1.实验名称及目的

1.1 实验名称

NX与Pixhawk6x联合硬件在环仿真

1.2 实验目的

实现NX与Pixhawk6x联合硬件在环仿真，控制飞机进行穿环。

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, 使用mavros 去控制飞机，C++实现穿环。关键代码解析如下：

本例子和其他分布式例子的区别，主要在于进行了更加复杂的控制，结合opencv库，实现飞机穿环功能，整个流程主要是启动mavros，开启图像读取，从图像中选点作为飞机穿环的位置，从而最终实现穿环。

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":[0.3,0,0]：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/s
rospy.spin() \# 进入一个循环，保持ROS节点运行，以便处理回调函数
```

5）vision_ctrl.cpp重要部分代码说明

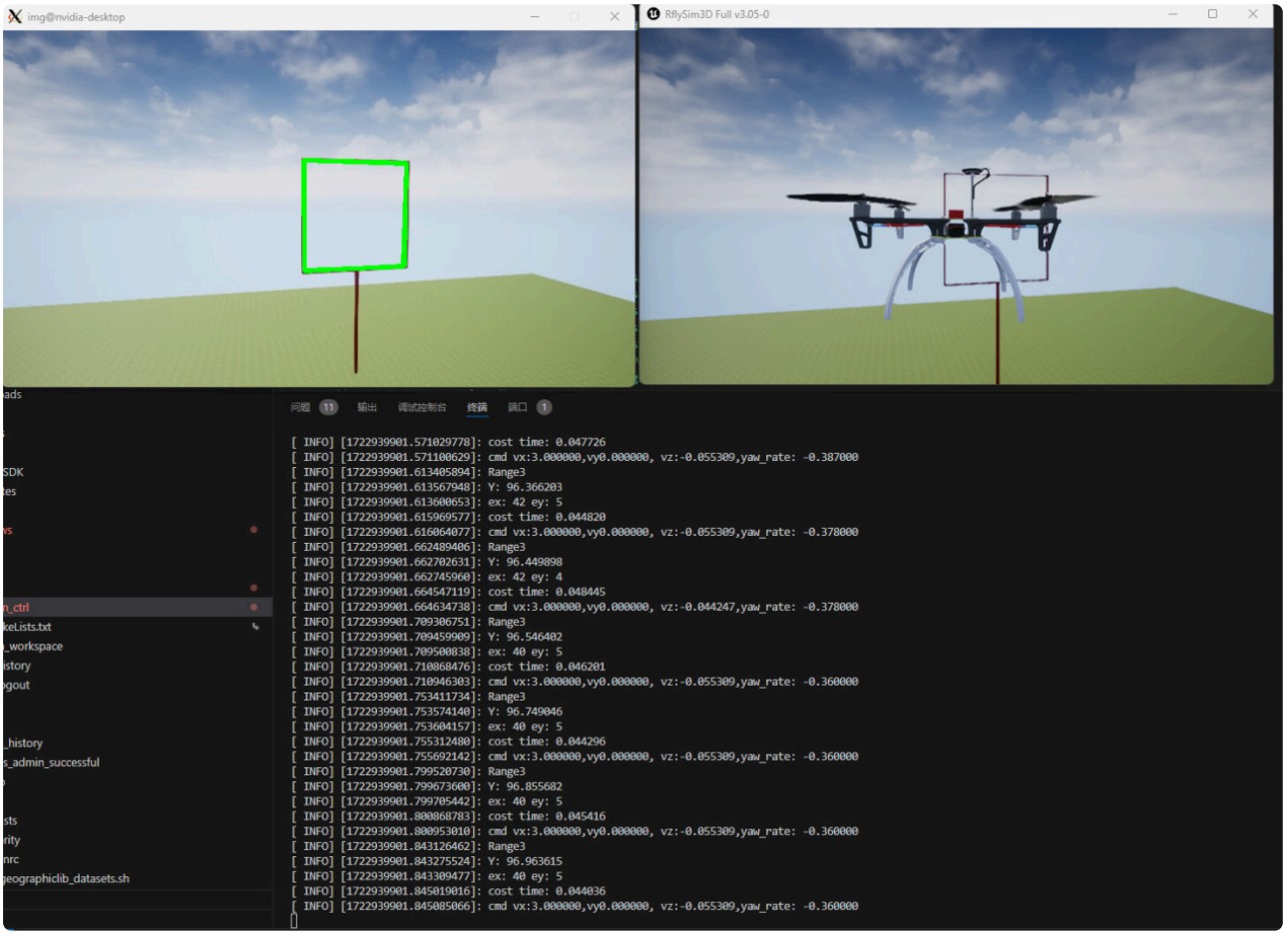
```
213     if(ex != -1)
214     {
215         cmd.velocity.x = sat(ros::Time::now().toSec() - start_time.toSec(),3);
216         cmd.velocity.y =0 ;
217         cmd.velocity.z = -K_z * ey;
218         cmd.yaw_rate = -K_yawrate *ex;
219         cmd_pub.publish(cmd);
220         ros::spinOnce();
221 
222     }
```

这一部分就是当飞机检测到目标（环或者框），选择中心点，给予飞机速度控制的代码。通过修改系数K_z和K_yawrate来改变飞机上升以及偏转的幅度，从而控制飞机成功穿环。

6）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\0.Preparation\3.NXwithPX4Config\0.PX4+NX

