1.实验名称及目的

1.1 实验名称

ROS订阅得到图像数据实验

1.2 实验目的

通过ROS订阅得到RflySIm的图像数据。

1.3 关键知识点

本实验主要是实现通过Python接口ReqCopterSim.py使用（见RflySimAPIs\RflySimSDK\ctrl目录）自动获取的IP，去建立远端电脑与本机RflySim3D（发送图片）与CopterSim（接收控制指令）的联机仿真，同时通过ROS订阅得到图像数据。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Config.json中，SendProtocol[0]设置为了1，即UDP压缩传图模式。

关键知识点1：SendProtocol[0]决定了图像的传出模式。SendProtocol[0]=0：共享内存（仅限Windows下获取图像），1：UDP直传png压缩，2：UDP直传图片不压缩（只适用图片类传感器），3：UDP直传jpg压缩（只适用图片类传感器）。如果是激光雷达数据只有0或1（共享内存和UDP网络传输）。

关键知识点2：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）视觉接口使用

```
vis.jsonLoad(3) \# \# 使用jpeg压缩方式加载Config.json中的传感器配置文件

isSuss = vis.sendReqToUE4() \# 向RflySim3D发送取图请求，发给ip为TargetIP的地址

vis.startImgCap() \# 开启取图

vis.hasData[i] \# 图片i数据是否更新

vis.Img[i] \# 图片i数据（像素矩阵）

cv2.imshow('Img'+str(i),vis.Img[i]) \# 显示图片i图像
```

2） 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

req.sendReSimIP(1) \# 请求mavlink数据到本电脑

req.sendReSimUdpMode(1,2) \# 强制切换MAVLINK_FULL

coptersim_ip = req.getSimIpID(1) #自动获取CopterSim的1号程序所在电脑的IP，作为目标IP。这里获

vis = VisionCaptureApi.VisionCaptureApi(coptersim_ip) #创建一个视觉传感器实例，这个实例对应
```

3）相机数量和参数配置

其中，视觉传感器的初始状态由本文件夹下的Config.json决定，主要包含以下配置项：

```
"SeqID":0：使用自动更新ID的方式，创建了SeqID为0的视觉传感器

"TypeID":1：传感器类型为RGB彩色图像

"TargetCopter":1：相机绑定在1号飞机上

"SendProtocol":[1,0,0,0,0,0,0,0]：传输模式为1：UDP网络传输模式（图片使用jpeg压缩，点云直传）。

"SensorPosXYZ":[0,0,-0.5]：相机分布位置。
```

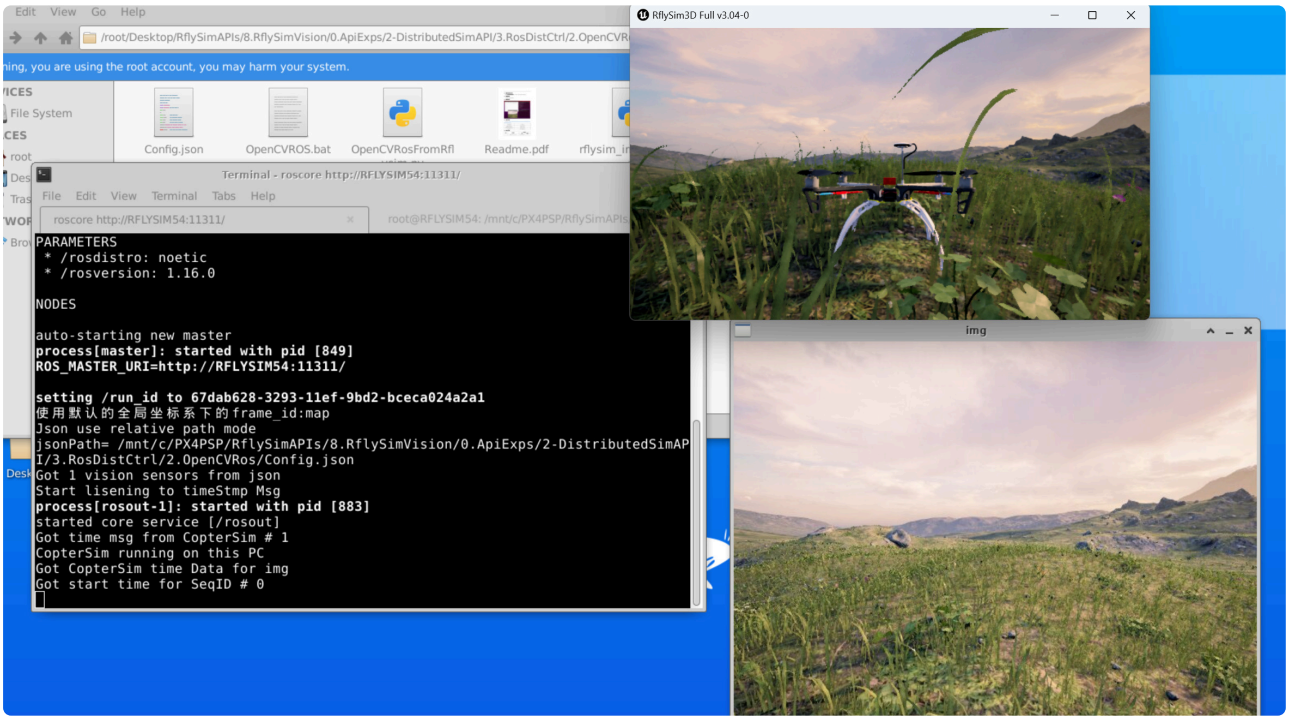
4）rospy接口使用

