

# 多无人机广播通信实验

## 1. 实验目的

本实验旨在演示如何通过RflySim平台实现多无人机之间的广播通信。通过使用NetSimAPIV4库和PX4MavCtrlV4库，实现多架无人机的状态数据广播与接收。学习者将掌握：

- 如何建立与多架无人机的通信连接
- 如何配置无人机数据广播转发
- 如何接收并解析来自多架无人机的状态数据
- 分布式仿真环境下多无人机协同工作的基本原理
- 广播通信机制在多无人机系统中的应用

## 2. 实验要求

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

## 3. 实验地址

例程目录：

[\[安装目录\]\RflySimAPIs\9.RflySimComm\0.ApiExps\e1.RflyNetAPIExps\2.BroadSendData](#)

[Python38Run.bat](#)：RflySim Python运行脚本

[SITLRun3MavlinkFull.bat](#)：启动3架无人机SITL仿真环境脚本

[UAV1Ctrl.py](#)：第1架无人机控制脚本

[UAV2Ctrl.py](#)：第2架无人机控制脚本

[UAV3Ctrl.py](#)：第3架无人机控制脚本

[UavPythonRunALL.bat](#)：批量启动所有无人机控制脚本

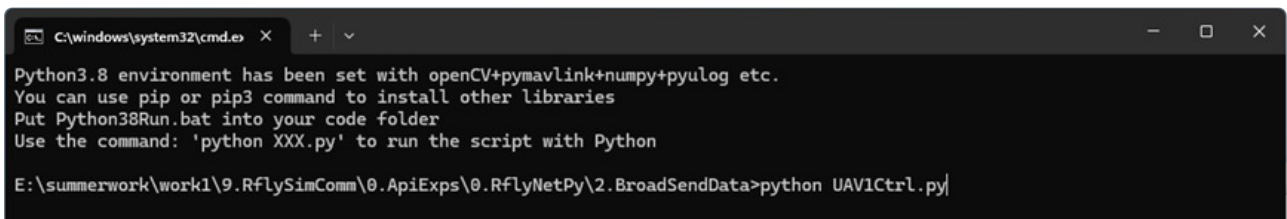
## 4. 实验内容或步骤

### 4.1 步骤1：本机实验

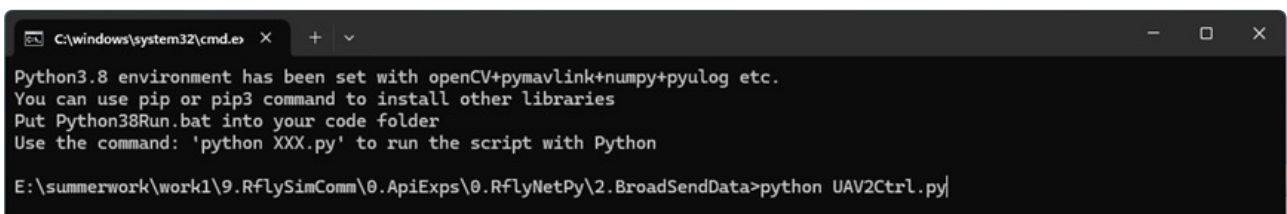
双击运行SITLRun.bat文件，等待仿真环境初始化完成。脚本将会启动 1 个 QGC 地面站，1 个 CopterSim、1 个 RflySim3D 软件，等待CopterSim软件下侧日志栏必须打印出 GPS 3D fixed & EKF initialization finished 字样代表初始化完成。如下图所示：



