

1. 实验名称及目的

1.1 实验名称

基于系统模板的差动无人车模型验证（Python位置控制）

1.2 实验目的

软硬件在环仿真模式下，以Python的方式通过平台位置控制接口实现单辆/多辆无人车位置控制。

1.3 关键知识点

软/硬件在环仿真的实现

从实现机制的角度分析，可将RflySim平台分为运动仿真模型、底层控制器、三维引擎、外部控制四部分。

- 运动仿真模型：这是模拟飞行器运动的核心部分。在RflySim平台中，运动仿真模型是通过MATLAB/Simulink开发的，然后通过自动生成的C++代码转化成DLL（动态链接库）文件。在使用RflySim平台进行软硬件在环仿真时，会将DLL模型导入到CopterSim，形成运动仿真模型。这个模型在仿真中负责生成飞行器的运动响应，它拥有多个输入输出接口与底层控制器、三维引擎、地面控制站和外部控制进行数据交互，具体数据链路、通信协议及通信端口号见[PX4PSP\RflySimAPIs\4.RflySimModel\API.pdf中的通信接口部分](#)。
- 底层控制器：在软/硬件在环仿真（SIL/HIL）中，真实的飞行控制硬件（如PX4飞行控制器）被集成到一个虚拟的飞行环境中。在软件在环仿真（SIL）中，底层控制器（通过wsl上的PX4仿真环境运行）通过网络通信与运动仿真模型交互数据。在硬件在环仿真（HIL）中，它（将PX4固件在真实的飞行控制器（即飞控）硬件上运行）则通过串口通信与运动仿真模型进行数据交互。飞控与CopterSim通过串口（硬件在环HITL）或网络TCP/UDP（软件在环SITL）进行连接，使用MAVLink进行数据传输，实现控制闭环。
- 三维引擎：这部分负责生成和处理仿真的视觉效果，提供仿真环境和模型的三维视图，使用户能够视觉上跟踪和分析飞行器的运动。CopterSim发送飞机位姿、电机数据到三维引擎，实现可视化展示。

外部控制 (offboard): 从仿真系统外部对飞行器进行的控制, 包括自动飞行路径规划、远程控制指令等。在平台例程中主要通过地面控制站 (QGC)、MATLAB和Python调用对应接口实现。

通过外部控制接口 (python) 进行单辆无人车位置控制

单机控制脚本 `CarR1Diff_OffboardPos1.py` 中依次调用了RflySim平台飞机控制接口协议文件PX4MavCtrlV4.py中定义的以下接口函数

创建通信示例

```
mav1 = PX4MavCtrl.PX4MavCtrler(1)
```

创建一辆无人车的通信示例

启用Mavlink消息监听循环

```
mav1.InitMavLoop()
```

配置CopterSim通信模式, 该函数的参数定义如下:

```
def InitMavLoop(self,UDPMode=2):
```

```
""" Initialize MAVLink listen loop from CopterSim
```

```
0 and 1 for UDP_Full and UDP_Simple Modes, 2 and 3 for MAVLink_Full and  
MAVLink_Simple modes, 4 for MAVLink_NoSend
```

```
The default mode is MAVLink_Full
```

```
"""
```

默认通信模式为**Mavlink_Full**: **Python直接发送MAVLink消息给CopterSim, 再转发给PX4, 数据量较大适合单机控制; 适合单机或少量载具仿真, 载具数量不大于4;

启动外部控制 (offboard)

```
mav1.initOffboard()
```

使px4控制器进入外部控制模式, 且以30HZ的频率发送offboard指令。

注, 虽然在此处已经启用了外部控制模式, 对于运行阶段中flag=0的部分 (解锁和移动到初始位置), 不需要外部控制模式, 实际指令还是由底层控制器完成的。

设定航路点

```
n = 30
```

```
r = 400
```

```
missionPoints=[]
```

```
for i in range(n):
```

```
    angle = 2*math.pi*i/n
```

```
    x=r*math.sin(angle)
```

```
    y=r*math.cos(angle)
```

```
    missionPoints.append([x,y,0])
```

用一组离散的点模拟圆形运动轨迹，并在循环中通过append方法逐个将相应的轨迹点存入目标点列表（missionPoints）。missionPoints.append([x,y,0])表示在missionPoints列表的末尾添加一个新的列表[x,y,0]。

根据欧拉公式：

$$e^{ix} = \cos x + i \sin x$$

这些点将在 x-y 平面上形成一个圆形轨迹。

● 任务阶段

完成上述设置后，程序会通过检查一个 flag 变量的值来决定无人车应该执行哪些动作。

当 flag == 0 时，解锁无人车

解锁车辆

```
mav1.SendMavArm(True)
```

设定启动目标点

```
targetPos=[0, 50, 0]
```

```
mav1.sendMavTakeOff(targetPos[0],targetPos[1],targetPos[2])
```

发送绝对的GPS坐标作为起飞目标点，使用sendMavTakeOffGPS命令，最后三位分别是经度、维度、和高度，会先从uavPosGPSHome向量中提取解锁GPS坐标，在此基础上用绝对坐标

当 flag == 1 时，无人车进入航路寻迹模式

位置检测

```
curPos=mav1.uavPosNED
```

```
dis = math.sqrt((curPos[0]-targetPos[0])**2+(curPos[1]-targetPos[1])**2)
```

计算当前位置和目标位置的水平距离，用于判断是否到达目标位置，以开始下一阶段任务。

航路寻迹模式

```
targetPos=missionPoints[flagI]
```

```
mav1.SendPosNED(targetPos[0], targetPos[1], targetPos[2])
```

会通过航路点索引flagI的值从missionPoints列表中读取相应的航点，并通过SendPosNED函数更新为下一个目标点。

当 flag == 2 时，无人车在航路点之间的运行

```
targetPos=missionPoints[flagI]
```

更新目标位置为 missionPoints 列表中的下一个点。

```
curPos=mav1.uavPosNED
```

```
dis = math.sqrt((curPos[0]-targetPos[0])**2+(curPos[1]-targetPos[1])**2)
```

再次计算无人车与目标位置之间的距离。

```
if dis < 50:
```

```
flagI=flagI+1
```

```
spd = 10+flagI/3.0
```

```
mav1.SendCruiseSpeed(spd)
```

如果距离小于 5米，更新 flagI 以切换到下一个航路点，并调整无人车的巡航速度。

```
else:
```

```
mav1.SendPosNED(targetPos[0], targetPos[1], targetPos[2])
```

如果距离不满足条件，则继续向当前目标位置运行。30个轨迹点取完后，无人车会脱离圆形轨迹

通过外部控制接口（python）进行多辆无人车位置控制

多机控制脚本 `CarR1Diff_OffboardPos4.py` 脚本的实现逻辑与单机控制相同，只是需要创建4辆无人车，再将相同的控制指令复制4份

创建通信示例

```
VehilceNum = 4
```

```
mav=[]
```

```
for i in range(VehilceNum):
```

```
mav=mav+[PX4MavCtrl.PX4MavCtrler(1+i)]
```