```
rospy.init_node('OpenCVROS',anonymous= True) #创建节点，设置anonymous=True时，ROS会在节点
rospy.Subscriber("/rflysim/sensor0/img_rgb",sensor.Image,ImgCallBack) #订阅/rflysim/se
rospy.spin() \# 进入一个循环，保持ROS节点运行，以便处理回调函数
```

5）其余代码说明

```
encoding_ ="bgr8" # 定义图像编码类型为BGR8
cv_bidge = CvBridge() # 创建CvBridge对象，用于图像消息与OpenCV图像之间的转换
img = cv_bidge.imgmsg_to_cv2(data,encoding_) #将ROS图像消息data转换为OpenCV格式的图像img,
```

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\2.OpenCVRos

文件夹/文件名称	说明
OpenCVROS.bat	启动仿真配置文件
Rflysim_img_node.py	Python实验程序
OpenCVRosFromRflysim.py	Python实验程序
Config.json	视觉传感器配置文件
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。

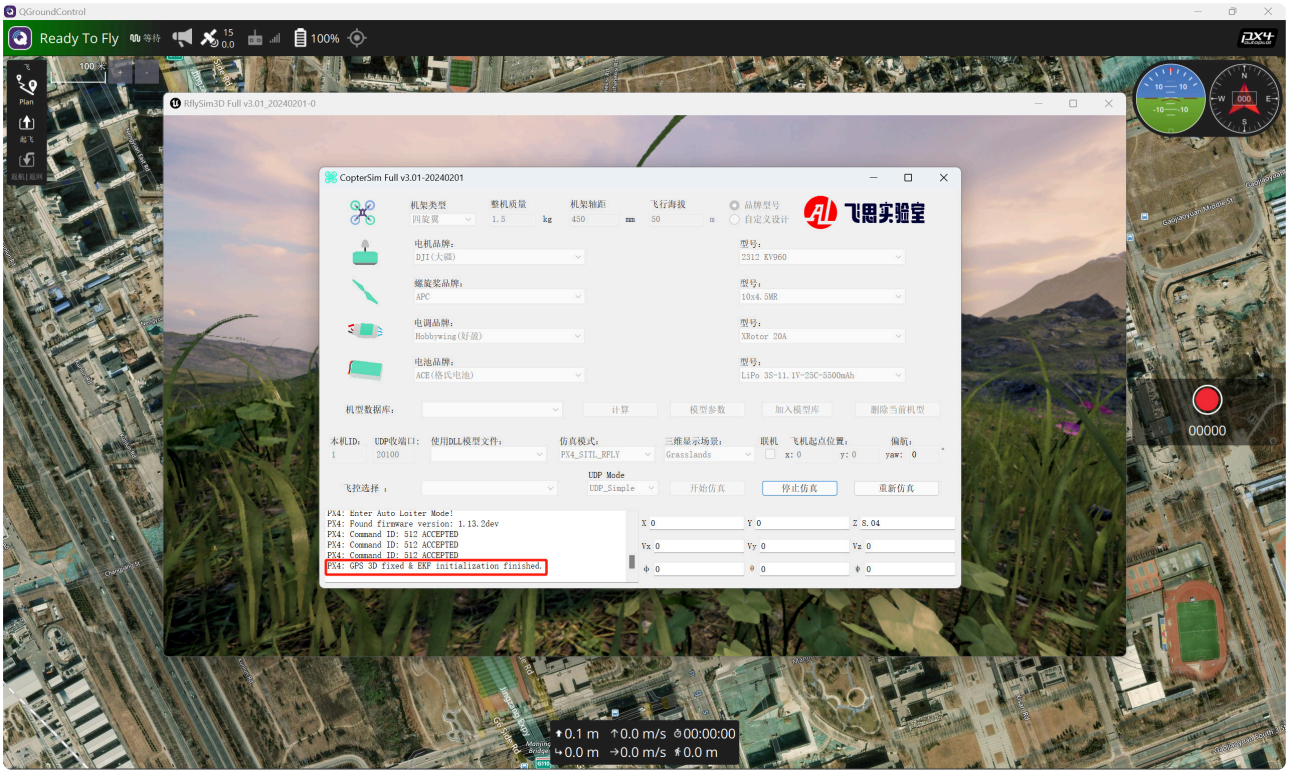
> ①：推荐配置请见：https://rflysim.com/doc/zh/HowToInstall.pdf

5.实验步骤

5.1 必做实验：WinsWSL控制

