1. 实验名称及目的

1.1. 实验名称

python 物体飞机相机信息获取接口 getCamCoptObj 验证实验

1.2. 实验目的

通过 python 接口获取飞机、物体和相机的信息。

1.3. 关键知识点

首先调用平台的接口创建物体，然后调用接口 reqCamCoptObj(1,[TargetCopterID,100]) 发送请求到 RflySim3D，返回飞机数据，1 号和 100 号飞机。然后通过接口 reqCamCoptObj (2,targetObj2)发送请求到 RflySim3D，返回物体数据，名字为 Landscape_1 的物体。最后通过接口 mav.getCamCoptObj(1,targetObj2)获取飞机的结构体数据。其他物体、相机的数据也是相同的操作原理。关键代码解析如下：

关键知识点 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() # 向 RflySim3D 发送取图请求
vis.startImgCap() # 开启取图
vis.Img[i] # 图片 i 数据（像素矩阵）
cv2.imshow('Img'+str(i),vis.Img[i]) # 显示图片 i 图像
```

关键知识点 2：UE 控制

接口详细使用方法见：UE4CtrlAPI.py

```
ue = UE4CtrlAPI.UE4CtrlAPI() # 创建 UE 控制实例
ue.sendUE4Pos(100,30,0,[2.5,0,-8.086],[0,0,math.pi]) # 创建 3D 人物模型
ue.sendUE4PosScale(101,30,0,[10.5,0,-8.086],[0,0,math.pi],[10,10,10])
# 创建 3D 人物模型
ue.reqCamCoptObj(0,TargetCamSeqID) #发送请求到 RflySim3D，返回相机数据，0 号相机
ue.reqCamCoptObj(1,[TargetCopterID,100]) #发送请求到 RflySim3D，返回飞机数据，1 和 100 号飞机
ue.reqCamCoptObj(2,targetObj2) #发送请求到 RflySim3D，返回物体数据，名字 为 Landscape_1 的物体
ue.initUE4MsgRec() # 开启数据监听
Cam = ue.getCamCoptObj(0,TargetCamSeqID) #获取目标 1 号飞机的结构体引用
```

2. 实验效果

本实验通过 python 接口获取飞机、物体和相机的信息，详细原理见“求取标志点协议

v3.txt”。

3. 文件目录

例程目录：[安装目录]\RflySimAPIs\6.RflySimExtCtrl\0.ApiExps\e15_CamObjGet\

文件夹/文件名称	说明
GetCamObjDemo.bat	软件在环仿真实验脚本
GetCamObjDemo.py	Python 实验脚本
GetCamObjDemoWithCam.py	Python 实验脚本
VisionCaptureApi.py	取图接口
Config.json	取图配置文件
Python38Run.bat	Python 环境启动脚本

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		

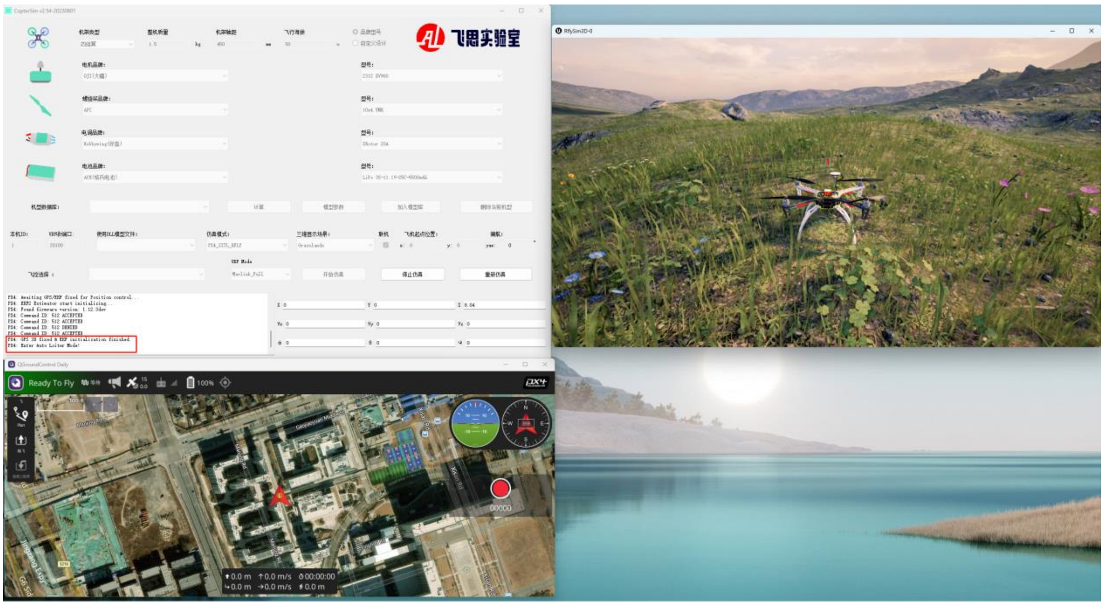
① ：推荐配置请见：https://rflysim.com/doc/zh/HowToInstall.pdf

5. 实验步骤

5.1. 必做实验：

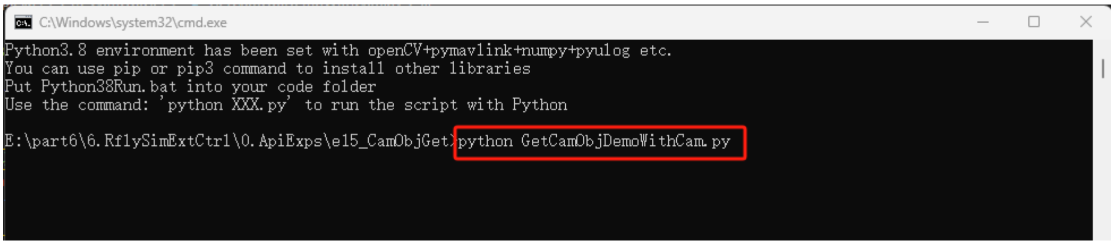
Step 1：开启仿真

以管理员方式运行 GetCamObjDemo.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：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 GetCamObjDemo.py（或 GetCamObjDemoWithCam.py，包含相机位置等信息）文件，输入 python GetCamObjDemo.py(或 python GetCamObjDemoWithCam.py)



Step 3：观察结果

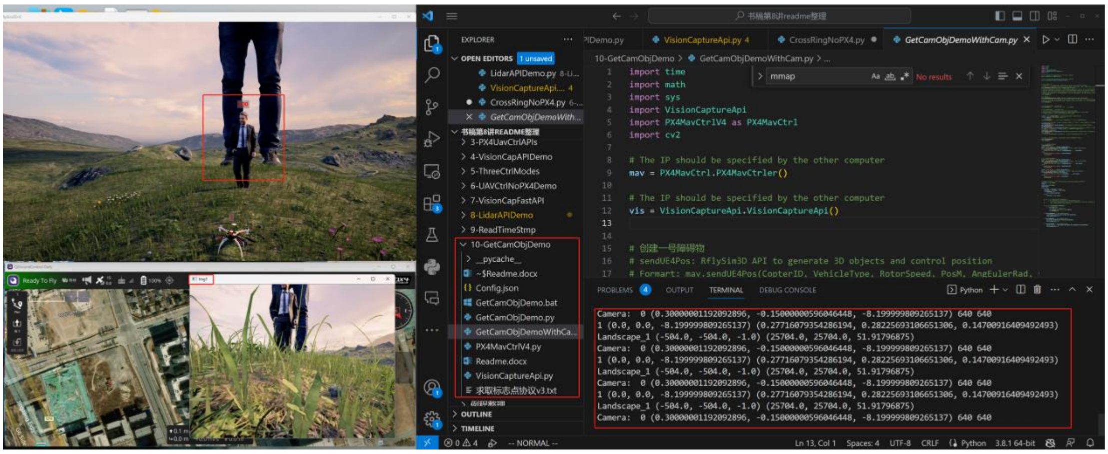
