

# 粗粒度集群组网实验

## 1. 实验目的

通过无人机集群组网发送的数据都会发送到粗粒度组网程序，然后根据粗粒度组网的规则判断能否到达目的无人机并计算丢包。

## 2. 实验要求

- 软件要求：Windows 10及以上版本；RflySim工具链<sup>[1]</sup>；Redis。
- 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\1.BasicExps\e4.RedisUAVsCommExps\2.NetSimMini\_redis\_nomat

- `./SITLRun4MavlinkFull.bat`：仿真环境一键启动脚本
- `./Python38Run.bat`：Python环境一键启动脚本
- `./UAV1Ctrl.py`：1号飞机控制程序
- `./UAV2Ctrl.py`：2号飞机控制程序
- `./UAV3Ctrl.py`：3号飞机控制程序
- `./UAV4Ctrl.py`：4号飞机控制程序
- `./NetworkSimulationNoRflysim.py`：仿真主要控制程序
- `./PyViaGui.py`：模拟效果展示程序
- `./RedisUtils.py`：Redis控制程序

## 4. 实验内容或步骤

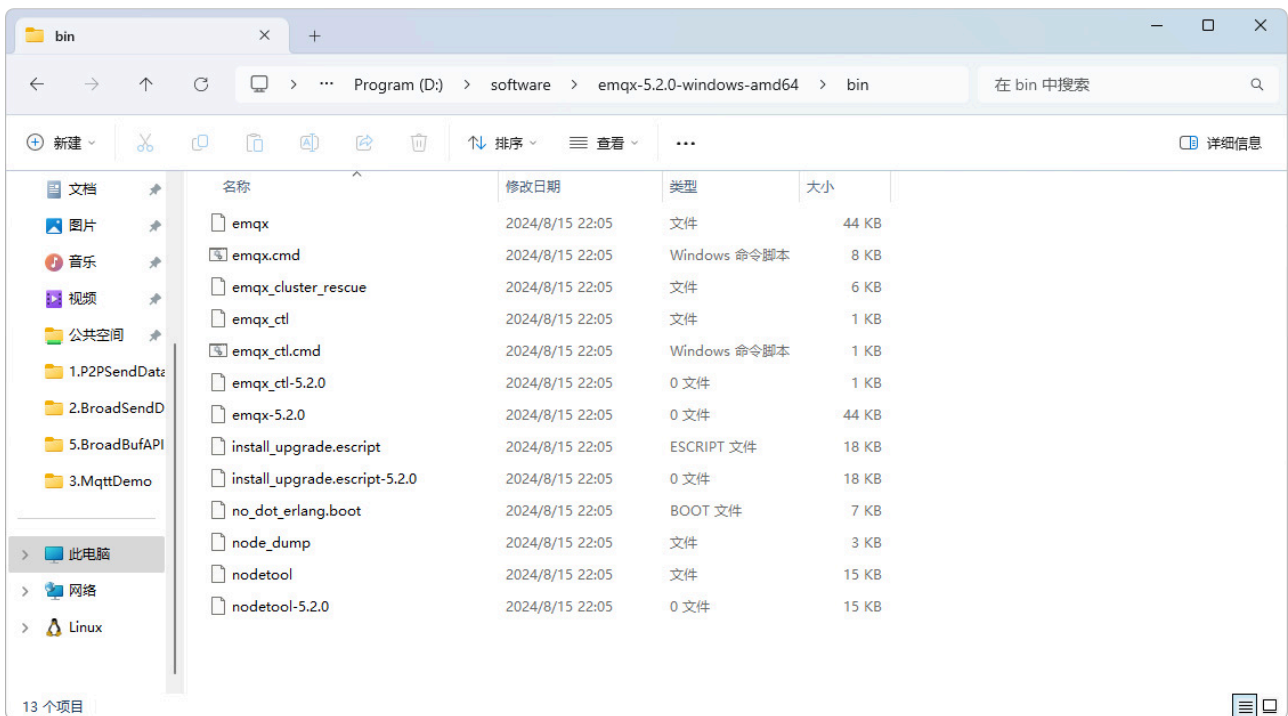
### 4.1 步骤1：Redis安装

如果没有安装过Redis，或者不知道Redis相关知识的，请查看文档：

[\\*:\PX4PSP\RflySimAPIs\9.RflySimComm\1.BasicExps\e0-ResourcesFile\Windows下Redis安装教程.pdf](*.:\PX4PSP\RflySimAPIs\9.RflySimComm\1.BasicExps\e0-ResourcesFile\Windows下Redis安装教程.pdf)