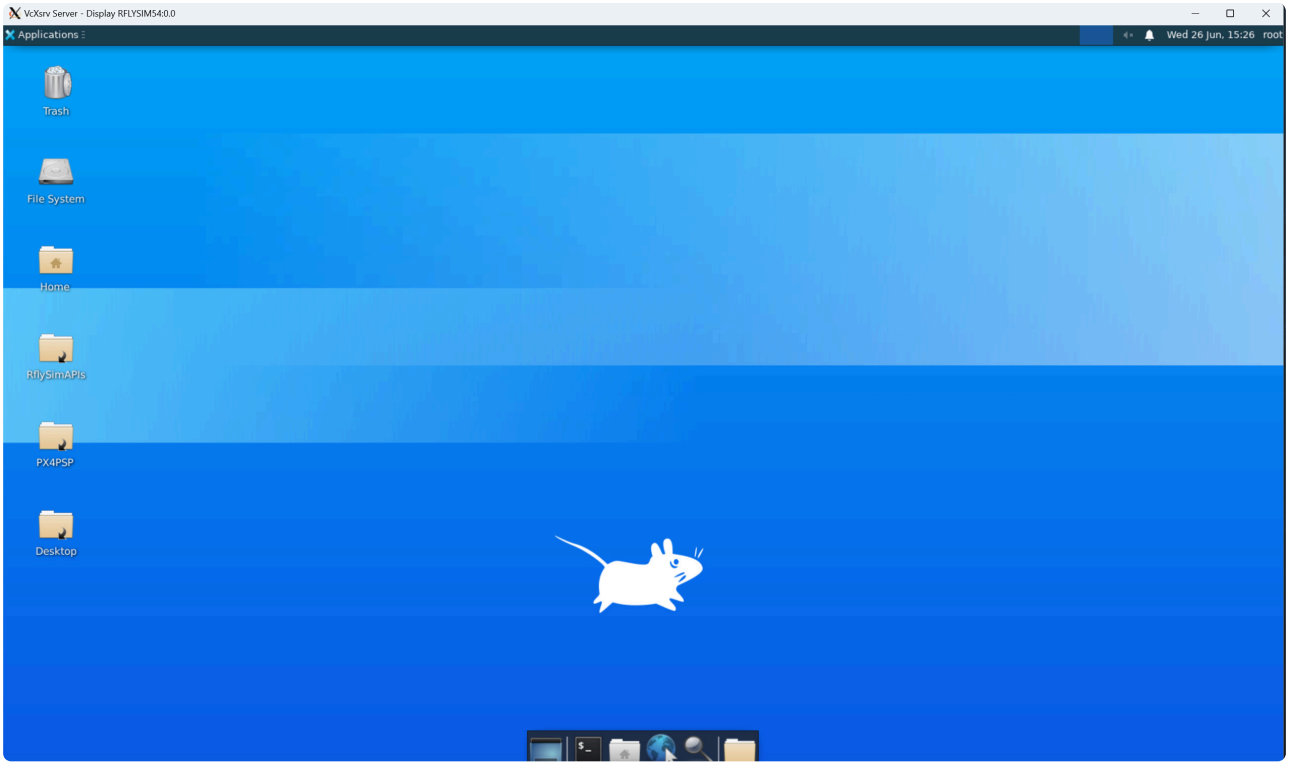
Step 1：开启仿真

在windows双击运行OpenCVROS.bat开启一个软件在环仿真，等待CopterSim日志打印初始化完成。



Step 2：开启WSL可视化界面

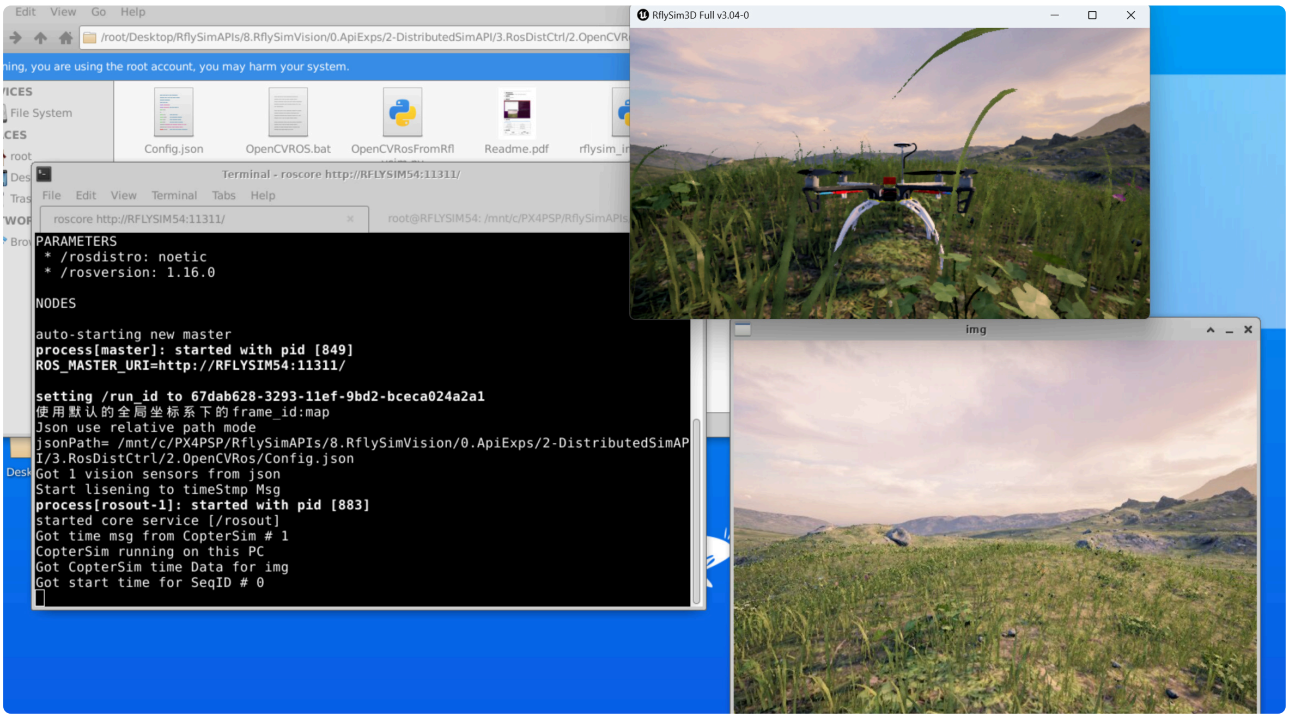
双击打开WslGUI.bat，启动WSL可视化界面。（注：如果打开发现窗口白屏，没有桌面，则关了重开一两次。）



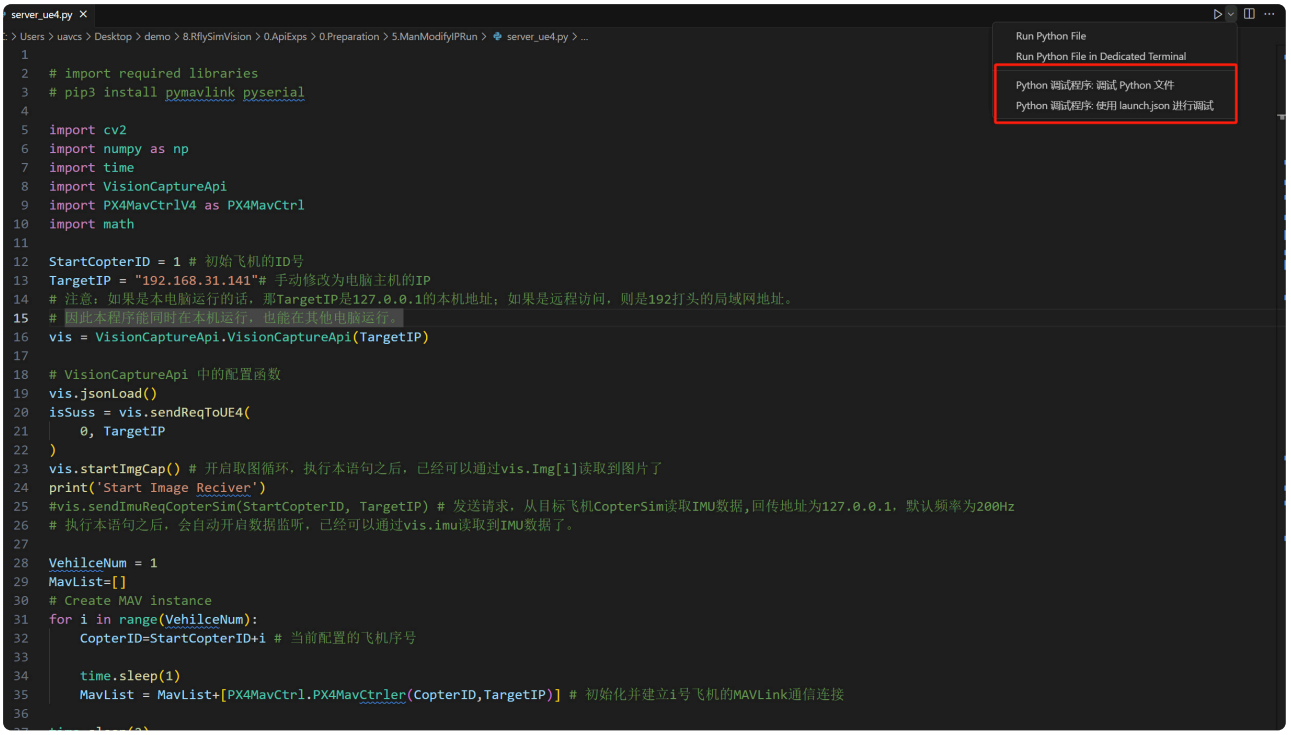
注意：参考 [安装目录]\RflySimAPIs\1.RflySimIntro\2.AdvExps\e7_WslGUI\Intro.pdf，来了解WslGUI的功能与使用。

Step 3：运行控制程序

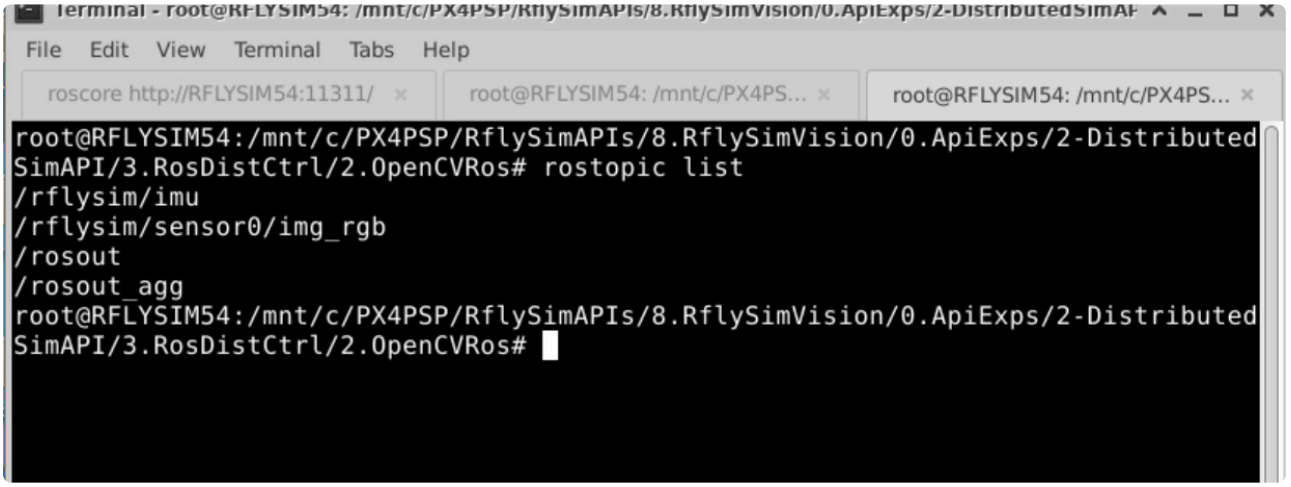
