
1. 实验名称及目的

1.1. 实验名称

UDP 直传点云数据入门实验

1.2. 实验目的

使用 UDP 直传方式进行点云数据传输实验

1.3. 关键知识点

本实验主要是实现通过 Python 接口 VisionCaptureApi.py 使用（见 RflySimAPIs\RflySimSDK\vision 目录）发送取图请求。通过 PX4MavCtrler.py（见 RflySimAPIs\RflySimSDK\ctrl 目录）使用创建控制接口，以此向 RflySim 发送控制指令，并使用 UDP 获取点云数据，将数据传输到定义的绘制点云数据的值，从而在虚拟机中绘制出点云图。关键代码解析如下：

本例子和其他分布式例子的区别，主要在于直接通过 UDP 直传传出点云数据。

关键知识点 1: `SendProtocol[0]` 决定了图像的传出模式。`SendProtocol[0]=0`: 共享内存（仅限 Windows 下获取图像），`1`: UDP 直传 png 压缩，`2`: UDP 直传图片不压缩（只适用图片类传感器），`3`: UDP 直传 jpg 压缩（只适用图片类传感器）。如果是激光雷达数据只有 `0` 或 `1`（共享内存和 UDP 网络传输）。

1) 视觉接口使用

```
vis = VisionCaptureApi.VisionCaptureApi() # 创建一个视觉传感器实例
vis.jsonLoad() # 加载 Config.json 中的传感器配置文件
isSuss = vis.sendReqToUE4(0, TargetIP) # 向 RflySim3D 发送取图请求，发给 ip 为 TargetIP 的地址
vis.startImgCap() # 开启取图
vis.hasData[i] # 图片 i 数据是否更新
vis.Img[i] # 图片 i 数据（像素矩阵）
```

2) 相机数量和参数配置

其中，视觉传感器的初始状态由本文件夹下的 `Config.json` 决定，主要包含以下配置项：

`"SeqID":0`: 使用自动更新 ID 的方式，创建了 SeqID 为 0 的视觉传感器
`"TypeID":20`: 传感器类型为激光雷达
`"TargetCopter":1`: 相机绑定在 1 号飞机上

"SendProtocol":[1,0,0,0,0,0,0,0]: 传输模式为 1: UDP 网络传输模式（图片使用 jpeg 压缩，点云直传）。

"SensorPosXYZ":[0,0,-0.3]: 相机分布位置。

3) Open3DShow 接口使用

```
show3d=Open3DShow.Open3DShow() # 创建 3D 点云显示实例
show3d.CreatShow(0) # 创建点云显示窗口
show3d.UpdateShow(vis.Img[0]) # 更新点云
```

4) 飞机控制指令

```
mav = PX4MavCtrl.PX4MavCtrler(1) # 创建飞机控制实例
mav.InitMavLoop() # 初始化 Mavlink 监听程序，读取飞机数据
mav.InitTrueDataLoop() # 初始化来自 CopterSim 的 UDP True 数据监听循环
mav.initOffboard() # 进入 Offboard 模式
mav.SendMavArm(True) # 解锁飞控
mav.SendPosNED(0, 0, -8, 0) # 发送 8 米高的位置控制指令
mav.SendCopterSpeed(3) # 发送速度为 3m/s 控制指令
mav.endOffboard() # 结束 offboard 模式
mav.stopRun() # 结束 mav 飞机实例运行
```

5) 其余代码说明

```
timeInterval = 1/30.0 # 以 30hz 的频率进行控制
lastTime = lastTime + timeInterval # 设置每一帧的处理结束时间
sleepTime = lastTime - time.time() # 计算休息时间，从而保持按照设定的频率执行代码
targetPosE=targetPosE+Error2UE4Map[j] # 设置飞机位置
```

2. 实验效果

本实验通过平台接口进行 RflySim3D 直接 10hz 频率 UDP 直传点云数据。

3. 文件目录

例程目录: [\[安装目录\]\RflySimAPIs\8.RflySimVision\0.ApiExps\8.LidarAPIDemo\1.Share dMemoryClientServer](#)

