

1. 实验名称及目的

平台建模模板之最大模板使用介绍：该例程对如何使用平台最大模板进行软件在环和硬件在环仿真进行介绍。

2. 实验原理

Exp2_MaxModelTemp.slx 是最大系统的模版，相对于最小系统模板，包含了更丰富的附加功能。附加的输入、输出和参数不需要全部加上，只需要在使用本功能时加入模型即可，但是务必保证名称、维度和数据格式一致。

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

Exp2_MaxModelTemp_init.m 中定义了最大系统模型的各种参数，关键数据如下。

飞机的三维显示样式

```
ModelParam_uavType = int16(3); %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_GPSLatLong = [40.1540302 116.2593683]; %飞机初始的纬度和精度，单位度。  
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 0 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
```

故障接口参数

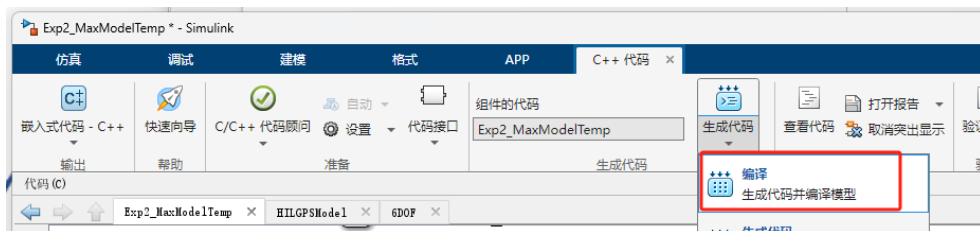
```
FaultInParams: 可通过外部消息动态改变的 32 维参数向量，在故障注入或者可变形的异构飞行器上有用，也可动态地调整传感器模型噪声等；与 inSILInts 和 inSILFloats 形成功能互补。  
FaultParamAPI.FaultInParams = zeros(32,1); % 定义了一个名为 FaultInParams 的 32 维向量，该向量被初始化为所有元素都为零。  
FaultParamAPI.FaultInParams(3)=1; %将 FaultInParams 向量的第三个元素设置为 1。
```

2) 参数调用过程

Exp2_MaxModelTemp.slx 是 DLL 模型生成的完整模板，模型启动运行（编译）时会调用 Exp2_MaxModelTemp_init.m



Exp2_MaxModelTemp_init.m 中包含了模型的参数信息，本脚本会在 Exp2_MaxModelTemp.slx 编译（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。



工作区	
名称 ▲	值
ext	'slx'
FaultParamAPI	1x1 struct
filepath	'E:\d2\4.RflySimMo...
HILGPS	1x1 Bus
InitFileName	'Exp2_MaxModelTe...
MavLinkGPS	1x1 Bus
MavLinkSensor	1x1 Bus
MavLinkStateQuat	1x1 Bus
MavVehileInfo	1x1 Bus
ModellInit_AngE...	[0,0,0]
ModellInit_Inputs	1x16 double
ModellInit_PosE	[0,0,0]
ModellInit_RateB	[0,0,0]
ModellInit_RPM	0
ModellInit_VelB	[0,0,0]
ModelParam_en...	-50
ModelParam_GP...	[40.1 1x1 double]
ModelParam_m...	842.1000
ModelParam_m...	1.2870e-04
ModelParam_m...	0.0500
ModelParam_m...	0.0214
ModelParam_m...	22.8300
ModelParam_rot...	2.7830e-07
ModelParam_rot...	1.6810e-05
ModelParam_ua...	[0.0035,0.0039,0.00...
ModelParam_ua...	0.0550
ModelParam_ua...	0.1200
ModelParam_uavJ	[0.0211,0,0;0,0.0219...
ModelParam_ua...	1.5150
ModelParam_ua...	4
ModelParam_ua...	0.2250
ModelParam_ua...	3
name	'Exp2_MaxModelTe...

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为复杂地形的地表面 (例如 RflySim3D 中的 Grassland 地图)。

3) inSILInts

8 维 Int32 型输入, 通过 UDP 协议获取, 来自 30100++2 系列端口号, 软硬件在环仿真时, 可通过该端口向模型输入一些量; 同时, 该接口是实现综合模型的关键接口。

4) inSILFloats

20 维 float 型输入, 通过 UDP 协议获取, 来自 30100++2 系列端口号, 软硬件在环仿真时, 可通过该端口向模型输入一些量; 同时, 该接口是实现综合模型的关键接口。

