# 1. 实验名称及目的

**Python 场景控制接口验证实验：**在进行仿真时，Python 函数通过发送 UDP 消息给 RflySim3D，实现包括发送命令、更新无人机状态等操作。

# 2、实验原理

RflySim3D 关于 Python 的外部接口都在“UE4CtrlAPI.py”文件中定义，主要是“UE4CtrlAPI”类的场景控制接口，该类的作用是收发 UDP 消息且定义了控制 RflySim3D 场景中物体的方法。该类可将各种消息封装为 UDP 然后发送出去，同时还可以接收 RflySim3D 发送场景中的各类 UDP 消息。其构造函数如下：

```
    # constructor function
        def __init__(self, ip='127.0.0.1'):
            self.ip = ip

            self.udp_socket = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)  # Create sock
et
            self.udp_socket.setsockopt(socket.SOL_SOCKET, socket.SO_BROADCAST, 1)
            self.udp_socketUE4 = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)  # Create s
ocket
            self.udp_socketUE4.setsockopt(socket.SOL_SOCKET,socket.SO_REUSEADDR,1)
            self.inSilVect = []
            self.inReqVect = []

            self.stopFlagUE4=True
            self.CoptDataVect=[]
            self.ObjDataVect=[]
            self.CamDataVect=[]
```

self.udp_socket 用于发送广播消息，它被设置为支持广播，self.udp_socketUE4 用于接收来自 RflySim3D 的数据，它被设置为支持地址重用。本例程只关注发送消息给 RflySim3D 从而控制三维场景内物体的接口，其余接口的使用方法可在 API 文档中的外部接口文件[3]中找到。

利用“UE4CtrlAPI.py”库文件中的“UE4CtrlAPI”类可以传入发送端的端口与 IP 地址进行实例化：

```
    import UE4CtrlAPI as UE4CtrlAPI
    ue = UE4CtrlAPI.UE4CtrlAPI()
```

# 3、实验效果

本实验利用 python 接口创建了两个飞机，1 号飞机在前进，并且 2 号飞机在围绕着 1 号飞机旋转并随之移动。

图 1

# 4、文件目录

| 文件夹/文件名称 | 说明 |
|---|---|
| | |
| PythonCMDDemo.py | 此文件调用了“UE4CtrlAPI.py”中的接口 |
| PythonSendUE4Pos.py | 此文件调用了“UE4CtrlAPI.py”中的接口 |

# 5、运行环境

| 序号 | 软件要求 | 硬件要求 | |
|---|---|---|---|
| | | 名称 | 数量 |
| 1 | Windows 10 及以上版本 | 笔记本/台式电脑① | 1 |
| 2 | RflySim 平台免费版 | | |
| 3 | Python 3.8 | | |

