1.实验名称及目的

1.1 实验名称

Mavros C++版本控制接口实验

1.2 实验目的

运行mavros 的C++程序，进行mavros控制。

1.3 关键知识点

运行一个python程序，打开mavros主节点，将mavlink消息转化成ros消息,然后，再运行一个mavros 的C++程序，进行mavros控制。

本实验主要是实现通过Python接口RflyRosStart.py（见RflySimAPIs\RflySimSDK\ctrl目录），展示用Python mavros进行飞机控制。关键代码解析如下：

本例子和其他分布式例子的区别，主要在于使用mavros进行飞机控制，其中控制代码是c++编写的，采用c++语言进行控制。同时本例程也用了多种方式控制飞机，并不仅仅是让飞机起飞而已，还进行了更多控制操作。

关键知识点1：通过ReqCopterSim可以自动从局域网获取到仿真电脑的IP地址，从而自动建立连接，不再需要手动指定IP地址。不过，此种连接方式，可能在局域网中产生干扰（多台电脑同时打开多个CopterSim会产生误识别），不适合多个实验同时进行的场景。

1） ReqCopterSim接口使用（自动获取ip接口）

```
req = ReqCopterSim.ReqCopterSim() \# 获取局域网内所有CopterSim程序的电脑IP列表

TargetIP = req.getSimIpID(StartCopterID) \#自动获取CopterSim的StartCopterID号程序所在电脑

req.sendReSimIP(CopterID) \# 请求mavlink数据到本电脑

req.sendReSimUdpMode(TargetID,2) \# 强制切换MAVLINK_FULL
```

2）ros/ros.h使用

```
ros::init(argc, argv, "offb_node"); \# 创建节点

ros::NodeHandle nh; \# 创建句柄

ros::Subscriber state_sub = nh.subscribe\<mavros_msgs::State\>("mavros/state",
10, state_cb); \#订阅mavros/state话题消息，该消息类型是State，使用state_cb做回调函数，限制排队

ros::Publisher local_pos_pub =nh.advertise\<geometry_msgs::PoseStamped\>("mavros/setp
10); \#发布mavros/setpoint_position/local话题消息，消息类型是PoseStamped，限制排队的消息数量

ros::ServiceClient arming_client = nh.serviceClient\<mavros_msgs::CommandBool\>
("mavros/cmd/arming"); \# 在mavros/cmd/arming服务可用之前一直阻塞

ros::Rate rate(20.0); \# 以 20 赫兹（20 次每秒）的频率运行代码

cmd.coordinate_frame = mavros_msgs::PositionTarget::FRAME_LOCAL_NED; \#选择控制坐标系，

cmd.type_mask = \~uint16_t(0); \# 所有位都取反，即设置为1，确保 type_mask
的所有位最初都被设置为 1

cmd.type_mask &= \~mavros_msgs::PositionTarget::FORCE; \# 将 type_mask 中与FORCE 对应的

ros::Time::now() - last_request \> ros::Duration(5.0) \#检查当前时间和上次请求时间之间的间

ros::spinOnce(); \# 在不阻塞其他任务的情况下，执行一次 ROS回调函数。注意如果用ros::spin()代替，

ROS_INFO("Offboard enabled"); \# 终端打印OFFBOARD enabled信息

set_mode_client.call(offb_set_mode) &&offb_set_mode.response.mode_sent \# 向/mavros/s
```

3）其余代码说明

```
ros = RflyRosStart.RflyRosStart(TargetID,TargetIP) #创建ros实例，也就是发起roscore

mavros_msgs::PositionTarget cmd; \# cmd设置为PositionTarget消息类型

mavros_msgs::SetMode offb_set_mode \# offb_set_mode设置为SetMode 消息类型
```

2.实验效果

3.文件目录

例程目录：
[安装目录]\RflySimAPIs\8.RflySimVision\0.ApiExps\2-DistributedSimAPI\3.RosDistCtrl\1.MavrosCtrlC++