双击打开WinWSL.bat，运行命令 `python3 rflysim_img_node.py` 运行Rflysim_img_node.py程序，可以看到提示，启动了ROS、识别到了1个传感器数据，并和CopterSim建立通信。



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。

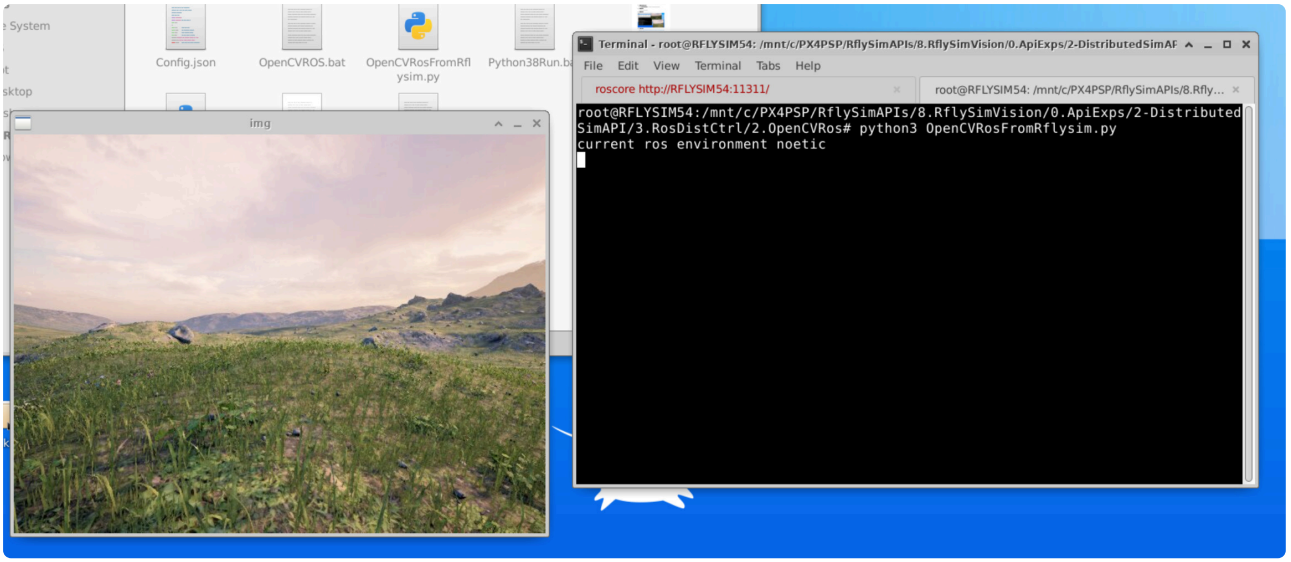


在WinWSL的可视化Ubuntu界面中，打开新终端，在终端输入`rostopic list`



可以看到，发出了1路（sensor0的名称）彩色图像（img_rgb的名称）。之后，可以订阅/rflysim/sensor0/img_rgb的ROS消息，并进行Opencv的图像处理。

