本章目录

用API与SDK组织可重复的模型实验

第四章 载具运动建模与仿真

4.1 运动建模基础与运动学
4.2 刚体动力学与运动求解
4.3 动力单元、受力与观测建模
4.4 配平、线性化与动态系统基础
4.5 多旋翼模型逐步编程与仿真
4.6 参数测量与静态模型辨识
4.7 动态系统辨识、验证与模型修正
4.8 典型无人载具建模与扩展
4.9 建模工具、代码生成与模型部署
4.10 API与SDK建模接口
4.11 例程体系与综合实践
4.12 习题案例分析与课程设计

本节 6 个教学单元

4.10.1 按任务选择入口
教材对应：第4.2.6节（二次开发接口）

4.10.2 端口、字段与单位
教材：4.2.3、4.2.5.2–4.2.5.3节

4.10.3 配置与可重复初始化
教材对应：第4.2.3节（初始化与场景配置）

4.10.4 外部输入与日志
教材对应：第4.2.3节（扩展输入与自定义输出）

4.10.5 直接加载与时间推进
教材对应：第4.2.4节（综合模型的Python调用）

4.10.6 生命周期与故障定位
实验生命周期、资源清理与输出不变的定位

4.10.1 按任务选择入口：接口职责

API与SDK：按实验任务选入口

- 应用编程接口（API）规定调用名称、参数与行为；软件开发工具包（SDK）提供封装、文档与例程。
- 实验从相同初态改变一个已支持输入，再读取新状态；输入的物理含义取决于当前模型内部接线。

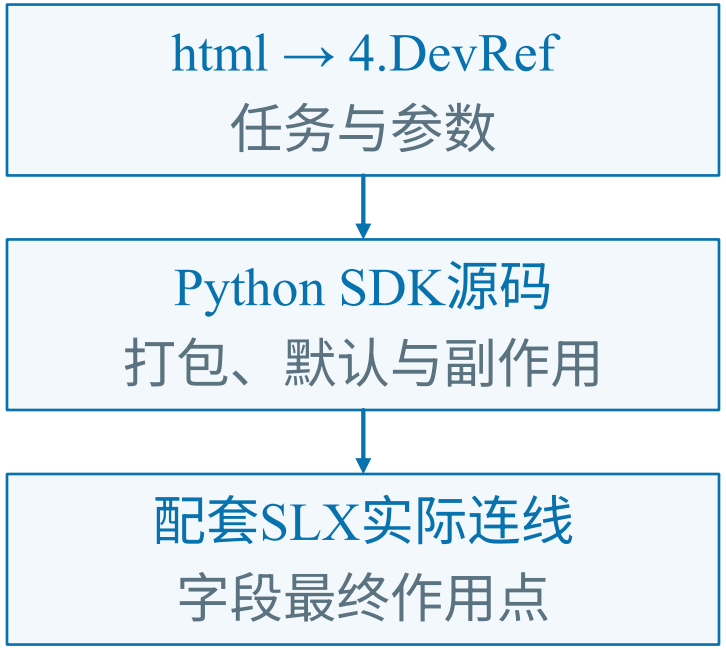
任务	入口	运行位置
配置初态与场景	ReqCopterSim	请求发往运行中的CopterSim
发送扩展输入、读回状态	DllSimCtrlAPI网络类	CopterSim推进模型，Python经UDP收发
每次精确推进若干步	ModelLoad直接加载	DLL在Python进程内运行

接口规定如何交换数据，开发包把这些约定组织成可调用工具。

4.10.1 按任务选择入口：本地文档与源码

从本地文档找到实际实现

- 打开[安装目录]/RflySimAPIs/RflySimSDK/html/index.html，它会转到4.DevRef/index.html；从Python SDK的ctrl分类进入具体接口页。
- 先读方法解决的任务和参数说明，再打开同名Python源文件，核对默认值、打包方式、返回值以及构造或调用的副作用。
- 最后打开0.ApiExps中的配套SLX，沿输入连线确认字段用途；同一SDK函数对不同模型未必具有相同作用。



文档帮助导航，当前源码与配套模型给出最后的接口约定。

4.10 API与SDK建模接口

4.10.1 按任务选择入口：Python环境

先使用安装版的配套Python环境

```
Python3.8 environment has been set with openCV+pymavlink+numpy+pyulog etc.
You can use pip or pip3 command to install other libraries
Put Python38Run.bat into your code folder
Use the command: 'python XXX.py' to run the script with Python
```

图：配套Python终端入口示意

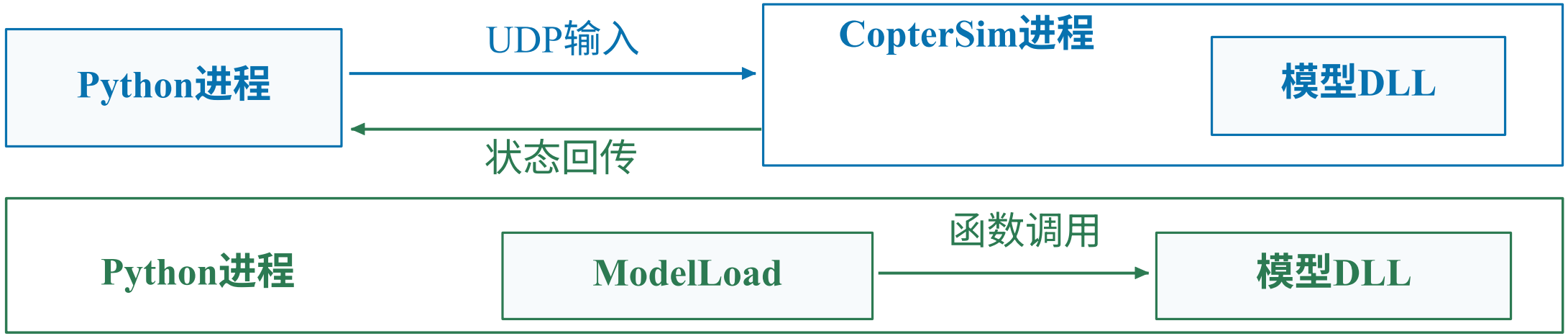
- 在例程目录运行Python38Run.bat，使用安装版配套解释器；记录Python版本、SDK来源与模型文件。
- 导入失败先查解释器与SDK路径；DLL加载失败查位数、依赖与文件位置，不能统称网络问题。

解释器、模块路径和模型文件要来自一套可追溯的环境。

4.10 API与SDK建模接口

4.10.1 按任务选择入口：同模块不同角色

同一模块里的两个类有不同边界



图：上路跨进程传输，下路直接调用DLL

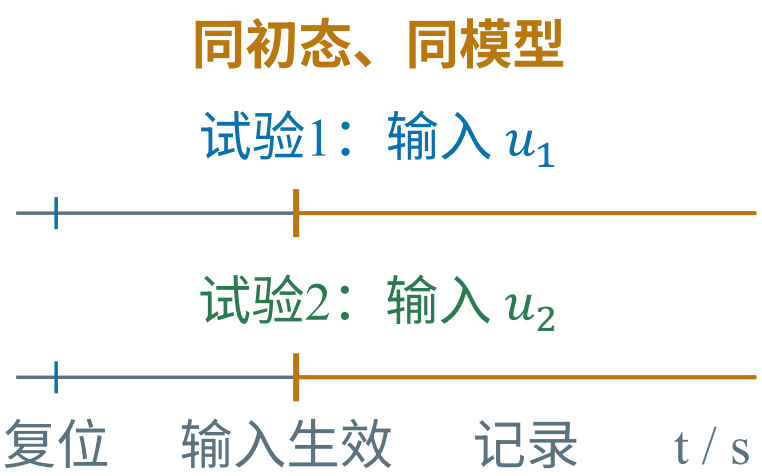
- 网络类通过UDP与CopterSim交互，由CopterSim推进模型。
- ModelLoad在Python内加载；MultiCopterDll是内含控制器的子类。

网络类发送数据，ModelLoad类加载并推进模型。

4.10.1 按任务选择入口：短实验设计

一个短实验只回答一个接口问题

- 配置实验回答初态是否生效，输入实验回答某字段是否进入指定连线，时间实验回答调用是否推进了模型状态。
- 每次只改变一项已支持的配置或输入，并写出预期变化；同时切换模型、地图、通信模式和控制命令会掩盖因果。
- 保存调用参数、输入生效时刻、带时间的输出与结束状态；只有一张正常显示的飞机截图，无法说明数据路径是否正确。



