Intelligent unmanned cluster system development and practice Full-stack development case based on RflySim toolchain

Lecture 9 Communication protocol and cluster networking

飞思实验室

outline

1. Experimental platform configuration
2. Introduction to key interfaces
3. Basic experimental cases
   (free version)
4. Advanced interface experiment
   (personal version)
5. Advanced case experiments
   (collection version)
6. Extended case (full version)
7. Summary

1. Installation method

- 1.1 Components that need to be installed
- • Visual Studio 2017 (both trial and full versions need to be installed)
- • Configure the C++ compiler for MATLAB (both trial and full versions need to be installed)
- • Matlab 2023a* (advanced full version installation)

The following describes the installation method of Visual Studio 2017 (requires Internet connection): In this platform, the installation package of Visual Studio 2017 has been placed

1. Installation method

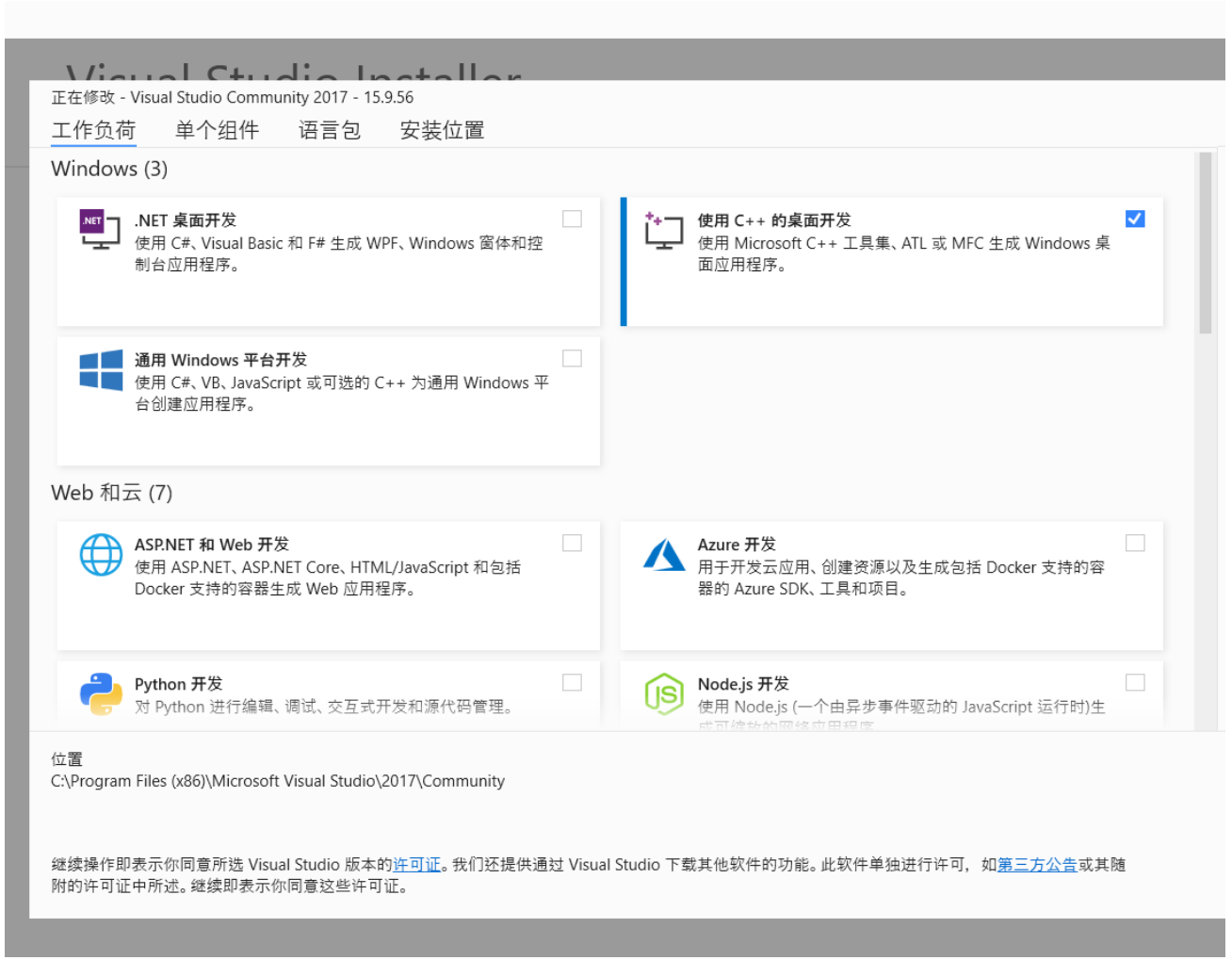
- 1.2 Installation method of Visual Studio 2017
- • First, we can open the platform installation location and find the location *:\PX4PSP\RflySimAPIs. Here are some routines in the platform and software installation packages.
- • After that, we can open the content of Chapter 4 and find the basic version of the routine, 4.RflySimModel .BasicExps, where we can find the folder named VS2017Installer, which is the installation package of Visual Studio 2017.

Online installation steps (requires Internet connection) are as follows: Visual Studio Older Downloads - 2019, 2017, 2015, and previous versions (microsoft.com)

1. Installation method

1.2 Installation method of Visual Studio 2017

- Install Visual Studio 2017 (you can also use other versions, as long as MATLAB can recognize it).
- The Visual Studio compiler will be used in many areas of subsequent courses, such as the use of MATLAB S-Function Builder module, Simulink automatically generating C/C++ model code, etc.
- For this course content, you only need to check "Desktop Development in C++" in the picture on the right.



1. Installation method

- **1.2 Installation method of Visual Studio 2017**

• Note: Higher versions of MATLAB can also install VS2019, but MATLAB can only recognize Visual Studio versions lower than its own, so MATLAB 2017b cannot recognize VS 2019.

• Note: Please do not change the default installation directory of VS (for example, install to drive D), otherwise MATLAB will not be recognized.

• Cannot use Mingw compiler, requires VS

1. Installation method

- **1.3 Configure the C++ compiler for MATLAB**

• Enter the command “mex - setup” in the MATLAB command line window

• Generally speaking, the VS 2017 compiler will be automatically recognized and installed. As shown in the picture on the right, "MEX is configured to use 'Microsoft Visual C++ 2017' for compilation", indicating that the installation is correct.