文件夹/文件名称	说明
PX4+NX_SITL	启动软件仿真配置文件
PX4+NX_HITL	启动硬件仿真配置文件
Config.json	视觉传感器配置文件
vision_ctrl	控制功能包
rflysim_img_node	Python实验程序
server_ue4_Serial	Python实验程序
mavros	Python实验程序

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）。

> 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；Orin nx 1台。

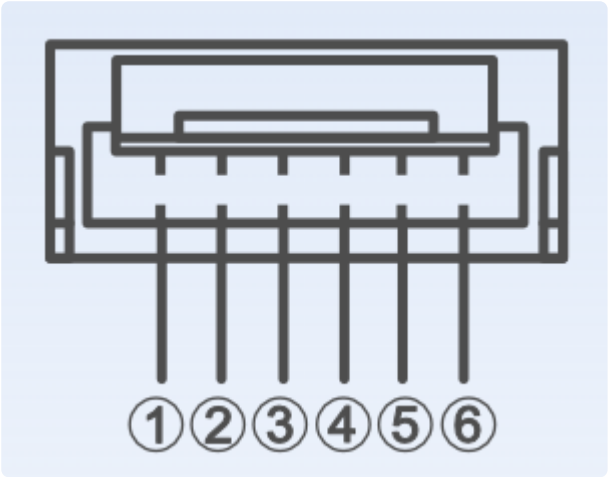
> ①：推荐配置请见：https://rflysim.com/doc/zh/HowToInstall.pdf

5.实验步骤

5.1 必做实验：硬件在环穿环控制

Step 1：飞控基本配置

1.首先连接NX的电源线，然后将飞控与NX使用根据官方文档提供的接口线序自制的线材连接到TELEM1口。

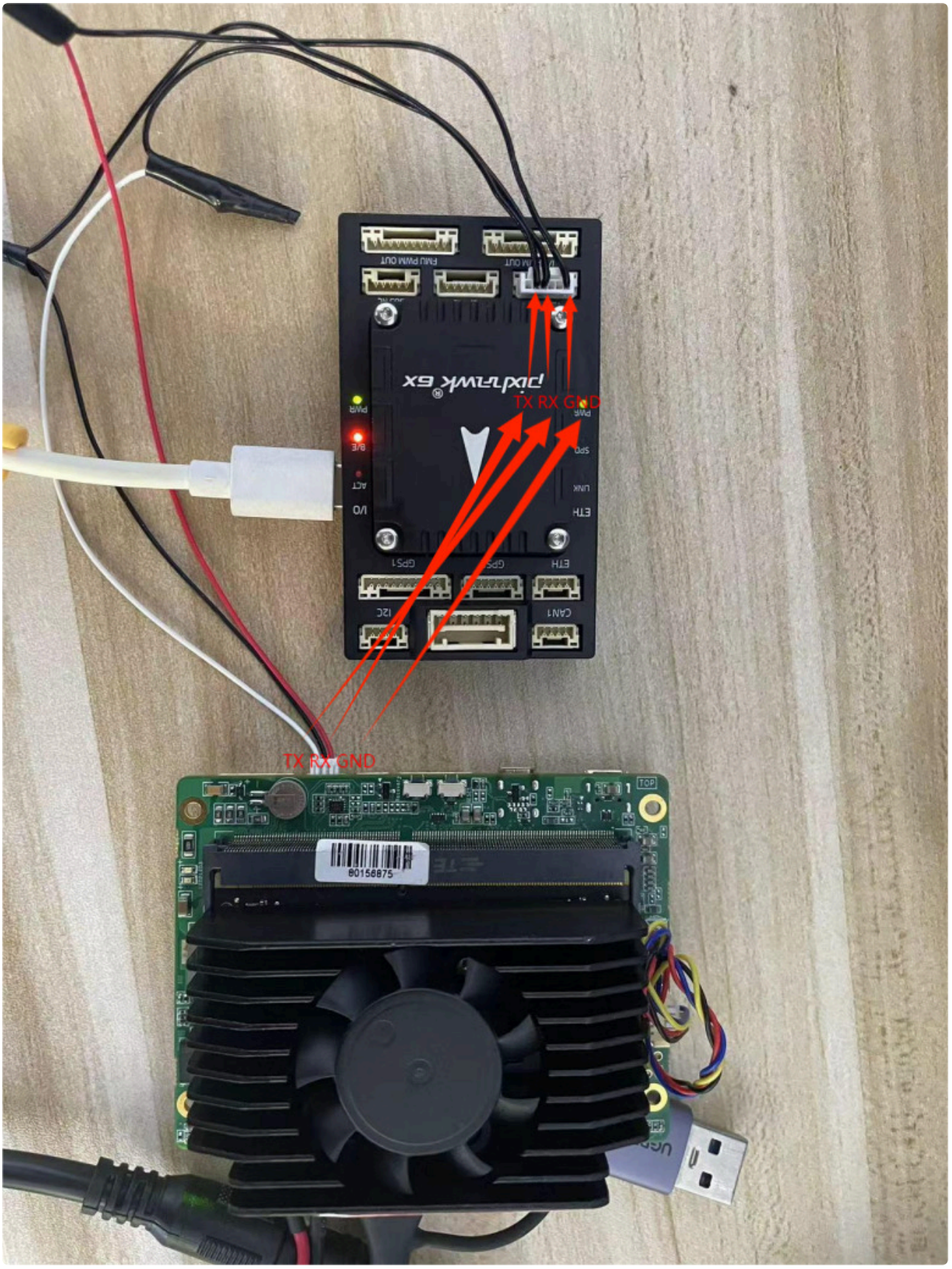


Telem1, Telem2, Telem3 ports

Pin	Signal	Volt
1(red)	VCC	+5V
2(black)	TX7/5/2 (out)	+3.3V
3(black)	RX7/5/2 (in)	+3.3V
4(black)	CTS7/5/2 (in)	+3.3V
5(black)	RTS7/5/2 (out)	+3.3V
6(black)	GND	GND

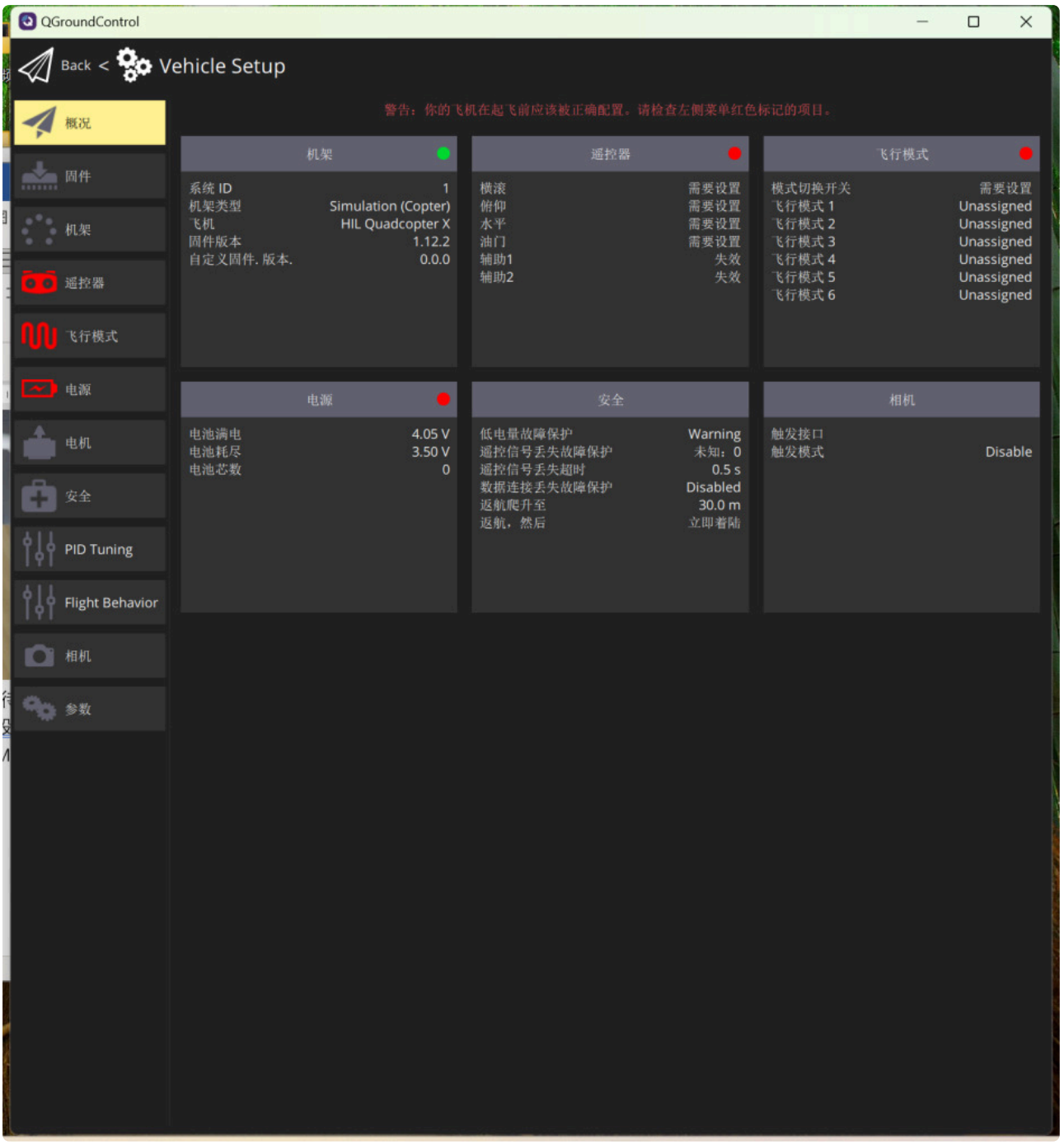

VCC TX RX CTS RTS GND
注意线序


Netac
NVMe SSD
WARRANTY VOID IF ANY LABEL REMOVED
M.2 2242 PCIe Gen3*2
Model:NS128GSSD510
128GB
SN:W4SI005034T
18000156 04/2023
QC PASS
MADE IN CHINA
注意线序
JXD 2328Q
G2436S
