本章目录

建模工具、代码生成与模型部署

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 7 个教学单元

4.9.1 工具与运行链	教材4.2.1、4.2.2.3
4.9.2 MuJoCo多体建模（拓展）	多关节与接触力
4.9.3 最小模板与可重复工程	教材4.2.2.2、4.2.3
4.9.4 数值时间与调度	步长、多速率、墙钟
4.9.5 从SLX生成C++	教材4.2.3、4.2.6
4.9.6 封装、部署与加载	教材4.2.3、4.2.4
4.9.7 实现一致性与排错	SLX/DLL同条件比较

4.9.1 工具与运行链：方程、工程与宿主

从方程到可执行模型：脚本、模型与宿主各做什么

- MATLAB脚本固定参数、初值和输入；Simulink用模块及信号线表达方程，由求解器推进状态。
- SLX保存模型连接与求解设置；C/C++代码经编译和平台封装成为动态链接库（DLL），由宿主调用。
- 例如修改初始化脚本中的质量，再运行、保存日志；只换三维外观不会改变受力与运动。



方程决定计算关系，工程保证可复现，运行产物由宿主执行。

4.9.1 工具与运行链：系统建模工具

按需要选择模型库

- 航空航天模块集（Aerospace Blockset）：刚体六自由度、环境、传感器等白盒模块，RflySim PSP模型库多基于它开发。
- RflySim模型库封装电机、力与力矩、地面和平台输出等环节，并留有参数接口，可用实验数据辨识修正。
- Simscape：以白盒物理组件连接为主，适合机械抓取、投放、能源管理等需要细化多体约束的任务。
- 无人机工具箱（UAV Toolbox）以质点模型为基础、偏灰盒，服务路径规划与导航；车辆动力学模块集面向地面车辆。

先确定需要保留的物理过程，再选择能表达它的模型组件。

4.9 建模工具、代码生成与模型部署

4.9.1 工具与运行链：整个平台的数据流

平台数据交互：沿执行器和量测追踪闭环

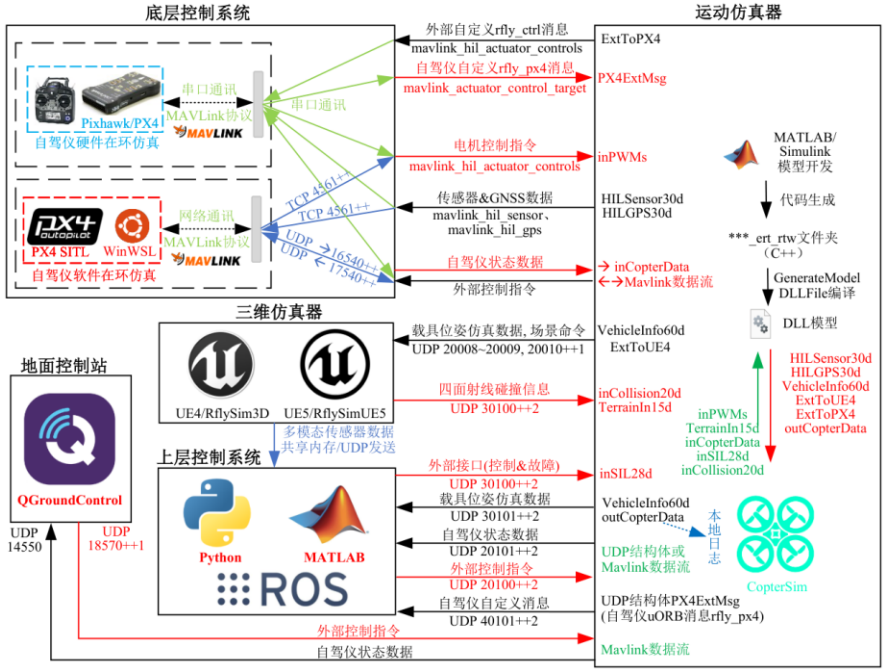


图4.11 运动仿真模型与其他模块数据交互图解

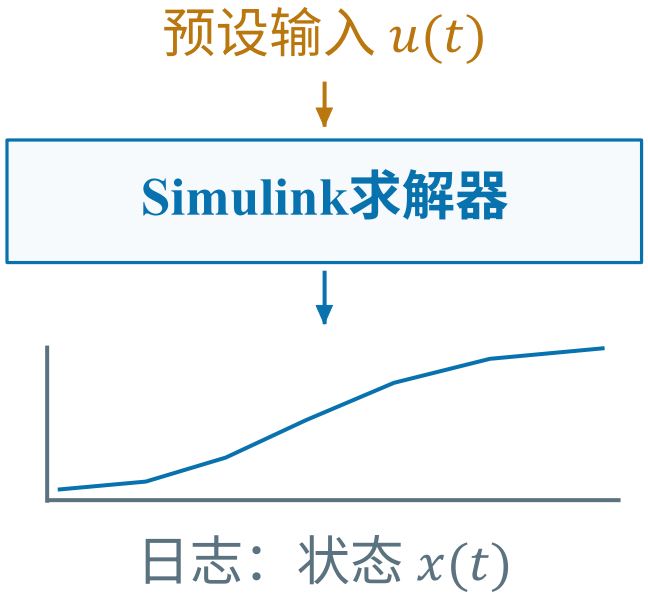
- **闭环：** PX4经inPWMs把执行器量送入CopterSim中的DLL，DLL算出的量测经HILSensor30d、HILGPS30d回到PX4。
- **连接：** SITL（软件在环）的PX4在电脑上经TCP 4561起的端口通信；HITL（硬件在环）的PX4在飞控上经串口通信。
- **外部程序：** Python等经UDP 20100系列端口发指令、20101系列收状态，30101系列取真值。

沿“模型开发—DLL—CopterSim—飞控”追踪运行链路。

4.9.1 工具与运行链：离线求解

离线链路：先检查被控对象

- 离线仿真由Simulink求解器推进时间，输入来自预设信号或数据回放；最初可以不接飞控，单独检查被控对象。
- 记录电机转速、合力与力矩、位置和姿态，能够判断问题发生在执行器、受力合成还是刚体运动环节。
- 停止时间和记录周期由试验设置确定；动画只是观察窗口，关闭显示后仍应能得到可解释的时间序列。

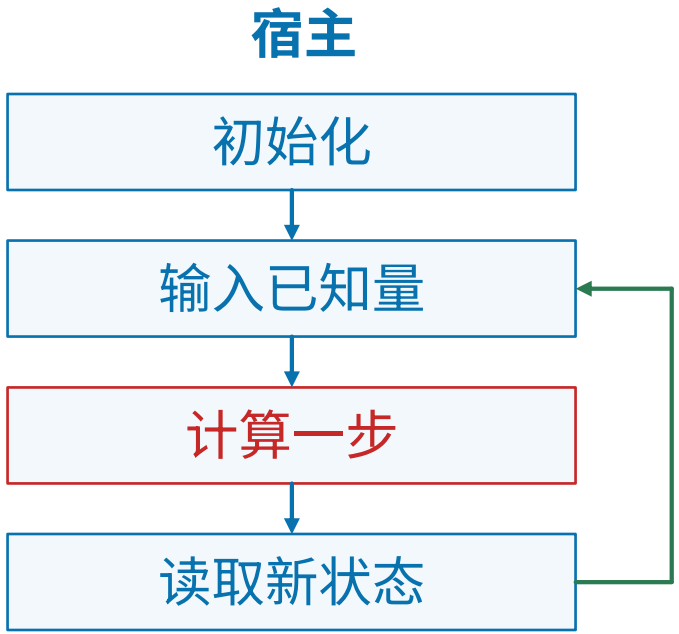


离线试验把已知输入直接送入模型，适合追查物理因果。

4.9.1 工具与运行链：宿主调用

DLL链路：宿主负责调用

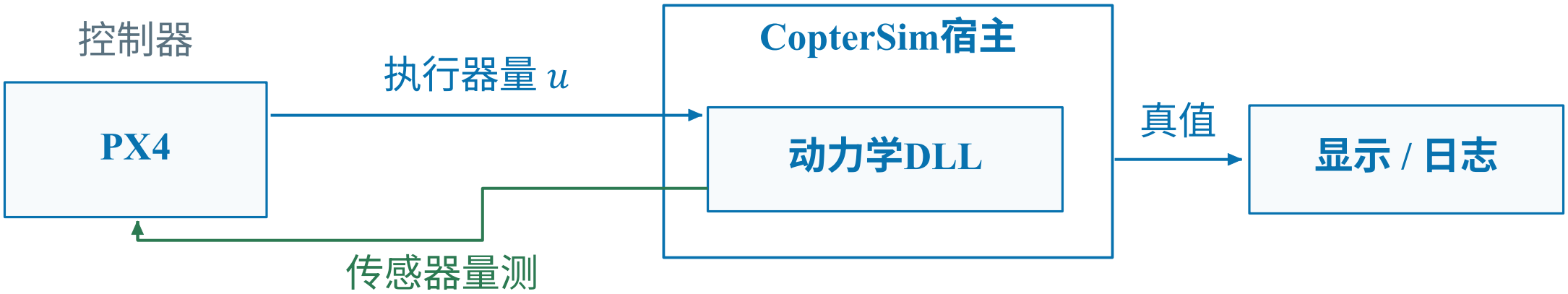
- 生成代码把模型结构转为程序，编译再产生DLL；DLL通常提供初始化、输入、推进一步和读取输出等函数入口。
- 宿主程序决定何时调用这些入口，并提供数据与运行环境；仅把DLL文件复制到目录中，还没有产生新的仿真状态。
- 同一个接口名只有在当前DLL实际导出且内部接线支持时才有效；可调用范围随所编译的模型内容而变化。



DLL保存计算函数；加载、初始化、输入和步进由宿主组织。

4.9.1 工具与运行链：闭环运行

CopterSim链路：动力学与飞控在哪里



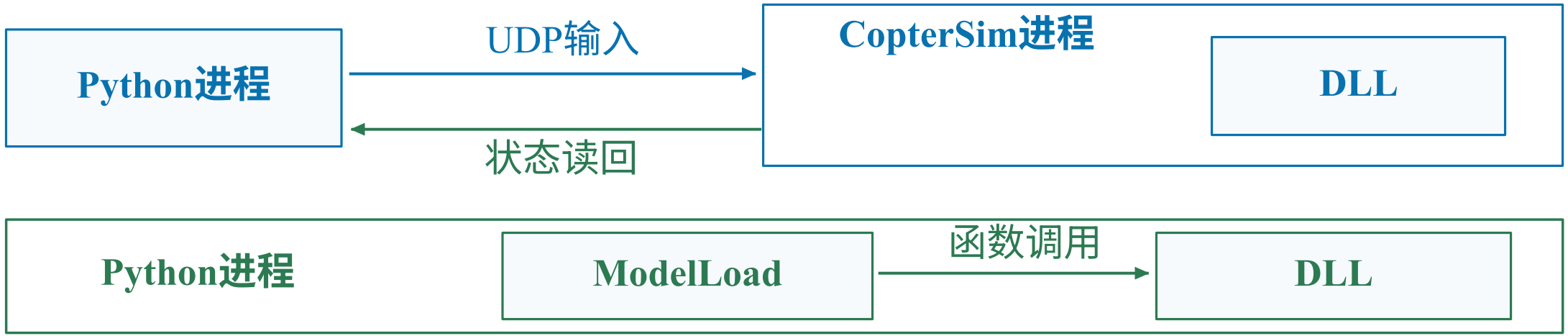
图：量测返回PX4，真值分支用于观察

- **CopterSim加载DLL：**执行器量进入，模型量测返回PX4。
- SITL（软件在环）的PX4在电脑；HITL（硬件在环）在飞控；3D显示真值。

CopterSim推进动力学，PX4根据量测计算执行器输出。

4.9.1 工具与运行链：Python的两类角色

Python两种入口：网络交互与直接加载



图：UDP跨进程通信；直接加载在同一进程调用

- 用户数据报协议（UDP）跨进程通信，CopterSim负责推进。
- 直接加载：ModelLoad在本进程步进DLL；综合DLL含控制器（课件4.5.6）。

网络交互连接现有实例，直接加载则在Python进程中推进DLL。

4.9.1 工具与运行链：追踪生成与传输

局部放大：执行器、量测与代码生成

- 执行器量从飞控进入inPWMs；传感器输出经通信返回PX4。
- 右侧由Simulink生成代码，再编译DLL；CopterSim加载DLL执行计算。
- 图中的平台端口与课件4.10字段定义配合阅读；同名数据在不同传输层各有下标。

