1.实验名称及目的

1.1.实验名称

三无人机分布式控制实验

1.2.实验目的

通过三个python运行文件，使得三架飞机分布进行穿环。

1.3.关键知识点

本实验与以往的双机穿环实验不同点在于没有使用取图接口选择使用屏幕截图的方式与UE4内部共享内存传图不同的是，屏幕截图取图方式一个窗口只能一路图像，图像为最终渲染效果，在例程中首先通过sca.getHwndInfo()获取句柄的窗口信息，然后通过sca.moveWd()将窗口始终保持在最顶层，再通过sca.getCVImg()从RflySim3D窗口的工作区获取图像，再通过处理图像得到无人机速度控制命令，从而实现双机穿环。

1）PX4MavCtrlV4接口说明

```
img_bgr=sca.getCVImg(ImgInfo1) \# 获取图像

window_hwnds = sca.getWndHandls() \# 获取所有 UE4/RflySim3D 窗口的句柄

xx,yy=sca.moveWd(window_hwnds[0],0,0,True) \# 将 UE4 窗口 0移至所需位置并保持顶部

ImgInfo1 = sca.getHwndInfo(window_hwnds[0]) \# 获取第一个窗口（索引=0）的信息

cv2.moveWindow("dilated",0,yy) \#移动窗口到屏幕上的指定位置，(0，yy）窗口左上角的坐标

cv2.imshow('Img'+str(i),vis.Img[i]) \# 显示图片i图像
```

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和1的两个视觉传感器

"TypeID":1：传感器类型为RGB彩色图像

"TargetCopter":1：相机绑定在1号飞机上

"SendProtocol":[0,0,0,0,0,0,0,0]：传输模式为0共享内存机制，因此本例程只能运行在Windows环境下。

"SensorPosXYZ":[0.3,-0.15,0]和"SensorPosXYZ":[0.3,0.15,0]：两个RGB相机一左一右分布。
```

3）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.initOffboard() \# 进入Offboard模式

mav.SendMavArm(True) \# 解锁飞控

mav.SendVelFRD(ctrl[0], ctrl[1], ctrl[2], ctrl[3]) \#设置飞机的速度，分别是x、y、z以及yaw角
```

4）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.sendUE4Cmd('RflyChangeMapbyName VisionRingBlank') \#更新地图场景，场景类型为VisionRing

ue.sendUE4Cmd('RflyChangeViewKeyCmd B 1',0) \# 将目标车辆改为copterID=1的车辆

ue.sendUE4Cmd('RflyCameraPosAng 0.3 0 0.05 0 0 0',0) \#将相机移动到与身体相关的位置[0.3,0,

ue.sendUE4Cmd('r.setres 1280x720w',0) \#发送指令，设置UE4窗口分辨率，注意本窗口仅限于显示，取

ue.sendUE4Cmd('t.MaxFPS 30',0) \#发送指令，设置UE4最大刷新频率30Hz，同时也是取图频率

ue.sendUE4Pos(100,152,0,[3,0,-2],[0,0,0]) \#创建一个物体，id号为100，类型为152，速度为0，位
```

5）其余代码说明