推荐配置请见：https://doc.rflysim.com

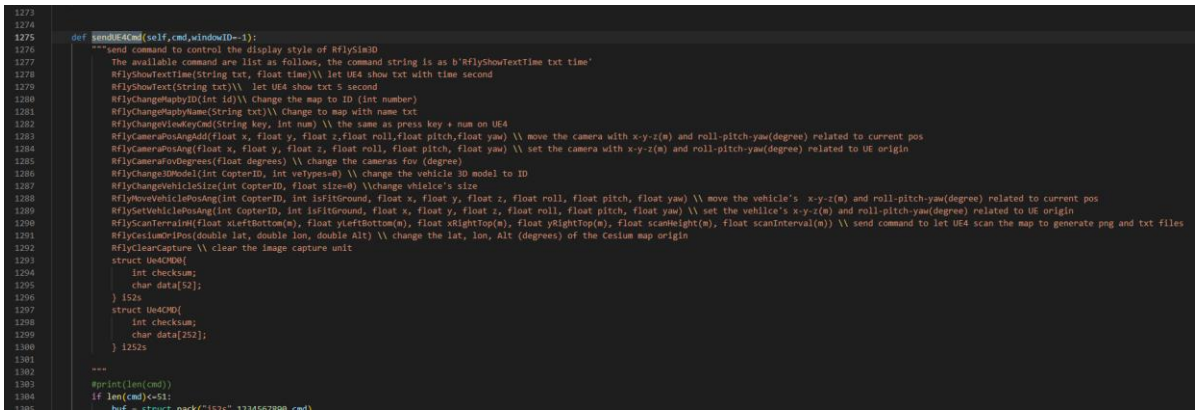
# 6、实验步骤

## Step 1: 在库文件 UE4CtrlAPI.py 中找到对应接口函数定义和用法[2]

**[3]**错误!未找到引用源。

在平台安装路径下的 ue 库文件目录 PX4PSP\RflySimAPIs\RflySimSDK\ue 找到“UE4CtrlAPI.py”文件。

其中定义了大量 RflySim 平台内置的 python 接口函数，在 UE4CtrlAPI.py 文件的 UE4CtrlAPI 类[3]下可以找到与 RflySim3D 三维场景内物体控制相关的接口，与 RflySim3D 场景内物体控制相关的函数都带着前缀“sendUE4”。

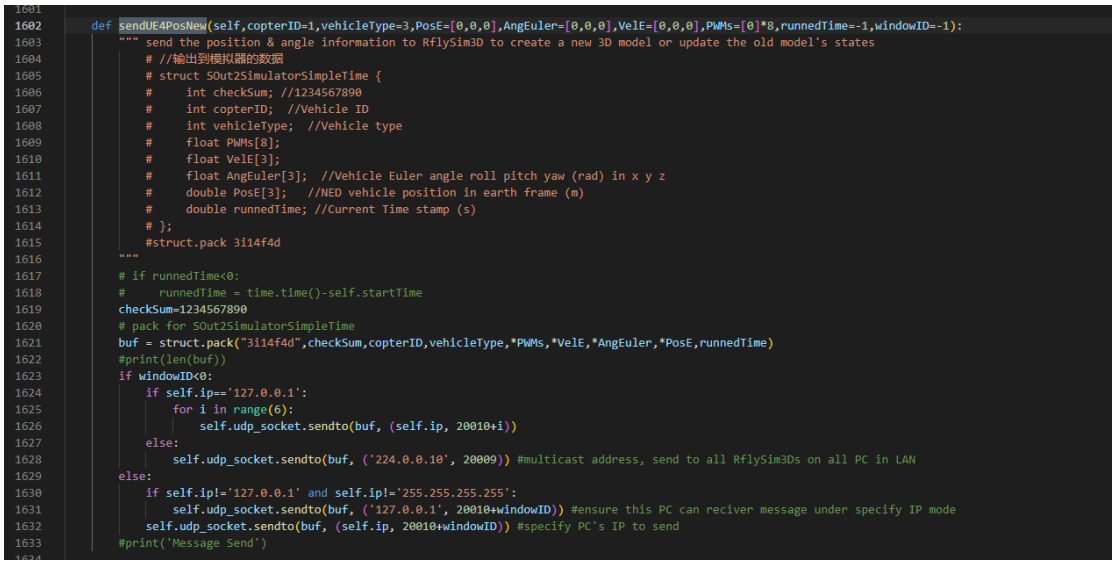


```
def sendUE4Cmd(self,cmd,windowID=-1):
    """send command to control the display style of RflySim3D
        The available command are list as follows, the command string is as b'RflyShowTextTime txt time'
        RflyShowTextTime(String txt, float time)\\ let UE4 show txt with time second
        RflyShowText(String txt)\\  let UE4 show txt 5 second
        RflyChangeMapbyID(int id)\\ Change the map to ID (int number)
        RflyChangeMapbyName(String txt)\\ Change to map with name txt
        RflyChangeViewKeyCmd(String key, int num) \\ the same as press key + num on UE4
        RflyCameraPosAngAdd(float x, float y, float z,float roll,float pitch,float yaw) \\ move the camera with x-y-z(m) and roll-pitch-yaw(degree) related to current pos
        RflyCameraPosAng(float x, float y, float z, float roll, float pitch, float yaw) \\ set the camera with x-y-z(m) and roll-pitch-yaw(degree) related to UE origin
        RflyCameraFovDegrees(float degrees) \\ change the cameras fov (degree)
        RflyChange3DModel(int CopterID, int veTypes=0) \\ change the vehicle 3D model to ID
        RflyChangeVehicleSize(int CopterID, float size=0) \\change vhielce's size
        RflyMoveVehiclePosAng(int CopterID, int isFitGround, float x, float y, float z, float roll, float pitch, float yaw) \\ move the vehicle's  x-y-z(m) and roll-pitch-yaw(degree) related to current pos
        RflySetVehiclePosAng(int CopterID, int isFitGround, float x, float y, float z, float roll, float pitch, float yaw) \\ set the vehilce's x-y-z(m) and roll-pitch-yaw(degree) related to UE origin
        RflyScanTerrainH(float xLeftBottom(m), float yLeftBottom(m), float xRightTop(m), float yRightTop(m), float scanHeight(m), float scanInterval(m)) \\ send command to let UE4 scan the map to generate png and txt files
        RflyCesiumOriPos(double lat, double lon, double Alt) \\ change the lat, lon, Alt (degrees) of the Cesium map origin
        RflyClearCapture \\ clear the image capture unit
        struct Ue4CMD0{
            int checksum;
            char data[52];
        } 152s
        struct Ue4CMD{
            int checksum;
            char data[252];
        } 1252s

    """
    #print(len(cmd))
    if len(cmd)<=51:
```