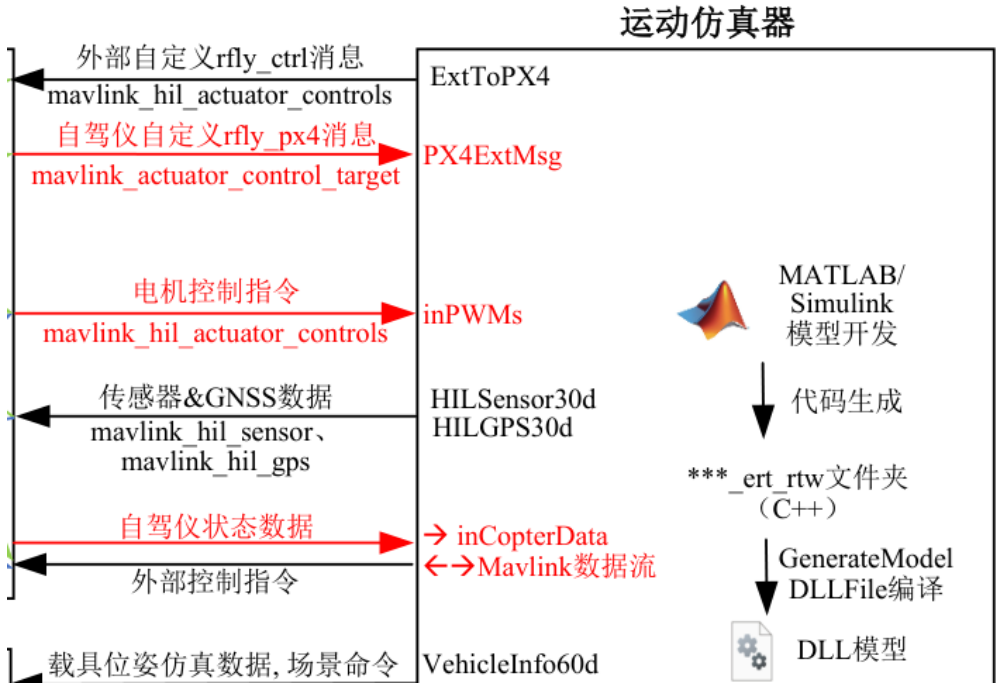


图4.11（右侧）运动模型的输入、输出与生成链

执行器输入与量测回传各有路径，须沿实际接口核对。

4.9.1 工具与运行链：在环连接

局部放大：底层控制系统的两种在环连接

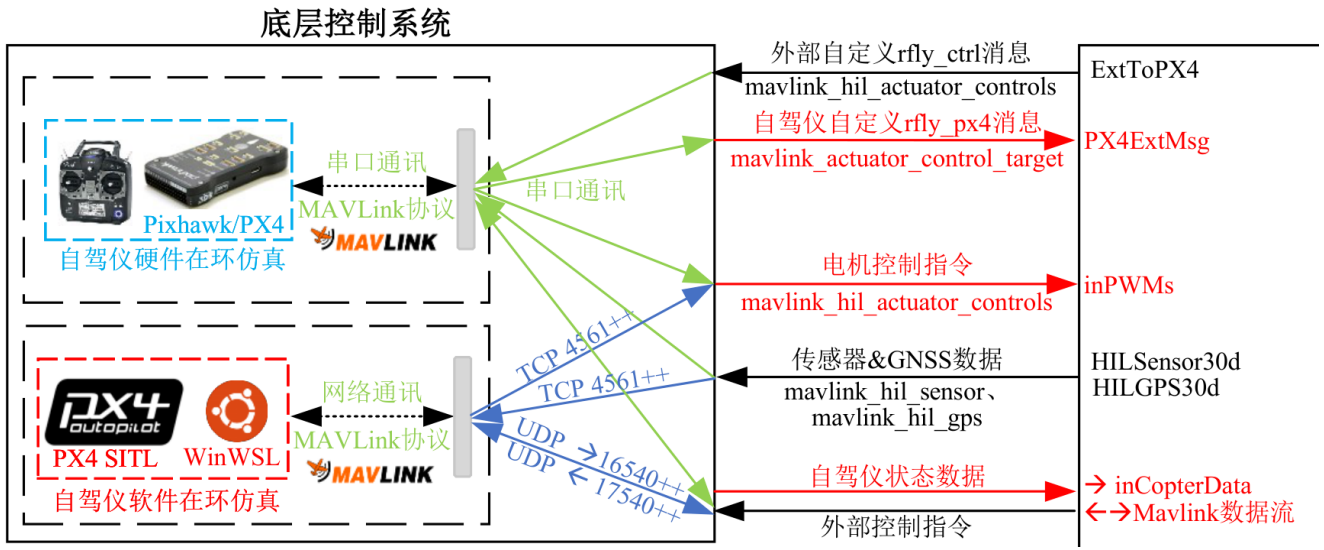


图4.11（左上）底层控制系统与运动仿真器的连接

- **读图：**绿线为HITL串口、蓝线为SITL网络；红线送入运动仿真器，黑线返回飞控。

- **HITL：**Pixhawk飞控经串口，MAVLink协议。
- **SITL：**PX4在电脑（WinWSL），经TCP 4561++。
- **执行器：**mavlink_hil_actuator_controls → inPWMs。
- **量测：**HILSensor30d、HILGPS30d → hil_sensor、hil_gps消息。
- **状态：**→ inCopterData；SITL另经UDP 16540++、17540++。

4.9 建模工具、代码生成与模型部署

4.9.1 工具与运行链：外部接口端口

局部放大：三维、地面站与上层控制的UDP端口

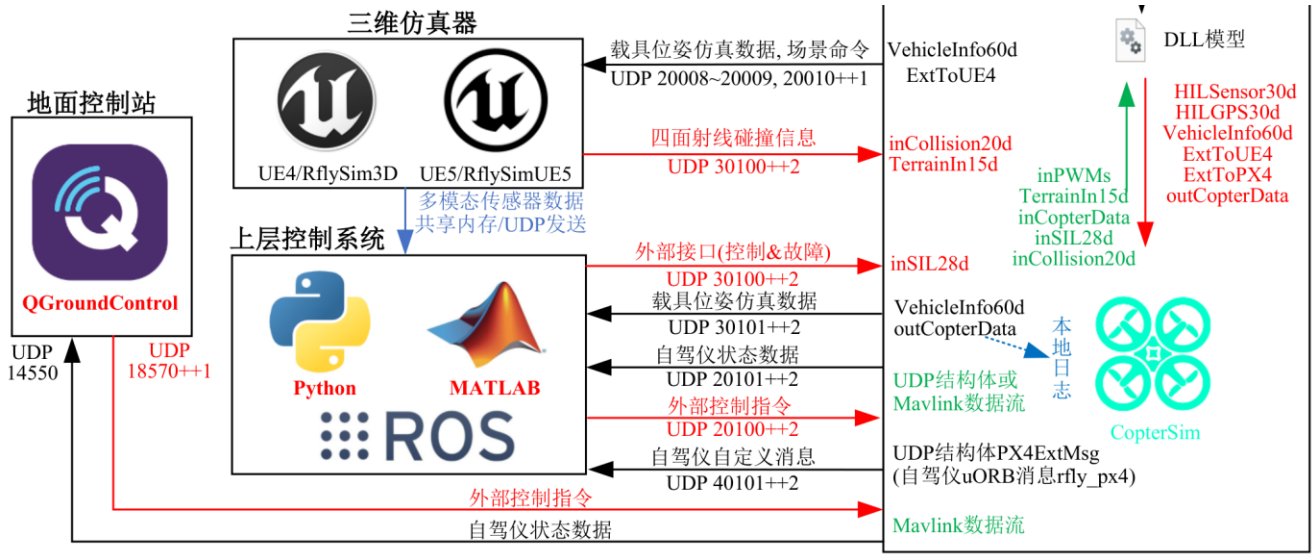


图4.11（下部）三维仿真器、地面站与上层控制系统

- **读图：**红线送入运动仿真器，黑线由它发出；“++2”表示端口按载具编号每台加2。

- **三维仿真器：**收位姿与场景命令（UDP 20008~20009、20010++1），回送碰撞与地形到inCollision20d、TerrainIn15d。
- **上层控制：**inSIL28d走UDP 30100++2，外部控制指令走20100++2。
- **读回：**真值走UDP 30101++2，自驾仪状态走20101++2。
- **地面站：**QGC经UDP 14550收状态，经18570++1发指令。

本章目录

建模工具、代码生成与模型部署

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 7 个教学单元

4.9.1 工具与运行链	教材4.2.1、4.2.2.3
4.9.2 MuJoCo多体建模（拓展）	多关节与接触力
4.9.3 最小模板与可重复工程	教材4.2.2.2、4.2.3
4.9.4 数值时间与调度	步长、多速率、墙钟
4.9.5 从SLX生成C++	教材4.2.3、4.2.6
4.9.6 封装、部署与加载	教材4.2.3、4.2.4
4.9.7 实现一致性与排错	SLX/DLL同条件比较

4.9.2 MuJoCo多体建模（拓展）：多关节与接触系统

多关节与接触系统，由物理引擎按模型自动生成方程

- **单刚体模型：**多旋翼、固定翼和车辆的主体运动可先用一个刚体的方程求解；吊载、系留和拖车还需描述相对状态与约束。
- **多关节系统：**机械臂和课件4.8.4 的吊载、倾转机构由多个刚体经关节相连，彼此的惯性与重力项随关节角耦合，关节越多，手推方程越长。
- **接触系统：**着陆、抓取和行走中的接触力时有时无，还受摩擦限制；多体物理引擎按模型描述自动建立方程并处理接触。

- **建模信息：**给出各刚体的质量与惯量、关节自由度、执行器作用点和接触参数；运行后检查运动状态、约束力与接触力是否符合题设。

选学环境：Python + MuJoCo，独立于RflySim例程入口。

4.9.2 MuJoCo多体建模（拓展）：多体动力学方程

MuJoCo把各种多体模型都写成同一形式的动力学方程

$$M(q)\,\dot{v} + c(q,v) = \tau + J^{\mathrm{T}} f$$

- **惯性矩阵：** $M(q)$ 为关节空间惯性矩阵，由复合刚体算法按姿态算出，始终可逆。
- **偏置力：** $c(q,v)$ 即 qfrc_bias，含科氏力、离心力等速度项与重力，用递归牛顿–欧拉算法计算。
- **外力与约束：** τ 为被动力、执行器力与外力之和；接触、限位等约束力 f 经 J^{T} 进入关节空间，J 为约束雅可比。
- **广义坐标：** q 存于 qpos，v 存于 qvel；free 关节用四元数表示姿态，qpos 7 维、qvel 6 维，只有一个 free 关节时即本章的单刚体方程。

