1.实验名称及目的

1.1实验名称

复合型多旋翼综合模型设计及仿真验证（python外部控制）（仅限完整版及以上版本）

1.2实验目的

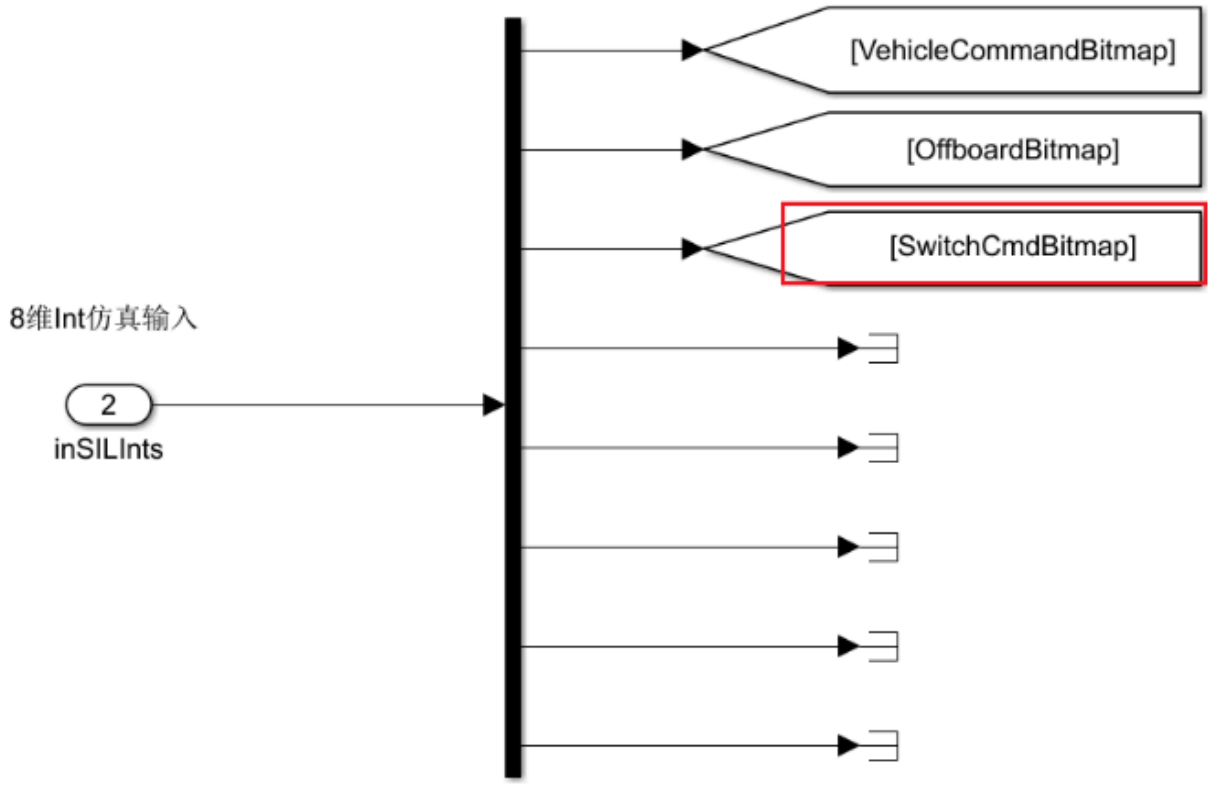
在四旋翼综合模型基础上对指定部分进行修改得到的复合型多旋翼综合模型，能同时支持四旋翼、六旋翼、八旋翼和四轴八旋翼四种类型多旋翼模型的仿真，多旋翼切换标志位由inSILInts第3位数接口接收。在该例程中，Python通过UDP30100端口向复合型多旋翼综合模型发送期望位置、期望速度等控制指令，以及多旋翼模型的切换指令。

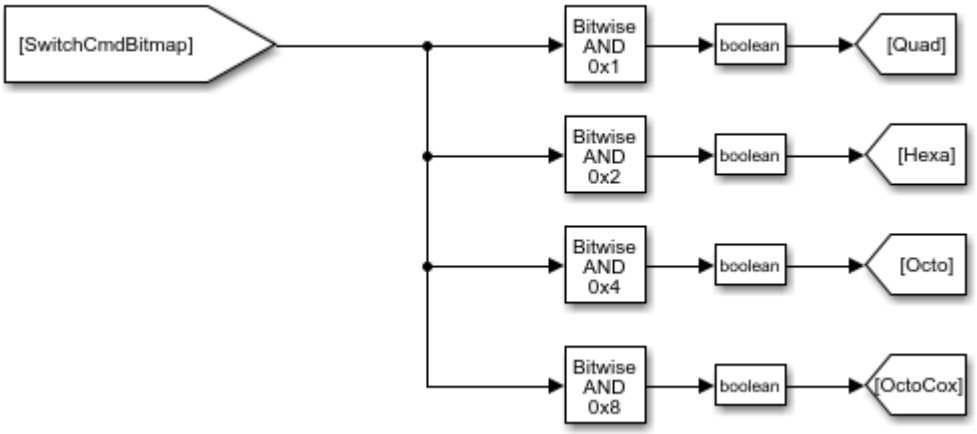
1.3关键知识点

本实验需要电脑中部署Visual Studio 2022环境，部署方式见：[安装目录]\RflySimAPIs\1.RflySimIntro\2.AdvExps\e6.VisualStudioInstall