保存输入生效时刻与响应

固定模型和初态，输入、观察量和成功判据一一对应。

本章目录

用API与SDK组织可重复的模型实验

第四章 载具运动建模与仿真

4.1 运动建模基础与运动学
4.2 刚体动力学与运动求解
4.3 动力单元、受力与观测建模
4.4 配平、线性化与动态系统基础
4.5 多旋翼模型逐步编程与仿真
4.6 参数测量与静态模型辨识
4.7 动态系统辨识、验证与模型修正
4.8 典型无人载具建模与扩展
4.9 建模工具、代码生成与模型部署
4.10 API与SDK建模接口
4.11 例程体系与综合实践
4.12 习题案例分析与课程设计

本节 6 个教学单元

4.10.1 按任务选择入口
教材对应：第4.2.6节（二次开发接口）

4.10.2 端口、字段与单位
教材：4.2.3、4.2.5.2–4.2.5.3节

4.10.3 配置与可重复初始化
教材对应：第4.2.3节（初始化与场景配置）

4.10.4 外部输入与日志
教材对应：第4.2.3节（扩展输入与自定义输出）

4.10.5 直接加载与时间推进
教材对应：第4.2.4节（综合模型的Python调用）

4.10.6 生命周期与故障定位
实验生命周期、资源清理与输出不变的定位

4.10.2 端口、字段与单位：接口约定的检查项

模型端口值的结构、坐标和时间怎样解释

- **接口约定：**模块之间约定的输入输出、单位、坐标系、频率与时序；对象、尺寸与类型说明数据属于谁、怎样组织。
- 坐标确定方向和原点，单位确定尺度；同一数组中各段的坐标与单位可以不同。
- 时间说明样本何时有效：记录模型时间戳，与程序接收时刻分开保存。

对象	**载具1**
表示	**数组／总线**
坐标单位	**NED，m**
有效时间	**模型时间戳**

表：检查项与本例取值

对象、尺寸、类型、坐标、单位和时间共同定义数据含义。

4.10.2 端口、字段与单位：向量和总线的不同组织

根端口、向量和总线如何对应

- 最小模型M0（课件4.3起使用）根输入为inPWMs[16]和TerrainIn15d[15]；根输出为HILSensor30d、HILGPS30d和60维VehileInfo60d。
- 向量通过下标区分元素，总线通过信号名区分内容；6DOF总线含速度向量、位置向量和姿态矩阵等多种对象。
- 总线项数不等于状态维数，30维数组也不是30个同类传感器；逐个展开本次使用的字段再连接。

端口连接数据通道，总线按名称组合多种信号。

4.10.2 端口、字段与单位：直接60维数组

直接60维输出中的常用字段

- 直接60维输出由Data3D.update解析。Python切片从0起，右端下标不包含在结果中。
- 例如位置是6:9，共3项；优先读out3Ddata的命名属性，避免每段脚本重写下标。
- 右侧表格只列常用字段；网络包由另一解析器读取，偏移不同。

2	模型时间	runnedTime
3:6	NED速度	VelE
6:9	NED位置	PosE
9:12	欧拉角	AngEuler
12:16	四元数	AngQuatern
16:24	电机转速	MotorRPMS
27:30	机体系角速	RateB

表：Python切片、含义与out3Ddata属性

直接DLL向量按已定义的索引解释，优先使用解析后的属性。

4.10.2 端口、字段与单位：网络封装差异

网络包与DLL数组不能共用下标

- 直接60维向量的时间在下标2；网络真值包按4i24f7d（4个int、24个float、7个double）解包，前导字段含校验和与编号，时间在另一位置。
- SDK分别用Data3D.update与updateUdp解析两种表示；自行读原包时，核对消息长度、格式和校验字段。
- 有名属性统一了两种读法，解析前仍按源数据的接口约定核对；自定义输出是另一种32维消息。

直接DLL：60维

…	时间	速度	位置

下标2

网络：4i24f7d

校验	编号	…	状态

相同物理量经过网络封包后，前导字段和数据排列可能变化。

4.10.2 端口、字段与单位：两套模板的根端口

两个固定翼模板的根端口名称与顺序并不相同

e2_FixWingModelCtrl	
入1	inPWMs
入2	inCopterData
入3	TerrainIn15d
出1	HILSensor30d
出2	HILGPS30d
出3	VehileInfo60d

e5_FixedWingManuscriptMdl	
入1	inPWMs
入2	TerrainZ
入3	inCopterData
出1	MavHILSensor
出2	MavHILGPS
出3	MavVehile3DInfo

接口约定：模块间约定的输入输出、单位、坐标系、频率与时序；按所选模型逐项接线。

模板适配要按实际接口约定，不能只复制前一个模型的接线。

4.10.2 端口、字段与单位：车辆根输出的排序

车辆根输出排序不同，迁移时必须逐个对应

共同输入：inPWMs / inCopterData / TerrainIn15d

模型	Out1	Out2	Out3
差动车	HILSensor30d	HILGPS30d	VehileInfo60d
阿克曼	VehileInfo60d	HILSensor30d	HILGPS30d

真值：差动车为Out3，阿克曼为Out1；逐项核对名字、维数和物理意义后再接线。

输出的职责相似，序号不同仍会造成传感器与真值接错。

4.10.2 端口、字段与单位：归一化输入与物理量

归一化执行器输入不等于轮速，也不等于转角

- 模板执行器输入inPWMs是归一化量；12 rad/s、2 m/s或26.565°要经内部映射才对应轮速、转角与驱动力。
- 差动车轮速换算的局部角速度为左转正；与本章NED右转正比较时，应在接口明确转换符号。
- 阿克曼分支使用转向、驱动及固定尺度系数；MotorRads只是变量名，各元素单位按其计算关系逐个确认。



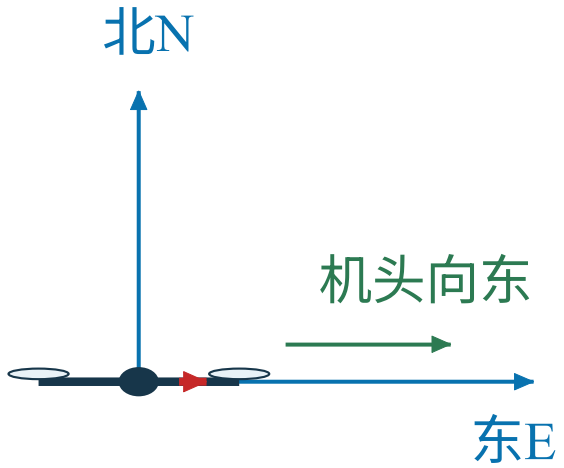
图：先读映射，再定物理单位

输入尺度和正号必须经过所选模板的映射才能对应物理量。

4.10.2 端口、字段与单位：坐标与正号

NED与FRD怎样进入接口值

- NED按北、东、下排列，位置用米、速度用米每秒；向上1 m/s的NED速度写成(0,0,−1)。
- FRD按机头前、右、下排列；机头朝东时，向前速度即地面向东，同一数组不能当NED直接发送。
- 还要核对位置相对哪个原点；ModelLoad的SendPosNED在当前实现中会加InitPos，局部位置与全局位置读数可能因此不同。



