

## 1. 实验名称及目的

固定翼飞机模型 DLL 生成及 SIL/HIL 实验(含碰撞检测): 在 Matlab 将 Simulink 文件编译生成固定翼的 DLL 模型文件; 并对生成的固定翼模型进行软硬件在环仿真测试, 通过本例程熟悉平台固定翼模型的使用。

## 2. 实验原理

AircraftMathworksWithCollision.slx 是基于系统模版构建的固定翼模型, 本 simulink 模型相对基于最小系统模板构建的固定翼模型, 主要增加了碰撞检测的功能模块

### 2.1. 模型参数介绍 (参考 [API.pdf 中 DLL/SO 模型与通信接口的重 要参数部分](#))

#### 1) 重要参数

AircraftMathworksWithCollision\_Init.m 中定义了固定翼模型仿真所需的常规参数, 关键数据如下。

飞机的三维显示样式

```
ModelParam_uavType = int16(100); %定义机型为固定翼, 这个参数决定了飞机的三维显示样式, 需要和 RflySim3D 的 XML 文件中的 ClassID 相匹配
```

飞机的初始位姿参数

```
ModelInit_PosE=[0,0,0]; %用于设置飞机的初始位置, 对应了 CopterSim 上的 X 和 Y 初始值。Z 值利用 TerrainZ 实现了从 CopterSim 中读取当前地形高度数据, 使得飞机可以初始化在复杂地形的地表面 (例如 Grassland 地图)。
```

```
ModelInit_AngEuler=[0,0,0]; %用于设定飞机的初始姿态。飞机姿态角的前两位 (俯仰和滚转角) 可以通过 ModelInit_AngEuler 参数来配置, 但是偏航角需要在 CopterSim 中配置。针对导弹等竖直升空的飞行器, 需要设定合适的俯仰和滚转值。
```

QGC 中显示的地图坐标和高度原点 (在 RflySim3D 的 Cesium 大场景中能任意指定飞机在地球三维场景中的坐标)

```
ModelParam_envLongitude = 116.259368300000; %经度, 初始大地经度
ModelParam_envLatitude = 40.1540302; %纬度, 初始大地纬度
ModelParam_GPSLatLong = [ModelParam_envLatitude ModelParam_envLongitude]; %初始大地经度和纬度
ModelParam_envAltitude = -50; %原点的海拔高度, 竖直向下为正, 高于海平面填负值, 单位米。
```

执行器的初始参数

```
ModelInit_Inputs = [0 0 -1 0 0 0 0 0 0 0 0 0 0 0 0 0]; %十六维输入向量, 定义电机 PWM 初始值, 默认全 0, 对固定翼和小车需要修改, 因为它们的油门在初始状态处于最小值 (-1), 见 "Motor Model" 模块
```

还需其余各传感器噪声参数

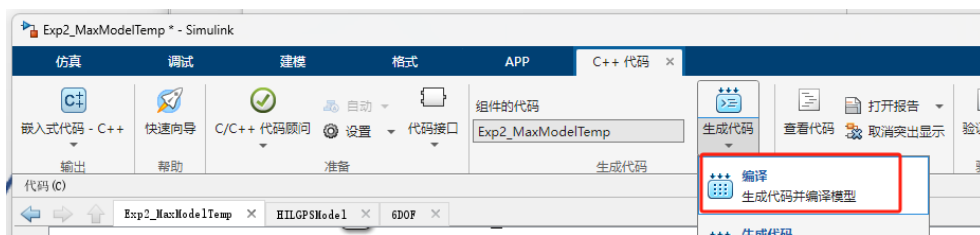
initData.m 中定义了固定翼模型仿真的详细参数 (包括机型参数、电机指令、环境参数等)































#### 2) 参数调用过程

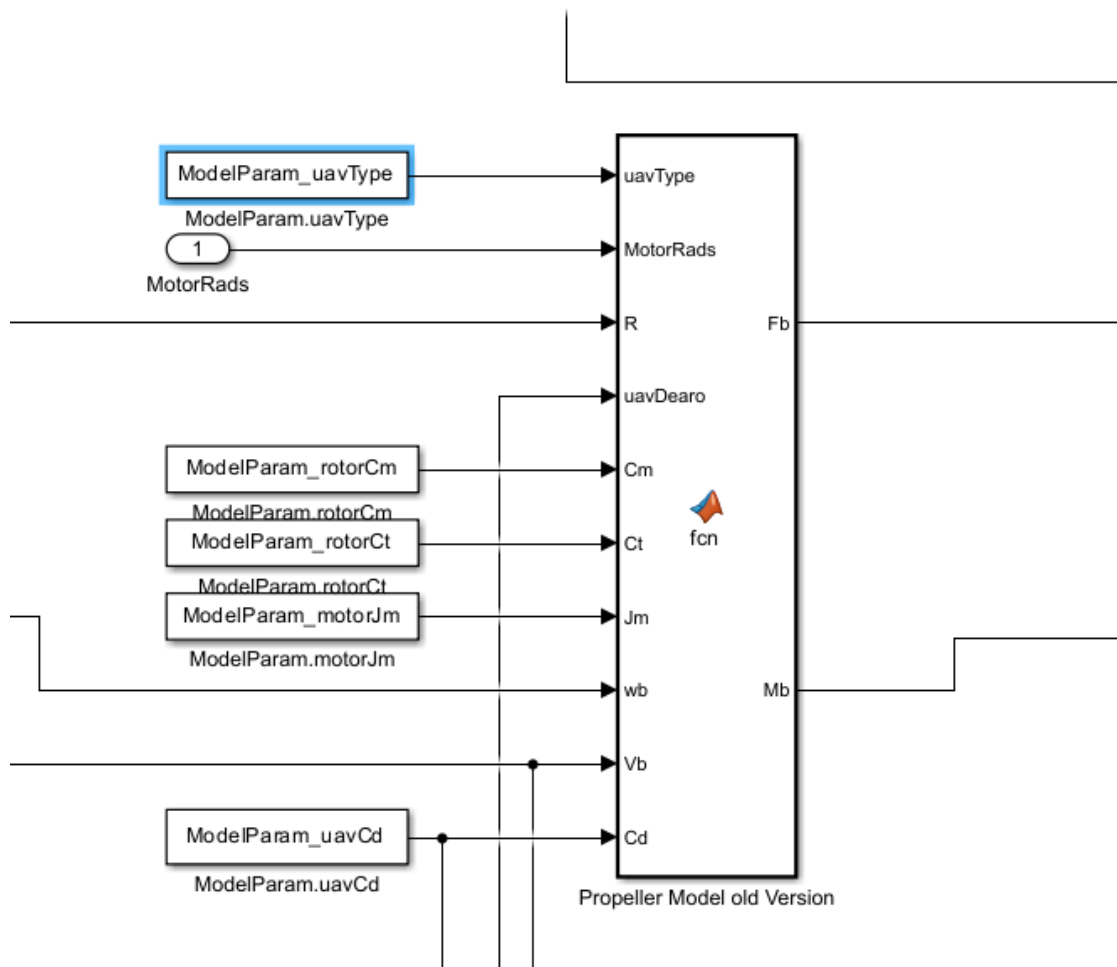
AircraftMathworksWithCollision.slx 是用于生成 DLL 模型的 simulink 动态建模模板, simulink 模型启动运行 (编译) 时会调用 AircraftMathworksWithCollision\_init.m



AircraftMathworksWithCollision \_init.m 中包含了模型的参数信息，本脚本会在 AircraftMathworksWithCollision.slx 编译（simulink 模型编译所需环境配置参考 [API.pdf 中的环境配置](#)）时被调用将参数载入 MATLAB 工作空间，也可以直接运行 AircraftMathworksWithCollision \_init.m 将参数载入工作空间。Simulink 模型会通过参数名称读取工作空间中的参数，故需要保证 simulink 模型中设置的参数名称与\*\*\*\_init.m 中的参数名称相同。



工作区		
名称 ▲	值	
 FaultParamAPI	<i>1x1 struct</i>	
 filepath	'F:\d2\4.RflySimMo...	
 HILGPS	<i>1x1 Bus</i>	
 MavLinkGPS	<i>1x1 Bus</i>	
 MavLinkSensor	<i>1x1 Bus</i>	
 MavLinkStateQuat	<i>1x1 Bus</i>	
 MavVehileInfo	<i>1x1 Bus</i>	
 ModelInit_AngEuler	[0,0,0]	
 ModelInit_Inputs	<i>1x16 double</i>	
 ModelInit_PosE	[0,0,0]	
 ModelInit_RateB	[0,0,0]	
 ModelInit_RPM	0	
 ModelInit_VelB	[0,0,0]	
 ModelParam_envAltitude	-50	
 ModelParam_GPSLatLong	[40.1540,116.2594]	
 ModelParam_motorCr	842.1000	
 ModelParam_motorJm	1.2870e-04	
 ModelParam_motorMinThr	0.0500	
 ModelParam_motorT	0.0214	
 ModelParam_motorWb	22.8300	
 ModelParam_rotorCm	2.7830e-07	
 ModelParam_rotorCt	1.6810e-05	
 ModelParam_uavCCm	[0.0035,0.0039,0.00...	
 ModelParam_uavCd	0.0550	
 ModelParam_uavDearo	0.1200	
 ModelParam_uavJ	[0.0211,0,0;0,0.0219...	
 ModelParam_uavMass	1.5150	
 ModelParam_uavMotNumbs	4	
 ModelParam_uavR	0.2250	
 ModelParam_uavType	3	



GenerateModelDLLFile.p 是将 slx 模型转化为 DLL 模型文件的脚本，使用 RflySim 平台进行载具软硬件在环仿真时，需要将 DLL(windows 下)/SO (Linux 下) 模型导入到 CopterSim，形成运动仿真模型，因此，在 Simulink 模型编译完成后，需要将模型对应的 C++ 文件打包成 DLL/SO 模型。