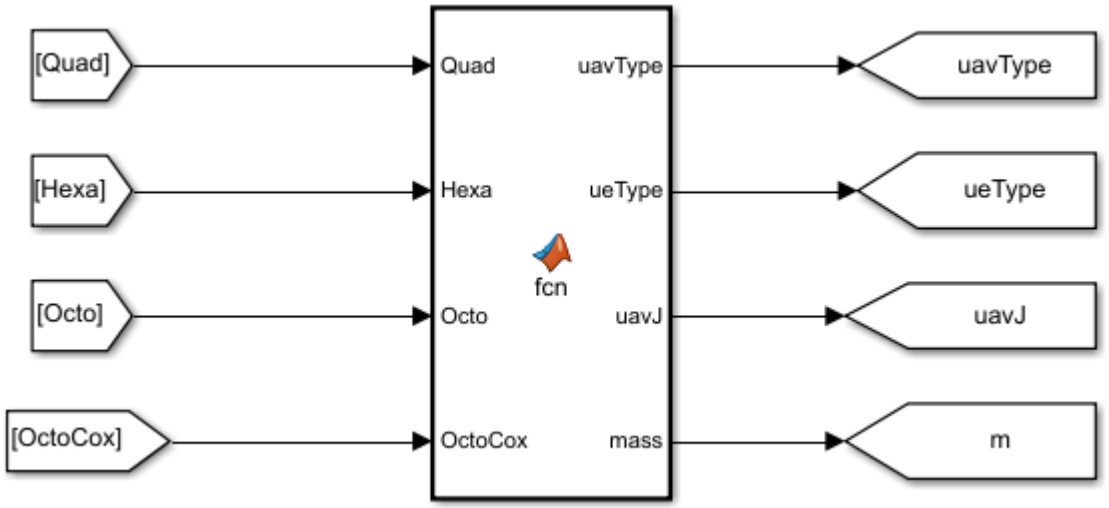
复合多旋翼综合模型新增了一些参数及模块，是结合了平台所支持的所有多旋翼类型的综合模型，不同的多旋翼之间有着电机数量，载具质量，转动惯量，力和力矩分配方式，三维显示和控制器这些模块的差异，本复合多旋翼综合模型Python通过sendSILIntFloat接口向UDP30100端口动态发送指定结构体来修改参数来切换标志位统一实现上述差异模块的切换，从而实现复合多旋翼综合模型在四旋翼、六旋翼、八旋翼和四轴八旋翼之间的切换。

设置8维Int仿真输入inSILInts的第3位用于模型切换，使用其低4位代表不同的机型，并用Bitwise
Operator模块对其进行位解析，转换成bool类型的值。





根据得到的4个bool类型的值确定机型，选择对应的uavType、ueType、uavJ、mass，默认为四旋翼，使用MATLAB
Function模块，部分代码如下：




```
function [uavType,ueType,uavJ,mass] = fcn(Quad,Hexa,Octo,OctoCox)
% 默认为四旋翼
 uavType = 3;
 ueType = 3;
 uavJ = [0.0241,0,0;0,0.0239,0;0,0,0.0386];
 mass = 1.515;

 if Quad && ~Hexa && ~Octo && ~OctoCox
    uavType = 3;
    ueType = 3;
    uavJ = [0.0241,0,0;0,0.0239,0;0,0,0.0386];
    mass = 1.515;
```

uavType用于控制分配、力和力矩模块，ueType确定RflySim3D中的显示样式，uavJ和mass用于6DOF模型之中。根据不同的值确定相应的机型与混控器。

2.实验效果

启动综合模型仿真后，通过python外部控制的方法实现了多旋翼起飞飞前的类型切换，起飞悬停与位置控制；并且在飞行过程中进行了一次旋翼类型的切换，证明了在该多旋翼综合模型在仿真全过程都能自由的切换多旋翼的类型。

3.文件目录

例程目录：
[安装目录]\RflySimAPIs\4.RflySimModel\3.CustExps\e5_CopterSimSILNoPX4\7.MixedMultiRotorNoPX4_Py

文件夹/文件名称	说明
Multicopter.slx	多旋翼综合模型
CopterSender.py	外部控制文件
Multicopter.bat	多旋翼综合模型启动脚本
Multicopter.dll	多旋翼综合模型动态链接库， 由Multicopter.slx自动代码生成后打包形成
GenerateModelDLLFile.p	用于将自动代码生成的C++文件封装成动态链接库