5) 碰撞数据 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

6) 飞控状态量(uORB 数据)输入 inCopterData

32 维, 其中后 8 维接收 PX4 消息, 数据来自 uORB msg rfly_px4.control[0:7]。

7) inFromUE

6 维 double 型数据, 来自三维引擎 (Rflysim3D/RflySimUE5), 可用于实现地面交互、碰撞引擎等需要与三维引擎进行数据交互的相关功能。

8) inCtrlExt1~5

和 inSILInts 和 inSILFloats 类似, 用于接收局域网内的故障注入消息

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

推荐不改变本模型模版, 而是在其中修改模型参数 (质量、转动惯量等) 和电机模块

（转速动态响应）与力和力矩模块（驱动力、气动力、地面支撑和阻力等）两个自定义模块，来适配不同的载具的模型。

1) Motor Model 电机模块

在该模块中输入为 PWM 值（通过 inPWMs 接口获取），经过各电机的非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 and 力矩模型的电机转速（弧度每秒）；输入给 UE 的电机转速（转每分）

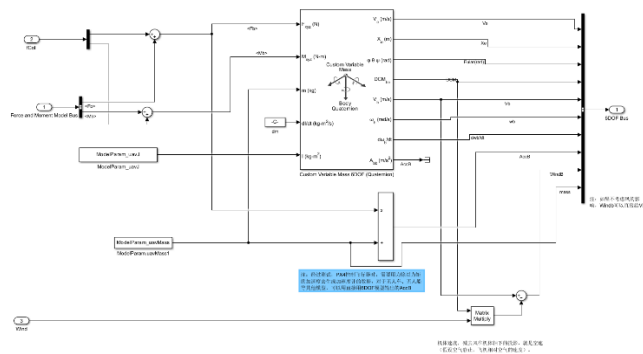
2) Force and Moment Model 力和力矩模块

该模块输入为电机转速 MotorRads、飞机运动学姿态 6DOF 和地形高度输入 TerrainZ，输出为多旋翼合力、合力矩 Force and Moment Model Bus。

3) 6DOF 六自由度刚体运动学模块

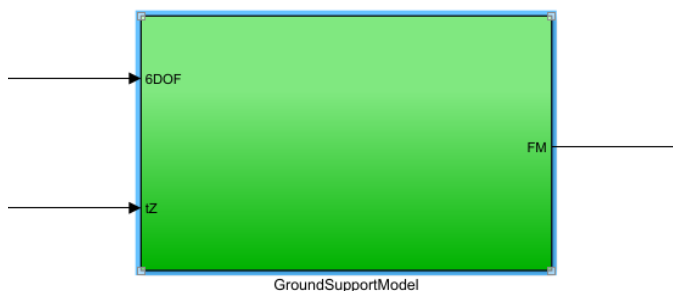
用于描述无人机在空中运动时的姿态和位置变化。考虑了无人机在三个坐标轴上的旋转运动（俯仰、横滚和偏航）以及机体与地球坐标系上的平移运动（前后、左右和上下）。

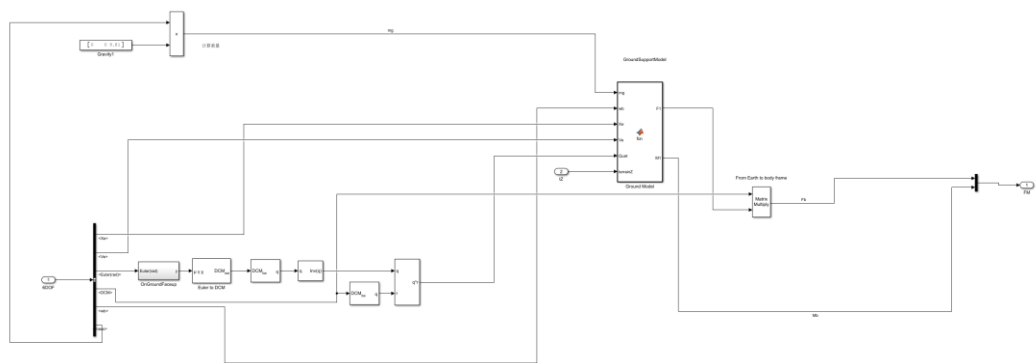
还可以根据实际需求对模型进行扩展，考虑更多的因素，如飞行器的非线性特性、气动力和惯性矩等。



4) GroundSupportModel 地面支撑模块

GroundSupportModel 地面支撑模块实际上是 [PhysicalCollisionModel 碰撞检测模块](#) 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。