### 4.2 步骤2：本机实验

进入Redis的安装目录中。



在该目录中打开cmd，输入如下的命令，来开启Redis服务器。

```
.\redis-server.exe
```

```
Windows PowerShell
安装最新的 PowerShell, 了解新功能和改进! https://aka.ms/PSWindows

PS D:\software\Redis\Redis-x64-5.0.14.1> .\redis-server.exe
[33144] 31 Aug 18:03:09.030 # o000o000o000o Redis is starting o000o000o000o
[33144] 31 Aug 18:03:09.030 # Redis version=5.0.14.1, bits=64, commit=ec77f72d, modified=0, pid=33144, just started
[33144] 31 Aug 18:03:09.030 # Warning: no config file specified, using the default config. In order to specify a config
file use d:\software\redis\redis-x64-5.0.14.1\redis-server.exe /path/to/redis.conf

Redis 5.0.14.1 (ec77f72d/0) 64 bit

Running in standalone mode
Port: 6379
PID: 33144

http://redis.io

[33144] 31 Aug 18:03:09.033 # Server initialized
[33144] 31 Aug 18:03:09.034 * DB loaded from disk: 0.001 seconds
[33144] 31 Aug 18:03:09.034 * Ready to accept connections
```

注意，默认情况下该服务是前台运行的服务，如果关闭该命令框，则Redis服务器会被关闭。

双击运行"[SITLRun4MavlinkFull.bat](#)"自动创建四个飞机。等待所有CopterSim的左下角打印"PX4: GPS 3D fixed & EKF initialization finished."。