## 2.2. 输入信号（参考 [API.pdf 中 DLL/SO 模型与通信接口的数据协议部分](#)）

最大模板的 12 个输入数据包括电机控制量、地形数据、

### 1) 电机数据 inPwms

输入接口 inPWMs，16 维执行器控制量输入，已归一化到-1 到 1 尺度(通常电机是 0-1，舵机是-1~1)，它的数据来自飞控回传的电机控制 MAVLink 消息 mavlink\_hil\_actuator\_controls\_t 的 controls，具体定义如下：

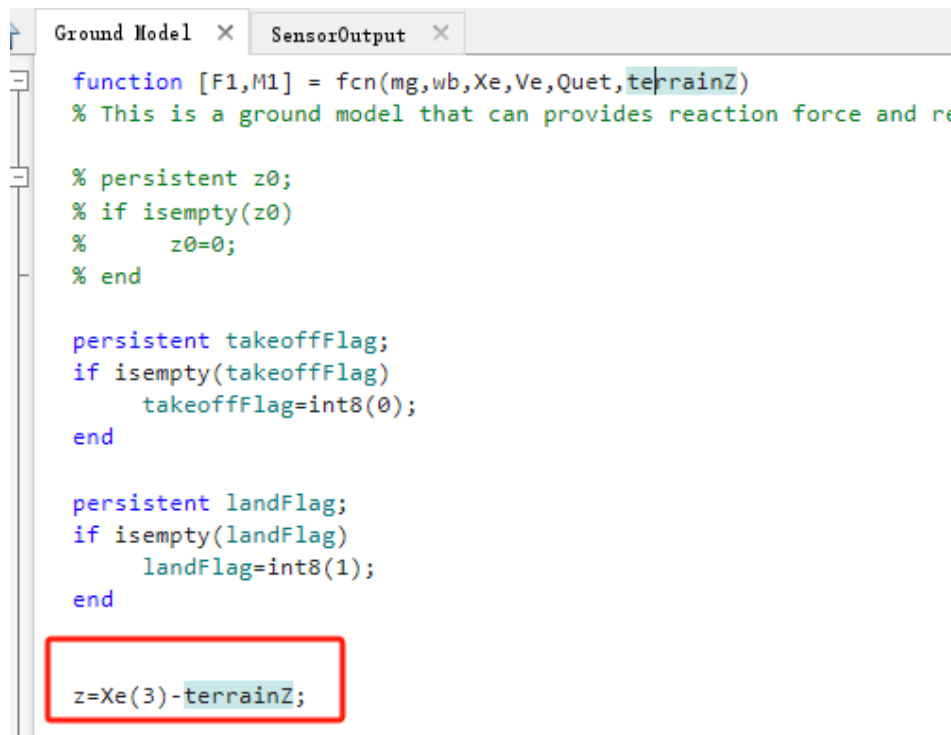
```
typedef struct __mavlink_hil_actuator_controls_t {
    uint64_t time_usec; //时间戳，从开机后的时间，单位 ms
    uint64_t flags; //标志位，用于显示当前的飞行状态
    float controls[16]; //控制量，16 维电机的控制量，发送到模型中，驱动飞机飞行
    uint8_t mode; //模型，用于显示飞机当前的飞行模式和是否上锁等信息 }
```

```
mavlink_hil_actuator_controls_t;
```

软件在环仿真时，电机控制指令从 PX4 SITL 控制器通过 TCP 4561 系列端口以 MAVLink 协议发送到运动仿真模型的 inPWMs 接口，而硬件在环仿真时，该指令是从飞控通过串口以 MAVLink 协议发送到运动仿真模型的 inPWMs 接口。

## 2) 地形高度 terrainZ

最大模板可以利用 TerrainZ 实现从 CopterSim 中读取当前地形高度数据，使得飞机初始在复杂地形的地表面（例如 RflySim3D 中的 Grassland 地图）。这个值是由 CopterSim 读取 DLL 模型初始位置参数 ModelInit\_PosE 中的 xy 坐标，根据地形校准文件及高程信息解算出地形高度 TerrainZ，通过 Mavlink 消息传输给 DLL 模型的 TerrainZ 接口，在 DLL 模型中通过 PhysicalCollisionModel/ GroundSupportModel/ Ground Model 函数中重新定义模型初始位置的高度，最后会通过 MavVehile3DInfo 接口传给 RflySim3D 中的三维显示模型。



```
function [F1,M1] = fcn(mg,wb,Xe,Ve,Quet,terrainZ)
% This is a ground model that can provides reaction force and re

% persistent z0;
% if isempty(z0)
%     z0=0;
% end

persistent takeoffFlag;
if isempty(takeoffFlag)
    takeoffFlag=int8(0);
end

persistent landFlag;
if isempty(landFlag)
    landFlag=int8(1);
end

z=Xe(3)-terrainZ;
```

## 3) 飞控状态量输入 inCopterData

inCopterData 是 32 维 double 型数据，前 8 维存储 PX4 的状态，目前 1-6 维数据，依次为：

inCopterData(1): PX4 的解锁标志位

inCopterData(2): 接收到的 RC 频道总数。当没有可用的 RC 通道时，该值应为 0。

inCopterData(3): 仿真模式标志位，0: HITL，1: SITL，2: SimNoPX4。

inCopterData(4): CoperSim 中的 3D fixed 标志位。

inCopterData(5): 来自 PX4 的 VTOL\_STATE 标志位。

inCopterData(6): 来自 PX4 的 LANDED\_STATE 标志位。

9-24 维接收 ch1-ch16 RC 通道信号（遥控器输入），25-32 维监听 rfly\_px4 uORB 消息。

#### 4) 碰撞数据接收接口 inFloatsCollision

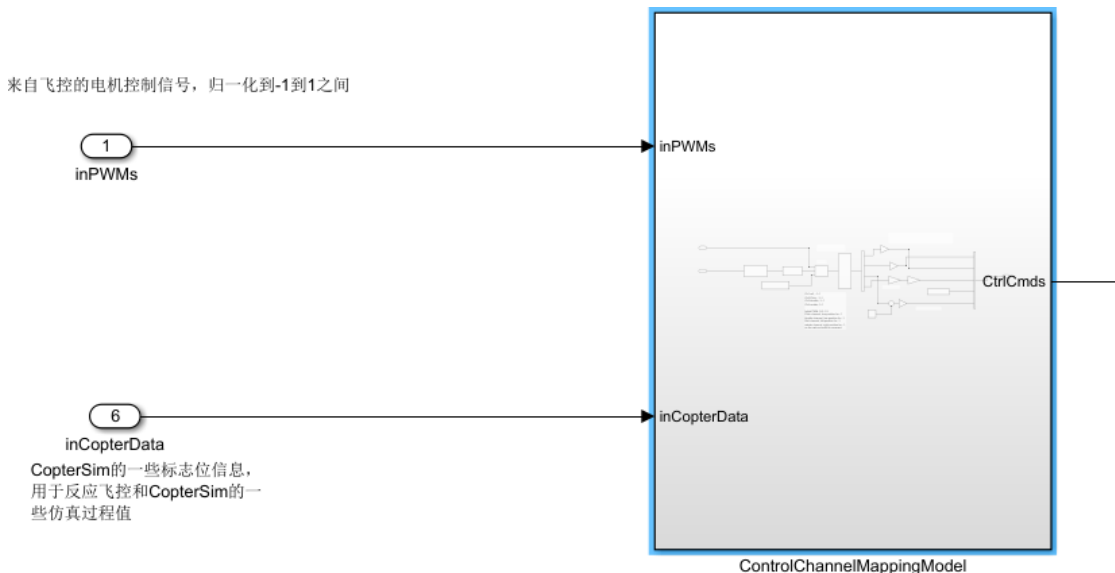
利用 inFloatsCollision 实现了一个简单地物理引擎，可以根据 RflySim3D 回传的四周距离数据，实现碰到障碍物的回弹、碰到其他飞机便坠毁等功能

### 2.3. 模型模块（参考 [API.pdf](#) 中的 [Simulink 建模模板介绍](#)）

这里为适配固定翼模型，虽然模型总的输入和输出与建模模板相同，但相对于最大模型模版，修改了模型参数（执行器初始参数等），并用其它动力学模块替换了电机模块（转速动态响应）、力和力矩模块（驱动力、气动力、地面支撑和阻力等）以及运动学模块。

#### 1) 电机模块

##### ControlChannelMappingModel 控制通道映射



针对 inPWMs 接口

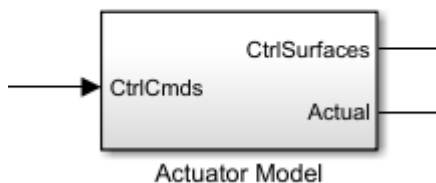
模块会提取前 4 维通道，并将-1 到 1 的 pwm 控制信号，映射为合适的范围。例如，将方向舵从-1 到 1 映射为-30 到 30 度。其他的通道也进行同样操作

针对 inCopterData 接口

会提取第 1 位数据，决定飞机是否解锁，只有飞机解锁，才让飞控的控制量传入模型

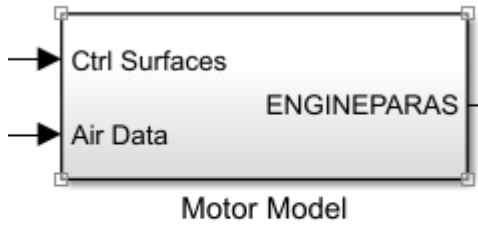
##### Actuator Model 执行器模型

执行器的动态模型，用于模拟螺旋桨、舵机的动态响应过程，这里用二阶环节来模拟



在该模块中输入为 CtrlCmds（通过 ControlChannelMappingModel 模块获取），经过各舵面的二阶线性动力学模型后得到各电机转速，该模块的输出分别为输入给力矩模型的电机转速 CtrlSurfaces（弧度每秒）；输入给 UE 的电机转速 Actual（转每分）

