

1. 实验名称及目的

1.1 实验名称

Python Mavros控制实验

1.2 实验目的

展示用Python mavros进行飞机控制。

1.3 关键知识点

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。

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）

```
1 req = ReqCopterSim.ReqCopterSim() \# 获取局域网内所有CopterSim程序的电脑IP列表
2
3 TargetIP = req.getSimIpID(StartCopterID) \#自动获取CopterSim的StartCopterID号程序所在电脑
4
5 req.sendReSimIP(CopterID) \# 请求mavlink数据到本电脑
6
7 req.sendReSimUdpMode(TargetID,2) \# 强制切换MAVLINK_FULL
```

2) rospy接口使用

```
1 | rospy.init_node("offb_node_py") \# 创建节点
2 |
3 | state_sub = rospy.Subscriber("mavros/state", State, callback=state_cb) \#订阅mavros/st
4 |
5 | local_pos_pub = rospy.Publisher("mavros/setpoint_position/local", PoseStamped,
6 | queue_size=10) \#发布mavros/setpoint_position/local话题消息, 消息类型是PoseStamped, 限制排
7 |
8 | rospy.wait_for_service("/mavros/cmd/arming") \# 等待/mavros/cmd/arming服务
9 |
10 | arming_client = rospy.ServiceProxy("mavros/cmd/arming", CommandBool) \#在mavros/cmd/a
11 |
12 | rate = rospy.Rate(20) \# 以 20 赫兹 (20 次每秒) 的频率运行代码
13 |
14 | local_pos_pub.publish(pose) \#发布pose消息, 这个消息属于前面说的mavros/setpoint_position/l
15 |
16 | rospy.loginfo("OFFBOARD enabled") \# 终端打印OFFBOARD enabled信息
17 |
18 | set_mode_client.call(offb_set_mode).mode_sent \# 向 /mavros/set_mode服务发送请求, 尝试将
```

3) 其余代码说明

```
1 | ros = RflyRosStart.RflyRosStart(TargetID,TargetIP) \#创建ros实例, 也就是发起roscore
2 |
3 | offb_set_mode = SetModeRequest() \# offb_set_mode设置为SetModeRequest()服务类型
```

3. 实验效果



4. 文件目录

例程目录：

[安装目录]\RflySimAPIs\8.RflySimVision\0.ApiExps\2-DistributedSimAPI\2.UavDistCtrl
\3.MavRosPyApiTest

文件夹/文件名称	说明
MavRosPyApiTest.bat	软件在环一键启动脚本
mavros_offboard_posctl_test.py	Mavros控制offboard例程
mavrostest.py	Mavros启动程序
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