向上：$v_D = -1$

$$\mathbf{v}_{NED} = [0,0,-1]^T \ \mathrm{m/s}$$

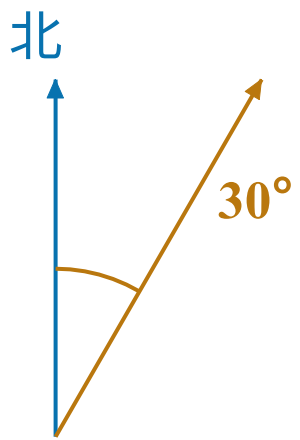
方向和原点明确以后，数字才对应唯一的物理指令。

4.10.2 端口、字段与单位：角度单位

初始化角度与状态角度不同单位

- XYyaw与XyzRPYaw初态请求使用度；模型欧拉角和Data3D.AngEuler状态使用弧度。
- 请求30度航向，状态应接近0.5236 rad；先换算，再用首次新样本确认生效。
- 机体系角速度用rad/s，与姿态角不同；日志列保留物理量、方向和单位。

$$\psi_{rad} = \psi_{deg}\frac{\pi}{180}, \qquad 30^\circ = 0.5236\ \text{rad}$$



读回0.5236 rad

配置入口用度，状态输出用弧度，转换应在接口边界完成。

4.10.2 端口、字段与单位：编码尺度

转速和GPS编码也有尺度转换

- 受力计算的转速用rad/s，MotorRPMS用rpm；470 rad/s约为4488.2 rpm，换算后才可比较。
- 全球定位系统（GPS）将经纬度乘10^7编码，高度用mm、速度用cm/s；解码后再与模型状态比较。
- 50 m海拔为50000 mm，1 m/s为100 cm/s；还需区分NED向下与海拔向上及原点。

$$n_{rpm} = \frac{60}{2\pi}\omega_{rad/s}$$

算例：单位换算

470 rad/s × 60/(2π)
≈ 4488.2 rpm

50 m × 1000
= 50000 mm

1 m/s × 100
= 100 cm/s

物理量、显示单位与传输整数编码需要分别标注。

4.10.2 端口、字段与单位：新样本判据

怎样判断读到了新状态

- runnedTime表示模型运行时间，单位为秒；每次记录保存这个时间，并与上一次样本比较，识别重复数据。
- 墙钟接收时间可用于分析网络与调度延迟，但不能替代仿真时间做轨迹对齐；两种时间分别保存才能解释差异。
- 复位可能使仿真时间回到起点，应新建一轮试验编号；若只要求时间单调增加，会把合法复位误判为坏数据。

相同时间：旧缓存

接收1 → 0.10 s

接收2 → 0.10 s

接收3 → 0.12 s

接收4 → 0.00 s

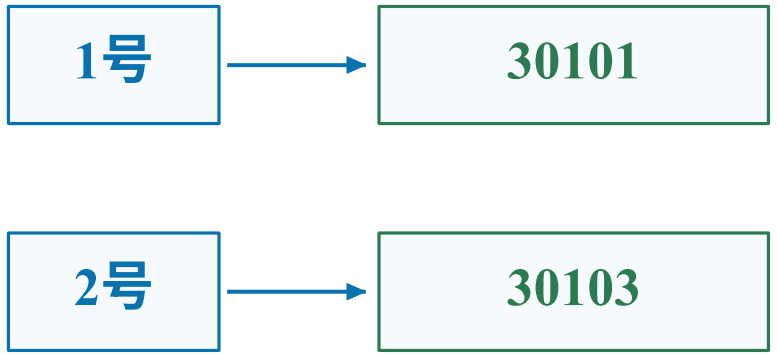
复位→新轮次

仿真时间变化说明样本更新，接收时间只说明程序何时读到。

4.10.2 端口、字段与单位：网络对象编号

网络真值的对象编号需要额外核对

- 当前网络解析updateUdp只更新运动状态与时间，没有写入Data3D的CopterID；该属性可能仍为构造时默认值0。
- 因此确认目标要结合发现列表、接收端口和原始包中的对象编号，out3Ddata.CopterID只作参考。
- 对1号载具监听30101，对2号监听30103；若自行解析原包，同时核对校验和、消息中的编号与发送端地址。



属性CopterID可能仍为0

按发现记录与数据端点确认目标，不能信任未更新的默认字段。

本章目录

用API与SDK组织可重复的模型实验

第四章 载具运动建模与仿真

4.1 运动建模基础与运动学
4.2 刚体动力学与运动求解
4.3 动力单元、受力与观测建模
4.4 配平、线性化与动态系统基础
4.5 多旋翼模型逐步编程与仿真
4.6 参数测量与静态模型辨识
4.7 动态系统辨识、验证与模型修正
4.8 典型无人载具建模与扩展
4.9 建模工具、代码生成与模型部署
4.10 API与SDK建模接口
4.11 例程体系与综合实践
4.12 习题案例分析与课程设计

本节 6 个教学单元

4.10.1 按任务选择入口
教材对应：第4.2.6节（二次开发接口）

4.10.2 端口、字段与单位
教材：4.2.3、4.2.5.2–4.2.5.3节

4.10.3 配置与可重复初始化
教材对应：第4.2.3节（初始化与场景配置）

4.10.4 外部输入与日志
教材对应：第4.2.3节（扩展输入与自定义输出）

4.10.5 直接加载与时间推进
教材对应：第4.2.4节（综合模型的Python调用）

4.10.6 生命周期与故障定位
实验生命周期、资源清理与输出不变的定位

4.10.3 配置与可重复初始化：目标发现

发现实例以后再发送配置

- ReqCopterSim构造时监听心跳并建立实例记录，getAllCopterID和getSimIpList可以帮助查看编号及地址。
- getSimIpID会查找并尝试广播查询，找不到时存在回退行为；返回一个可打印的IP，不自动证明目标实例已经发现。
- 先启动SITL（软件在环，课件4.9已介绍）并确认目标编号，再创建请求对象；未发现目标先查运行状态与网卡。

```
Start listening CopterSim heartbeat Msg ...
End listening CopterSim heartbeat.
Got 1 CopterSim on the LAN.
[(1, '192.168.1.100')]
192.168.1.100
发送联机模式重设命令
```

图：发现编号与IP（例程说明截图）

发现结果说明当前有哪些实例，默认IP只是一种寻址回退。

4.10.3 配置与可重复初始化：发现代码

短代码：验证目标出现在发现列表

- 先启动目标SITL；未发现目标时停止配置。

```
import ReqCopterSim
req = ReqCopterSim.ReqCopterSim()
vehicle_id = 1
if vehicle_id not in req.getAllCopterID():
    raise RuntimeError("Target not discovered")
print(req.getSimIpList())
print(req.getSimIpID(vehicle_id))
```

- **目标编号：**先查getAllCopterID是否含1；未发现则停止配置。

- **返回地址：**先核编号，再读IP列表与该目标IP；默认地址回退不能证明目标已发现。

4.10.3 配置与可重复初始化：例程入口

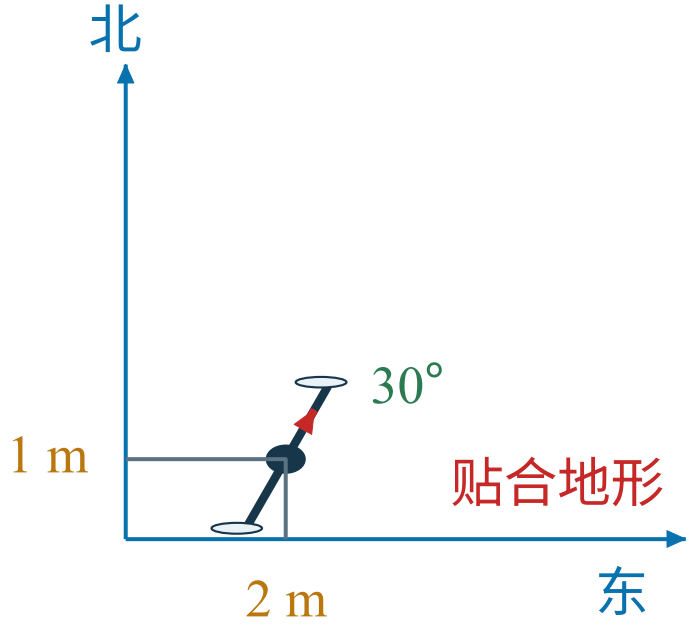
例程：用ReqCopterSim重设初态与场景

- **目录：**[安装目录]/RflySimAPIs/4.RflySimModel/0.ApiExps/4.InitAPI/1.ReqCopterSim/，先读其中的Readme.pdf。
- **入口：**MulticopterNoCtrl_SITLRun.bat、Python38Run.bat、testReqCopterSim.py。
- **步骤：**把HexarotorModelCTRL.dll复制到[安装目录]/CopterSim/external/model；以管理员身份运行SITLRun.bat并输入1；初始化完成后在Python38Run.bat终端运行python testReqCopterSim.py。
- **现象：**UDP_Mode由UDP_Full变为MAVLink_Full；x、y、yaw初值改变，UE中载具突变；模型换成六旋翼；地图由Grasslands变为VisionRing。
- **副作用：**bat启动时先结束已运行的CopterSim，按任意键退出时关闭CopterSim、QGC和RflySim3D；脚本最后会解锁并以Offboard飞向z=−20 m。
- **排查：**换模型无效先查external/model中有无目标DLL；发现列表为空先查SITL是否在运行、所用网卡是否正确。

