1.实验名称及目的

1.1实验名称

自动化测试航迹跟踪。

1.2实验目的

掌握航迹跟踪的基本结构和使用流程。

1.3关键知识点

本实验在python脚本进行参数设置，通过算法进行航迹跟踪。

导入模块

导入了Python标准库模块和一些第三方库，如numpy、matplotlib、scipy等，以及自定义模块，如ProfustSA、AutoREG、AutoMavDB、AutoMavCmd、UE4CtrlAPI和PX4MavCtrlV4。

添加当前目录到系统路径

使用sys.path.append(os.getcwd())确保脚本能够导入当前目录下的模块。

UE4控制API初始化

创建UE4CtrlAPI实例，用于与UE4（Unreal Engine 4）进行交互。

定义辅助函数和类

GetPath函数用于获取仿真配置文件和启动脚本的路径。

SimAPI类用于启动和结束仿真环境。

MavAPI类继承自SimAPI，添加了无人机控制和测试相关的功能。

GenSPo类用于生成特殊轨迹（SPo）。

主要功能和方法

SimStart和SimEnd方法用于启动和结束仿真。

IninMavEnv和EndMavEnv方法用于初始化和结束无人机环境。

InitMavConf方法用于初始化无人机配置。

GetCmd方法用于从JSON配置文件中获取控制指令序列。

AutoMavRun方法是一个主循环，用于执行无人机的自动测试流程。

CMDANA方法用于分析控制指令。

calculate_flight_status方法用于计算飞行状态。

GetDesiredPV方法用于获取期望的位置和速度。

DataRecord方法用于记录测试数据。

特殊轨迹生成和测试执行

GenSPo类中的Gen_Circle_SPo方法用于生成圆形轨迹。

在__main__部分，创建GenSPo实例并生成圆形轨迹，然后创建MavAPI实例，使用生成的轨迹作为特殊轨迹进行测试。

2.实验效果

运行setpointctrl.py可以看到飞机会按照预期的航线执行。

3.文件目录

例程目录：[安装目录]\RflySimAPIs\7.RflySimPHM\2.AdvExps\e9_SetpointCtrlExp

文件夹/文件名称	说明
data文件夹	仿真数据存放处
Model文件夹	仿真用到的模型和bat脚本
setpointctrl.py	软件在环航迹跟踪运行脚本
db.json	数据库文件

文件夹/文件名称	说明
ProfustSA.py	python文件
setpointctrl.py	用于生成圆形轨迹的路径
QuadModelSITL.bat	软件在环仿真一键启动脚本
Python38Run.bat	Python程序执行脚本。

4.运行环境

4.1 软件要求

Windows 10及以上版本；RflySim平台免费版及以上；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/

5.实验步骤

5.1. 必做实验

Step 1：查看python文件

打开setpointctrl.py文件，可以看到有三个类，分别为仿真管理类、航线管理类和飞机管理类，其中航线有三条轨迹，用户可以在此处自定义轨迹。

```python
class GenSPo:
    def __init__(self) -> None:
        pass

    def Gen_Rectangle_SPo(self, length, width, height, num_points):···

        # # 生成矩形轨迹
        # length_rect = 2
        # width_rect = 1
        # height_rect = 10
        # num_points = 1000
        # x_rect, y_rect, z_rect = SPo.Gen_Rectangle_SPo(length_rect, width_rect, height_rect, num_points)
        # SPo.Plot_Trajectory(x_rect, y_rect, z_rect, 'Rectangle Trajectory')

    def Gen_Circle_SPo(self, radius, height, num_points):···

        # 生成圆形轨迹
        # radius_circle = 8
        # height_circle = -15
        # num_points = 1000
        # x_circle, y_circle, z_circle = SPo.Gen_Circle_SPo(radius_circle, height_circle, num_points)
        # # SPo.Plot_Trajectory(x_circle, y_circle, z_circle, 'Circle Trajectory')

    def Gen_Sinewave_SPo(self, amplitude, frequency, length, height, num_points):···

        # # 生成正弦波轨迹
        # amplitude = 10
        # frequency = 0.5
        # length = 4 * np.pi
        # height_sine = 10
        # num_points = 1000
        # x_sine, y_sine, z_sine = SPo.Gen_Sinewave_SPo(amplitude, frequency, length, height_sine, num_points)
        # SPo.Plot_Trajectory(x_sine, y_sine, z_sine, 'Sine Wave Trajectory')

    def Plot_Trajectory(self, x, y, z, title):···
```

仿真管理类可用于自动化打开和关闭仿真软件。

```python
class SimAPI:
    def __init__(self, frame = 'Quadcopter', SimMode = 'SITL') -> None:···

    def SimStart(self):···

    def SimEnd(self):···
```

