1.实验名称及目的

1.1实验名称

dll模型ReqCopterSim.py接口库使用说明

1.2实验目的

以平台进行仿真时，用户不仅可以通过SITL/HITLRun.bat来设置模型初始化相关信息，还可以通过Python的方式进行设置。平台ReqCopterSim.py支持通过Python的方式对CopterSim相关初始化参数进行设置，参数包括：

1）是否联机

2）指定CopterSim 回传数据IP

3）北东地坐标系下的x、y、z

4）姿态，滚转、俯仰、偏航

5）仿真模式

6）DLL模型

7）DLL序号

8）三维场景

9）三维场景序号

通过该例程熟悉ReqCopterSim.py接口库的使用

1.3关键知识点

ReqCopterSim接口库的核心思想为，通过该接口获取CopterSim电脑IP，进而发送指令对CopterSim相关参数进行设置，下面对ReqCopterSim.py接口库中关键函数索引进行介绍：

1）

def sendReSimDllName(self,CopterID=1,name=''):

功能：该函数可以通过DLL模型名称修改CopterSim所调用的DLL。

说明：

Self：表明ReqCopterSim共享对象会影响该函数的行为；

CopterID：CopterSim ID号，默认为1.

name：DLL模型名称。

注：要注意的是，这里默认搜索的DLL模型名称路径为C:\PX4PSP\CopterSim\external\model（以平台安装在C盘为例进行说明），因此在使用该接口修改DLL模型时要确保C:\PX4PSP\CopterSim\external\model路径下是否存在新的DLL模型。

2)

def sendReSimDllIdx(self,CopterID=1,index=-1):

功能：该函数可以通过DLL序号来修改CopterSim所调用的DLL。

说明：

Self：表明ReqCopterSim共享对象会影响该函数的行为；

CopterID：CopterSim ID号，默认为1.

index：选项ID序号，这里对应了DLL的序号。

3)

def sendReSimMapName(self,CopterID=1,name=''):

功能：该函数可以通过地图名称来修改三维引擎仿真场景。

说明：

Self：表明ReqCopterSim共享对象会影响该函数的行为；

CopterID：CopterSim ID号，默认为1.

name：地图名称。

4)

def sendReSimMapIdx(self,CopterID=1,index=-1):

功能：该函数可以通过地图序号来修改三维引擎仿真场景。

说明：

Self：表明ReqCopterSim共享对象会影响该函数的行为；

CopterID：CopterSim ID号，默认为1.

index：选项ID序号，这里对应了地图的序号。

5)

def sendReSimIP(self,CopterID=1):

功能：请求指定CopterSim将数据回传到本电脑，适用于联机设置Coptersim消息。

说明：

Self：表明ReqCopterSim共享对象会影响该函数的行为；

CopterID：指定CopterSim ID号。

6)

def sendReSimUdpMode(self,CopterID=1,UDP_mode=-1):

功能：请求指定CopterSim更换UDP_Mode为指定值。

说明：

Self：表明ReqCopterSim共享对象会影响该函数的行为；

CopterID：指定CopterSim ID号。

UDP_mode：通讯模式，0-UDP_FULL，1-UDP_Simple，2-Mavlink_Full,3-Mavlink_Simple，4-Mavlink_Nosend，5-Mavlink_NoGPS，6-Redis_Full，7-Redis_Simple

7)

def sendReSimXYyaw(self,CopterID=1,xyYaw=[0,0,0]):

功能：请求指定CopterSim更换x、y、yaw为指定值，x、y单位为米，北东地，yaw，单位为度

说明：
Self：表明ReqCopterSim共享对象会影响该函数的行为；

CopterID：指定CopterSim ID号。

xyYaw=[0,0,0]：分别为期望x、y、yaw。

8）

def sendReSimXyzRPYaw(self,CopterID=1,Xyz=[0,0,0],RPYaw=[0,0,0]):

功能：请求指定CopterSim更换xyz为指定值，单位m，北东地，roll、pitch、yaw为指定值，单位度。

Self：表明ReqCopterSim共享对象会影响该函数的行为；

CopterID：指定CopterSim ID号。

Xyz=[0,0,0]：期望xyz。

RPYaw=[0,0,0]：期望滚转、俯仰、偏航。

更多实验原理见：..\4.RflySimModel\0.ApiExps\4.InitAPI\Readme.pdf

2.实验效果

运行MulticopterNoCtrl_SITLRun.bat后testReqCopterSim.py，会看到以下效果：

1、当vscode终端提示“发送联机模式重设命令”时，表明已完成CopterSim消息获取设置，在本例程中设置获取数据的CopterSim
ID为1。

2、当vscode终端提示“发送UDP仿真模式重设命令”时，在CopterSim界面的“UDP mode”栏可以看到仿真通讯模式从UDP_Full重设成Mavlink_full。

3、当vscode终端提示“发送XYyaw初始值重设命令”时，可以分别在CopterSim和RflySim3D中看到变化，在CopterSim右下角会看到x为1，y为2，偏航为30°，在RflySim3D中能观察到飞机位置和姿态发生了变化。

4、当vscode终端提示“发送XyzRPYaw初始值重设命令”时，可以分别在CopterSim和RflySim3D中看到变化，在RflySim3D中可以看到飞机从地面变化到了空中，并且自由落体直到落在地面上，偏航角为45°。

5、当vscode终端提示“发送DLL模型重设命令”时，可以分别在CopterSim和RflySim3D中看到变化，在CopterSim界面的“使用DLL模型文件”一栏中切换为了“HexarotorModelCTRL”，在RflySim3D中能看到显示模型从四旋翼切换为了六旋翼。

6、当vscode终端提示“发送map地图重设命令”时，可以分别在CopterSim和RflySim3D中看到变化，在CopterSim界面“三维显示场景”一栏中切换为了VisionRing，同时可以在RflySim3D中场景改为了VisionRing。