4.10.3 配置与可重复初始化：平面初态请求

只请求一次XY与航向重置

- 已发现1号实例后，请求北1 m、东2 m、航向30度；此入口没有显式设置高度。
- 高度不由本请求指定：源码默认贴合地形、滚转俯仰置零；读首次新样本确认位置与航向。
- 接PX4时还需估计初始化就绪；重置初态之后，飞行器仍会继续运动。



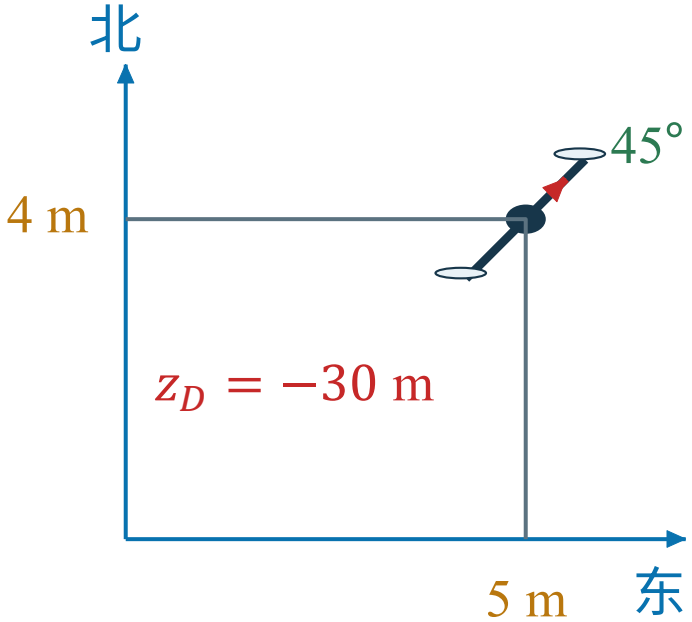

```
req.sendReSimXYyaw(1, [1, 2, 30])
```

一次请求只改变已声明的初态，便于比较前后数据。

4.10.3 配置与可重复初始化：完整初态请求

需要高度和姿态时使用明确的入口

- **[4,5,−30]是NED位置：**相对局部原点上方30 m，不是海拔30 m；姿态请求仍使用度。
- 读最早的新样本检验初始化。没有支持重量的推力时，飞机从空中初值继续下降，复位不会保持位置。




```
req.sendReSimXyzRPYaw(1, [4, 5, -30], [0, 0, 45])
```

XYZ和滚转、俯仰、偏航分成两个向量，单位分别为米和度。

4.10.3 配置与可重复初始化：复位读回

复位后的确认需要三个信号

- 复位后建立新试验编号，核对首次新样本的位置与姿态；时间递增只说明模型在推进。
- 接PX4时等EKF（扩展卡尔曼滤波）估计就绪；等待设上限，超时保存最后状态。
- 同初态与输入重复两次，在共同时间上比较；有噪声时固定随机条件或比较统计量。

新轮次／时间复位

↓

首次新位置与姿态

↓

PX4：EKF就绪

相同输入，再试一次

新轮次、新初态和估计就绪分别回答不同问题。

4.10.3 配置与可重复初始化：模型与地图配置

地图、模型和GPS原点怎样管理

- 更换DLL改变模型结构或参数，更换地图可能改变地形和参考环境；两者都会影响重复试验，要与控制命令一并记录。
- GPS原点用sendReGPSOrin(ID, [纬度°, 经度°, 海拔m])，海拔向上为正；地图、DLL分别用sendReSimMapName、sendReSimDllName。
- 配置入口不可用时，可使用同一启动配置重新启动基准；只要初态、输入与记录条件一致，仍能练习重复性验证。

会改变比较条件的配置，一次只变一项并记录来源。

4.10.3 配置与可重复初始化：两类初始化入口

启动参数初始化与运行中网络重置

- **模板参数：**前三行写在模型的*_init.m（例程2.AdvExps/e1_MaxModelTemp）中，编译时载入，CopterSim启动时改写部分分量；第四行是运行中请求。

入口	单位	何时生效、由谁改写
ModelInit_PosE [x, y, z]	m，NED	启动时：x、y换成CopterSim界面值，z取地图地形高度
ModelInit_AngEuler	rad	启动时：滚转、俯仰取此值，偏航由CopterSim界面填入
ModelParam_GPSLatLong 与envAltitude	度； m，向下为正	启动时：定义GPS原点，可被地图txt中的坐标改写
sendReSimXYyaw、sendReGPSOrin、 sendReSimXyzRPYaw	m与度	运行中：请求CopterSim重置，读首个新样本确认

本章目录

用API与SDK组织可重复的模型实验

第四章 载具运动建模与仿真

4.1 运动建模基础与运动学
4.2 刚体动力学与运动求解
4.3 动力单元、受力与观测建模
4.4 配平、线性化与动态系统基础
4.5 多旋翼模型逐步编程与仿真
4.6 参数测量与静态模型辨识
4.7 动态系统辨识、验证与模型修正
4.8 典型无人载具建模与扩展
4.9 建模工具、代码生成与模型部署
4.10 API与SDK建模接口
4.11 例程体系与综合实践
4.12 习题案例分析与课程设计

本节 6 个教学单元

4.10.1 按任务选择入口
教材对应：第4.2.6节（二次开发接口）

4.10.2 端口、字段与单位
教材：4.2.3、4.2.5.2–4.2.5.3节

4.10.3 配置与可重复初始化
教材对应：第4.2.3节（初始化与场景配置）

4.10.4 外部输入与日志
教材对应：第4.2.3节（扩展输入与自定义输出）

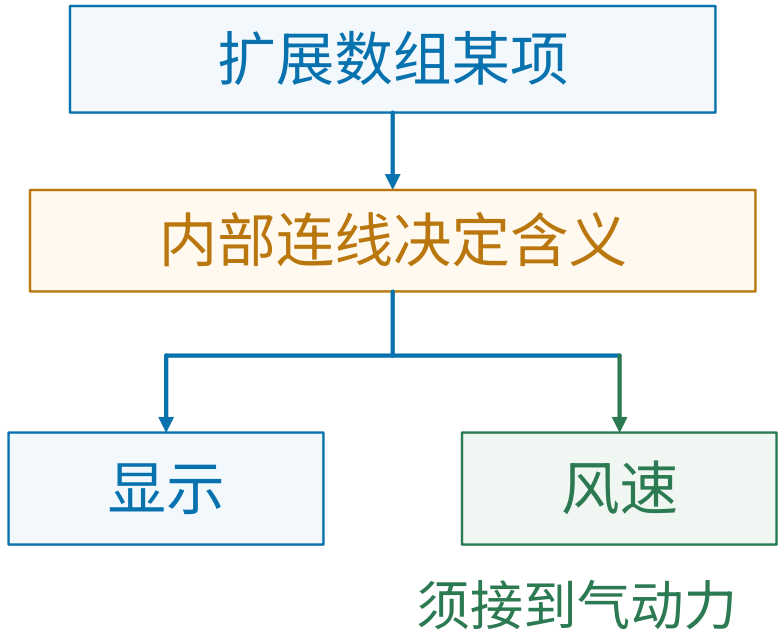
4.10.5 直接加载与时间推进
教材对应：第4.2.4节（综合模型的Python调用）

4.10.6 生命周期与故障定位
实验生命周期、资源清理与输出不变的定位

4.10.4 外部输入与日志：输入最终作用点

外部输入由内部连线获得含义

- 发送函数只组织类型与长度；数字改变质量、风速、故障还是显示，由当前SLX连线决定。
- **同时记录端口收到的值和内部受影响量：**显示类别读3DType，风输入读相对空速与气动力。
- 输入已变化而受力没变，先查连线、单位和使能；可调参数公开也不自动建立网络写入路径。
- **其他通道：**FaultParamAPI.FaultInParams收故障参数，inCollision20d收碰撞信息。