写好MJCF后，惯性、偏置力与约束力都由引擎按当前状态计算。

4.9.2 MuJoCo多体建模（拓展）：MuJoCo的定位与数据结构

MuJoCo按MJCF建立模型，逐步更新仿真状态

- **定位：**MuJoCo 即多关节接触动力学，是通用物理引擎；由 Google DeepMind 维护，以 Apache-2.0 许可开源（截至2026-09）。
- **建模格式：**MJCF 是 MuJoCo 的 XML 模型格式，用嵌套标签依次写出刚体、关节、形状、执行器和传感器，加载时编译成模型。
- **数据结构：**MjModel 存放加载后不变的结构与参数；MjData 存放随时间变化的状态（qpos、qvel、ctrl）与中间结果。
- **仿真循环：**每次 mj_step 先由当前状态算出受力与加速度，再按积分器前进一步；默认用半隐式 Euler（先更新速度再更新位置），步长 0.002 s。

模型只描述一次，每步由引擎按当前状态算出受力并推进。

4.9.2 MuJoCo多体建模（拓展）：MJCF模型元素

MJCF按嵌套写出运动学树，执行器与传感器挂在其上

世界体 worldbody：场景的根，固定不动；默认重力 9.81 m/s^2 沿 $-z$。

刚体 body：带质量与惯量；body 里再套 body，组成运动学树。

关节 joint：子体相对父体的自由度：hinge 转、slide 移、ball 球铰、free 六自由度。

几何体 geom：碰撞与显示的形状；未写 mass 时按密度 1000 kg/m^3 推算质量与惯量。

标记点 site：固定在 body 上的点，不碰撞、无质量，常作施力点和测量点。

顶层元素：写在 worldbody 之外，按名称引用 joint 或 site。

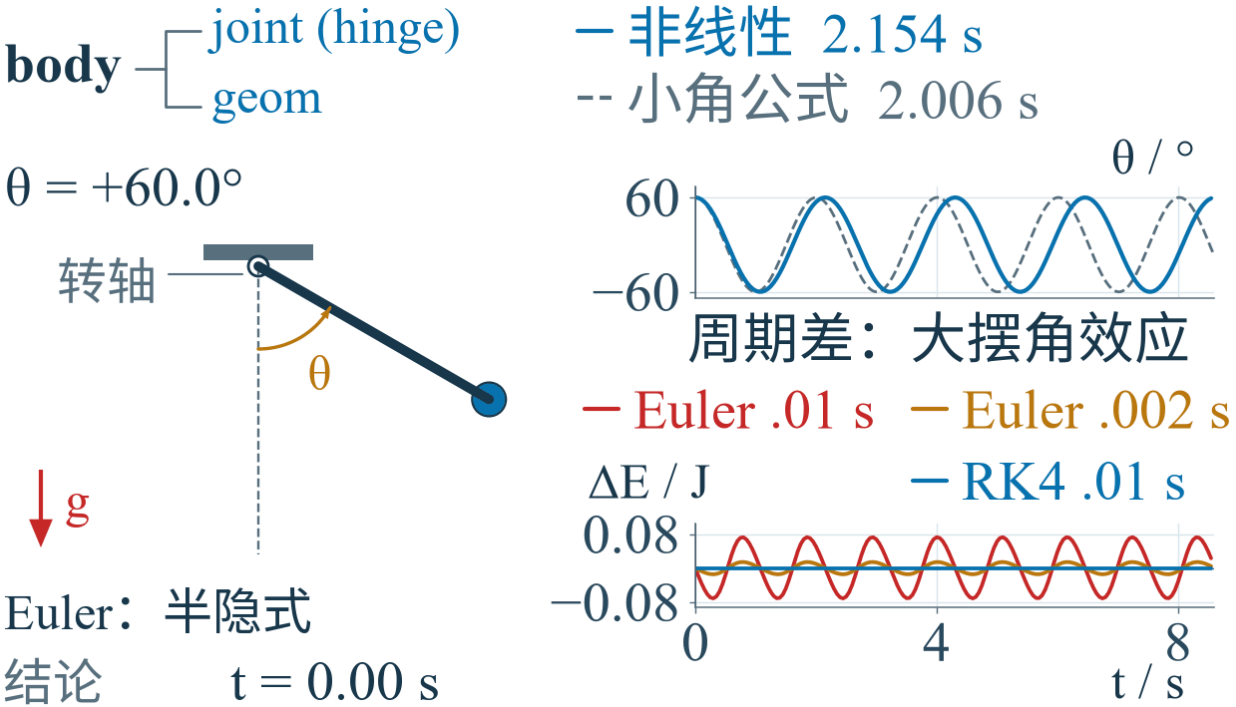
执行器 actuator：如 motor 把控制量 ctrl 乘以传动比 gear，作用到关节或 site。

传感器 sensor：读出位置、速度、力等仿真量，写入 MjData 的 sensordata。

4.9.2 MuJoCo多体建模（拓展）：单摆与积分器

单摆只需body、joint和geom，积分器与步长决定精度

- **模型：** worldbody 下一个 body，hinge 关节给出 1 个转动自由度；小球 1 kg，细杆长 1 m、无质量。
- **周期：** 60° 释放时仿真周期 2.154 s，与精确解一致；大摆角使它比小角度公式长约 7%。
- **积分器：** 只换积分器时，步长 0.01 s 的半隐式 Euler 能量误差 0.073 J，RK4 仅约 5×10^{-8} J。



图：单摆摆角与积分器能量误差（MuJoCo 仿真）

模型结构决定方程，积分器和步长决定数值解离真解有多近。

4.9.2 MuJoCo多体建模（拓展）：Python最小例子

用MjModel、MjData和mj_step三步完成单摆仿真

```
import mujoco
XML = """<mujoco><worldbody><body>
<joint type="hinge" axis="0 -1 0"/>
<geom type="sphere" pos="0 0 -1" size="0.05" mass="1"/>
</body></worldbody></mujoco>"""
model = mujoco.MjModel.from_xml_string(XML)  # 常量
data = mujoco.MjData(model)  # 状态
data.qpos[0] = 1.0472  # 60° 静止释放
for _ in range(500):  # 默认步长 0.002 s，共 1 s
        mujoco.mj_step(model, data)
print(round(data.time, 2), round(data.qpos[0], 4))
```

- **试一试：** option设integrator="RK4"、timestep="0.01"，循环100次，输出−1.0221。

- **加载：** from_xml_string 把 MJCF 编译成 MjModel，MjData 按模型存放状态。

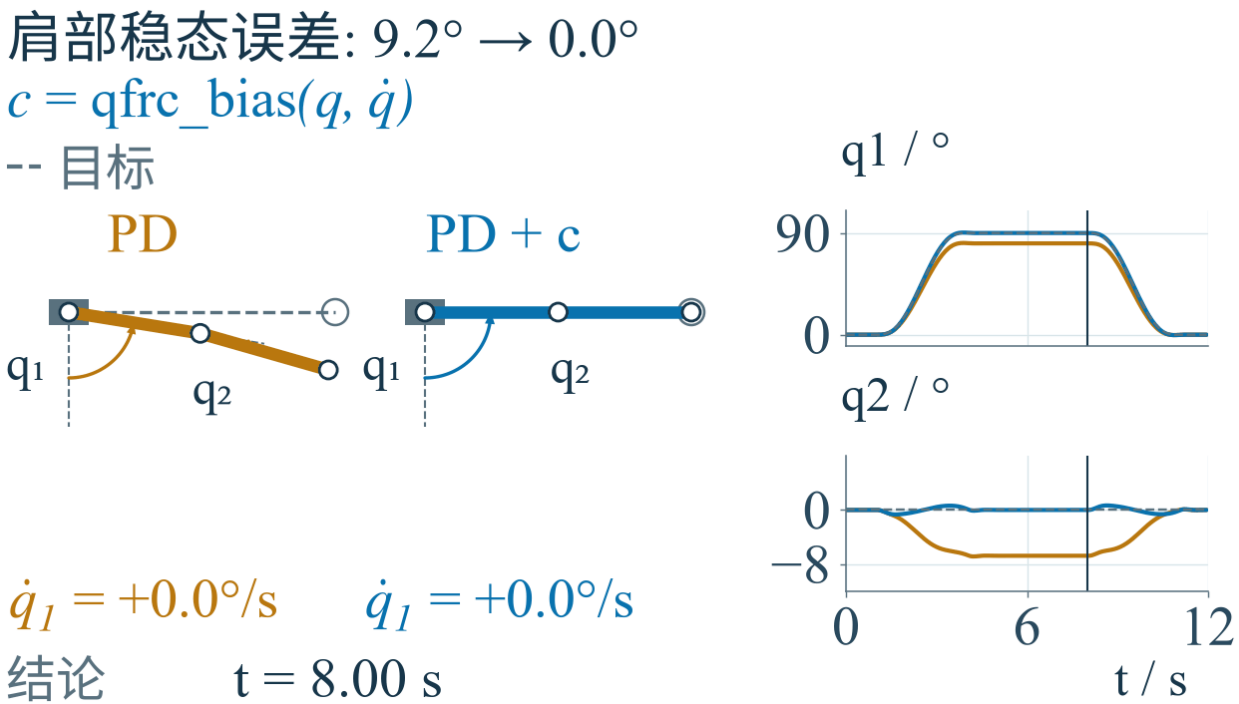
- **初态与推进：** 60° 静止释放；默认半隐式 Euler，步长 0.002 s；mj_step 500次推进1 s。

- **结果：** 输出 1.0 -1.0228，即 1 s 时摆到另一侧约 −58.6°，接近半周期时的 −60°。

4.9.2 MuJoCo多体建模（拓展）：两连杆PD与偏置补偿

加上qfrc_bias补偿后，机械臂不再靠误差顶住重力

- **对比：**左右两条两连杆臂相同，左臂只用比例–微分（PD）控制，右臂另加 qfrc_bias 补偿。
- **仅PD：**静止时 PD 只剩 $K_p e$ 一项来平衡重力矩 τ_g，故 $e \approx \tau_g / K_p$；肩部 $9.8 \div 60 \approx 0.16$ rad$\approx 9.4°$。
- **加补偿：**每步按当前状态加上 qfrc_bias，电机直接提供重力矩 (9.8, 2.5) N · m，稳态误差为 0。