创建4辆无人车的通信示例

启用Mavlink消息监听循环

```
for i in range(VehilceNum):
```

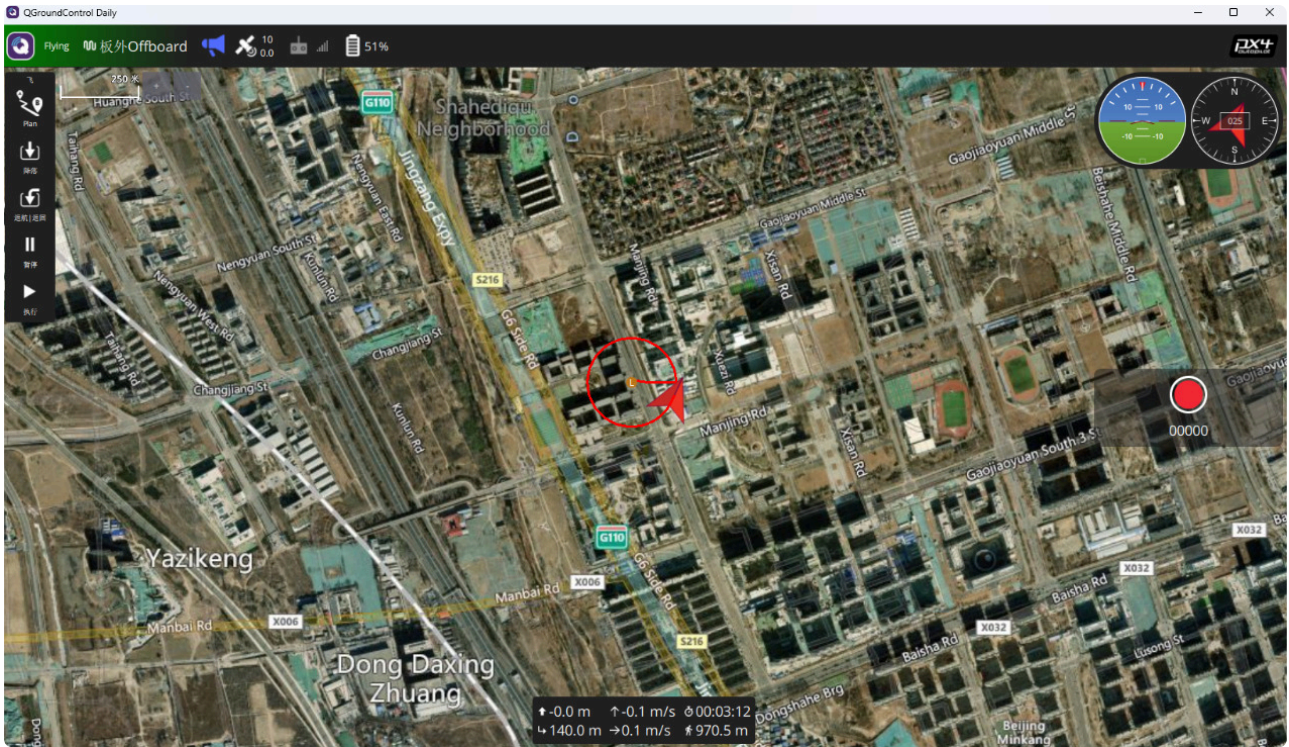
```
mav[i].InitMavLoop()
```

配置4辆无人车的CopterSim通信模式

.....

2.实验效果

通过平台位置控制接口以Python控制单辆/多辆无人车的位置实现画圆。



3. 文件目录

例程目录：

[安装目录]\RflySimAPIs\4.RflySimModel\2.AdvExps\e6_CarR1DiffCtrl\1.CarR1DiffPosCtrl_Py

文件夹/文件名称	说明
CarR1Diff_OffboardPos1.bat	单辆无人车位置控制软件在环仿真批处理文件。
CarR1Diff_OffboardPos1.py	单辆无人车位置控制脚本。
CarR1Diff_OffboardPos4.bat	多辆无人车位置控制软件在环仿真批处理文件。
CarR1Diff_OffboardPos4.py	多辆无人车位置控制脚本。
CarR1Diff_HITLRun.bat	硬件在环批处理文件
CarR1Diff.dll	阿克曼底盘小车DLL模型文件

4. 运行环境

4.1 软件要求

Windows 10及以上版本；RflySim工具链；Python。

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。

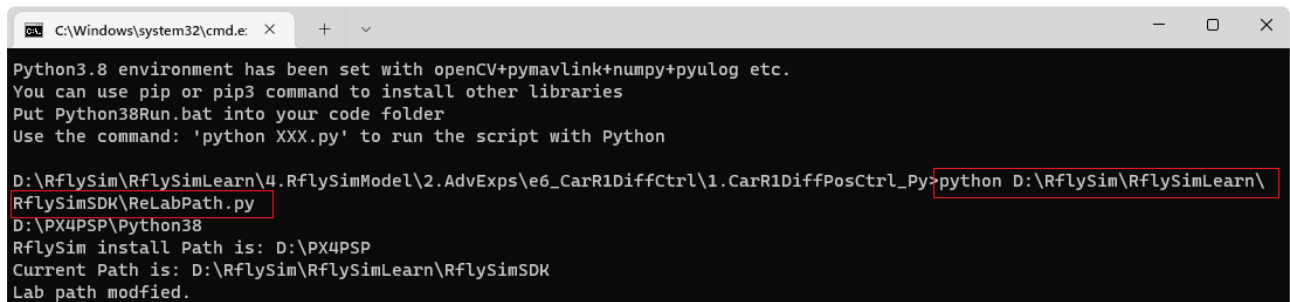
①：推荐配置请见：<https://rflysim.com/>

5. 实验步骤

5.1. Python库文件部署

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 `ReLabPath.py`文件，输入

```
python C:\PX4PSP\RflySimAPIs\RflySimSDK\ReLabPath.py
```

