

1. 实验名称及目的

Python 控制阿克曼底盘无人车位置软/硬件在环仿真: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。

2. 实验原理

2.1. 软/硬件在环仿真 (SIL/HIL) 的实现[1][2]

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 [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。

注意,针对仿真时的机架设置,在进行软件在环仿真时,需要在对应的 bat 脚本中 set PX4SITLFrame=generic_ground_vehicle,硬件在环时需要在 QGC 机架设置页面进行相同设置。

2.2. 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 CarAckermanOffboardPos1.py 中依次调用了 RflySim 平台载具控制接口协议文件 PX4MavCtrlV4.py 中定义的以下接口函数

创建通信示例

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

创建一架飞机的通信示例

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。

运行阶段

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、纬度、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)
```

计算飞机当前位置和起飞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)
```

再次计算无人机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

2.3. 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 `CarAckermanOffboardPos4.py` 脚本的实现逻辑与单机控制相同，只是需要创建 4 架飞机，再将相同的控制指令复制 4 份

创建通信示例

```
VehilceNum = 3
mav=[]
for i in range(VehilceNum):
    mav=mav+[PX4MavCtrl.PX4MavCtrler(1+i)]
```

创建 3 架飞机的通信示例

启用 Mavlink 消息监听循环

```
for i in range(VehilceNum):  
    mav[i].InitMavLoop()
```

配置 3 架飞机的 CopterSim 通信模式

3. 实验效果

通过平台位置控制接口以 Python 控制单量/多辆无人车的位置实现画圆。

4. 文件目录

文件夹/文件名称	说明
CarAckermanOffboardPos1.bat	单辆无人车位置控制软件在环仿真批处理文件。
CarAckermanOffboardPos1.py	单辆无人车位置控制脚本。
CarAckermanOffboardPos4.bat	多辆无人车位置控制软件在环仿真批处理文件。
CarAckermanOffboardPos4.py	多辆无人车位置控制脚本。
CarAckermanHITLRun.bat	硬件在环批处理文件
CarAckerman.dll	无控制器的阿克曼底盘小车 DLL 模型文件

5. 运行环境

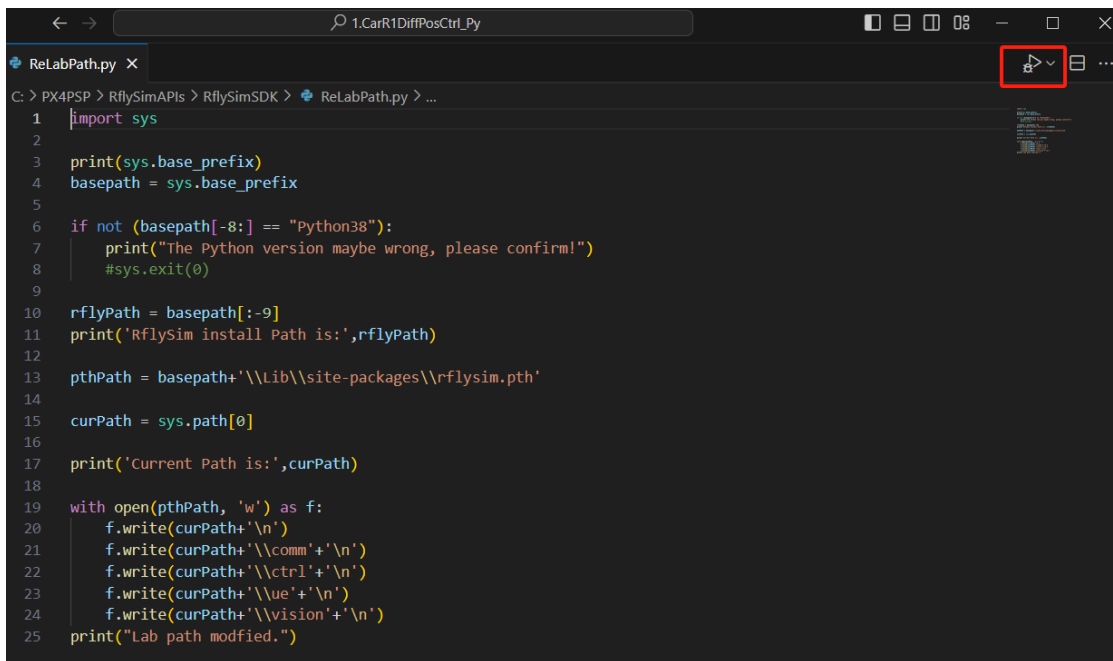
序号	软件要求	硬件要求	
		名称	数量
1	Windows 10 及以上版本	笔记本/台式电脑 ^①	1
2	RflySim 平台高级版	PX4 飞控 ^②	1
3	Python	数据线	1