4.运行环境

4.1 软件要求

Windows 10及以上版本；RflySim工具链；MATLAB 2017b及以上。

> 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；\ \台；\ \台。

> ①：推荐配置请见： https://rflysim.com/

5.实验步骤

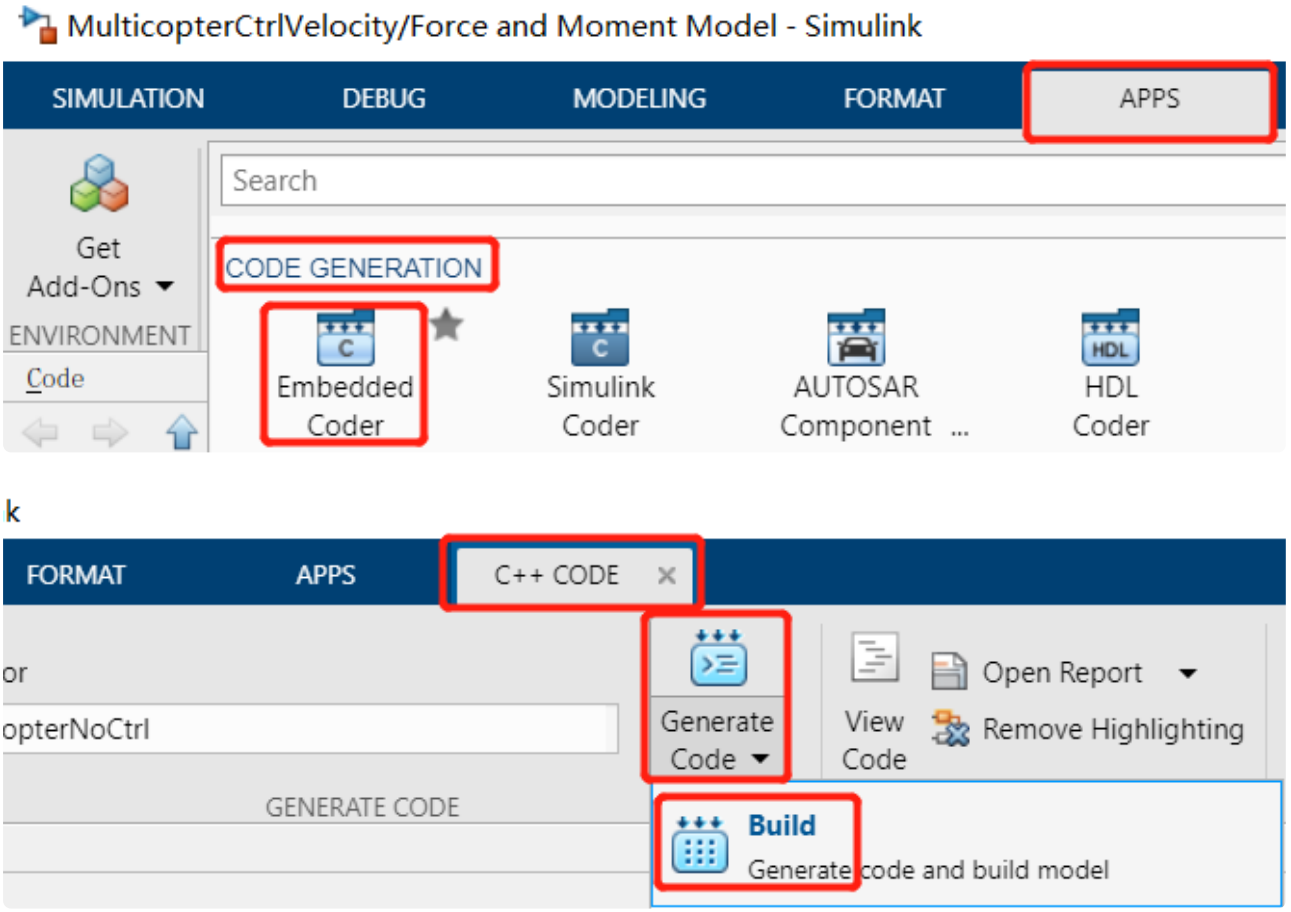
5.1. 必做实验：复合多旋翼综合模型仿真

Step 1：编译模型

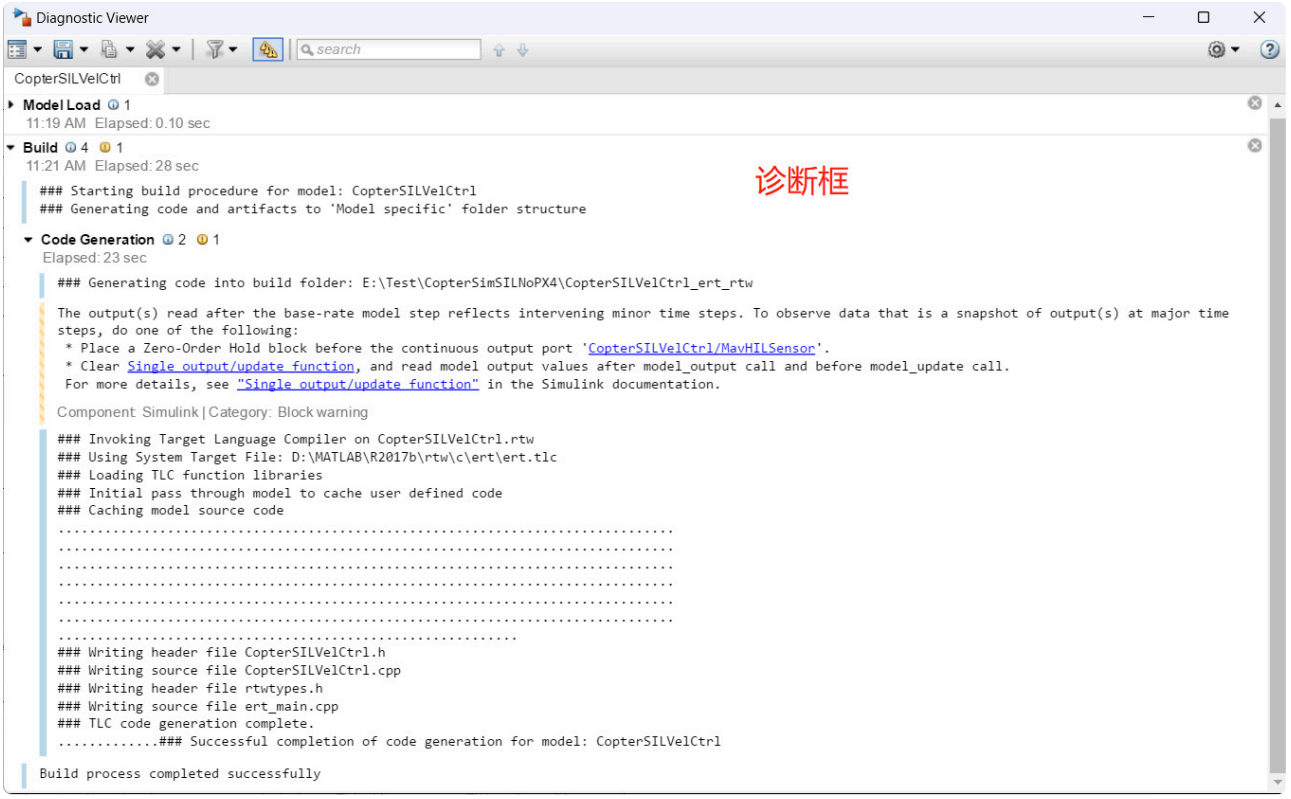
在MATLAB中打开Multicopter.slx文件，在Simulink中，点击编译命令。编译配置可参考4.RflySimModel\0.ApiExps\2.UserDefinedC++\2.GenC++\Readme.pdf

对于MATLAB 2019a及之前版本，工具栏样式见下图，直接点击它的编译按钮“Build” 即可。

对于2019b及之后版本，点击APPS - CODE GENERATION –Embedded Coder才能弹出代码生成工具栏，在其中如下图所示点击“C++CODE” - “Generate Code” - “Build”按钮就能编译生成代码。

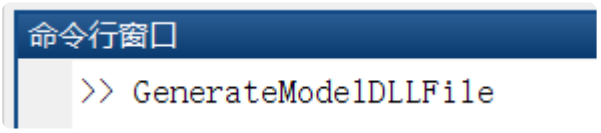


在Simulink的下方点击View diagnostics指令，即可弹出诊断对话框，可查看编译过程。在诊断框中弹出Build process completed successfully，即表示编译成功。