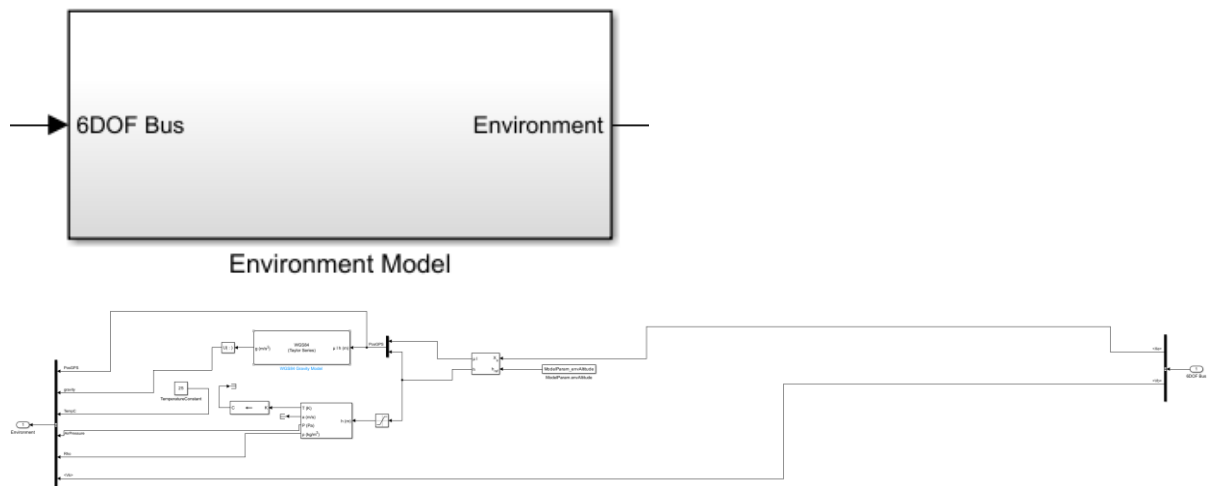


5) SensorOutput 传感器输出模块

该模块中包括了环境模型、传感器模型和 GPS 模型

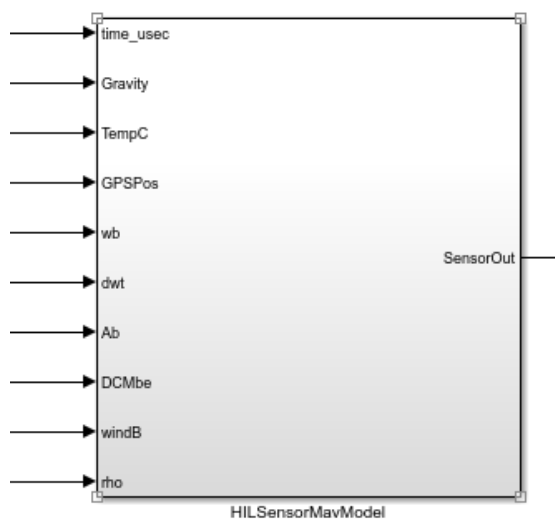
环境模型

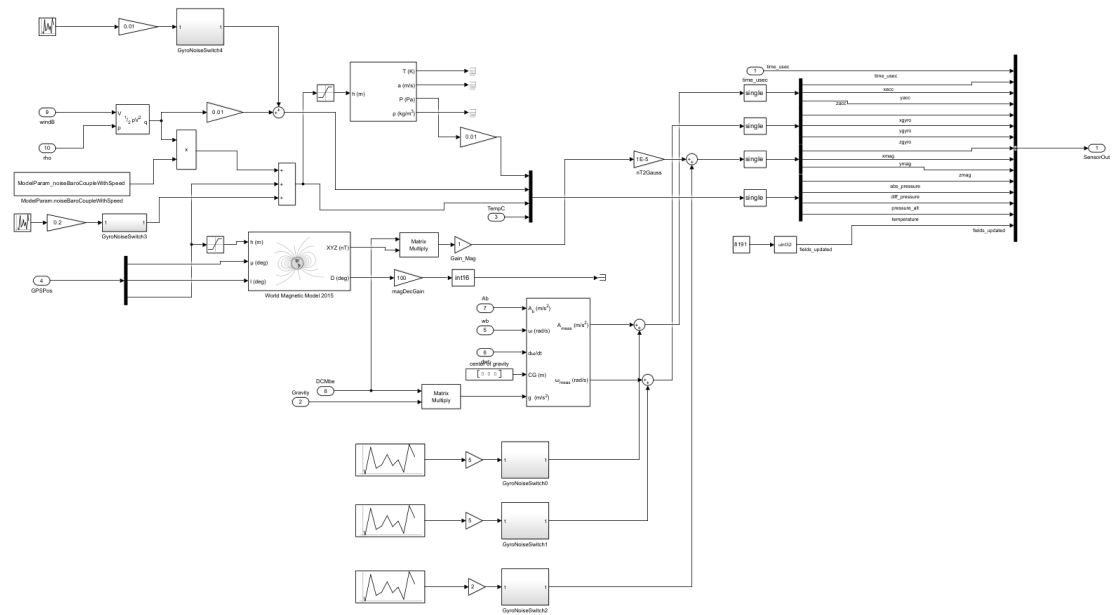
环境模型对重力和大气压强对无人系统飞行产生的影响进行了模拟



传感器模型

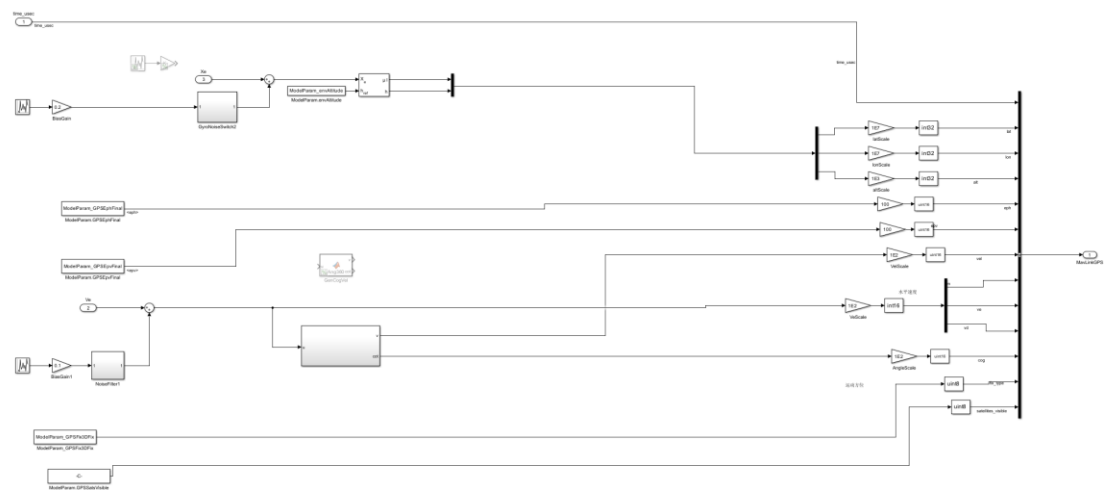
传感器模型中不仅对磁力计、惯性导航进行了建模,同时加入了噪声模拟





GPS 模型

GPS 模型用于计算 GPS 数据，在仿真时反馈回 PX4 控制器



6) 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
};
```

4) outCopterData (自定义日志输出)

32 维 double 型, 里面的内容可自定义发送数据。发往本接口的数据, 一方面会写入到本地的 log 日志中 (在 C:\PX4PSP\CopterSim 下新建 CopterSim*.csv, 才会开始记录*号飞机的数据, 注意这里*要换成飞机的 ID)。另一方面, 本数据会通过 UDP 传输到 30101 系列端口

5) ExtToUE4 (自定义显示数据输出)

16 维 double 型数据, 通过 20100 系列端口发送给 RflySim3D 作为第 9-24 维执行器控制消息显示