5. 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。

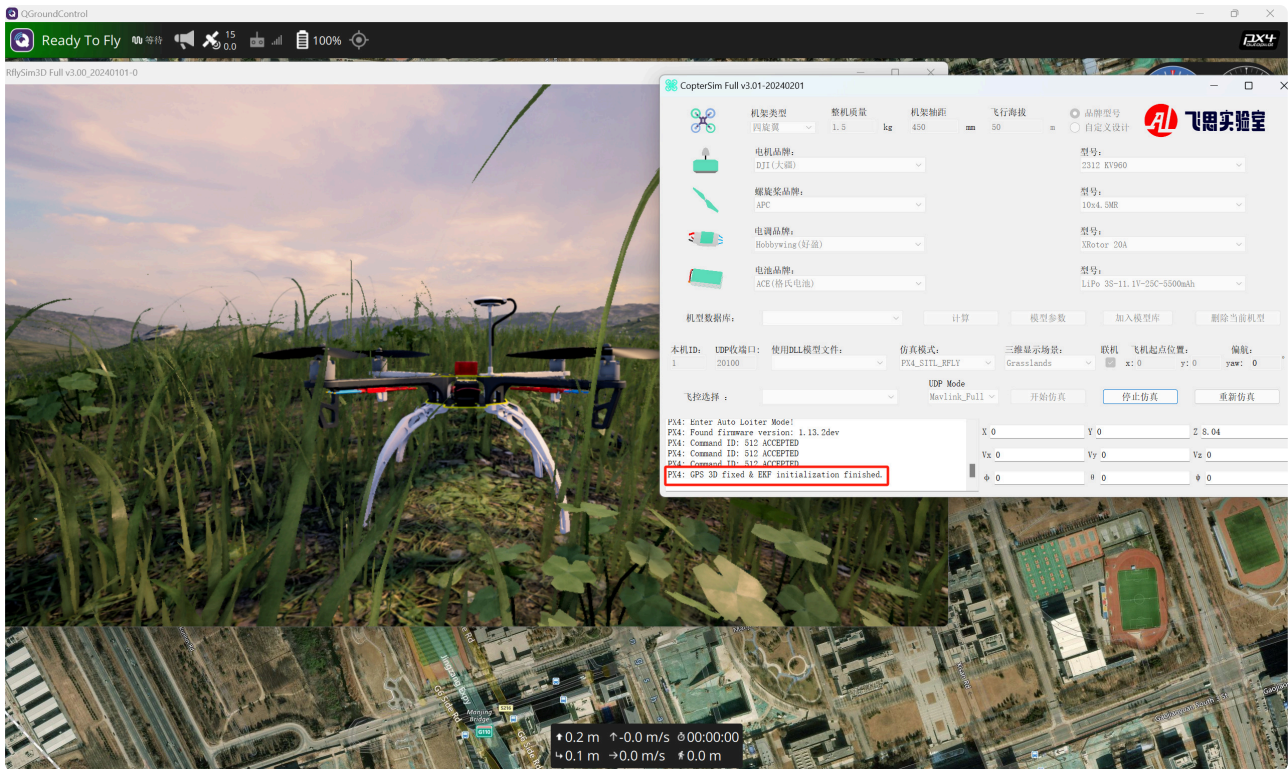
①：推荐配置请见：<https://rflysim.com/doc/zh/HowToInstall.pdf>