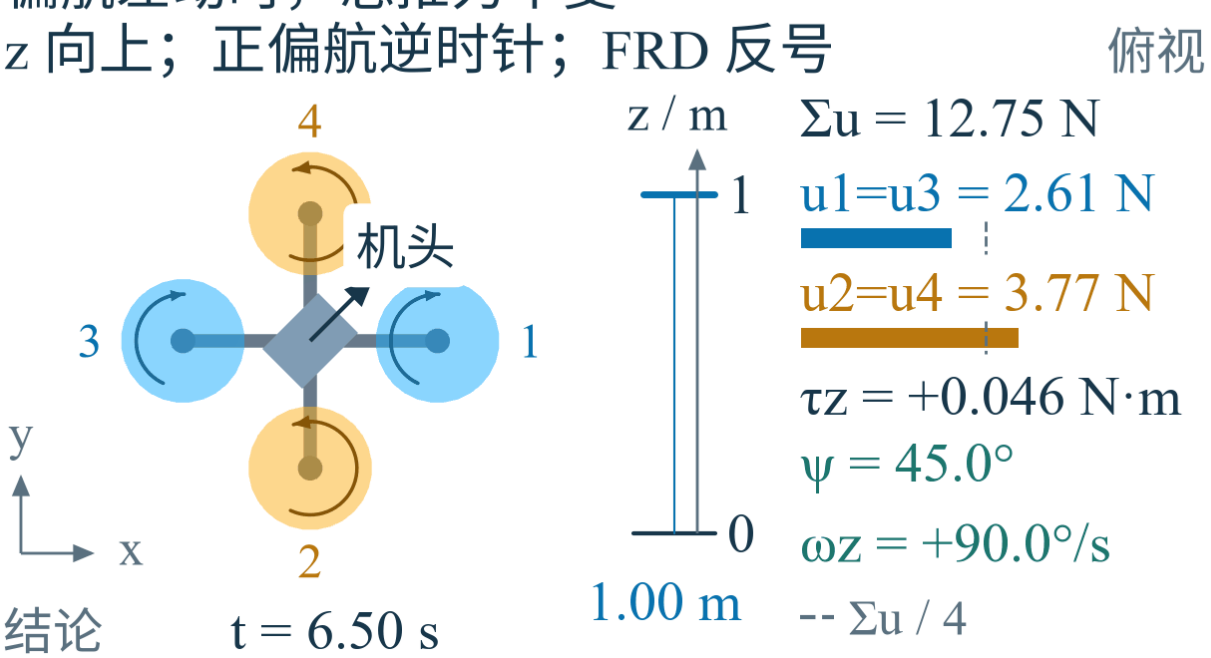
图：两连杆臂仅PD与加偏置补偿（MuJoCo 仿真）

理想偏置补偿可承担抗重力矩，使PD不再靠稳态误差维持平衡。

4.9.2 MuJoCo多体建模（拓展）：自由关节四旋翼

site与gear让一个控制量同时产生推力和反扭矩

- **机体：**一个 free 关节刚体，qpos 7 维、qvel 6 维，即六自由度模型；z 轴向上，与 FRD 相反。
- **旋翼：**桨心 site 上挂 motor，取 gear=(0, 0, 1, 0, 0, $\pm k$)，控制量 u_i 产生推力 u_i 与反扭矩 $\pm k u_i$。
- **推力：**悬停时每桨推力为 $mg/4$ 即 3.19 N；对角两组 $\pm\Delta$ 时滚转、俯仰力矩抵消，只剩 $\tau_z = 4k\Delta$。



图：自由关节四旋翼推力与反扭矩（MuJoCo 仿真）

引擎按桨位与方向合成力矩，控制器仍须分配各桨输入。

4.9.2 MuJoCo多体建模（拓展）：学习资源

从官方文档和模型库继续学习MuJoCo建模

- **入门概览：** MuJoCo 的定位、MJCF 模型与 MjModel、MjData 的分工
  https://mujoco.readthedocs.io/en/stable/overview.html
- **元素手册：** body、joint、geom、site、actuator 等元素的全部属性与默认值
  https://mujoco.readthedocs.io/en/stable/XMLreference.html
- **计算原理：** 动力学方程、约束与接触求解，以及 Euler、RK4 等积分器
  https://mujoco.readthedocs.io/en/stable/computation/index.html
- **编程接口：** Python 绑定的安装（pip install mujoco）与 mj_step 等函数
  https://mujoco.readthedocs.io/en/stable/python.html
- **模型库：** MuJoCo Menagerie 中的机械臂、足式机器人与 Skydio X2 四旋翼等模型
  https://github.com/google-deepmind/mujoco_menagerie

先读概览建立概念，查手册写模型，再从模型库的现成模型改起。

本章目录

建模工具、代码生成与模型部署

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 7 个教学单元

4.9.1 工具与运行链	教材4.2.1、4.2.2.3
4.9.2 MuJoCo多体建模（拓展）	多关节与接触力
4.9.3 最小模板与可重复工程	教材4.2.2.2、4.2.3
4.9.4 数值时间与调度	步长、多速率、墙钟
4.9.5 从SLX生成C++	教材4.2.3、4.2.6
4.9.6 封装、部署与加载	教材4.2.3、4.2.4
4.9.7 实现一致性与排错	SLX/DLL同条件比较

4.9.3 最小模板与可重复工程：通用载具控制模型框架

通用载具框架：沿推力追踪运动状态

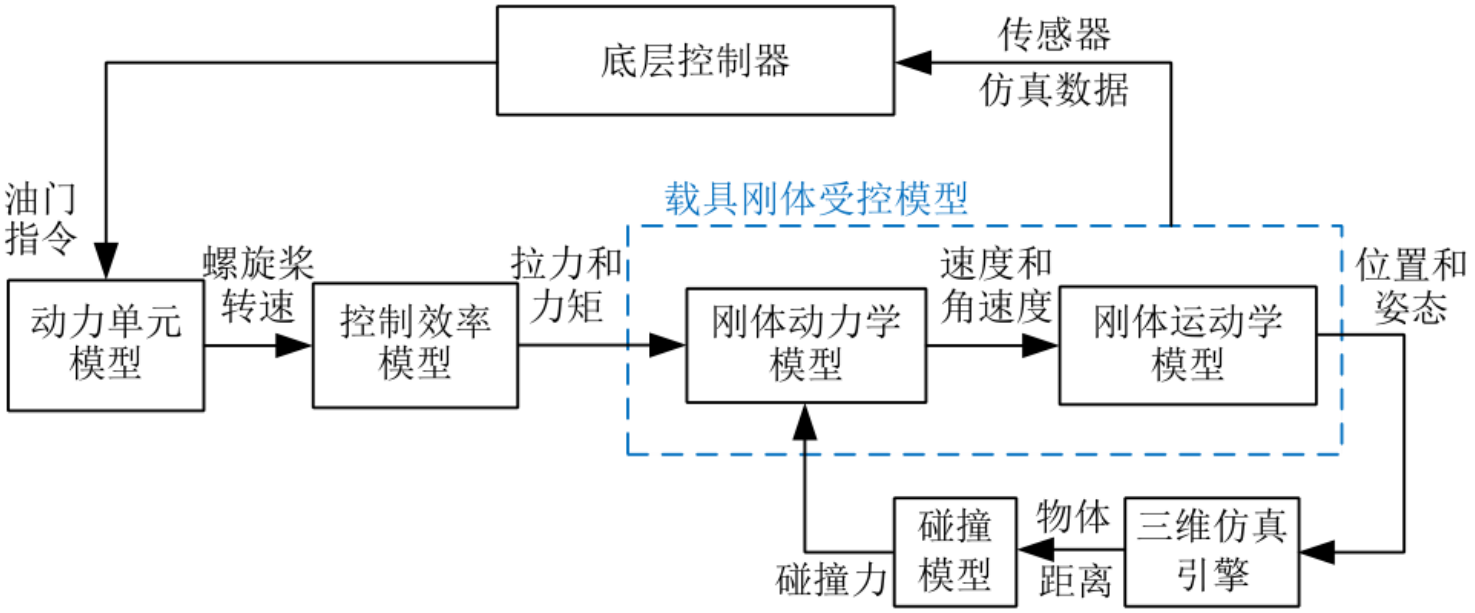


图4.12 通用载具控制模型框架

- **主线：** 油门→转速→拉力和力矩→速度和角速度→位置和姿态。
- **碰撞支路：** 三维引擎给出物体距离，碰撞力接回刚体动力学。
- **虚线框：** 刚体受控模型。
- **回路：** 传感器仿真数据回到底层控制器。

模块边界要对应可解释的输入、输出和状态。

4.9.3 最小模板与可重复工程：工程文件

M0工程目录：模型、初始化、打包、启动与产物

- 目录：[安装目录]/RflySimAPIs/4.RflySimModel/1.BasicExps/e0_MinModelTemp/，以Exp1_MinModelTemp.slx为部署基线M0，先读Readme.pdf。

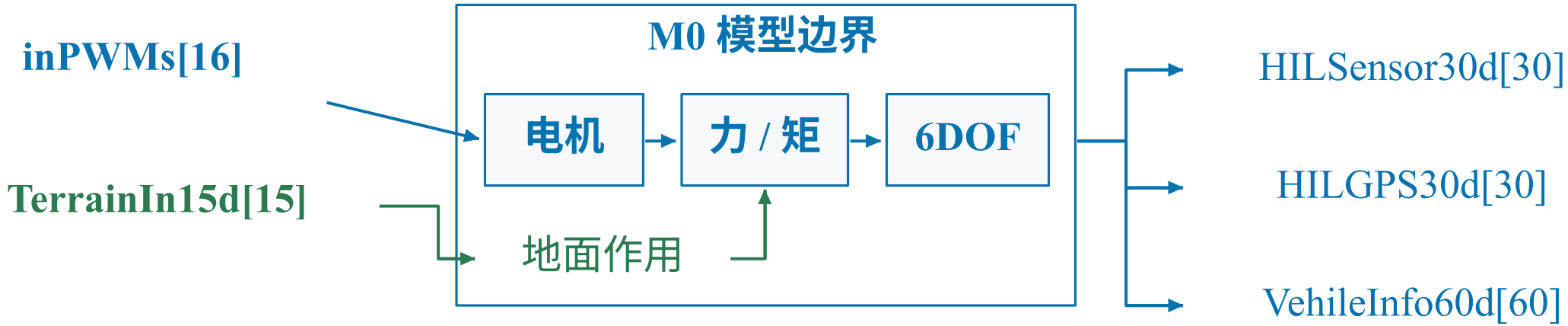
```
e0_MinModelTemp/
  Readme.pdf
  Exp1_MinModelTemp.slx
  Exp1_MinModelTemp_init.m
  GenerateModelDLLFile.p
  Exp1_MinModelTemp_SITL.bat / _HITL.bat
  Exp1_MinModelTemp.dll
  MulticopterModel.zip
```

- **模型与初始化：** SLX定义方程与根端口，_init.m保存参数和初态。
- **打包与启动：** .p文件打包DLL；两个批处理分别启动SITL与HITL。
- **发布产物：** DLL可直接加载；zip内是已打包的生成代码。

模型定义、初始化与运行产物对应同一版本，部署才可复现。

4.9.3 最小模板与可重复工程：根端口

M0的两输入与三输出



图：执行器与地形输入，形成状态和观测输出；VehileInfo60d为源码拼写

- 16维执行器输入、15维地形输入；地形第1项是地面高度。
- 6DOF总线含向量与矩阵；先选信号，再读三类输出。

根端口是模型与宿主约定的数据边界。

4.9.3 最小模板与可重复工程：沿实际接线定位信号

从一条信号追踪到状态

- 从inPWMs进入Motor Model，观察MotorRads角速度，再进入Force and Moment Model计算合力与力矩。
- 6DOF1推进运动状态，并把位置、速度和姿态等送往SensorOutput与3DOutput1；地面作用经接触模型反馈。
- 6DOF输出是含向量与矩阵的总线，不是一个固定13维数组；从总线选择器展开信号，才能给日志命名并标单位。