文件夹/文件名称	说明
client_ue4_SITL.bat	Windows 客户端启动仿真配置文件

server_ue4.py	取图控制程序（支持 Windows 或 Linux 分布式运行）
Config.json	视觉传感器配置文件
lidar.rviz	Rviz 配置文件
Python38Run.bat	Windows 下 Python 程序运行脚本
WinWSL.bat	WSL1/Ubuntu 20.04 环境程序运行脚本
WslGUI.bat	WSL1/Ubuntu 20.04 可视化界面脚本

4. 运行环境

序号	软件要求	硬件要求	
		名称	数量(个)
1	Windows 10 及以上版本	笔记本/台式电脑 ^①	1
2	RflySim 工具链		
3	Visual Studio Code		可选
4	Linux (Ubuntu 20.04)	WinWSL	1
5	Linux (Ubuntu 20.04)	虚拟机/视觉盒子/其他板卡	可选

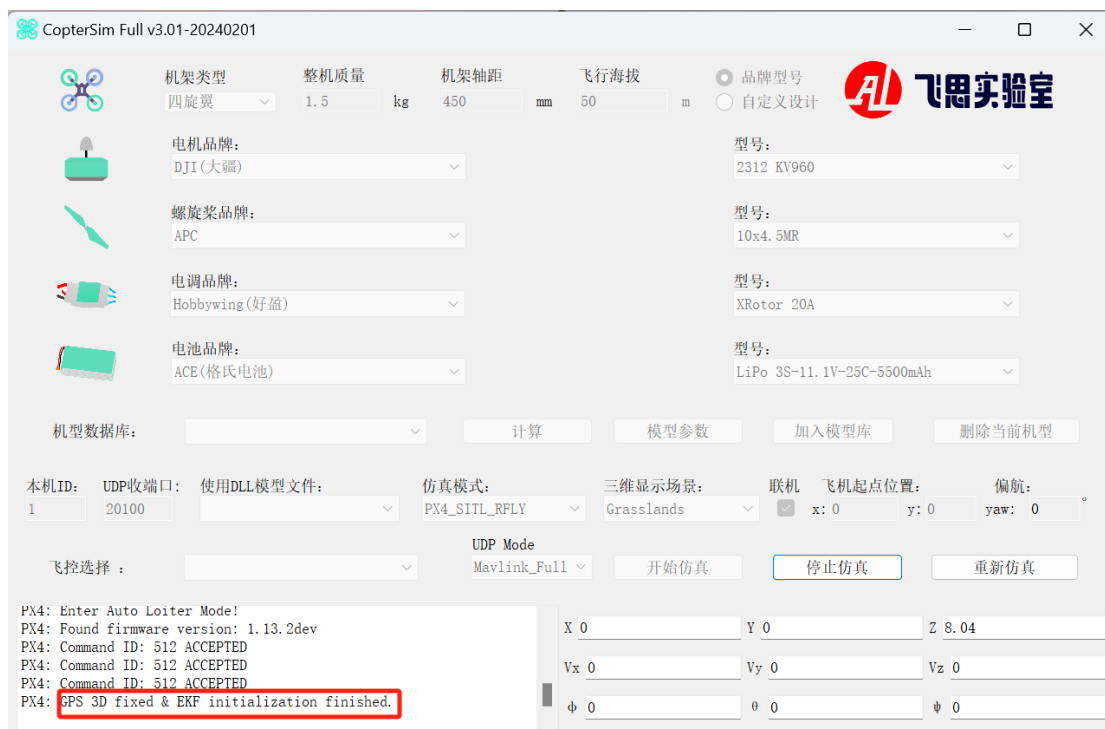
①：推荐配置请见：<https://rflysim.com>

5. 实验步骤

5.1. 必做实验：WinsWSL 控制

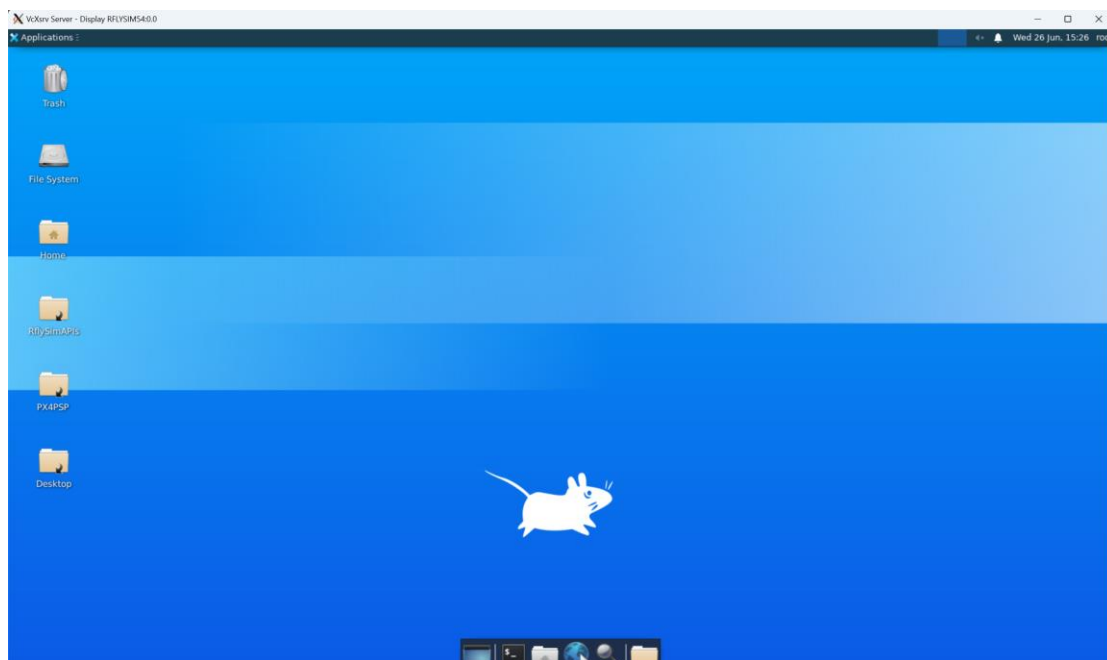
Step 1: 开启仿真

双击运行 LidarAPIDemo.bat 开启一个飞机的软件在环仿真。将会启动 1 个 QGC 地面站，1 个 CopterSim 软件且其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，并且 RflySim3D 软件内有 1 架无人机。