飞机管理类用于管理飞机的连接、执行命令、数据下载等功能，高度集成了自动化测试的核心步骤。

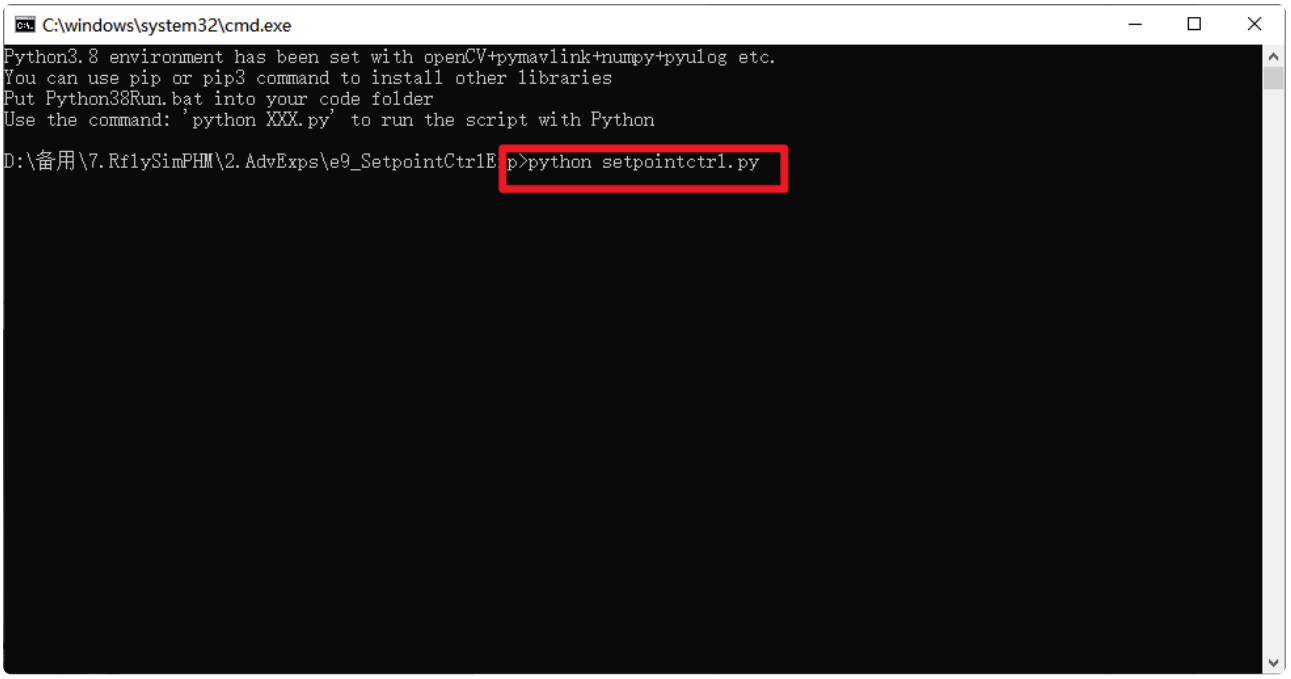
```
class MavAPI(SimAPI):
    def __init__(self, frame = 'Quadcopter', SimMode = 'SITL', ID = 1) -> None:…

    def IninMavEnv(self):…

    def EndMavEnv(self):…

    def InitMavConf(self):…

    def GetCmd(self):…

    def AutoMavRun(self):…

    def GetDesiredPV(self):…

    def CMDANA(self,cmd):…

    def calculate_flight_status(self, angular_velocity_data):…

    def Get_SPVo(self, spv_x, spv_y, spv_z):…

    def SPVoThrd(self):…

    def SPVoPub(self, ctrlseq):…

    def TRIGGERMAVCMD(self, ctrlseq):…

    def Deviation(self, desired, true):…

    def Deviation_SPVo(self, desired, true):…

    def Get_TestResult(self, test_result):…

    def DataRecord(self):…
```

