UDP模式发送故障注入参数模块的学习与使用

1. 实验目的

通过本次例程学习使用UDP模式发送故障注入参数代码。

2. 实验要求

- 软件要求：Windows 10及以上版本；RflySim工具链[1]；Visual Studio Code。
- 硬件要求：笔记本/台式电脑1台[2]。

> 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

3.实验地址

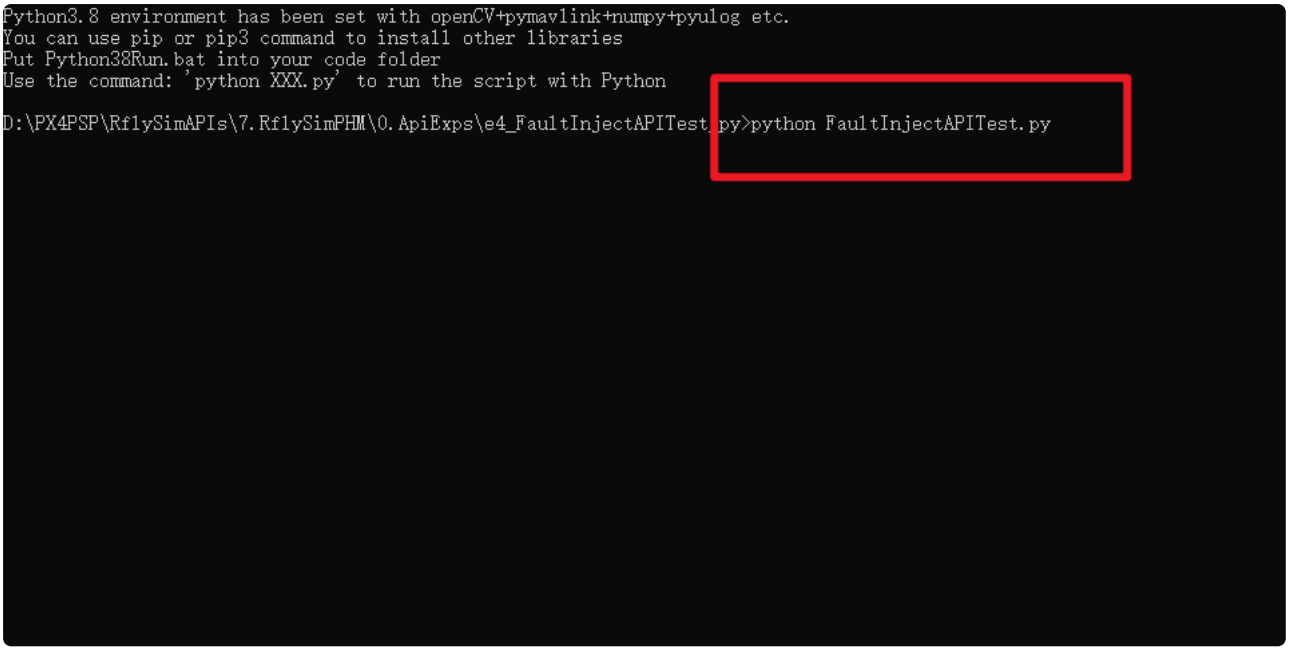
例程目录：[安装目录]\RflySimAPIs\7.RflySimPHM\0.ApiExps\e4_FaultInjectAPITest_py

