

多机组播通信状态协同监测实验

1. 实验目的

本实验旨在通过RflySim平台搭建多无人机协同仿真环境，实现多架无人机的状态数据共享与实时监测。通过本实验，学习者可以掌握基于网络通信的多无人机协同控制技术，理解组播通信机制在无人机集群中的应用，熟悉分布式仿真系统的构建与调试方法。实验重点在于实现无人机间的状态数据交换、组播通信的建立与维护，以及多无人机系统的协同工作原理。

2. 实验要求

- 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\7.PythonAllUavDemo](#)

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

[SITLRun4MavlinkFull.bat](#)：启动4架无人机SITL仿真环境脚本

[UAV1Ctrl.py](#)：控制第1架无人机的Python脚本，包含无人机状态获取和组播通信功能

[UAV2Ctrl.py](#)：控制第2架无人机的Python脚本，包含无人机状态获取和组播通信功能

[UAV3Ctrl.py](#)：控制第3架无人机的Python脚本，包含无人机状态获取和组播通信功能

[UAV4Ctrl.py](#)：控制第4架无人机的Python脚本，包含无人机状态获取和组播通信功能

[UavPythonRunALL.bat](#)：一键启动所有无人机控制脚本的批处理文件

[Readme.md](#)：项目说明文档

[Readme_en.md](#)：英文版项目说明文档

[assets/bk/index.html](#)：备份资源文件

4. 实验内容或步骤

4.1 本机实验

4.1.1 仿真环境初始化

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，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：



4.1.2 运行控制程序

双击运行 "`UavPythonRunALL.bat`"，一键启动所有飞机控制程序。

```
C:\Python38\python.exe
Current list: [1, 2, 3]
Current list: [1, 2, 3, 4]
All uav data received.
Point Ref for Copter#2: 2 (-4.4955857447348535e-05, 6.40355356154032e-05, -0.002260153880342841) (0.0016098285830393434, 0.0031892755068838596, 8.84996734384913e-06) (-4.4955857447348535e-05, 6.40355356154032e-05, -0.002260153880342841) (0.028511105105280876, -0.03107389621436596, -0.003916979767382145) (0.06182294767882324, 1.9626633661734763, -7.819916668184577)
Dict search for Copter#3: 3 (0.00016246897575911134, 4.1558621887816116e-05, -0.0014033479383215308) (0.0053089819848537445, 0.0032647093757987022, 0.0070496308617293835) (0.00016246897575911134, 4.1558621887816116e-05, -0.0014033479383215308) (0.060711875557899475, 0.03498402610421181, -0.0023941462859511375) (0.0, 0.0, 0.0)
Point Ref for Copter#2: 2 (0.0001652172504691407, -6.860690336907282e-05, -0.00248155253933058) (0.0011133475927636027, -0.0011178120039403439, -0.0007149921148084104) (0.0001652172504691407, -6.860690336907282e-05, -0.00248155253933058) (0.03236064687371254, -0.036078162491321564, -0.008778754621744156) (0.0656724894472549, 1.957659098965206, -7.824778443038939)
Dict search for Copter#3: 3 (-5.008259540772997e-05, -0.00014205412298906595, -0.002264508744701743) (0.0005990957724861801, -0.006618101615458727, -0.0020672788377851248) (-5.008259540772997e-05, -0.00014205412298906595, -0.002264508744701743) (0.048337046056985855, 0.02958933636546135, -0.013817772269248962) (2.035928144277995, 0.04662982120664094, -8.150817460962626)
Point Ref for Copter#2: 2 (-0.00012586994853336364, 7.385794742731377e-05, -0.002108974615111947) (0.0016596588538959627, -0.004372101277112961, -0.002884883666411042) (-0.00012586994853336364, 7.385794742731377e-05, -0.002108974615111947) (0.0348447784781456, -0.030020808801054955, -0.008294710889458656) (0.06815662105168796, 1.9637164535867873, -7.824294399306654)
Dict search for Copter#3: 3 (1.1769193406507839e-05, -7.016659219516441e-05, -0.002109116641804576) (0.005074663553386927, 0.004853731952607632, -0.0032455776818096638) (1.1769193406507839e-05, -7.016659219516441e-05, -0.002109116641804576) (0.050840944051742554, 0.04654892161488533, -0.00843723677098751) (2.0384320422727518, 0.06358940645606492, -8.145436925464365)
```