把数字送到端口以后，还要知道它作用于哪一个模型环节。

4.10.4 外部输入与日志：扩展输入容量

sendSILIntFloat实际发送哪些量

- sendSILIntFloat把输入补齐或截断为8个整数、20个浮点，前加校验和与对象编号，按10i20f（10个int、20个float）打包。
- **输入端口为**$30100+2(ID-1)$：1号30100、2号30102；真值回传端口在下一奇数端口。
- 真值回传：1号监听30101、2号监听30103，即输入端口号加1。

$$P_{in}=30100+2(ID-1)$$

网络头：校验＋编号



8个整数



20个浮点数

协议约定数据怎样送达，模型连线决定数据怎样起作用。

4.10.4 外部输入与日志：例程入口

例程：用inSIL扩展输入改变显示模型

- **定义：** inSIL是DLL模型的外部整数／浮点输入口。
- **目录：** [安装目录]/RflySimAPIs/4.RflySimModel/0.ApiExps/11.inSILAPI/1.InSILIntsFloats/3.inSILIntsFloats_py/，先读其中的Readme.pdf。
- **入口：** MulticopterNoCtrl_SITLRun.bat、Python38Run.bat、inSIL.py。
- **步骤：** 以管理员身份运行SITLRun.bat并输入1；在Python38Run.bat终端运行python inSIL.py。改动SLX后须重新编译，并运行GenerateModelDLLFile.p生成DLL。
- **现象：** 把silInt[0]依次改为5、8、10、50，RflySim3D中的显示模型依次变为六旋翼、八旋翼、四轴八旋翼和小车。
- **副作用：** bat启动时先结束已运行的CopterSim，按任意键退出时关闭CopterSim、QGC和RflySim3D。
- **排查：** 外观不变先查MulticopterNoCtrl.slx中inSILInts与inSILFloats是否一并接通，再查CopterID与输入端口30100。

4.10.4 外部输入与日志：发送代码

短代码：明确填写第一项扩展输入

- 沿用inSIL例程的SITL实例，把首项设为5发送。

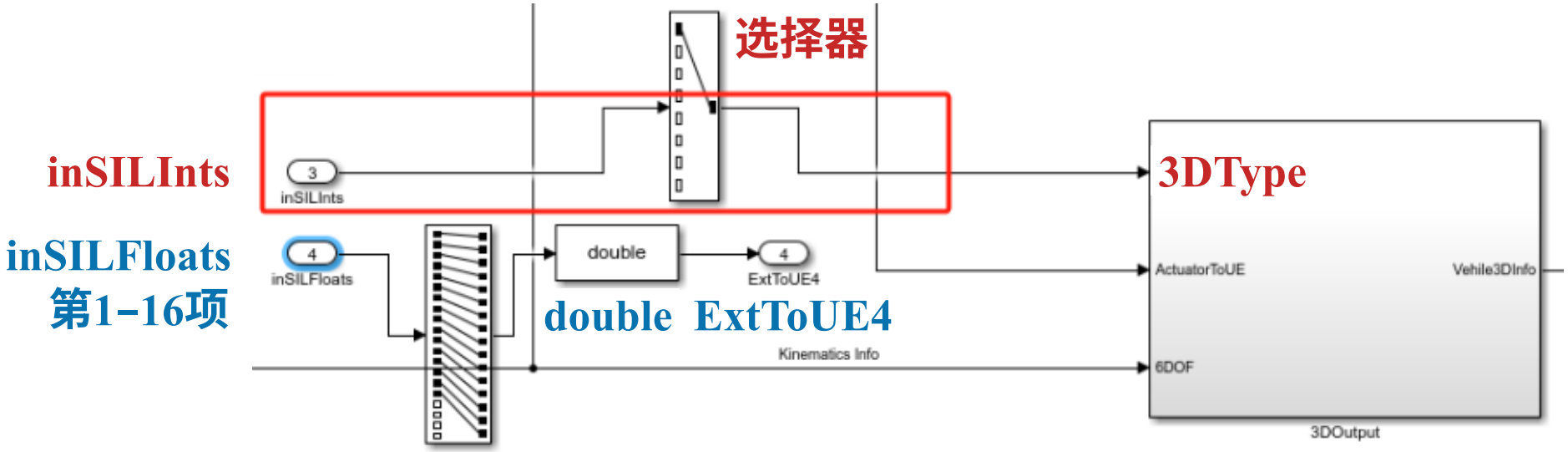
```
from DllSimCtrlAPI import DllSimCtrlAPI

link = DllSimCtrlAPI(CopterID=1)
ints = [0] * 8
floats = [0.0] * 20
ints[0] = 5
floats[0] = 5.0
link.sendSILIntFloat(ints, floats)
```

- **接口约定：**目标1号；整数8项、浮点20项，首项5与5.0，其余为零。
- **模型用途：**整数首项接3DType，现象是三维场景中外观改变。

4.10.4 外部输入与日志：显示类别连线

整数示例实际接向显示类别

- **整数通路（红框）**：inSILInts[1] → 选择器 → 3DType；[1]即Python下标0。
- **浮点通路**：inSILFloats第1–16项 → 选择器 → double → ExtToUE4。



图：两条扩展输入在例程SLX中的实际连线（局部）

从发送下标走到真实SLX连线，才能说明预期变化。

4.10.4 外部输入与日志：浮点连线

浮点示例按当前模型连接解释

- 当前示例的inSILFloats为20维，选择第1至16项并转成double后送到ExtToUE4输出，形成外部显示数据通道。
- ExtToUE4把这些量交给RflySim3D作显示或特效触发，不进入受力计算；整数首项另走3DType通路。
- 若要注入风速，需要在模型内把对应分量连接到相对气流或气动力环节，并新增单位、方向和响应验证。



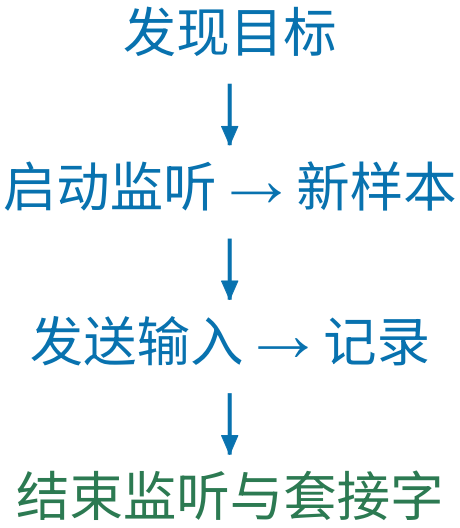
图：浮点扩展输入的实际通路

实际SLX比旧截图更能说明当前字段的去向。

4.10.4 外部输入与日志：读回运动状态

监听真值的开始与结束要配对

- 先启动CopterSim并发现目标，InitTrueDataLoop绑定回传端口、启动线程，随后确认新样本。
- 发送一个已支持字段，记录输入、模型时间与输出；CopterSim推进动力学，Python不调用Dllstep。
- 有限条件结束后，EndTrueDataLoop停止接收并关闭套接字；端口冲突先解决占用。



接收线程负责刷新属性，读取属性不会主动推进CopterSim。

4.10.4 外部输入与日志：读回运动状态

短代码：读回运动状态并识别新样本

```
import time, DllSimCtrlAPI
link = DllSimCtrlAPI.DllSimCtrlAPI(CopterID=1)
link.InitTrueDataLoop()
rows, t_end = [], time.time() + 5
try:
    while time.time() < t_end:
        d = link.out3Ddata
        if d.runnedTime != (rows[-1][0] if rows else 0):
            rows.append([d.runnedTime, *d.PosE])
        time.sleep(0.005)
finally:
    link.EndTrueDataLoop()
```

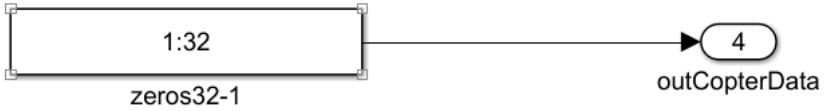
- **新样本：** runnedTime不同于上一行才追加，首行与默认值0比较；复位后按轮次分段。

- **有限结束：** 5 s上限让循环必然结束；finally中EndTrueDataLoop停止线程并关闭套接字。

4.10.4 外部输入与日志：自定义输出

outCopterData把内部量带到模型外

- outCopterData是32维自定义double输出；示例用1:32验证顺序，不是32个运动状态。
- **CopterSim1.csv对应1号载具：**检查新增行、列数和1至32的顺序；文件存在不代表正在采样。
- SDK将同一32项保存到outCopterVect，与out3Ddata分属不同消息；改装日志时同时保存字段与单位。



