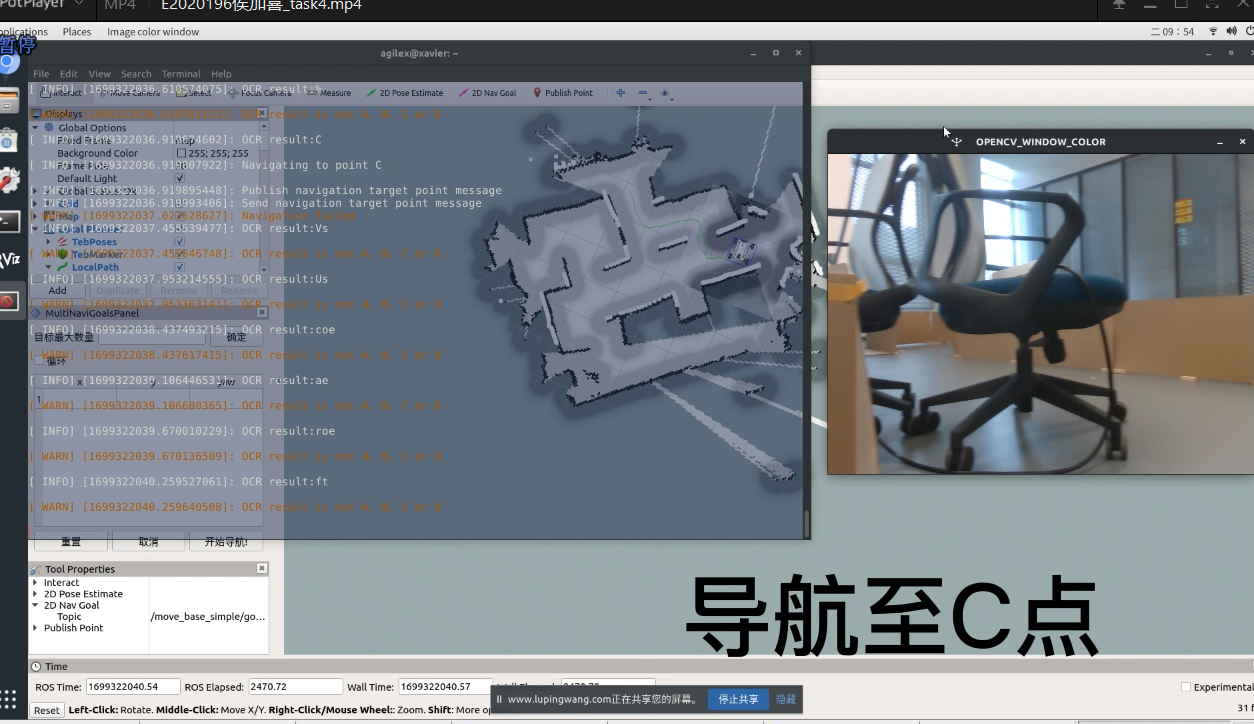
ros::spin();

return 0;

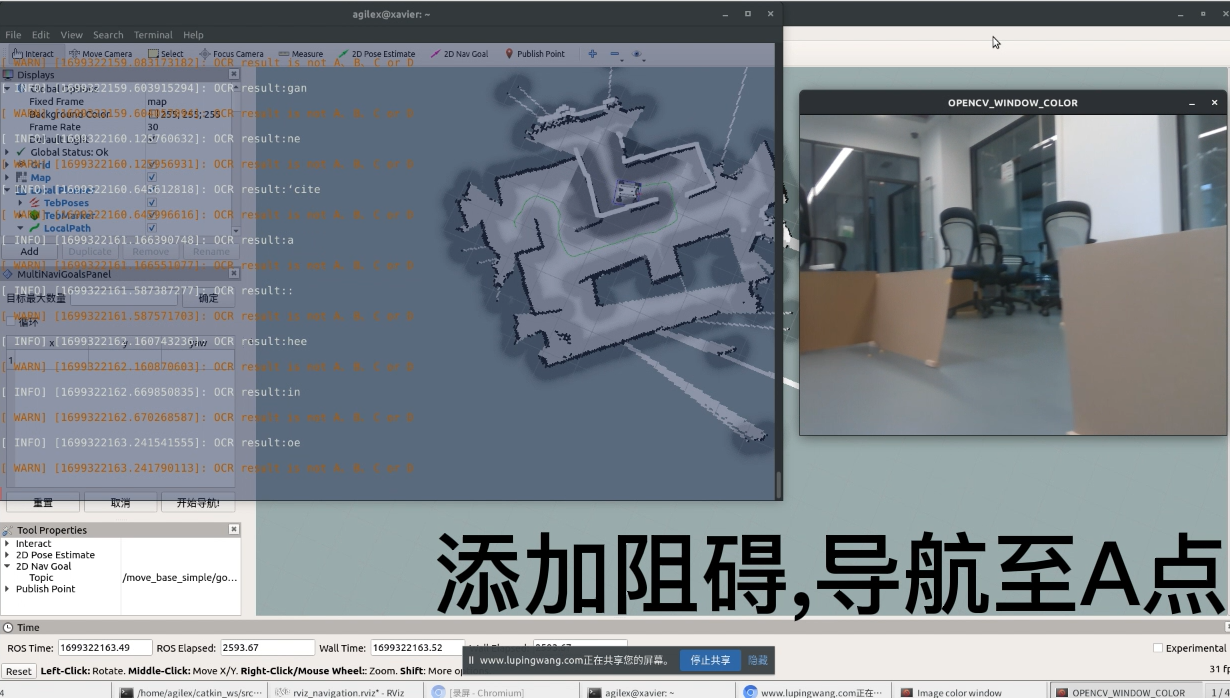
}

**四、程序效果展示**

1.进行文字识别并规划导航路线



2.发现新障碍自动更换路线



**五、总结分析**

此次使用基本达到预期效果，难度不算大但出现过不少”意外故障”。

1.比如松灵小车官方提供的ugv\_sdk可以通过cmake和catkin\_make两种方式进行编译，cmake可以生成demo目录下的可执行文件，然后直接执行即可。但catkin\_make方式却无法生成，首先考虑是不是编译环境有问题，排错花费了很多时间，最后阅读CMakeLists.txt才发现：使用catkin\_make方式编译时，会直接忽略demo目录，那么也不会生成下面的可执行文件。

3.相机功能包节点运行后显示无法正常显示画面,多次重启后方才重新有效。

4.can总线不能正常使用,对进行连接的功能包函数逐步测试后生效。

5.在行驶途中若仍不断进行识别,则可能造成误识别,故需等待导航完成再进行识别。

**六、参考文献**

[1] ROS的三种基本通信机制. http://t.csdn.cn/5KJ2z,2022-05-29.

[2] 松灵机器人Scout mini 使用Ros例程控制.zhuanlan.zhihu.com/p/550920775,2022-09-17.

[3] ROS订阅odom里程计消息，并在RVIZ中显示轨迹. http://t.csdn.cn/psy6w,2021-07-15.

[4] ROS学习（四）：消息格式. http://t.csdn.cn/hzVyi,2019-03-04.

[5][美]Stephen Prata.C++ Primer plus[M].张海龙 袁国忠.北京:人民邮电出版社,2015:96-183.

[6] ROS-Mobile.https://github.com/ROS-Mobile/ROS-Mobile-Android/blob/master,2023-04-16.