Python网络仿真API实验

1. 实验目的

本实验旨在深入理解和掌握多无人机系统的网络仿真API应用。通过搭建SITL（Software In The Loop）仿真环境，学习如何使用NetSimAPI实现多架无人机之间的协同控制与数据共享。实验将使学习者熟悉以下关键概念和技能：

- 理解多无人机系统中网络通信的基本原理和应用场景
- 掌握NetSimAPI网络仿真接口的设计与实现方法
- 学习多线程编程在无人机通信中的应用
- 掌握基于PX4MavCtrlV4的MAVLink通信机制
- 实现跨平台（Windows/Linux）的无人机协同通信机制
- 学习无人机编队控制和追踪算法的实现方法

2. 实验要求

此编写实验目的

- 软件要求：Windows 10及以上版本；RflySim工具链[1]。
- 硬件要求：笔记本/台式电脑1台[2]。

3.实验地址

例程目录：
[安装目录]\RflySimAPIs\9.RflySimComm\0.ApiExps\e1.RflyNetAPIExps\6.PythonNetSimAPI

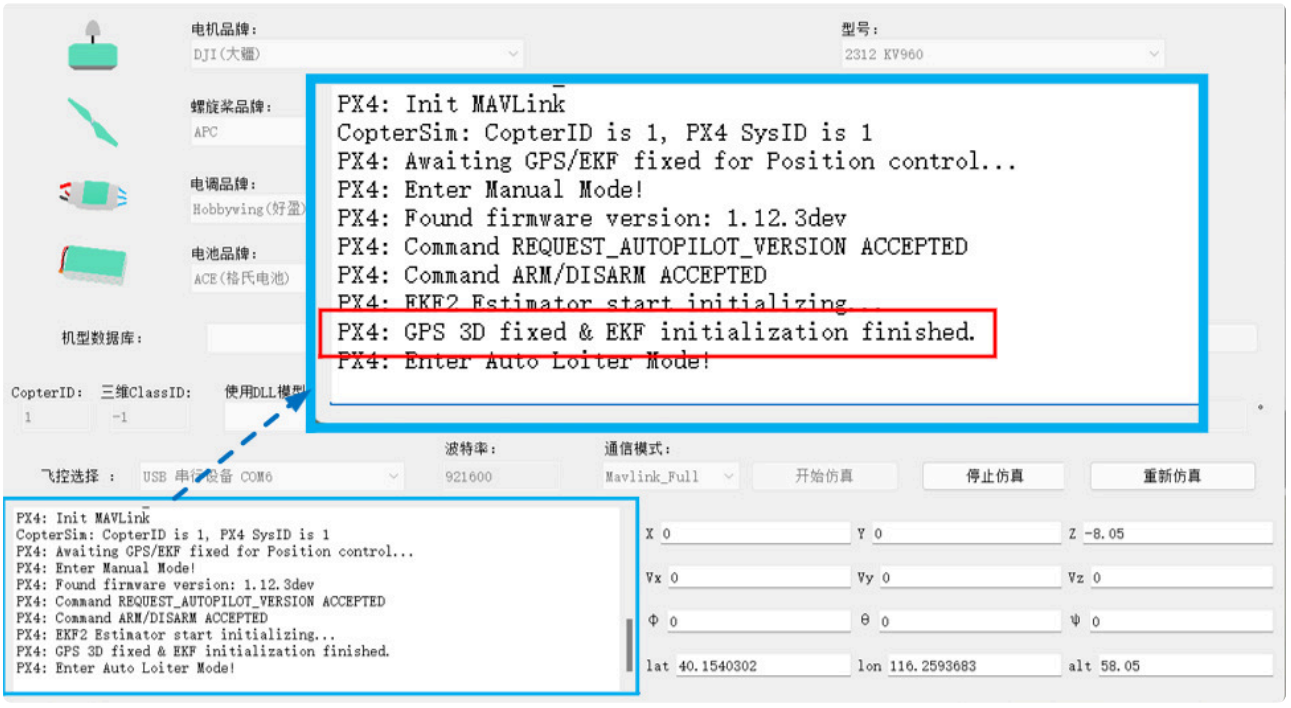
AddCopterArrayTest.py：添加无人机阵列测试脚本，演示如何向网络中添加无人机节点
Python38Run.bat：RflySim Python运行脚本，提供Python3.8环境及相关库支持
SITLRun4MavlinkFull.bat：启动4机SITL仿真环境脚本，启动QGC地面站及4个CopterSim和RflySim3D软件