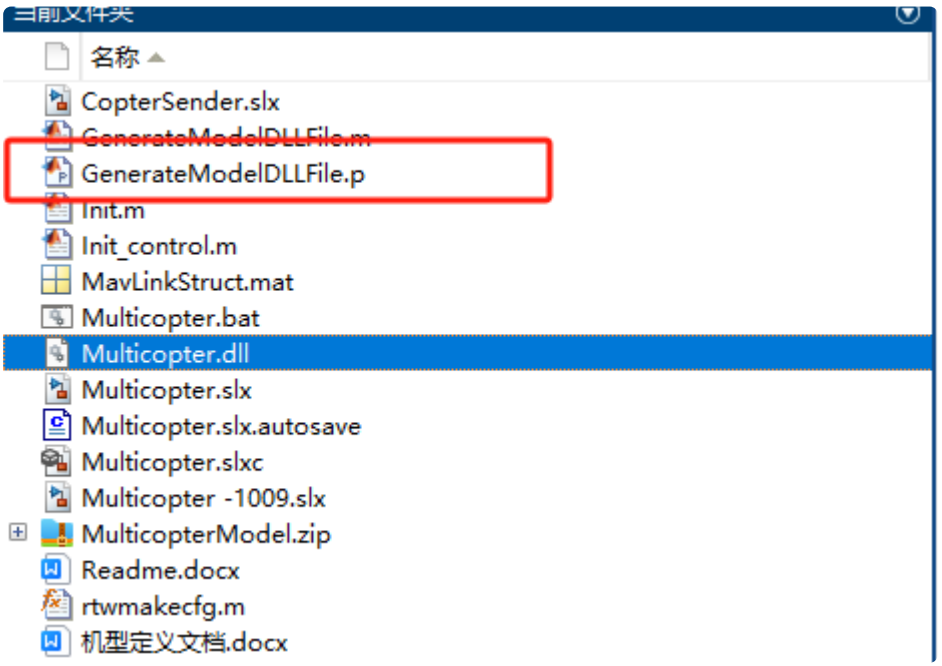


Step 2：生成DLL文件

右键运行GenerateModelDLLFile.p文件或在命令行窗口中输入GenerateModelDLLFile后回车，得到综合模型动态链接库Multicopter.dll。

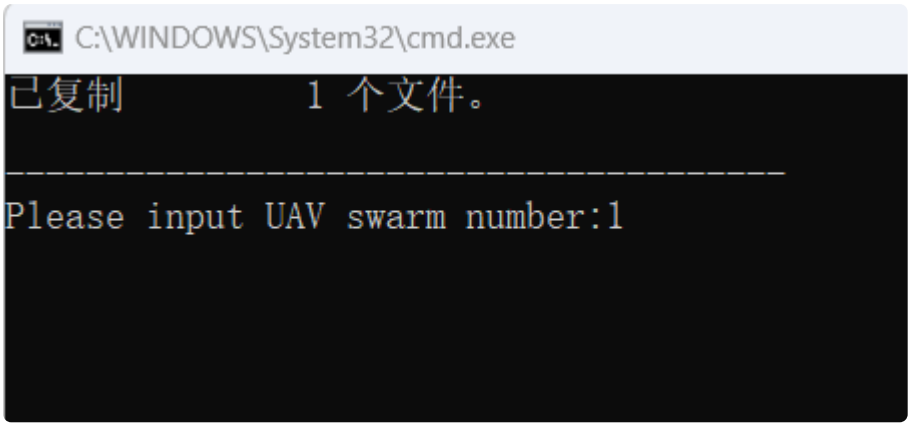


或



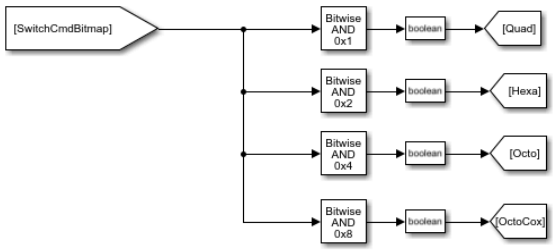
Step 3：启动仿真

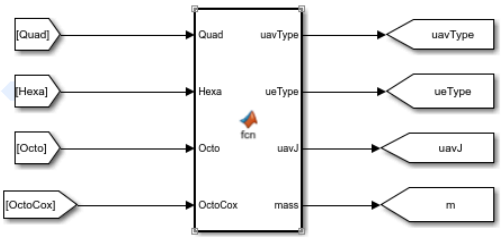
右键点击Multicopter.bat并以管理员身份运行，输入1，启动1架多旋翼综合模型的软件在环仿真。



Step 4：模型切换方法

根据多旋翼综合模型中的设定，inSILInts[2]的1、2、4、8取值分别对应了四、六、八旋翼和四轴八旋翼的综合模型，当输入为对应数字时则多旋翼综合模型会切换至对应模型进行仿真（不选择机型时，会默认为四旋翼综合模型）。

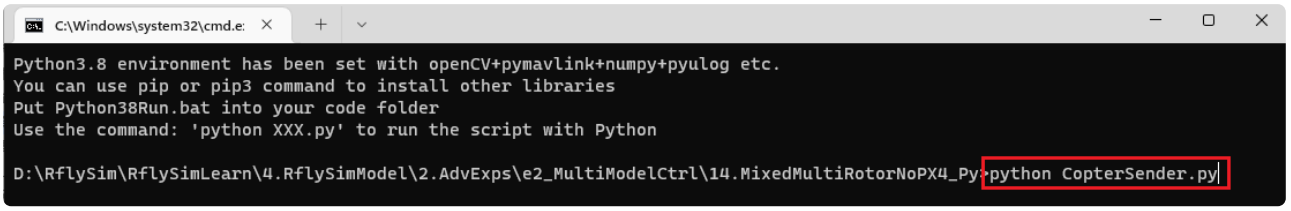




```
    # Create the mission to vehicle 1
    if flag == 0:
        # The following code will be executed at 5s
        silInt=np.zeros(8).astype(int).tolist()
        silFloat=np.zeros(20).astype(float).tolist()
        # 以四旋翼解锁起飞
        silInt[0]=263
        silInt[2]=1
        print('以四旋翼综合模型解锁起飞')
        dll.sendSILIntFloat(silInt)
        flag = 1


    if flag==1:
        silInt[0]=7
        silInt[1]=2
        silFloat[5]=-3
        dll.sendSILIntFloat(silInt,silFloat)
        print('综合模型固定以3m/s速度上升')
        flag=2


    if time.time() - startTime > 7 and flag==2:
        silInt[2]=2
        dll.sendSILIntFloat(silInt,silFloat)
        print('将综合模型切换至六旋翼')
        flag = 3

    if time.time() - startTime > 12 and flag==3:
        silInt[2]=4
        dll.sendSILIntFloat(silInt,silFloat)
        print('将综合模型切换至八旋翼')
        flag = 4

    if time.time() - startTime > 17 and flag==4:
        silInt[2]=8
        dll.sendSILIntFloat(silInt,silFloat)
        print('将综合模型切换至四轴八旋翼')
        flag = 5
```