图：M0执行器输入的计算路径

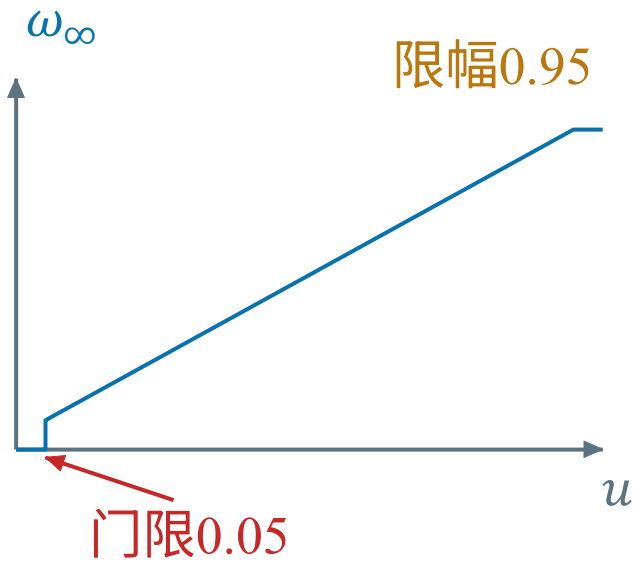
先沿实际接线寻找因果，再使用输出端口记录结果。

4.9.3 最小模板与可重复工程：输入尺度

执行器输入不是微秒脉宽

- M0前四通道用归一化电机量，舵机常用−1到1；误写1500会被限幅成0.95，接近满油门。
- 输入限幅0至0.95；达到0.05时目标转速线性给定，否则为零；实际转速始终按一阶响应。
- 16维端口预留扩展；当前M0电机数为4，几何位置和旋向由布局决定。

$$u_s = \min(0.95, \max(0, u))$$

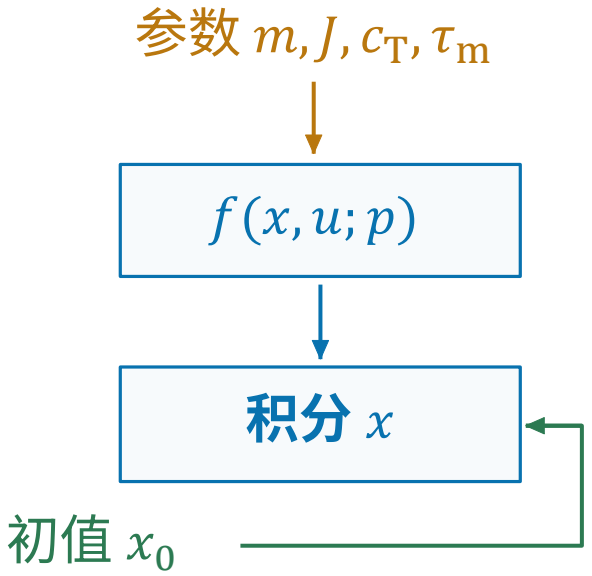


示意：门限后线性映射

端口名含PWM，实际数据仍须按归一化尺度使用。

4.9.3 最小模板与可重复工程：参数与初态

参数决定模型，初值决定起点



图：模型属性与试验起点

- M0质量为1.515 kg，惯量矩阵对角为0.0211、0.0219、0.0366 kg·m²；它们决定相同力与力矩产生的加速度。
- 位置、速度、姿态、角速度和电机转速属于初始状态；同一输入从静止或已有速度开始，会得到不同的位置曲线。
- M0脚本初始位置为零，速度和角速度也为零；若测试自由运动，要在教学副本中声明新的高度及接触处理方式。

改变物理参数和改变试验起点，会产生不同类型的响应变化。

4.9.3 最小模板与可重复工程：两种模板

D0提供方程对照，M0提供部署起点

- **位置：**两份模板都在[安装目录]/RflySimAPIs/4.RflySimModel/下，用途不同。

M0：部署起点

- **目录：**1.BasicExps/e0_MinModelTemp
- **文件：**Exp1_MinModelTemp.slx＋_init.m
- **生成：**嵌入式C++，保留平台根端口

D0：方程对照

- **目录：**2.AdvExps/e12_ModelTempId/3.ModelingTemp/code
- **文件：**dynamics.slx＋Init.m，先运行
- **生成：**默认C、参数内联

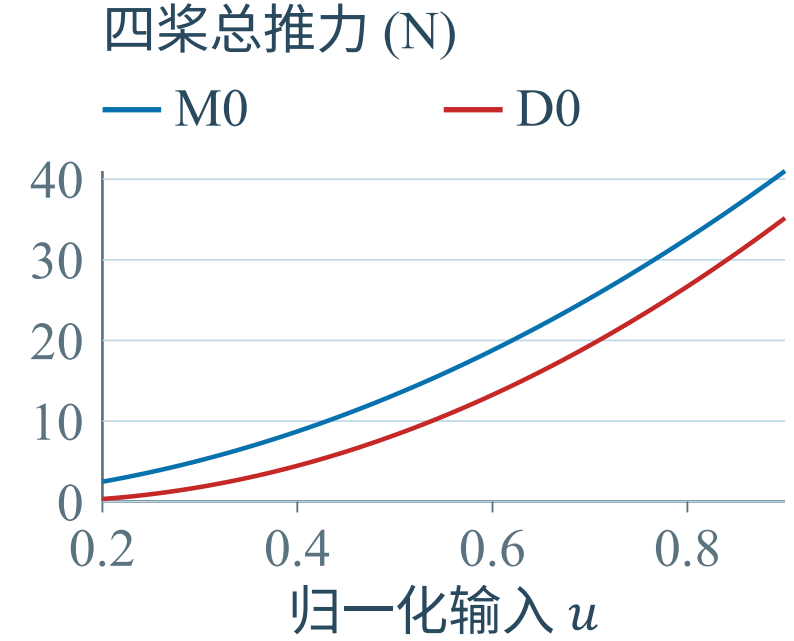
- D0不是M0的同参数副本：质量、电机映射与初始高度都不同；两者的静态推力对比按映射公式计算。

先识别模板来源与用途，再解释默认参数带来的差异。

4.9.3 最小模板与可重复工程：统一比较条件

M0与D0为什么不能直接比曲线

- M0质量1.515 kg，映射为 $\omega = 842.1u + 22.83$ ；D0质量1.4 kg，映射为$\omega = 1148u - 141.4$。
- 转速ω与截距单位均为rad/s，输入u无量纲；两者拉力系数分别为1.681×10^{-5}与1.105×10^{-5}，单位N·s²/rad²。
- 初始z_D分别为0与负100 m；同一输入给出的总推力也不同，比较前须统一参数、初态与模型边界。



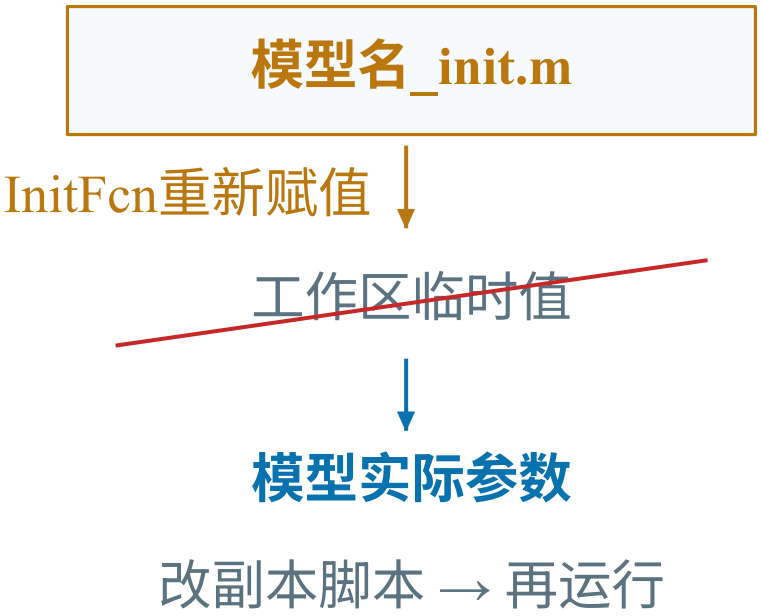
图：同一输入的静态推力差异

模板参数与初态不同，轨迹差异不能直接归因于实现。

4.9.3 最小模板与可重复工程：初始化回调

初始化脚本何时执行

- M0初始化回调（InitFcn）会执行同名_init.m，重新赋值参数和初态，覆盖工作区临时改值。
- 修改教学副本的_init.m并核对回调；测试封装也可取消副本回调，再注入固定参数。
- D0先在code目录执行Init.m；它含clear，先保存数据，再核对实际质量、初态和时间常数。



模型名称、当前目录和变量工作区共同影响初始化。

4.9.3 最小模板与可重复工程：输出边界

真值、量测和估计分别是什么

真值： $a_D = 0$

↓

水平静止： $f_{b,z} = -g$

采样、噪声与编码

↓

PX4融合后得到估计

三种数据不可直接互换

- 真值是模型计算的运动状态，用于核对力与运动关系；VehileInfo60d提供真值及显示数据。
- 量测由状态经过传感器模型产生，含噪声、采样与编码；加速度计比力不同于地面系加速度。
- 估计是PX4融合量测后的状态，会受滤波和初始化影响；比较前先说明读取的是哪一层。

同一架飞机的不同数据出口，回答的是不同问题。

4.9.3 最小模板与可重复工程：冻结试验条件

按Run之前固定一次试验

- 复制工程并固定参数、初态、NED/FRD坐标与环境开关；有限停止时间、输入序列和日志共同构成试验。
- M0当前为ode4、1 ms、10 s；D0默认无限时长，需先运行Init.m、替换零输入并设置有限停止时间。
- 离线检查：4.RflySimModel/1.BasicExps/e6_ModelingLearningLab/05_M0Checks/下的m0_offline_checks（先读Readme.pdf），全流程见课件4.11.2。

参数：质量1.515 kg

初态：位置、速度为0

求解：ode4、1 ms、10 s

日志：输入、转速、状态

图：M0试验卡样例

输入、初值、时间和记录量确定以后，运行才有可判断的结果。

本章目录

建模工具、代码生成与模型部署

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 7 个教学单元

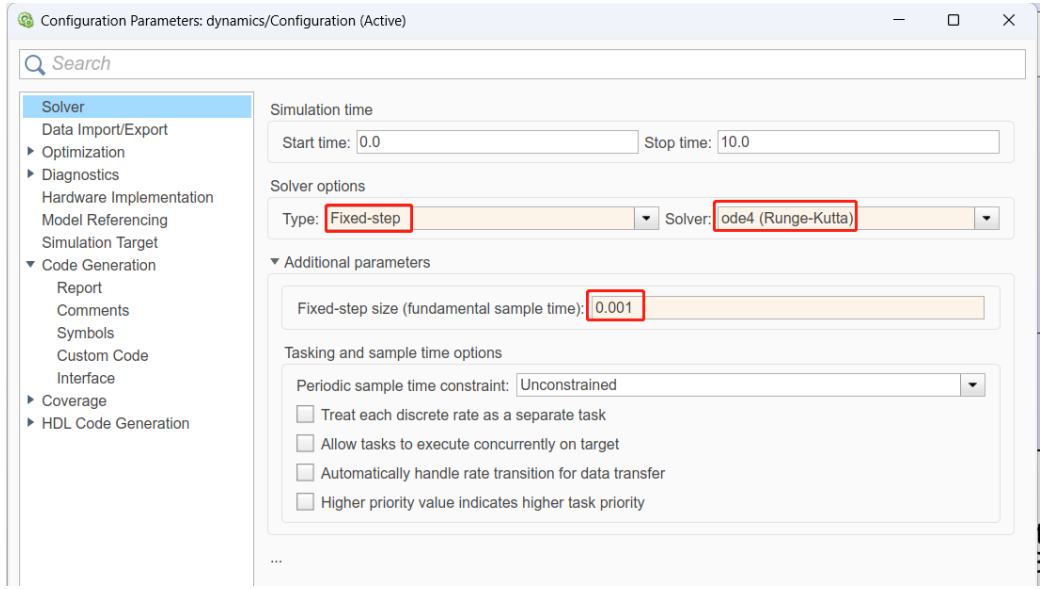
4.9.1 工具与运行链	教材4.2.1、4.2.2.3
4.9.2 MuJoCo多体建模（拓展）	多关节与接触力
4.9.3 最小模板与可重复工程	教材4.2.2.2、4.2.3
4.9.4 数值时间与调度	步长、多速率、墙钟
4.9.5 从SLX生成C++	教材4.2.3、4.2.6
4.9.6 封装、部署与加载	教材4.2.3、4.2.4
4.9.7 实现一致性与排错	SLX/DLL同条件比较