Step 2: 开启 WSL 可视化界面

双击打开 WslGUI.bat，启动 WSL 可视化界面。（注：如果打开发现窗口白屏，没有桌面，则关了重开一两次。）

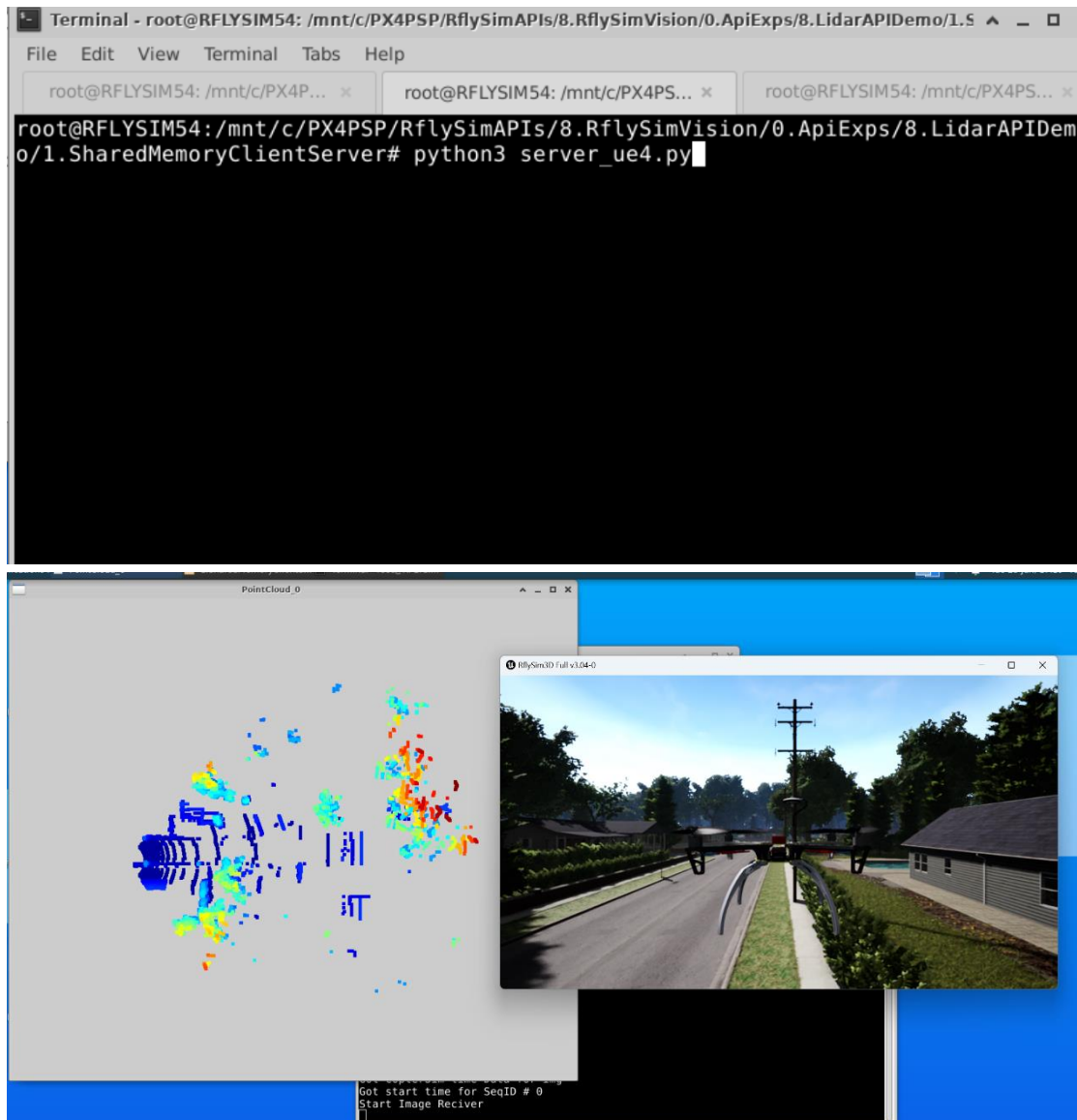


注意：参考[\[安装目录\RflySimAPIs\1.RflySimIntro\2.AdvExps\7_WslGUIIntro.pdf\]](#)，了解 WslGUI 的功能与使用。