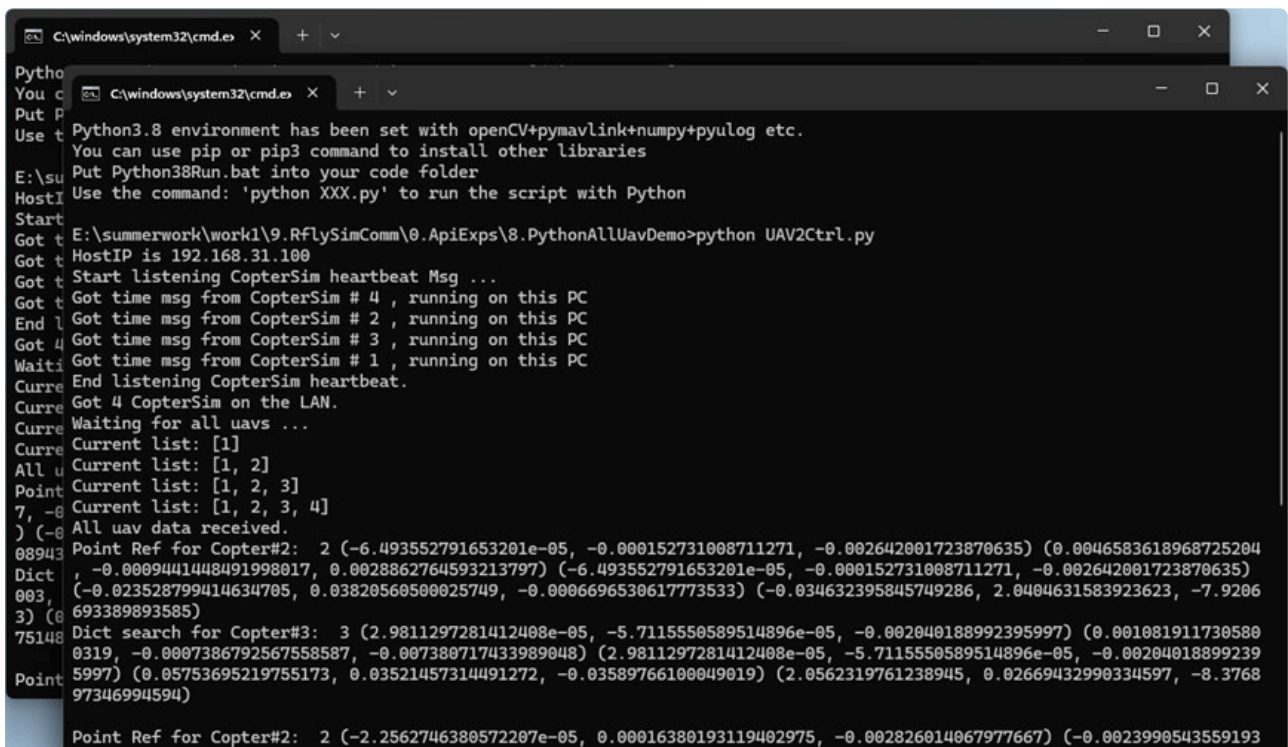
4.2 联机实验

- 1) 准备4台任意系统主机（可以是NX主机、Ubuntu虚拟机、Windows电脑）。确保两台机器是在同一个局域网内。
- 2) 按照[RflySim按安装理解]\RflySimAPIs\1.RflySimIntro\2.AdvExps\e3.PythonConfig实验将[RflySim安装路径]\RflySimAPIs\RflySimSDK\RflySimSDK库复制到每台主机上，并确保RflySimSDK已经导入当前主机的Python环境中。

在主机电脑上，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，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：