4.9.4 数值时间与调度：求解设置

步长决定多久推进一次状态

- 步长h是相邻积分时刻间隔；固定步长便于代码周期调用，过大可能漏掉快速变化。
- 在Solver页核对Fixed-step、ode4与步长0.001 s；M0用四阶龙格—库塔法。
- 停止时间也属条件；步长是否足够准确，要结合电机和接触工况检查。

$$\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}, t), \qquad t_{k+1} = t_k + h$$

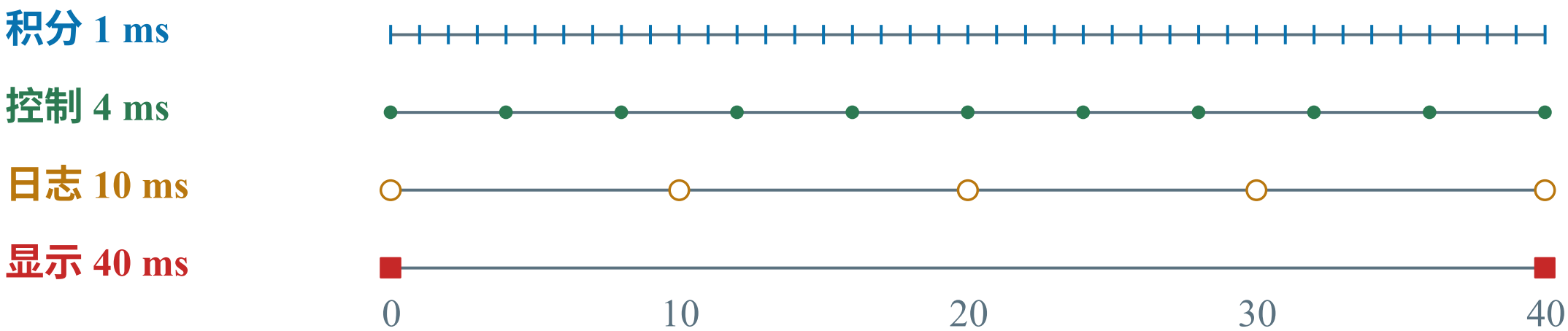


图：求解器配置（资料截图，示例模型dynamics；M0设置相同）

连续方程通过有限时间步得到数值轨迹。

4.9.4 数值时间与调度：多速率任务

一套仿真可以有多种周期



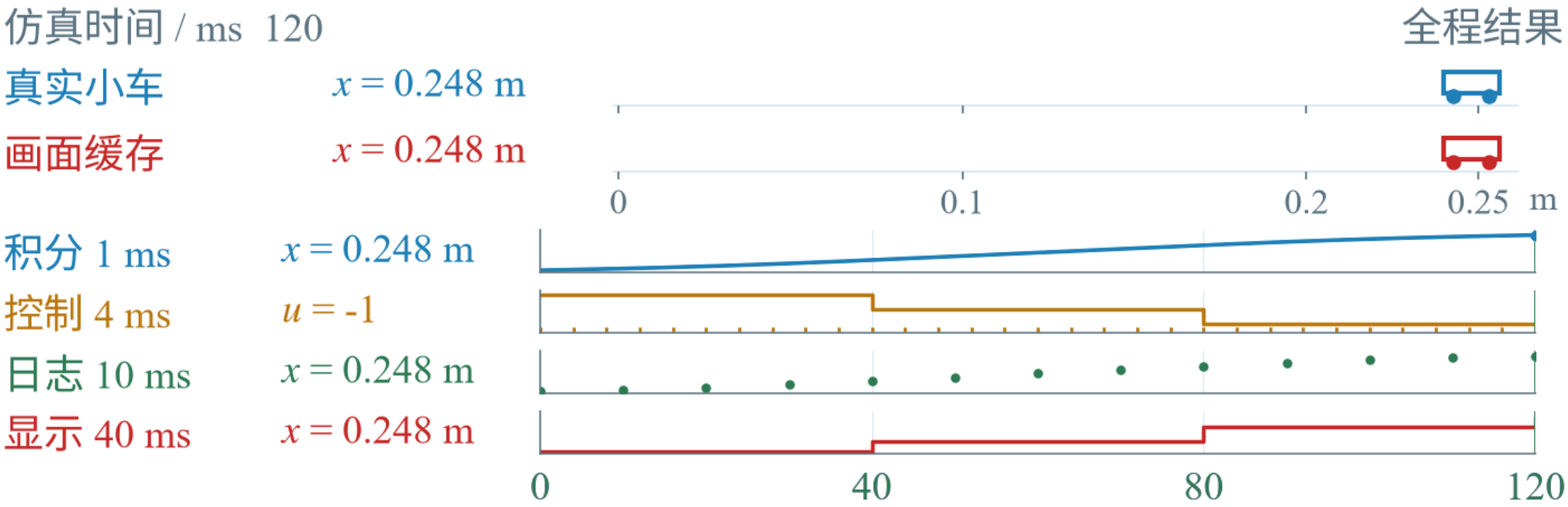
图：多速率节拍示意（横轴单位ms）

- 控制命令在两次更新间保持；日志采样与显示刷新各自定时。
- 1次40 ms显示间隔含40个积分步；读取缓存不会推进物理。

积分更新、输入保持、日志采样和画面刷新承担不同职责。

4.9.4 数值时间与调度：同一状态的不同节拍

真实运动持续推进，画面缓存按显示节拍跳变



图：同一运动的四种节拍（位置单位m；时间单位ms）

积分更新、输入保持、日志采样和画面刷新承担不同职责。

4.9.4 数值时间与调度：仿真与墙钟时间

仿真时间与墙钟时间分别记录

- 仿真时间是状态方程所处的时间，墙钟时间是现实经过的时间；离线计算可以快于或慢于实际过程。
- 实时倍率可用仿真时间增量除以墙钟时间增量表示；输出轨迹比较使用仿真时间，性能比较再使用墙钟时间。
- 直接DLL按整数个基本步推进；正请求时长除以步长后取整数部分，实际推进Nh，余下不足一步的时间不会执行。

$$\rho = \frac{\Delta t_{\mathrm{sim}}}{\Delta t_{\mathrm{wall}}}$$

$$N = \mathrm{int}(\Delta t / h)$$

模型演化多少秒，与计算机等待多少秒，是两种时间。

4.9.4 数值时间与调度：快速动态与接触

接触和快速动态为何更挑步长

- 地面接触常在短时间内产生较大的弹簧与阻尼作用；步长过大可能错过接触过程，形成穿透、反弹或数值振荡。
- 电机时间常数变小，同一步长覆盖的响应比例变大；改参数后做步长减半：课件4.11.2中M0短时基准1 ms与0.5 ms的位置差约10^{-8} m。
- 长时间姿态积分还要检查四元数范数或旋转矩阵正交性；位置曲线平滑并不能证明所有内部状态约束正确。

$$\|\mathbf{q}\| \approx 1, \qquad R^T R \approx I$$

变化越快，越需要检验有限步长能否解析关键过程。

本章目录

建模工具、代码生成与模型部署

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

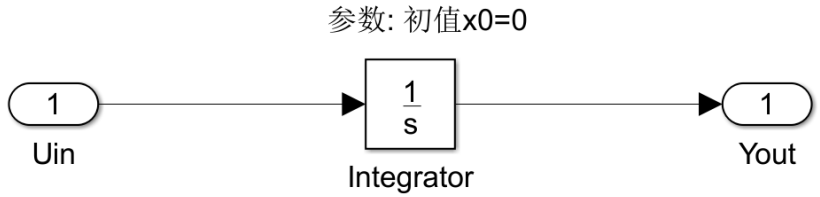
本节 7 个教学单元

4.9.1 工具与运行链	教材4.2.1、4.2.2.3
4.9.2 MuJoCo多体建模（拓展）	多关节与接触力
4.9.3 最小模板与可重复工程	教材4.2.2.2、4.2.3
4.9.4 数值时间与调度	步长、多速率、墙钟
4.9.5 从SLX生成C++	教材4.2.3、4.2.6
4.9.6 封装、部署与加载	教材4.2.3、4.2.4
4.9.7 实现一致性与排错	SLX/DLL同条件比较

4.9.5 从SLX生成C++：最小可追踪样例

先用一个积分器观察自动代码生成

- **入口：**4.RflySimModel/0.ApiExps/2.UserDefinedC++/2.GenC++/demo.slx，先读Readme.pdf。
- **对应关系：**Uin输入经Integrator积分后成为Yout；初始状态由X0给出，模型只含一条动态关系。
- **预测与判据：**教学副本输入常数1、X0=0、运行1 s，Yout应线性增长到1，1 s处与1的偏差在数值误差量级。



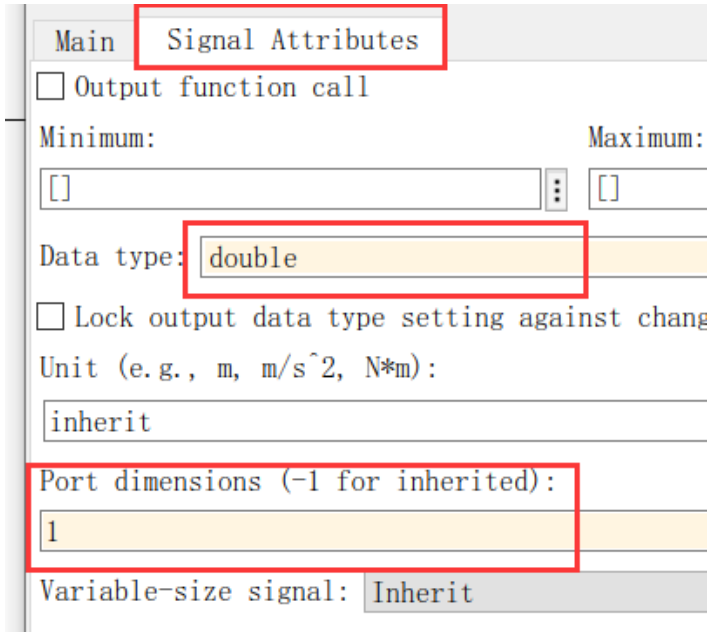
图：GenC++积分器例程

用一个已知解的积分器，把输入、状态、输出与生成程序逐项对应。

4.9.5 从SLX生成C++：端口与状态配置

生成接口以前先固定类型、维数与初值

- **端口：** Uin设为double、维数1，Yout同样设置；端口名、类型、维数、单位与调用时序合称接口约定。
- **初态：** 积分器初值填写X0，模型InitFcn中设置X0=0；重新运行会执行回调，临时改工作区值可能被覆盖。
- **核对：** 先保存教学副本，再给有限停止时间；端口名、double类型、标量维数和初始值一起写入实验记录。