- ① 推荐配置请见：<https://doc.rflysim.com>
- ② 须保证平台安装时的编译命令为：px4_fm4-v6c_default，固件版本为：1.13.3。其他配套飞控请见：<http://doc.rflysim.com/hardware.html>

6. 实验步骤

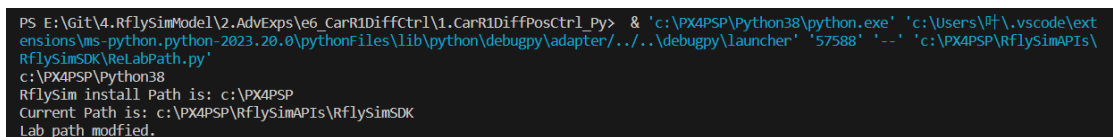
6.1. Python 库文件部署

以 VsCode 打开 “C:\PX4PSP\RflySimAPIs\RflySimSDK\ReLabPath.py”，并运行。



```
1 import sys
2
3 print(sys.base_prefix)
4 basepath = sys.base_prefix
5
6 if not (basepath[-8:] == "Python38"):
7     print("The Python version maybe wrong, please confirm!")
8     #sys.exit(0)
9
10 rflyPath = basepath[:-9]
11 print('Rflysim install Path is:',rflyPath)
12
13 pthPath = basepath+'\\Lib\\site-packages\\rflysim.pth'
14
15 curPath = sys.path[0]
16
17 print('Current Path is:',curPath)
18
19 with open(pthPath, 'w') as f:
20     f.write(curPath+'\n')
21     f.write(curPath+'\\comm'+'\n')
22     f.write(curPath+'\\ctrl'+'\n')
23     f.write(curPath+'\\ue'+'\n')
24     f.write(curPath+'\\vision'+'\n')
25 print("Lab path modified.")
```

完成 Python 公共库环境部署。









```
PS E:\Git\4.RflySimModel\2.AdvExps\6.CarR1DiffCtrl\1.CarR1DiffPosCtrl_Py> & 'c:\PX4PSP\Python38\python.exe' 'c:\Users\H\OneDrive\Documents\ms-python.python-2023.20.0\pythonFiles\lib\python\debugpy\adapter\..\..\debugpy\launcher' '57588' '--' 'c:\PX4PSP\RflySimAPIs\RflySimSDK\ReLabPath.py'
c:\PX4PSP\Python38
RflySim install Path is: c:\PX4PSP
Current Path is: c:\PX4PSP\RflySimAPIs\RflySimSDK
Lab path modified.
```

6.2. 软件在环仿真

6.2.1. 单辆无人车仿真

Step 1:

右键以管理员身份运行 CarAckermanOffboardPos1.bat 批处理文件。

	CarAckerman.dll	2023/11/10 14:04	应用程序扩展	218 KB
	CarAckermanboardPos1.py	2023/10/24 15:33	PY 文件	4 KB
	CarAckermanHITLRun.bat	2023/11/10 14:11	Windows 批处理...	6 KB
	CarAckermanOffboardPos1.bat	2023/11/10 14:11	Windows 批处理...	5 KB
	CarAckermanOffboardPos4.bat	2023/11/10 14:11	Windows 批处理...	5 KB
	CarAckermanOffboardPos4.py	2023/10/24 15:33	PY 文件	5 KB
	PX4MavCtrlV4.py	2023/10/24 15:33	PY 文件	139 KB








Step 2:

等待 CopterSim 中显示已连接上 RflySim3D 。

```
CopterSim: Receive Mavlink heartbeat
PX4: Init MAVLink
PX4: Awaiting GPS/EKF fixed for Position control...
PX4: Enter Other Mode!
PX4: Enter Manual Mode!
PX4: EKF2 Estimator start initializing...
PX4: [logger] ./log/2021-11-29/09_55_29.ulog
PX4: GPS 3D fixed & EKF initialization finished.
PX4: Enter Auto Loiter Mode!
```

Step 3:

右键以 VsCode 打开 CarAckermanOffboardPos1.py 脚本，并点击运行该脚本。

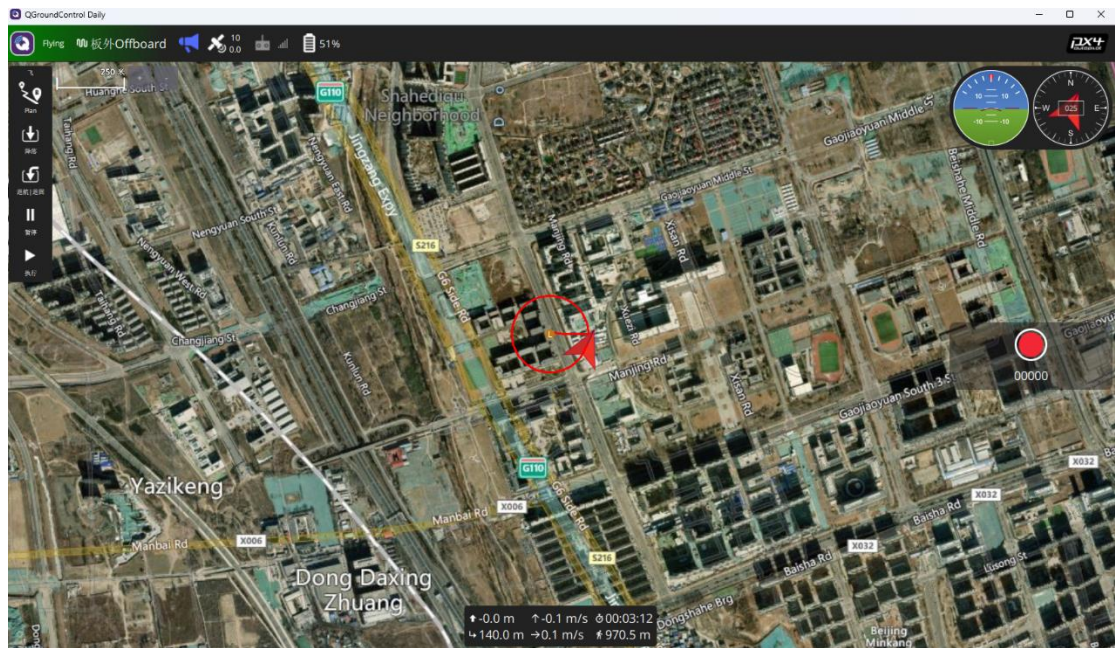
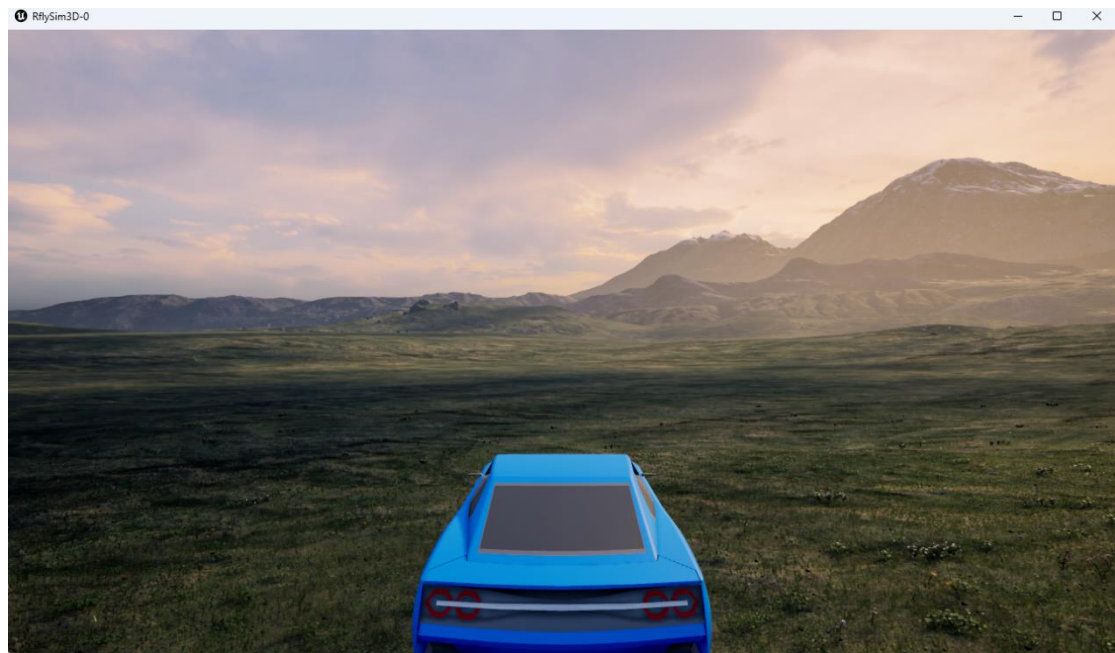
	CarAckerman.dll	2023/11/10 14:04	应用程序扩展	218 KB
	CarAckermanboardPos1.py	2023/10/24 15:33	PY 文件	4 KB
	CarAckermanHITLRun.bat	2023/11/10 14:11	Windows 批处理...	6 KB
	CarAckermanOffboardPos1.bat	2023/11/10 14:11	Windows 批处理...	5 KB
	CarAckermanOffboardPos4.bat	2023/11/10 14:11	Windows 批处理...	5 KB
	CarAckermanOffboardPos4.py	2023/10/24 15:33	PY 文件	5 KB
	PX4MavCtrlV4.py	2023/10/24 15:33	PY 文件	139 KB

```
CarNoCtrlOffboardPos1.py 9+ X
D: > OneDrive > 桌面 > 阿克曼底盘无人车位置控制-Python > CarNoCtrlOffboardPos1.py > ...
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.PX4MavCtrlr(20100)
10 # mav2 = PX4MavCtrl.PX4MavCtrlr(20102)
11 # mav2 = PX4MavCtrl.PX4MavCtrlr(20104)
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

PROBLEMS 55 OUTPUT TERMINAL DEBUG CONSOLE
PS D:\OneDrive\桌面\阿克曼底盘无人车速度控制-Python> d:; cd 'd:\OneDrive\桌面\阿克曼底盘无人车速度控制-PythonFiles\lib\python\debugpy\adapter\..\..\debugpy\launcher' '56405' '--' 'D:\OneDrive\桌面\阿克曼底盘无人车速度控制-PythonFiles\lib\python\debugpy\adapter\..\..\debugpy\launcher'
开始进入Offboard模式
```