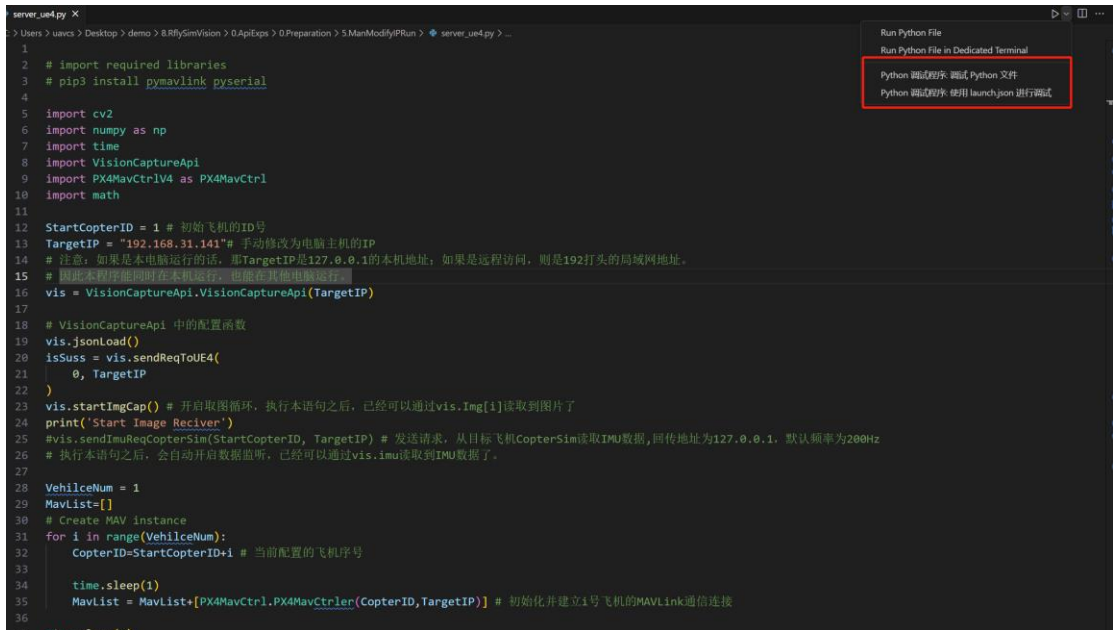
Step 3: 运行控制程序

双击打开 WinWSL.bat，运行 `roscore` 指令，

双击打开 WinWSL.bat，运行 `pip install matplotlib --upgrade`，更新 `matplotlib` 库，再运行命令 `python3 server_ue4.py` 运行脚本 `server_ue4.py`。可见绘制的点云图不断更新：



备注：可以参考[\[安装目录\]\RflySimAPIs\1.RflySimIntro\2.AdvExps\e8_WslVsCode\Intro.pdf](#)来使用 VS Code 开发并调试 Ubuntu 下 python 文件。



```
1 > Users > users > Desktop > demo > 8.RflySimVision\0.ApiExps\0.Preparation\5.ManModfyIPRun > server_un4.py > ...
2 # import required libraries
3 # pip3 install pymavlink pyserial
4
5 import cv2
6 import numpy as np
7 import time
8 import VisionCaptureApi
9 import PX4MavCtrlV4 as PX4MavCtrl
10 import math
11
12 StartCopterID = 1 # 初始飞机的ID号
13 TargetIP = "192.168.31.141" # 手动修改为电脑主机的IP
14 # 注意：如果是本电脑运行的话，那TargetIP是127.0.0.1的本地地址；如果是远程访问，则是192打头的局域网地址。
15 # 因此本程序能随时在本机运行，也能在其他电脑运行。
16 vis = VisionCaptureApi.VisionCaptureApi(TargetIP)
17
18 # VisionCaptureApi 中的配置函数
19 vis.JsonLoad()
20 isSuss = vis.sendReqToUE4(
21     0, TargetIP
22 )
23 vis.startImgCap() # 开启取图循环，执行本语句之后，已经可以通过vis.Img[i]读取到图片了
24 print('Start Image Receiver')
25 #vis.sendImuReqCopterSim(StartCopterID, TargetIP) # 发送请求，从目标飞机CopterSim读取IMU数据，回传地址为127.0.0.1，默认频率为200Hz
26 # 执行本语句之后，会自动开启数据监听，已经可以通过vis.imu读取到IMU数据了。
27
28 VehicNum = 1
29 MavList=[]
30 # Create MAV instance
31 for i in range(VehicNum):
32     CopterID=StartCopterID+i # 当前配置的飞机序号
33
34     time.sleep(1)
35     MavList = MavList+[PX4MavCtrl.PX4MavCtrl(CopterID,TargetIP)] # 初始化并建立1号飞机的MAVLink通信连接
36
37 # MavList = MavList+[PX4MavCtrl.PX4MavCtrl(StartCopterID,TargetIP)] # 初始化并建立1号飞机的MAVLink通信连接
```

Step 4: 结束仿真

在下图“LidarAPIDemo.bat”脚本开启的命令提示符 CMD 窗口中，按下回车键（任意键）就能快速关闭 CopterSim、QGC、RflySim3D 等所有程序。

5.2. 选作实验

准备工作：

虚拟机或 NX 的配置方法是相同的。

1) Ubuntu 虚拟机环境下，进行分布式联机实验。先参考[\[安装目录\RflySimAPIs\8.RflySimVision\0.ApiExps\0.Preparation\1.VMwareUbuntu\Readme.pdf\]](#)，完成虚拟机的下载与配置。

2) 用第二台 Ubuntu 电脑或 NX 板卡，实现联机实验。其他 Ubuntu 电脑的配置，先看[\[安装目录\RflySimAPIs\8.RflySimVision\0.ApiExps\0.Preparation\2.GenenalUbuntuConfig\Readme.pdf\]](#)；NX 板卡的配置方法，先看[\[安装目录\RflySimAPIs\8.RflySimVision\0.ApiExps\0.Preparation\3.NXwithPX4Config\Readme.pdf\]](#)。