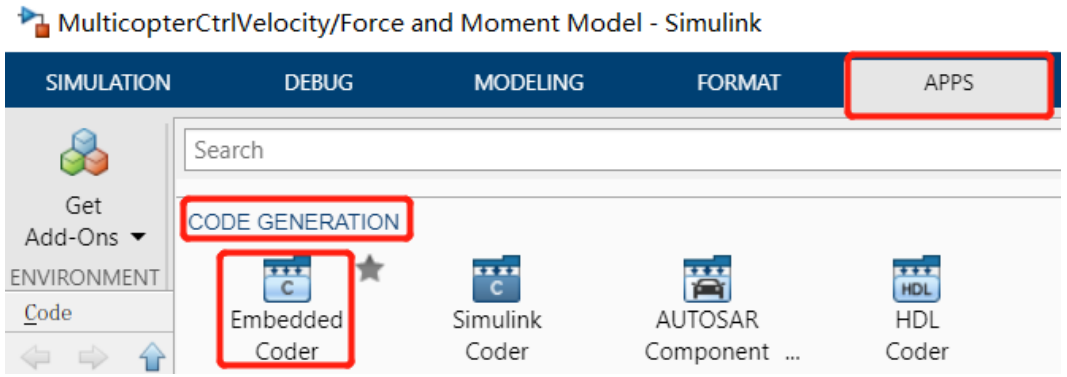
图：标量端口配置（资料局部）

类型、维数和初始化决定调用者怎样正确提供输入并开始一次计算。

4.9.5 从SLX生成C++：生成条件与入口

确认生成条件，再从Embedded Coder启动Build

- **模型依赖：**先更新模型图，修复缺参数和引用库；M0用ert.tlc（Embedded Coder的嵌入式实时目标）生成C++。
- **编译器：**在MATLAB用mex -setup C++确认；工具环境沿用第二章配置。
- **对象切换：**demo用来看清U、Y、X0与积分状态；以下生成设置以M0教学副本为对象，截图取自资料示例。

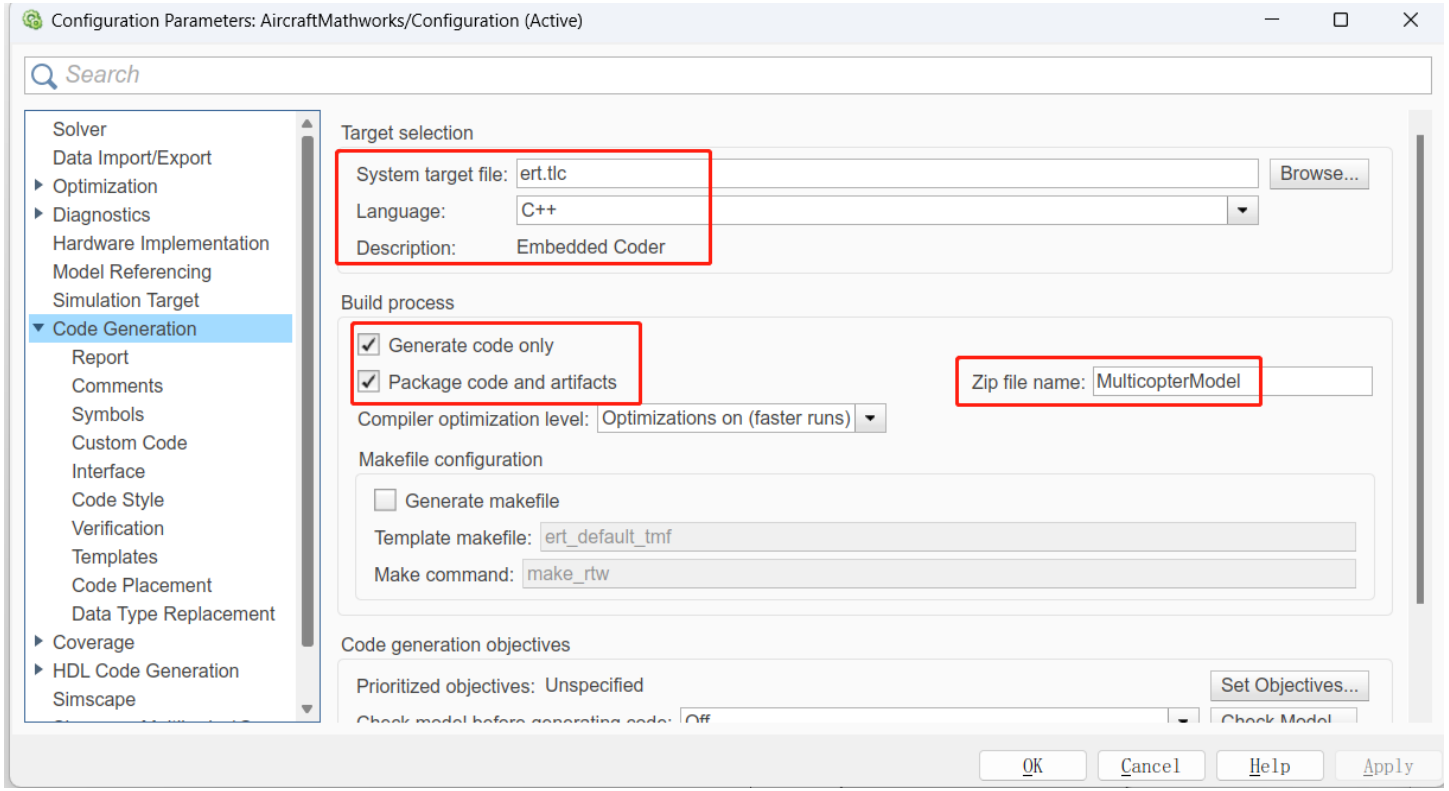


图：Embedded Coder入口（例程资料截图）

- **Build：**在APPS → CODE GENERATION → Embedded Coder中执行，对当前M0教学副本生成程序文件。
- **Run：**运行当前模型得到轨迹；菜单位置随版本变化。

4.9.5 从SLX生成C++：目标与语言

目标、语言与打包设置一起核对

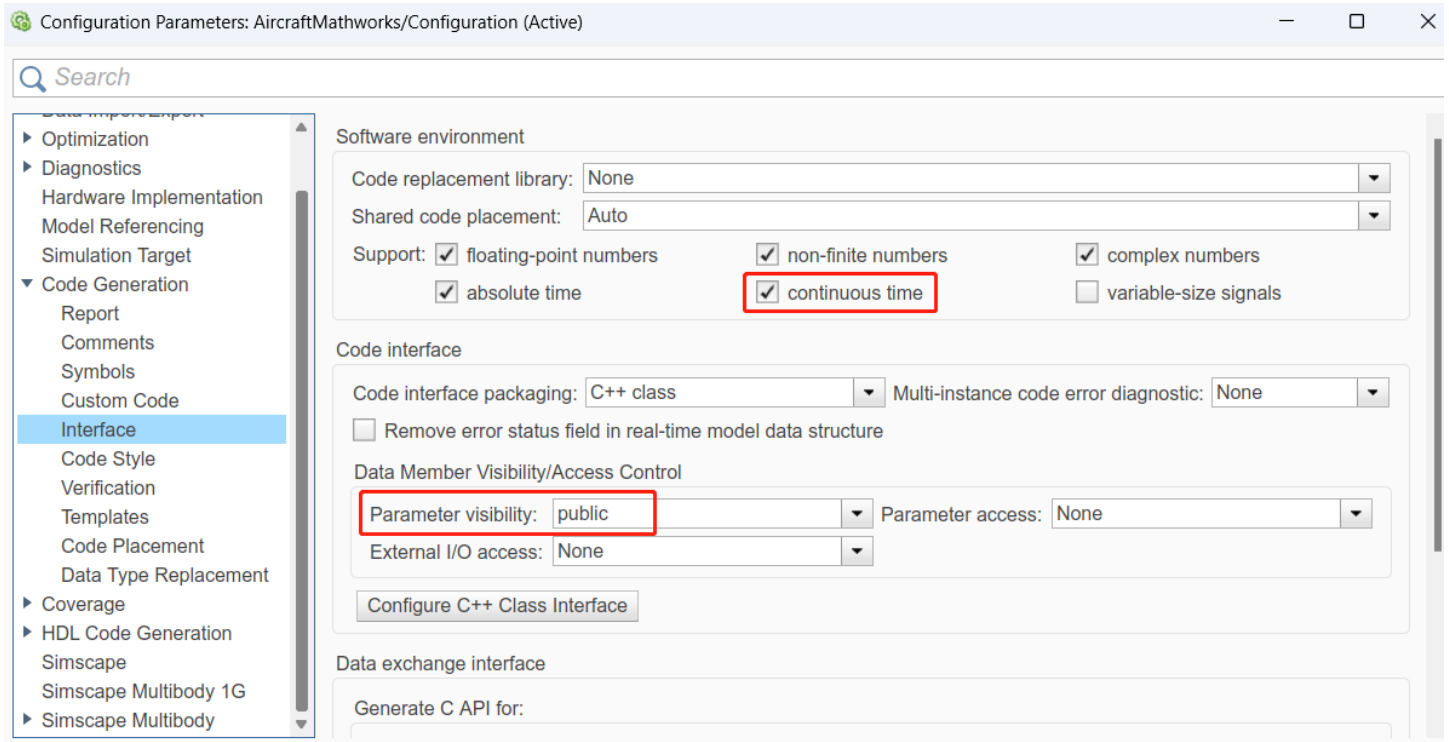


图：目标语言与代码打包设置（API资料截图）

- **目标与语言：**资料图使用ert.tlc与C++；M0当前保存值与此一致。
- **生成与打包：**勾选“仅生成代码”与“打包代码及产物”；打包名为MulticopterModel。
- **入口匹配：**后续流程使用此名；只改文件名或语言可能使入口失配。

4.9.5 从SLX生成C++：连续状态和参数可见性

连续状态支持与可见参数



图：连续状态与参数可见性（API资料截图）

- **continuous time**：含连续积分状态时需支持连续时间；否则模型与生成目标配置冲突。
- **public参数**：程序可读写生成类参数；要从Python改质量，需要DLL导出相应的写入函数。

4.9.5 从SLX生成C++：生成类的可见边界

代码：从生成类识别U、Y、P

- **文件：**解压e0_MinModelTemp/MulticopterModel.zip，进入Exp1_MinModelTemp_ert_rtw：头文件声明接口，CPP实现，ert_main.cpp演示调用。

```
class MulticopterModelClass {
public:
  ExtU_Exp1_MinModelTemp_T Exp1_MinModelTemp_U;
  ExtY_Exp1_MinModelTemp_T Exp1_MinModelTemp_Y;
  static P_Exp1_MinModelTemp_T Exp1_MinModelTemp_P;
  void initialize();
  void step();
  static void terminate();
};
```

- **U：**当前实例的根输入。
- **Y：**当前实例的根输出；内部状态不在其中。
- **P为static：**多个实例不自动得到独立参数副本；隔离方式须核对生成配置及包装。