在Windows主机中，运行 [UAV1Ctrl.py](#) 和 [UAV2Ctrl.py](#)。



在两台虚拟机中，分别运行 [UAV3Ctrl.py](#) 和 [UAV4Ctrl.py](#)。

```
活动 终端 9月2日 22:48 rflysim@rflysim: ~
rflysim@rflysim:~$ python3 UAV3Ctrl.py
HostIP is 192.168.31.83
Start listening CopterSim heartbeat Msg ...
Got time msg from CopterSim # 4 , not on this PC
Got time msg from CopterSim # 2 , not on this PC
Got time msg from CopterSim # 3 , not on this PC
Got time msg from CopterSim # 1 , not on this PC
End listening CopterSim heartbeat.
Got 4 CopterSim on the LAN.
Waiting for all uavs ...
Current list: [1]
Current list: [1, 2]
Current list: [1, 2, 3]
Current list: [1, 2, 3, 4]
All uav data received.
Point Ref for Copter#2: 2 (-7.95064406702295e-05, -0.0001446126407245174, -0.002644282067194581) (0.004877585452049971, -0.000791623315308243, 0.002486615441739559) (-7.95064406702295e-05, -0.0001446126407245174, -0.002644282067194581) (-0.023803111165761948, 0.038238395005464554, -0.000576525111682713) (-0.03490670759687653, 2.0404959483975693, -7.920576211039264)
Dict search for Copter#3: 3 (2.1020887288614176e-05, -5.5715907365083694e-05, -0.0020503185223788023) (0.0014772535068914294, -0.0012235379545018077, -0.006883303634822369) (2.1020887288614176e-05, -5.5715907365083694e-05, -0.0020503185223788023) (0.05759647861123085, 0.034872911870479584, -0.03583742305636406) (2.0562915025375736, 0.026352668628912834, -8.376837109050468)

Point Ref for Copter#2: 2 (-2.7839094400405884e-05, 0.0001590502797625959, -0.0028422377072274685) (-0.0023876477498561144, 0.004177102353423834, 0.0067320591770112514) (-2.7839094400405884e-05, 0.0001590502797625959, -0.002842237707227
```

```
活动 终端 9月3日 13:48 zh nvidia@nvidia-virtual-machine: ~
nvidia@nvidia-virtual-machine:~$ python3 UAV4Ctrl.py
No Redis labs
HostIP is 192.168.31.235
Start listening CopterSim heartbeat Msg ...
Got time msg from CopterSim # 2 , not on this PC
Got time msg from CopterSim # 3 , not on this PC
Got time msg from CopterSim # 1 , not on this PC
Got time msg from CopterSim # 4 , not on this PC
End listening CopterSim heartbeat.
Got 4 CopterSim on the LAN.
Waiting for all uavs ...
Current list: [1]
Current list: [1, 2]
Current list: [1, 2, 3]
Current list: [1, 2, 3, 4]
All uav data received.
Point Ref for Copter#2: 2 (-6.493552791653201e-05, -0.000152731008711271, -0.002642001723870635) (0.0046583618968725204, -0.0009441448491998017, 0.0028862764593213797) (-6.493552791653201e-05, -0.000152731008711271, -0.002642001723870635) (-0.023528799414634705, 0.03820560500025749, -0.0006696530617773533) (-0.034632395845749286, 2.0404631583923623, -7.9206693389893585)
Dict search for Copter#3: 3 (2.9811297281412408e-05, -5.7115550589514896e-05, -0.002040188992395997) (0.0010819117305800319, -0.0007386792567558587, -0.007380717433989048) (2.9811297281412408e-05, -5.7115550589514896e-05, -0.002040188992395997) (0.05753695219755173, 0.03521457314491272, -0.03589766100049019) (2.0562319761238945, 0.02669432990334597, -8.376897346994594)

Point Ref for Copter#2: 2 (-2.2562746380572207e-05, 0.00016380193119402975, -0.002826014067977667) (-0.002399054355919361, 0.0038158234674483538, 0.007549607
```

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

实验整体思路和框架

本实验的核心是实现多架无人机之间的状态数据共享与协同监测。通过组播通信机制，各无人机控制节点可以接收来自其他无人机的状态信息，形成一个多节点协同的仿真系统。实验采用主从架构，其中一台计算机作为主控节点运行仿真环境，多台计算机作为从属节点运行控制脚本，通过网络通信实现数据交换和协同控制。

Python程序解析

UAV1Ctrl.py / UAV2Ctrl.py / UAV3Ctrl.py / UAV4Ctrl.py 程序解析

这些脚本文件是控制各无人机的核心程序，它们具有相同的结构，只是通过不同的CopterID进行区分。

初始化部分：

```
1 import time
2 import math
3 import sys
4 import ReqCopterSim
5 import PX4MavCtrlV4 as PX4MavCtrl
6 import NetSimAPIV4
7
8 CopterID=1 # 在UAV2Ctrl.py中是2，在UAV3Ctrl.py中是3，在UAV4Ctrl.py中是4
```

这部分代码导入必要的模块并设置无人机ID，用于标识当前脚本控制的无人机编号。

IP请求和配置部分：

```
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 # 获取到指定CopterID的CopterSim所在电脑的IP
6 # 注：Windows下运行本函数获取TargetIP=127.0.0.1，如果在其他电脑上运行，获取CopterID对应电脑的
7 IP
8 TargetIP = req.getSimIpID(CopterID)
9
10 # 请求目标CopterSim将数据返回到本电脑
11 # 通过本接口，可以不用再去bat脚本里面填写IP地址了
    req.sendReSimIP(CopterID)
```

这一部分用于获取指定无人机所在的计算机IP地址，并配置该无人机将数据回传到当前计算机。这使得系统可以在分布式仿真环境中正常工作，无需手动配置IP地址。

通信实例创建部分：

```
1 # 创建#CopterID号飞机的通信实例，和CopterSim#CopterID号相连，使用TargetIP确保分布式仿真也能
2 用
3 mav = PX4MavCtrl.PX4MavCtrl(CopterID, TargetIP)
4 net = NetSimAPIV4.NetSimAPI(mav)
5
6 # 这里使用MAVLink_Full模式来传输数据
    mav.InitMavLoop()
```