Step 2：配置参数

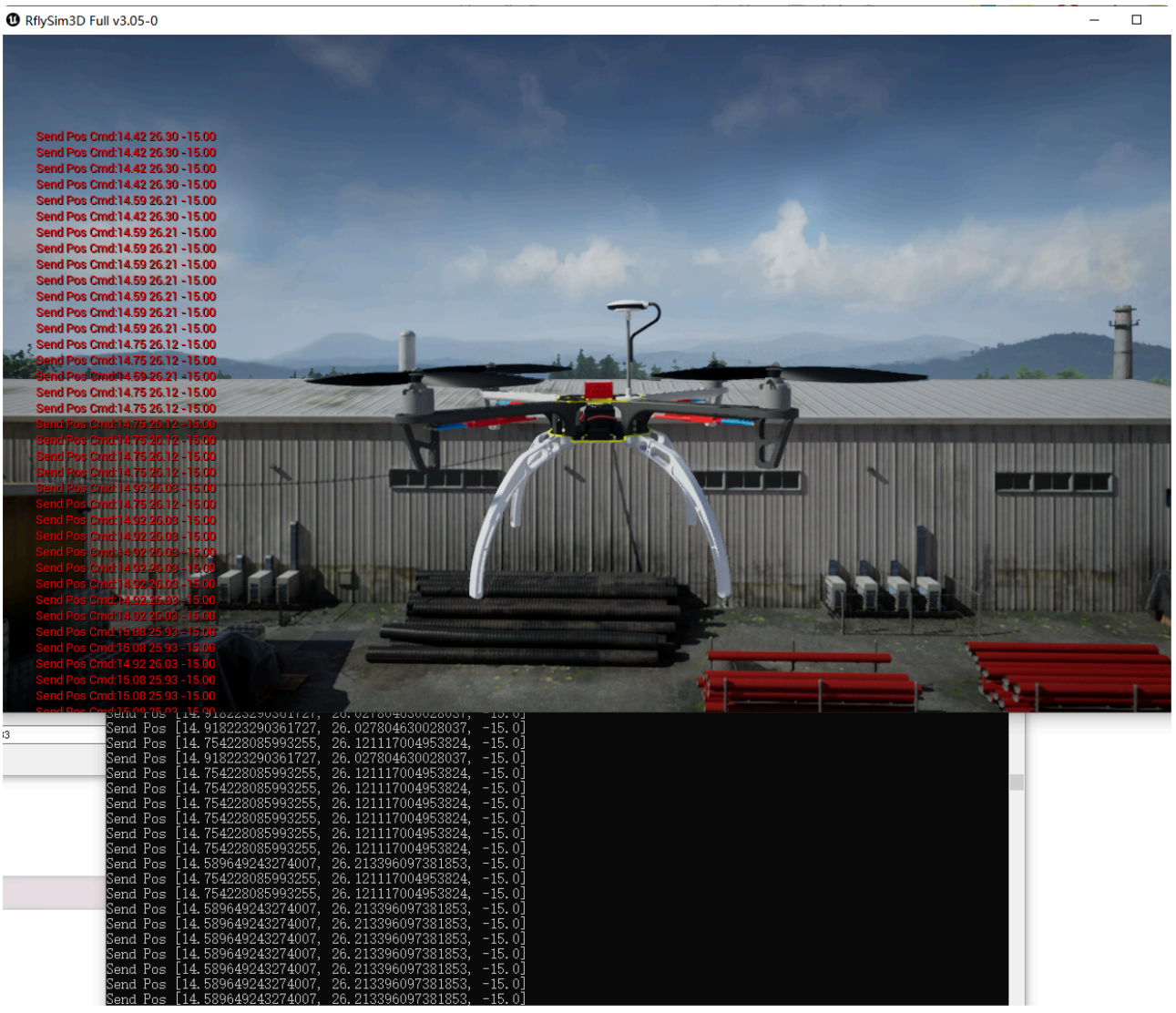
打开db.json，配置控制序列和参数。

Step 3：运行python文件

在文件夹下，双击Python38AdminRun.bat，打开集成好的python环境，在该环境下运行setpointctrl.py文件，输入
python setpointctrl.py，运行setpointctrl.py。



可以看到飞机会按照预期的航线执行。运行结束后会在该目录下生成data文件夹，存放了此次运行产生的飞行数据。



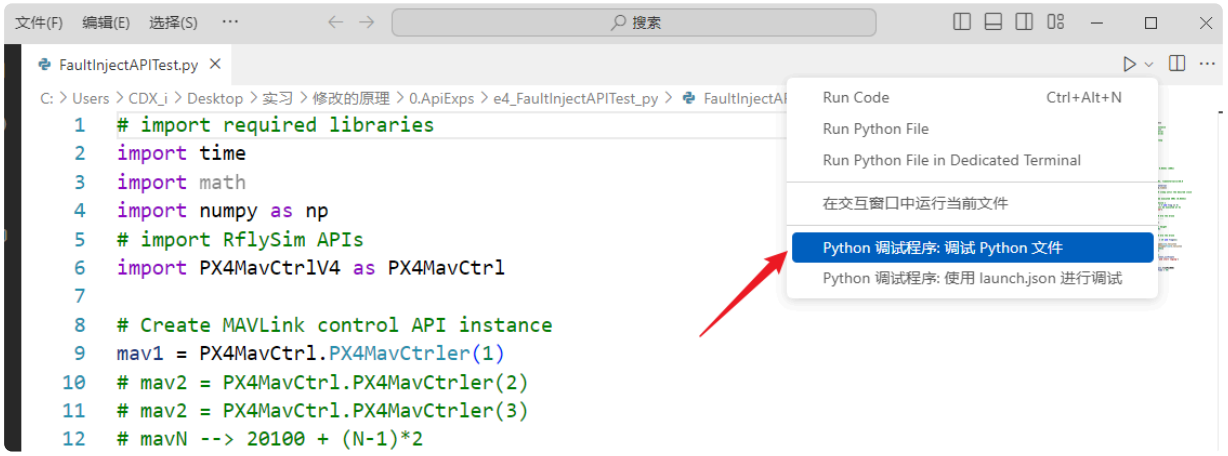
5.2. 选作实验（VS Code调试运行）

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，在运行setpointctrl.py时，可使用VS Code（或Pycharm等工具）来打开setpointctrl.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(管理员方式打开)阅读setpointctrl.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。



6.参考资料

1. 无

7.常见问题

Q1：无

A1：无