### Motor Model 发动机模型

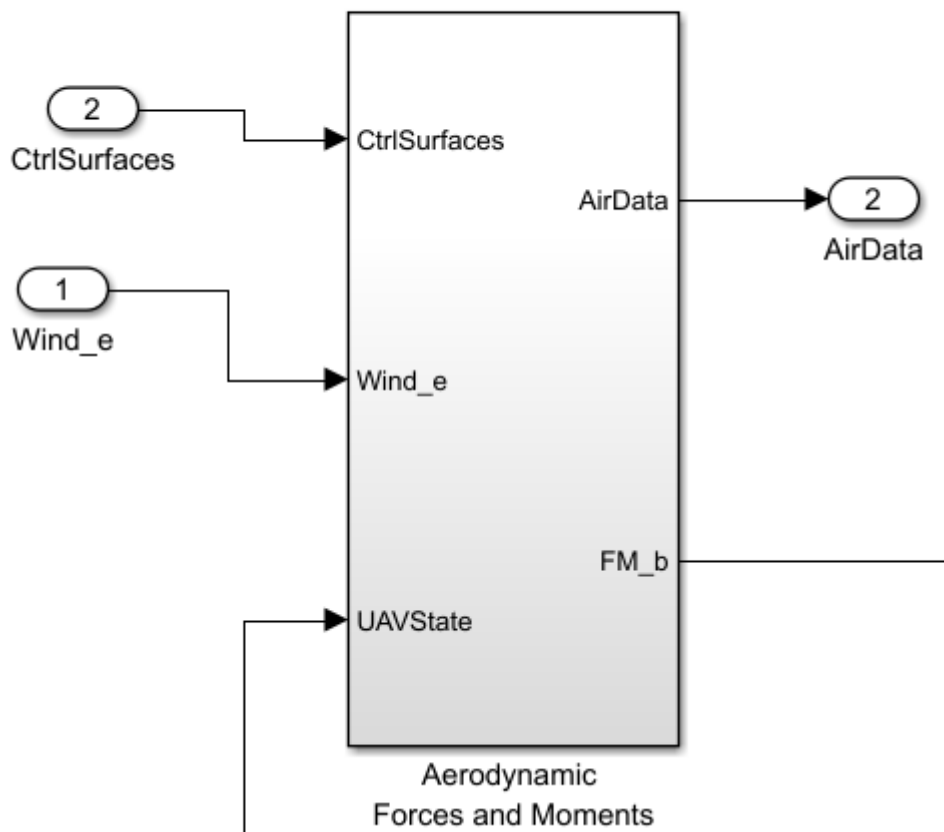


飞机发动机模型，这里加入了引擎的一些特性。包括在来流情况下，推力效率降低。

## 2) Force and Moment Model 力和力矩模块

该模块输入为电机转速 CtrlSurfaces、飞机运动学姿态 6DOF 和 wind\_e，输出为气动力和力矩。

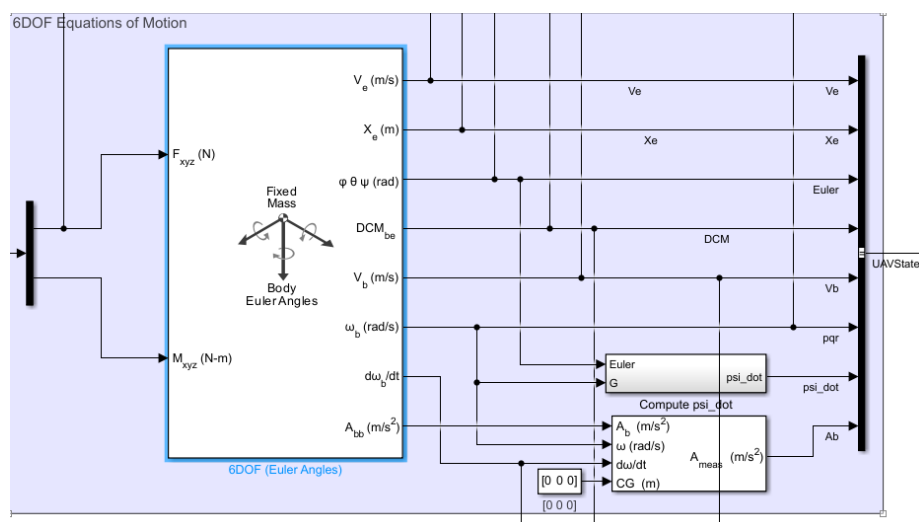
固定翼动力学模型，主要计算了气动力和力矩



### 3) 6DOF 六自由度刚体运动学模块

用于描述无人机在空中运动时的姿态和位置变化。考虑了无人机在三个坐标轴上的旋转运动（俯仰、横滚和偏航）以及机体与地球坐标系上的平移运动（前后、左右和上下）。

还可以根据实际需求对模型进行扩展，考虑更多的因素，如飞行器的非线性特性、气动力和惯性矩等。



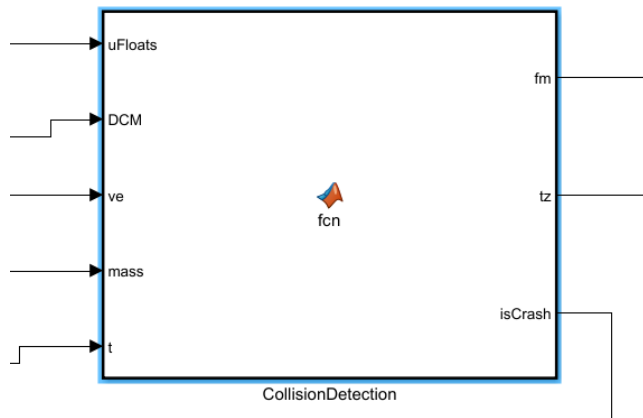
### 4) PhysicalCollisionModel 物理碰撞模块

碰撞模块能检测多旋翼飞行过程中是否碰撞到了物体，以及所碰撞物体的类型并做出符合物理规律的反应。开启碰撞模式后，在碰撞到飞机时多旋翼的速度满足冲量定理，在碰撞到房屋等固定物体时，多旋翼会朝着障碍物反向的反弹回去，反弹速度为原速度的 1/10。

由图可知，碰撞检测模块的输入为  $uFloats$ 、DCM、 $ve$ 、 $mass$ 、 $t$ ，输出为  $fm$ 、 $tz$ 、 $isCrash$ 。

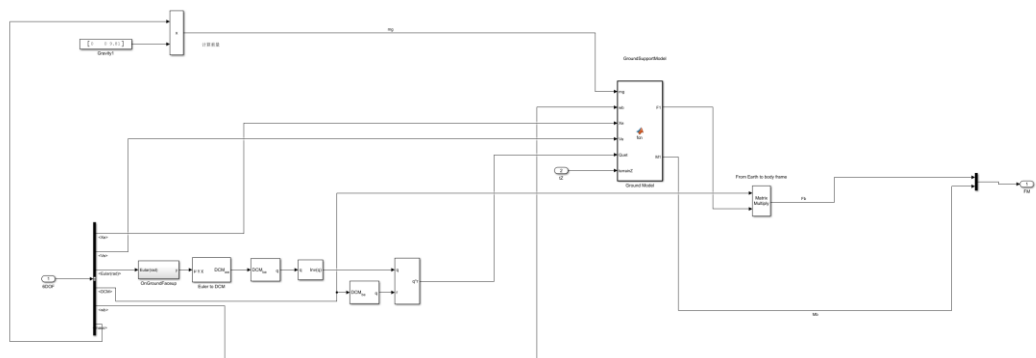
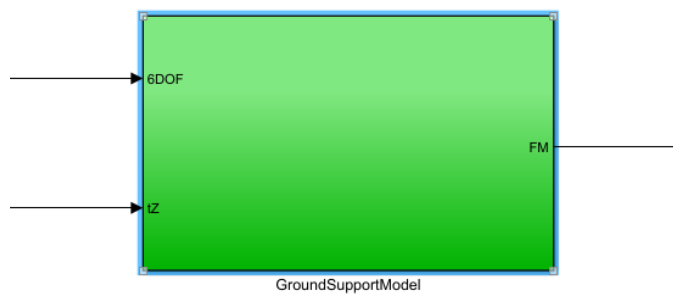
其中  $uFloats$  为 20 维外部输入浮点信号，该端口为碰撞模型预留，可以通过 UDP 网络从 UE4 传输。DCM 为方向余弦矩阵， $ve$  为碰撞时的速度， $mass$  为多旋翼质量， $t$  为时间戳。输出  $fm$  为力和力矩直接作用到 6DOF 对机体运动产生影响， $tz$  则表示多旋翼离地面的高度和 XYZ 的坐标， $isCrash$  则为碰撞判断，如果碰撞发生则三个电机损坏。具体碰撞逻辑可以点击进入该模块详细了解。





## 5) GroundSupportModel 地面支撑模块

GroundSupportModel 地面支撑模块在这里是的一个子模块，这里将所有物体简化为较为简单的基本几何体（例如圆柱体或者长方体）来计算其与地面之间的物理接触受力。