6. 实验步骤

5.1 必做实验：WinsWSL控制

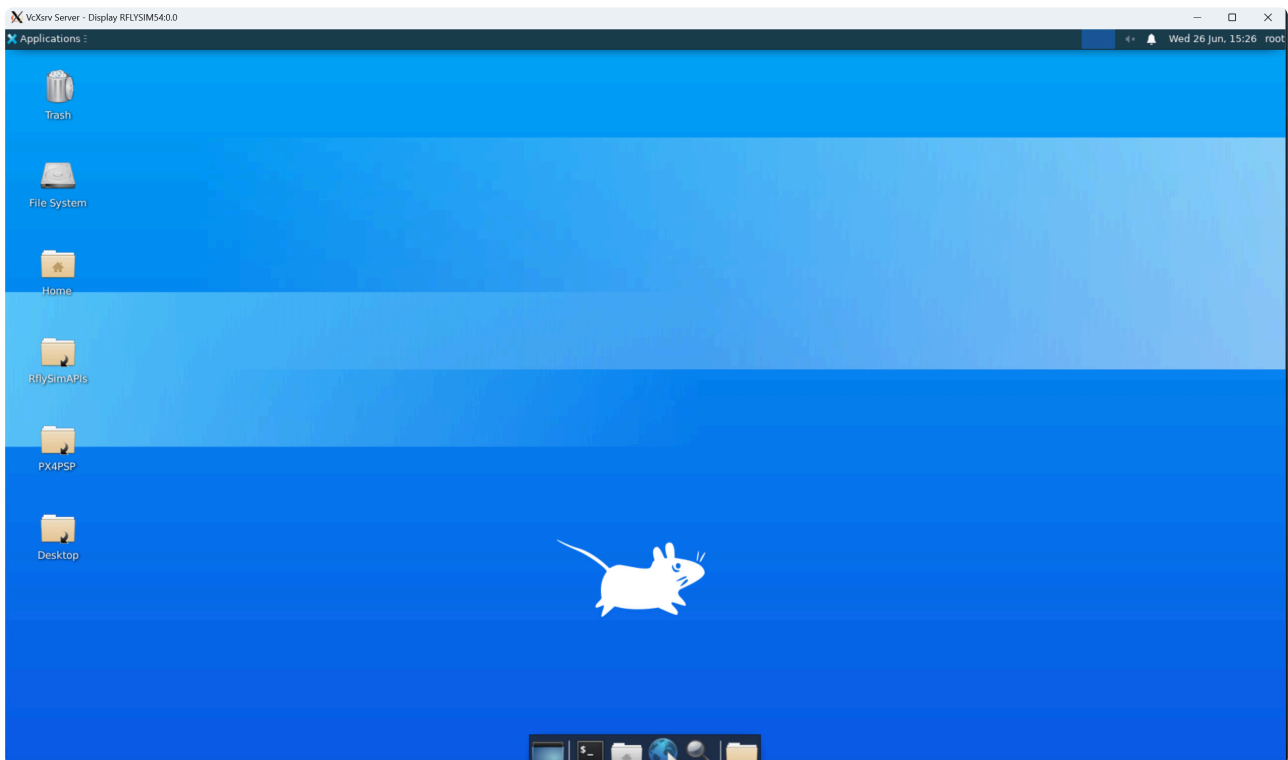
Step 1：开启仿真

在Windows主机中，双击运行 [MavRosPyApiTest.bat](#) 启动仿真界面。等待CopterSim界面中返回如下语句即可。



Step 2: 开启WSL可视化界面

双击打开 [WslGUI.bat](#)，启动WSL可视化界面。（注：如果打开发现窗口白屏，没有桌面，则关了重开一两次。）



注意：参考 [\[安装目录\]\RflySimAPIs\1.RflySimIntro\2.AdvExps\e7_WslGUI\Intro.pdf](#)，了解WslGUI的功能与使用。

Step 3: 运行控制程序

双击打开WinWSL.bat，运行命令 `python3 mavrotest.py` 可以看到mavros成功启动。

再新开一个WinWSL.bat窗口，用 `python3 mavros_offboard_posctl_test.py` 命令运行 `mavros_offboard_posctl_test.py` 可以看到飞机控制



备注：可以参考

[\[安装目录\]\RflySimAPIs\1.RflySimIntro\2.AdvExps\e8.WslVsCode\Intro.pdf](#) 来使用VS Code开发并调试Ubuntu下python文件。

```
server_ue4.py X
> Users > uavcs > Desktop > demo > 8.RflySimVision > 0.ApiExps > 0.Preparation > 5.ManModifyIPRun > server_ue4.py > ...
1
2 # import required libraries
3 # pip3 install pymavlink pyserial
4
5 import cv2
6 import numpy as np
7 import time
8 import VisionCaptureApi
9 import PX4MavCtrlV4 as PX4MavCtrl
10 import math
11
12 StartCopterID = 1 # 初始飞机的ID号
13 TargetIP = "192.168.31.141"# 手动修改为电脑主机的IP
14 # 注意：如果是本电脑运行的话，那TargetIP是127.0.0.1的本地地址；如果是远程访问，则是192打头的局域网地址。
15 # 因此本程序能同时在本机运行，也能在其他电脑运行。
16 vis = VisionCaptureApi.VisionCaptureApi(TargetIP)
17
18 # VisionCaptureApi 中的配置函数
19 vis.jsonLoad()
20 isSuss = vis.sendReqToUE4(
21     0, TargetIP
22 )
23 vis.startImgCap() # 开启取图循环，执行本语句之后，已经可以通过vis.Img[1]读取到图片了
24 print('Start Image Receiver')
25 #vis.sendImuReqCopterSim(StartCopterID, TargetIP) # 发送请求，从目标飞机CopterSim读取IMU数据，回传地址为127.0.0.1。默认频率为200Hz
26 # 执行本语句之后，会自动开启数据监听，已经可以通过vis.imu读取到IMU数据了。
27
28 VehicelNum = 1
29 MavList=[]
30 # Create MAV instance
31 for i in range(VehicelNum):
32     CopterID=StartCopterID+i # 当前配置的飞机序号
33
34     time.sleep(1)
35     MavList = MavList+[PX4MavCtrl.PX4MavCtrlIer(CopterID,TargetIP)] # 初始化并建立i号飞机的MAVLink通信连接
36
37
```