文件夹/文件名称	说明
ros1_node	ROS1 版本控制测试文件夹
Ros2_node	ROS2 版本控制测试文件夹
RflySim.bat	软件在环一键启动脚本
Python38Run.bat	Windows下Python程序运行脚本
WinWSL.bat	WSL1/Ubuntu 20.04环境程序运行脚本
WslGUI.bat	WSL1/Ubuntu 20.04可视化界面脚本

4.运行环境

4.1 软件要求

Windows 10及以上版本；RflySim工具链；Visual Studio Code；Linux（Ubuntu 20.04）；Linux（Ubuntu 20.04）。

> ①：若使用Pixhawk 6X飞控，平台安装时的编译命令为：px4_fmu-v6x_default，推荐PX4固件版本为：1.12.3。其他配套飞控及编译命令请见：https://rflysim.com/doc/zh/1/Hardware.html

4.2 硬件要求

笔记本/台式电脑① 1台；WinWSL 1台；虚拟机/视觉盒子/其他板卡 可选台。

> ①：推荐配置请见：https://rflysim.com/doc/zh/HowToInstall.pdf

5.实验步骤

5.1. 必做实验：WinsWSL控制

Step 1：WSL里编译工作空间

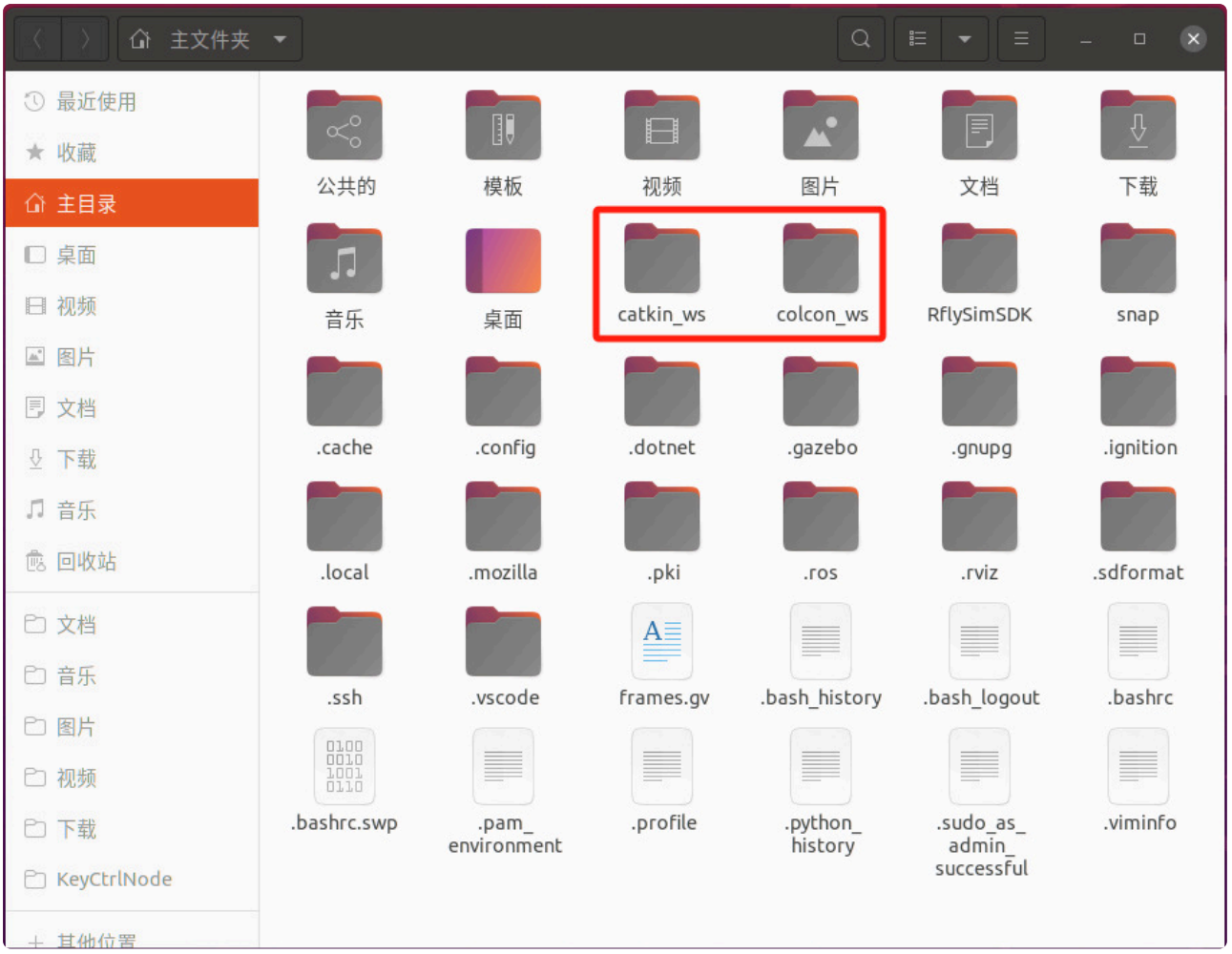
在ubuntu20.04里分别构建ros1和ros2的工作空间。

双击打开WslGUI.bat，启动WSL可视化界面。

打开终端运行命令:

```
mkdir -p \~/catkin_ws/src
```

```
mkdir -p \~/colcon_ws/src
```



2.把文件ros1_node 拷贝到 ~/catkin_ws/src里面，把 ros2_node 拷贝到~/colcon_ws/src里面；

3.编译：

编译之前请确保WSL中的ROS环境为ROS Noetic，可运行"*\桌面\RflyTools\RosSwitch.lnk"进行切换。

打开终端进入到~/catkin_ws 目录下，运行命令 `catkin_make` 进行编译;