6) ExtToPX4 (自定义 uORB 数据输出)

16 维 float 型数据, 以串口的方式发送给 PX4 的 uORB 消息 rfly_ext, 用于传输其他传感器或必要数据给飞控

3. 实验效果

将平台最大模板生成 dll 文件后, 可进行软硬件在环仿真。

4. 文件目录

文件夹/文件名称	说明
Exp2_MaxModelTemp.slx	最大模板源程序文件。
MavLinkStruct.mat	包含数据结构体。
Exp2_MaxModelTempSITL.bat	软件在环仿真批处理文件。
Exp2_MaxModelTempHITL.bat	硬件在环仿真批处理文件。
GenerateModelDLLFile.p	DLL 格式转化文件。
Exp2_MaxModelTemp_init.m	最大模型参数初始化文件。

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 ^①	1
2	RflySim 平台免费版	Pixhawk 6C ^②	1
3	MATLAB 2017B 及以上 ^③	数据线	1

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

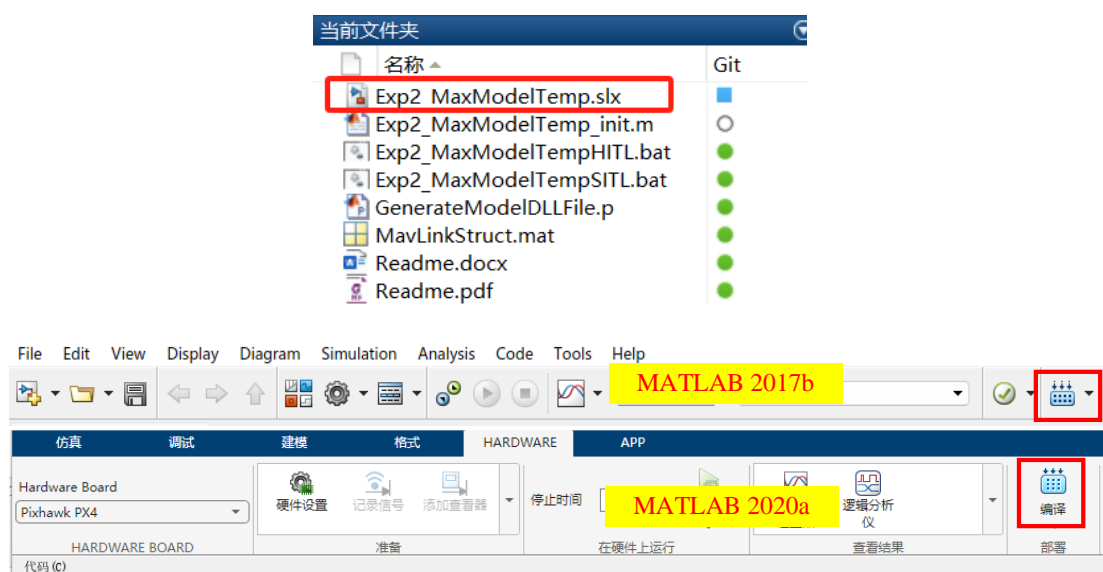
②：须保证平台安装时的编译命令为：px4_fmu-v6c_default，固件版本为：1.13.3。其他配套飞控请见：<http://doc.rflysim.com>。