UAV1Ctrl.py：1号无人机控制程序，作为主控无人机程序，负责协调其他无人机
UAV2Ctrl.py：2号无人机控制程序，接收其他无人机数据并追踪1号飞机
UAV3Ctrl.py：3号无人机控制程序，接收其他无人机数据并追踪2号飞机
UAV4Ctrl.py：4号无人机控制程序，接收其他无人机数据并追踪3号飞机，同时控制UE4视角
UavPythonRunALL.bat：批量启动所有无人机控制程序脚本，一键启动所有无人机控制脚本

4. 实验内容或步骤

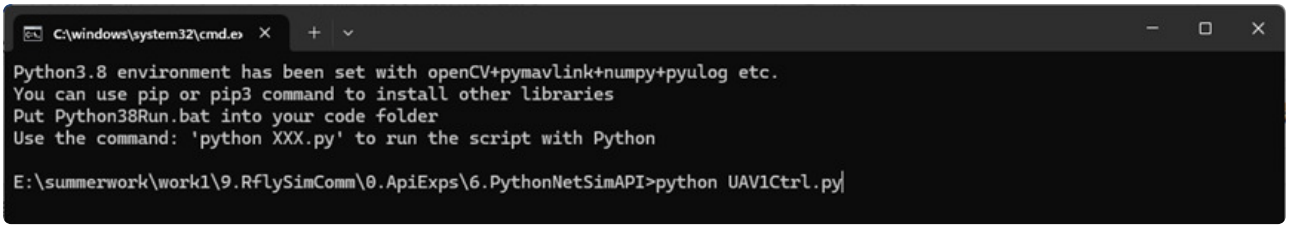
4.1 仿真初始化

双击运行SITLRun4MavlinkFull.bat文件，等待仿真环境初始化完成。脚本将会启动 1 个 QGC 地面站， 4 个 CopterSim 、1 个 RflySim3D 软件， 等待CopterSim软件下侧日志栏必须打印出 `GPS 3D fixed & EKF initialization finished` 字样代表初始化完成。如下图所示：

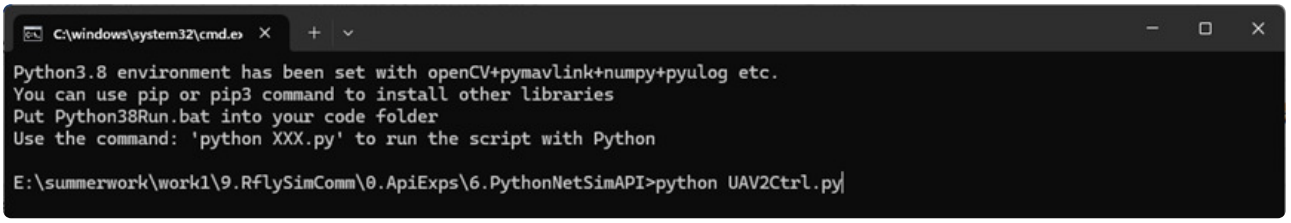


4.2 运行控制程序

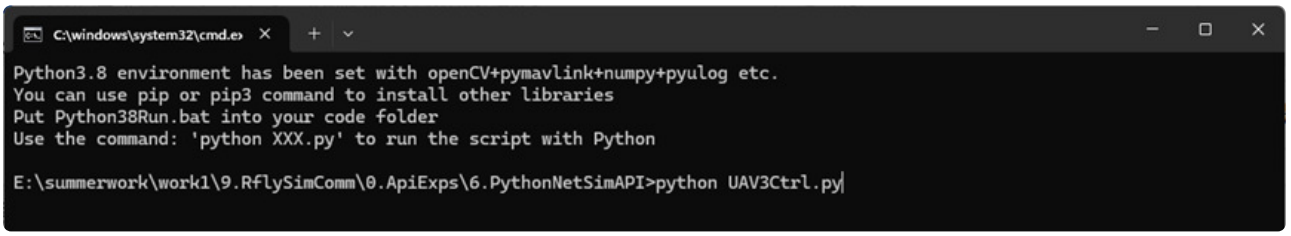
双击运行Python38Run.bat，在其中输入"python UAV1Ctrl.py"。