## 6) SensorOutput 传感器输出模块

该模块中包括了环境模型、传感器模型和 GPS 模型

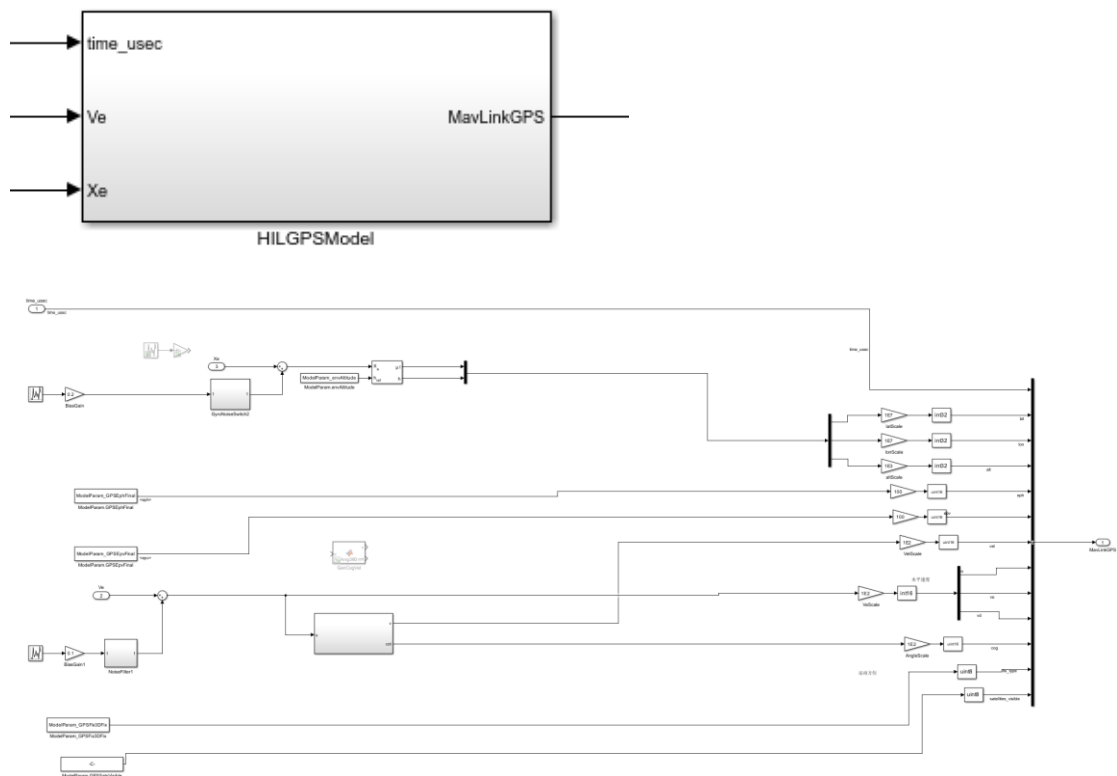
### 环境模型

环境模型对重力和大气压强对无人系统飞行产生的影响进行了模拟



## GPS 模型

GPS 模型用于计算 GPS 数据，在仿真时反馈回 PX4 控制器



## 7) 3DOutput 三维显示模块

该模块会将\*\*\*\_init.m 中的 ModelParam\_uavType（三维显示 ID）、来自电机模型的 ActuatorToUE 以及来自 6DOF 模型的 6DOF Bus 的位置、速度、姿态和加速度等输出为 MavVehile3DInfo，并按协议对输入信息进行数据打包后通过该接口将数据发送至三维引擎

## 2.4. 输出信号（参考 [API.pdf](#) 中 DLL/SO 模型与通信接口的数据协议部分）

\*\*\*\_init.m 会调用 MavLinkStruct.mat 导入四个输出结构体 bus (MavLinkGPS、MavLinkSensor、MavLinkStateQuat 以及 MavVehileInfo) 的定义到工作空间。

```
load MavLinkStruct;
```

最大模型模版包含了 6 个输出信号，分别是 MavHILSensor、MavHILGPS、MavVehile3Dinfo、outCopterData、ExtToUE4、ExtToPX4。

### 1) MavHILSensor (传感器接口集合)

模型发送给 RflySim3D 的真实仿真数据，是平滑的理想值，这些数据可用于 Simulink 下的飞控与模型进行软件仿真测试。对应了 MAVLink 的 `mavlink_hil_sensor_t` 消息，本结构体包含了，加速度传感器的加速度值、陀螺仪传感器的角速度值、磁罗盘传感器的磁场

值，气压和空速传感器的气压值等。这些传感器的值在仿真时由我们的模型提供，在真机飞行时由真实传感器芯片提供。

```
typedef struct __mavlink_hil_sensor_t {
    uint64_t time_usec; /*时间戳，单位毫秒 ms*/
    float xacc; /*机体坐标系 x 方向加速度，单位 m/s^2 */
    float yacc; /*机体坐标系 y 方向加速度，单位 m/s^2 */
    float zacc; /*机体坐标系 z 方向加速度，单位 m/s^2 */
    float xgyro; /*机体坐标系 x 方向角加速度，单位 rad/s */
    float ygyro; /*机体坐标系 y 方向角加速度，单位 rad/s */
    float zgyro; /*机体坐标系 z 方向角加速度，单位 rad/s */
    float xmag; /*机体坐标系 x 方向磁通量，单位 Gauss =T/10000*/
    float ymag; /*机体坐标系 y 方向磁通量，单位 Gauss =T/10000*/
    float zmag; /*机体坐标系 z 方向磁通量，单位 Gauss =T/10000*/
    float abs_pressure; /*绝对气压值，单位 millibar=100Pa*/
    float diff_pressure; /*相对气压值，单位 millibar=100Pa*/
    float pressure_alt; /*气压解算高度值，单位 m*/
    float temperature; /*温度，单位摄氏度*/
    uint32_t fields_updated; /*传感器参数初始化标志位， bit 0 = xacc, bit 12: temperature, bit 31:
全部重新初始化 */
} mavlink_hil_sensor_t;
```

## 2) MavHILGPS (GPS 接口)

模型发送给飞控的 GPS 数据值，它对应了 MAVLink 消息的 `mavlink_hil_gps_t` 结构体。输出信号中包含了经纬高、水平竖直精度、地速、北东地的速度、偏航角、定位状态和卫星数量等数据。这些传感器的值在仿真时由我们的模型提供，在真机飞行时由真实 GPS 模块提供。

```
typedef struct __mavlink_hil_gps_t {
    uint64_t time_usec; /*时间戳，单位毫秒 ms*/
    int32_t lat; /*纬度(WGS84 地球模型)，单位度，再乘以 1E7*/
    int32_t lon; /*经度(WGS84 地球模型)，单位度，再乘以 1E7*/
    int32_t alt; /*高度 (AMSL 地球模型，而不是 WGS84)，单位 m，再乘以 1000 (向上为正)*/
    uint16_t eph; /*GPS 水平方向定位精度，单位 cm，如果不知道设为 65535*/
    uint16_t epv; /*GPS 竖直方向定位精度，单位 cm，如果不知道设为 65535*/
    uint16_t vel; /*GPS 地速，单位 cm/s，如果不知道设为 65535*/
    int16_t vn; /*GPS 地速朝北方向分量，单位 cm/s */
    int16_t ve; /*GPS 地速朝东方向分量，单位 cm/s */
    int16_t vd; /*GPS 地速朝下方向分量，单位 cm/s */
    uint16_t cog; /*运动方向，单位和范围 0~359.99 度，再乘以 100 degrees * 100，如果不知道设为 65535*/
    uint8_t fix_type; /*定位类型 0-1: no fix, 2: 2D fix, 3: 3D fix. */
    uint8_t satellites_visible; /*可见卫星数，如果不知道设为 255*/
} mavlink_hil_gps_t;
```

注：GPS 数据的发送频率与真实传感器硬件基本相同为 10Hz，因此飞控的实时位置并不能靠 GPS 直接提供，需要与 IMU 等传感器进行融合滤波估计得到。

## 3) MavVehicle3Dinfo (真实仿真数据输出)

模型发送给飞控的各种传感器数据的集合，对应了 MAVLink 的 `mavlink_hil_sensor_t` 消息。输出信号中包括了加速度传感器的加速度值、陀螺仪传感器的角速度值、磁罗盘传感器的磁场值，气压和空速传感器的气压值等。

```
struct SOut2Simulator {
```

```

int copterID; //飞机 ID, 用于区分局域网内不同飞机
int vehicleType; //飞机样式, 区分同种飞机 (如四旋翼) 下的不同样式 (例如, 大疆、AR.Drone)
double runnedTime; //时间戳, 当前时刻的时间, 单位毫秒
float VelE[3]; //速度向量, 地球坐标系的 xyz 速度 (z 向下为正), 单位 m/s
float PosE[3]; //位置向量, 地球坐标系下的 xyz 方向 (z 向下为正, 单位 m, 以起飞点为坐标原点
float AngEuler[3]; //姿态角, 飞机的欧拉角, 定义于机体坐标系, 单位弧度
float AngQuatern[4]; //四元数, 飞机姿态的四元数, 定义于机体坐标系
float MotorRPMs[8]; //电机转速, 飞机的各个旋翼转速, 单位转每分
float AccB[3]; //加速度, 飞机的运动加速度, 单位 m/s^2
float RateB[3]; //角速度, 飞机的转动角速度, 单位 rad/s
double PosGPS[3]; //GPS 坐标, 飞机的经纬高坐标, 单位度、度、米
};

```

### 3. 实验效果

实现固定翼飞机 DLL 模型文件生成, 以及完成固定翼软硬件在环仿真。

### 4. 文件目录

文件夹/文件名称	说明
AircraftMathworksWithCollision.slx	固定翼飞机模型文件。
AircraftMathworksWithCollisionHITLRun.bat	硬件在环仿真批处理文件。
AircraftMathworksWithCollisionSITLRun.bat	软件在环仿真批处理文件。
GenerateModelDLLFile.p	DLL 格式转化文件。
AircraftMathworksWithCollision_Init.m	动力学模型相关参数。
InitData.m	控制器初始化参数。
MavLinkStruct.mat	MavLink 数据结构体 mat 文件

### 5. 运行环境

序号	软件要求	硬件要求	
		名称	数量
1	Windows 10 及以上版本	笔记本/台式电脑 <sup>①</sup>	1
2	RflySim 平台免费版	卓翼 H7 飞控 <sup>②</sup>	1
3	MATLAB 2017B 及以上 <sup>③</sup>	数据线	1

① 推荐配置请见: <https://doc.rflysim.com/1.1InstallMethod.html>

② 须保证平台安装时的编译命令为: droneeye\_zyfc-h7\_default, 固件版本为: 1.12.3。其他配套飞控请见: <http://doc.rflysim.com/hardware.html>