• If there are other compilers, you can also switch to other compilers such as VS 2013/2015 on this page

1. Installation method

- **1.4 Installation method of Matlab 2023a**
- **MATLAB installation package download path:**
- **https://ww2.mathworks.cn/products/matlab.html**

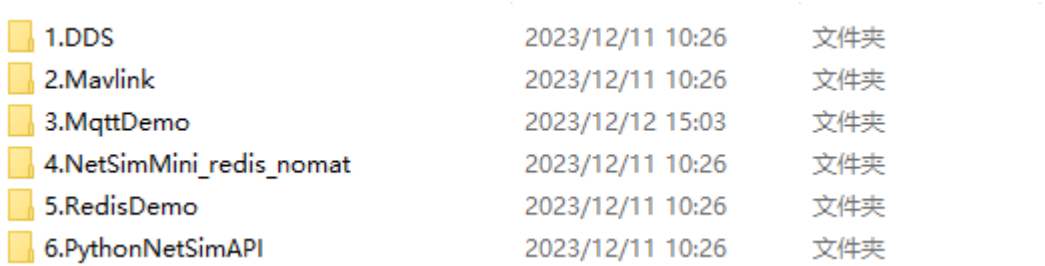
outline

1. Experimental platform configuration
2. Introduction to key interfaces
3. Basic experimental cases (free version)
4. Advanced interface experiment (personal version)
5. Advanced case experiments (collection version)
6. Extended case (full version)
7. Summary

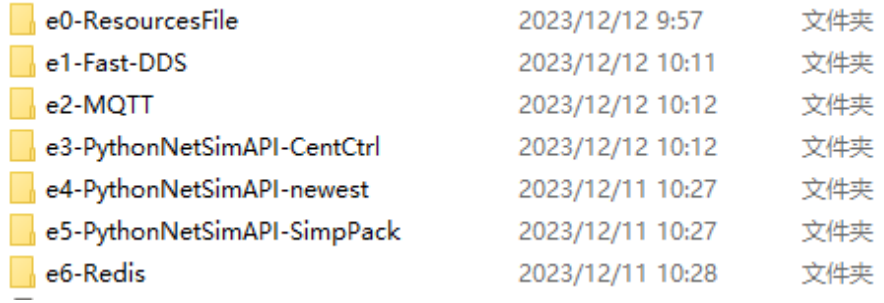
2. Introduction to key interfaces

- **2.0 Overview of Basic Experiments**

Including basic function interface "RflySimAPIs/9.RflySimComm



For details, see API_en.pdf and Readme_en.pdf



2. Introduction to key interfaces

- **2.1 DDS networking communication experiment**
- **Configure the environment required for DDS networking. Create DDS protocol and transceiver port by yourself to implement DDS communication.**

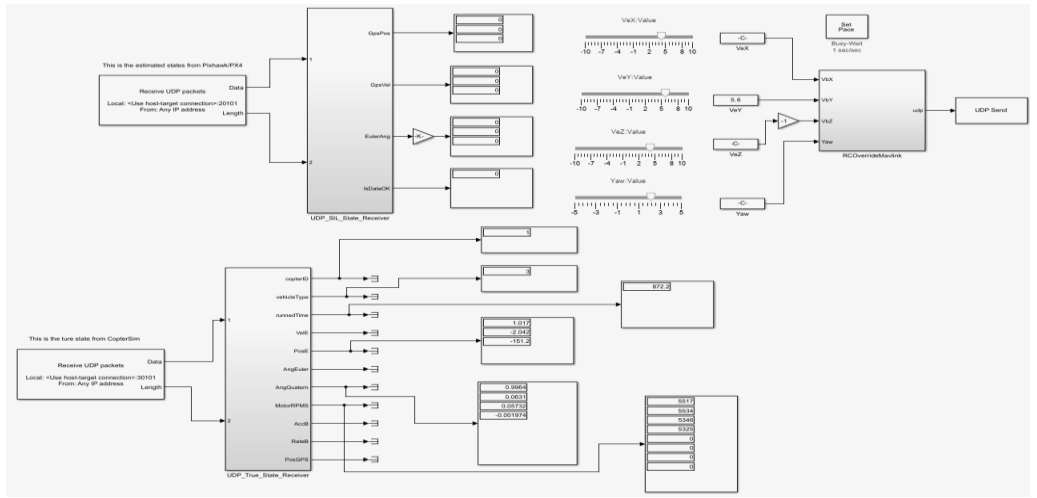
- **For detailed operations and experimental results, see 0.ApiExps\1.DDS\readme_En.pdf**

```
F:\work\Git\9.RflySimComm\SourceCode\e3\DDS组网通信例程\windows\Fast-DDS-python-1.2.0\Demo>python HelloWorldExample.py -
p publisher
Received {message}
Creating Start.
Creating publisher.
Writer is waiting discovery...
Publisher matched subscriber 1.f.a7.60.60.69.24.f7.0.0.0.0.0.0.1.4
Writer discovery finished...
Sending Hello World : 0
Sending Hello World : 1
Sending Hello World : 2
Sending Hello World : 3
Sending Hello World : 4
Sending Hello World : 5
Sending Hello World : 6
Sending Hello World : 7
Sending Hello World : 8
Sending Hello World : 9
```

```
F:\work\Git\9.RflySimComm\SourceCode\e3\DDS组网通信例程\windows\Fast-DDS-python-1.2.0\Demo>python HelloWorldExample.py -
p subscriber
Received {message}
Creating Start.
Creating subscriber.
Press any key to stopSubscriber matched publisher 1.f.a7.60.44.27.94.4b.0.0.0.0.0.0.1.3
Received Hello World : 0
Received Hello World : 1
Received Hello World : 2
Received Hello World : 3
Received Hello World : 4
Received Hello World : 5
Received Hello World : 6
Received Hello World : 7
Received Hello World : 8
Received Hello World : 9
Subscriber unmatched publisher 1.f.a7.60.44.27.94.4b.0.0.0.0.0.0.1.3
```

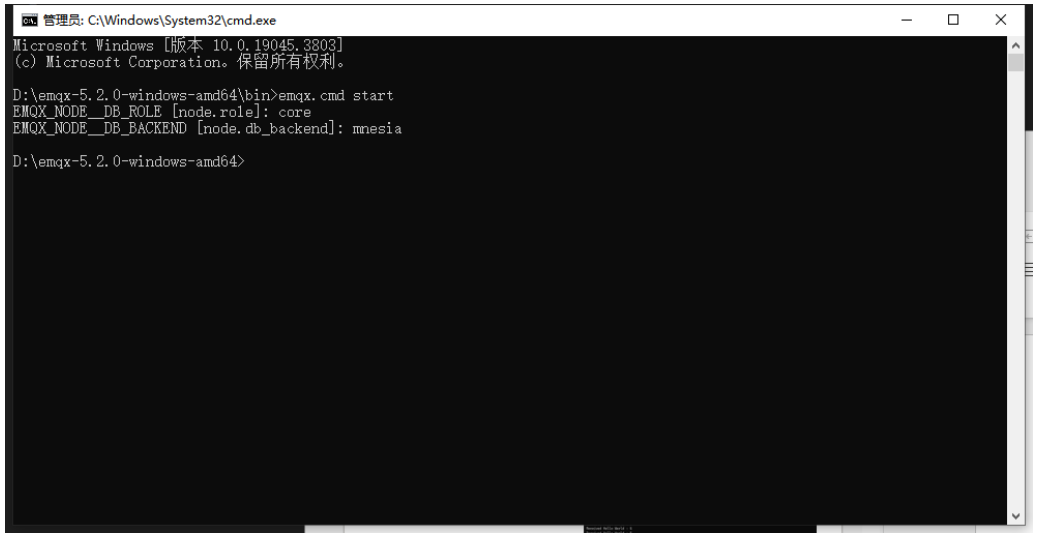
2. Introduction to key interfaces

- **2.2 MAVlink communication experiment**
- **Use MAVlink communication to achieve aircraft control and obtain aircraft flight status information using different communication modes.**
- **For detailed operations and experimental results, see [0.ApiExps\2.Mavlink\readme_En.pdf]**

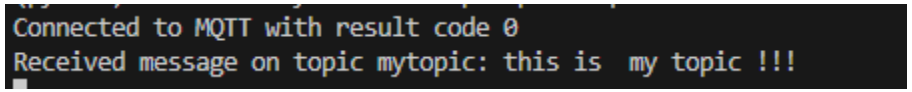


2. Introduction to key interfaces

- **2.3 Mqtt communication experiment**
- **Use mqtt to start the server, connect the publisher and subscriber to the server, and implement communication through topic sending and receiving.**