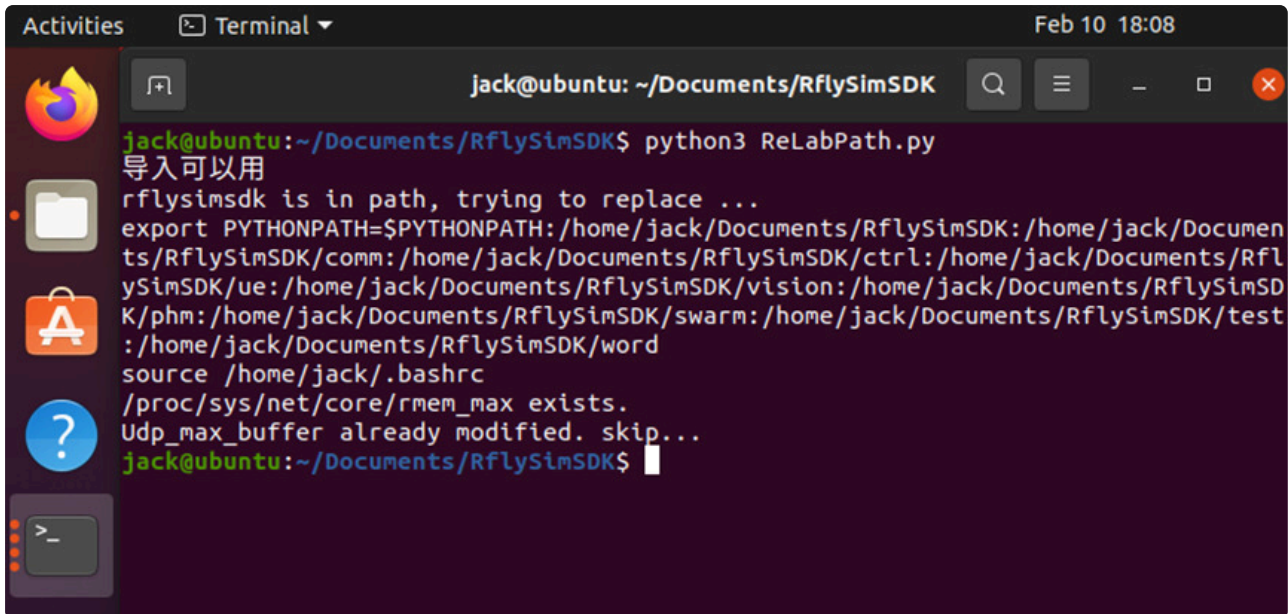
双击运行"Python38Run.bat", 在其中输入"python UAV1Ctrl.py", 启动1号飞机广播收发程序。



双击运行"Python38Run.bat", 在其中输入"python UAV2Ctrl.py", 启动2号飞机广播收发程序。

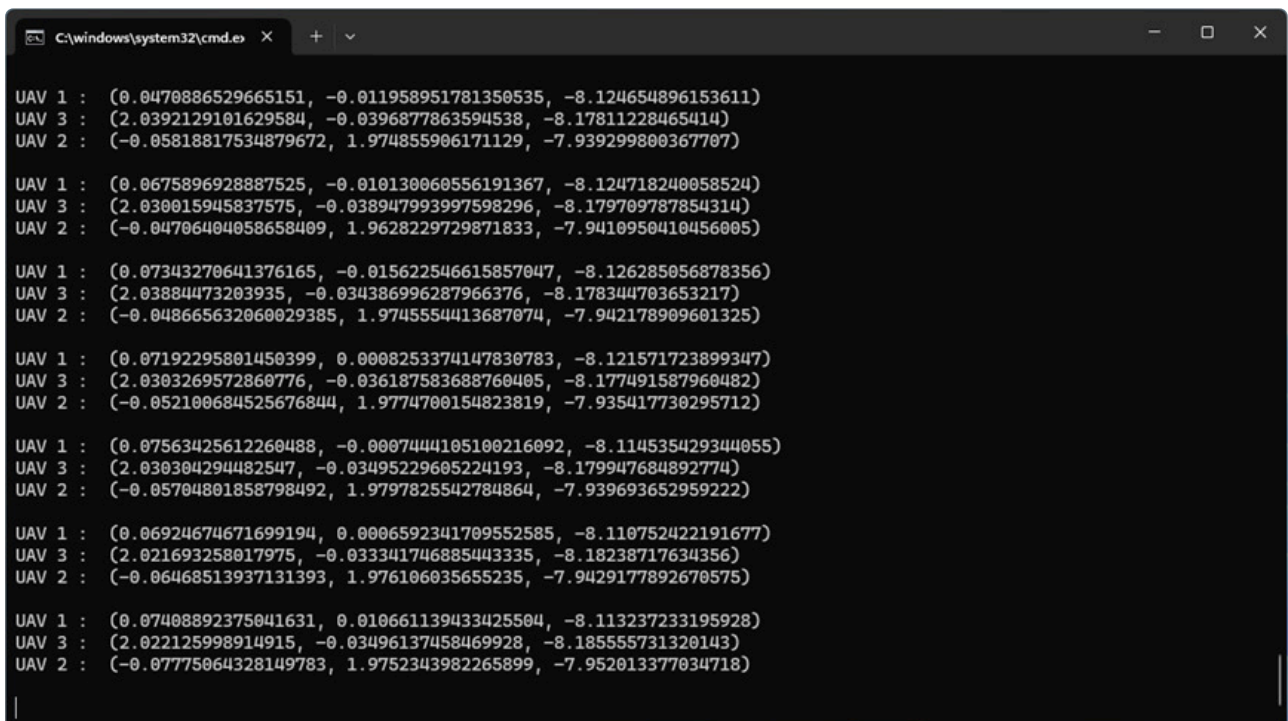






```
jack@ubuntu: ~/Documents/RflySimSDK
jack@ubuntu:~/Documents/RflySimSDK$ python3 ReLabPath.py
导入可以用
rflysimSDK is in path, trying to replace ...
export PYTHONPATH=$PYTHONPATH:/home/jack/Documents/RflySimSDK:/home/jack/Documents/RflySimSDK/comm:/home/jack/Documents/RflySimSDK/ctrl:/home/jack/Documents/RflySimSDK/ue:/home/jack/Documents/RflySimSDK/vision:/home/jack/Documents/RflySimSDK/phm:/home/jack/Documents/RflySimSDK/swarm:/home/jack/Documents/RflySimSDK/test:/home/jack/Documents/RflySimSDK/word
source /home/jack/.bashrc
/proc/sys/net/core/rmem_max exists.
Udp_buffer already modified. skip...
jack@ubuntu:~/Documents/RflySimSDK$
```