图 2 sendUE4Cmd 函数

该函数是与 RflySim3D 相关的最重要的函数，从它的大量注释可以看到我们之前介绍的 RflySim3D 命令接口[2]基本都在上面。



```
def sendUE4PosNew(self,copterID=1,vehicleType=3,PosE=[0,0,0],AngEuler=[0,0,0],VelE=[0,0,0],PWMs=[0]*8,runnedTime=-1,windowID=-1):
    """ send the position & angle information to RflySim3D to create a new 3D model or update the old model's states
        # //输出到模拟器的数据
        # struct SOut2SimulatorSimpleTime {
        #     int checkSum; //1234567890
        #     int copterID;  //Vehicle ID
        #     int vehicleType;  //Vehicle type
        #     float PWMs[8];
        #     float VelE[3];
        #     float AngEuler[3];  //Vehicle Euler angle roll pitch yaw (rad) in x y z
        #     double PosE[3];   //NED vehicle position in earth frame (m)
        #     double runnedTime; //Current Time stamp (s)
        # };
        #struct.pack 3i14f4d
    """
    # if runnedTime<0:
    #     runnedTime = time.time()-self.startTime
    checkSum=1234567890
    # pack for SOut2SimulatorSimpleTime
    buf = struct.pack("3i14f4d",checkSum,copterID,vehicleType,*PWMs,*VelE,*AngEuler,*PosE,runnedTime)
    #print(len(buf))
    if windowID<0:
        if self.ip=='127.0.0.1':
            for i in range(6):
                self.udp_socket.sendto(buf, (self.ip, 20010+i))
        else:
            self.udp_socket.sendto(buf, ('224.0.0.10', 20009)) #multicast address, send to all RflySim3Ds on all PC in LAN
    else:
        if self.ip!='127.0.0.1' and self.ip!='255.255.255.255':
            self.udp_socket.sendto(buf, ('127.0.0.1', 20010+windowID)) #ensure this PC can reciver message under specify IP mode
        self.udp_socket.sendto(buf, (self.ip, 20010+windowID)) #specify PC's IP to send
    #print('Message Send')
```