### 6. 实验步骤









#### 6.1. DLL 模型生成

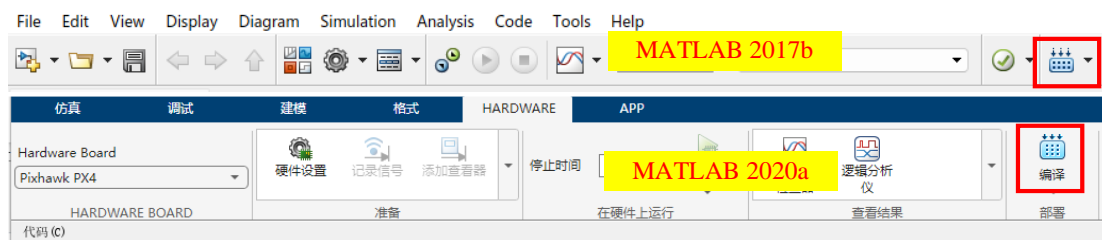
##### Step 1:

打开 “Init.m” 文件, 并运行。

## Step 2:

打开“AircraftMathworksWithCollision.slx” Simulink 文件，点击“Build Model”按钮。

	GenerateModelDLLFile.p	2022/12/16 16:35	MATLAB P-code	5 KB
	Init.m	2023/7/3 17:22	MATLAB Code	4 KB
	Init_control.m	2019/7/30 22:18	MATLAB Code	2 KB
	MavLinkStruct.mat	2022/5/9 10:27	MATLAB.mat.9.1...	5 KB
	MulticopterNoCtrlWithCollision.slx	2022/10/5 23:30	Simulink Model	89 KB
	MulticopterNoCtrlWithCollisionHITL...	2023/6/13 15:11	Windows 批处理...	6 KB
	MulticopterNoCtrlWithCollisionSITL...	2023/6/13 15:11	Windows 批处理...	6 KB
	Readme.docx	2023/8/11 10:20	Microsoft Word ...	5,395 KB

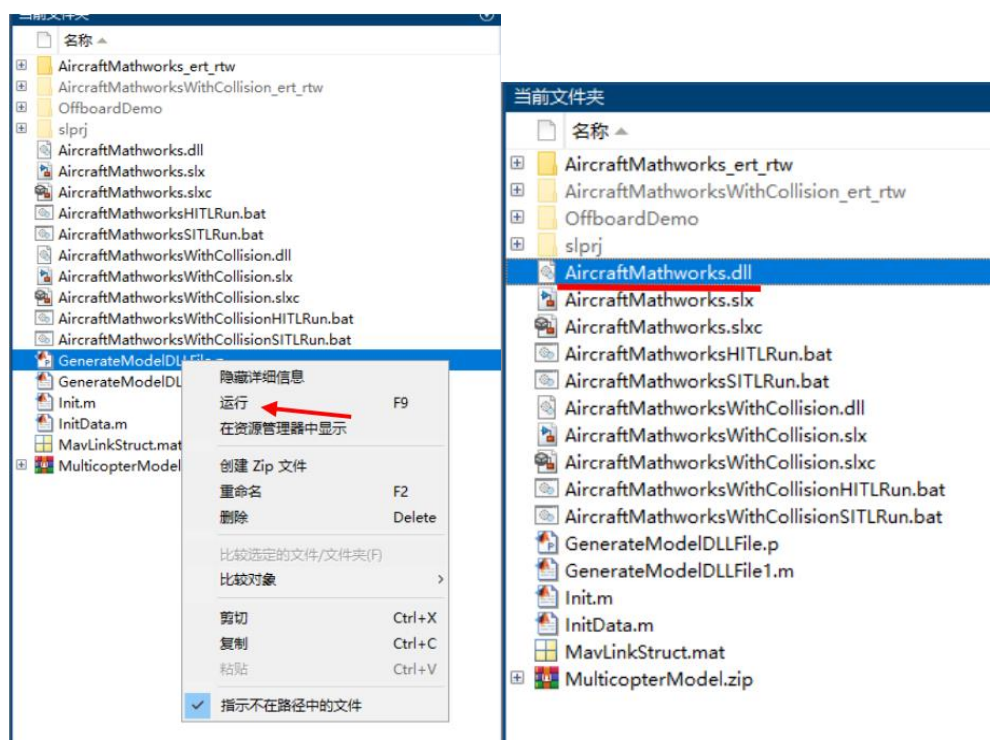


注意事项：与固定翼模型 AircraftMathworks.slx 相比，该固定翼模型不同点如下：

- 1) AircraftMathworksWithCollision.slx 中碰撞模块中除了地面模块之外，还有碰撞判断模块，在确认碰撞后会输出力和力矩并影响电机的输出。
- 2) UE 通过 inFloatsCollision 端口给固定翼模型发送碰撞信号。

## Step 3:

代码生成完毕后，在 Matlab 中右键“GenerateModelDLLFile.p”文件，点击运行，生成 DLL 文件。

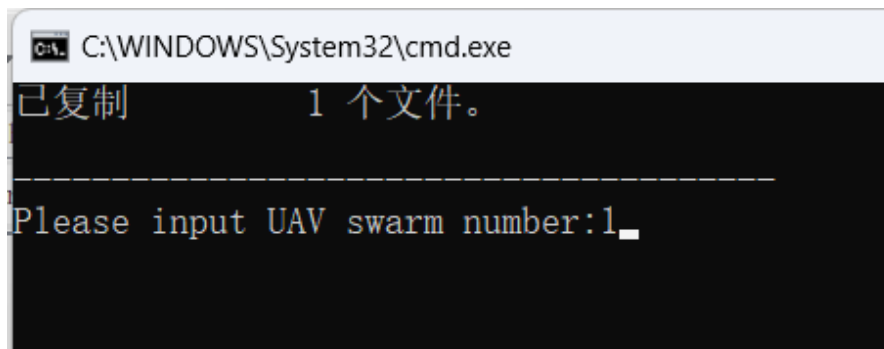


## 6.2. 软件在环仿真

### Step 1:

右键以管理员身份运行“AircraftMathworksWithCollisionSITLRun.bat”批处理文件，在弹出的终端窗口中输入 1，启动一架飞机的软件在环仿真。

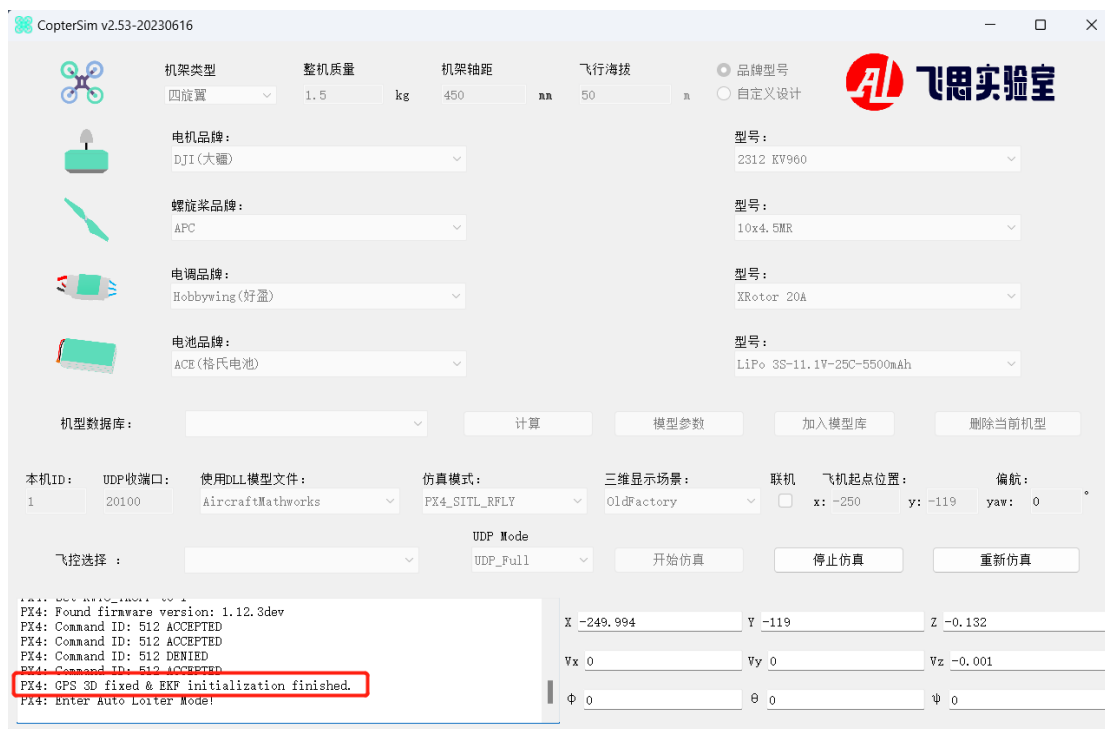
OffboardDemo	2023/6/28 5:35	文件夹	
VisionCtrlDemo	2023/6/28 5:35	文件夹	
AircraftMathworks.dll	2022/10/10 13:20	应用程序扩展	255 KB
AircraftMathworks.slx	2022/10/10 13:19	Simulink Model	104 KB
AircraftMathworksHITLRun.bat	2023/6/13 15:11	Windows 批处理...	6 KB
AircraftMathworksSITLRun.bat	2023/6/13 15:11	Windows 批处理...	6 KB
AircraftMathworksWithCollision.dll	2022/10/5 23:22	应用程序扩展	261 KB
AircraftMathworksWithCollision.slx	2022/10/10 13:25	Simulink Model	107 KB
AircraftMathworksWithCollisionHITL...	2023/6/13 15:11	Windows 批处理...	6 KB
AircraftMathworksWithCollisionSITL...	2023/6/13 15:11	Windows 批处理...	6 KB
GenerateModelDLLFile.p	2022/12/16 16:35	MATLAB P-code	5 KB
Init.m	2023/7/3 17:21	MATLAB Code	5 KB
InitData.m	2022/8/16 20:45	MATLAB Code	4 KB
MavLinkStruct.mat	2022/5/9 10:27	MATLAB Data	5 KB
MulticopterModel.zip	2022/10/10 13:25	压缩(zipped)文件...	130 KB
Readme - 固定翼.docx	2023/7/3 9:30	Microsoft Word ...	11,844 KB