Step 5：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CopterSender.py文件，输入
python CopterSender.py

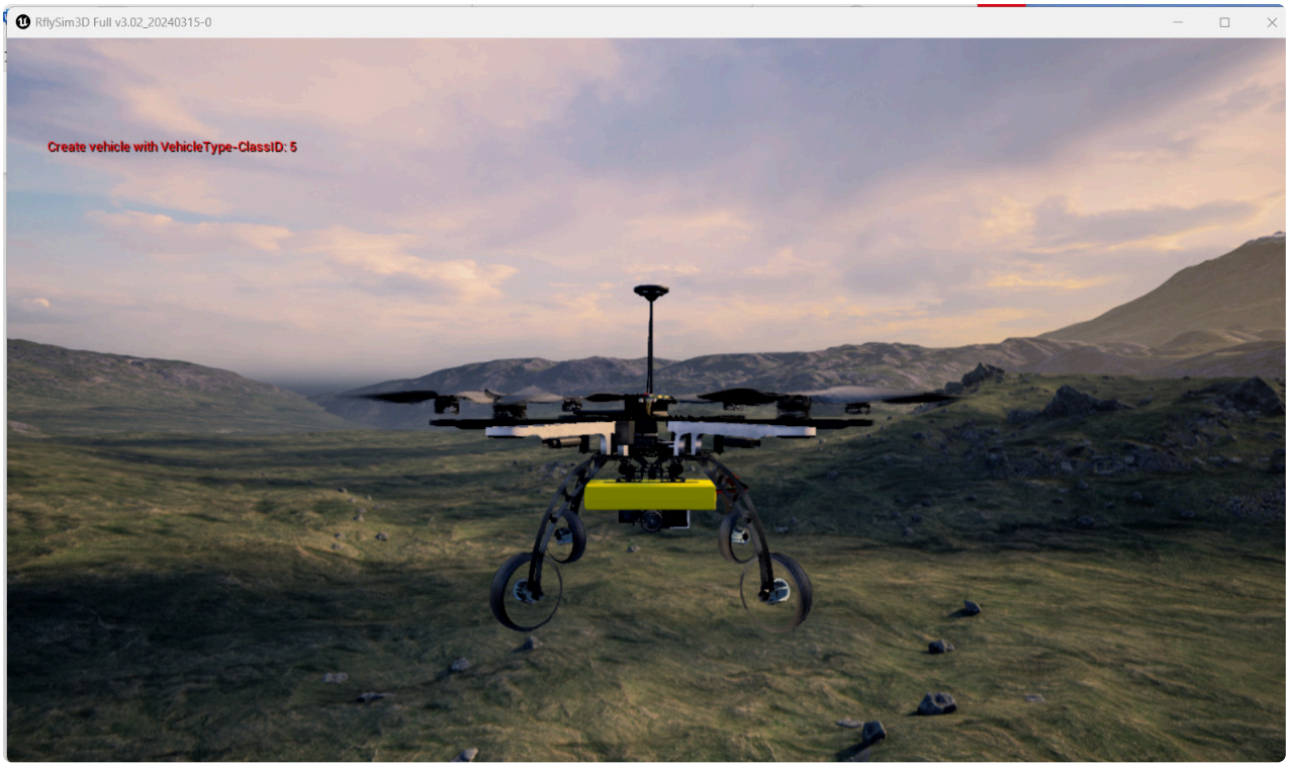


Step 6：观察结果

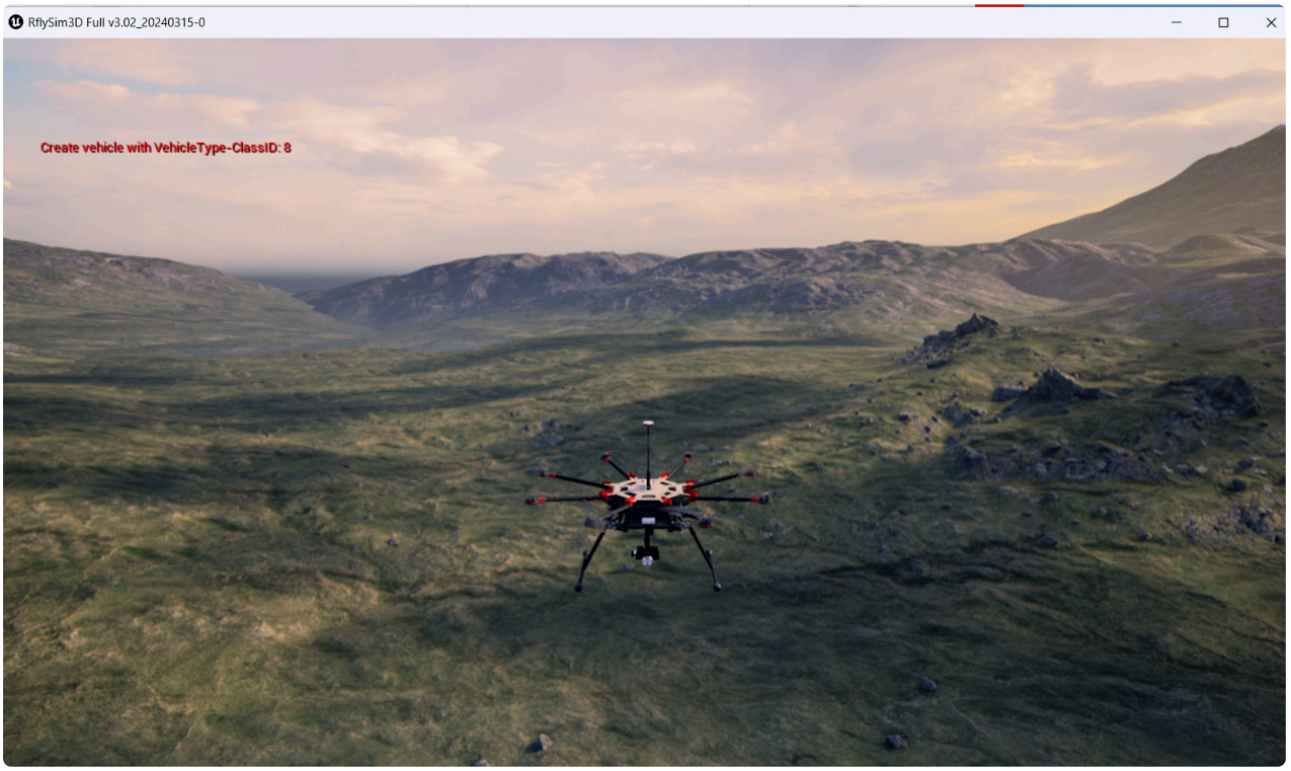
运行后可观察到以下现象：

1、四旋翼综合模型以z方向3m/s的速度向上飞行。

2、综合模型切换为六旋翼，切换完成后依然按照z方向3m/s的速度飞行。



3、综合模型切换为八旋翼，切换完成后依然按照z方向3m/s的速度飞行。



4、综合模型切换为四轴八旋翼，切换完成后依然按照z方向3m/s的速度飞行。

5.2. 选做实验（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，运行CopterSender.py时，可使用VS Code（或Pycharm等工具）来打开CopterSender.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CopterSender.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。

```python
if flag == 0:
    # The following code will be executed at 5s
    silInt=np.zeros(8).astype(int).tolist()
    silFloat=np.zeros(20).astype(float).tolist()
    # 以四旋翼解锁起飞
    silInt[0]=263
    silInt[2]=1
    print('以四旋翼综合模型解锁起飞')
    dll.sendSILIntFloat(silInt)
    flag = 1


if flag==1:
    silInt[0]=7
    silInt[1]=2
    silFloat[5]=-3
    dll.sendSILIntFloat(silInt,silFloat)
    print('综合模型固定以3m/s速度上升')
    flag=2


if time.time() - startTime > 7 and flag==2:
    silInt[2]=2
    dll.sendSILIntFloat(silInt,silFloat)
    print('将综合模型切换至六旋翼')
    flag = 3

if time.time() - startTime > 12 and flag==3:
    silInt[2]=4
    dll.sendSILIntFloat(silInt,silFloat)
    print('将综合模型切换至八旋翼')
    flag = 4
```

6.参考资料

1. PX4PSP\RflySimAPIs\4.RflySimModel\API.pdf中DLL/SO模型与通信接口的重要参数部分。
2. PX4PSP\RflySimAPIs\4.RflySimModel\API.pdf中的环境配置
3. PX4PSP\RflySimAPIs\4.RflySimModel\API.pdf中的Simulink建模模板介绍