图：用已知序列核对自定义输出（例程说明截图）

自定义输出允许记录所关心的内部状态，但字段含义由模型定义。

4.10.4 外部输入与日志：例程入口

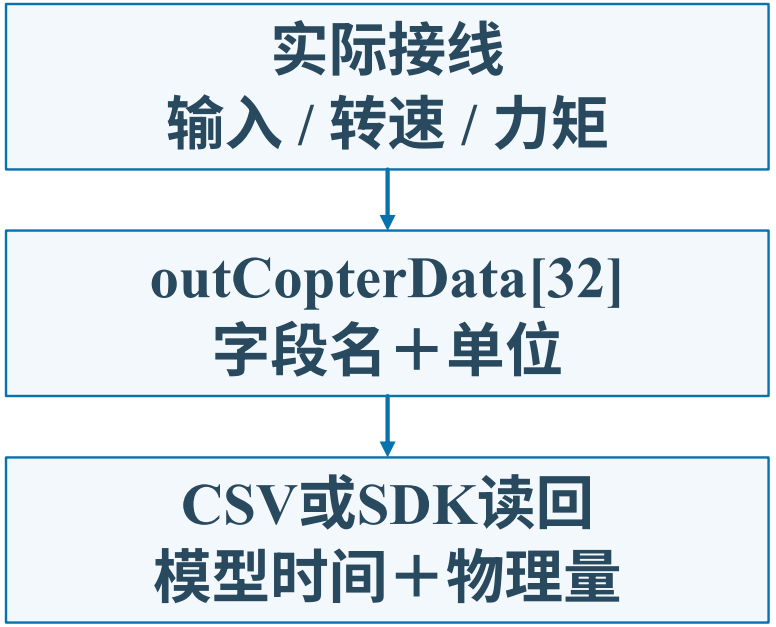
例程：outCopterData自定义输出与日志

- **目录：**[安装目录]/RflySimAPIs/4.RflySimModel/0.ApiExps/8.OutCopterData/，先读其中的Readme.pdf。
- **入口：**Exp2_MaxModelTempSITL.bat；字段在Exp2_MaxModelTemp.slx的LogSelectModel中定义，示例为1至32。
- **步骤：**先在[安装目录]/CopterSim新建空文件CopterSim1.csv；以管理员身份运行Exp2_MaxModelTempSITL.bat并输入1；仿真一段时间后在命令行窗口按任意键结束。
- **现象：**CopterSim1.csv出现新增数据，各列依次为1至32；同一组数据还经UDP发往30101系列端口，SDK存入outCopterVect。
- **副作用：**bat启动时先结束已运行的CopterSim，退出时关闭CopterSim、QGC和RflySim3D；每次实验换用新文件或先备份旧日志。
- **排查：**没有数据先查文件是否建在CopterSim目录、文件名编号是否等于CopterID；改动字段后须重新编译并生成DLL。

4.10.4 外部输入与日志：自定义日志字段

把内部信号变成一份可辨识的数据记录

- **先校验通道：**沿0.ApiExps/8.OutCopterData的1:32示例核对输出顺序；确认新增行后，再把常量换成需要的内部信号。
- **再定义字段：**例如记录模型时间、实际归一化输入、电机角速度、机体系力矩和角速度；每一列保存名称、单位、坐标与采样位置。
- **保持同步：**从同一个模型时间点输出这些量；显示刷新时间或电脑收包时间不能直接替代积分器的仿真时间。



图：内部信号记录到日志

字段定义把模型内部的物理量变成可复算的数据，而不是一串无名数字。

4.10.4 外部输入与日志：从日志走向辨识

从输入端口圈出待辨识系统

实际电机输入 —u_norm→ 驱动＋电机 —omega_rad_s→ 转速

- **执行器对象：** 输入是实际归一化电机输入，不能用位置目标代替。

实际机体系合矩 —$\mathbf{M}_B$ / N·m→ 刚体转动 —$\boldsymbol{\omega}_B$ / rad/s→ 机体系角速度

- **机体对象：** 合矩已含阻尼等外部作用，辨识阻尼时不能再减一次。

速度目标 —$\mathbf{v}_{\mathrm{ref}}$ / m/s→ 控制器＋被控对象 —$\mathbf{v}$ / m/s→ 实际速度

- **闭环对象：** 控制器已在路径中，结果要标明闭环边界（方法见课件4.7）。

图：不同日志入口对应不同待辨识系统

先圈定输入和输出之间的系统，再解释所辨识参数属于哪一段物理过程。

4.10.4 外部输入与日志：数据进入算法前的自检

导入日志后先检查时间与有效值

- **文件约定：**将刚才定义的三列导出为trial.csv：t_s、u_norm、omega_rad_s；此文件由本次模型实验生成。
- **先看数据：**检查样本数、有限值与时间递增；进入等间隔辨识前，还须检查采样间隔是否近似一致，再分析输入与响应（算法见4.7）。
- **处理规则：**发现重复、乱序或缺失数据时，保留原始日志，说明筛选与对齐方法，再另存供算法使用的数据。

先确认时间和数据有效，再让算法解释输入怎样引起输出变化。

4.10.4 外部输入与日志：数据进入算法前的自检

代码：检查日志时间与有限值

```
import numpy as np
a = np.genfromtxt(
    'trial.csv', delimiter=',', names=True)
t, u, w = a['t_s'], a['u_norm'], a['omega_rad_s']
assert t.size > 2
assert np.isfinite([t, u, w]).all()
assert np.all(np.diff(t) > 0)
print('样本数、时长:', t.size, t[-1] - t[0])
```

- **采样间隔：**严格递增不代表等间隔。
  等间隔辨识前检查采样间隔，并保留原始日志。

- **t.size > 2：**至少3项；不足先核对导出。
- **isfinite：**时间、输入、转速都须为有限值；排查NaN与Inf。
- **diff(t) > 0：**时间严格递增；重复或回退先查轮次与数据顺序。

本章目录

用API与SDK组织可重复的模型实验

第四章 载具运动建模与仿真

4.1 运动建模基础与运动学
4.2 刚体动力学与运动求解
4.3 动力单元、受力与观测建模
4.4 配平、线性化与动态系统基础
4.5 多旋翼模型逐步编程与仿真
4.6 参数测量与静态模型辨识
4.7 动态系统辨识、验证与模型修正
4.8 典型无人载具建模与扩展
4.9 建模工具、代码生成与模型部署
4.10 API与SDK建模接口
4.11 例程体系与综合实践
4.12 习题案例分析与课程设计

本节 6 个教学单元

4.10.1 按任务选择入口
教材对应：第4.2.6节（二次开发接口）

4.10.2 端口、字段与单位
教材：4.2.3、4.2.5.2–4.2.5.3节

4.10.3 配置与可重复初始化
教材对应：第4.2.3节（初始化与场景配置）

4.10.4 外部输入与日志
教材对应：第4.2.3节（扩展输入与自定义输出）

4.10.5 直接加载与时间推进
教材对应：第4.2.4节（综合模型的Python调用）

4.10.6 生命周期与故障定位
实验生命周期、资源清理与输出不变的定位

4.10.5 直接加载与时间推进：构造与真实起点

直接DLL例程的准备条件

- **目录：**[安装目录]/RflySimAPIs/4.RflySimModel/0.ApiExps/12.DllModelImport/4.ModelLoadMannual/入口ModelLoadApiTest.py与Python38Run.bat。
- ModelLoad用Python标准库ctypes加载DLL，再CreateVehicle设置位置、姿态、地形；构造时已初始化并首次推进。
- 记录构造后的真实时间和状态作为起点，不再重复CreateVehicle；依赖文件需与环境配套。

ctypes加载DLL
↓
CreateVehicle初始化
↓
首次推进
↓
读取真实起点

无需启动CopterSim，不代表无需模型文件、地图和配套库。

4.10.5 直接加载与时间推进：导出函数能力

先核对当前DLL支持的导出函数

- 不同DLL可以导出不同函数；当前SDK常用hasattr检查，缺失时可能跳过调用或返回默认值，所以要用导出检查确认支持。
- 逐步推进至少核对Dllstep、真实步长与状态输出入口；缺少关键函数时明确停止该实验，避免把零数组当作静止状态。
- DllGetStep0缺失时SDK会回退0.001 s，这个返回值不能证明当前DLL真实使用1 ms，需要从导出或生成配置取得证据。