TX RX GND

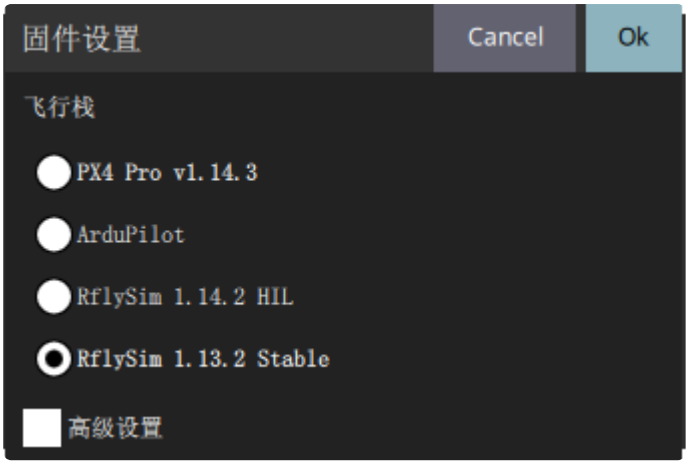


如上图所示，TELEM1口从左到右为VCC TX RX CTS RTS GND,本实验采用串口通信的方式将NX与飞控进行连接，因此我们需要将NX的TX接到TELEM1口的RX上，再将NX的RX接到TELEM1口的TX上，最后把GND连接到一起即可。（一定要注意接线顺序若接反将飞控TELEM1口的VCC接到NX的GND上会导致NX损坏）

2. 使用USB线连接电脑与飞控的Type-c口并打开QGC会自动跳转到下面界面（如果未出现跳转，多次插拔重新连接）。



3. 进入固件烧录模式，选择Rflysim 1.13.2 Stable版本。



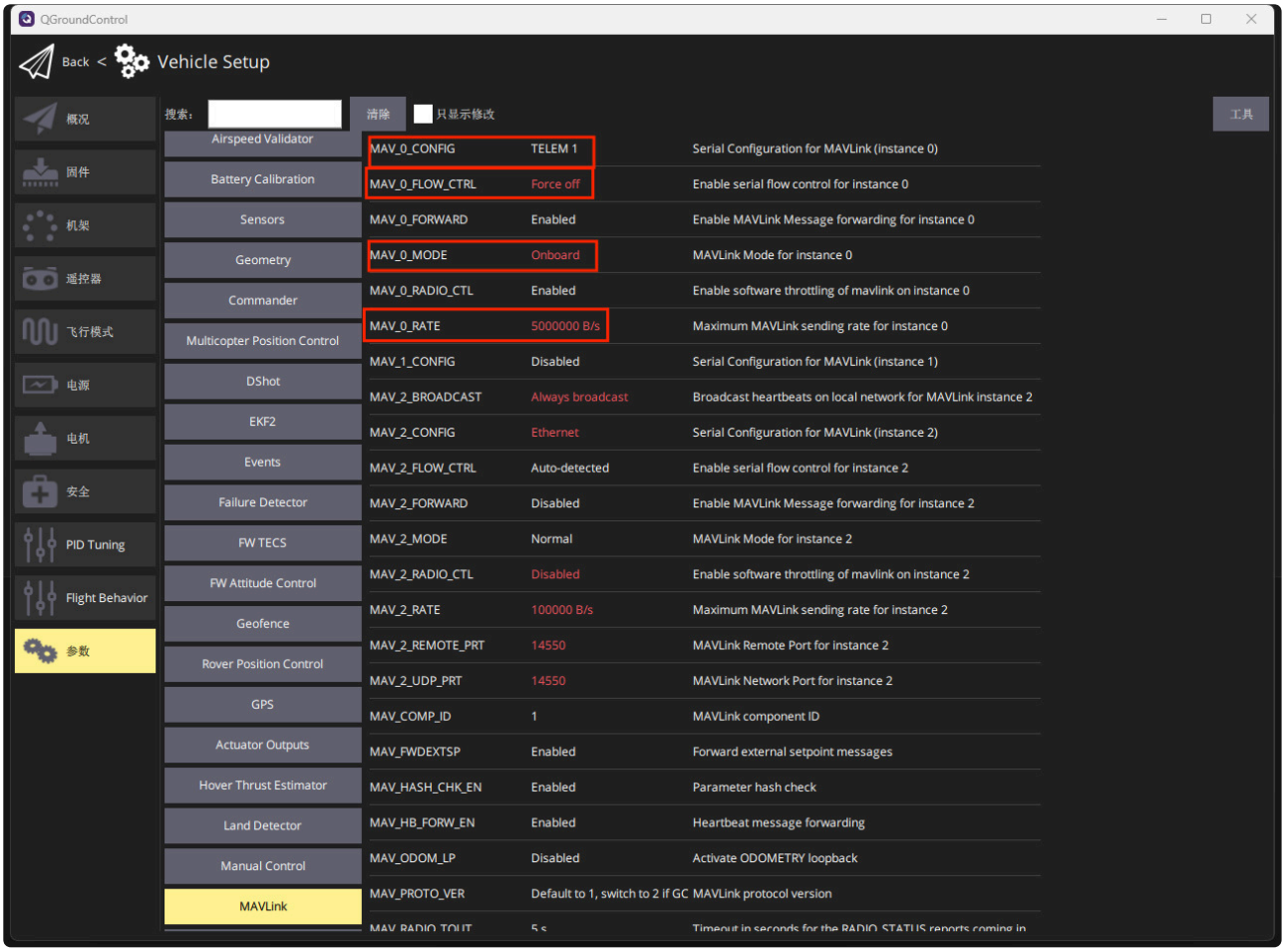
4. 选择机架，将机架设置为HIL Quadcopter X，设置完成后选择应用并重启，等待QGC重新加载。

5. 进入参数设置界面的MAVLink控制台
   设置MAV_0_CONFIG为TELEM1该参数为选择mavlink消息输出的接口。
   设置MAV_1_FLOW_CTRL: force off否则可能会导致无法连接到机载电脑
   并将MAV_0_MODE设置为Onboard用于和板载计算机通信。

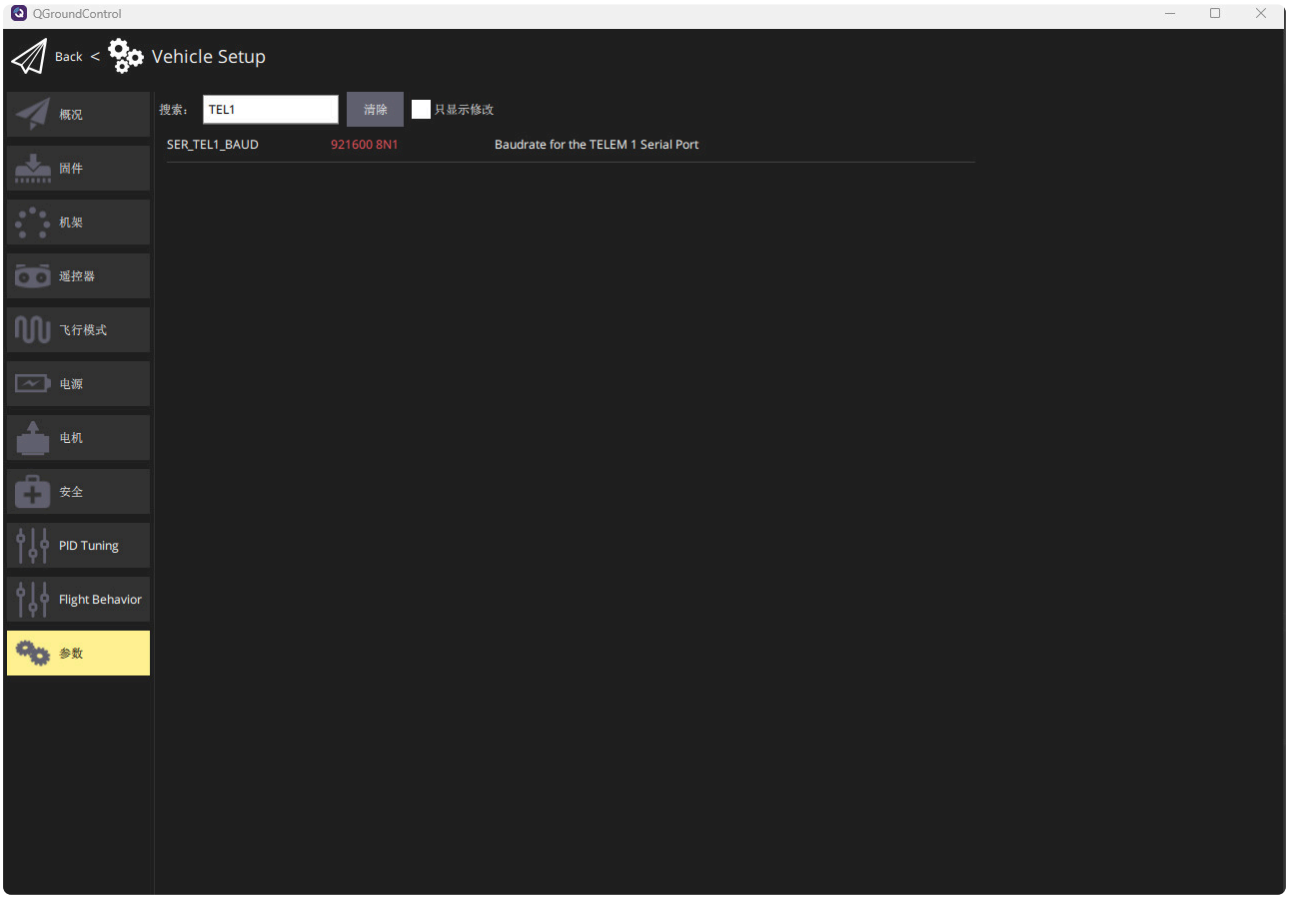
MAV_0_MODE有以下几种模式可选，各模式用途如下：

Normal	**用于和地面站通信**
Custom	自定义模式，一般是开发者使用
Onboard	用于和板载计算机通信
OSD	用于OSD显示
Config	用于快速连接（如USB）
Minimal	与地面站通信，通行量最小，适用于高延时的链路
ExtVision or ExtVisionMin	用于板外的视觉里程计数据通信
Iridium	用于铱卫星通信系统

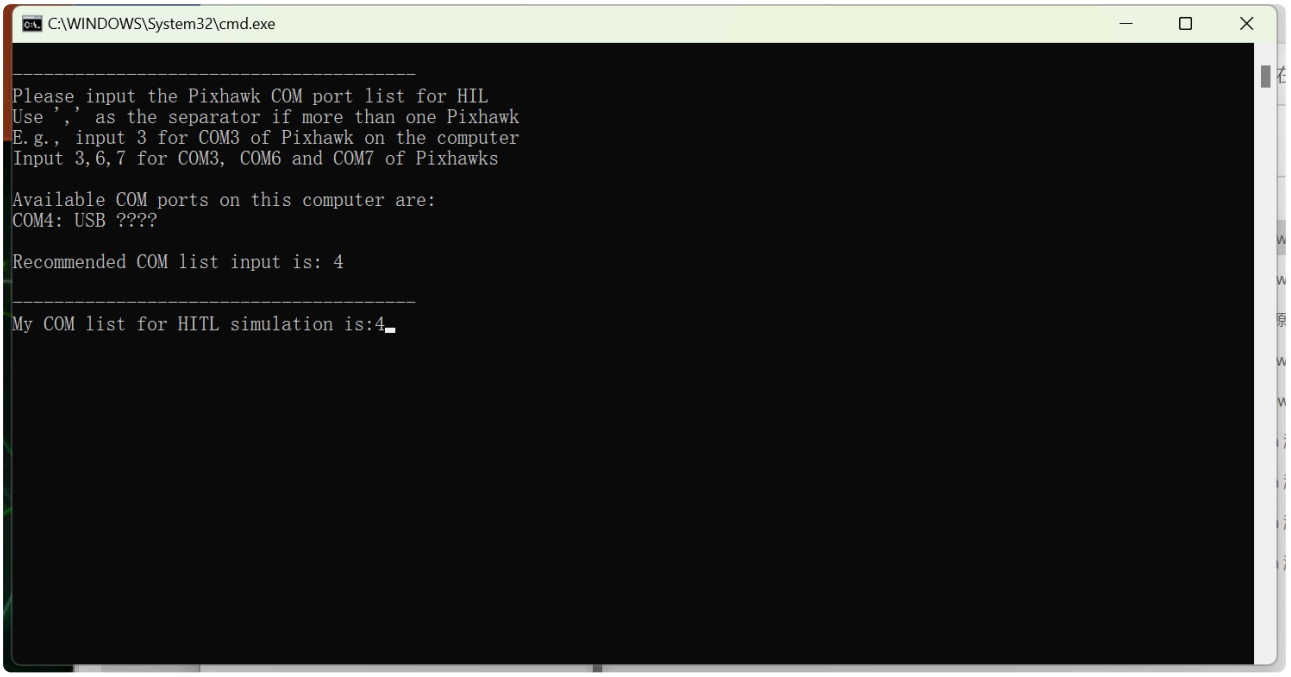
设置MAV_0_RATE为5000000
该参数为通信速率，如果总的通信速率超过此值，则会降低单个消息的发送频率，设置为0将数据速率设置为理论最大值的一半。

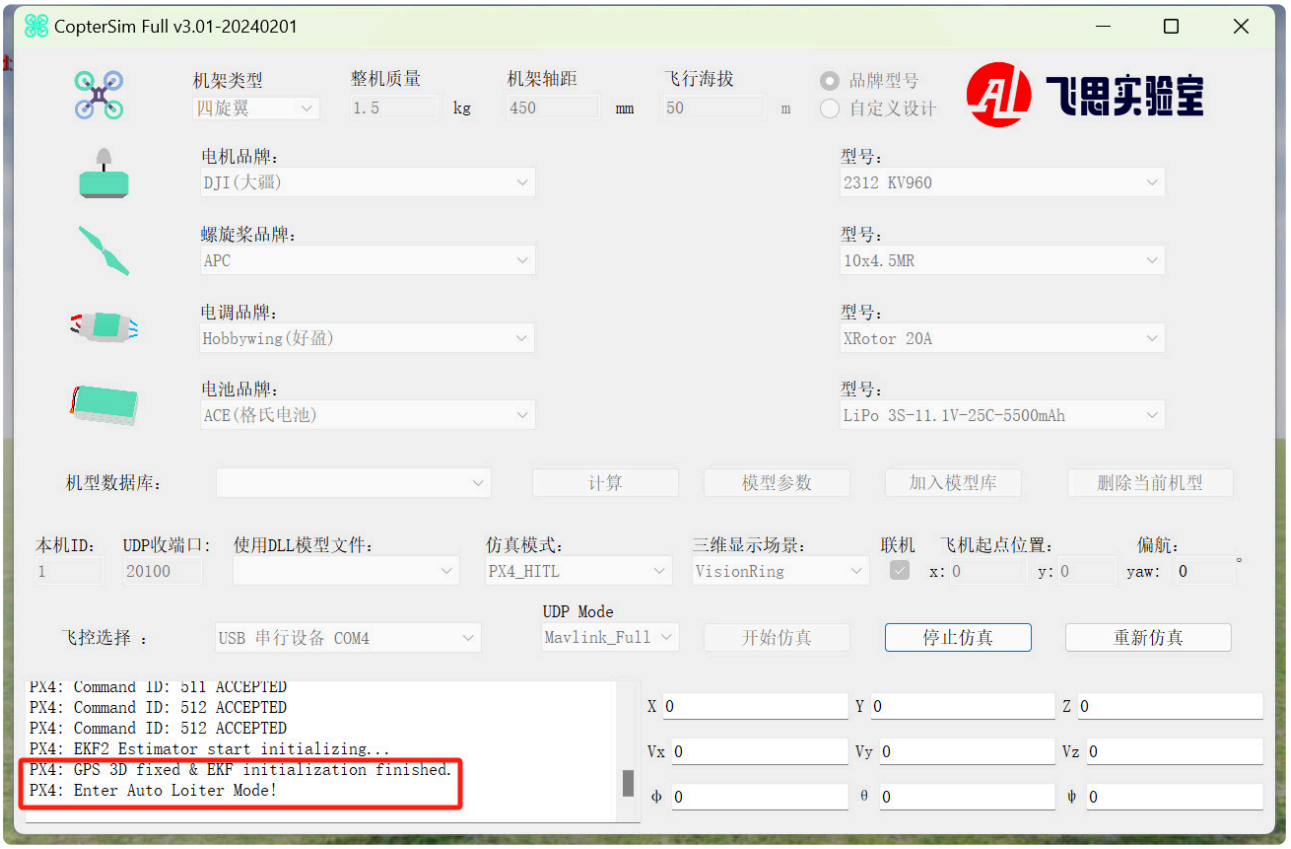