```
def angle_cos(p0, p1, p2) \# 计算三个点形成的向量之间的夹角的余弦值

def diagonal_check(p) \# 计算一个四边形的两条对角线的长度差异

def saturationYawRate(yaw_rate) \# 限制偏航速率在 -20.0 和 20.0 之间

def taskChange(pos_x) \# 根据当前位置 pos_x 决定当前任务

def sat(inPwm, thres=1) \# 将输入的 PWM 信号限制在 [-1, 1] 范围内

def objectDetect(task) \# 根据当前任务检测图像中的目标

def squareDetect(img_bgr, img_edge) \#检测图像中的方形，并返回其中心和对角线长度

def circleDetect(img_bgr, img_edge, img_b) \#使用霍夫变换检测图像中的圆形，并返回其中心和半径

def approachObjective() \# 控制无人机接近目标的全过程函数

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
```

2.实验效果

三架无人机分次穿环。

3.文件目录

例程目录：
[安装目录]\RflySimAPIs\8.RflySimVision\1.BasicExps\1-VisionCtrlDemos\e5_ScreenCapAPI\2-CrossRing\ThreeUAVDemo

文件夹/文件名称	说明
CrossRing3HITL.bat	硬件在环一键启动脚本
CrossRing3SITL_ThreeVehicle.bat	软件在环一键启动脚本
Python38Run	Python一键运行脚本

文件夹/文件名称	说明
CrossRing3_vehicle1.py	无人机穿环例程
CrossRing3_vehicle2.py	无人机穿环例程
CrossRing3_vehicle3.py	无人机穿环例程

4.运行环境

4.1 软件要求

Windows 10及以上版本；RflySim工具链；Visual Studio Code。

> ①：若使用Pixhawk 6X飞控，平台安装时的编译命令为：px4_fmu-v6x_default，推荐PX4固件版本为：1.12.3。其他配套飞控及编译命令请见：https://rflysim.com/doc/zh/1/Hardware.html

4.2 硬件要求

笔记本/台式电脑① 1台。

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

5.实验步骤

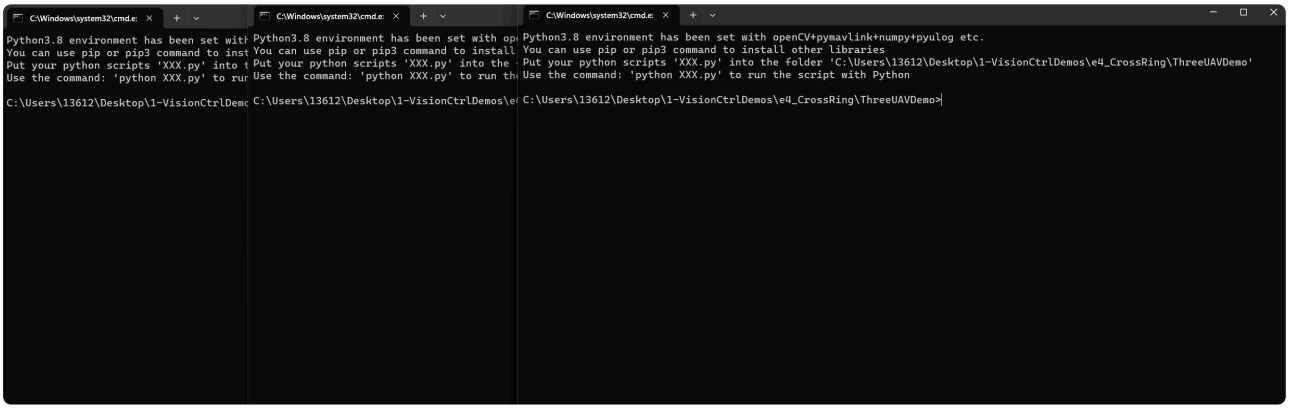
5.1.必做实验：Windows控制

Step 1：开启仿真

双击运行“CrossRing3SITL_ThreeVehicle.bat”文件开启软件在环仿真系统。也可插入飞控，并运行硬件在环仿真脚本“CrossRing3HITL.bat”，输入串口号来开启HITL仿真。

Step 2：观察结果

双击“Python38Run.bat”三次，打开三个Python环境的黑色窗口。



在第一个黑框窗口中输入下面指令，先不要回车