导出函数	作用
Dllstep	推进一步
DllGetStep0	真实步长
DlloutVehileInfo60d	60维输出

默认值≠支持证据

Python方法存在，不代表所有DLL都实现对应计算入口。

4.10.5 直接加载与时间推进：起点读取代码

短代码：读取初始化后的真实起点

- 创建配套综合模型，检查基本步进与真值导出。

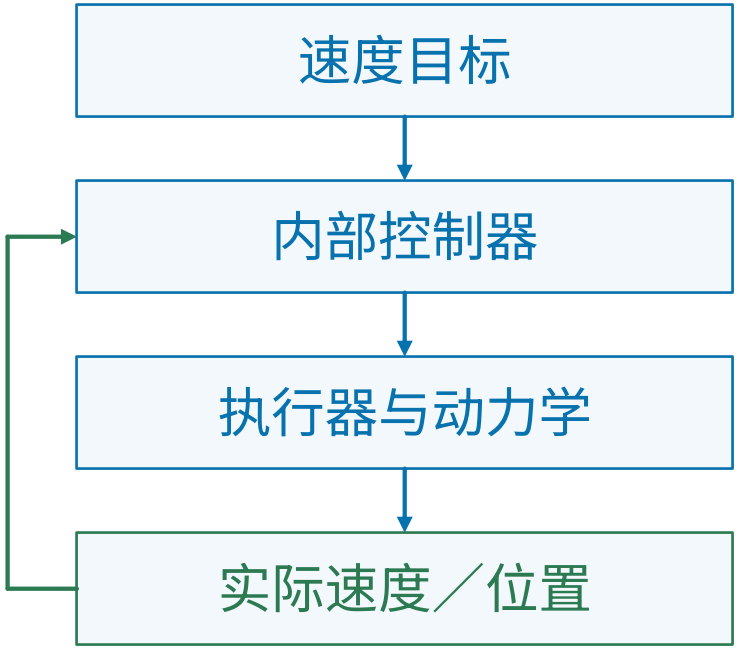
```
import DllSimCtrlAPI
model = DllSimCtrlAPI.MultiCopterDll()
required = ["Dllstep", "DllGetStep0",
            "DlloutVehileInfo60d"]
if not all(hasattr(model.dll, f) for f in required):
    raise RuntimeError("Required DLL export missing")
print(model.out3Ddata.runnedTime)
print(model.out3Ddata.PosE)
```

- **支持检查：** Dllstep、DllGetStep0及60维真值导出均须存在；缺失就停止本实验。
- **真实起点：** 构造已初始化并首次推进；保存runnedTime与PosE，不假定高度为零。

4.10.5 直接加载与时间推进：综合模型的速度目标

速度目标先经过综合模型控制器

- SendVelNED(0,0,−1)给出向上1 m/s的速度目标；内部控制器根据状态误差产生执行器动作，再由动力学计算实际运动。
- 实际速度带滞后地趋近目标，位移通常小于目标速度乘推进时间；用VelE和PosE读回实际响应。
- 综合响应包含控制器与动力学；辨识裸电机或刚体时，应改用直接作用于执行器的测试输入。

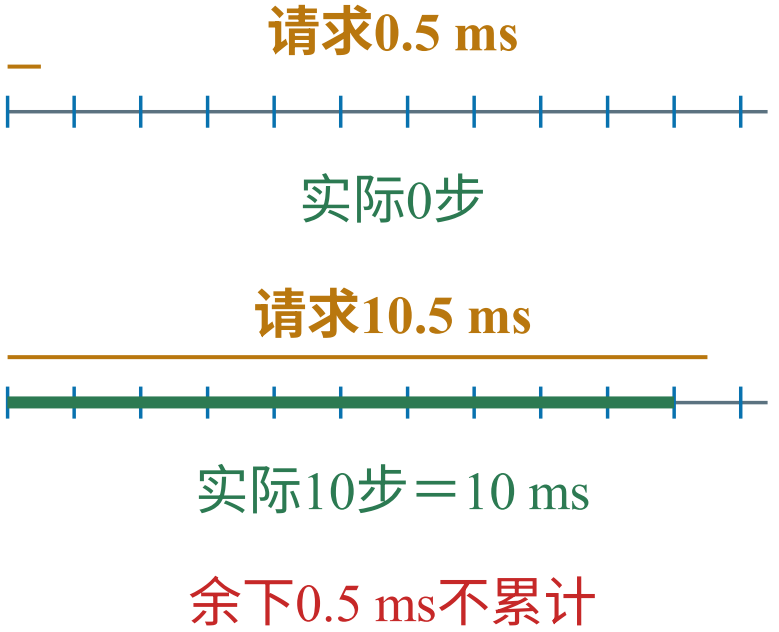


命令给出期望运动，实际速度和位移由闭环响应形成。

4.10.5 直接加载与时间推进：时间取整

ModelUpdate怎样把时间变成步数

- ModelUpdate按$N = \mathrm{int}(\Delta t/h)$步推进，不保存小数余量；$h = 1$ ms时，0.5 ms推进0步，10.5 ms推进10步。
- 定量比较用手动整数步，并读回实际模型时间确认推进量。



当前实现取整数步推进，未满一步的余量不会累积到下次。

4.10.5 直接加载与时间推进：整数步代码

短代码：推进1000步再读回

- model沿用“读取初始化后的真实起点”代码；施加向上速度目标，推进1000步。

```
h, t0 = model.DllGetStep0(), model.out3Ddata.runnedTime
model.SendVelNED(0, 0, -1)
for _ in range(1000):
    model.Dllstep()
model.Updata3Doutput()
print("expected dt", 1000 * h)
print("actual dt", model.out3Ddata.runnedTime - t0)
print(model.out3Ddata.PosE)
```

- **请求：**向上1 m/s目标为NED(0,0,-1)；推进1000个基本步。
- **预测：**h=1 ms时actual dt应为1 s；VelE[2]带滞后趋近−1 m/s，上升通常不足1 m。

4.10.5 直接加载与时间推进：发布例程的输出

从发布示例看实际读回的量

- **命令：** SendVelNED(0,0,−10)后ModelUpdate(1)，再SendVelNED(10,0,0)后ModelUpdate(10)；每次打印localPos、PosE、VelE；目标−10 m/s，“推进1000步”代码用−1 m/s。

```
[1.6666424061912851e-16, 4.180508356564208e-17, -5.6306766265134165] [1.6666424061912851e-16, 4.180508356564208e-17, -13
.680676817248 28] [5.737159294107992e-17, 1.7120228701808106e-16, -9.866896579793057]
[78.56886111450616, -2.3218608213043204e-15, -9.955108932450404] [78.56886111450616, -2.3218608213043204e-15, -18.005109
123185267] [9.959637020888362, -4.772923161087532e-16, 0.02391732914028788]
```

图：手动推进后的状态输出（例程说明中的运行输出）

- **读数：** 1 s后局部z=−5.63 m、竖直速度−9.87 m/s，仍在加速；两组位置z相差约8.05 m，来自起点InitPos的地形高度。

示例输出的是状态响应，不是简单抄回目标命令。

4.10.5 直接加载与时间推进：自动更新与积分频率

自动更新的一轮会推进多少模型时间

- **调用关系：** StartAutUpdate默认约30 Hz组织外层更新，按墙钟间隔和倍率请求ModelUpdate；30 Hz是刷新率，每轮可含多个积分步。
- **计算示例：** 若这一轮请求0.0337 s、基本步为0.001 s，当前ModelUpdate取整后推进33步，即0.033 s。
- **记录方法：** 保存每轮前后模型时间和墙钟时间；自动播放观察趋势，定量重复试验优先明确整数步数。

$$N = \left\lfloor \frac{0.0337}{0.001} \right\rfloor = 33, \qquad \Delta t_{\text{model}} = Nh = 0.033\,\text{s}$$