，注意输入要根据实际路径。回车，完成Python公共库环境部署。

```
C:\Windows\system32\cmd.e: X + v
Python3.8 environment has been set with openCV+pymavlink+numpy+pyulog etc.
You can use pip or pip3 command to install other libraries
Put Python38Run.bat into your code folder
Use the command: 'python XXX.py' to run the script with Python

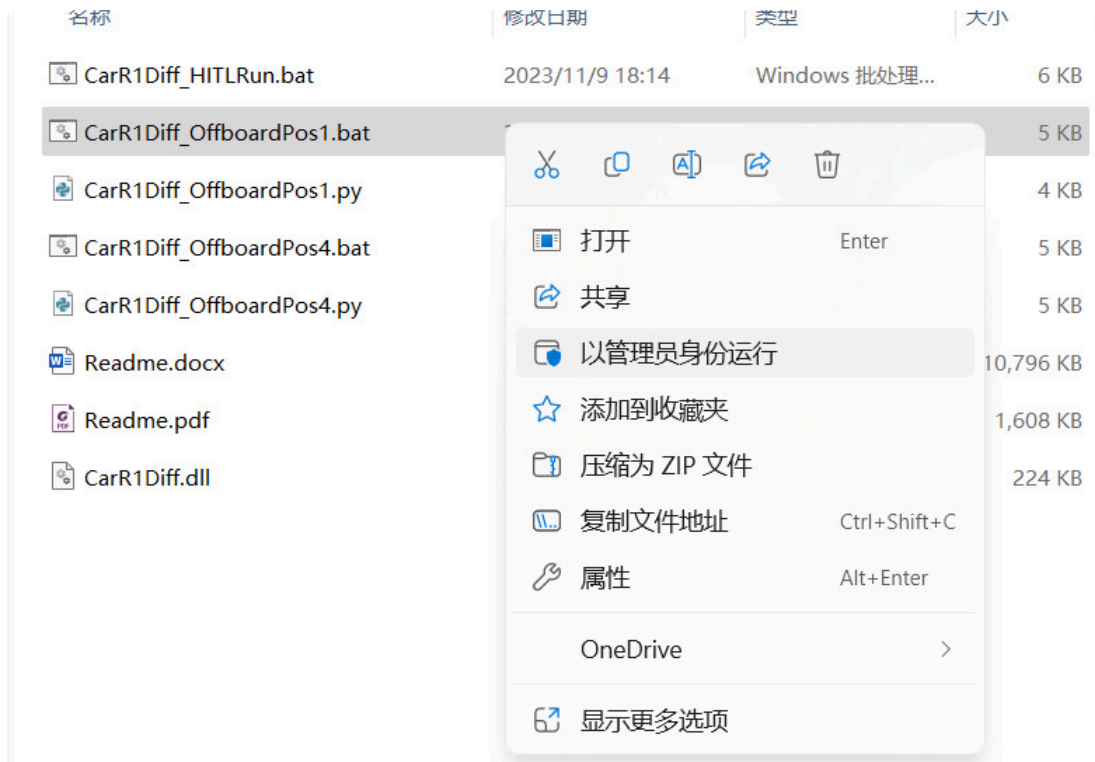
D:\RflySim\RflySimLearn\4.RflySimModel\2.AdvExps\6.CarR1DiffCtrl\1.CarR1DiffPosCtrl_Py>python D:\RflySim\RflySimLearn\
RflySimSDK\ReLabPath.py
D:\PX4PSP\Python38
RflySim install Path is: D:\PX4PSP
Current Path is: D:\RflySim\RflySimLearn\RflySimSDK
Lab path modified.
```