搜索TEL1将SER_TEL1_BAUD波特率设置为921600



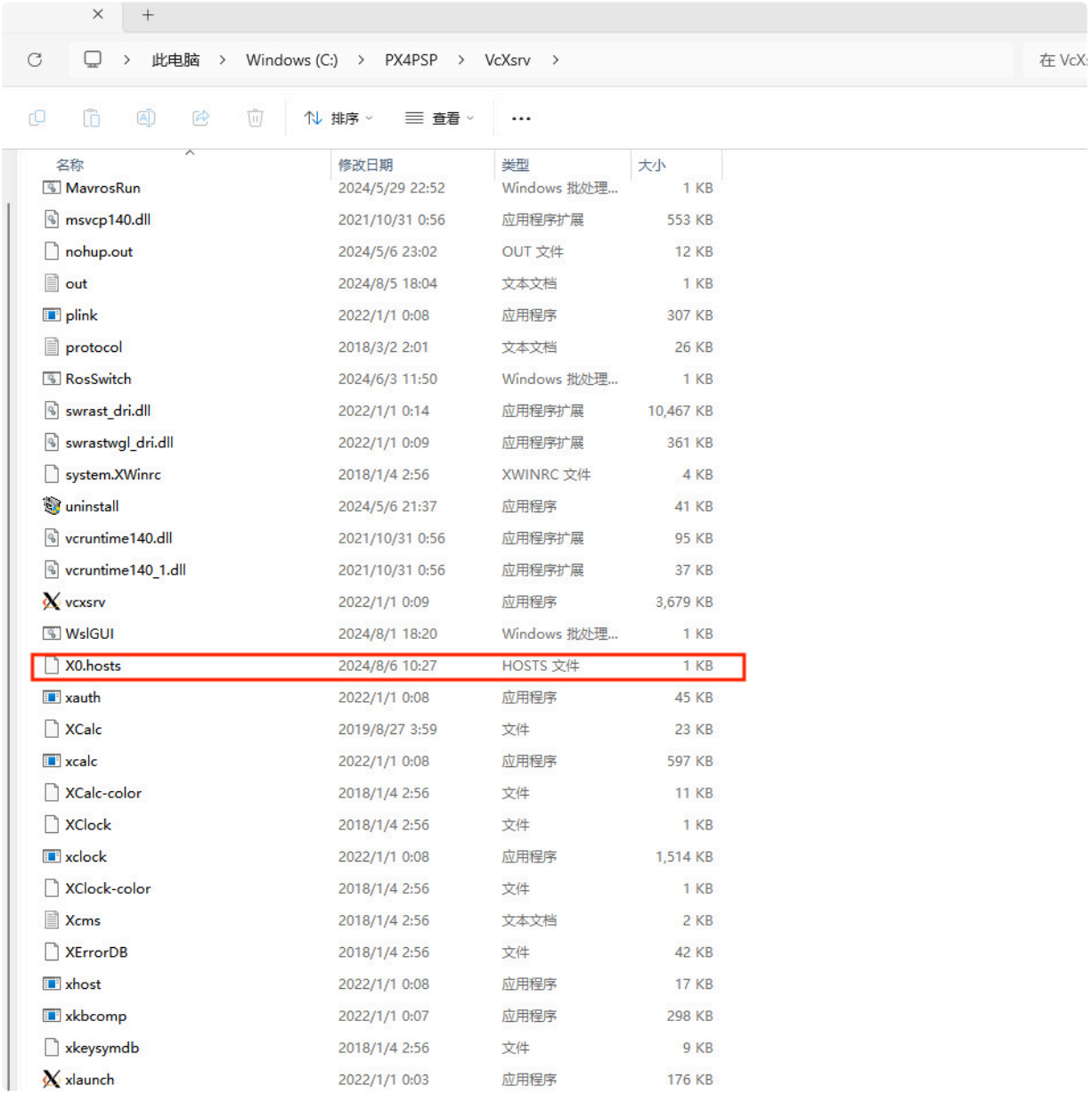
6. 打开8.RflySimVision\0.ApiExps\0.Preparation\3.NXwithPX4Config\0.PX4+NX以管理员身份运行PX4+NX_HITL.bat，输入推荐的COM回车打开硬件仿真，等待CopterSim初始化完成。





Step 2：配置NX与可视化

1. 将NX与主机连接至同一个局域网下推荐使用网线连接传输速率比较快。
2. 进入[安装目录]\VcXsrv下找到X0.hosts文件并在localhost下填入NX的IP地址



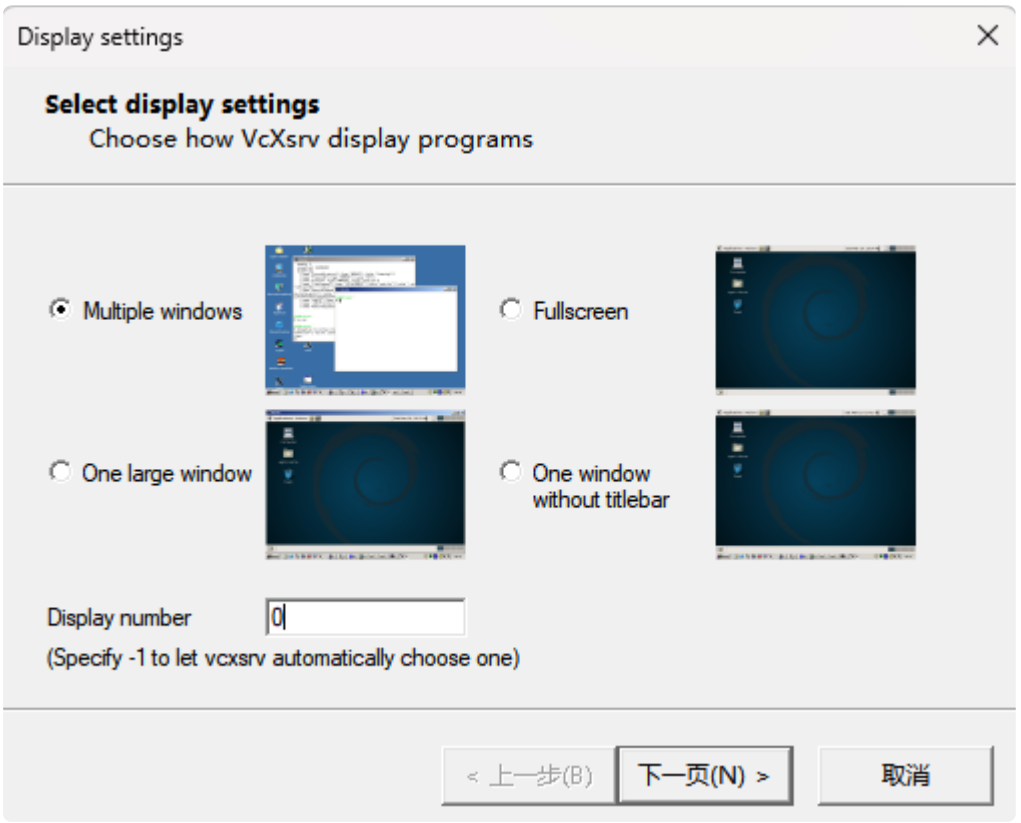

```
localhost
inet6:localhost
192.168.1.182
```

3. 在NX的.bashrc中最后添加一行export DISPLAY=主机IP:0,
   如我的windows电脑IP为192.168.1.3，则export DISPLAY=192.168.1.3:0

```
export DISPLAY=192.168.1.3:0
```

4. 找到`[安装目录]\\VcXsrv\\xlaunch.exe`在Display number处填0。一直点下一步即可。

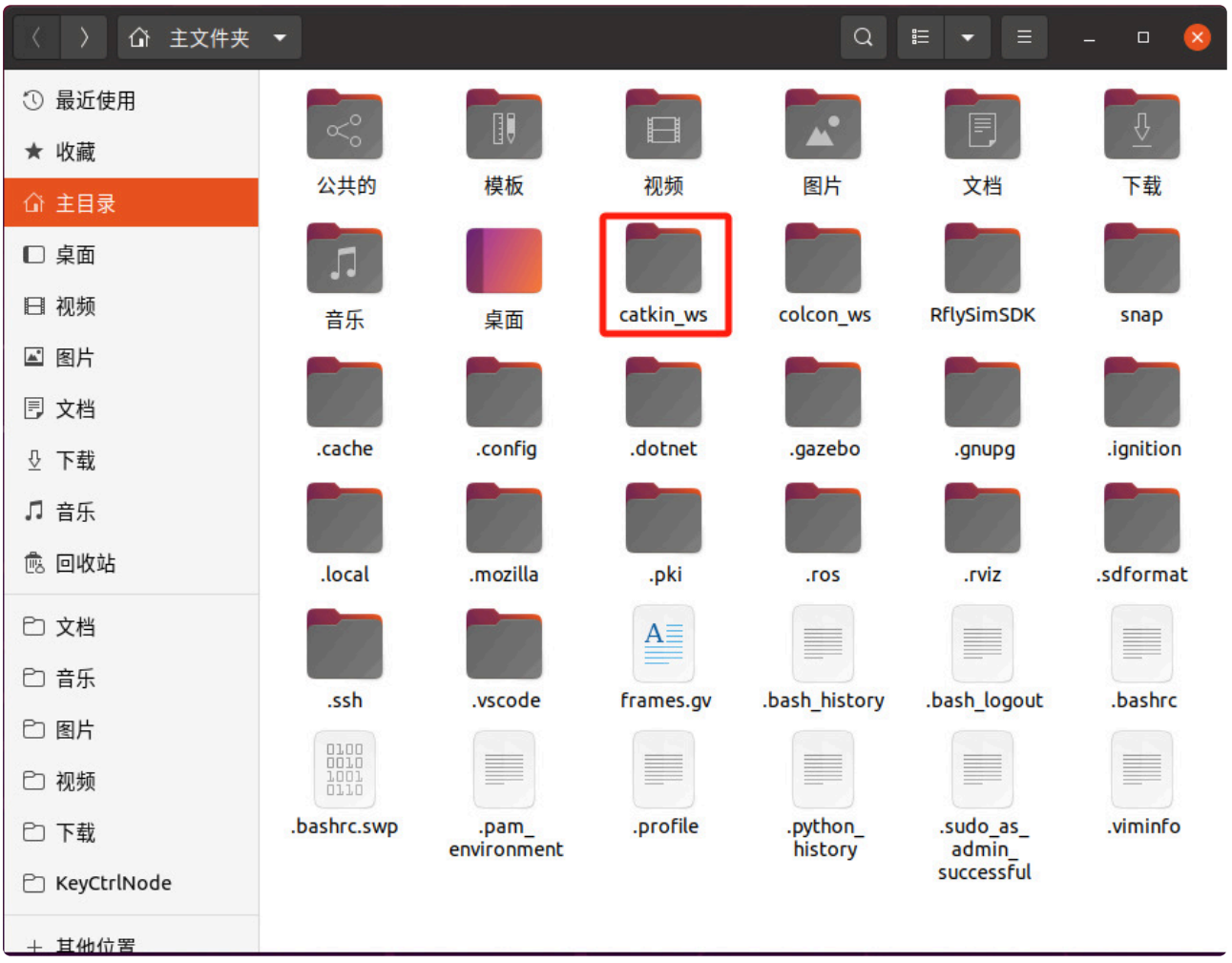
MavrosRun	2024/5/29 22:52	Windows 批处理...	1 KB
msvcp140.dll	2021/10/31 0:56	应用程序扩展	553 KB
nohup.out	2024/5/6 23:02	OUT 文件	12 KB
out	2024/8/5 18:04	文本文档	1 KB
plink	2022/1/1 0:08	应用程序	307 KB
protocol	2018/3/2 2:01	文本文档	26 KB
RosSwitch	2024/6/3 11:50	Windows 批处理...	1 KB
swrast_dri.dll	2022/1/1 0:14	应用程序扩展	10,467 KB