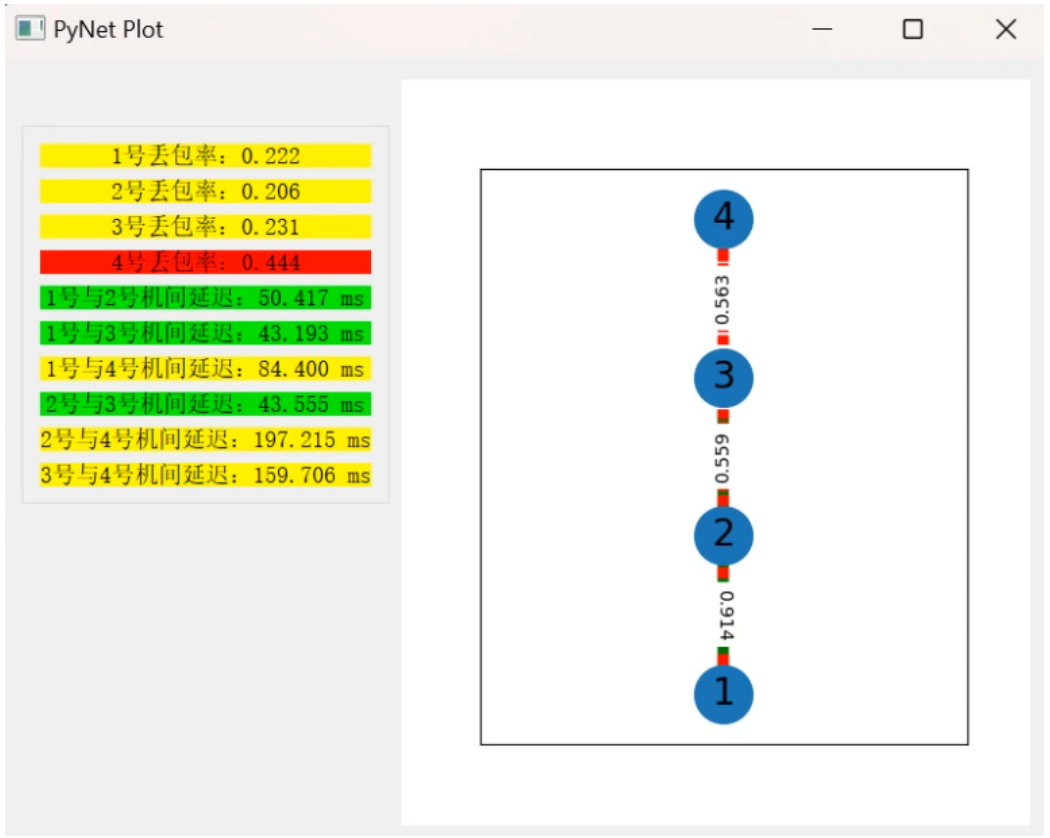
- **For detailed operations and experimental results, see 0.ApiExps\3.MqttDemo\readme_En.pdf**



2. Introduction to key interfaces

- ## 2.4 Network simulation experiment

- **By creating Redis communication, the impact of terrain on signal transmission is simulated. Obtain communication quality between drones.**
- **For information about the installation and use of Redis, see 1.BasicExps 0-ResourcesFile\Redis networking communication routine under Windows.pdf**
- **For detailed operations and experimental results, see 0.ApiExps\4.NetSimMini_redis_nomat\readme_En.pdf**



2. Introduction to key interfaces

- **2.5 Redis communication simulation experiment**
- **Start the Redis server, create a communication link, and simulate Redis communication.**

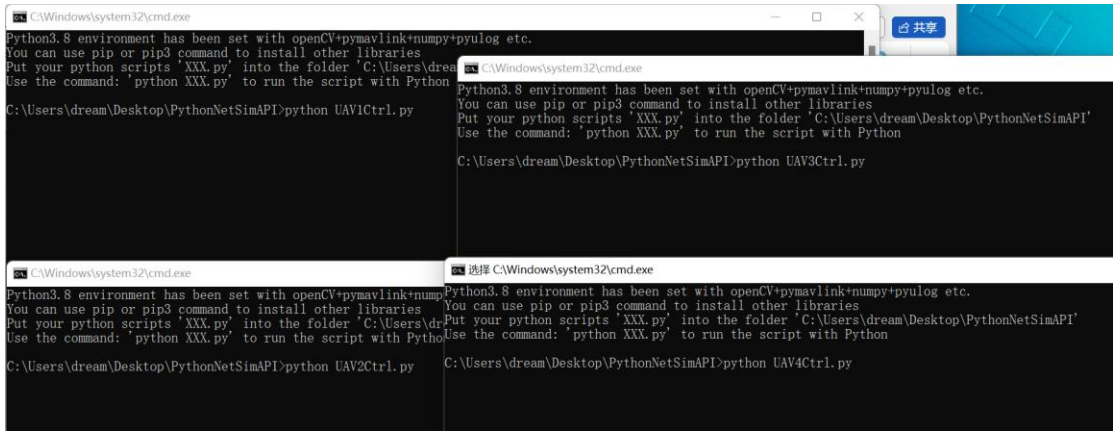
- **For detailed operations and experimental results, see 0.ApiExps\5.RedisDemo\readme_En.pdf**

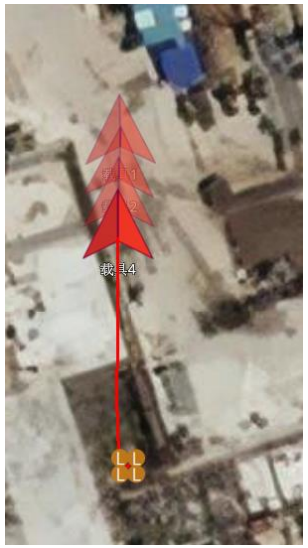
```
channel: This is a example channel
0.00012874603271484375
channel: This is a example channel
0.00015306472778320312
channel: This is a example channel
0.00016355514526367188
channel: This is a example channel
0.0001513957977294922
channel: This is a example channel
0.0001938343048095703
channel: This is a example channel
0.00018262863159179688
channel: This is a example channel
0.0001699244689941406
channel: This is a example channel
0.0001423358917236328
channel: This is a example channel
0.000141859054565429 7
channel: This is a example channel
0.00017142295837402344
channel: This is a example channel
0.0001523494720458984 4
channel: This is a example channel
0.0001685619354248047
channel: This is a example channel
0.0002028942108154297
channel: This is a example channel
0.00019931793212890625
```

2. Introduction to key interfaces

- **2.6 net networking experiment**
- **Realize data sharing by sending data to different ports under the same IP and understand the communication principle.**

- **For detailed operations and experimental results, see 0.ApiExps\6.PythonNetSimAPI\readme_En.pdf**