```
python CrossRing3_vehicle1.py
```

在第二个黑框窗口中输入下面指令，也先不要回车

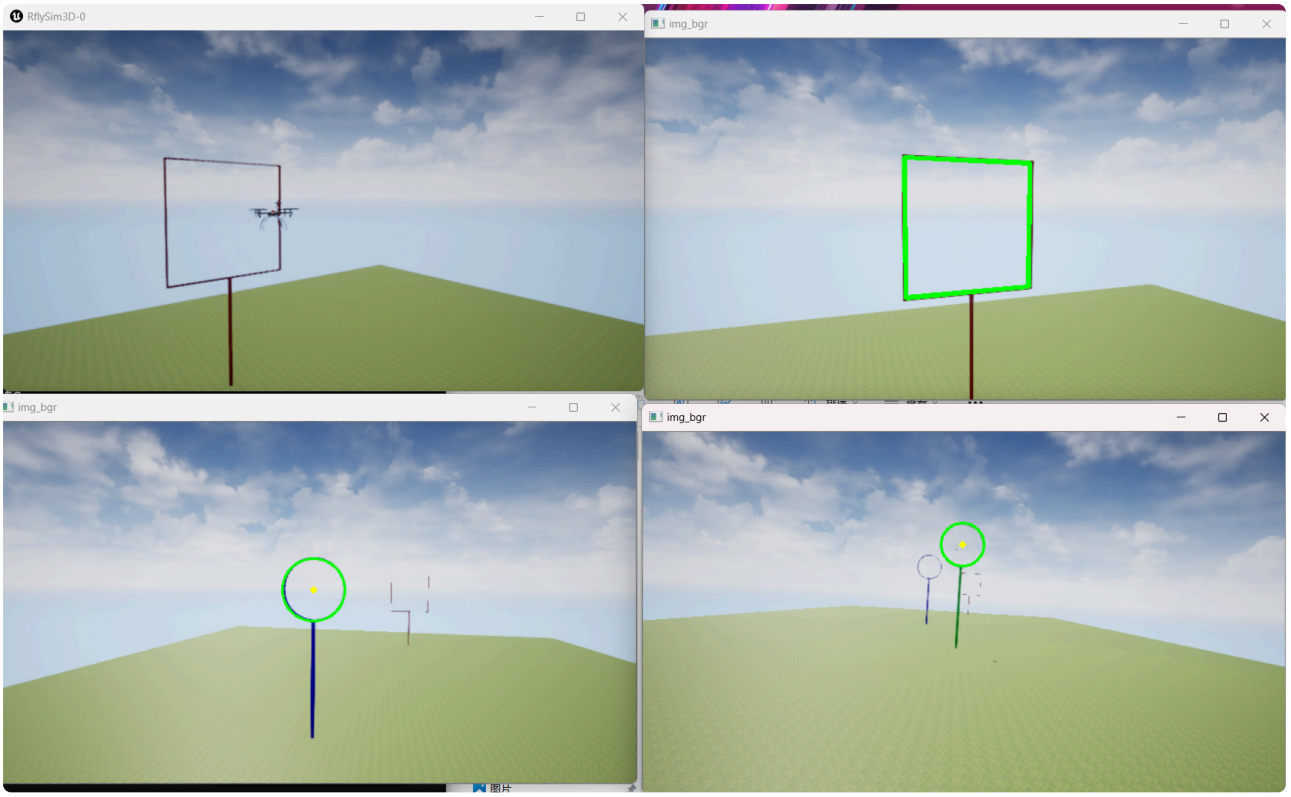
```
python CrossRing3_vehicle2.py
```

在第三个黑框窗口中输入下面指令，也先不要回车

```
python CrossRing3_vehicle3.py
```

在第一个Python黑窗口中按下回车，几秒钟后，在第二个Python黑窗口按下回车，再过几秒，在第三个Python黑窗口按下回车。

可以看到三架无人机依次起飞。



5.2.选作实验（VS Code调试运行）

准备工作：

- 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CrossRing3_vehicle1.py时，可使用VS Code（或Pycharm等工具）来打开CrossRing3_vehicle1.py文件，并阅读代码，修改代码，调试执行等。

扩展实验：

- 请自行使用VS Code阅读CrossRing3_vehicle1.py源码，通过程序跳转，了解每条代码的执行原理；再通过调试工具，验证每条指令的执行效果。

VisionCapAPIDemo.py

> RflySimAPIs > 8.RflySimVision > 0.ApiExps > 1-UsageAPI > 0.VisionSenorAPI > 1.Came

```
ue = UE4CtrlAPI.UE4CtrlAPI()

#Create a new MAVLink communication instance, UDP sending
mav = PX4MavCtrl.PX4MavCtrler(1)

# The IP should be specified by the other computer
vis = VisionCaptureApi.VisionCaptureApi()

# Send command to UE4 Window 1 to change resolution
ue.sendUE4Cmd('r.setres 1280x720w',0) # 设置UE4窗口分辨率，
ue.sendUE4Cmd('t.MaxFPS 30',0) # 设置UE4最大刷新频率，同时也
time.sleep(2)

# VisionCaptureApi 中的配置函数
vis.jsonLoad() # 加载Config.json中的传感器配置文件
```

- 请尝试修改代码，实现飞机位置改变、相机姿态角改变、相机参数改变等功能。

6.参考资料

无

7.常见问题

Q1：无

A1：无