扩展实验：

5.2.1 本机 Windows 接收图像实验

Step 1: 开启仿真

步骤 1 同 6、实验步骤的 Step1 步骤。

Step 2: 运行控制程序

在文件夹下，双击 Python38Run.bat，打开集成好的 python 环境，在该环境下运行 server_ue4.py 文件，输入 python server_ue4.py

注：也可以使用 VScode 打开运行（如果安装好了 VScode），VS Code 打开 server.py 并执行

5.2.2 远端 Windows 电脑接收图像实验

Step 1: 开启仿真

步骤 1 同 6、实验步骤的 Step1 步骤。

Step 2: 运行控制程序

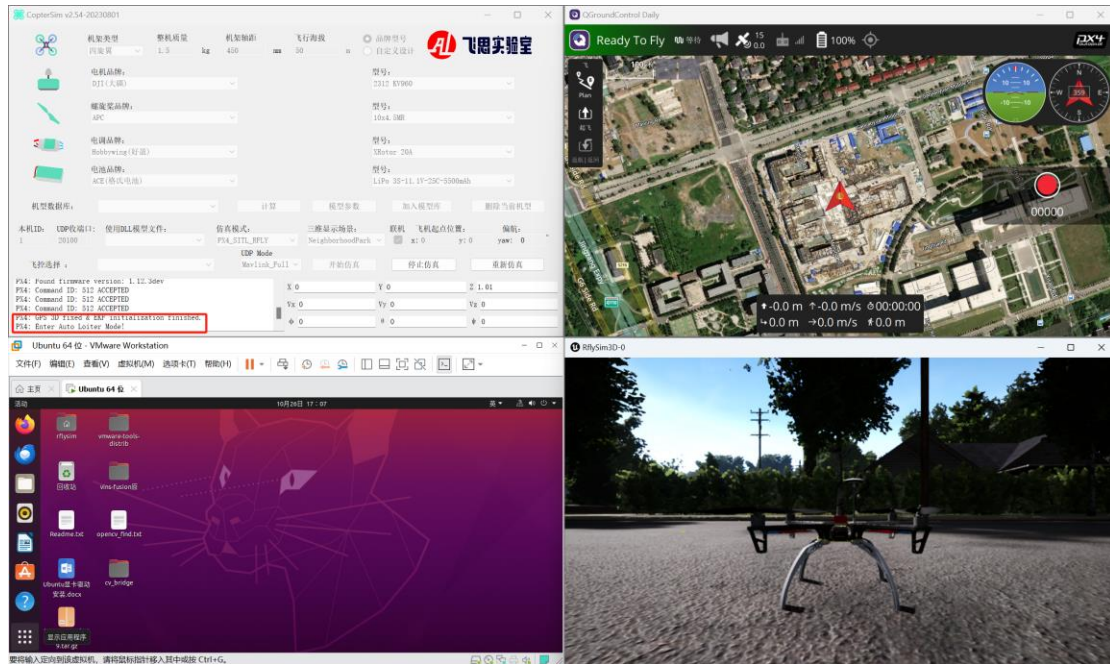
在另一台 Windows 电脑上双击 Python38Run.bat，再运行 server.py。