7、以上修改依次完成后，可以进行正常的初始化与操作，让飞控重新fixed后依然可以进行offboard控制。

3.文件目录

例程目录：
[安装目录]\RflySimAPIs\4.RflySimModel\0.ApiExps\4.InitAPI\1.ReqCopterSim

文件夹/文件名称	说明
MulticopterNoCtrl.dll	四旋翼模型dll动态链接库
testReqCopterSim.py	接口测试程序
MulticopterNoCtrl_SITLRun.bat	四旋翼软件在环启动脚本
HexarotorModelCTRL	六旋翼模型dll动态链接库
Python38Run.bat	Python程序运行脚本

4.运行环境

4.1 软件要求

Windows 10及以上版本；RflySim工具链；MATLAB 2017b及以上③。

> 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；Pixhawk 6X或其它飞控② 1台；数据线 1台。

> ①：推荐配置请见： https://rflysim.com/

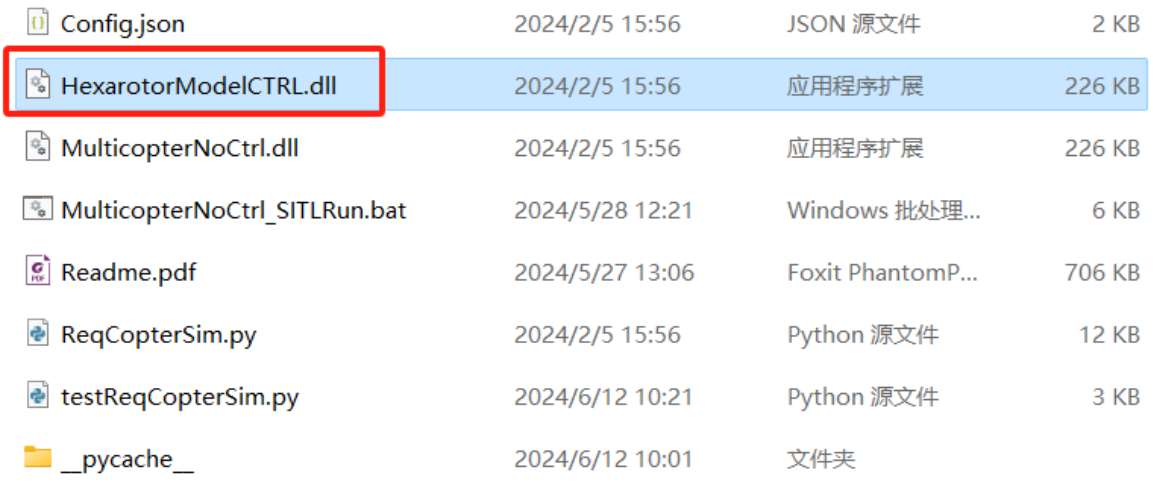
5.实验步骤

必做实验：ReqCopterSim.py接口库使用

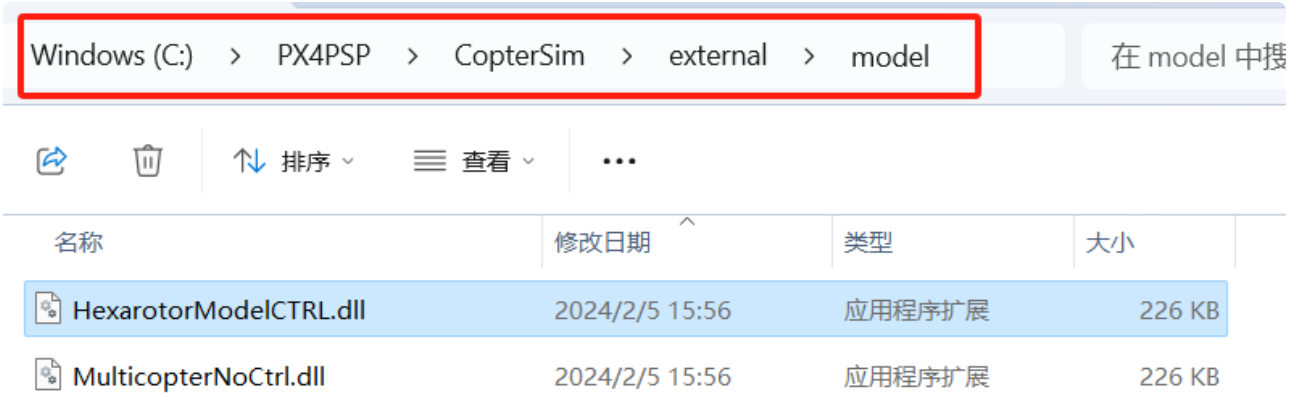
Step 1：添加DLL模型文件

为了保证sendReSimDllName()函数能成功修改CopterSim当前所调用的DLL模型，在运行testReqCopterSim.py前，需要将目标DLL模型文件复制到“*:\PX4PSP\CopterSim\external\model”文件目录（*由平台安装路径决定，默认为C盘）。