- FaultInjectAPITest.py：故障注入PYTHON运行文件。
- Python38Run.bat：集成好的python运行环境。

4. 实验内容或步骤

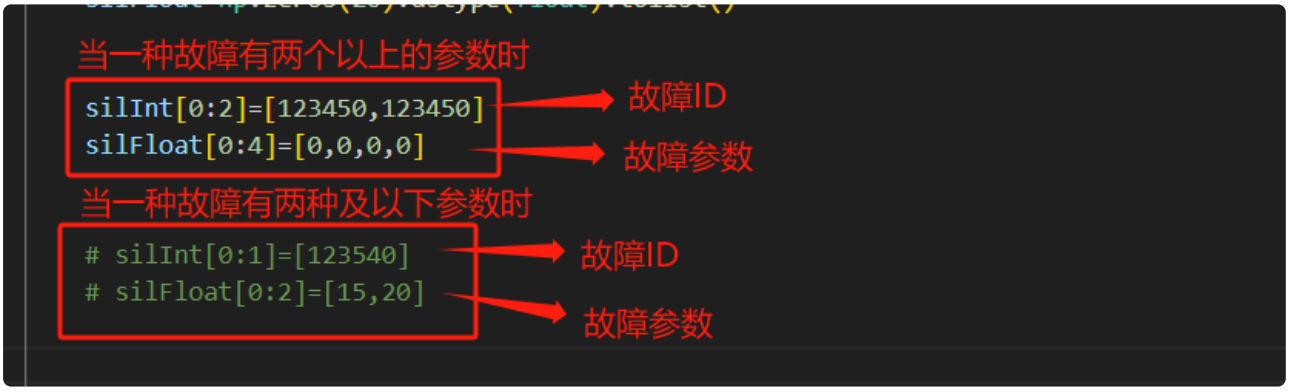
4.1 步骤1：打开文件

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FaultInjectAPITest.py文件，输入
python FaultInjectAPIT.py，运行FaultInjectAPITest.py。



4.2 步骤2：修改故障参数并运行python程序

对例程中的故障ID以及故障参数进行更改。



注：每个位置的故障ID对应下方的两个故障参数。

有两个参数以上的，如电机故障等，有两个以上的故障参数，就需要在上方注入一个相同的故障ID来进行拓展，如上图中所示，电机共有四个，就需要有两个代表电机故障的ID，同时对应下方的四个参数。

如果想要注入的故障只有一种故障参数时，在更改故障参数时，也只能更改一个参数，而故障ID对应的另一个参数只能为0，同时也不能用于下一种故障。

如传感器故障中的加速度计噪声干扰：

silInt[0:1]=[123542]

silFloat[0:2]=[15,0]

如果想要同时注入多种故障，可以在上方故障ID中输入多种故障，并在下方对应的位置注入故障参数。

如GPS故障拥有三种故障参数，如果想要全部注入，就需要在上方输入两个GPS故障ID，同时在下方对应的位置更改三个故障参数，同时第四个参数不能进行更改，必须为0，而且不能够用于后续的故障。代码便需要改为：

silInt[0:2]=[123546,123546]

silFloat[0:4]=[1,2,3,0]