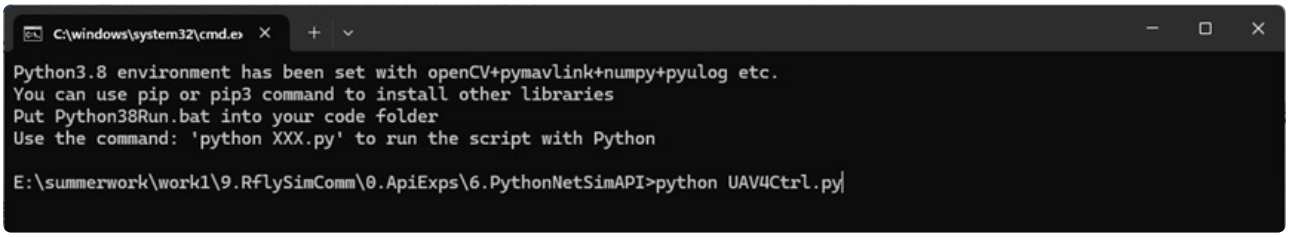
双击运行Python38Run.bat，在其中输入"python UAV2Ctrl.py"。



双击运行Python38Run.bat，在其中输入"python UAV3Ctrl.py"。



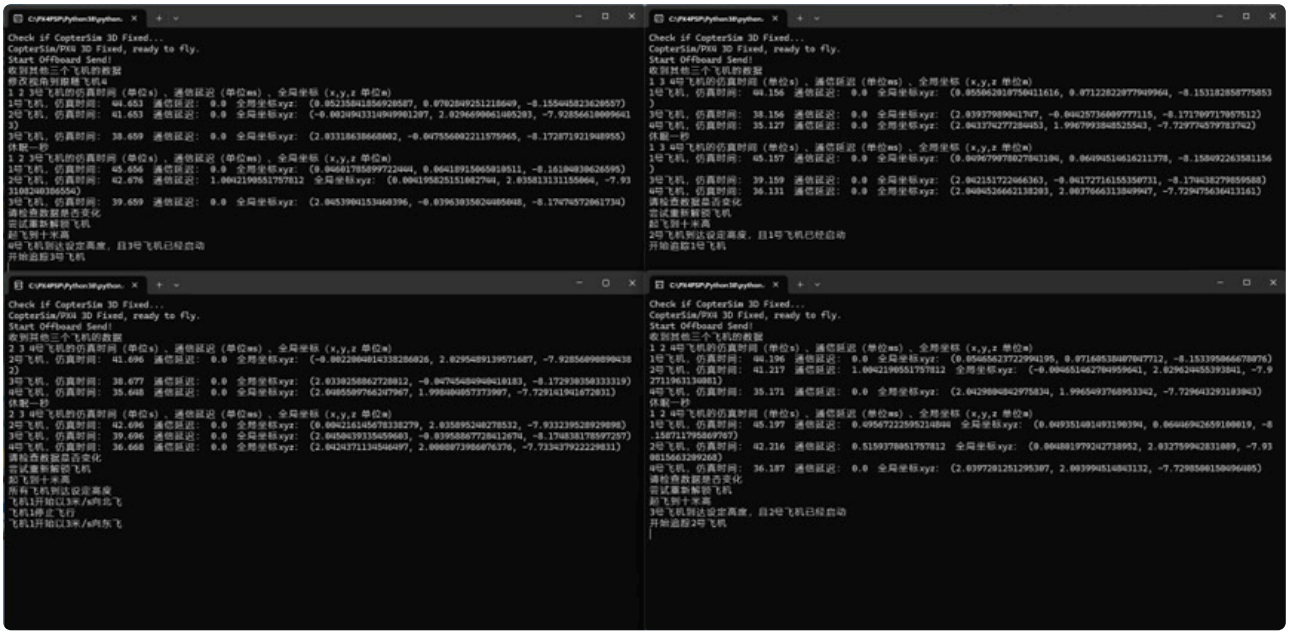
双击运行Python38Run.bat，在其中输入"python UAV4Ctrl.py"。



上述Step2-step5，可以通过一键启动脚本`UavPythonRunALL.bat`进行运行。

4.3 运行日志查看

观察命令行窗口，可以看到飞机的信息被打印。



5. 关键知识点

本实验的整体思路是构建一个多无人机系统间的网络仿真框架，通过NetSimAPI实现无人机间的数据交换与协同控制。整个系统采用分布式控制模式，每架无人机都有独立的控制程序，通过UDP网络协议进行数据通信。