在Windows主机中，双击打开 [SITLRun3MavlinkFull.bat](#) 自动创建三个飞机。等待所有 CopterSim的左下角打印"PX4: GPS 3D fixed & EKF initialization finished."。再运行 [UAV1Ctrl.py](#)，启动1号飞机广播收发程序。



```
C:\windows\system32\cmd.exe
UAV 1 : (0.0470886529665151, -0.011958951781350535, -8.124654896153611)
UAV 3 : (2.0392129101629584, -0.0396877863594538, -8.17811228465414)
UAV 2 : (-0.05818817534879672, 1.974855906171129, -7.939299800367707)

UAV 1 : (0.0675896928887525, -0.010130060556191367, -8.124718240058524)
UAV 3 : (2.030015945837575, -0.038947993997598296, -8.179709787854314)
UAV 2 : (-0.04706404058658409, 1.9628229729871833, -7.9410950410456005)

UAV 1 : (0.07343270641376165, -0.015622546615857047, -8.126285056878356)
UAV 3 : (2.03884473203935, -0.034386996287966376, -8.178344703653217)
UAV 2 : (-0.048665632060029385, 1.9745554413687074, -7.942178909601325)

UAV 1 : (0.07192295801450399, 0.0008253374147830783, -8.121571723899347)
UAV 3 : (2.0303269572860776, -0.036187583688760405, -8.177491587960482)
UAV 2 : (-0.052100684525676844, 1.9774700154823819, -7.935417730295712)

UAV 1 : (0.07563425612260488, -0.0007444105100216092, -8.114535429344055)
UAV 3 : (2.030304294482547, -0.03495229605224193, -8.179947684892774)
UAV 2 : (-0.05704801858798492, 1.9797825542784864, -7.939693652959222)

UAV 1 : (0.06924674671699194, 0.0006592341709552585, -8.110752422191677)
UAV 3 : (2.021693258017975, -0.033341746885443335, -8.18238717634356)
UAV 2 : (-0.06468513937131393, 1.976106035655235, -7.9429177892670575)

UAV 1 : (0.07408892375041631, 0.010661139433425504, -8.113237233195928)
UAV 3 : (2.022125998914915, -0.03496137458469928, -8.185555731320143)
UAV 2 : (-0.07775064328149783, 1.9752343982265899, -7.952013377034718)
```