在该例程中，目标DLL模型文件（即需要CopterSim重新调用的DLL模型）为HexarotorModelCTRL.dll。

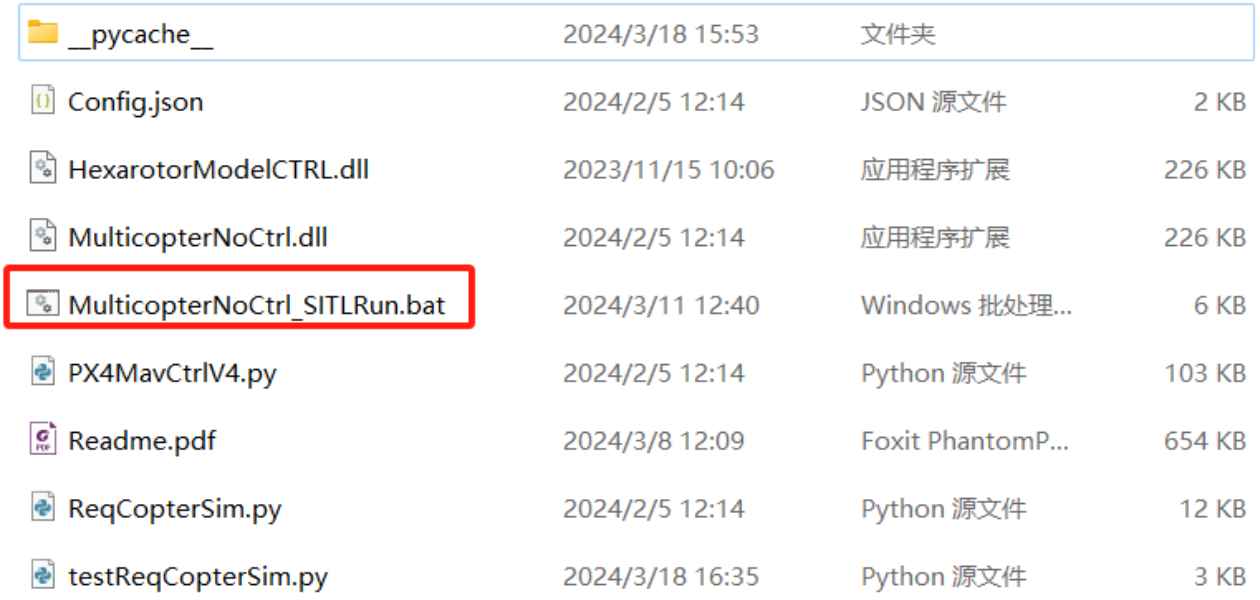


在实验开始前，将其复制到“*:\PX4PSP\CopterSim\external\model”文件目录下，这样即可通过sendReSimDllName()将CopterSim当前调用的DLL模型修改为目标DLL模型。



Step 2：启动仿真

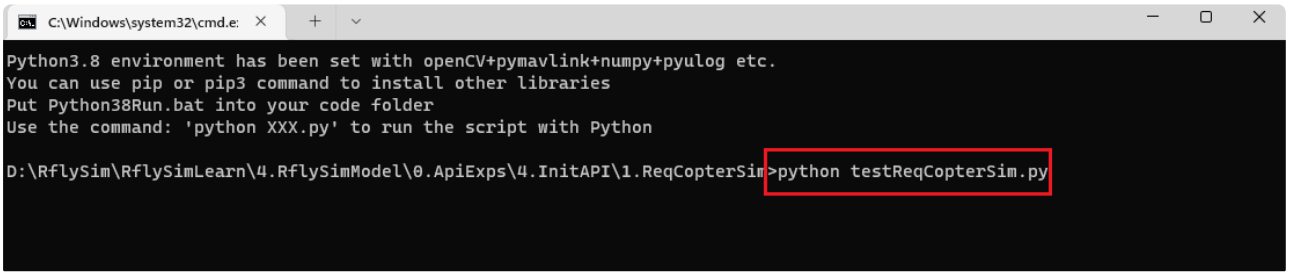
启动四旋翼的软件在环仿真并创建一个飞机。



Step 3：运行控制程序

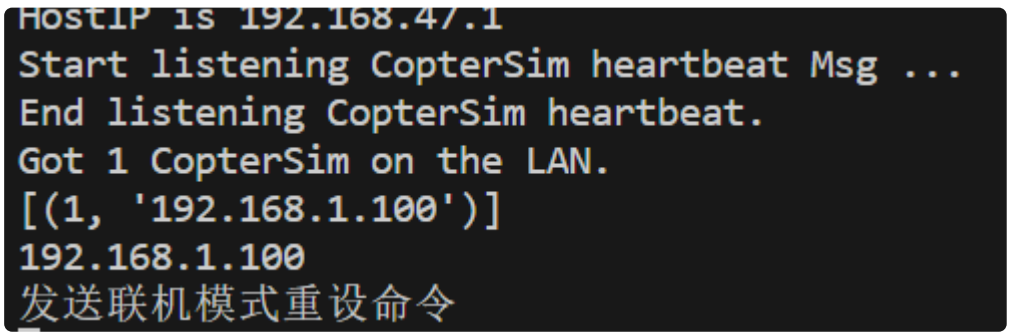
等待初始化完成。

在文件夹下，双击Python38Run.bat，打开集成好的python环境，在该环境下运行testReqCopterSim.py文件，输入
python testReqCopterSim.py