再打开一个新终端，输入`python3 OpenCVRosFromRflysim.py`运行OpenCVRosFromRflysim.py程序，可以看到图像。



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：

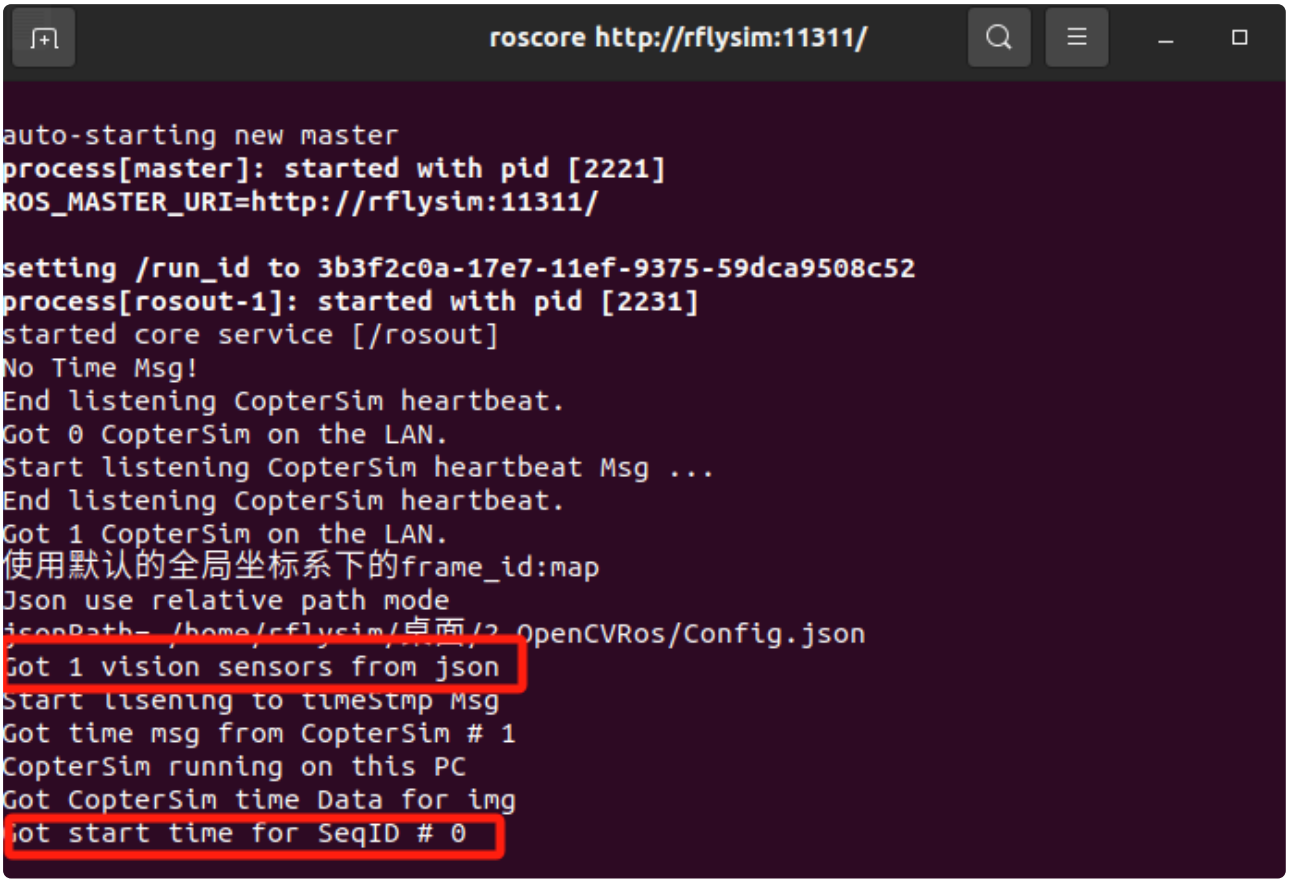
5.2.1在虚拟机/视觉板卡/另一台Ubuntu上接收图像实验

Step 1：开启仿真

步骤1同上面的Step1步骤。

Step 2：运行控制程序

拷贝整个文件夹到虚拟机（或Ubuntu电脑/板卡），用ROS1环境输入`python3 rflysim_img_node.py`运行Rflysim_img_node.py程序，可以看到提示，启动了ROS、识别到了1个传感器数据，并和CopterSim建立通信。



通过观察config.json文件可知，本例程创建了1个RGB相机。