5.2. 必做实验：软件在环仿真

5.2.1 单辆无人车

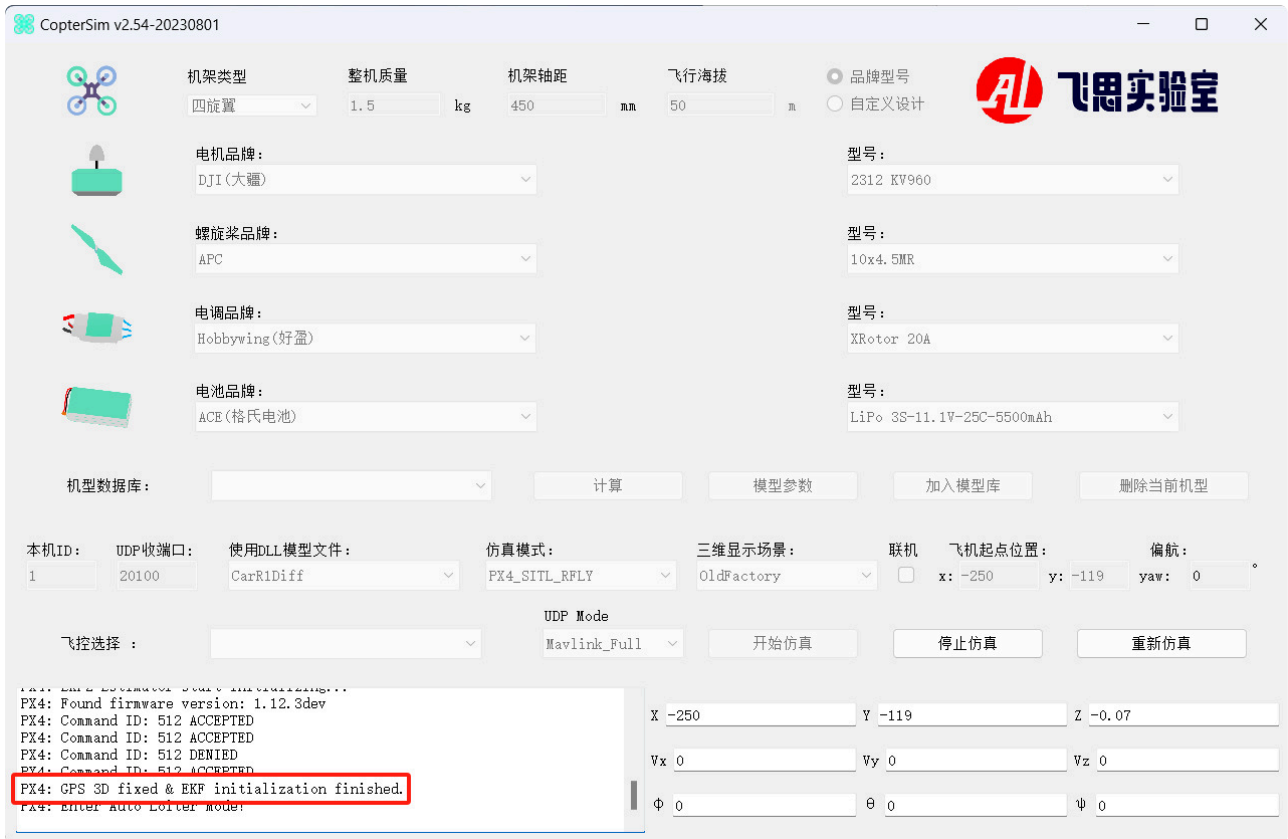
Step 1: 启动仿真

右键以管理员身份运行 `CarR1Diff_OffboardPos1.bat` 批处理文件。



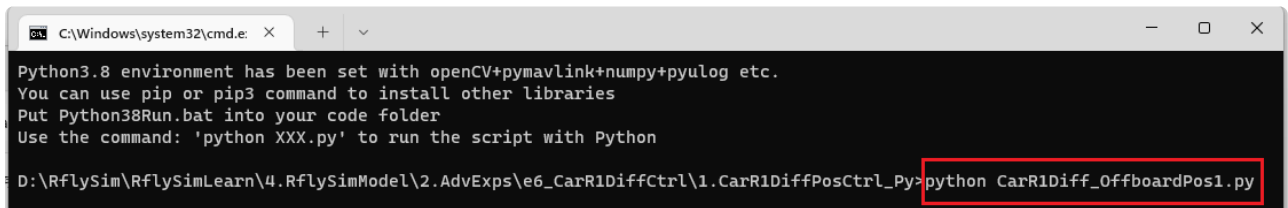
Step 2: 等待初始化完成

等待 CopterSim 中显示 “GPS 3D fixed &EKF initialization finished.”，表明已完成仿真初始化。