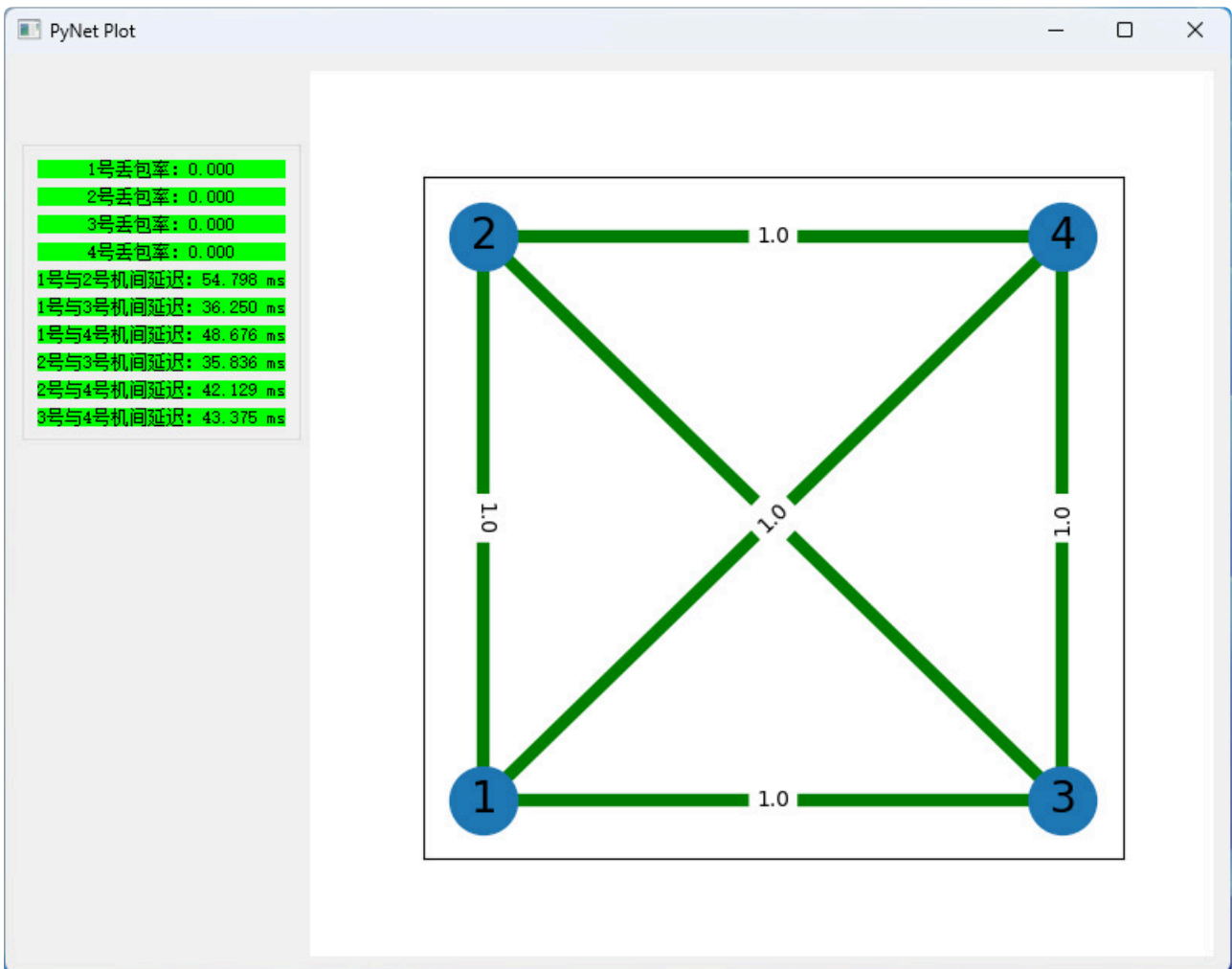


双击运行"Python38Run.bat", 在其中输入"python NetworkSimulationNoRflsim.py"。

```
C:\windows\system32\cmd.exe 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

E:\summerwork\work1\9.RflySimComm\0.ApiExps\7.NetSimMini_redis_nomat>python NetworkSimulationNoRflysim.py
NetworkSimulation Start .....
```

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



使用 [Python38Run.bat](#) , 运行四个UAV控制程序, 也可以使用 [UavPythonRunALL.bat](#) 一键启动。

可以看到控制台打印了飞机的信息。

```
C:\PX4PSP\Python38\python. x + v
Check i
CopterS:
Start 0
收到其他
1 3 4号
1号飞机
36]
3号飞机
4号飞机
休眠一秒
1 3 4号
1号飞机
14]
3号飞机
4号飞机
请检查数
尝试重新
起飞到十
2号飞机:
开始追踪
3号飞机:
开始追踪
请检查数
尝试重新
起飞到十
开始追踪
请检查数是否变化
尝试重新解锁飞机
起飞到十米高
1725098863.6682127
1725098863.6682127
39.428
收到其他三个飞机的数据
NetworkSimulation Start .....
2 3 4号飞机的仿真时间(单位s)、通信延迟(单位ms)、全局坐标(x,y,z 单位m)
2号飞机, 仿真时间: 158.72 通信延迟: 0.4978179931640625 全局坐标xyz: [-0.05203836062696787, 1.9447221170647349, -7.9
22814337814809]
3号飞机, 仿真时间: 155.644 通信延迟: 0.0 全局坐标xyz: [2.0217606221502527, 0.03527126573642003, -8.2532613010923]
4号飞机, 仿真时间: 152.592 通信延迟: 0.5784034729003906 全局坐标xyz: [2.0640146367926464, 1.9559278634063348, -7.65
602824126531]
2号飞机, 仿真时间(单位s)、通信延迟(单位ms)、全局坐标(x,y,z 单位m)
2号飞机, 仿真时间: 158.72 通信延迟: 0.4978179931640625 全局坐标xyz: [-0.05203836062696787, 1.9447221170647349, -7.9
22814337814809]
3号飞机, 仿真时间: 155.644 通信延迟: 0.0 全局坐标xyz: [2.0217606221502527, 0.03527126573642003, -8.2532613010923]
4号飞机, 仿真时间: 152.592 通信延迟: 0.5784034729003906 全局坐标xyz: [2.0640146367926464, 1.9559278634063348, -7.65
602824126531]
请检查数据是否变化
尝试重新解锁飞机
起飞到十米高
1725098863.6682127
1725098900.3399732
36.67176055908203
36.136
所有飞机到达设定高度
飞机1开始以3米/s向北飞
```