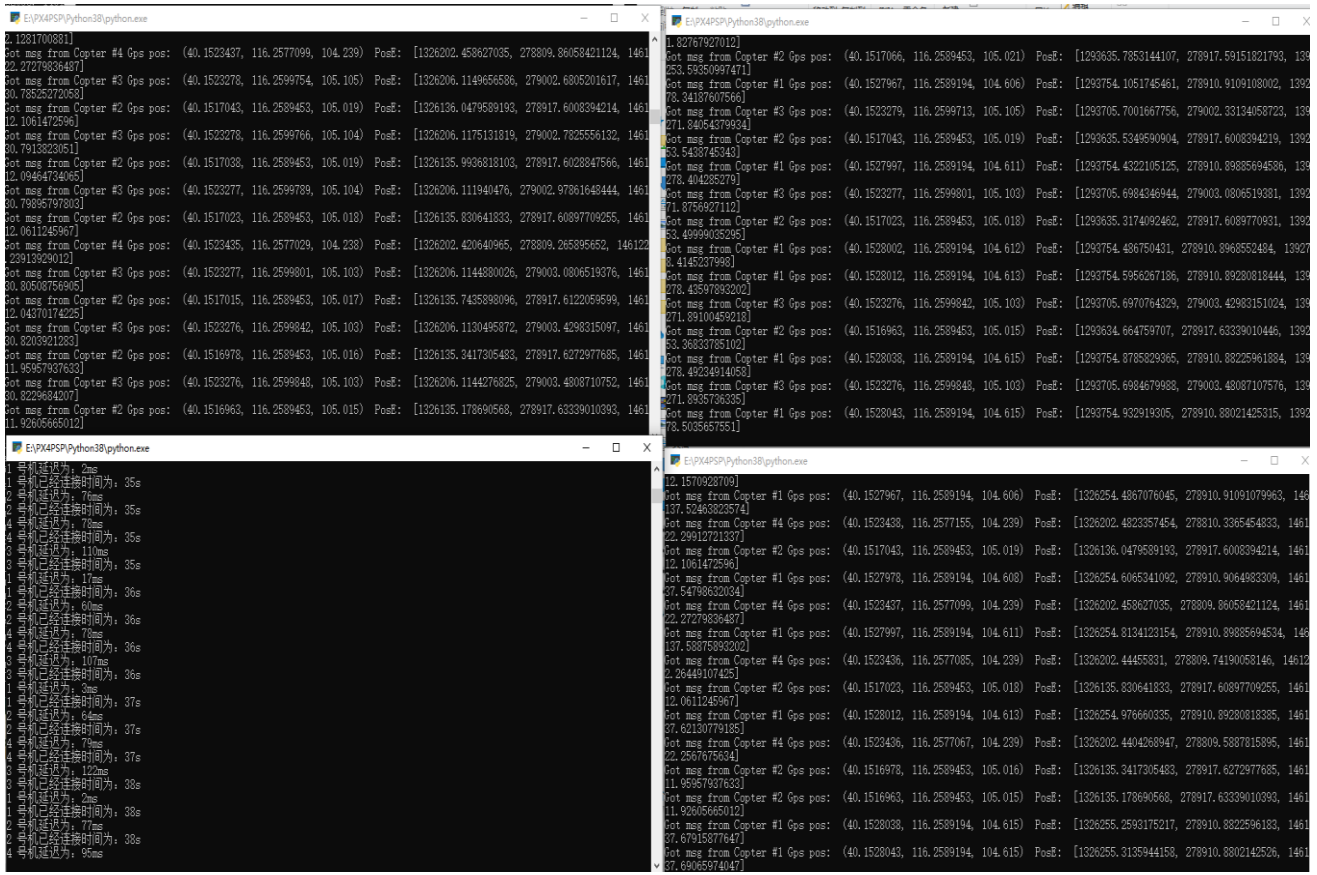




2. Introduction to key interfaces

- **2.7 Coarse-grained cluster networking experiment**
- **The data sent through the drone cluster network will be sent to the 30000 port monitored by the coarse-grained networking program, and then based on the coarse-grained networking rules, it will be judged whether it can reach the destination drone and packet loss will be calculated.**

- **For detailed operations and experimental results, see [0.ApiExps\7.NetSimMini_redis_nomat\readme_En.pdf]**



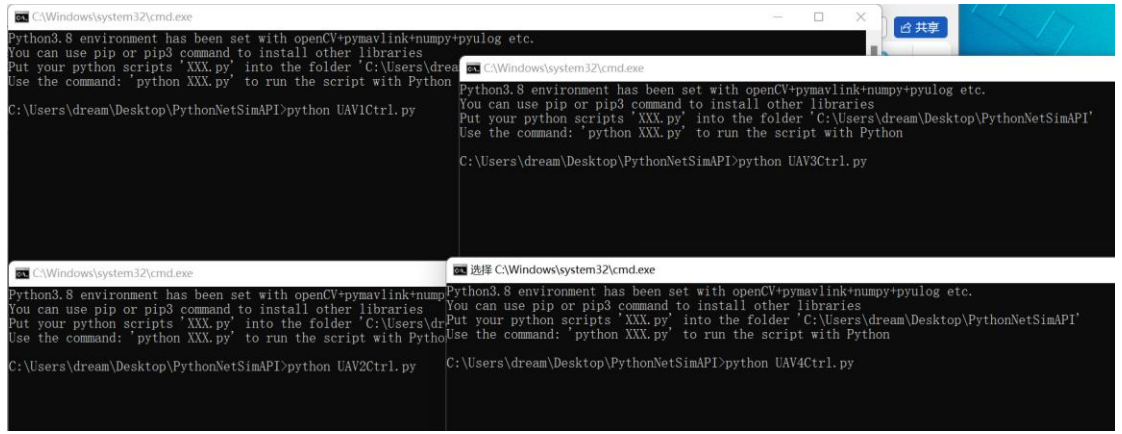
outline

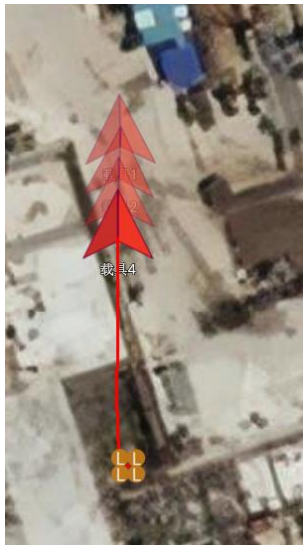
1. Experimental platform configuration
2. Introduction to key interfaces
3. Basic experimental cases (free version)
4. Advanced interface experiment (personal version)
5. Advanced case experiments (collection version)
6. Extended case (full version)
7. Summary

3. Basic experimental cases

- **3.1 fast-DDS communication networking experiment**
- **Use fast-DDS to realize information exchange between aircraft.**

- **For detailed operations and experimental results, see [1.BasicExps\e1-Fast-DDS\readme_En.pdf]**