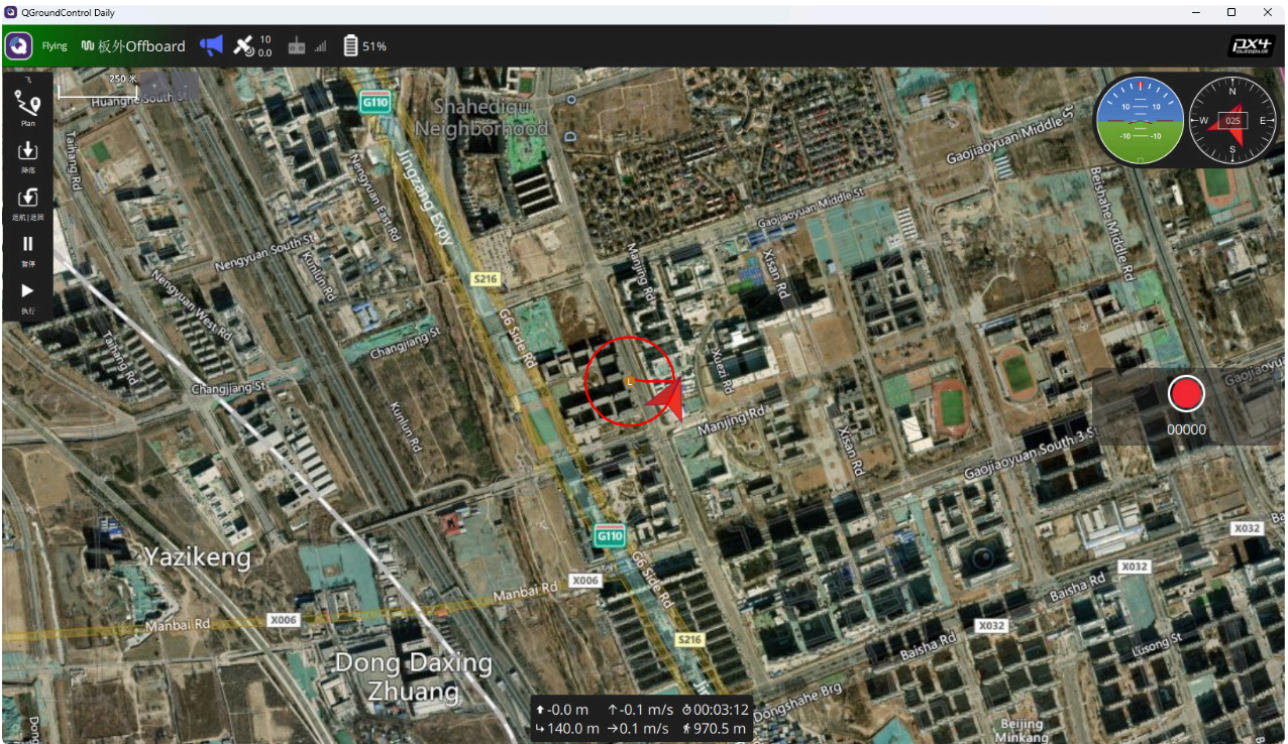
Step 3: 运行控制程序

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 [CarR1Diff_OffboardPos1.py](#) 文件，输入 `python CarR1Diff_OffboardPos1.py`



Step 4: 观察结果

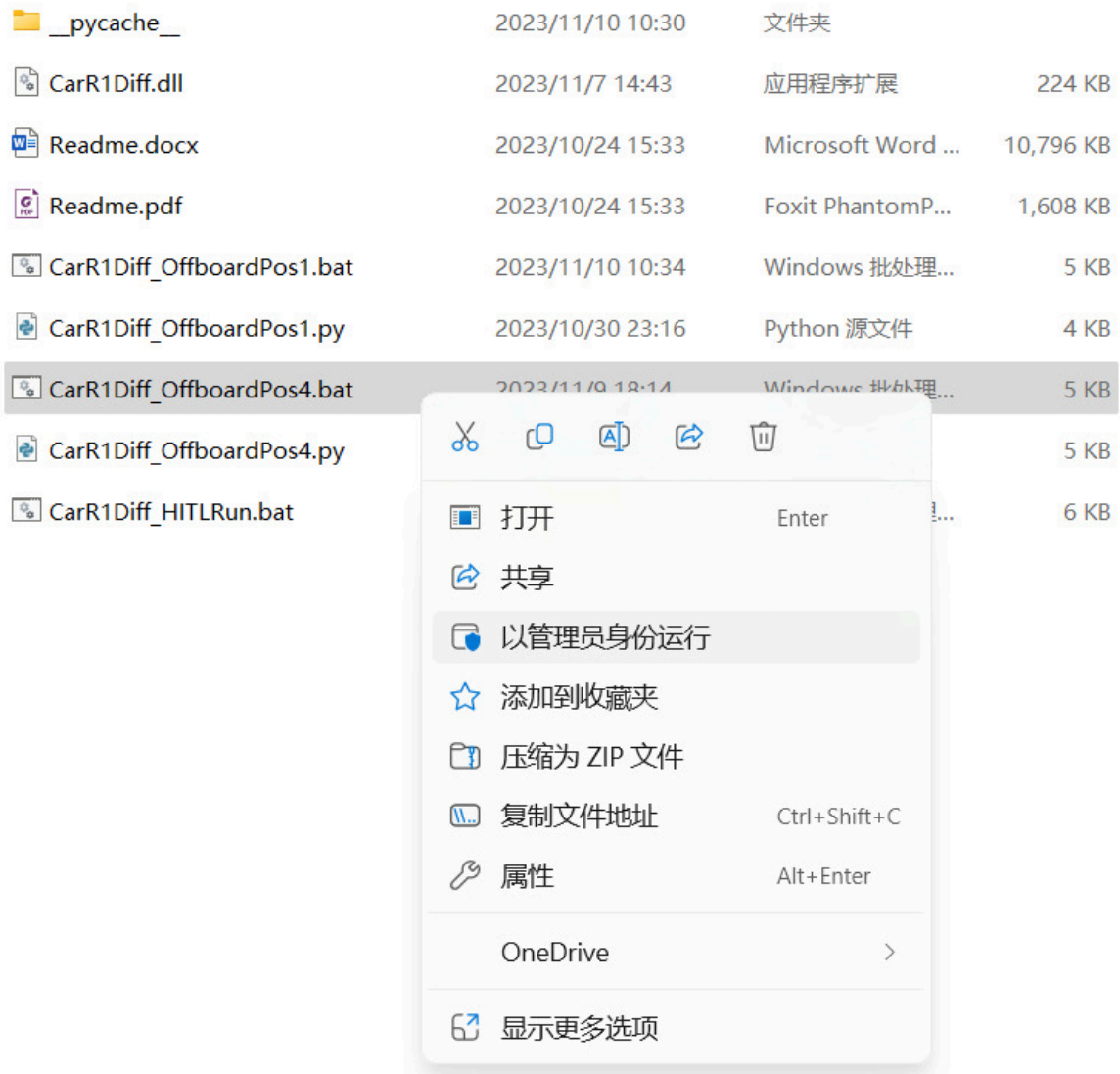
在RflySim3D中观察无人车运行状态，观察 QGC 中无人车的运动轨迹是否为圆形。