打开终端进入到~/colcon_ws 目录下，运行命令 `colcon build` 编译。

Step 2：开启仿真和运行控制程序

ROS1 版本控制测试：

1.在 windows端双击运行RflySim.bat脚本，等待CopterSim日志打印初始化完成。

2.打开终端进入到ros1_node目录下，运行`python3 movros.py`程序启动mavros。

3.打开终端，切换到~/catkin_ws目录下，运行`source devel/setup.bash`,再运行`roslaunch ros1_node ros1_node.launch`，即可在RflySim3D里面看到飞机的飞行效果。

ROS2 版本控制测试：

1.在 windows端双击运行RflySim.bat脚本，等待CopterSim日志打印初始化完成。

2.打开终端进入到ros2_node目录下，运行`python3 movros.py`程序启动mavros。

3.打开终端，切换到~/colcon_ws目录下，运行`source install/setup.bash`，再运行`ros2 launch ros2_node ros2_node.launch`，即可在RflySim3D里面看到飞机的飞行效果。

5.2. 选作实验

准备工作：

虚拟机或NX的配置方法是相同的。

1）Ubuntu虚拟机环境下，进行分布式联机实验。先参考
[安装目录]\RflySimAPIs\8.RflySimVision\0.ApiExps\0.Preparation\1.VMwareUbuntu\Readme.pdf
，完成虚拟机的下载与配置。

2）用第二台Ubuntu电脑或NX板卡，实现联机实验。其他Ubuntu电脑的配置，先看

[安装目录]\RflySimAPIs\8.RflySimVision\0.ApiExps\0.Preparation\2.GenenralUbuntuConfig\Readme.pdf
；NX板卡的配置方法，先看
[安装目录]\RflySimAPIs\8.RflySimVision\0.ApiExps\0.Preparation\3.NXwithPX4Config\Readme.pdf
。