可以获取到飞机和物体的信息，以及两个摄像头窗口，创建的几个物体，如下图所示。请查看注释理解 python 实现原理。详细原理见“求取标志点协议 v3.txt”。注意：在 RflySim3D 窗口按 T 键开启或关闭飞机轨迹记录功能，T+数字*开启/更改轨迹粗细为*号。

Step 4：结束仿真

在下图“GetCamObjDemo.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（VS Code 调试运行）

- 先确保已经按 RflySimAPIs\1.RflySimIntro\2.AdvExps\e3_PythonConfig\Readme.pdf 步骤，正确配置 VS Code 环境。或者配置了自己的 Pycharm 等自定义 Python 环境。
- 其他步骤与上文相同，在 Step2 运行 GetCamObjDemo.py(或 python GetCamObjDemoWithCam.py)时，可使用 VS Code（或 Pycharm 等工具）来打开 GetCamObjDemo.py(或 python GetCamObjDemoWithCam.py)文件，并阅读代码，修改代码，调试执行等。



6. 参考资料

[1]. 求取标志点协议 v3:

一、接口介绍:

1. 请求相机、飞机、物体数据返回的接口用 UE 命令行 RflyReqObjData 来实现，可以通过 UDP 发送命令，或直接在 RflySim3D 中输入命令

2. 相机、飞机和物体的类型 ID 分别是 0 1 2

3. 发送请求数据后，在 RflySIM3D 中创建待发送相机+飞机+物体列表，以后每个 Tick 会将相机和物体结构体发布出去（注，只有位置移动时才会每帧更新，否则每 1s 更新一次）

4. 飞机+物体发布的结构体为 CoptReqData 和 ObjReqData，包含了校验位，物体位置 PosUE，物体姿态 angEuler，物体立方体原点 boxOrigin，物体立方体长宽高 BoxExtent（见后文注释）

5. 相机发布的结构体为 CameraData，包含了校验位，相机 ID 号（json 中定义的），相机位置，相机姿态等

6. 如果只需要计算某一个物体在某一个相机的像素坐标，那么就只需要请求一个相机+一个物体，并进行如下处理：

1）确定物体几何中心的位置（通常是 boxOrigin，需要与物体坐标中心区别）。很多情况下，PosUE 是以底部中心为坐标中心原点（人为选定），boxOrigin 才是物体的几何中心（随着一些舵面偏转，中心会移动，可能有所偏移）。

2）以 boxOrigin+BoxExtent+angEuler，确定八个顶点的三维坐标。

3）求八个顶点+中心点三维坐标，映射到相机二维平面的像素坐标

4）在图片中画一个框，将物体框出来。

5）在点云中画一个长方体的框，将点云分割出来。

7.如果要计算多个物体，相对多（单）个相机的相对关系与像素坐标，可重复上述步骤，方法类似。

二、发送请求数据，在 RflySIM3D 中创建待发送相机+物体+飞机列表，以后每个 Tick 会将相机和物体结构体发布出去

Void RflyReqObjData(int opFlag, FString objName, FString colorflag);

对应命令行

RflyReqObjData opFlag objName colorflag

标志位说明如下：

Opflag 操作标识符:

0-> 创建一个相机，1->创建一个飞机，2->创建一个物体

10-> 删除一个相机，11->删除一个飞机，12->删除一个物体

20->清除所有相机，21->清除所有飞机，22->清除所有物体，23->清除所有物体+飞机，24->清除所有物体、飞机+相机

ObjName 物体名字: 飞机-> CopterID 号的数字（1、2、3 格式），场景物体->双击物体确定名字（Landscape_1 格式），相机->相机的 ID 的序号（0、1、2、3 格式）

colorflag 颜色标志（将来用于生成分割图用，目前未启用）：用于设置飞机显示颜色；可以 red/white 等字符串，对应了一个颜色字符串；也可以是 int 型的纯数字，对应了颜色列表的序号；也可以是 255:0:255 这种格式的 RGB 数值。

注：colorflag 对应了分割图中显示的颜色.

三、物体和飞机信息发布协议：