5.2.2 多辆无人车

Step 1: 启动仿真

右键以管理员身份运行 `CarR1Diff_OffboardPos4.bat` 批处理文件。



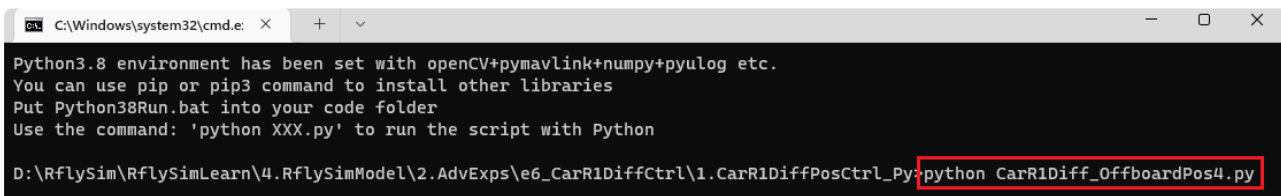
Step 2: 等待初始化完成

等待四辆无人车初始化完成。



Step 3: 运行控制程序

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 `CarR1Diff_OffboardPos4.py` 文件，输入 `python CarR1Diff_OffboardPos4.py`



Step 4: 观察结果

观察 QGC 和 RflySim3D 中无人车的运动轨迹是否为圆形，如下图所示。



5.3. 选做实验：硬件在环仿真

5.3.1 飞控硬件设置

Step 1: 连接飞控

如下图所示，将飞控通过USB线连接电脑，并确保完成硬件在环仿真配置。注意，本图使用Pixhawk6x飞控，其他飞控配置方法类似（推荐使用Pixhawk飞控）。

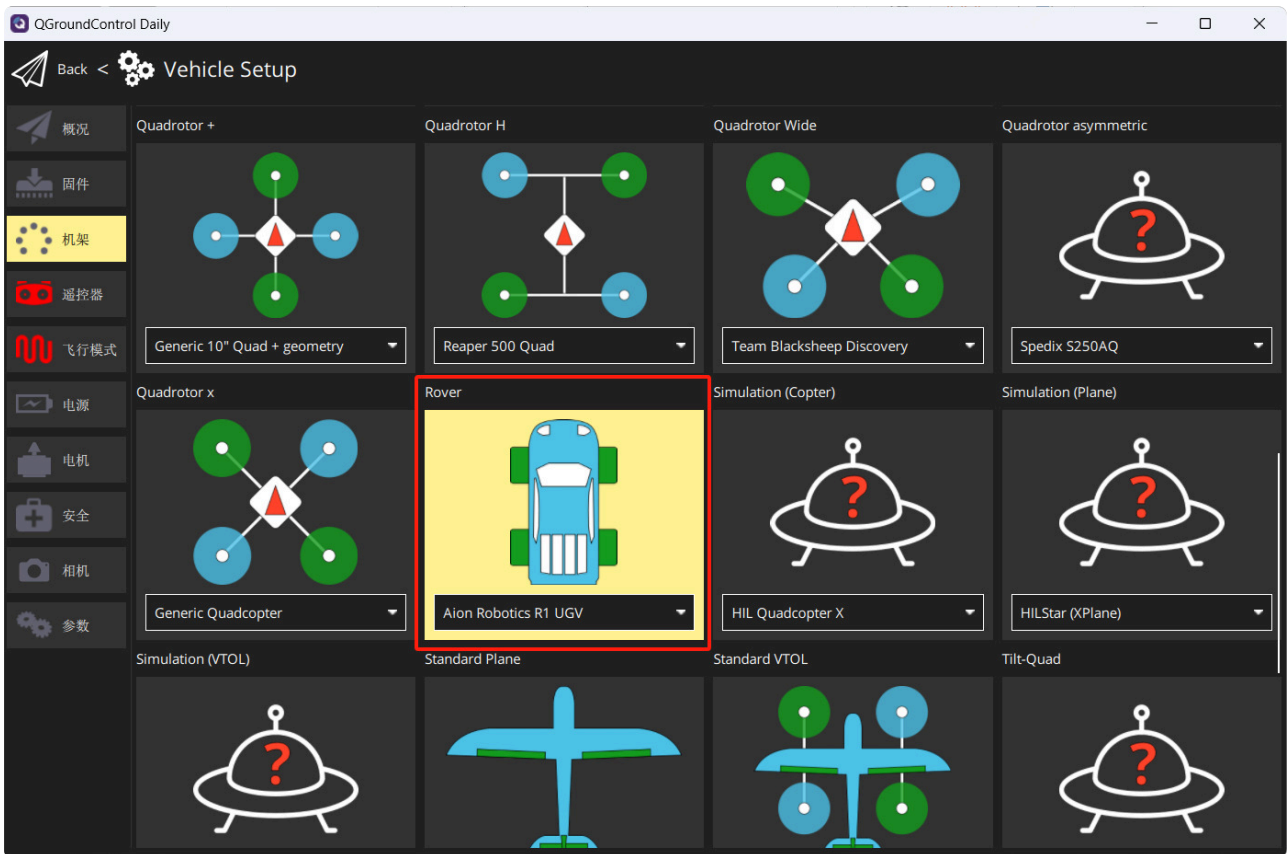


Step 2: 设置硬件在环机架

在 Rflytools 文件夹中打开 QGC 地面站。

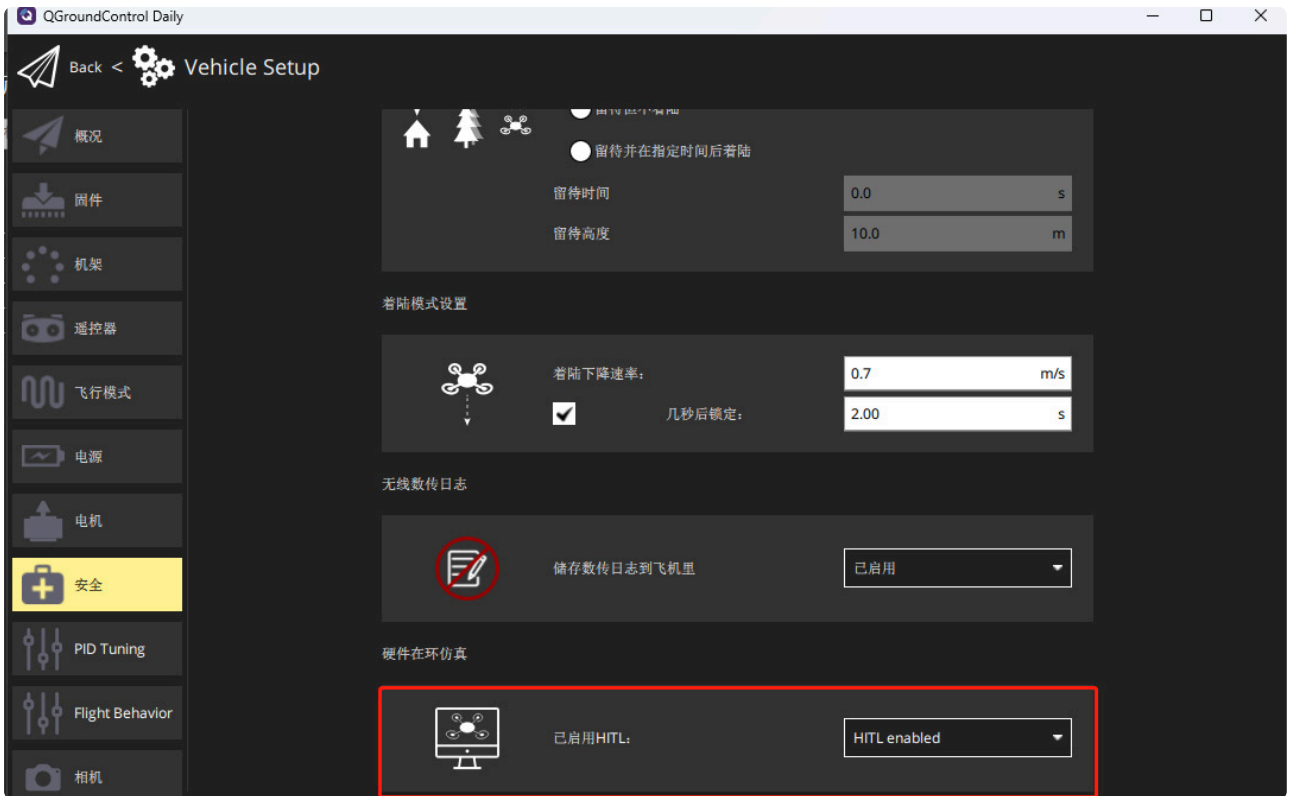
	3DDisplay	2023/7/27 15:02	快捷方式	1 KB
	CopterSim	2023/7/27 15:02	快捷方式	1 KB
	FlightGear-F450	2023/7/27 15:02	快捷方式	2 KB
	HITLRun	2023/7/27 15:02	快捷方式	2 KB
	Python38Env	2023/7/27 15:02	快捷方式	2 KB
	QGroundControl	2023/7/27 15:02	快捷方式	1 KB
	RflySim3D	2023/7/27 15:02	快捷方式	1 KB
	RflySimAPIs	2023/7/27 15:02	快捷方式	1 KB
	RflySimUE5	2023/7/27 15:02	快捷方式	1 KB
	SITLRun	2023/7/27 15:02	快捷方式	2 KB
	Win10WSL	2023/7/27 15:02	快捷方式	2 KB