在第二行进行故障参数修改时，前三个可以随意设置，但是在第四个位置上必须为0。

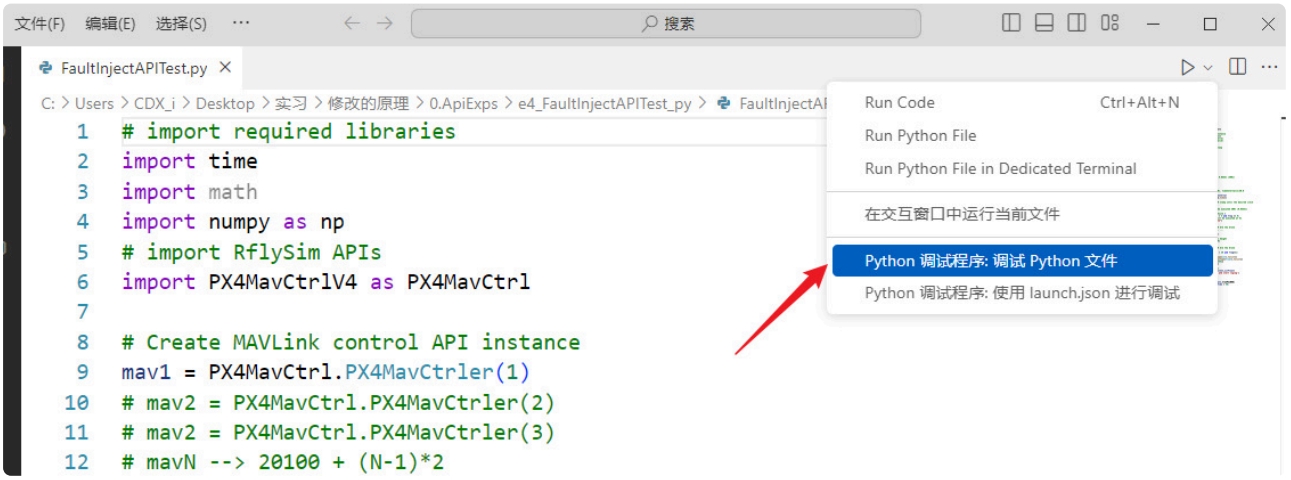
4.3 步骤3：选作实验（VS Code调试运行）

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FaultInjectAPITest.py时，可使用VS Code（或Pycharm等工具）来打开FaultInjectAPITest.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FaultInjectAPITest.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。



5. 关键知识点

5.1 关键知识点1：原理

借助平台自带的PX4MavCtrlV4库PX4MavCtrler类下的sendSILIntFloat函数将故障信息(silInt,silFloat)注入仿真。（引用：\PX4PSP\RflySimSDK\ctrl\PX4MavCtrlV4.py及其html）

程序主要流程：

1. 导入所需库：导入了时间、数学、NumPy等标准库，以及用于无人机控制的PX4MavCtrlV4库。
    i. 创建无人机控制API实例：使用PX4MavCtrl.PX4MavCtrler(1)创建一个无人机控制器的实例，编号为1。
    ii. 初始化MAVLink数据接收循环：调用mav1.InitMavLoop()初始化无人机的数据接收循环，以接收无人机的状态信息。
    iii. 初始化真实数据循环：调用mav1.InitTrueDataLoop()初始化接收无人机真实飞行数据的循环。
    iv. 初始化非板载控制模式：使用mav1.initOffboard()初始化无人机的非板载控制模式，允许通过外部命令控制无人机。
    v. 设置时间间隔和循环标志：设置时间间隔为30Hz，即每0.0333秒执行一次循环体内的代码。
    vi. 主循环：使用一个无限循环来持续发送命令和接收数据。
    vii. 执行起飞任务：当仿真时间超过5秒后，发送指令给无人机进行解锁并起飞至10米高度。
    viii. 注入故障并开始记录：当仿真时间超过20秒后，向无人机发送一组特定的整型和浮点型数据（可能用于模拟故障），然后开始记录。
    ix. 打印无人机位置信息：如果已经注入故障（flag == 2），则打印无人机的仿真位置和真实位置。
    x. 结束条件：当仿真时间超过50秒后，退出循环，结束仿真。
    xi. 清理和结束：调用mav1.endMavLoop()和mav1.EndTrueDataLoop()结束数据接收循环。

通过发送故障注入参数，接受故障注入参数，实现故障注入效果。

主要要求掌握：PX4MavCtrlV4库PX4MavCtrler类下的sendSILIntFloat函数用法。

6.参考资料

1. RflySim官方文档
2. PX4MavCtrlV4.py API参考
3. RflySim故障注入机制详解

7.常见问题

Q1：如何正确设置故障参数?

A1：每个位置的故障ID对应下方的两个故障参数。如果有两个以上参数的故障（如电机故障），需要在上方注入一个相同的故障ID来进行扩展。例如，若有四个电机需要故障注入，则需要两个代表电机故障的ID，并对应下方的四个参数。如果故障只有一个参数，更改时也只能更改一个参数，而故障ID对应的另一个参数只能为0。

Q2：如何同时注入多种故障?

A2：可以在上方故障ID数组中输入多种故障ID，并在下方对应的位置注入故障参数。例如，GPS故障有三种故障参数，如果想要全部注入，就需要在上方输入两个GPS故障ID（123546,123546），同时在下方对应位置更改三个故障参数（如：silFloat[0:4]=[1,2,3,0]）。注意第四个参数必须为0，且不能用于后续故障。

Q3：故障注入后如何验证故障是否生效?

A3：在代码中设置了时间判断，当仿真时间超过20秒后，会向无人机发送特定的整型和浮点型数据（模拟故障），然后开始记录。可以通过观察无人机行为变化或打印的无人机位置信息来确认故障是否已成功注入。还可以通过地面站软件（如QGC）监测飞行器状态变化。

---

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