6. 实验步骤

6.1. DLL 模型生成

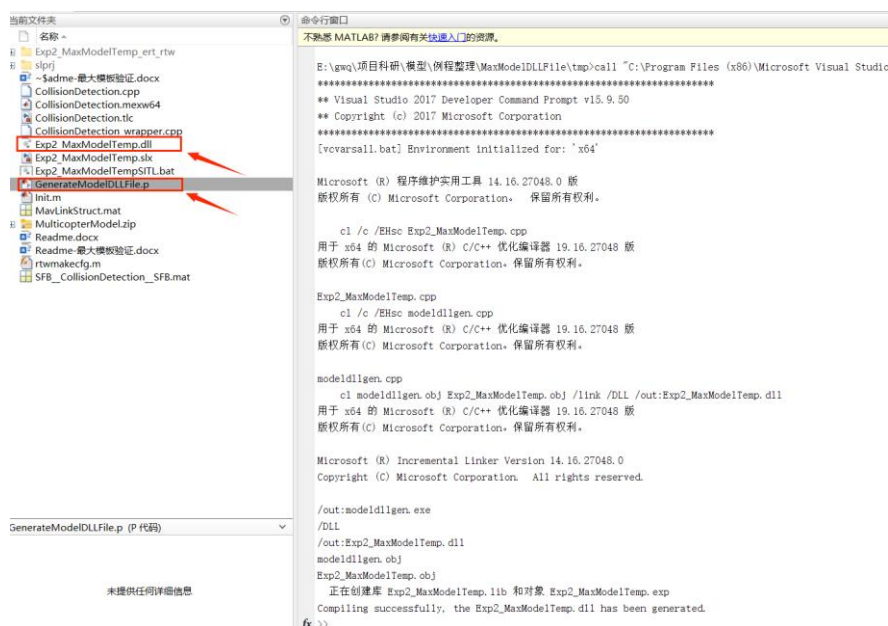
Step 1:

打开“Exp2_MaxModelTemp.slx”文件，点击编译（“Build Model”）按钮。



Step 2:

编译完成后，会自动生成多个相关文件，右键 **GenerateModelDLLFile.p** 并点击运行，就可以生成 “Exp2_MaxModelTemp.dll” 的 DLL 模型文件。











注意事项：与最小模型 **Exp1_MinModelTemp** 相比,最大模型新增了如下输入、输出接口，可实现更多功能。

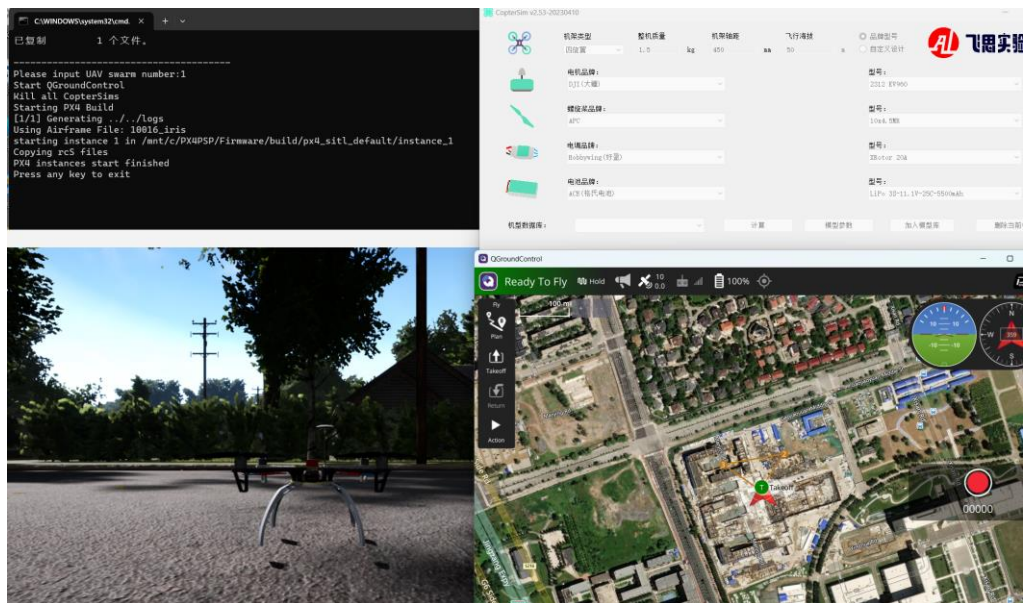
- 1) **inSILInts**: 一个 8 维的自定义输入端口。
- 2) **inSILFloats**: 一个 20 维的自定义输入端口。
- 3) **inFloatsCollision**: 一个为碰撞模型预留的 20 维端输入口，信号从 UE4 传输。
- 4) **inCopterData**: 一个信号由 **CopterSim** 传输的 32 维输入端口。
- 5) **inFromUE**: UE 发送的 32 维输入端口，用于处理模型和场景的交互。
- 6) **inCtrlExt1~5**: 和 **inSILInts** 和 **inSILFloats** 类似，用于接收局域网内的故障注入消息。