扩展实验：

5.2.1在虚拟机/视觉板卡/另一台Ubuntu上接收图像实验

Step 1：虚拟机里编译工作空间

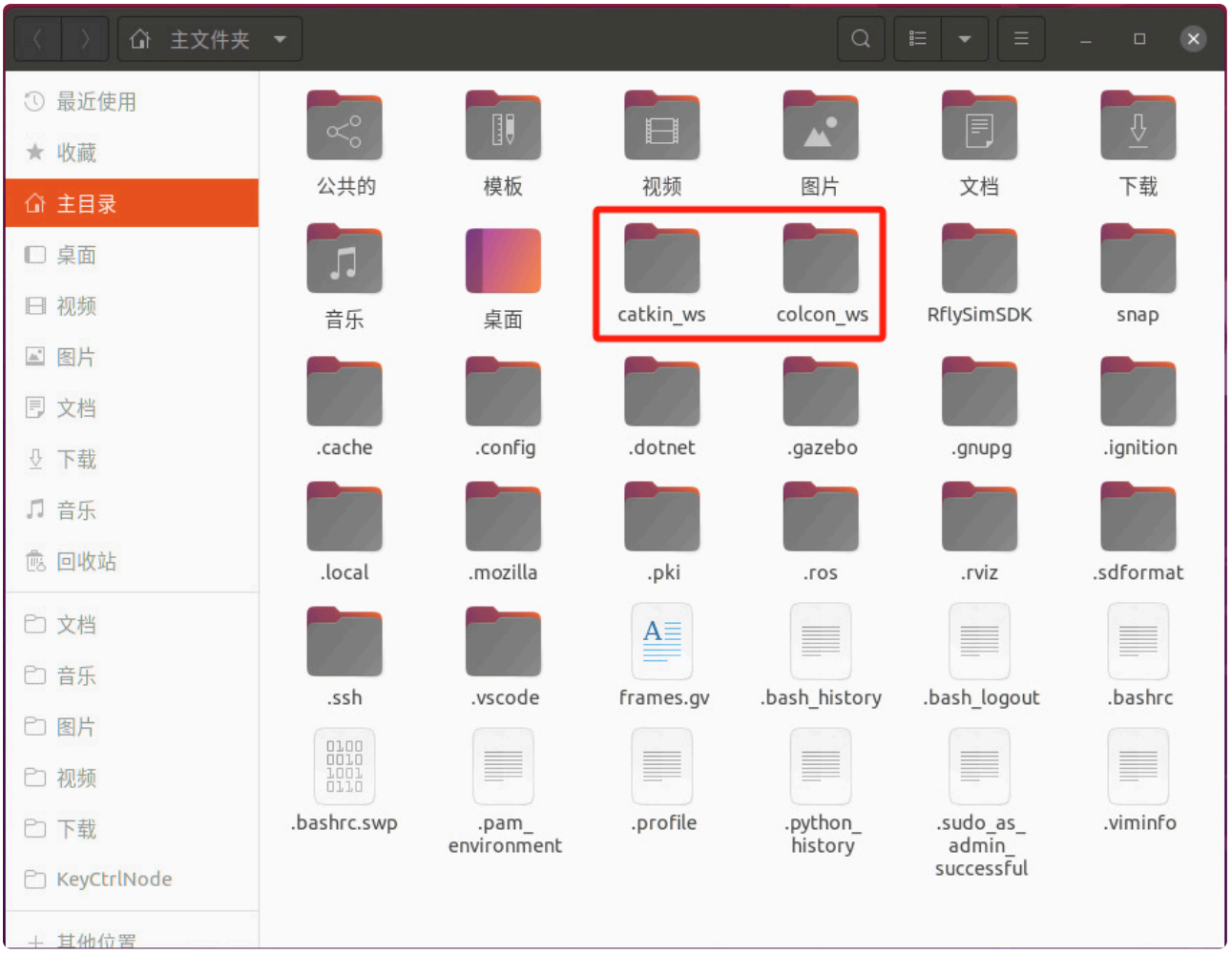
1.在ubuntu20.04里构建工作空间。

打开虚拟机，进入虚拟机界面。

打开终端运行命令:

```
mkdir -p \~/catkin_ws/src
```

```
mkdir -p \~/colcon_ws/src
```



2.把文件ros1_node 拷贝到 ~/catkin_ws/src里面，把 ros2_node 拷贝到~/colcon_ws/src里面；

3.编译：

打开终端进入到~/catkin_ws 目录下，运行命令`catkin_make`进行编译;

打开终端进入到~/colcon_ws 目录下，运行命令`colcon build`编译。

Step 2：运行控制程序

步骤2同上面的Step2步骤。

6.参考文献

无

7.常见问题

Q1：无

A1：无