在一台Ubuntu虚拟机中，运行 [UAV2Ctrl.py](#)，启动2号飞机广播收发程序。

```
rflysim@rflysim: ~
UAV 2 : (-0.055876731438794014, 1.9844536545727038, -7.935905411773974)
UAV 1 : (0.0568349064569631, -0.0013265038829626263, -8.128576786086438)
UAV 3 : (2.01665941666134, -0.03470891538646281, -8.18402625564611)

UAV 2 : (-0.0624692919983052, 1.9885780766646945, -7.938765181163007)
UAV 1 : (0.05314677220393804, 0.003789331113141614, -8.119022002039952)
UAV 3 : (2.0346000729433475, -0.03476591232802928, -8.18162178937471)

UAV 2 : (-0.060278305350103256, 1.9850311439524972, -7.935527858724499)
UAV 1 : (0.05712261807729391, 0.0012137548077522098, -8.11986917961995)
UAV 3 : (2.034693726741451, -0.029103356521690493, -8.179467014410854)

UAV 2 : (-0.05029769896925318, 1.9857304880436146, -7.934803069037998)
UAV 1 : (0.07236558210660604, 0.007687676047128278, -8.11899008189003)
UAV 3 : (2.0425442497317894, -0.034253396201992636, -8.177881005593896)

UAV 2 : (-0.059444620771565315, 1.9687509538646424, -7.935812937961573)
UAV 1 : (0.07857956749011663, 0.0027215194899259387, -8.117110291790691)
UAV 3 : (2.0304759176752745, -0.0368795638122803, -8.17813993189609)

UAV 2 : (-0.05491743286535966, 1.9677910452749217, -7.9340294007488135)
UAV 1 : (0.0715836064200559, 0.005663478096169072, -8.115211366039244)
UAV 3 : (2.0281588783792435, -0.03903934184100688, -8.179989240506053)

UAV 2 : (-0.06128867299691665, 1.9778028525918359, -7.929968145144785)
UAV 1 : (0.06055734318305639, -0.00866434788222925, -8.117257868467373)
UAV 3 : (2.038827674866396, -0.03703745651629031, -8.180709599891067)
```

在另一台Ubuntu虚拟机中，运行 `UAV3Ctrl.py`，启动3号飞机广播收发程序。

```
nvidia@nvidia-virtual-machine: ~  
UAV 1 : (0.0635977016072431, -0.012132415267485541, -8.113935970064727)  
UAV 3 : (2.009769658029753, -0.03722892153647006, -8.19092501923567)  
UAV 2 : (-0.06934135736361968, 1.9807782268192256, -7.940993914315158)  
  
UAV 1 : (0.05898279708434728, -0.018740595969218177, -8.114335123278675)  
UAV 3 : (2.0230048060721932, -0.03498077589657367, -8.191174135917187)  
UAV 2 : (-0.06114450798765647, 1.973174317231738, -7.941539422543401)  
  
UAV 1 : (0.05712327372838644, -0.010547138470071715, -8.11318154895353)  
UAV 3 : (2.0147440288520633, -0.041817735988760596, -8.18419626671946)  
UAV 2 : (-0.0599411209432863, 1.970217626732179, -7.941448881225342)  
  
UAV 1 : (0.06595690560151724, -0.010904059464889926, -8.116572657429156)  
UAV 3 : (2.0228185080296575, -0.04852891165401996, -8.181969139167428)  
UAV 2 : (-0.06756107093230712, 1.9690297890495145, -7.942685353504175)  
  
UAV 1 : (0.05603883427192358, -0.0175891674112858, -8.116375231944499)  
UAV 3 : (2.0272294044799386, -0.048291003439689284, -8.184511724301935)  
UAV 2 : (-0.06707169630960452, 1.9712267026345813, -7.945607010209614)  
  
UAV 1 : (0.058163367329136406, 0.0023502272564588367, -8.11325249524462)  
UAV 3 : (2.036866703473705, -0.040278656516337996, -8.18979130163914)  
UAV 2 : (-0.06614709974662292, 1.9786443055647993, -7.942898639412278)  
  
UAV 1 : (0.06330513965894369, 0.003318212275546628, -8.108215467833457)  
UAV 3 : (2.0373584278232633, -0.02846566877898038, -8.190555876494706)  
UAV 2 : (-0.06164207079843509, 1.9728023437358075, -7.942980532468909)
```

能看到和本地测试一样的效果，说明测试正确。

## 5. 关键知识点

### 关键知识点1：整体思路和框架

本实验采用RflySim平台的NetSimAPIV4库实现多无人机广播通信。整体架构包括：

1. 通过ReqCopterSim模块动态获取CopterSim仿真器的IP地址
2. 使用PX4MavCtrlV4建立与特定无人机的MAVLink通信
3. 利用NetSimAPIV4启用广播转发并接收多无人机数据
4. 循环读取并显示所有无人机的状态信息