- 7) **outCopterData**: 向 **CopterSim** 输出外部信号的 32 维输出端口，定义如下。数据将存储在日志文件 **CopterSim*.csv** 中(应该创建一个文件来存储数据)，其中*是 **CopterSim** 的 **CopterID**。
- 8) **ExtToUE4**: 一个发送给 UE4 的 16 维输出端口。
- 9) **ExtToPX4**: 一个发送给 PX4 的 16 维输出端口。

6.2. 软件在环仿真

Step 1:

运行 “Exp2_MaxModelTempSITL.bat” 脚本，可以自动打开软件在环仿真，并完成所有需要配置。

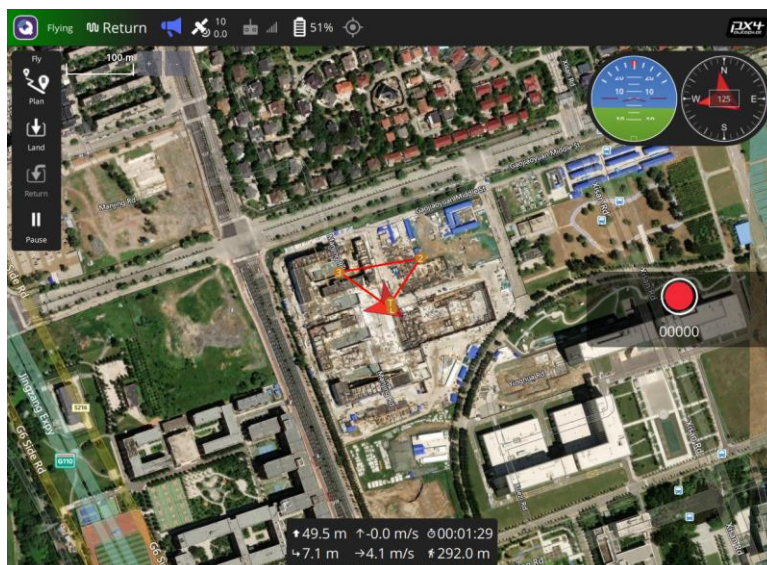
 Exp2_MaxModelTemp.slx	2023/10/23 16:30	Simulink Model	72 KB
 Exp2_MaxModelTempHITL.bat	2023/10/17 15:11	Windows 批处理...	6 KB
 Exp2_MaxModelTempSITL.bat	2023/10/17 15:11	Windows 批处理...	6 KB
 GenerateModelDLLFile.p	2023/10/17 15:11	MATLAB.p.9.14.0	6 KB
 MavLinkStruct.mat	2023/10/17 15:11	MATLAB Data	5 KB