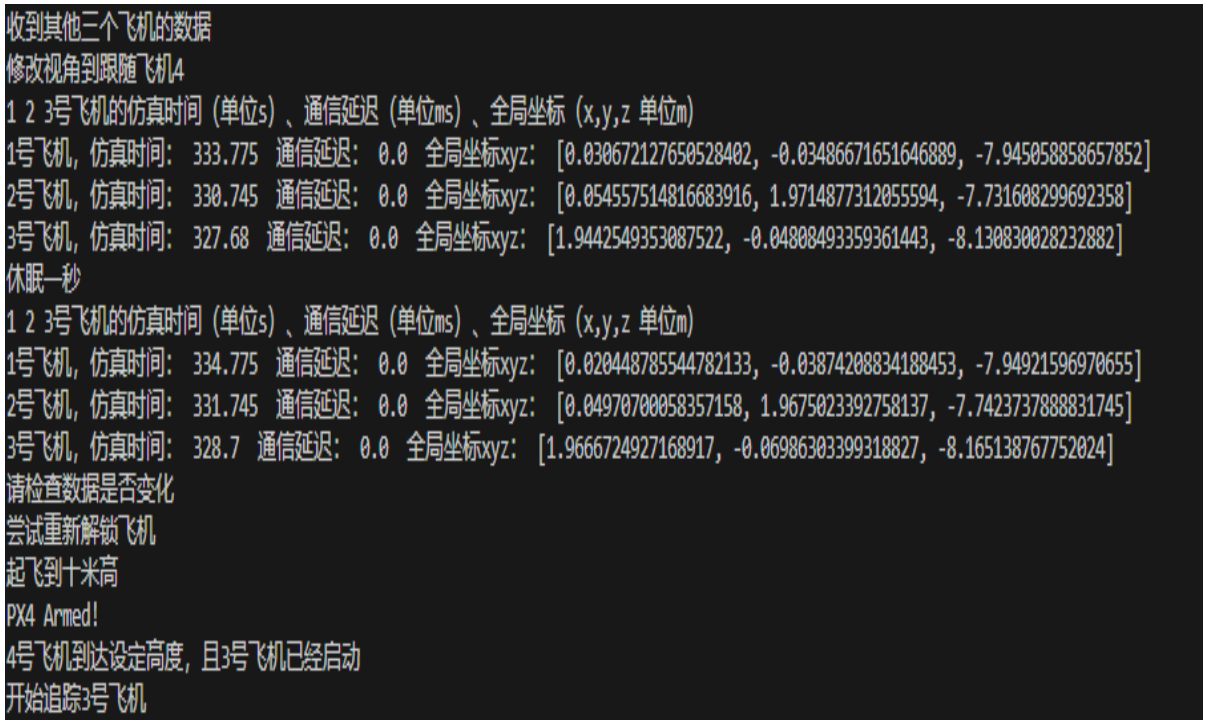




3. Basic experimental cases

- **3.2 MQTT multi-drone control experiment**
- **Use mqtt to realize information exchange between aircraft, and use Mavlink to realize the control of the aircraft itself.**

- **For detailed operations and experimental results, see 1.BasicExps\e2-MQTT\readme_En.pdf**



3. Basic experimental cases

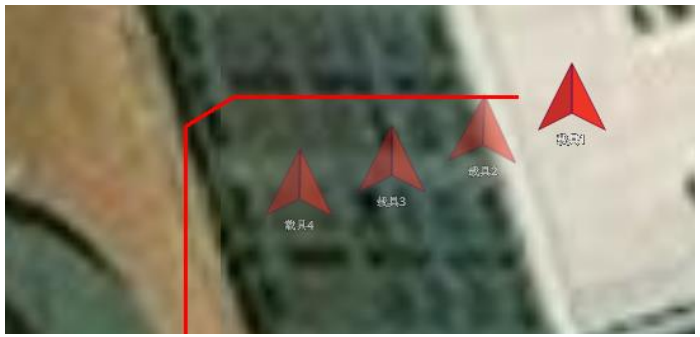
- **3.3 Net-CentCtrl aircraft communication experiment**
- **Use Net to complete information exchange between aircraft.**
- **For detailed operations and experimental results, see 1.BasicExps\e3-PythonNetSimAPI-CentCtrl\readme_En.pdf**

```
收到4号飞机的数据
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）
4号飞机，仿真时间： 42.51 通信延迟： 0.0 全局坐标xyz： (0.0, 0.0, 0.0)
休眠一秒
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）
4号飞机，仿真时间： 43.53 通信延迟： 0.0 全局坐标xyz： (0.0, 0.0, 0.0)
请检查数据是否变化
尝试重新解锁飞机
起飞到十米高
PX4 Armed!
1号飞机到达设定高度，且4号飞机已经启动
飞机1开始以3米/s向北飞
飞机1停止飞行
飞机1开始以3米/s向东飞
```

```
收到1号飞机的数据
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）
1号飞机，仿真时间： 46.005 通信延迟： 0.0 全局坐标xyz： (-0.017929215153247746, -0.076881639750
73925, -7.9771995483187705)
休眠一秒
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）
1号飞机，仿真时间： 47.025 通信延迟： 0.0 全局坐标xyz： (-0.007656581868917378, -0.081644065769
45415, -7.994564840033439)
请检查数据是否变化
尝试重新解锁飞机
起飞到十米高
2号飞机到达设定高度，且1号飞机已经启动
开始追踪1号飞机
```

```
收到2号飞机的数据
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： 48.43 通信延迟： 0.0 全局坐标xyz： (0.000919220774451723 3, 1.983118
8, -12.115752877215677)
休眠一秒
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： 49.39 通信延迟： 0.0 全局坐标xyz： (0.022019957928935 52, 1.977986785
-14.662516774157815)
请检查数据是否变化
尝试重新解锁飞机
起飞到十米高
3号飞机到达设定高度，且2号飞机已经启动
开始追踪2号飞机
```

```
收到3号飞机的数据
修改视角到跟随飞机4
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）
3号飞机，仿真时间： 49.515 通信延迟： 0.0 全局坐标xyz： (1.978295037388547, 0.004737986 18
64, -9.33546677295139)
休眠一秒
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）
3号飞机，仿真时间： 50.54 通信延迟： 0.0 全局坐标xyz： (1.9741302652803498, 0.004282861 88
7, -11.89173308077822)
请检查数据是否变化
尝试重新解锁飞机
起飞到十米高
4号飞机到达设定高度，且3号飞机已经启动
开始追踪3号飞机
```



3. Basic experimental cases

- **3.4 Net aircraft communication experiment**
- **Use Net to complete data interaction between aircraft. In this routine, the aircraft shares its own information to the networking program and then the networking program distributes it to each aircraft.**
- **For detailed operations and experimental results, see 1.BasicExps\e4-PythonNetSimAPI-newest\readme_En.pdf**

```
net.enNetForward([61000],'224.0.0.10')

print('Check if CopterSim 3D Fixed...')
while True:
    if mav.isPX4Ekf3DFixed:
        print('CopterSim/PX4 3D Fixed, ready to fly.')
        break
    time.sleep(0.5)

time.sleep(1)
print('Start Offboard Send!')
# 启用Offboard控制
mav.initOffboard()
time.sleep(1)


# 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')
```

```
def receiveFromUavCommunicationDatas(self,  port:int = 61000):
    sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM, socket.IPPROTO_UDP)
    sock.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
    sock.bind(('0.0.0.0', port))
    # time.sleep(3)
    mreq = struct.pack("=4sl", socket.inet_aton("224.0.0.10"), socket.INADDR_ANY)
    sock.setsockopt(socket.IPPROTO_IP,socket.IP_ADD_MEMBERSHIP,mreq)

    while True:
        buf, addr= sock.recvfrom(1024)
        cksum,cpID,sendMode,StartIdx,SendMask,TimeUnix=struct.unpack('iiiiQd',buf[0:32])
        destIds = self.getDests(SendMask, StartIdx)
        delaysandnewDestIds =[]
        curTime = time.time_ns()/1e9
        targetTimePortList=[]
        for tgID in destIds:
            targetTimePortList.append((TimeUnix+0.005,60000+tgID))# 统一添加3毫秒的延迟
        # 使用互斥锁
        self.uavsSendmutex.acquire()
        self.uavsPacketBufList.append([buf,targetTimePortList])
        self.uavsSendmutex.release()
```