5.2.3 在虚拟机/视觉板卡/另一台 Ubuntu 上接收图像实验

Step 1: 开启仿真

步骤 1 同上面的 Step1 步骤。

并且启动一个 Ubuntu 虚拟机。



Step 2: 运行控制程序

在虚拟机中新建一个文件夹，并将该实验文件全拷贝过去，然后在终端中运行 `roscore` 指令，将起新终端路径选择到新建的文件夹路径，运行 `pip install matplotlib --upgrade`，更新 `matplotlib` 库，再运行命令 `python3 server_ue4.py` 运行脚本 `server_ue4.py`。可见绘制的点云图不断更新：

```

roscore http://ubuntu:11311/
uav11@ubuntu:~/桌面$ roscore
... logging to /home/uav11/.ros/log/1772f622-ab9e-11ee-a0bc-8dc56b095be1/ros-launch-ubuntu-4050.log
Checking log directory for disk usage. This may take a while.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

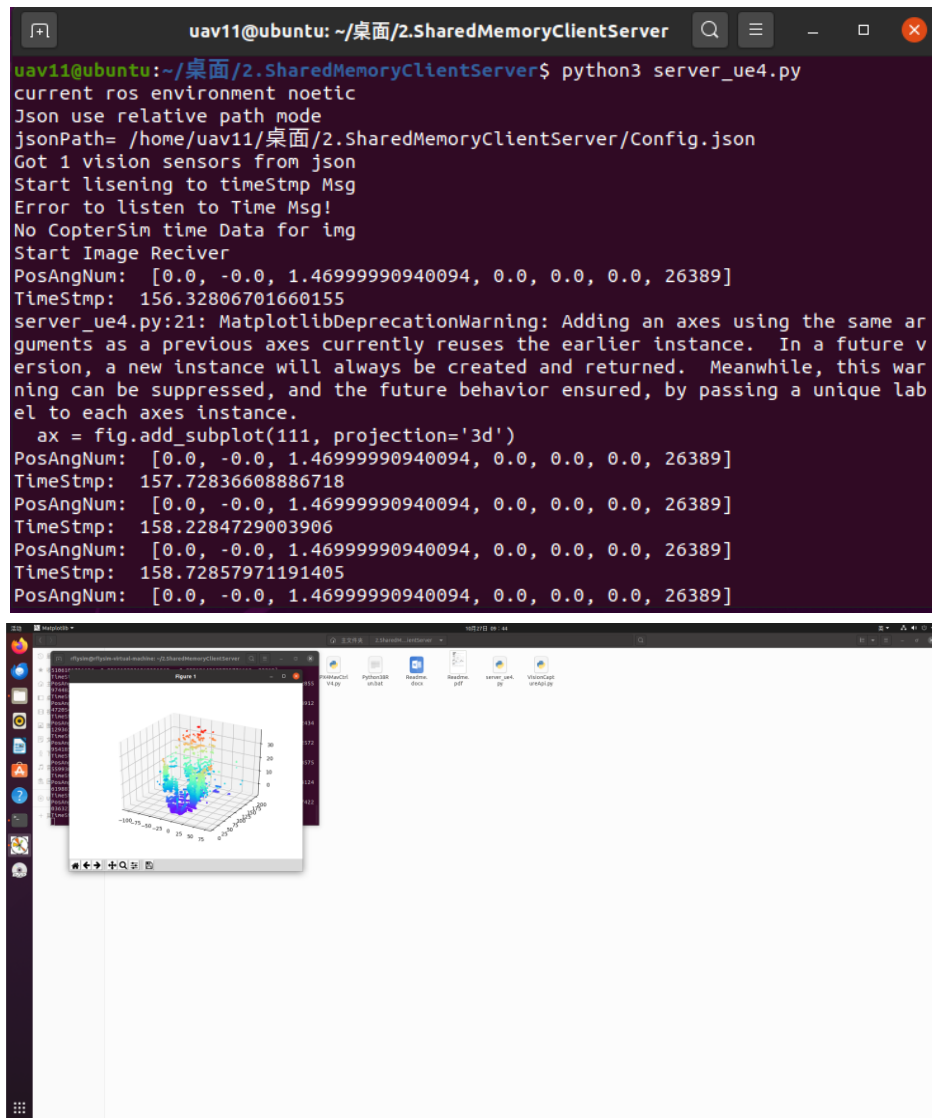
started roslaunch server http://ubuntu:44525/
ros_comm version 1.16.0

SUMMARY
=====
PARAMETERS
* /roscore: noetic
* /rosversion: 1.16.0

NODES

auto-starting new master
process[roscore]: started with pid [4058]
ROS_MASTER_URI=http://ubuntu:11311/

```



6. 参考资料

[1]. 无

7. 常见问题

Q1: 运行 server_ue4.py 时报错 ValueError: Unknown projection '3d'

A1: 这是因为 matplotlib 库的版本的原因,可通过添加如下语句解决

```
from mpl_toolkits.mplot3d import Axes3D
```