Config.json

C: > Users > dream > Desktop > 2.OpenCVRos > {} Config.json > [ ] VisionSensors > {} 0 >

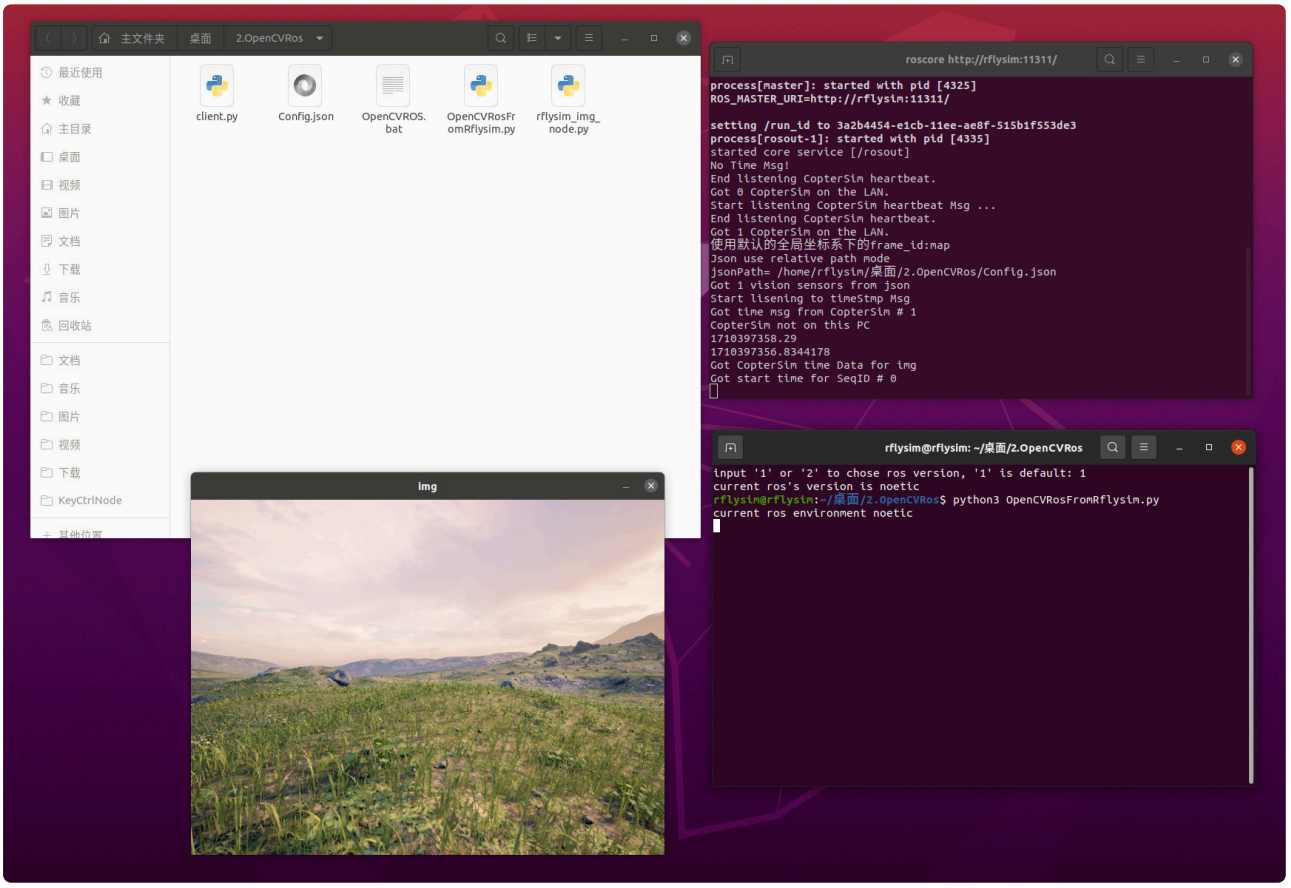
```
{
    "VisionSensors":[
        {
            "SeqID":0,
            "TypeID":1,
            "TargetCopter":1,
            "TargetMountType":0,
            "JointType":1,
            "DataWidth":640,
            "DataHeight":480,
            "DataCheckFreq":30,
            "SendProtocol":[1,127,0,0,1,9999,0,0],
            "CameraFOV":90,
```

传感器0
RGB图像
绑定在1号飞机上
其他相机参数

在Ubuntu终端中，输入`rostopic list`

可以看到，发出了1路（sensor0的名称）彩色图像（img_rgb的名称）。之后，可以订阅/rflysim/sensor0/img_rgb的ROS消息，并进行Opencv的图像处理。

再打开一个新终端，输入`python3 OpenCVRosFromRflysim.py`运行OpenCVRosFromRflysim.py程序，可以看到图像。（用R0S2环境实验步骤一样）



6.参考资料

无

7.常见问题

Q1：无

A1：无