创建MAVLink通信实例和网络API实例，初始化与指定无人机的通信连接。

组播通信配置部分：

```
1 # 将本机数据转发给组播IP和端口：'224.0.0.10'，60000，所有飞机都可以订阅
2 net.enNetForward()
3
4 # 开始监听所有发给60000端口的数据，所有飞机数据，会存放到net.UavData列表中
5 net.StartNetRec()
```

启用网络数据转发功能，将本机的无人机数据发送到组播地址'224.0.0.10'的60000端口，同时开始监听该端口接收其他节点发送的数据。

数据同步等待部分：

```
1 # 请求的飞机列表，等到列表中飞机数据都收到后，再开始运行后续程序
2 reqUavList=[1,2,3,4]
3
4 curList=[]
5 print('Waiting for all uavs ...')
6
7 # 下面的函数检查是否收到指定的无人机数据，这里以1 2 3 4号飞机为例
8 while True:
9     isAllRec=True
10
11     for ID in reqUavList:
12         if not net.UavDict.__contains__(str(ID)):
13             isAllRec=False # 只要有一个飞机数据没收到，就继续等待
14             break
15         else:
16             if ID not in curList:
17                 curList=curList + [ID]
18                 curList.sort()
19                 print('Current list:',curList)
20     if isAllRec: # 所有飞机数据收到后，往下跳出等待循环
21         print('All uav data received.')
22         break
23
24     time.sleep(0.5)
```

等待直到所有指定的无人机数据都接收到，确保协同工作的基础条件满足。

数据访问和显示部分:

```
1 # 分别获取1/2/3/4号飞机的数据指针
2 uav1 = net.UavDict['1']
3 uav2 = net.UavDict['2']
4 uav3 = net.UavDict['3']
5 uav4 = net.UavDict['4']
6
7
8 # 每隔1s打印指定飞机的数据
9 while True:
10     # 直接打印指针
11     print('Point Ref for Copter#2:
12 ',uav2.CopterID,uav2.uavAngEular,uav2.uavVeLNED,uav2.uavAngEular,uav2.uavPosNED,uav
13 2.uavGlobalPos)
14
15     # 直接查询字典
16     print('Dict search for Copter#3:
17 ',net.UavDict['3'].CopterID,net.UavDict['3'].uavAngEular,net.UavDict['3'].uavVeLNED
18 ,net.UavDict['3'].uavAngEular,net.UavDict['3'].uavPosNED,net.UavDict['3'].uavGlobal
19 Pos)
20     print('')
21     time.sleep(5)
```

获取各无人机的数据指针，并持续打印指定无人机的状态信息，用于监控无人机的协同状态。

UavPythonRunALL.bat程序解析

```
` ` batch
if not defined PSP_PATH (
SET PSP_PATH=C:\PX4PSP
SET PSP_PATH_LINUX=/mnt/c/PX4PSP
)
start %PSP_PATH%\Python38\python.exe "%~dp0\UAV1Ctrl.py"
start %PSP_PATH%\Python38\python.exe "%~dp0\UAV2Ctrl.py"
start %PSP_PATH%\Python38\python.exe "%~dp0\UAV3Ctrl.py"
start %PSP_PATH%\Python38\python.exe "%~dp0\UAV4Ctrl.py"
```

```
1  这是一个批处理脚本，用于一键启动所有无人机控制脚本。它设置了Python路径并依次启动四架无人机的控制程
2  序。
3
4  ### [SITLRun4MavlinkFull.bat](SITLRun4MavlinkFull.bat)程序解析
5
6  这个脚本负责启动完整的4架无人机仿真环境，包括QGroundControl地面站、RflySim3D可视化界面和4个
7  CopterSim实例。它还负责启动PX4 SITL仿真进程，实现完整的多无人机仿真环境。
8
9  # 6.参考资料
10
11  1. RflySim官方文档：https://rflysim.com/doc/zh/
12  2. PX4飞控开发指南：https://docs.px4.io/master/zh/
13  3. MAVLink协议详解：https://mavlink.io/zh/
14
15  # 7.常见问题
16
17  ## Q1：运行[UavPythonRunALL.bat](UavPythonRunALL.bat)后，没有反应或者报错
18  A1：检查PSP_PATH环境变量是否正确设置，确认C:\PX4PSP路径下存在所需的软件包。另外，检查Python38
19  是否正确安装并配置了相关依赖库。
20
21  ## Q2：无法接收到来自其他无人机的数据
22  A2：检查网络配置是否正确，确保所有计算机在同一个局域网内。确认防火墙设置没有阻止UDP端口60000的通信。验证组播地址'224.0.0.10'是否在网络中被正确路由。

```

Q3：仿真环境启动失败，CopterSim没有显示GPS固定信息
A3：等待更长时间让系统初始化完成，通常需要几十秒到一分钟。检查是否有限制问题，尝试以管理员身份运行批处理文件。确认PX4 SITL和相关组件是否正确安装。

1. <https://rflysim.com/> ↩

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