Step 4:








在 UE4 中观察无人车运行状态，观察 QGC 中无人车的运动轨迹。



6.2.2. 多辆无人车仿真

Step 1:

右键以管理员身份运行 CarAckermanOffboardPos4.bat 批处理文件。

	CarAckerman.dll	2023/11/10 14:04	应用程序扩展	218 KB
	CarAckermanboardPos1.py	2023/10/24 15:33	PY 文件	4 KB
	CarAckermanHITLRun.bat	2023/11/10 14:11	Windows 批处理...	6 KB
	CarAckermanOffboardPos1.bat	2023/11/10 14:11	Windows 批处理...	5 KB
	CarAckermanOffboardPos4.bat	2023/11/10 14:11	Windows 批处理...	5 KB
	CarAckermanOffboardPos4.py	2023/10/24 15:33	PY 文件	5 KB
	PX4MavCtrlV4.py	2023/10/24 15:33	PY 文件	139 KB









Step 2:

等待四辆无人车初始化完成。



Step 3:

右键以 VsCode 打开并运行 CarAckermanOffboardPos4.py 文件，

	CarAckerman.dll	2023/11/10 14:04	应用程序扩展	218 KB
	CarAckermanboardPos1.py	2023/10/24 15:33	PY 文件	4 KB
	CarAckermanHITLRun.bat	2023/11/10 14:11	Windows 批处理...	6 KB
	CarAckermanOffboardPos1.bat	2023/11/10 14:11	Windows 批处理...	5 KB
	CarAckermanOffboardPos4.bat	2023/11/10 14:11	Windows 批处理...	5 KB
	CarAckermanOffboardPos4.py	2023/10/24 15:33	PY 文件	5 KB
	PX4MavCtrlV4.py	2023/10/24 15:33	PY 文件	139 KB
	Readme.docx	2023/11/10 14:54	Microsoft Word ...	10,710 KB

```
CarNoCtrlOffboardPos4.py 9+ X