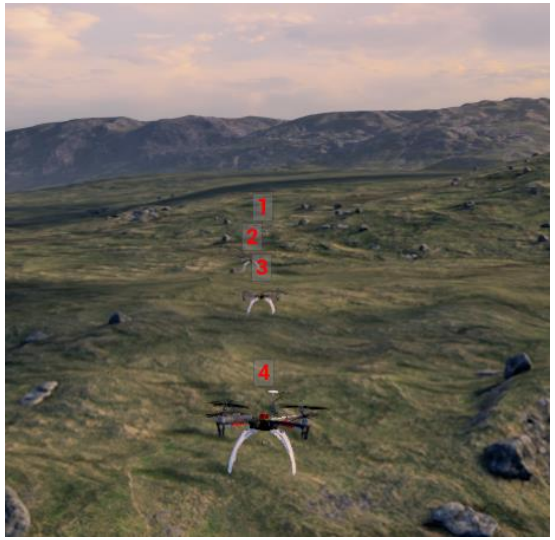
3. Basic experimental cases

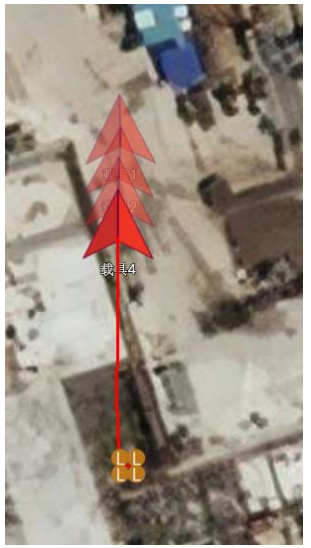
- **3.5 Net aircraft communication experiment**

- **Use Net to complete information exchange between aircraft.**

- **For detailed operations and experimental results, see 1.BasicExps\e5-PythonNetSimAPI-SimpPack\readme_En.pdf**

```
收到4号飞机的数据
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）
4号飞机，仿真时间： 42.51 通信延迟： 0.0 全局坐标xyz： (0.0, 0.0, 0.0)
休眠一秒
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）
4号飞机，仿真时间： 43.53 通信延迟： 0.0 全局坐标xyz： (0.0, 0.0, 0.0)
请检查数据是否变化
尝试重新解锁飞机
起飞到十米高
PX4 Armed!
1号飞机到达设定高度，且4号飞机已经启动
飞机1开始以3米/s向北飞
飞机1停止飞行
飞机1开始以3米/s向东飞
```





outline

1. Experimental platform configuration
2. Introduction to key interfaces
3. Basic experimental cases (free version)
4. Advanced interface experiment (personal version)
5. Advanced case experiments (collection version)
6. Extended case (full version)
7. Summary

4. Advanced interface experiment

- **4.1 Redis communication experiment**

- **Send multiple times, subscribe multiple times. Implement inter-aircraft communication based on Redis.**

- **For detailed operations and experimental results, see 2.AdvExps\e1-Redis\e6.1\readme_En.pdf**

```
Check if CopterSim 3D Fixed...
CopterSim/PX4 3D Fixed, ready to fly.
Start Offboard Send!
Failsafe mode deactivated
PX4 Armed!
收到其他三个飞机的数据
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： 213.635 通信延迟： 0.0 全局坐标xyz： [69.08198264406333, -0.35688943640049975, -7.75258577019893]
3号飞机，仿真时间： 210.63 通信延迟： 0.0 全局坐标xyz： [68.25676685900378, -0.4029859306441932, -7.908592274856632]
4号飞机，仿真时间： 207.56 通信延迟： 0.0 全局坐标xyz： [67.29705473668884, -0.492072955089202, -8.089055331127543]
休眠一秒
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： 214.63 通信延迟： 0.0 全局坐标xyz： [69.08603385255942, -0.3520948387747673, -7.758044683942426]
3号飞机，仿真时间： 211.65 通信延迟： 0.0 全局坐标xyz： [68.264266553828, -0.3942648472415595, -7.898917785358494]
4号飞机，仿真时间： 208.56 通信延迟： 0.0 全局坐标xyz： [67.34635588414977, -0.4868220243030692, -8.09900900878181]
请检查数据是否变化
尝试重新解锁飞机
起飞到十米高
所有飞机到达设定高度
飞机1开始以3米/s向北飞
飞机1停止飞行
飞机1开始以3米/s向东飞
```

```python
def PublicUavData():
    while True:
        # 如果mav收到了消息
        mav.netEvent.wait()
        # double uavTimeStmp 时间戳
        # float uavAngEular[3] 欧拉角 弧度
        # float uavVelNED[3] 速度 米
        # double uavPosGPSHome[3] GPS 维度（度）、经度（度
        # double uavPosNED[3] 本地位置 米 （相对起飞点）
        # double uavGlobalPos[3] 全局位置 （相对与所有飞机
        # d6f9d = 长度104
        TimeUnix = time.time_ns()/1e9 # 使用time_ns能获取
        UAV1 = {
            "timeDelay":TimeUnix,
            "CopterID":mav.CopterID,
            "uavTimeStmp":mav.uavTimeStmp,
            "uavAngEular":mav.uavAngEular,
            "uavVelNED":mav.uavVelNED,
            "uavPosGPSHome":mav.uavPosGPSHome,
            "uavPosNED":mav.uavPosNED,
            "uavGlobalPos":mav.uavGlobalPos}
        mav.netEvent.clear()
        for uav in enNetForwardList:
            redisConnect.pub_data(uav,UAV1)
        time.sleep(0.1)

def sub_callback(channel,data):
        TimeUnix = time.time_ns()/1e9 # 使用time_ns能获取
        data["timeDelay"] = TimeUnix - data["timeDelay"]
        if data["CopterID"] == 2: # 如果收到#2号飞机数据
            if  len(UavData) < 1:
                UavData.append(data)
            else :
                UavData[0] = data
        elif data["CopterID"] == 3: # 如果收到#2号飞机数据
            if  len(UavData) < 2:
                UavData.append(data)
            else :
                UavData[1] = data
        elif data["CopterID"] == 4: # 如果收到#2号飞机数据
            if  len(UavData) < 3:
                UavData.append(data)
            else :
                UavData[2] = data
```