## Step 2:

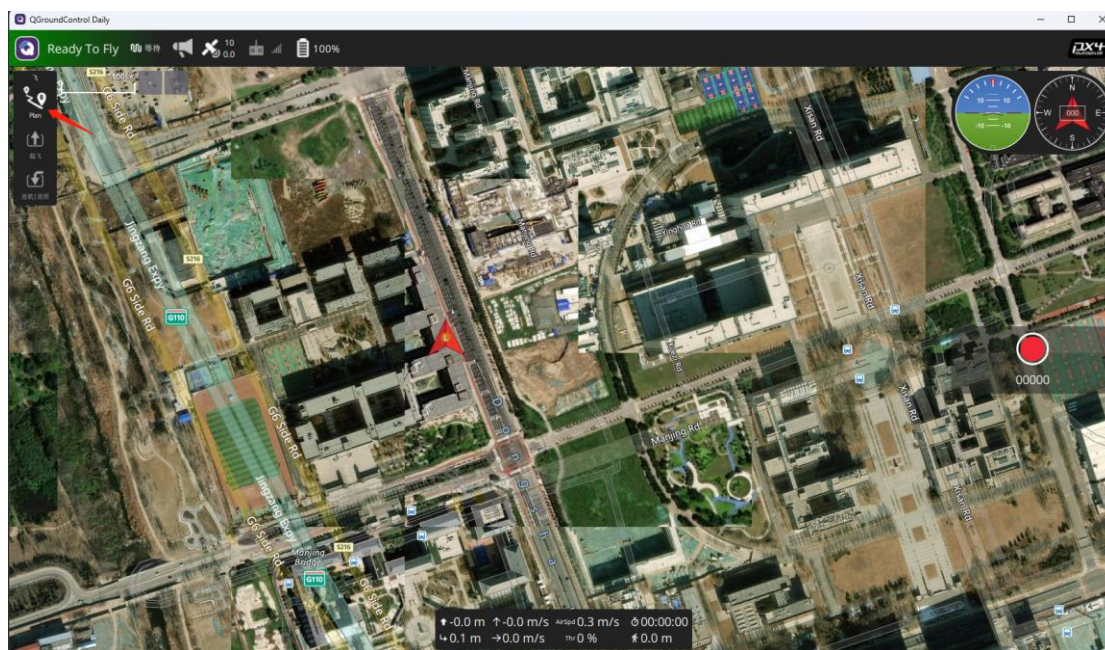
等待 CopterSim 中显示连接上 RflySim3D，完成初始化。





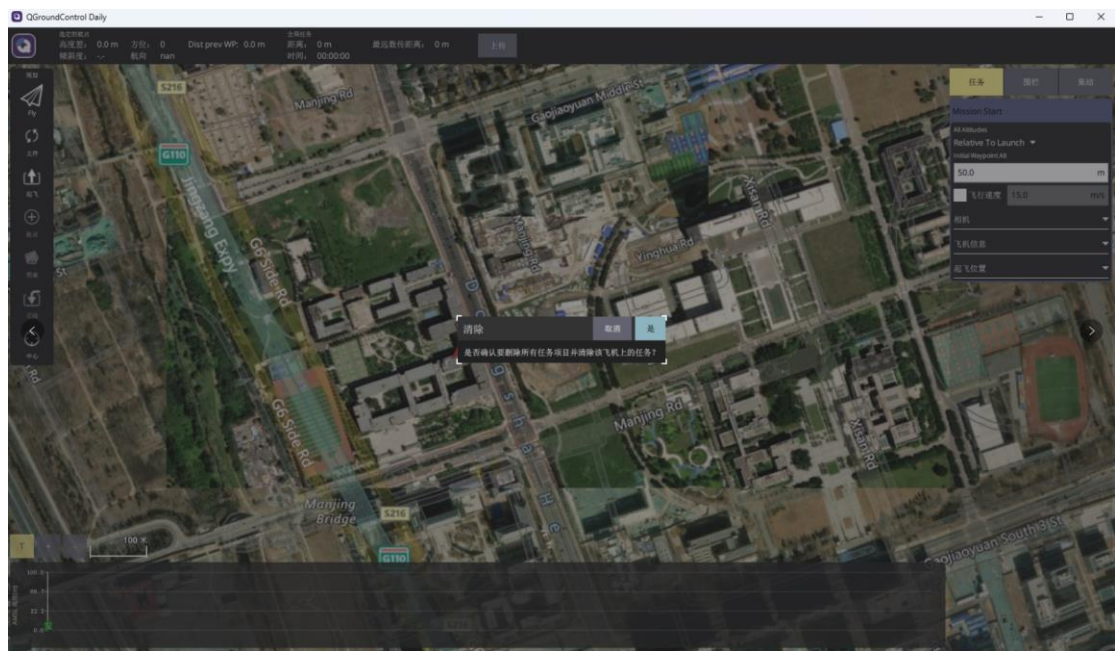
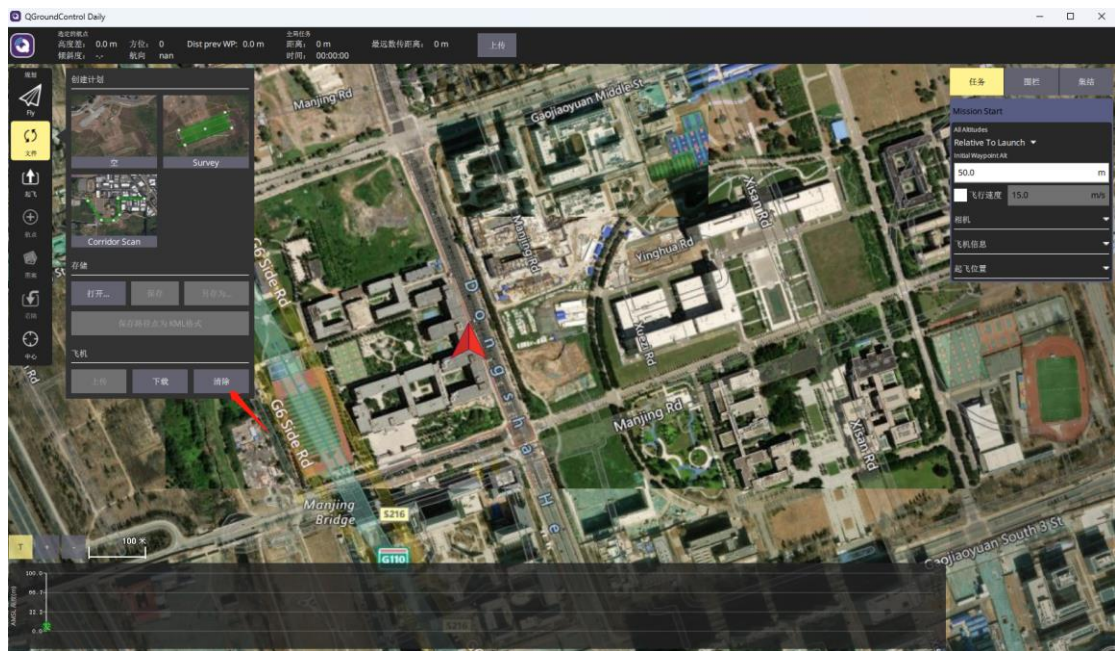
## Step 3:

点击 QGC 左上角的 Plan，进入航路设置页面。



## Step 4:

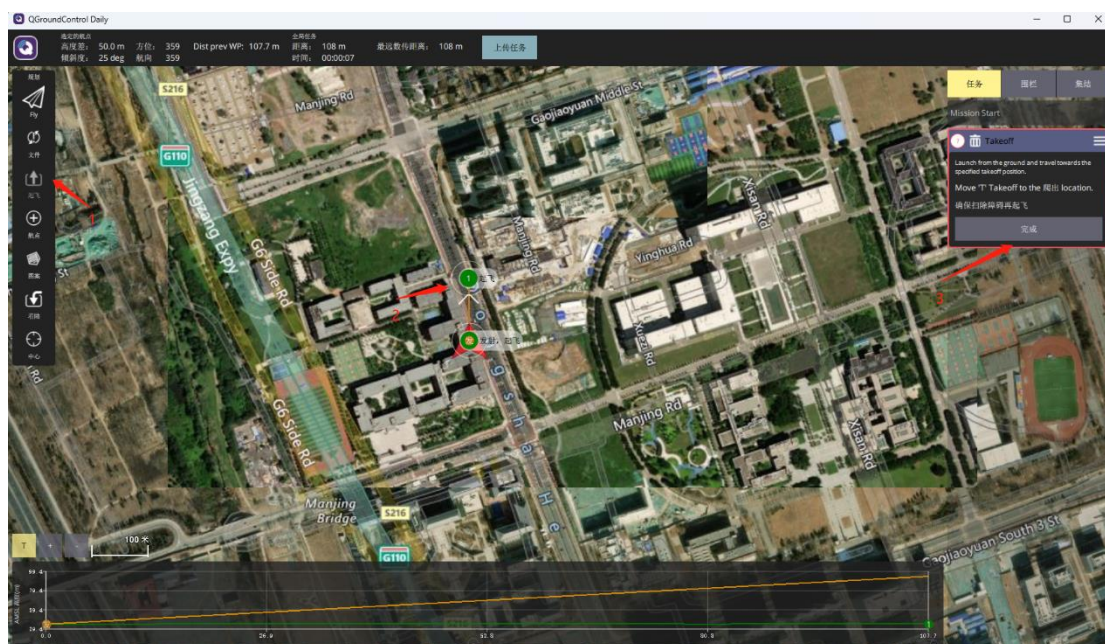
若已存在航路，则先点击文件按钮，之后点击“清除”按钮清除航路，在弹出界面选择“是”。