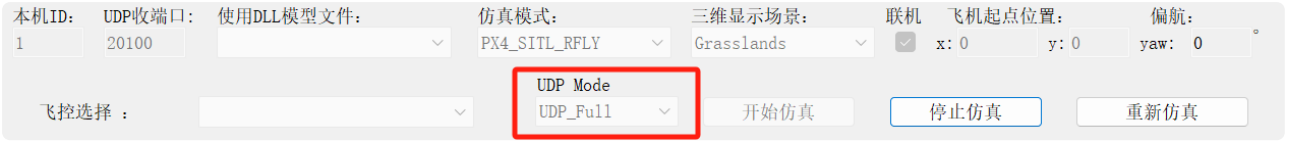
Step 4：观察结果

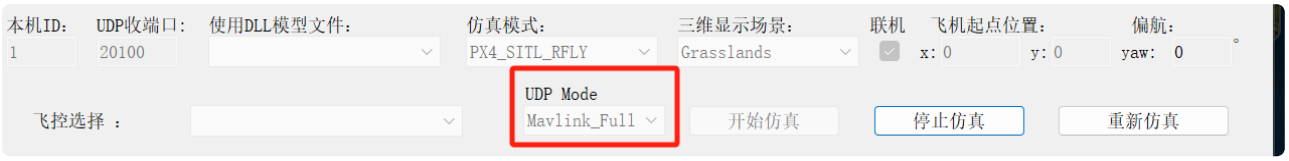
1．可以看到该程序获取了CopterSim所在电脑的IP，并进行配置。



2． 运行至“sendReSimUdpMode”时可以在CopterSim上观察到UDP_Mode由UDP Full转换为了MAVLink Ful1。

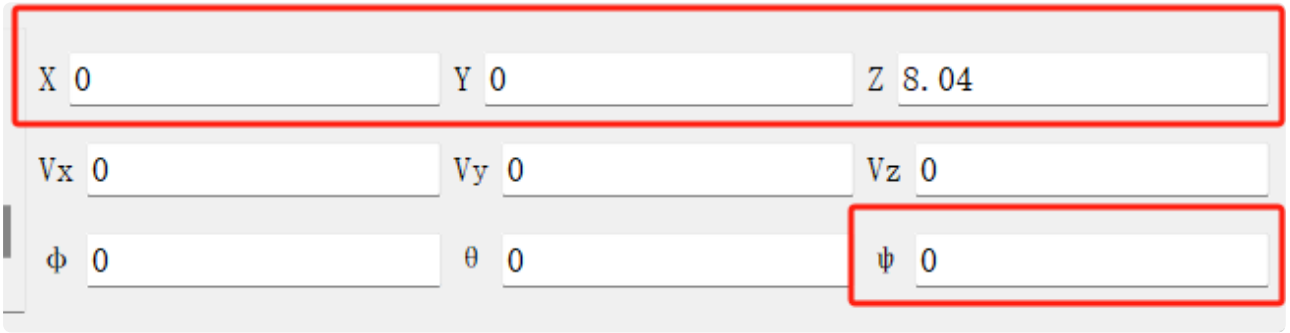
```
# 通过本接口，可以强制修改CopterSim的UDP_Mode，这里只有要时，才发送
new_udp_mode=2 # 注，SITLRun模式是0模式，也就是UDP_Full，这里我们强制修改为2模式
req.sendReSimUdpMode(CopterID,new_udp_mode) # 强制申请CopterSIM转换为MAVLink_Full通信模式
print('发送UDP仿真模式重设命令')
time.sleep(5)
```

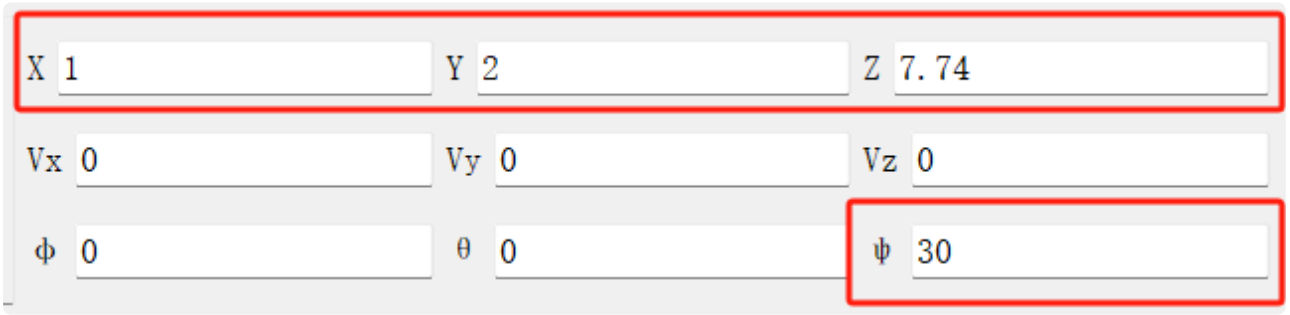




3．运行至“sendReSimXYyaw”时可以在CopterSim看到载具的x y 位置和yaw的初始值的改变，同时也可以在UE界面观察到载具的突变。

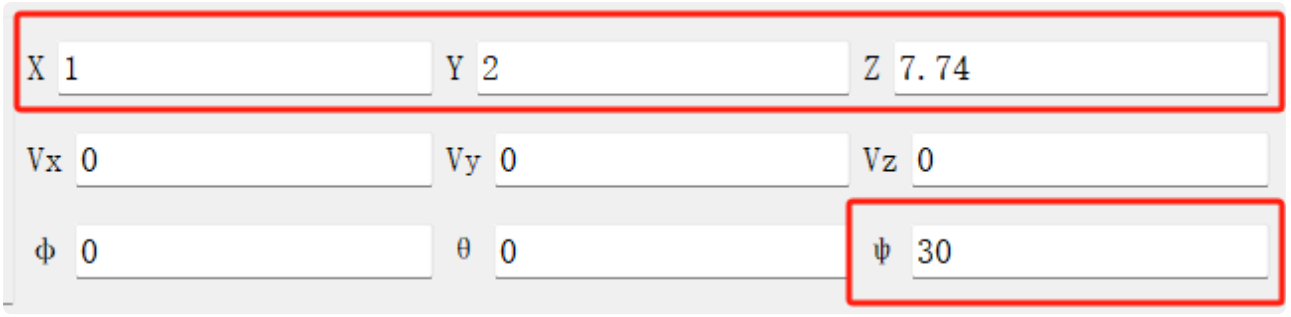
```
req.sendReSimXYyaw(CopterID,[1,2,30]) # 强制申请CopterSIM切换初始位置为x=1m,y=2m,yaw=30degree
print('发送XYyaw初始值重设命令')
```