机架设置为“Aion Robotics R1 UGV”，点击QGC右上角的“应用并重启”。

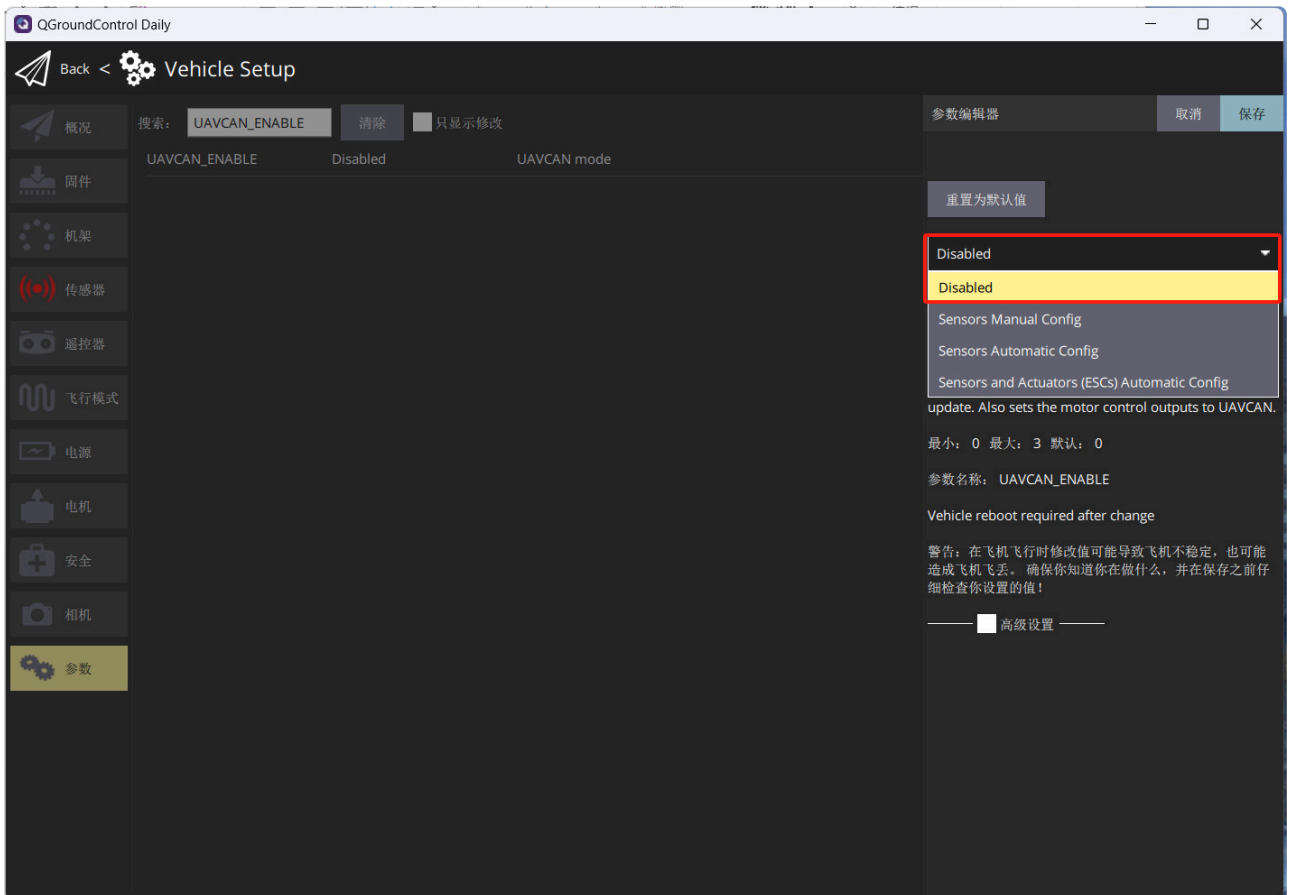


Step 3: 配置硬件在环参数

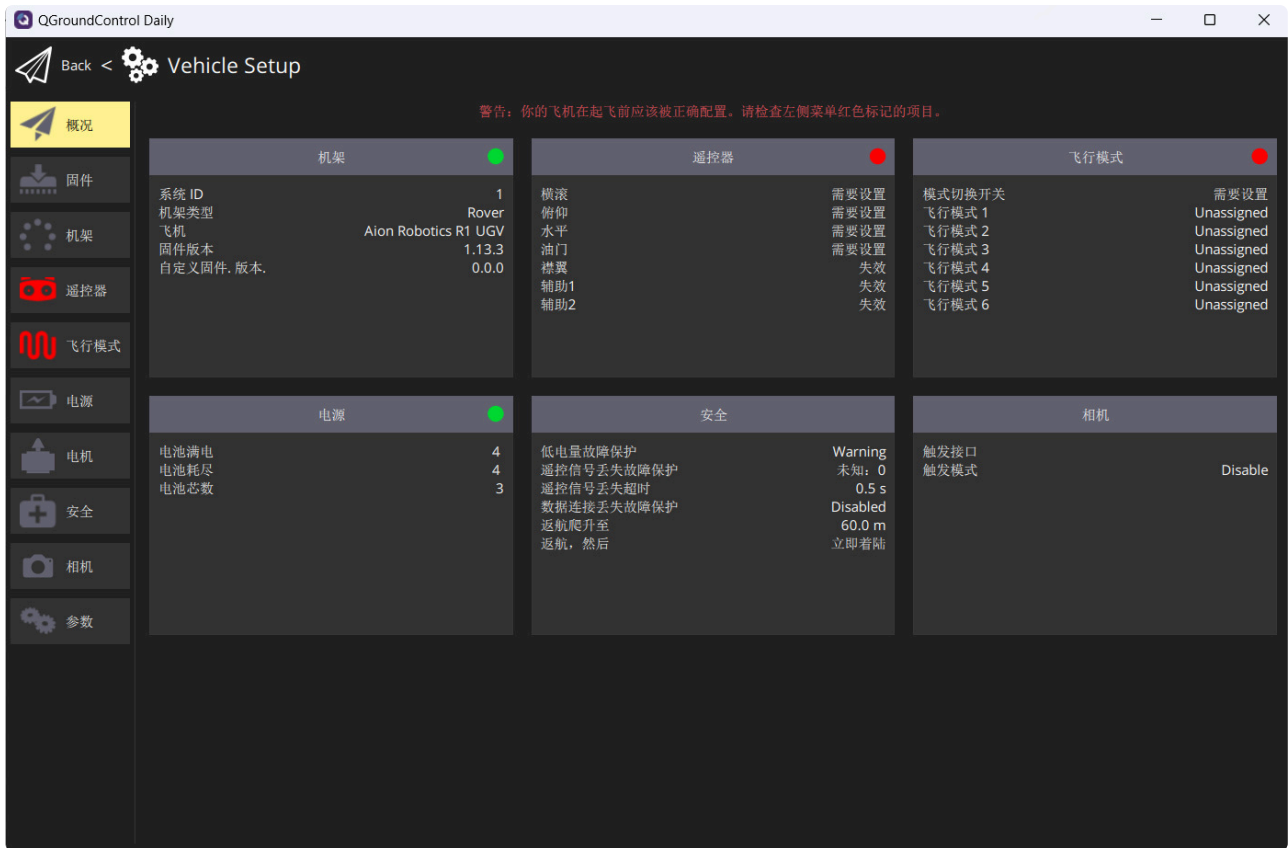
点击“安全”，设置硬件在环仿真为“HITL enabled”，重新插拔飞控。



使用1.13版本固件，还需点击“参数”，在搜索栏中输入“UAVCAN_ENABLE”，在弹出框中设置为“Disabled”，保存后重新插拔飞控即可。



下图为完成硬件在环仿真相关配置后的示意图。



5.3.2 单辆无人车

Step 1: 启动仿真

右键以管理员身份运行 [CarR1Diff_HITLRun.bat](#) 批处理文件。

文件夹	__pycache__	2023/11/10 10:30	文件夹	
应用程序扩展	CarR1Diff.dll	2023/11/7 14:43	应用程序扩展	224 KB
Microsoft Word 文档	Readme.docx	2023/10/24 15:33	Microsoft Word ...	10,796 KB
PDF 文件	Readme.pdf	2023/10/24 15:33	Foxit PhantomP...	1,608 KB
Windows 批处理文件	CarR1Diff_OffboardPos1.bat	2023/11/10 10:34	Windows 批处理...	5 KB
Python 源文件	CarR1Diff_OffboardPos1.py	2023/10/30 23:16	Python 源文件	4 KB
Windows 批处理文件	CarR1Diff_OffboardPos4.bat	2023/11/9 18:14	Windows 批处理...	5 KB
Python 源文件	CarR1Diff_OffboardPos4.py	2023/10/30 23:16	Python 源文件	5 KB
Windows 批处理文件	CarR1Diff_HITLRun.bat	2023/11/9 18:14	Windows 批处理...	6 KB

在终端中根据提示输入串口号，启动一辆无人车的仿真。

```
C:\Windows\System32\cmd.exe
已复制 1 个文件。
-----
Please input the Pixhawk COM port list for HITL
Use ',' as the separator if more than one Pixhawk
E.g., input 3 for COM3 of Pixhawk on the computer
Input 3,6,7 for COM3, COM6 and COM7 of Pixhawks
Available COM ports on this computer are:
COM7: MindPX
Recommended COM list input is: 7
-----
My COM list for HITL simulation is:5
Kill all CopterSims
Start QGroundControl
请按任意键继续. . .
```

Step 2: 仿真过程

之后的与单辆无人车软件在环仿真中的相同，可进行位置控制仿真，运行后可在QGC中观察运行轨迹。

5.3.3 多辆无人车

Step 1: 启动仿真

以管理员身份运行 `CarR1Diff_HITLRun.bat` 脚本，然后输入多个飞控的串口号敲击回车就可以连接多辆无人车。

名称	修改日期	类型	大小
 CarNoCtrl.dll	2022/8/16 23:22	应用程序扩展	226 KB
 CarNoCtrlHITLRun	2022/9/20 17:07	Windows 批处理...	6 KB
 CarNoCtrlOffboardPos1	2022/9/20 17:07	Windows 批处理...	5 KB
 CarNoCtrlOffboardPos1	2022/8/15 13:13	PY 文件	4 KB
 CarNoCtrlOffboardPos4	2022/9/20 17:07	Windows 批处理...	5 KB
 CarNoCtrlOffboardPos4	2022/8/15 13:25	PY 文件	5 KB
 PX4MavCtrlV4	2023/6/7 10:43	PY 文件	137 KB