## Step 5:

点击“起飞”按钮，可拖动绿色“起飞”点来设置起飞位置，之后点击右侧“完成”按钮。





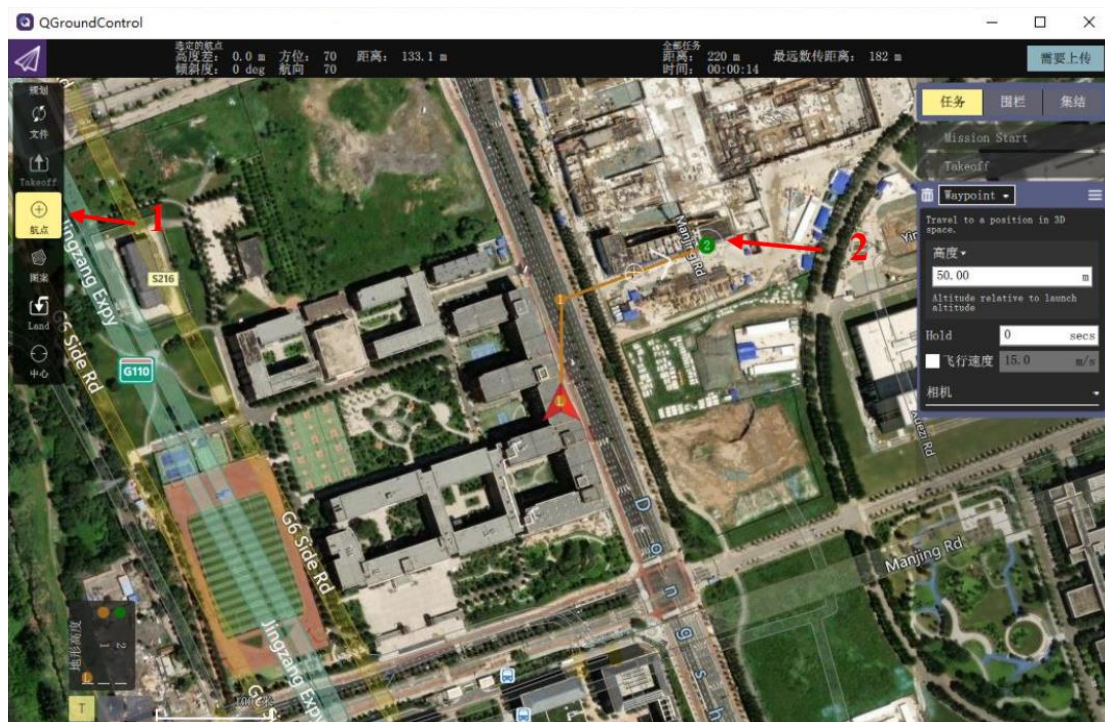
## Step 6:

可设置起飞高度和速度，本例中采用默认值。



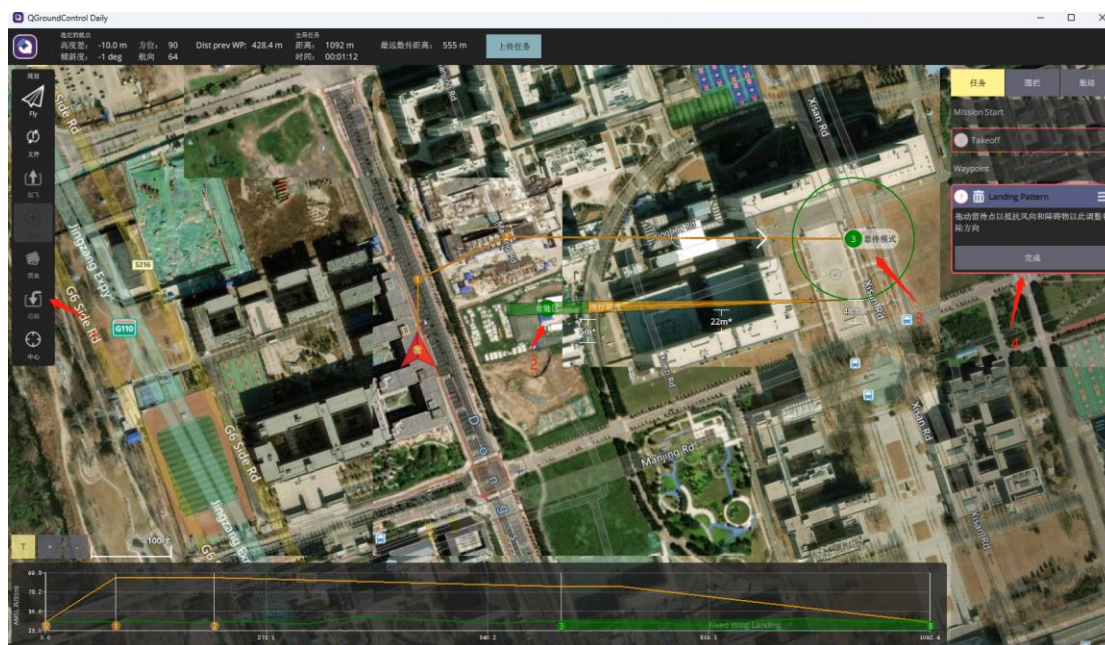
## Step 7:

点击“航点”按钮，之后在地图上点击任意位置可设置航点，同上一步可设置高度和速度（航点可设置多个，本例中只设置一个）。



## Step 8:

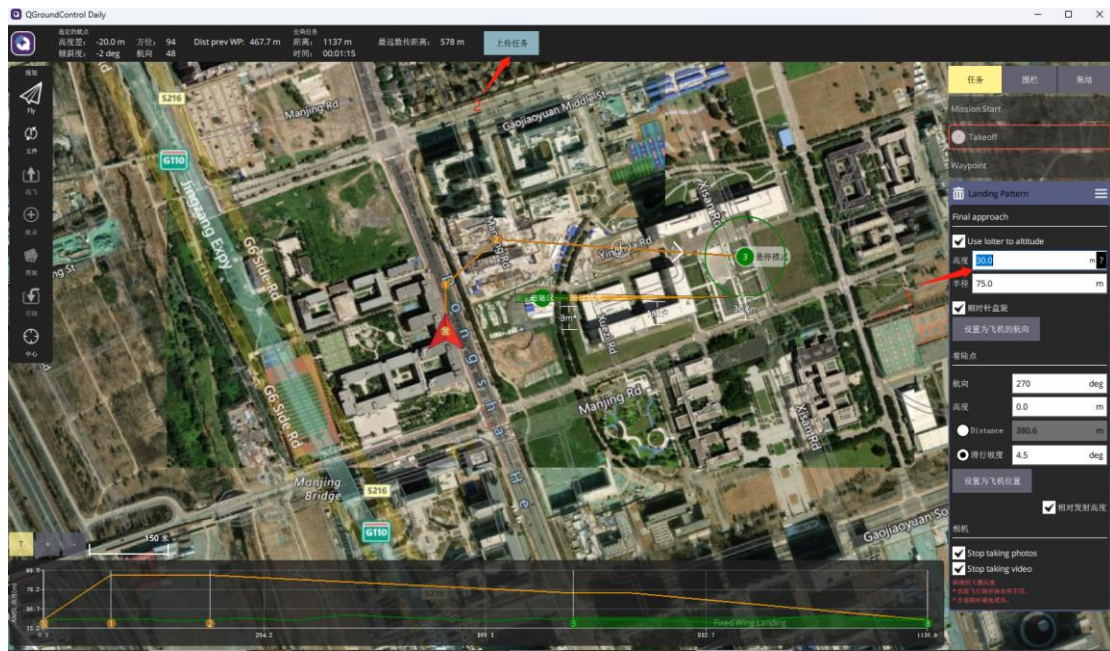
点击“着陆”按钮，并在地图上点击位置设置降落点，拖动绿色“悬停”图标可更改盘旋位置，之后点击右侧“完成”。



## Step 9:

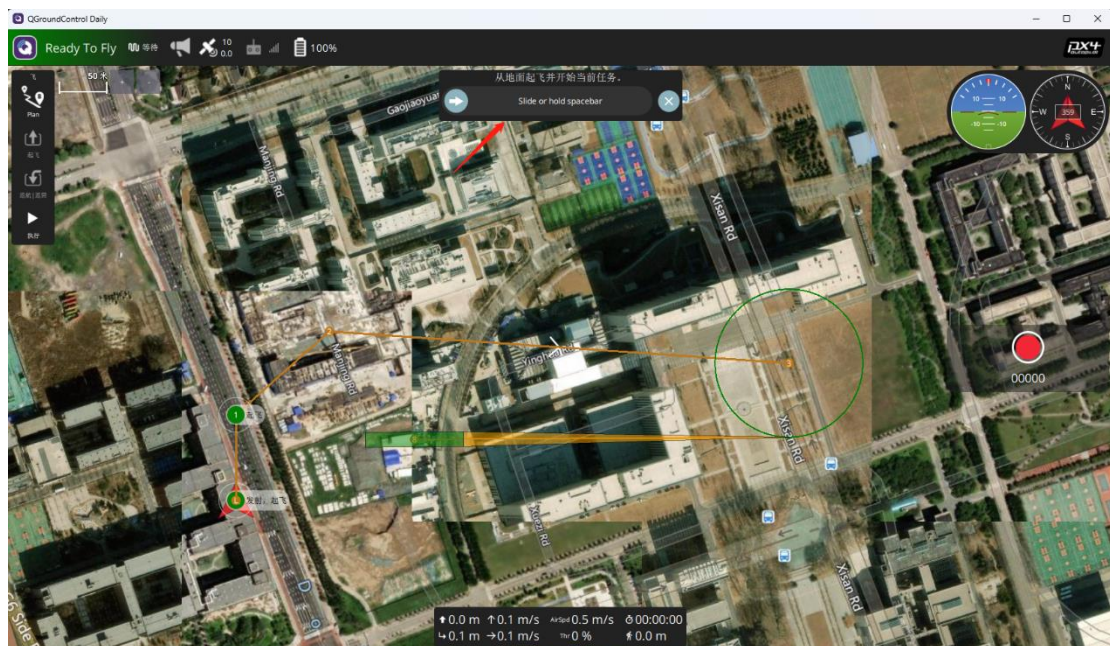
之后可设置降落高度等其他参数，本例中设置降落高度为 30m，之后点击“上传任务”按钮上传航路。