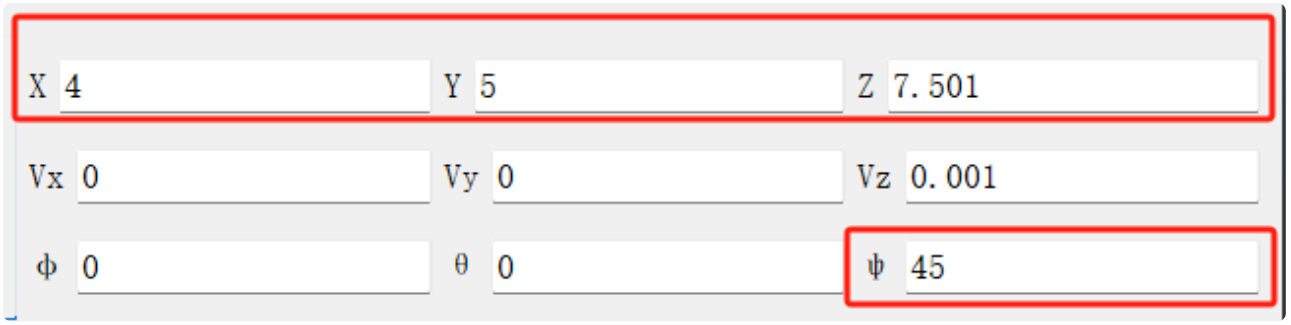




4．运行至“sendReSimXyzRPYaw”时可以在CopterSim看到载具位置与姿态角的变化，同时也可以在UE界面观察到载具的突变，此接口相较于“sendReSimXYyaw”能修改z方向位置与滚转和俯仰角度

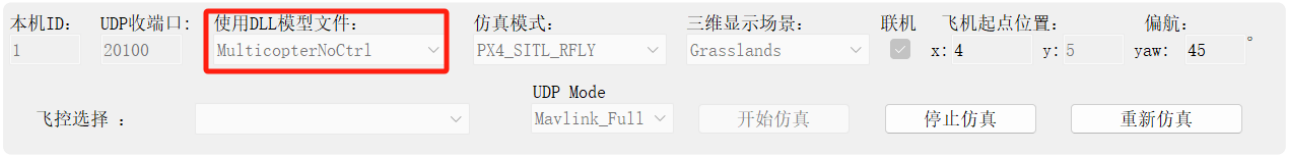
```
req.sendReSimXyzRPYaw(CopterID,[4,5,-30],[0,0,45]) # 强制申请CopterSIM切换初始位置为30米高，偏航角为45度
print('发送XyzRPYaw初始值重设命令')
```

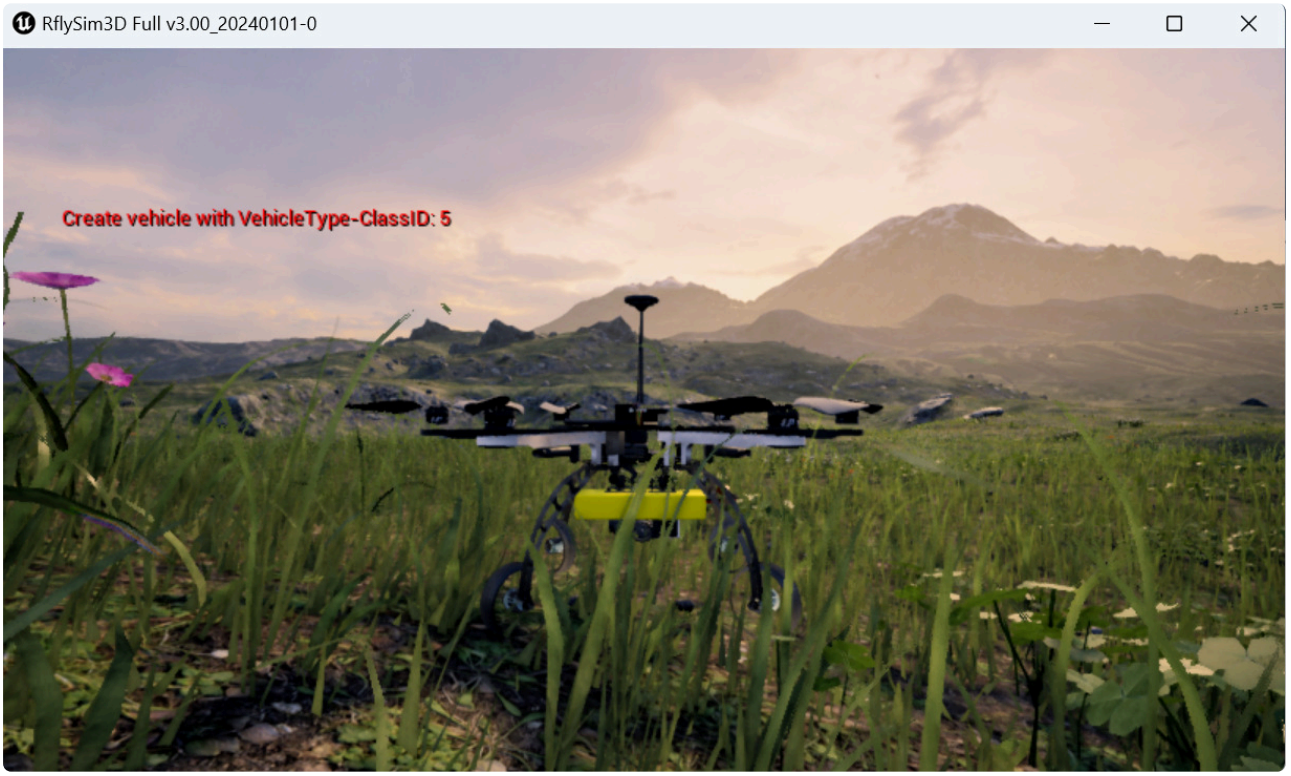




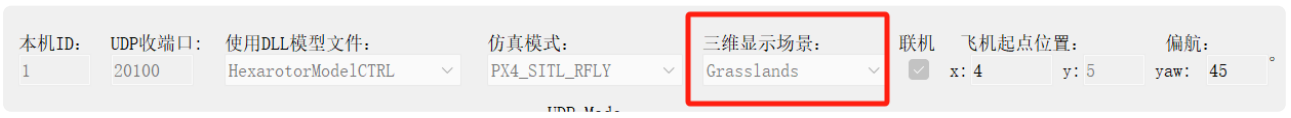
5．运行至“sendReSimDllName”时可以在CopterSim看到dll模型由四旋翼变为六旋翼，同时也可以在UE界面观察到载具类型由四旋翼修改为了六旋翼（注：在使用该函数前，一定要保证C:\PX4PSP\CopterSim\external\model文件目录下是存在目标DLL模型文件的，否则会导致修改不成功！）。