AddCopterArrayTest.py 解析：

该文件是一个简单的测试脚本，用于演示如何向网络中添加无人机节点，主要包括：

```
import time
import math
import sys

import PX4MavCtrlV4
import NetSimAPIV4

#Create a new MAVLink communication instance, UDP sending port (CopterSim's
receving port) is 20100
# 创建#1号飞机的通信实例，和CopterSim#1号相连
mav = PX4MavCtrlV4.PX4MavCtrler(1)
net = NetSimAPIV4.NetSimAPI(mav)

net.netAddUavSendList([2,100,900,901])
net.netAddUavSendList([6,107,909,1000])
print(net.netSendIDList)
print(net.netSendStarList)
print(net.netSendMaskList)
```

这段代码演示了如何创建MAVLink通信实例，并向网络中添加无人机节点。通过netAddUavSendList方法可以添加无人机的ID列表、星型拓扑列表和掩码列表。

UAV控制程序解析：

1. UAV1Ctrl.py：主控无人机程序，负责协调其他无人机。主要功能包括：
    - 初始化MAVLink通信实例，连接到指定编号的无人机
    - 配置网络转发，将本机数据转发到其他无人机端口
    - 监听其他无人机数据并解析
    - 执行基本的飞行控制任务，如起飞、移动等

```
# 将本机数据转发给'224.0.0.10' 下的 60002,60003,60004系列端口
# 这几个端口目前会被#2 #3 #4号飞机订阅，因此实际上是发给了#2 #3 #4号飞机
# 如果是组网仿真的话，先要发给组网程序的指定IP和端口，再转发到各飞机监听端口
net.enNetForward([60002,60003,60004],'224.0.0.10')

# 开始监听所有发给60001端口（目前协议里面对应#1号飞机）的数据
net.StartNetRec(60001,'224.0.0.10')
```

2. UAV2Ctrl.py/ UAV3Ctrl.py/UAV4Ctrl.py：追踪无人机程序，实现对其他无人机的追踪功能。主要功能包括：
    - 初始化通信实例并设置网络转发

- 接收并解析其他无人机的状态数据
- 实现追踪算法，保持与其他无人机的相对位置关系

```python
print("开始追踪1号飞机")
while True:
    targPos=uav1.uavGlobalPos
    Curpos = mav.uavGlobalPos

    # 比例控制，增益为1
    dVx = 1*(targPos[0] - 1 - Curpos[0]) # 始终在1号飞机后1米位置
    dVy = 1*(targPos[1] - Curpos[1])
    dVz = 1*(targPos[2] - Curpos[2])

    if abs(dVx)<0.2:
        dVx=0

    if abs(dVy)<0.2:
        dVy=0

    if abs(dVz)<0.2:
        dVz=0


    mav.SendVelNED(dVx, dVy, dVz, 0)

    # 睡眠一下，使得更新率接近30Hz
    time.sleep(0.03)
```

这些程序共同构成了一个完整的多无人机协同控制系统，实现了数据的接收、解析和协同控制功能。

6.参考资料

1. RflySim官方文档： https://rflysim.com/doc/zh/
2. PX4 官方文档： https://docs.px4.io/
3. MAVLink 协议规范： https://mavlink.io/en/

7.常见问题

Q1：运行程序时出现"无法找到NetSimAPIV4模块"错误?

A1：确保将必要的API文件（NetSimAPIV4.py、PX4MavCtrlV4.py等）文件放在与UAV控制程序相同的目录下，或将其放置在Python解释器能够找到的路径中。如果是在分布式环境中运行，需要确保目标机器上也存在这些API文件。

Q2：无人机之间无法接收到彼此的数据?

A2：首先确认SITL仿真环境已正确启动，各CopterSim实例正常运行。检查网络连接是否正常，确保UDP端口和多播地址设置正确。另外，检查网络转发设置是否正确（enNetForward），确保数据发送端口与接收端口匹配。

Q3：在Linux/Ubuntu环境下运行出现问题?

A3：确保Python环境配置正确，相关依赖库已安装。在Linux环境下可能需要调整某些路径设置，同时确保NetSimAPIV4.py和其他必要的API文件都已复制到Linux系统中。检查权限设置，确保脚本具有执行权限。

Q4：无人机追踪效果不稳定或振荡严重?

A4：这通常是由于控制参数设置不当造成的。可以尝试调整比例控制器的增益参数，减小增益值以降低系统响应速度，从而减少振荡。同时，检查时间步长设置是否合适，确保控制频率稳定。

---

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