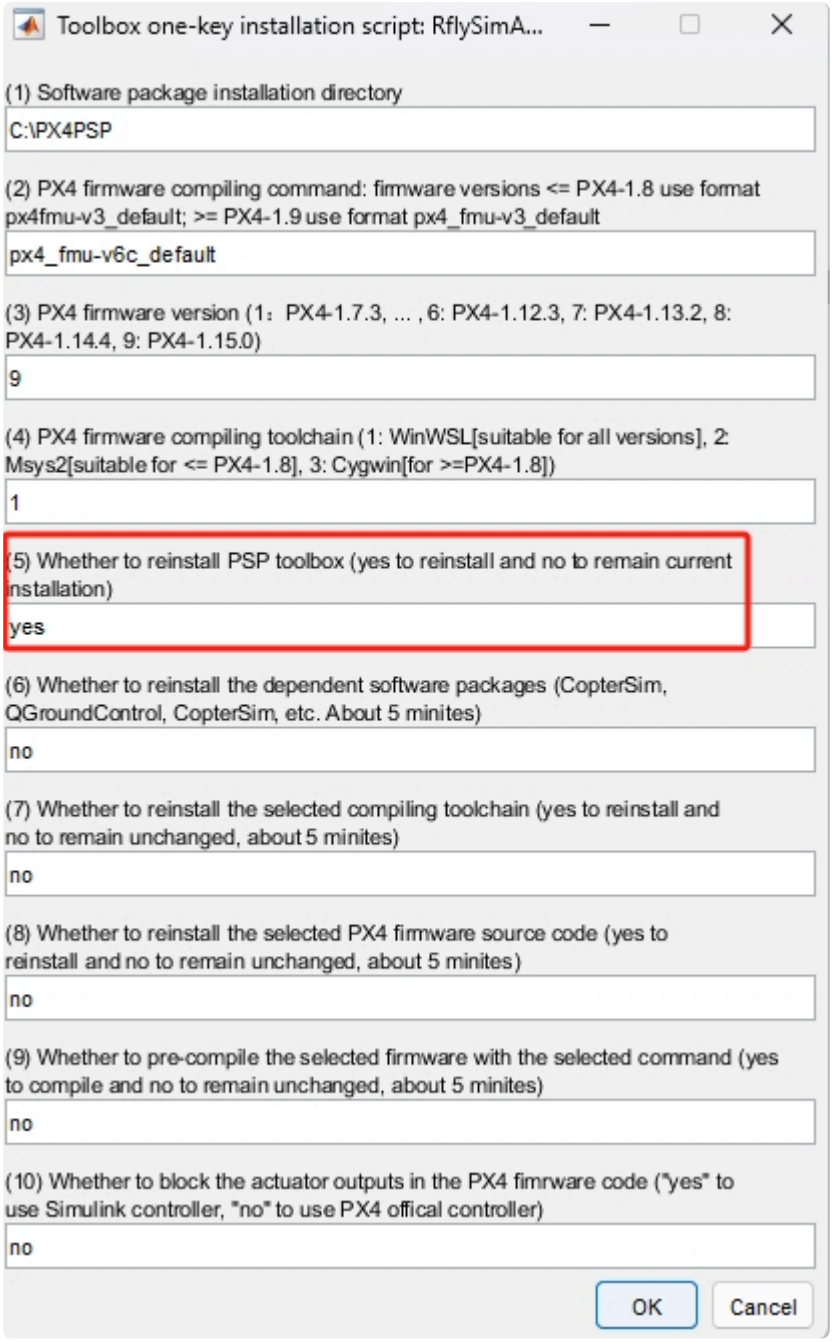
7.常见问题

Q1：

A1：

Q2：编译报错，无法加载库文件

A2：这可能是由于安装平台时PX4PSP工具箱未更新到最新版，更新RflySim安装包后按照如下配置重新安装平台即可


Toolbox one-key installation script: RflySimA...
(1) Software package installation directory
C:\PX4PSP
(2) PX4 firmware compiling command: firmware versions <= PX4-1.8 use format px4fmu-v3_default; >= PX4-1.9 use format px4_fmu-v3_default
px4_fmu-v6c_default
(3) PX4 firmware version (1：PX4-1.7.3, ... , 6: PX4-1.12.3, 7: PX4-1.13.2, 8: PX4-1.14.4, 9: PX4-1.15.0)
9
(4) PX4 firmware compiling toolchain (1: WinWSL[suitable for all versions], 2: Msys2[suitable for <= PX4-1.8], 3: Cygwin[for >=PX4-1.8])
1
(5) Whether to reinstall PSP toolbox (yes to reinstall and no to remain current installation)
yes
(6) Whether to reinstall the dependent software packages (CopterSim, QGroundControl, CopterSim, etc. About 5 minites)
no
(7) Whether to reinstall the selected compiling toolchain (yes to reinstall and no to remain unchanged, about 5 minites)
no
(8) Whether to reinstall the selected PX4 firmware source code (yes to reinstall and no to remain unchanged, about 5 minites)
no
(9) Whether to pre-compile the selected firmware with the selected command (yes to compile and no to remain unchanged, about 5 minites)
no
(10) Whether to block the actuator outputs in the PX4 fimrware code ("yes" to use Simulink controller, "no" to use PX4 offical controller)
no
OK
Cancel