D: > OneDrive > 桌面 > 阿克曼底盘无人车位置控制-Python > CarNoCtrlOffboardPos4.py > ...

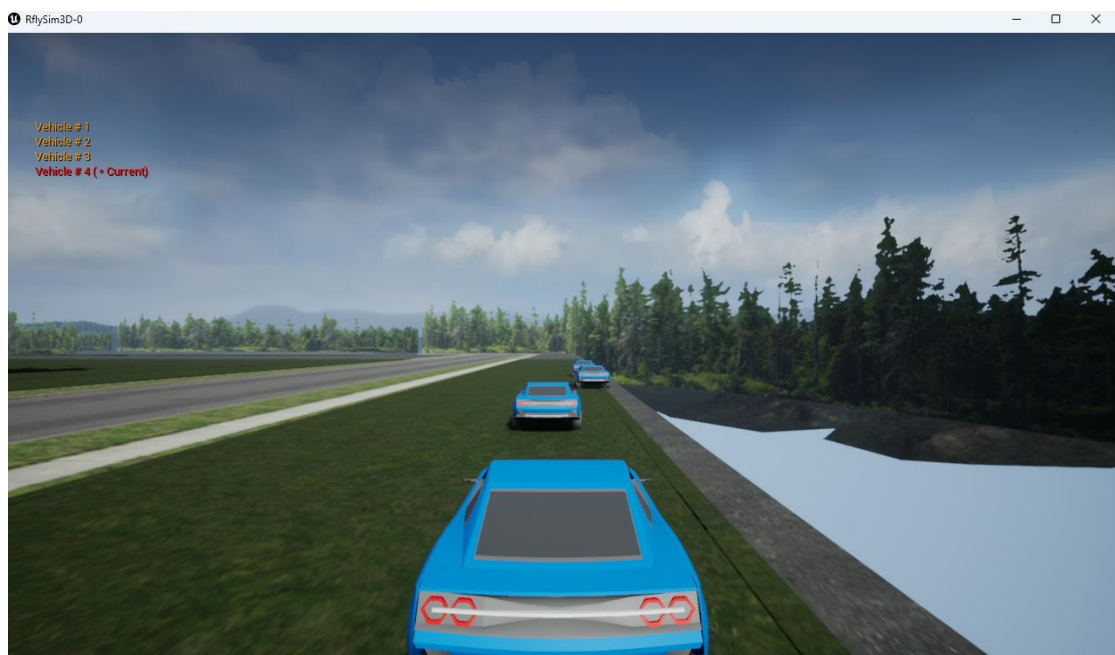
1  # import required libraries
2  import time
3  import math
4
5  # import RflySim APIs
6  import PX4MavCtrlV4 as PX4MavCtrl
7
8  VehicleNum = 4
9  # Create MAVLink control API instance
10 mav=[]
11 for i in range(VehicleNum):
12     mav=mav+[PX4MavCtrl.PX4MavCtrl(20100+i*2)]
13
14 # mavN --> 20100 + (N-1)*2
15
16
17 # Init MAVLink data receiving loop
18 print("开始进入Offboard模式")
19 for i in range(VehicleNum):
20     mav[i].InitMavLoop()
21     mav[i].initOffboard()
```

PROBLEMS 71 OUTPUT TERMINAL DEBUG CONSOLE

```
PS D:\OneDrive\桌面\阿克曼底盘无人车位置控制-Python> d:; cd 'd:\OneDrive\桌面\阿克曼底盘无人车位置控制-PythonFiles\lib\python\debugpy\adapter\...\debugpy\launcher' '56435' '--' 'D:\OneDrive\桌面\阿克曼底盘无人车位置控制-PythonFiles\lib\python\debugpy\adapter\...\debugpy\launcher'
开始进入Offboard模式
5s, Arm the drone
Arm the drone!
开始行驶
```

Step 4:

观察 QGC 和 RflySim3D 中无人车的运动轨迹如下图所示。





6.3. 硬件在环仿真

6.3.1. 飞控硬件设置












Step 1:

按下图所示将飞控与计算机链接，飞控上的接口名称为 USB。



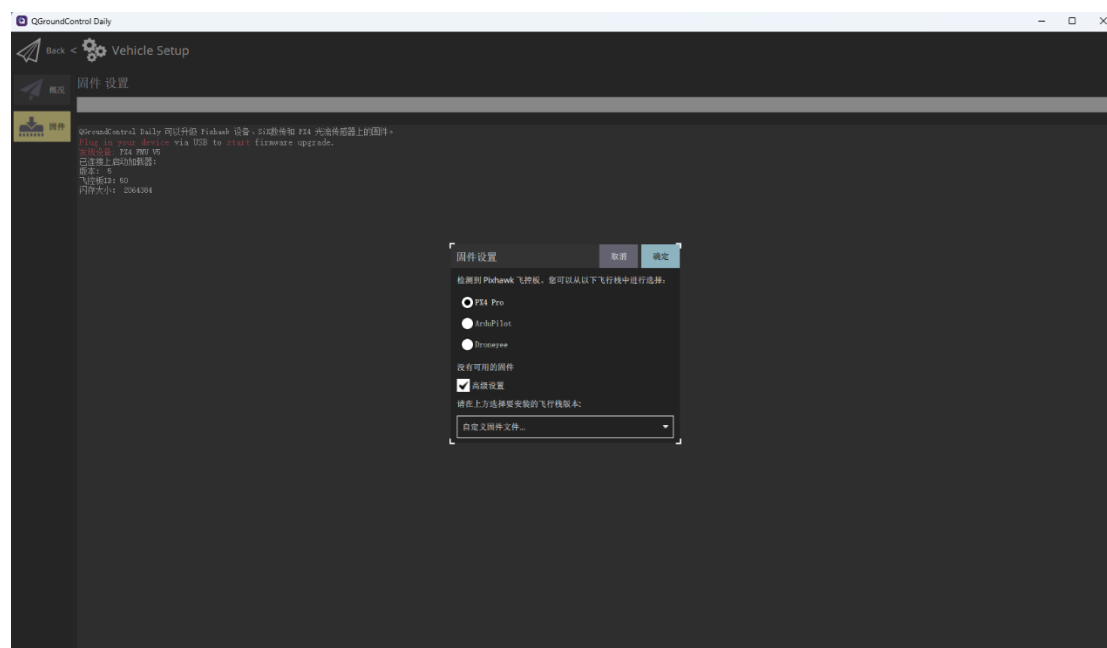
Step 2:

在 Rflytools 文件夹中打开 QGC 地面站。

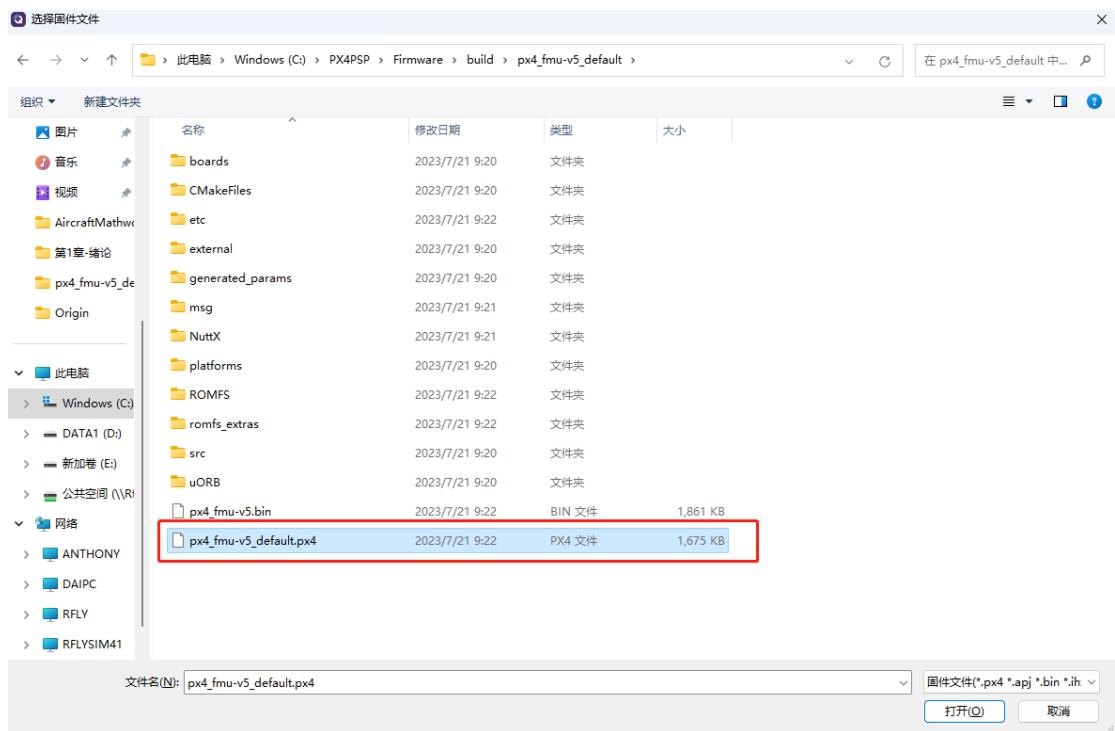
	3DDisplay	2023/7/27 15:02	快捷方式	1 KB
	CopterSim	2023/7/27 15:02	快捷方式	1 KB
	FlightGear-F450	2023/7/27 15:02	快捷方式	2 KB
	HITLRun	2023/7/27 15:02	快捷方式	2 KB
	Python38Env	2023/7/27 15:02	快捷方式	2 KB
	QGroundControl	2023/7/27 15:02	快捷方式	1 KB
	RflySim3D	2023/7/27 15:02	快捷方式	1 KB
	RflySimAPIs	2023/7/27 15:02	快捷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