swrastwgl_dri.dll	2022/1/1 0:09	应用程序扩展	361 KB
system.XWinrc	2018/1/4 2:56	XWINRC 文件	4 KB
uninstall	2024/5/6 21:37	应用程序	41 KB
vcruntime140.dll	2021/10/31 0:56	应用程序扩展	95 KB
vcruntime140_1.dll	2021/10/31 0:56	应用程序扩展	37 KB
vcxsrv	2022/1/1 0:09	应用程序	3,679 KB
WslGUI	2024/8/1 18:20	Windows 批处理...	1 KB
X0.hosts	2024/8/6 10:27	HOSTS 文件	1 KB
xauth	2022/1/1 0:08	应用程序	45 KB
XCalc	2019/8/27 3:59	文件	23 KB
xcalc	2022/1/1 0:08	应用程序	597 KB
XCalc-color	2018/1/4 2:56	文件	11 KB
XClock	2018/1/4 2:56	文件	1 KB
xclock	2022/1/1 0:08	应用程序	1,514 KB
XClock-color	2018/1/4 2:56	文件	1 KB
Xcms	2018/1/4 2:56	文本文档	2 KB
XErrorDB	2018/1/4 2:56	文件	42 KB
xhost	2022/1/1 0:08	应用程序	17 KB
xkbcomp	2022/1/1 0:07	应用程序	298 KB
xkeysymdb	2018/1/4 2:56	文件	9 KB
xlaunch	2022/1/1 0:03	应用程序	176 KB



5. 最后在nx的终端输入xclock测试若windows电脑上出现下面图片则配置完成

Step 3：编译工作空间

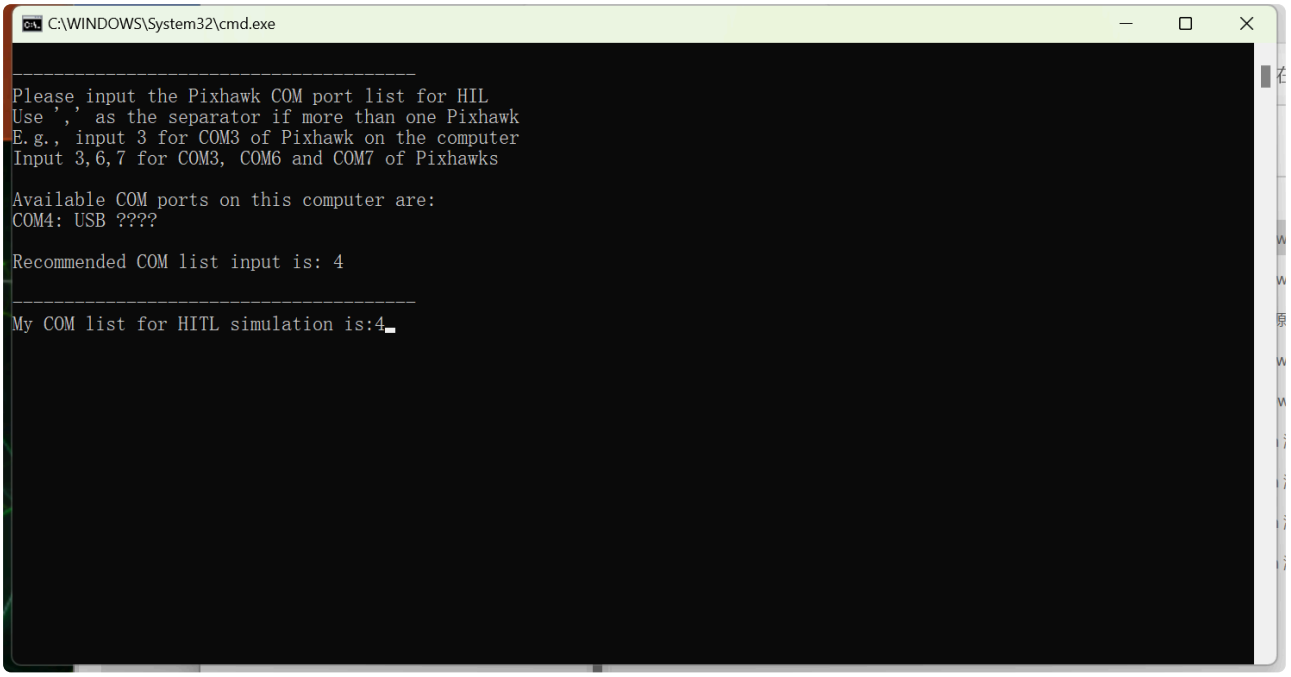
1. 在ubuntu里打开终端运行命令构建工作空间:mkdir -p catkin_ws/src。

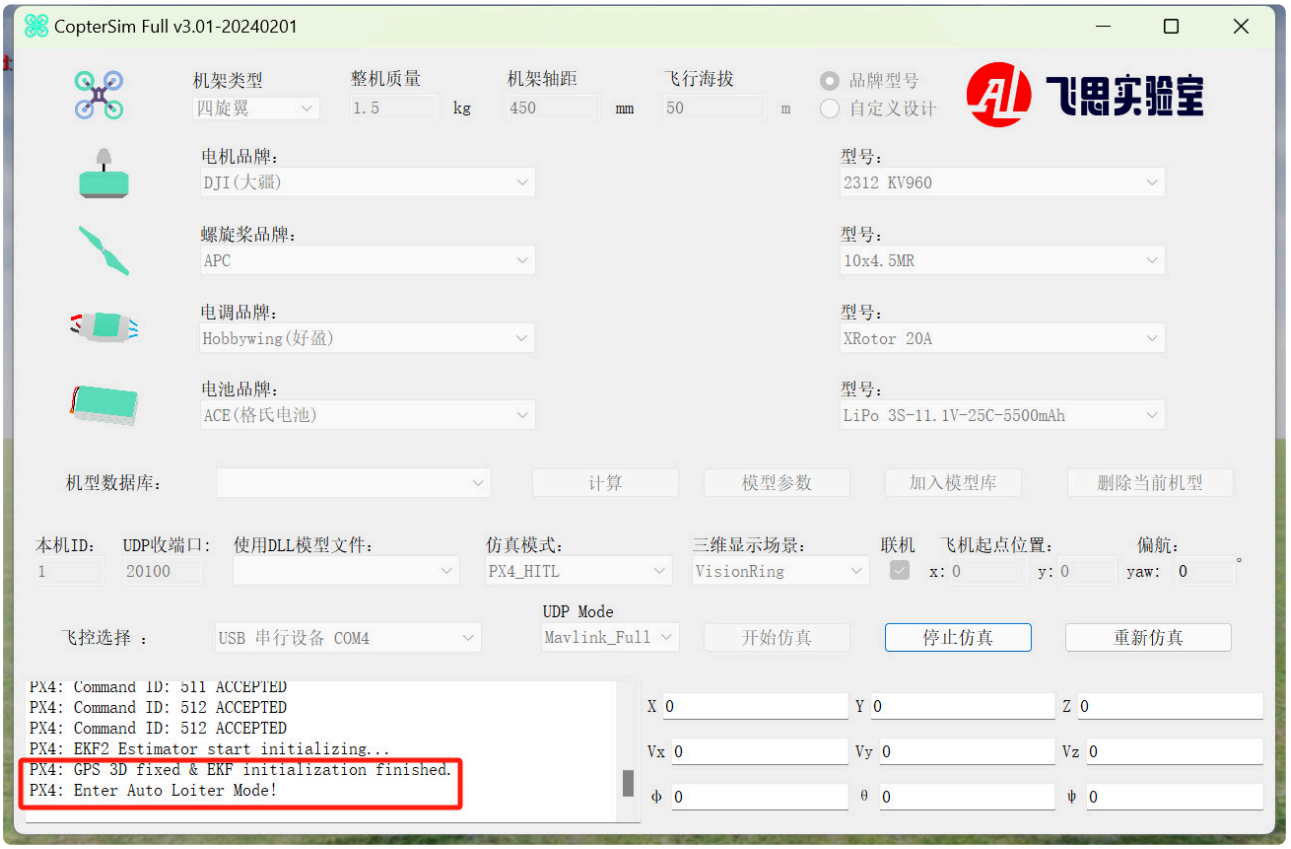


2. 把文件vision_ctrl拷贝到 catkin_ws/src里面。