在RflySim3D中，可以看到飞机正常编队飞行。



## 5. 关键知识点

### 关键知识点1：MAVLink通信实例

程序首先创建了一个MAVLink通信实例，PX4MavCtrler类的实例用于控制无人机（本例中为2号无人机）。使用InitMavLoop()方法初始化MAVLink通信，随后使用initOffboard()启用无人机的Offboard控制模式。

### 关键知识点2：CopterSim仿真平台通信

初始化后，程序与CopterSim仿真平台建立通信，通过UDP传输仿真数据。CopterSim负责将接收到的控制指令转换为MAVLink消息发送给PX4，同时将PX4的反馈信息转发回本程序。

### 关键知识点3：无人机状态数据发布

PublicUavData()函数不断地发布2号无人机的状态数据（如GPS位置、速度、欧拉角等）到Redis的频道"UAV2"中。这些数据包括时间戳、位置信息等，用于后续的多无人机协同操作。

### 关键知识点4：多无人机数据订阅

通过SubUavNet()函数，程序订阅了来自1号、3号和4号无人机的数据（通过频道"UAV1"、"UAV3"、"UAV4"）。这些数据由其他无人机的类似发布程序推送到Redis中，并在本程序中通过sub\_callback函数处理与存储。

### 关键知识点5：通信延迟计算

在接收到来自其他无人机的数据后，程序计算并记录了每个无人机数据的通信延迟，存储在全局变量UavData中。通过时间戳差值计算延迟，以评估网络通信的实时性和可靠性。

### 关键知识点6：协同追踪算法

在RunUavNet()函数中，程序首先等待接收到至少3个其他无人机的数据，然后执行协同操作。比如，它指示2号无人机追踪1号无人机的位置，保持与其相对固定的距离。控制指令通过比例控制算法计算无人机的速度，并实时更新发送给CopterSim和PX4。

## | 关键知识点7：延迟测量与评估

程序设计中，延迟测量是一个重要环节。通过timeDelay字段计算各无人机间的通信延迟，实验结果可以帮助评估网络条件下的仿真效果。

## | 关键知识点8：追踪算法验证

通过不断调整2号无人机的速度和方向，程序验证了基于比例控制的无人机追踪算法在仿真环境中的表现。

## | 关键知识点9：数据一致性验证

程序多次输出各无人机的仿真时间、通信延迟和全局坐标，验证仿真数据的一致性和正确性。

## | 6.参考资料

1. [RflySim官方文档](#)
2. [Windows下Redis安装教程](#)

## | 7.常见问题

### | Q1：Redis服务无法启动怎么办？

A1：检查Redis是否正确安装，确认Redis安装目录下存在redis-server.exe文件。在命令行中运行".\redis-server.exe"来启动服务。

### | Q2：CopterSim没有显示"PX4: GPS 3D fixed & EKF initialization finished."信息。

A2：等待一段时间让PX4完成初始化过程，确保仿真环境正常运行。如果长时间未显示，可能需要检查PX4和CopterSim的安装配置。

## Q3: Python程序运行报错。

A3: 确保Python环境正确配置，确认所需依赖库已安装，如mavlink、redis等库。检查 [Python38Run.bat](#) 是否能正常启动Python环境。

---

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