## 关键知识点2：Python程序解析

### 无人机控制脚本（UAV1Ctrl.py, UAV2Ctrl.py, UAV3Ctrl.py）

```
1 import time
2 import math
3 import sys
4 import ReqCopterSim
5 import PX4MavCtrlV4 as PX4MavCtrl
6 import NetSimAPIV4
7
8 CopterID=1 # 定义无人机ID, UAV1Ctrl.py为1, UAV2Ctrl.py为2, UAV3Ctrl.py为3
```

以上是脚本的导入部分，引入了时间、数学、系统库以及RflySim专用模块。CopterID变量标识当前控制的是哪架无人机。

```
1 # 创建一个CopterSim状态获取实例，并监听2s钟，获取局域网内所有CopterSim所在电脑的IP
2 req = ReqCopterSim.ReqCopterSim()
3
4 # 获取目标电脑IP，并且配置CopterSim回传数据到本电脑
5 TargetIP = req.getSimIpID(CopterID)
6
7 # 请求目标CopterSim将数据返回到本电脑
8 req.sendReSimIP(CopterID)
```

这部分代码实现了动态IP获取功能。ReqCopterSim.ReqCopterSim()创建了一个实例来检测局域网内的CopterSim实例，getSimIpID()方法获取指定无人机所在的计算机IP，sendReSimIP()方法配置该无人机将数据回传至本机。

```
1 # 创建通信实例
2 mav = PX4MavCtrl.PX4MavCtrler(CopterID,TargetIP)
3 net = NetSimAPIV4.NetSimAPI(mav)
4
5 # 初始化MAVLink通信
6 mav.InitMavLoop() # 连接飞控/CopterSim并准备数据交换
```

这里创建了MAVLink通信实例，PX4MavCtrler负责与指定无人机建立连接，NetSimAPI封装了更高层的网络通信功能。InitMavLoop()方法初始化MAVLink通信循环。

```

1 | time.sleep(2)
2 |
3 | print('Broadcast uav data')
4 | net.enUavForward() # 启用无人机数据转发功能
5 |
6 | time.sleep(2)
7 | print('Start receiving all uav data')
8 | net.StartNetRec() # 开始接收网络中的无人机数据
9 |
10 | while True:
11 |     for uav in net.UavData:
12 |         print('UAV',uav.CopterID,': ',uav.uavGlobalPos)
13 |         if len(net.UavData)>0:
14 |             print('') # 空一行，便于观察数据
15 |             time.sleep(1)

```

这是核心的数据广播和接收循环。enUavForward()启用数据转发功能，使本机可以接收其他无人机的数据；StartNetRec()开始接收网络中的无人机数据；循环体遍历net.UavData集合，打印每架无人机的ID和全局位置信息。

## 关键知识点3：技术要点

- 分布式仿真支持：通过ReqCopterSim模块动态获取IP，支持跨计算机的仿真环境
- 广播通信机制：启用数据转发后，各节点可以接收网络中其他无人机的数据
- 状态监控：实时显示所有无人机的位置信息，便于监控集群状态

## 6.参考资料

1. [RflySim官方文档](#)
2. [MAVLink协议详解](#)
3. [PX4开源飞控系统](#)

## 7. 常见问题

### Q1: 运行Python脚本时报错"ModuleNotFoundError: No module named 'ReqCopterSim'"。

A1: 这表示RflySim SDK库没有正确安装或未添加到Python环境中。请确认已安装RflySim工具链，并将RflySimSDK库路径添加到Python环境变量中。

### Q2: 运行脚本后无法看到其他无人机的数据，只显示自己的数据。

A2: 可能是数据转发功能未正常启用或网络配置问题。请检查是否正确调用了net.enUavForward()方法，并确认所有计算机处于同一局域网中，防火墙设置允许相应端口通信。

### Q3: 多台计算机联合实验时，无人机状态数据更新延迟较大。

A3: 这通常是由于网络延迟或带宽不足造成的。请确保局域网连接稳定，尝试减少同时运行的无人机数量，并检查防火墙和路由器设置是否限制了UDP广播包的传输。

---

1. <https://rflysim.com/> ↩

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