Step 3:

点击进入左侧“固件”界面后，勾选下方“高级设置”选择自定义固件文件。



在 C:\PX4PSP\Firmware\build\px4_fmu-v5_default 这个路径下选择确认 px4_fmu-v5_default.px4 文件（v6c 同理）。



如果选择卓翼 H7 飞控的话，则在 C:\PX4PSP\Firmware\build\droneyee_zyfc-h7_default 这个路径下确认 droneyee_zyfc-h7_default.px4 文件。

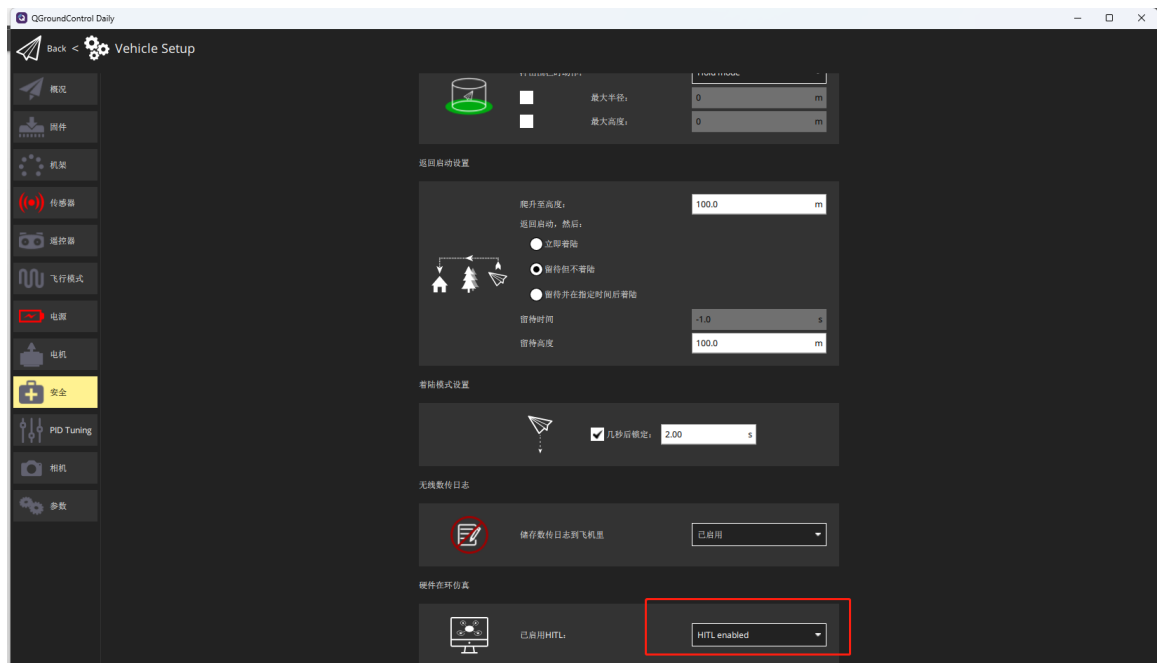
名称	修改日期	类型	大小
boards	2023/7/28 11:05	文件夹	
CMakeFiles	2023/7/28 11:05	文件夹	
etc	2023/7/28 11:07	文件夹	
external	2023/7/28 11:05	文件夹	
generated_params	2023/7/28 11:05	文件夹	
msg	2023/7/28 11:06	文件夹	
NuttX	2023/7/28 11:06	文件夹	
platforms	2023/7/28 11:05	文件夹	
ROMFS	2023/7/28 11:07	文件夹	
romfs_extras	2023/7/28 11:07	文件夹	
src	2023/7/28 11:05	文件夹	
uORB	2023/7/28 11:05	文件夹	
droneyee_zyfc-h7.bin	2023/7/28 11:07	BIN 文件	1,805 KB
droneyee_zyfc-h7_default.px4	2023/7/28 11:07	PX4 文件	1,617 KB