 Readme.docx	2023/10/17 15:11	Microsoft Word ...	8,760 KB
 Readme.pdf	2023/10/17 15:11	Foxit PhantomP...	1,109 KB
 Exp2_MaxModelTemp_init.m	2023/10/23 14:15	Objective C 源文件	3 KB



Step 2:

在 QGC 中可以如之前常规步骤一样，控制飞机的起飞、降落、航路飞行等。





注：由于本接口使用地形模块，利用 TerrainZ 实现了从 CopterSim 中读取当前地形高度数据，因此能适用于 Grassland 等带地形的地图。

6.3. 硬件在环仿真

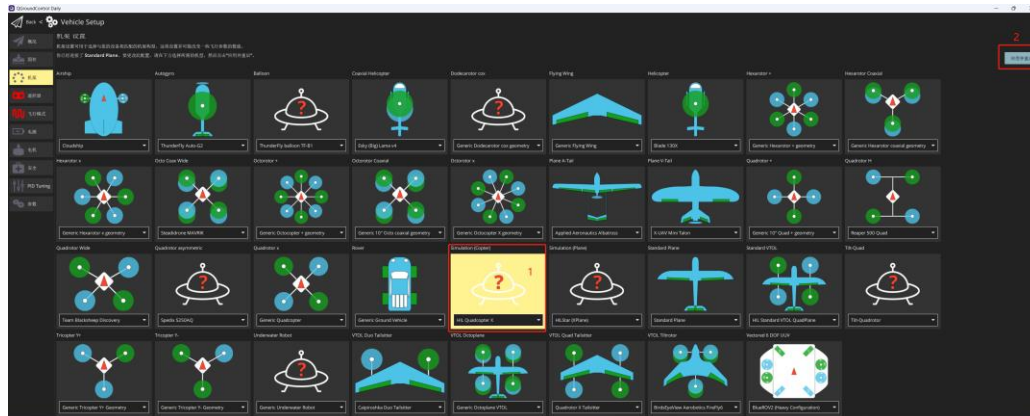
Step 1:

按下图所示将飞控与计算机链接。



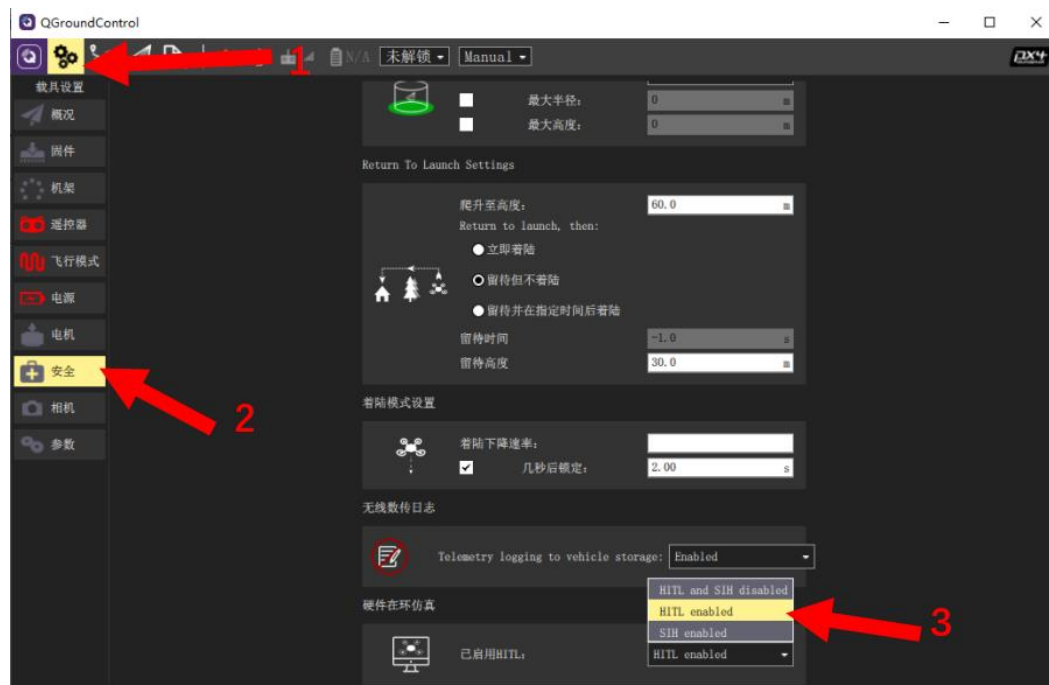
Step 2:

推荐使用 Pixhawk 6C 飞控进行硬件在环仿真，将飞控烧录至 1.13.3 固件版本，机架设置为 “HIL Quadcopter X”，点击 QGC 右上角的 “应用并重启”。



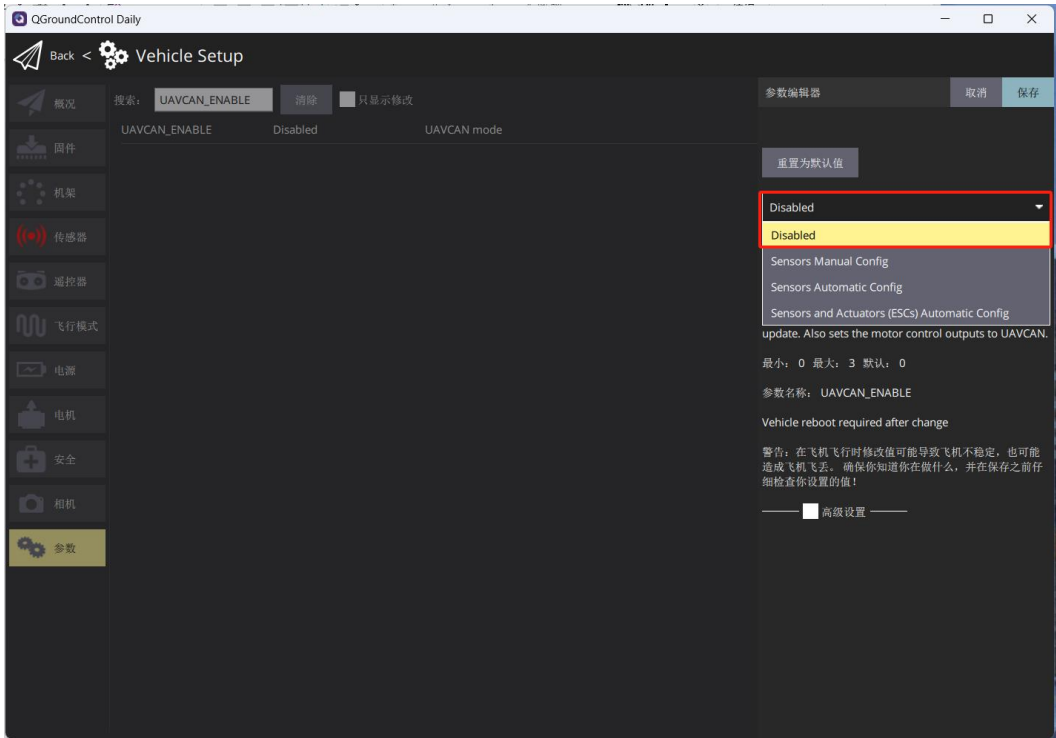
Step 3:

在“安全”界面，选择“HITL enabled”启动硬件在环仿真，之后重新插拔飞控完成设置。



Step 4:

点击“参数”，在搜索栏中输入“UAVCAN_ENABLE”，在弹出框中设置为“Disabled”，保存后重新插拔飞控即可。



Step 5:

运行“Exp2_MaxModelTempHITL.bat”脚本，可以自动打开硬件在环仿真，并完成所有需要配置，在弹出的窗口输入对应串口号，启动一架飞机的硬件在环仿真。

名称	修改日期	类型	大小
Exp2_MaxModelTemp.dll	2023/6/25 9:38	应用程序扩展	241 KB
Exp2_MaxModelTemp.slx	2023/6/8 17:44	Simulink Model	118 KB
Exp2_MaxModelTempHITL.bat	2022/9/20 17:07	Windows 批处理...	6 KB
Exp2_MaxModelTempSITL.bat	2022/9/20 17:07	Windows 批处理...	6 KB
GenerateModelDLLFile.p	2022/12/18 16:35	MATLAB P-code	5 KB
Init.m	2022/10/10 17:50	MATLAB Code	6 KB
MavLinkStruct.mat	2022/5/9 10:27	MATLAB Data	5 KB
Readme-最大模板验证.docx	2023/6/25 14:45	Microsoft Word ...	6,445 KB

Step 6:

之后测试步骤参照 5.2 软件在环中 Step2 步骤进行。

7. 参考资料

[1]. 无。

8. 常见问题

Q1: ****

A1: ****