## Step 10:

返回初始界面后，滑动上方滑块开始执行任务。



## Step 11:

在 RflySim3D 中观察是否按 QGC 规划轨迹飞行。



### 6.3. 硬件在环仿真

#### Step 1:

按下图所示将飞控与计算机链接。



#### Step 2:

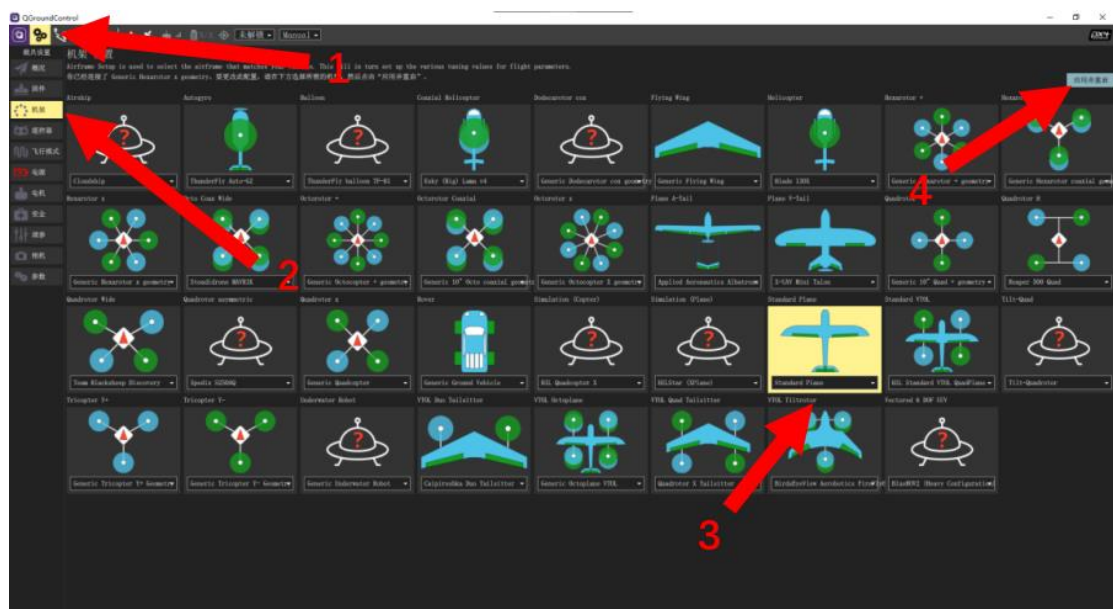
推荐使用 Pixhawk 6C 进行硬件在环仿真，固件版本为 1.13.3。

在 Rflytools 文件夹中打开 QGC 地面站。

3DDisplay	2023/7/27 15:02	快捷方式	1 KB
CopterSim	2023/7/27 15:02	快捷方式	1 KB
FlightGear-F450	2023/7/27 15:02	快捷方式	2 KB
HITLRun	2023/7/27 15:02	快捷方式	2 KB
Python38Env	2023/7/27 15:02	快捷方式	2 KB
QGroundControl	2023/7/27 15:02	快捷方式	1 KB
RflySim3D	2023/7/27 15:02	快捷方式	1 KB
RflySimAPIs	2023/7/27 15:02	快捷方式	1 KB
RflySimUE5	2023/7/27 15:02	快捷方式	1 KB
SITLRun	2023/7/27 15:02	快捷方式	2 KB
Win10WSL	2023/7/27 15:02	快捷方式	2 KB

### Step 3:

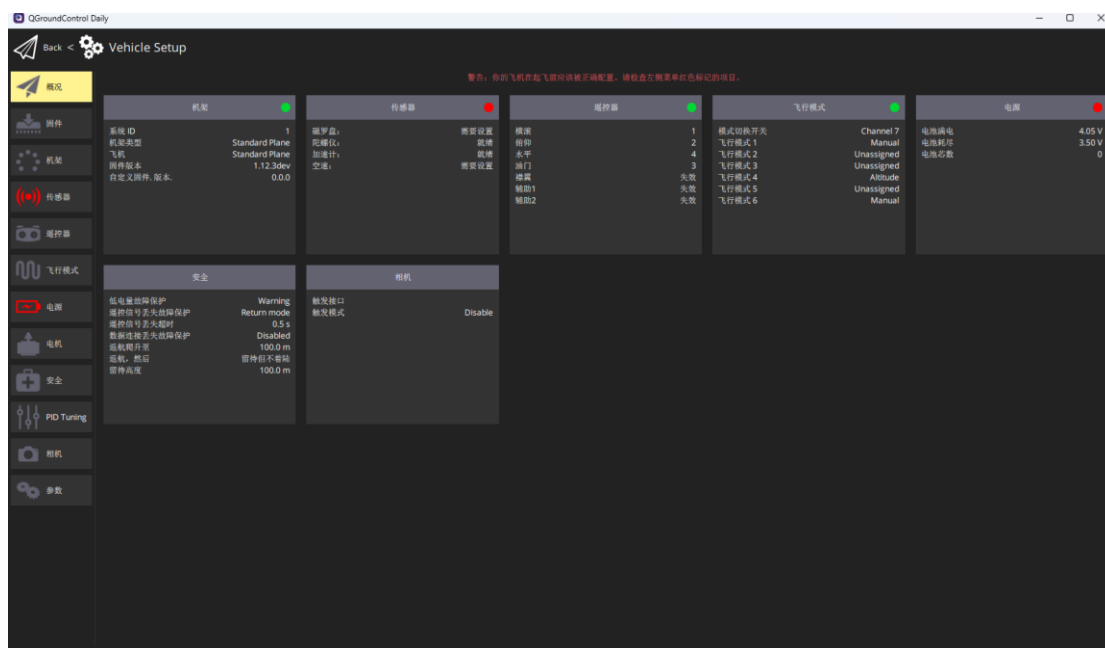
在机架界面设置机架型号为“Standard Plane”，设置完毕后点击右侧“应用并重启”。



### Step 4:

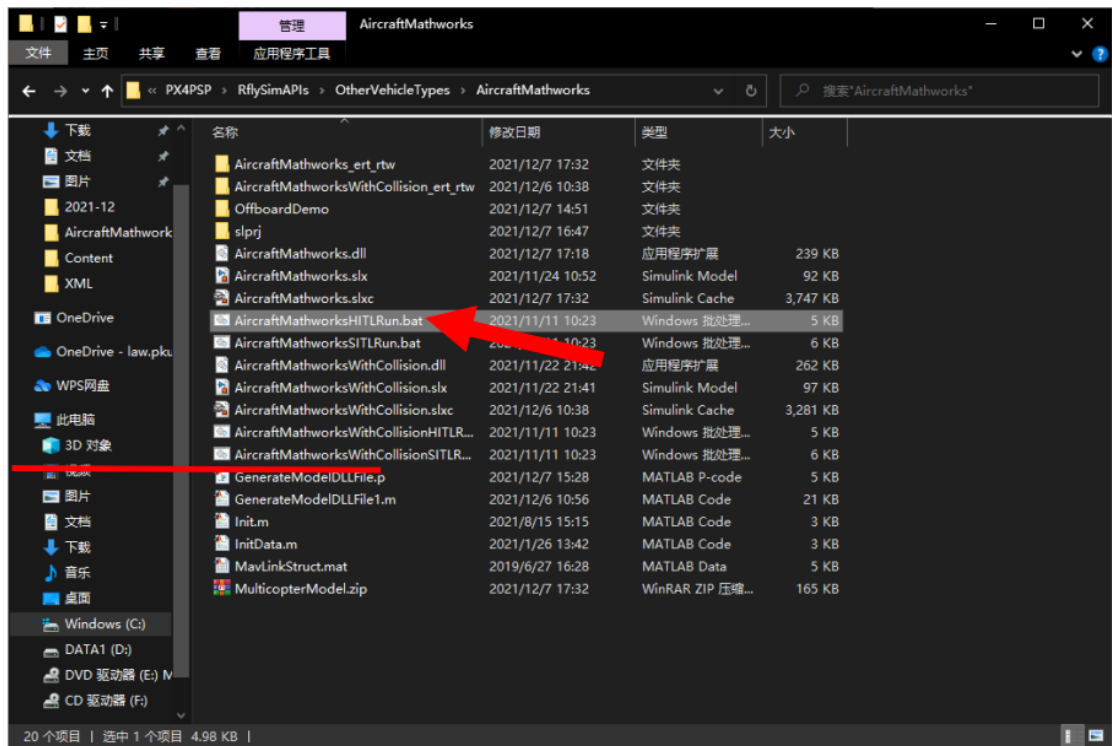
在“安全”界面，选择“HITL enabled”启动硬件在环仿真，之后在概况界面中确认配置完成后，重新插拔飞控完成设置。





右键以管理员身份运行“AircraftMathworksWithCollisionHITLRun.bat”批处理文件，在弹出的终端窗口中根据串口提示输入串口号5，启动一架飞机的硬件在环仿真。





```
C:\Windows\system32\cmd.e  X  +  v

已复制      1 个文件。

-----
Please input the Pixhawk COM port list for HIL
Use ',' as the separator if more than one Pixhawk
E.g., input 3 for COM3 of Pixhawk on the computer
Input 3,6,7 for COM3, COM6 and COM7 of Pixhawks

Available COM ports on this computer are:
COM3: ??????????
COM4: ??????????
COM5: USB ????.

Recommended COM list input is: 3,4,5

-----
My COM list for HITL simulation is:5
```

## Step 6:

之后测试步骤与软件在环的 Step3 到 Step11 相同，运行之后在 RflySim3D 中观察是否按 QGC 规划轨迹飞行。

## 7. 参考文献

[1]. 无。

[2].

## 8. 常见问题

Q1: \*\*\*\*

A1: \*\*\*\*

---