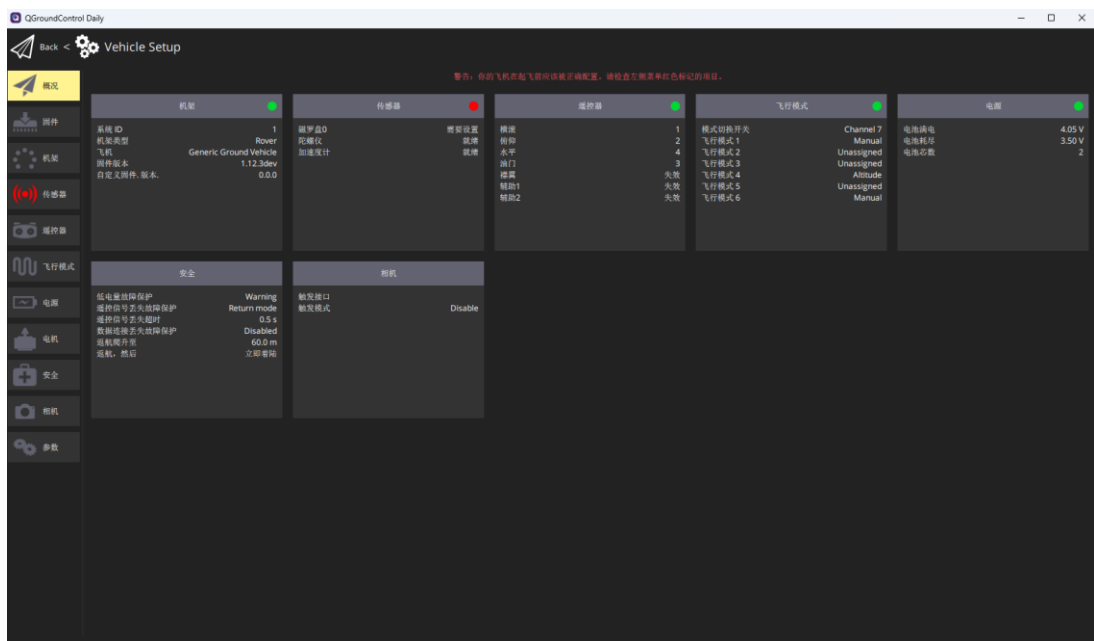
Step 4:

在机架界面设置机架型号为“Generic Ground Vehicle”，设置完毕后重插拔飞控

The screenshot displays the QGroundControl 'Vehicle Setup' interface. The left sidebar contains navigation options: Summary, Firmware, Airframe (selected), Sensors, Radio, Flight Modes, Power, Motors, Safety, Camera, and Parameters. The main panel is divided into a 3x3 grid of vehicle configuration slots. Each slot features a vehicle icon, a dropdown menu for the vehicle name, and a label below. A red arrow points to the 'Generic Ground Vehicle' dropdown in the top-left slot. The other slots are: 'HIL Quadcopter X' (top-middle), 'HILStar (XPlane)' (top-right), 'SIH Tailsitter Duo' (middle-left), 'Standard Plane' (middle-middle), 'HIL Standard VTOL QuadPlane' (middle-right), 'Tilt-Quad' (bottom-left), 'Tricopter Y+' (bottom-middle), and 'Tricopter Y-' (bottom-right).

在“安全”界面，选择“**HITL enabled**”启动硬件在环仿真，之后在概况界面中确认配置完成后，重新插拔飞控完成设置。





6.3.2. 单辆无人车仿真

Step 1:

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

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:

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 3:

之后步骤与单辆无人车软件在环仿真中的 Step3 到 Step4 相同，可进行位置控制仿真，运行后可在 QGC 中观察运行轨迹。

6.3.3. 多辆无人车仿真

Step 1:

4 辆无人车以管理员身份运行 CarAckermanHITLRun.bat 脚本，然后输入 4 个飞控的串口号敲击回车就可以连接 4 架无人车。

	CarAckerman.dll	2023/11/10 14:04	应用程序扩展	218 KB
	CarAckermanboardPos1.py	2023/10/24 15:33	PY 文件	4 KB
	CarAckermanHITLRun.bat	2023/11/10 14:11	Windows 批处理...	6 KB
	CarAckermanOffboardPos1.bat	2023/11/10 14:11	Windows 批处理...	5 KB
	CarAckermanOffboardPos4.bat	2023/11/10 14:11	Windows 批处理...	5 KB
	CarAckermanOffboardPos4.py	2023/10/24 15:33	PY 文件	5 KB
	PX4MavCtrlV4.py	2023/10/24 15:33	PY 文件	139 KB
	Readme.docx	2023/11/10 14:54	Microsoft Word ...	10,710 KB

Step 2:

之后步骤与多辆无人车软件在环仿真中的 Step3 到 Step4 相同，可进行位置控制仿真，运行后可在 QGC 中观察运行轨迹。

7. 参考资料

- [1]. DLL/SO 模型与通信接口 [..\..\API.pdf](#)
- [2]. 外部控制接口 [..\..\API.pdf](#)
- [3].

8. 常见问题

- Q1.
- A1.