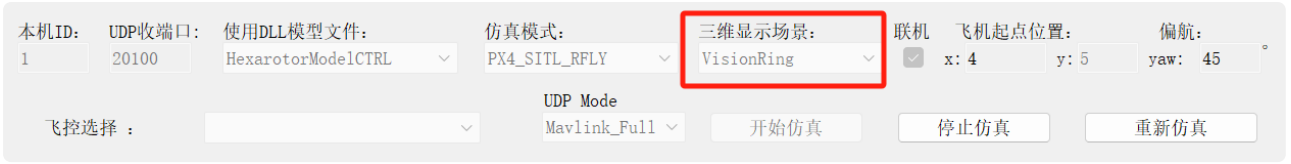
```
55    time.sleep(5)
56    req.sendReSimDllName(CopterID,'HexarotorModelCTRL')
57    print('发送DLL模型重设命令')
```

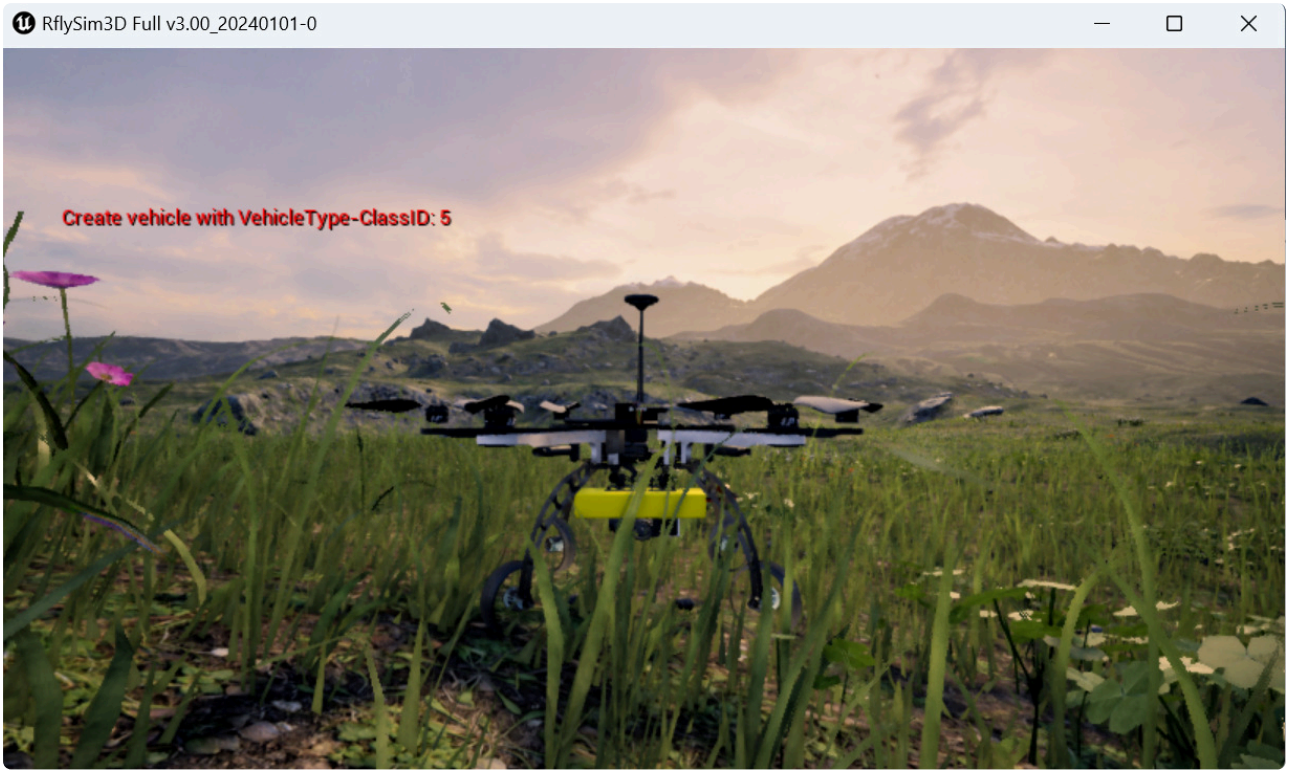


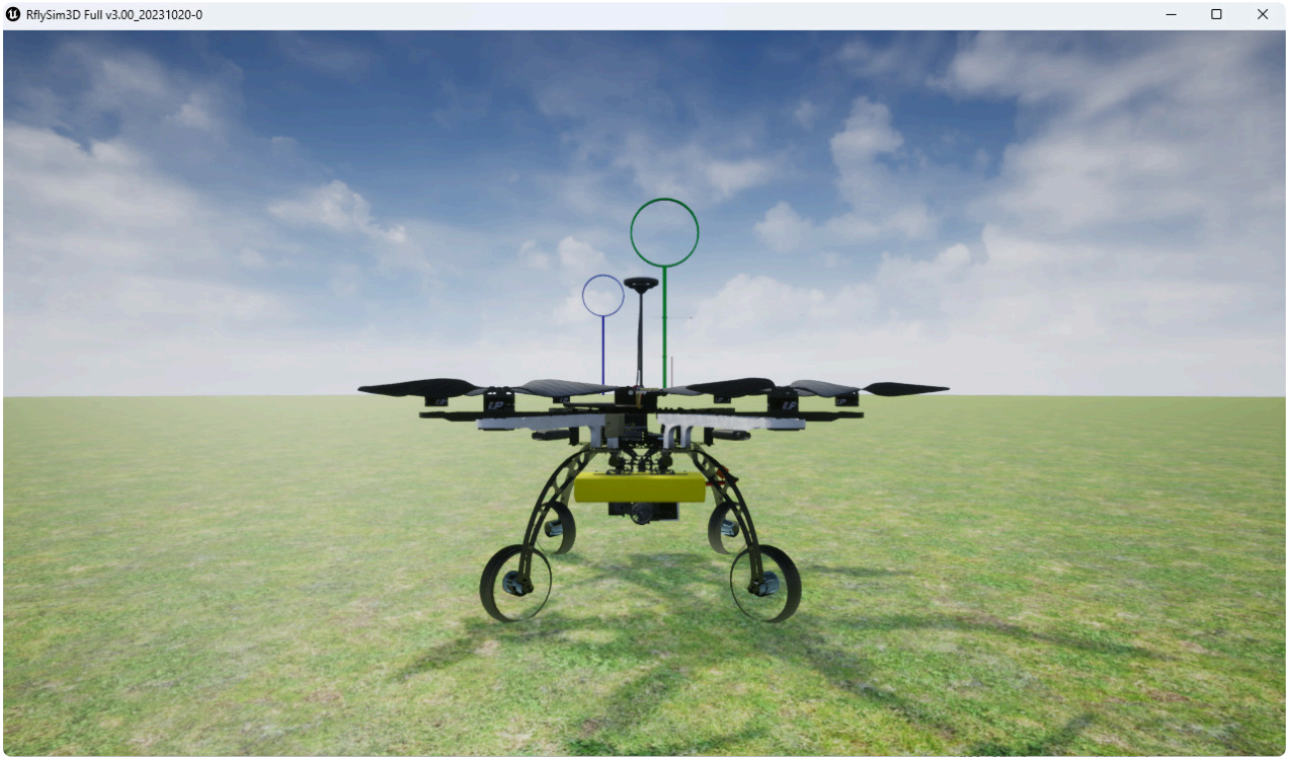


6．运行至“sendReSimMapName”时可以在CopterSim看到地图由Grasslands变化为VisionRing，同时也可以在UE界面观察到地图显示的变化。





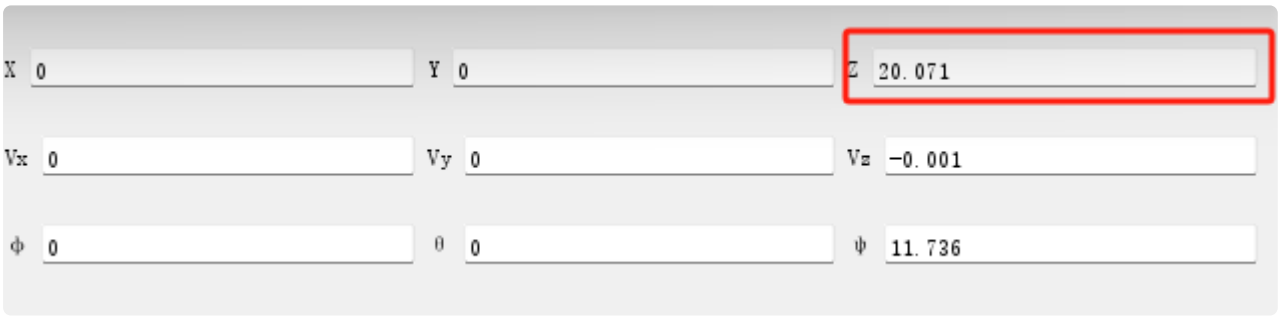




7．完成以上初始化修改操作，Python脚本运行到此处时，能让飞机重新EKF fixed完成，并且进行Offboard控制；用户也可以通过QGC来控制飞机。

```
85   # 开始控制飞机
86   print("5s, Arm the drone")
87   mav.initOffboard()
88   time.sleep(0.5)
89   mav.SendMavArm(True) # Arm the drone
90   print("Arm the drone!, and fly to NED 0,0,-5")
91   time.sleep(0.5)
92   mav.SendPosNED(0, 0, -20, 0) # Fly to target position 0,0, -1.5
93   time.sleep(5)
```

```
问题    输出    终端    调试控制台    端口

发送map地图重设命令
waiting for EKF fixed...
Ekf fixed, continue to control
5s, Arm the drone
Arm the drone!, and fly to NED 0,0,-5
PX4 Armed!
```



注：在一台电脑重新启动Step1的软件在环后，可以在另一台电脑运行“testReqCopterSim.py”程序，程序在获取到进行仿真电脑的IP后，可以远程实现上述现象。

选做实验（VS Code调试运行）

准备工作

- 先确保已经按RflySimAPIs\1.RflySimIntro\2.AdvExps\e3.PythonConfig\Readme.pdf步骤，正确配置VS Code环境。或者配置了自己的Pycharm等自定义Python环境。
- 其他步骤与上文相同，运行testReqCopterSim.py时，可使用VS Code（或Pycharm等工具）来打开testReqCopterSim.py文件，并阅读代码，修改代码，调试执行等。