显示节奏、请求推进量和实际模型时间是三个不同的记录量。

本章目录

用API与SDK组织可重复的模型实验

第四章 载具运动建模与仿真

4.1 运动建模基础与运动学
4.2 刚体动力学与运动求解
4.3 动力单元、受力与观测建模
4.4 配平、线性化与动态系统基础
4.5 多旋翼模型逐步编程与仿真
4.6 参数测量与静态模型辨识
4.7 动态系统辨识、验证与模型修正
4.8 典型无人载具建模与扩展
4.9 建模工具、代码生成与模型部署
4.10 API与SDK建模接口
4.11 例程体系与综合实践
4.12 习题案例分析与课程设计

本节 6 个教学单元

4.10.1 按任务选择入口
教材对应：第4.2.6节（二次开发接口）

4.10.2 端口、字段与单位
教材：4.2.3、4.2.5.2–4.2.5.3节

4.10.3 配置与可重复初始化
教材对应：第4.2.3节（初始化与场景配置）

4.10.4 外部输入与日志
教材对应：第4.2.3节（扩展输入与自定义输出）

4.10.5 直接加载与时间推进
教材对应：第4.2.4节（综合模型的Python调用）

4.10.6 生命周期与故障定位
实验生命周期、资源清理与输出不变的定位

4.10.6 生命周期与故障定位：网络宿主路线

网络实验的完整生命周期

- 网络路线四步依次为启动与发现、确认起点、推进与记录、停止与清理；接收代码见「短代码：读回运动状态并识别新样本」页。
- 结束条件写成有限的仿真时长或样本数，只关闭本次创建的资源，再保存日志与运行配置；下表与直接加载路线对照。

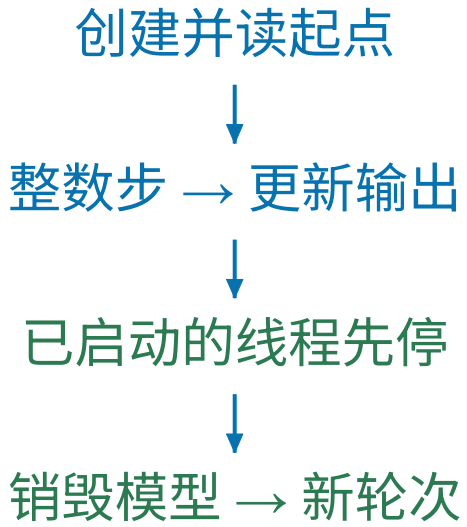
路线	启动与发现	确认起点	推进与记录	停止与清理
网络宿主	启动CopterSim，发现目标	InitTrueDataLoop，等runnedTime更新	CopterSim推进；发送输入并记录	EndTrueDataLoop关闭套接字
直接加载	创建模型，核对导出函数	读runnedTime与PosE起点	Dllstep整数步，Updata3Doutput	先停线程（若有），再DllDestroyModel

网络实验要把实际输入、模型时间与接收响应对应起来。

4.10.6 生命周期与故障定位：直接加载路线

直接加载实验怎样结束与重复

- 创建后核对导出函数、真实起点及输入边界；手动推进已知步数，再更新输出并保存实际时间。
- 只有启动过自动更新才先StopAutUpdate；之后DllDestroyModel释放内部模型资源。
- 清理本次创建的网络与显示对象；下一轮重新确认初态，不能把残余速度当成同条件重复。



停止已启动的线程，再销毁模型，并重新建立明确的下一轮初态。

4.10.6 生命周期与故障定位：异常退出也清理实例

代码：手动推进结束后销毁模型

```
try:
    model.SendVelNED(0, 0, -1)
    for _ in range(1000):
        model.Dllstep()
    model.Updata3Doutput()
    print(model.out3Ddata.runnedTime)
    print(model.out3Ddata.PosE)
finally:
    model.DllDestroyModel()
```

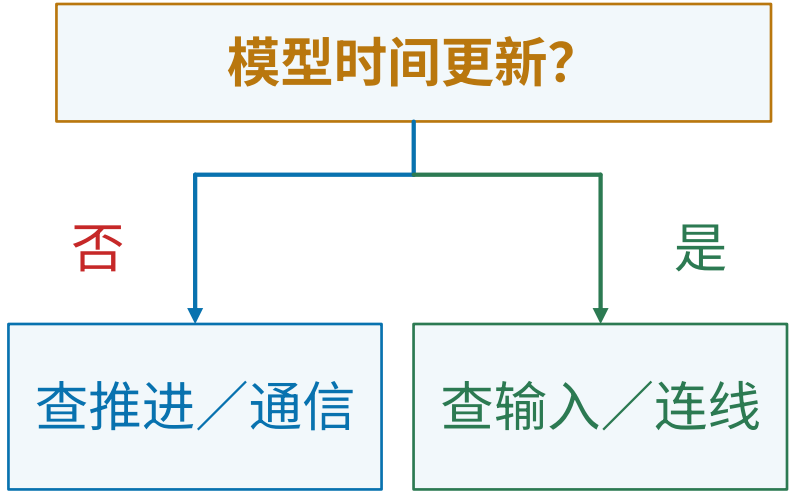
- **对象前提：** model来自“读取初始化后的真实起点”代码。
- **结束位置：** 正常或异常均执行finally；本例手动推进，无自动线程，直接销毁模型。

- **其他路线：** 已启动自动线程时先StopAutUpdate，再销毁模型；网络接收、显示对象由各自创建者清理，下一轮重新读取真实初态。

4.10.6 生命周期与故障定位：输入不生效的定位

发送完成但输出不变时怎样定位

- **时间不变：**查实例推进、端点与接收线程；时间更新但输入没到：查编号、端口、字段与内部连线。
- **方向或固定倍数量级错：**核NED/FRD、度/弧度、rpm与rad/s及GPS整数尺度，只转换一次。
- 状态像被限制到目标时，检查综合模型控制器边界。



方向、量级再核坐标单位

沿端点、端口接收和内部连线找第一处缺失证据。

4.10.6 生命周期与故障定位：接口阶段实验

阶段实验A3：形成一条完整证据链

- **A3选一路：**网络发送已定义扩展字段，或综合DLL给单轴速度目标；先预测，再记录实际输入、时间和响应。
- 保持模型、参数与初态，只改变一个输入，完成两次试验；提交字段表、短程序与共同时间轴曲线。
- A3是第四章第三次阶段实验，序列见课件4.3「阶段实验A1・执行器与合成」页；报告分清源码事实、预测、例程示例与本机记录。

网络路线

inSIL首项 → 3DType
→ RflySim3D外观变化

直接路线

SendVelNED → 1000步
→ 读VelE与PosE

只改变一个输入

明确初态并对齐输入与响应，才能解释一次接口实验的结果。

参考要点

阶段实验A3：先写预测，再按判据自查

- **网络·显示：**inSILInts首项改为5后，RflySim3D中显示为六旋翼（例程说明）；SDK网络解析当前未更新vehicleType，外观以三维显示验证。
- **网络·状态：**runnedTime递增只说明实例在推进；复位后按轮次分段。
- **记录路线：**CopterSim1.csv行数增长只属于outCopterData路线，不作inSIL判据。
- **直接·时间：**h=1 ms时推进1000步，预测actual dt≈1000h=1 s；偏小先查h与取整。
- **直接·运动：**预测VelE[2]带滞后趋近−1 m/s，1 s内上升通常不足1 m。
- **写报告：**标明每条结论来自源码、例程说明、预测还是本机记录；两次试验共用模型时间轴。

本节小结

本节小结：方法、关键判断与易错点

- **方法：**按任务选入口 → 核对接口约定（对象、坐标、单位、时间）→ 只改一个输入 → 读回新样本，按模型时间记录 → 按创建的逆序清理。
- **关键判断：**发送成功只说明数据已发出；输入含义由SLX连线决定；综合模型DLL内含控制器；runnedTime变化才是新样本。
- **易错点：**下面六处都在本节出现过，逐条对照自己的代码与记录。
- **下标：**60维数组与网络包偏移不同，解析器不能混用。
- **单位：**度与弧度、rpm与rad/s只在接口边界换算一次。
- **时间：**仿真时间与墙钟时间分开保存，复位后分段。
- **编号：**网络更新下out3Ddata.CopterID默认为0。
- **取整：**ModelUpdate按int(Δt/h)截断，余量不累计。
- **清理：**先停已启动的线程，再DllDestroyModel。