Run Python File
Run Python File in Dedicated Terminal
Python 调试程序, 调试 Python 文件
Python 调试程序, 使用 launch.json 进行测试

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.GeneralUbuntuConfig\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：

6.2.1 在虚拟机/视觉板卡/另一台Ubuntu上接收图像实验

Step 1: 开启仿真

步骤1同上面的Step1步骤。

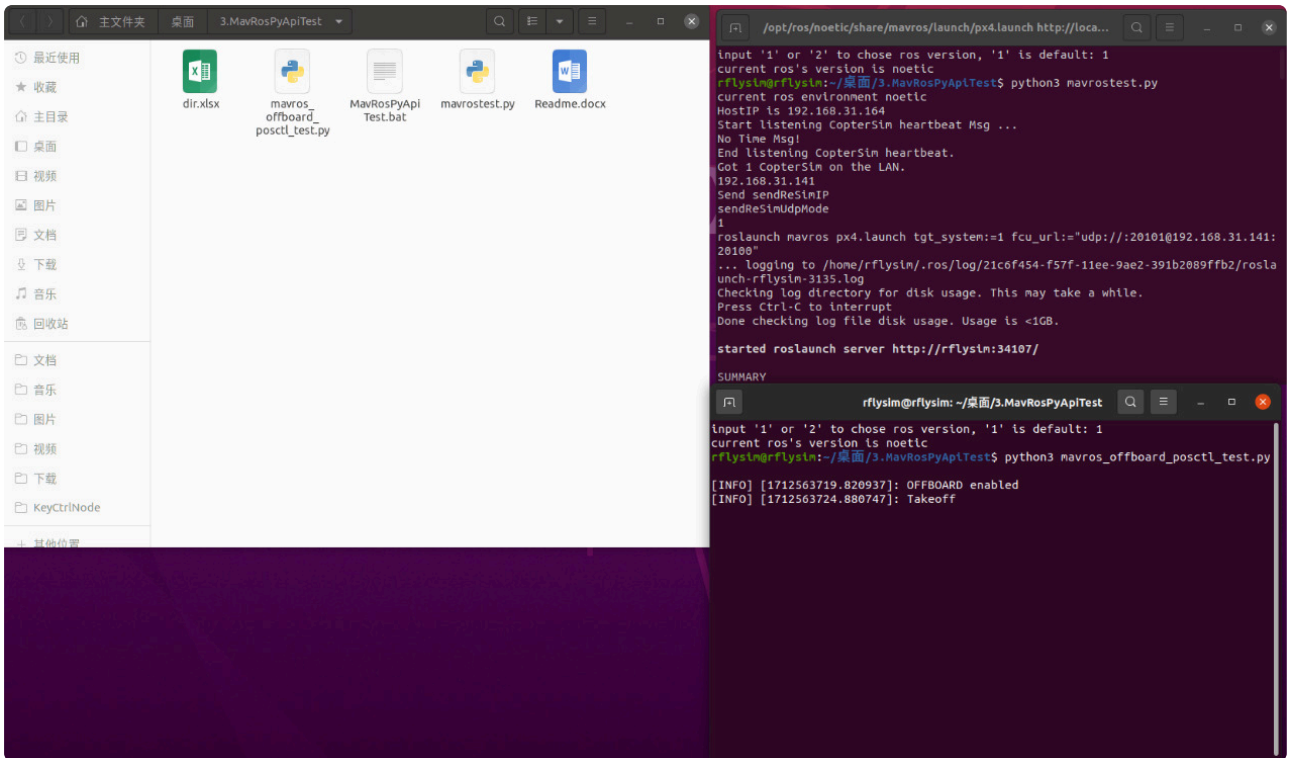
Step 2: 运行控制程序

将整个文件夹拷贝到Linux下，在虚拟机中，先用 `python3 mavrostest.py` 命令运行

`mavrostest.py` 可以看到mavros成功启动，再

用 `python3 mavros_offboard_posctl_test.py` 命令运行

`mavros_offboard_posctl_test.py` 可以看到飞机控制。



6.参考资料

无

7.常见问题

Q1: 无

A1: 无