扩展实验

请自行使用VS Code阅读testReqCopterSim.py源码，通过程序跳转，了解每条代码的执行原理；再通过调试工具，验证每条指令的执行效果。

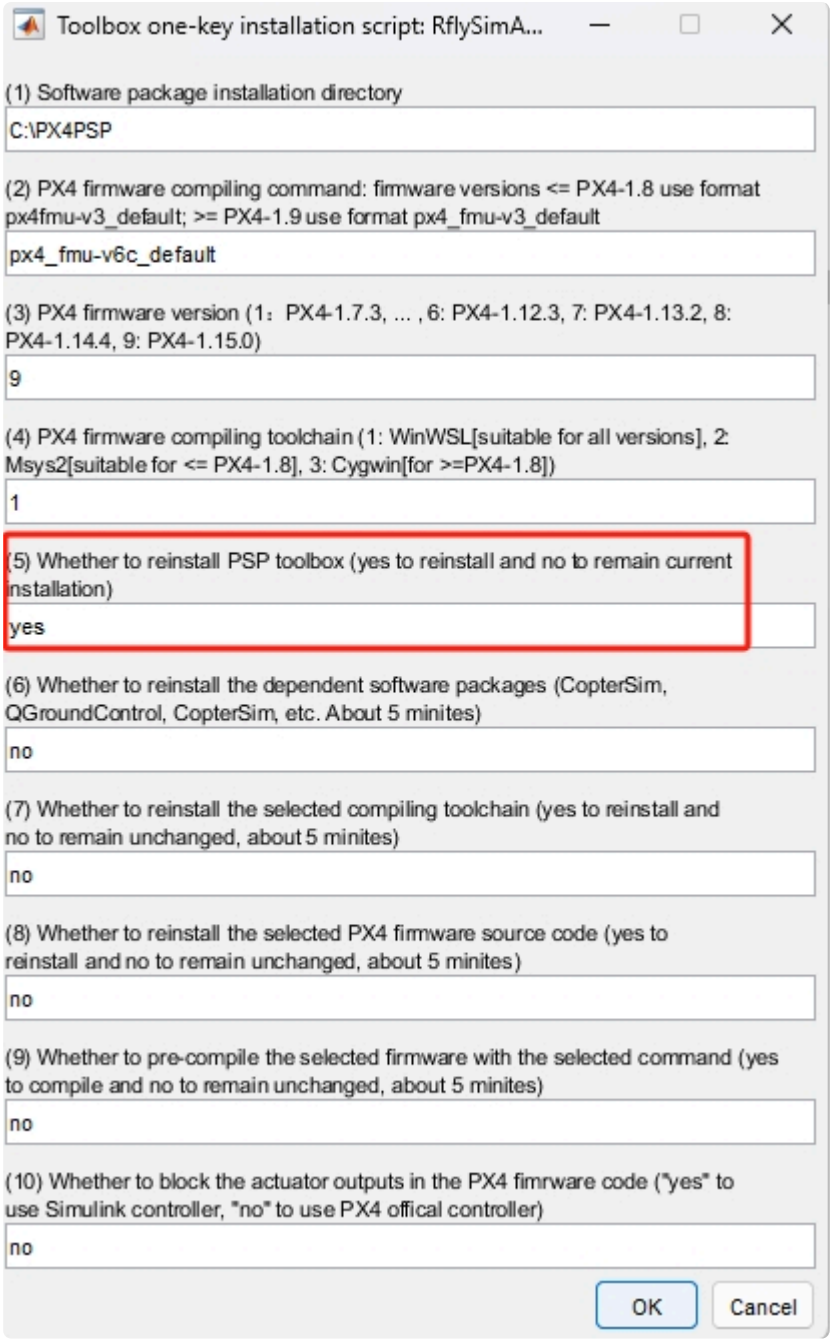
```
# 创建一个CopterSim状态获取实例，并监听2s钟，获取当前所有CopterSim列表数据
req = ReqCopterSim.ReqCopterSim()

# 获取ID和IP列表
IPList=req.getSimIpList()
print(IPList)

# 获取指定ID的电脑IP地址
IP = req.getSimIpID(1)
print(IP)


# 下面展示，如何使用本接口，不需要知道目标电脑IP的情况下，能够连上远程电脑的Copter
CopterID=1  # 计划仿真的飞机ID


## 获取目标电脑IP，并且配置CopterSim回传数据到本电脑
# 获取到指定CopterID的CopterSim所在电脑的IP
TargetIP = req.getSimIpID(CopterID)

# 请求目标CopterSim将数据返回到本电脑
# 通过本接口，可以不用再去bat脚本里面填写IP地址了
# 注：使用本模式，也不需要在bat脚本中开启联机模式
req.sendReSimIP(CopterID)
print('发送联机模式重设命令')
time.sleep(5)
```

6.参考资料

7.常见问题

Q2：编译报错，无法加载库文件

A2：这可能是由于安装平台时PX4PSP工具箱未更新到最新版，更新RflySim安装包后按照如下配置重新安装平台即可


Toolbox one-key installation script: RflySimA...
(1) Software package installation directory
C:\PX4PSP
(2) PX4 firmware compiling command: firmware versions <= PX4-1.8 use format px4fmu-v3_default; >= PX4-1.9 use format px4_fmu-v3_default
px4_fmu-v6c_default
(3) PX4 firmware version (1：PX4-1.7.3, ... , 6: PX4-1.12.3, 7: PX4-1.13.2, 8: PX4-1.14.4, 9: PX4-1.15.0)
9
(4) PX4 firmware compiling toolchain (1: WinWSL[suitable for all versions], 2: Msys2[suitable for <= PX4-1.8], 3: Cygwin[for >=PX4-1.8])
1
(5) Whether to reinstall PSP toolbox (yes to reinstall and no to remain current installation)
yes
(6) Whether to reinstall the dependent software packages (CopterSim, QGroundControl, CopterSim, etc. About 5 minites)
no
(7) Whether to reinstall the selected compiling toolchain (yes to reinstall and no to remain unchanged, about 5 minites)
no
(8) Whether to reinstall the selected PX4 firmware source code (yes to reinstall and no to remain unchanged, about 5 minites)
no
(9) Whether to pre-compile the selected firmware with the selected command (yes to compile and no to remain unchanged, about 5 minites)
no
(10) Whether to block the actuator outputs in the PX4 fimrware code ("yes" to use Simulink controller, "no" to use PX4 offical controller)
no
OK
Cancel