4. Advanced interface experiment

- **4.2 Redis communication experiment**

- **Use set and get for data interaction between aircraft.**

- **For detailed operations and experimental results, see** 2.AdvExps\e1-Redis\e6.2\readme_En.pdf

```python
def SetUavData():
    while True:
        # 如果mav收到了消息
        mav.netEvent.wait()
        # double uavTimeStmp 时间戳
        # float uavAngEular[3] 欧拉角 弧度
        # float uavVelNED[3] 速度 米
        # double uavPosGPSHome[3] GPS 维度（度）、经度（度）、高度（米）
        # double uavPosNED[3] 本地位置 米 （相对起飞点）
        # double uavGlobalPos[3] 全局位置 （相对与所有飞机的地图中心）
        # d6f9d = 长度104
        TimeUnix = time.time_ns()/1e9 # 使用time_ns能获取比time更高的精度
        UAV1 = {
            "timeDelay":TimeUnix,
            "CopterID":mav.CopterID,
            "uavTimeStmp":mav.uavTimeStmp,
            "uavAngEular":mav.uavAngEular,
            "uavVelNED":mav.uavVelNED,
            "uavPosGPSHome":mav.uavPosGPSHome,
            "uavPosNED":mav.uavPosNED,
            "uavGlobalPos":mav.uavGlobalPos}
        mav.netEvent.clear()
        redisConnect.set_data("UAV1",UAV1)
        time.sleep(0.1)

def GetCallback(UavList):
    while True:
        for uav in UavList:
            data = redisConnect.get_data(uav)
            if data != False :
                UavData[uav] = data
                TimeUnix = time.time_ns()/1e9 # 使用time_ns能获取比time更高的精度
                UavData[uav]["timeDelay"] =TimeUnix - UavData[uav]["timeDelay"]
        time.sleep(0.01)

def SetUavNet():
    thread = threading.Thread(target=SetUavData,)
    thread.start()


def GetUavNet():
    GetUavList =["UAV2","UAV3","UAV4"]
    thread = threading.Thread(target=GetCallback, args=(GetUavList,))
    thread.start()
```

4. Advanced interface experiment

- **4.3 Redis communication experiment**

- **Send once, subscribe multiple times. Realize data interaction between aircraft based on Redis.**

- **For detailed operations and experimental results, see 2.AdvExps\e1-Redis\e6.3\readme_En.pdf**

```python
def PublicUavData():
    while True:
        # 如果mav收到了消息
        mav.netEvent.wait()
        # double uavTimeStmp 时间戳
        # float uavAngEular[3] 欧拉角 弧度
        # float uavVelNED[3] 速度 米
        # double uavPosGPSHome[3] GPS 维度（度）、经度（度）、高度（米）
        # double uavPosNED[3] 本地位置 米 （相对起飞点）
        # double uavGlobalPos[3] 全局位置 （相对与所有飞机的地图中心）
        # d6f9d = 长度104
        TimeUnix = time.time_ns()/1e9 # 使用time_ns能获取比time更高的精度
        UAV1 = {
            "timeDelay":TimeUnix,
            "CopterID":mav.CopterID,
            "uavTimeStmp":mav.uavTimeStmp,
            "uavAngEular":mav.uavAngEular,
            "uavVelNED":mav.uavVelNED,
            "uavPosGPSHome":mav.uavPosGPSHome,
            "uavPosNED":mav.uavPosNED,
            "uavGlobalPos":mav.uavGlobalPos}
        mav.netEvent.clear()
        redisConnect.pub_data("UAV1",UAV1)
        time.sleep(0.1)

def sub_callback(channel,data):
    UavID = channel.decode('utf-8')
    UavData[UavID] = data
    TimeUnix = time.time_ns()/1e9 # 使用time_ns能获取比time更高的精度
    UavData[UavID]["timeDelay"] =TimeUnix - data["timeDelay"]

def PubUavNet():
    thread = threading.Thread(target=PublicUavData,)
    thread.start()


def SubUavNet():
    message_type = 'message'
    channels_to_subscribe = ["UAV2", "UAV3", "UAV4"]
    thread = threading.Thread(target=redisConnect.sub_data_multiple_channels, args=(message_type,channels_to_subscribe,sub_callback,))
    thread.start()
```

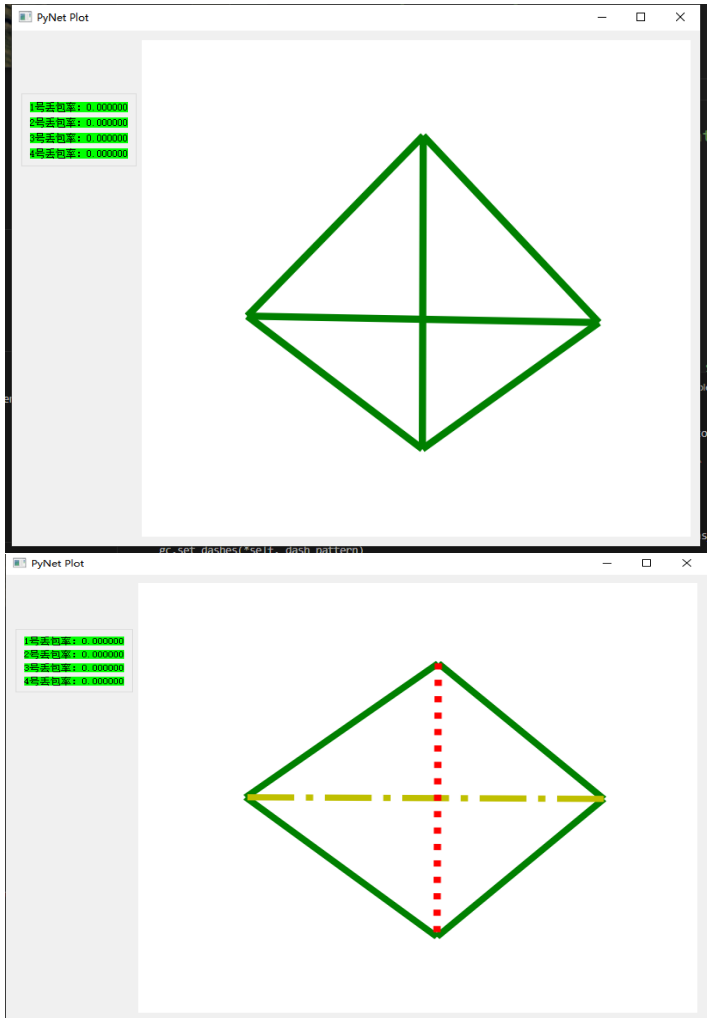
outline

1. Experimental platform configuration
2. Introduction to key interfaces
3. Basic experimental cases (free version)
4. Advanced interface experiment (personal version)
5. Advanced case experiments (collection version)
6. Extended case (full version)
7. Summary

5. Advanced case experiments

- 5.1 Net networking experiment
- This program will send the data to (netSimPort, netSimIP), transfer it through the network simulator, and then send it to the IP and port of the corresponding aircraft based on the target ID. And actually detect the communication quality between aircraft.