3. 把文件 mavros.py、 rflysim_img_node.py、 server_ue4_Serial.py 以及Config文件四个文件拷贝至catkin_ws/src/vision_ctrl下
4. 编译：打开终端进入到catkin_ws 目录下，运行命令 catkin_make 进行编译。

Step 4：开启仿真

1.在windows双击运行PX4+NX_HITL.bat开启一个硬件在环仿真，输入推荐的端口等待CopterSim日志打印初始化完成。





1. 在NX系统上新开一个终端，输入如下命令给相应的通信端口赋权限

sudo chmod 777 /dev/ttyTHS0

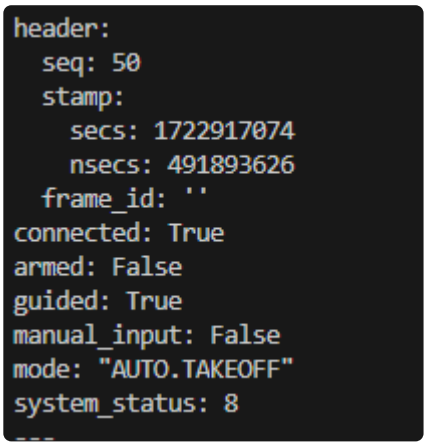
注：‘/dev/ttyTHS0’是用于飞控与NX通信的端口，其不一定就是THS0，但多数情况下为THS0，但都需要对其进行赋权限。

另开一个终端输入如下命令

roslaunch mavros px4.launch fcu_url:=/dev/ttyTHS0:921600

另起一个终端输入如下命令

rostopic echo /mavros/state



查看数据是否在更新，以及connected是否为True若数据在更新，则正常，若未更新且conneced为False则需要检查端口是否正确。

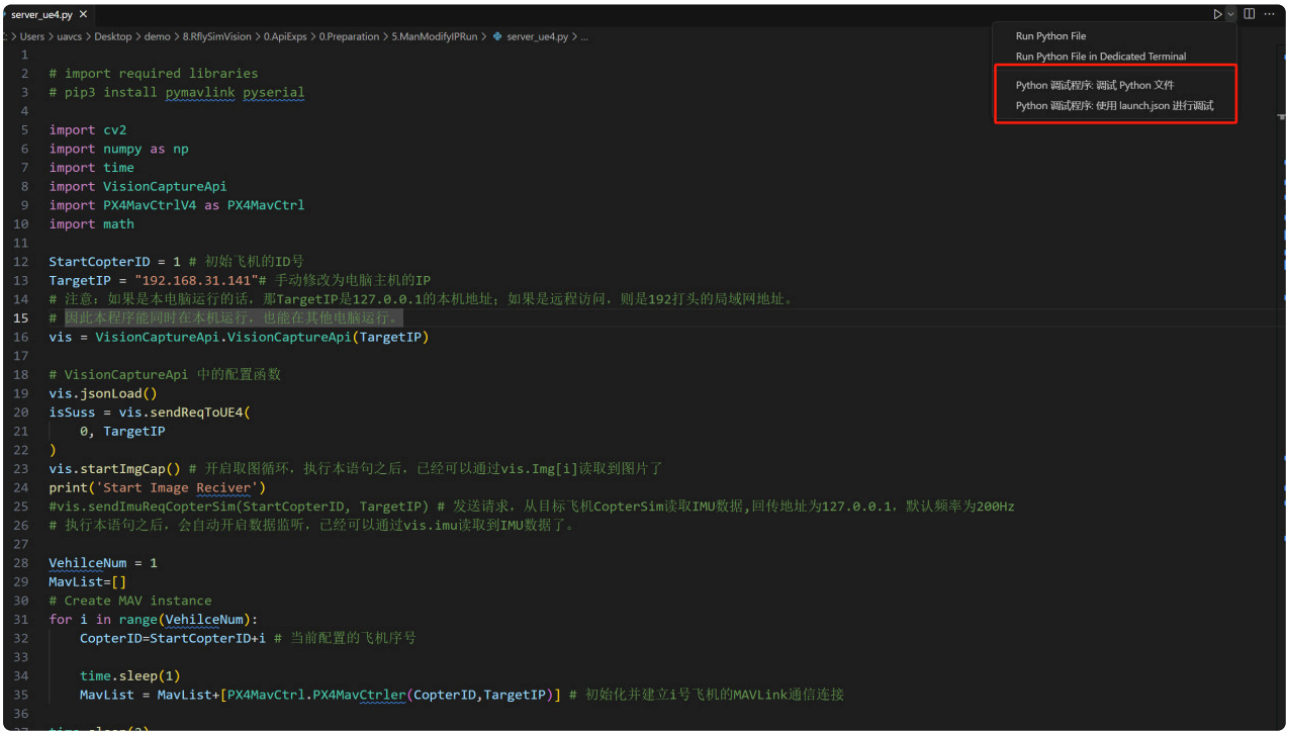
Step 5：运行控制程序

1. 打开终端进入到catkin_ws/src/vision_ctrl目录下，运行roslaunch mavros px4.launch fcu_url:=/dev/ttyTHS0:921600程序启动mavros。

```
nvidia@nvidia-desktop:~$ roslaunch mavros px4.launch fcu_url:=/dev/ttyTHS0:57600
... logging to /home/nvidia/.ros/log/ce294e4e-53d1-11ef-b2d5-68da73a8c446/roslaunch-nvidia-desktop-10392.log