```
struct CoptReqData
{ //64，发送飞机数据
	int checksum = 0; //接收端需确认 1234567891 作为校验
	int CopterID;//飞机 ID
	float PosUE[3]; //物体中心位置（人为三维建模时指定，姿态坐标轴，不一定在几何中心）
	float angEuler[3];//物体欧拉角
	float boxOrigin[3];//物体几何中心坐标（相对于 PosUE）
	float BoxExtent[3];//物体外框长宽高的一半
	double timestmp;//时间戳
};
struct ObjReqData { //96，发送物体数据
	int checksum = 0; //接收端需确认 1234567891 作为校验
	int seqID = 0;
	float PosUE[3]; //物体中心位置（人为三维建模时指定，姿态坐标轴，不一定在几何中心）
	float angEuler[3];//物体欧拉角
	float boxOrigin[3];//物体几何中心坐标（相对于 PosUE）
	float BoxExtent[3];//物体外框长宽高的一半
```

```
    double timestmp;//时间戳
    char ObjName[32] = { 0 };//碰物体的名字
};
```

PosUE 物体的位置：以设定的物体中心。

boxOrigin　　Set to the center of the actor in world space；物体的几何中心所在位置

BoxExtent　　Set to half the actor's size in 3d space；物体的半直径。

四、相机信息发布协议（主体和 json 相同）：

```
struct CameraData { //56
    int checksum = 0;//接收端需确认 1234567891 作为校验
    int SeqID; //相机序号
    int TypeID;//相机类型
    int DataHeight;//像素高
    int DataWidth;//像素宽
    float CameraFOV;//相机视场角
    float PosUE[3]; //相机中心位置
    float angEuler[3];//相机欧拉角
    double timestmp;//时间戳
};
```

注意：物体和相机的数据结构体，会议组播形式，传输给

五、所有数据包会发送到组播 IP 地址

组播地址 224.0.0.10 的 20006 端口

7. 常见问题

Q1：无

A1：无