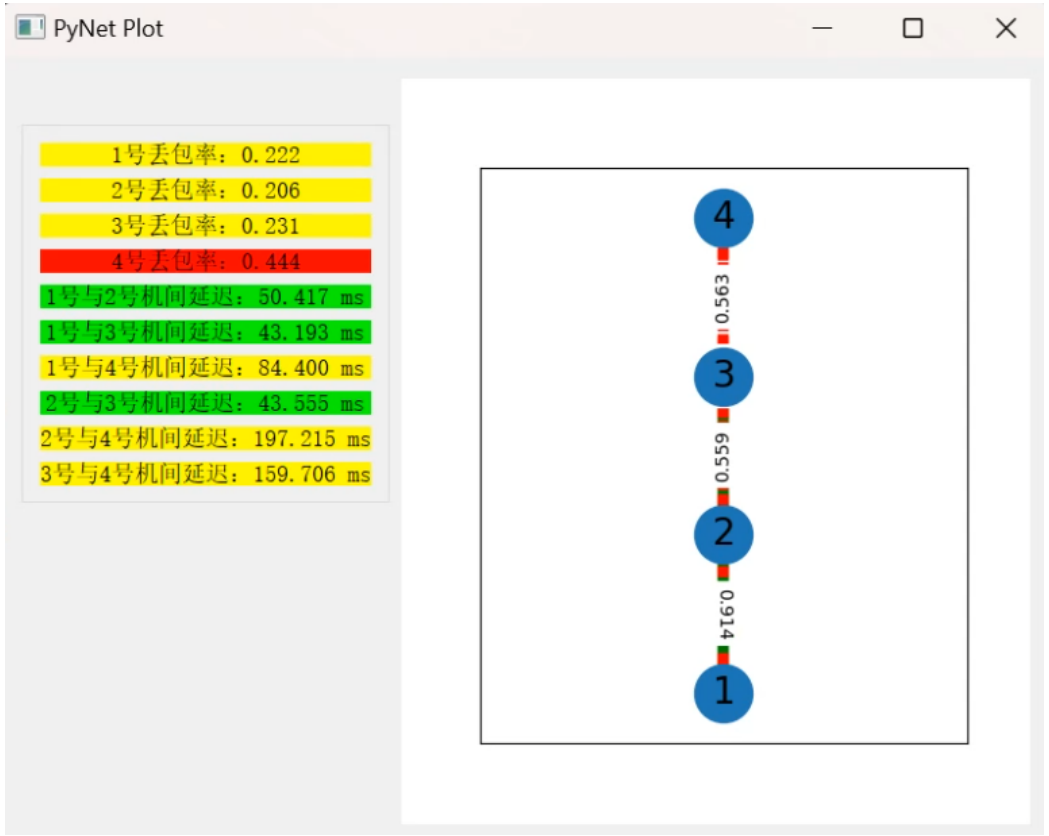
- For detailed operations and experimental results, see 2.AdvExps\e2-NetSim4Demo\readme_En.pdf



5. Advanced case experiments

- **5.2 Redis network signal quality detection experiment**
- **During simulation, create data interaction between multiple nodes, detect and return the communication quality between nodes. Deepen your understanding of Redis communication.**

- **For detailed operations and experimental results, see 2.AdvExps\e3-NetSimMini_redis_nomat\readme_En.pdf**



5. Advanced case experiments

- **5.3 Stand-alone control experiment and online detection experiment**
- **Add heartbeat detection function to detect its own communication status.**
- **For detailed operations and experimental results, see 2.AdvExps\e4-Python\readme_En.pdf**

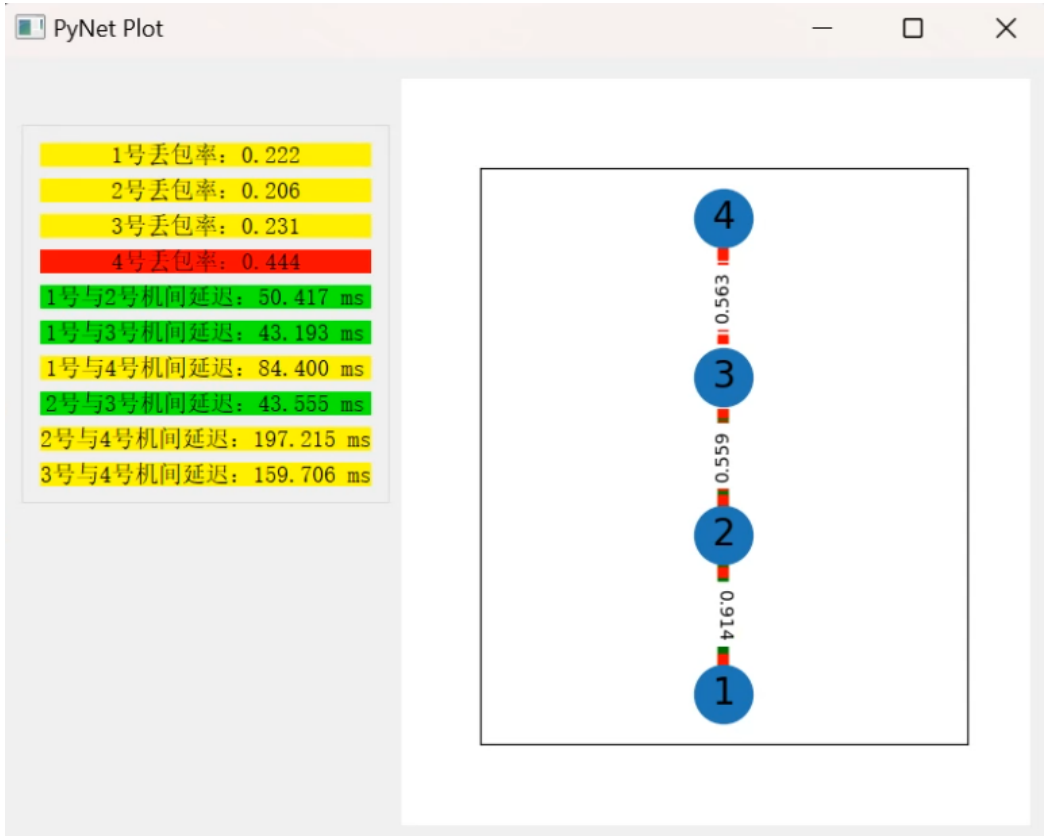
```
PX4 Armed!
port 22001
Start network serve.
[-1326379.6885100934, -278943.02584314503, -14
Send to takeoff.
1 号机延迟为: 1055ms
1 号机已经连接时间为: 2s
1 号机延迟为: 1041ms
1 号机已经连接时间为: 3s
1 号机延迟为: 1085ms
1 号机已经连接时间为: 4s
1 号机延迟为: 1167ms
1 号机已经连接时间为: 5s
1 号机延迟为: 1175ms
1 号机已经连接时间为: 6s
1 号机延迟为: 1233ms
1 号机已经连接时间为: 7s
1 号机延迟为: 1303ms
1 号机已经连接时间为: 8s
1 号机延迟为: 1350ms
1 号机已经连接时间为: 9s
1 号机延迟为: 1435ms
1 号机已经连接时间为: 10s
1 号机延迟为: 1457ms
1 号机已经连接时间为: 11s
1 号机延迟为: 1595ms
1 号机已经连接时间为: 12s
1 号机延迟为: 1672ms
1 号机已经连接时间为: 14s
1 号机延迟为: 1738ms
1 号机已经连接时间为: 15s
```

outline

1. Experimental platform configuration
2. Introduction to key interfaces
3. Basic experimental cases (free version)
4. Advanced interface experiment (personal version)
5. Advanced case experiments (collection version)
6. Extended case (full version)
7. Summary

6. Expand cases

- **6.1 Redis networking signal quality detection experiment**
- **The program communicates through Redis set and get.**
- **For detailed operations and experimental results, see 3.CustExps\e0-NetSimMini_redis_mat\readme_En.pdf**



outline

1. Experimental platform configuration
2. Introduction to key interfaces
3. Basic experimental cases (free version)
4. Advanced interface experiment (personal version)
5. Advanced case experiments (collection version)
6. Extended case (full version)
7. Summary

7. Summary

- **This lecture mainly explains the communication creation of drones and the networking between aircraft. It is divided into three parts: basic experiments, advanced experiments and extended cases. It can realize local area networking, drone cluster communication and different communication architectures.**

If you have any questions, please go to https://rflysim.com/ for more information.

More tutorials on RflySim

Scan the QR code for consultation and communication

RflySim technical exchange group

Thanks！