Checking log directory for disk usage. This may take a while.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://nvidia-desktop:35237/

SUMMARY
========

CLEAR PARAMETERS
 * /mavros/

PARAMETERS
 * /mavros/camera/frame_id: base_link
 * /mavros/cmd/use_comp_id_system_control: False
 * /mavros/conn/heartbeat_rate: 1.0
 * /mavros/conn/system_time_rate: 1.0
 * /mavros/conn/timeout: 10.0
 * /mavros/conn/timesync_rate: 10.0
 * /mavros/distance_sensor/hrlv_ez4_pub/field_of_view: 0.0
 * /mavros/distance_sensor/hrlv_ez4_pub/frame_id: hrlv_ez4_sonar
 * /mavros/distance_sensor/hrlv_ez4_pub/id: 0
 * /mavros/distance_sensor/hrlv_ez4_pub/orientation: PITCH_270
 * /mavros/distance_sensor/hrlv_ez4_pub/send_tf: True
 * /mavros/distance_sensor/hrlv_ez4_pub/sensor_position/x: 0.0
 * /mavros/distance_sensor/hrlv_ez4_pub/sensor_position/y: 0.0
 * /mavros/distance_sensor/hrlv_ez4_pub/sensor_position/z: -0.1
 * /mavros/distance_sensor/laser_1_sub/id: 3
 * /mavros/distance_sensor/laser_1_sub/orientation: PITCH_270