图 3 sendUE4PosNew 函数

它会发送一次无人机的数据，包括无人机的 ID、ClassID、位置、姿态、速度、各电机数据等。当 RflySim3D 收到这样的数据后，会检测无人机的 ID，如果发现该 ID 不存在于场景中，则会创建一个该飞机，如果已经存在该 ID 的飞机，则它会更新该无人机的各项数据。

## Step 2: sendUE4Cmd 函数（发送控制台命令[2][4]）

打开 RflySim3D，然后运行文档目录下“PythonCMDDemo.py”，可以看到地图被切换了：


图 4

该函数的逻辑其实就是把一个字符串“cmd”发送给 RflySim3D，如果该 cmd 是正确的“命令接口函数”则会直接调用 RflySim3D 的对应函数，产生的效果与 RflySim3D 控制台命令完全一样。

打开该文档目录下的“PythonCMDDemo.py”文件，它非常简单，只有 3 句代码。

```
import UE4CtrlAPI as UE4CtrlAPI
ue = UE4CtrlAPI.UE4CtrlAPI()
ue.sendUE4Cmd(b'RflyChangeMapbyName Grasslands')
```

第一句就是将 UE4CtrlAPI.py 文件的 UE4CtrlAPI 模块引入当前 py 程序；第二句代码表示创建了一个对象，在库文件中已包含了 RflySim 平台的通信的端口，这两句基本是 RflySim3D 的 python 脚本所必备的，不必太关心。第三句就是给 RflySim3D 发送了一个字符串“RflyChangeMapbyName Grasslands”，在 RflySim3D 控制台命令[2]中我们也介绍过了，该字符串表示调用了接口 RflyChangeMapbyName(String txt)，它的作用是将地图切换为“Grasslands”。

现在可以自由尝试 RflySim3D 控制台命令[2]中介绍的所有命令，或者把这些命令组合起来一同发送。

该函数用 UDP 发送的结构体[4]形式如下：

```
struct Ue4CMD0{
    int checksum;
    char data[52];
} i52s
struct Ue4CMD{
    int checksum;
    char data[252];
} i252s
```

其中 char 数组保存的就是发送的命令。

## Step 3: sendUE4PosNew 函数（创建模型并传入 8 位电机数据[4]）

打开 RflySim3D，再启动并运行此目录下的“PythonSendUE4Pos.py”，我们可以看见一个飞机被创建出来了：


图 5

该文件也只有三行：

```
import UE4CtrlAPI as UE4CtrlAPI
ue = UE4CtrlAPI.UE4CtrlAPI()
ue.sendUE4PosNew(10,3,[0,0,-10],[0,0,0],[0,0,0],[0,0,0,0,0,0,0,0])
```

第三句的意思是发送了一个 ID 为 10 的飞机的数据，它的 ClassID 为 3，表示我们创建的是一个四旋翼无人机，[0,0,-10]表示它距离水平面高 10 米。

该函数用 UDP 发送的结构体形式[4]如下：

```
struct SOut2SimulatorSimpleTime {
    int checkSum; //1234567890
    int copterID;  //Vehicle ID
    int vehicleType;  //Vehicle type
    float PWMs[8];
    float VelE[3];
    float AngEuler[3];  //Vehicle Euler angle roll pitch yaw (rad) in x y z
    double PosE[3];   //NED vehicle position in earth frame (m)
    double runnedTime; //Current Time stamp (s)
};
```

## Step 4:其他函数

还有许多函数，但功能大同小异，这里只简单介绍它们的功能，可以将例程稍加修改实验它们的作用。

# 7、参考资料

[1]. RflySim3D 快捷键接口总览

[2]. RflySim3D 控制台命令接口总览

[3]. UE 场景控制 python 接口总览

[4]. RflySim3D 外部交互接口数据协议

# 8、常见问题

1. 无