Step 2: 仿真过程

之后步骤与多辆无人车软件在环仿真中的相同，可进行位置控制仿真，运行后可在RflySim3D中观察运行轨迹。

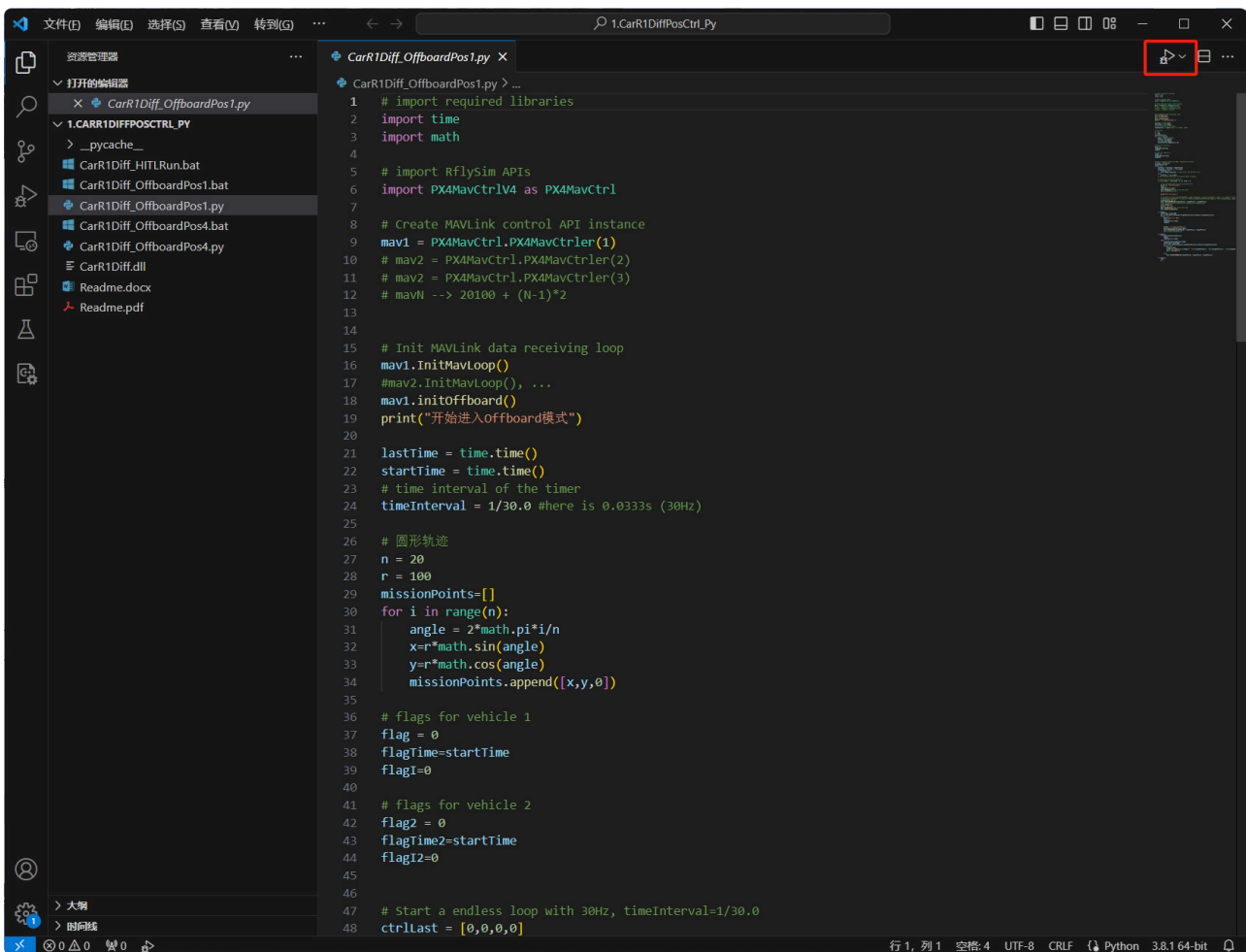
5.4. 选做实验（VS Code调试运行）

准备工作

- 先确保已经按 [RflySimAPIs\1.RflySimIntro\2.AdvExps\3.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 [CarR1Diff_OffboardPos1.py](#) 或者 [CarR1Diff_OffboardPos4.py](#) 时，可使用VS Code（或Pycharm等工具）来打开该文件，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 [CarR1Diff_OffboardPos1.py](#) 以及 [CarR1Diff_OffboardPos4.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。



```
1 # import required libraries
2 import time
3 import math
4
5 # import RflySim APIs
6 import PX4MavCtrlV4 as PX4MavCtrl
7
8 # Create MAVLink control API instance
9 mav1 = PX4MavCtrl.PX4MavCtrl(1)
10 # mav2 = PX4MavCtrl.PX4MavCtrl(2)
11 # mav2 = PX4MavCtrl.PX4MavCtrl(3)
12 # mavN --> 20100 + (N-1)*2
13
14
15 # Init MAVLink data receiving loop
16 mav1.InitMavLoop()
17 #mav2.InitMavLoop(), ...
18 mav1.initOffboard()
19 print("开始进入Offboard模式")
20
21 lastTime = time.time()
22 startTime = time.time()
23 # time interval of the timer
24 timeInterval = 1/30.0 #here is 0.0333s (30Hz)
25
26 # 圆形轨迹
27 n = 20
28 r = 100
29 missionPoints=[]
30 for i in range(n):
31     angle = 2*math.pi*i/n
32     x=r*math.sin(angle)
33     y=r*math.cos(angle)
34     missionPoints.append([x,y,0])
35
36 # flags for vehicle 1
37 flag = 0
38 flagTime=startTime
39 flagI=0
40
41 # flags for vehicle 2
42 flag2 = 0
43 flagTime2=startTime
44 flagI2=0
45
46
47 # Start a endless loop with 30Hz, timeInterval=1/30.0
48 ctrlLast = [0,0,0,0]
```

6.参考资料

1. DLL/SO模型与通信接口..\..\..\PX4PSP\RflySimAPIs\4.RflySimModel\API.pdf
2. 外部控制接口..\..\..\PX4PSP\RflySimAPIs\4.RflySimModel\API.pdf
- 3.

7.常见问题

Q1:

A1:

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 firmware code ("yes" to use Simulink controller, "no" to use PX4 official controller)
no

OK Cancel