 * /mavros/distance_sensor/laser_1_sub/subscriber: True
 * /mavros/distance_sensor/lidarlite_pub/field_of_view: 0.0
```

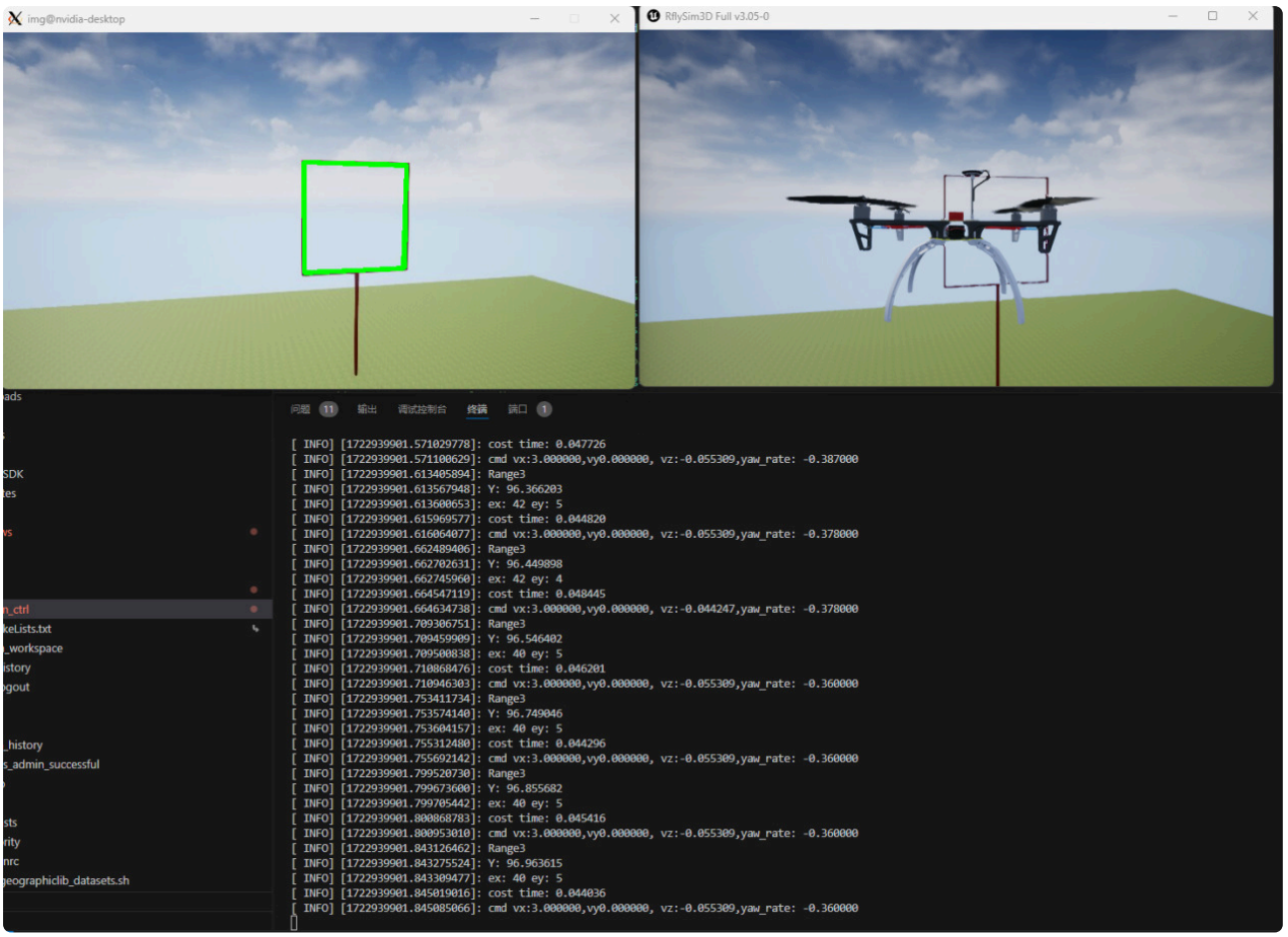
2. 在此目录下开启新终端运行python3 rflysim_img_node.py

备注：可以使用VS Code开发并调试Ubuntu下python文件。



3 .打开新终端，切换到catkin_ws目录下，运行source devel/setup.bash,
再运行运行节点rosrun vision_ctrl

visino_ctrl，可以看到如下图所示的效果。或关闭之前运行的程序直接运行python3 server_ue4_Serial.py 也可以得到下图实验效果



5.2 选作实验

准备工作：

虚拟机或NX的配置方法是相同的。

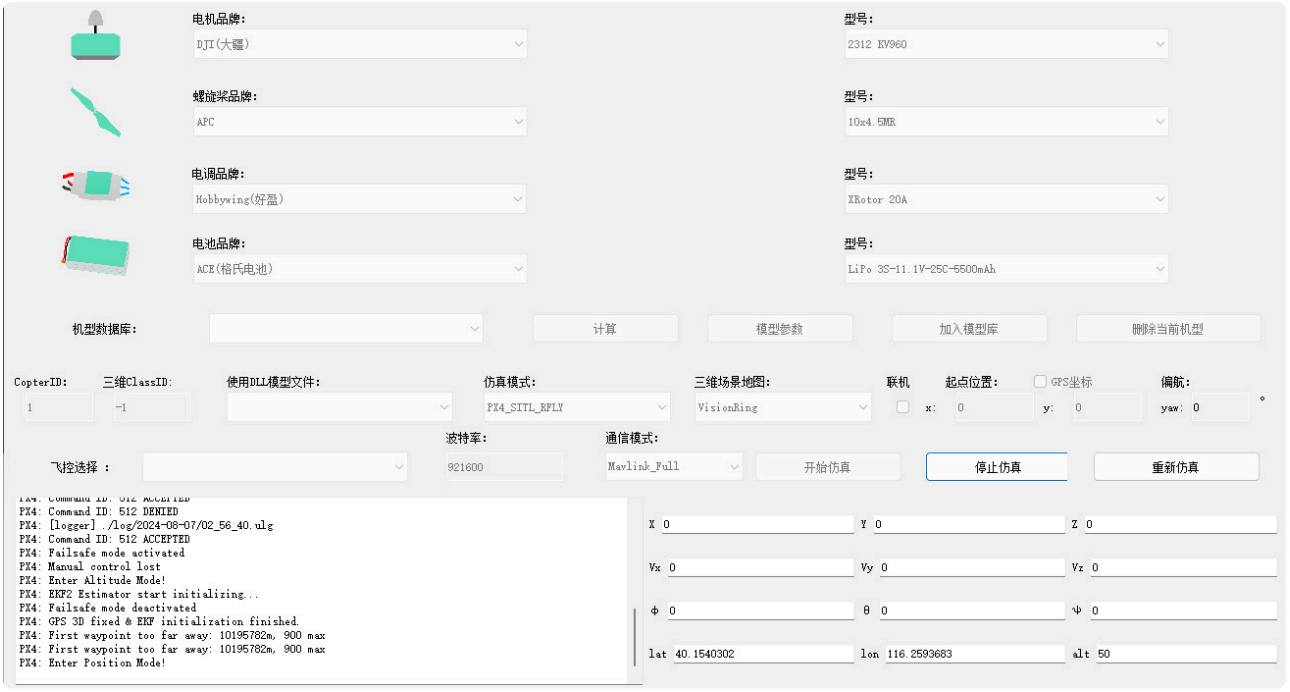
NX板卡的配置方法，先看 [安装目录]\RflySimAPIs\8.RflySimVision\0.ApiExps\0.Preparation\3.NXwithPX4Config\Readme.pdf

。

5.2.1在NX上进行软件在环仿真

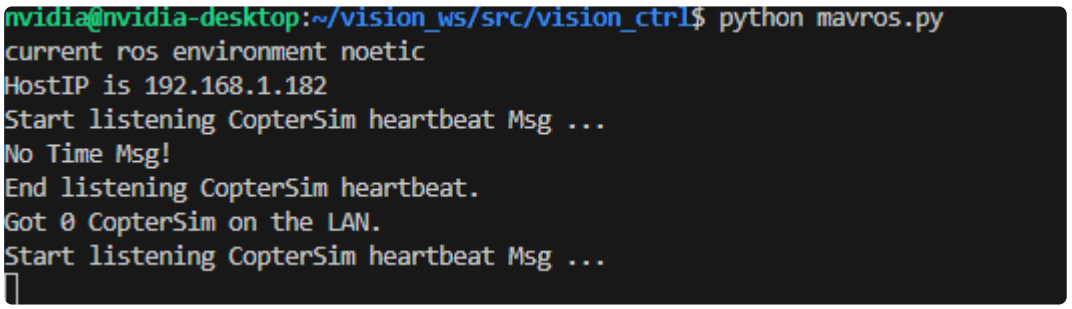
Step 1：开启仿真

1.在windows双击运行 `PX4+NX_SITL.bat` 开启一个软件在环仿真口等待CopterSim日志打印初始化完成。

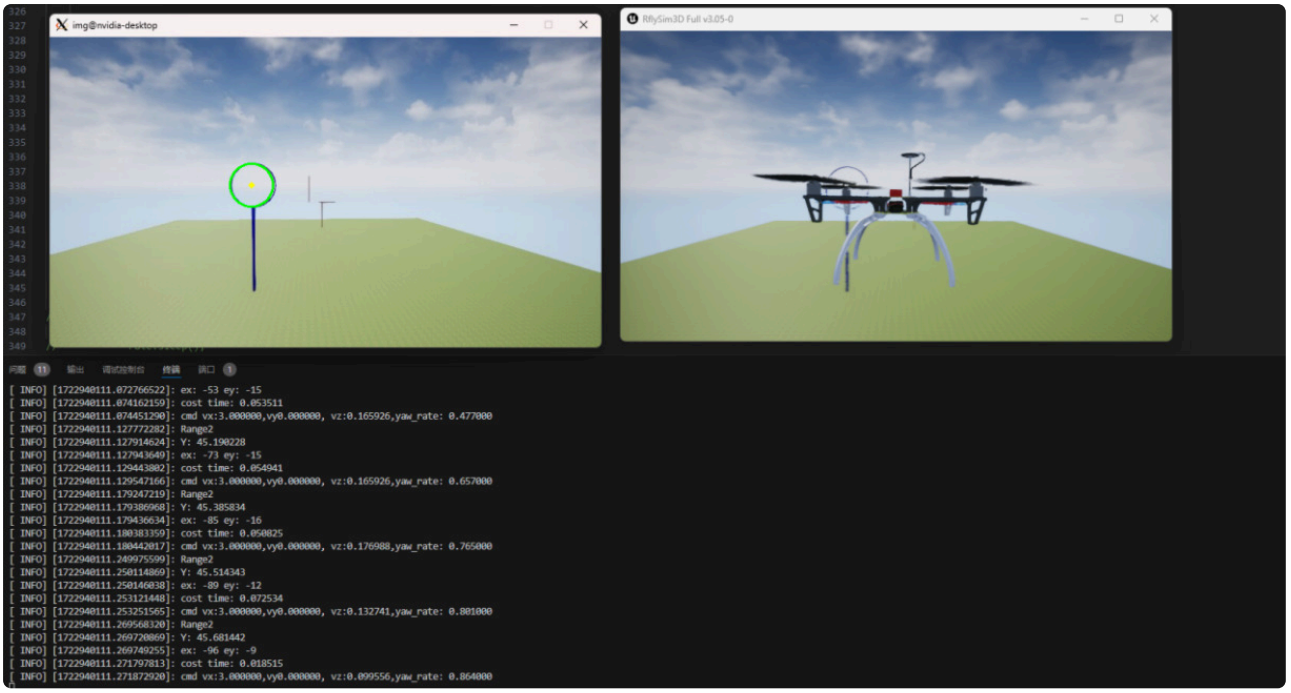


Step 2：运行控制程序

1. 打开终端进入到vision_ctrl目录下，运行`python3 mavros.py`程序启动mavros。再运行`rflysim_img_node.py`



2.打开终端，切换到catkin_ws目录下，运行`source devel/setup.bash`，再运行运行节点`rosrun vision_ctrl visino_ctrl`，可以看到如下图所示的效果。



6.参考资料

无

7.常见问题

Q1：无

A1：无