4.9.5 从SLX生成C++：模型生命周期

三个函数分别负责开始、推进和结束

- **initialize**：建立模型时间、连续状态和初始条件；每次独立试验只在开始时初始化，不能每步重置。
- **step**：计算当前基本步的输出与状态；M0包含ode4中间级计算，一次step不只代入一次公式。
- **terminate**：结束该实例的运行过程；宿主还需管理自己的线程、日志和外部资源。



图：一次模型试验的函数调用顺序

4.9.5 从SLX生成C++：调用顺序与记录

代码：一次初始化，有限次推进

```
MulticopterModelClass plant;
plant.initialize();
for (int k = 0; k < 1000; ++k) {
  // 在此按接口约定填写全部 U 输入
  plant.step();
  const auto& y = plant.Exp1_MinModelTemp_Y;
  // 在此保存 y 与其模型时间
}
plant.terminate();
```

- **建立起点：** initialize在循环外；每个独立试验调用一次。
- **每轮顺序：** 填全U → step → 读Y；时间戳取公开的模型时间，并核对首末样本时刻。

- **停止条件：** 1000次×1 ms＝1 s模型时间后terminate，与墙钟等待无关；工程编译、地形初始化与日志参照ert_main.cpp及模板。

4.9.5 从SLX生成C++：复用源代码

已有C函数可以接入模型的计算链

- **实际例程：**4.RflySimModel/0.ApiExps/2.UserDefinedC++/1.CallC/code/下的Max.c与Square.c提供比较及平方函数，mfunc.slx经MATLAB Function调用。
- **生成路径：**coder.cinclude声明头文件，coder.ceval调用C函数；编译还要能找到源文件与头文件，数据类型须与uint16_t形参对应。
- **区别：**仅在MATLAB宿主执行的外部调用，不会自动成为可部署模型代码；先用输入2、3检查最大值3及平方结果9。

```
coder.cinclude('Typedef.h');
y = coder.ceval('GetMaxValue', u1, u2);
y = coder.ceval('SquareOperation', y);
```

外部算法进入生成代码以前，先统一函数签名、数据类型与构建依赖。

本章目录

建模工具、代码生成与模型部署

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 7 个教学单元

4.9.1 工具与运行链	教材4.2.1、4.2.2.3
4.9.2 MuJoCo多体建模（拓展）	多关节与接触力
4.9.3 最小模板与可重复工程	教材4.2.2.2、4.2.3
4.9.4 数值时间与调度	步长、多速率、墙钟
4.9.5 从SLX生成C++	教材4.2.3、4.2.6
4.9.6 封装、部署与加载	教材4.2.3、4.2.4
4.9.7 实现一致性与排错	SLX/DLL同条件比较

4.9.6 封装、部署与加载：从代码到可调用产物

生成类与可装载DLL之间还有平台封装

- **生成类：** C++类提供成员和方法；外部程序知道类定义并链接相应代码后，才能直接使用这些成员。
- **平台封装：** 可装载DLL还需按平台约定暴露入口、传递数组和管理实例；接入CopterSim的DLL须符合这套约定。
- **现有路线：** M0配套GenerateModelDLLFile用于平台打包；直接加载例程再由SDK调用DLL导出函数，包装内容以实际产物为准。



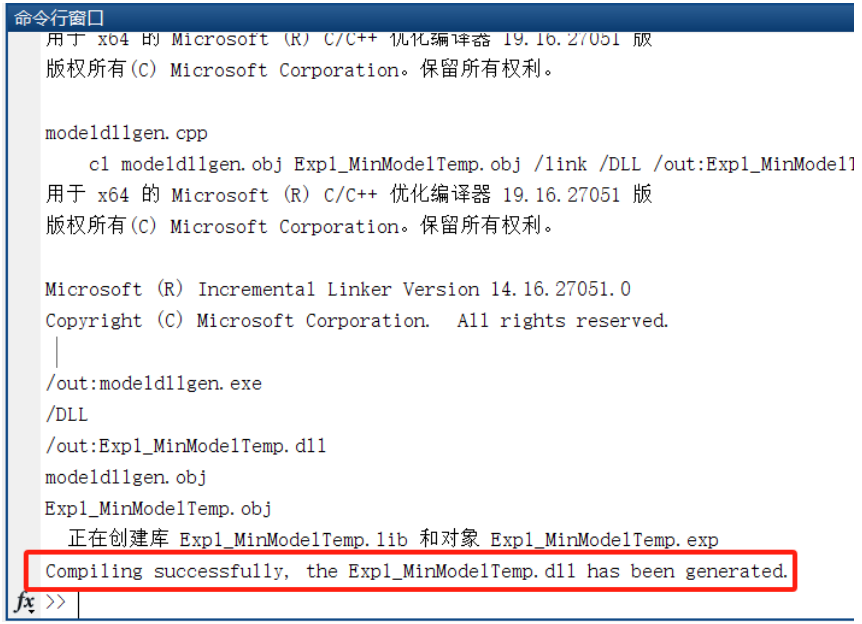
图：代码封装到平台DLL

生成代码提供计算，平台封装提供宿主能够识别的接口约定。

4.9.6 封装、部署与加载：生成产物与构建证据

从生成文件到DLL的可见结果

- Build后检查本模型的_ert_rtw目录和打包产物；在工程目录运行GenerateModelDLLFile。
- 核对新DLL的名称、时间与哈希，保存构建日志；以本次生成的新文件确认构建成功。
- 图为发布资料的成功提示；本机结果以当前日志和新文件为准。



图：DLL生成成功提示的资料示例

生成代码与生成DLL是连续但不同的两个步骤。

4.9.6 封装、部署与加载：部署文件

DLLModel决定加载哪一个动力学文件

```
REM Set use DLL model name or not, use number index or name string
REM This option is useful for simulation with other types of vehicles instead of multicopters
set DLLModel=Exp1_MinModelTemp
```

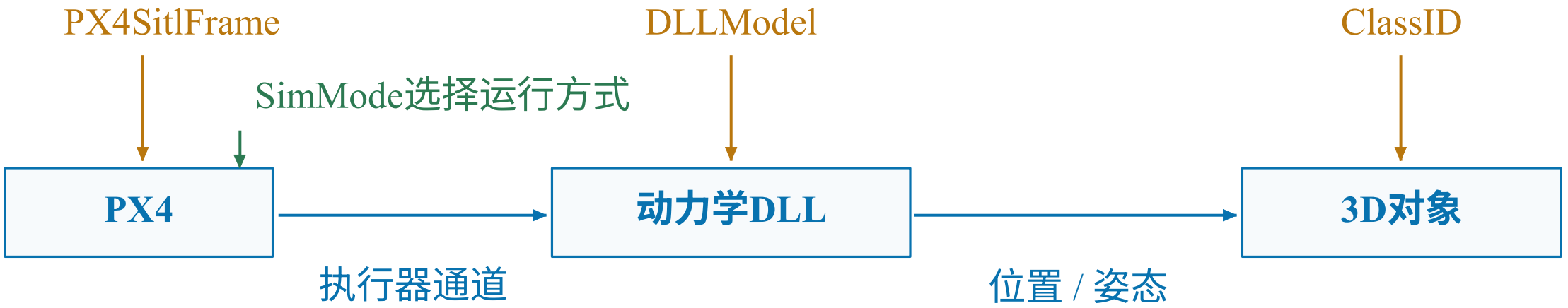
图：DLLModel加载名称（例程资料截图）

- 脚本将DLLModel设为Exp1_MinModelTemp，把同名DLL复制到CopterSim/external/model，再按此名称加载。
- 记录源与目标DLL的哈希以排除旧文件；修改机架或三维外观不会替换动力学DLL。

启动配置与实际部署文件一致，改动才会进入运行链。

4.9.6 封装、部署与加载：启动配置

机架、仿真模式和外观分别配置



图：配置字段指向不同运行对象，不互相替代

- SimMode=2选SITL；iris（四旋翼）机架文件引用混控器并带PID参数集。
- DLL布局决定受力，ClassID决定显示；改结构或质量后在QGC重调PID。

控制通道、运行方式与可视外形共同配套，但含义各不相同。

4.9.6 封装、部署与加载：界面配置与仿真场景

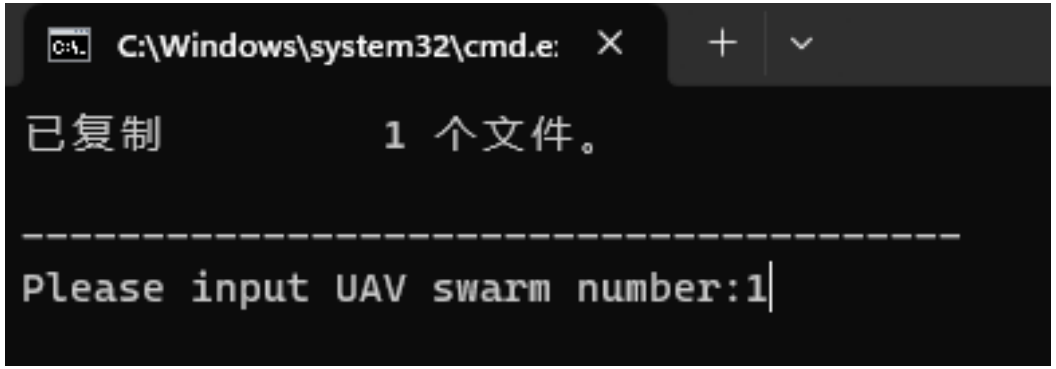
多旋翼可在CopterSim界面配置，场景按载具选择

- **界面配置：**多旋翼可在CopterSim模型配置区设定构型、尺寸和重量，点“计算”后“存储并使用参数”，无需导入DLL；要改方程或换用其他载具时，再按本节流程生成并导入DLL。
- **场景与初始位置：**按载具类型选择三维场景和初始位置，如固定翼需要起飞跑道；M0默认地面坡度为0，只适用于平坦地图。
- **多载具：**各机起点与运行区域可写入BAT启动脚本一键启动，也可用SimCreate生成协同仿真配置，见课件2.4「实验组织」页。
- **说明文档：**CopterSim与SimCreate子目录中的Readme.pdf，位于[安装目录]/RflySimAPIs/2.RflySimUsage/0.ApiExps/e1_RflySimSoftwareReadme/

4.9.6 封装、部署与加载：启动环境

启动前先保存正在进行的实验

- 保存当前模型、日志和其他CopterSim实验，再检查启动脚本中的安装路径、DLL名、模式、机架及载具数量。
- 当前脚本会复制模型文件，并结束已有CopterSim进程；在共享机位运行前结束或保存相关实验，避免丢失现场。
- 按例程说明启动SITL并输入1个载具，观察复制与启动信息；权限是否需要提升按安装目录和机位配置处理。



图：载具数量输入提示（例程资料截图）

批处理会部署文件并组织多个程序，启动具有实际副作用。

4.9.6 封装、部署与加载：初始化完成的证据

初始化就绪以后还要读新状态

- 资料图的CopterSim/PX4 EKF 3D Fixed显示1/1，表示实例及扩展卡尔曼滤波器（EKF）状态估计初始化就绪。
- 随后核对载具编号、位置、姿态和递增时间，再施加小范围已知输入；真值变化应与输入和模型方向相符。



图：初始化就绪（资料截图）

就绪提示说明观测链开始工作，运动验证还需要时间与输入证据。

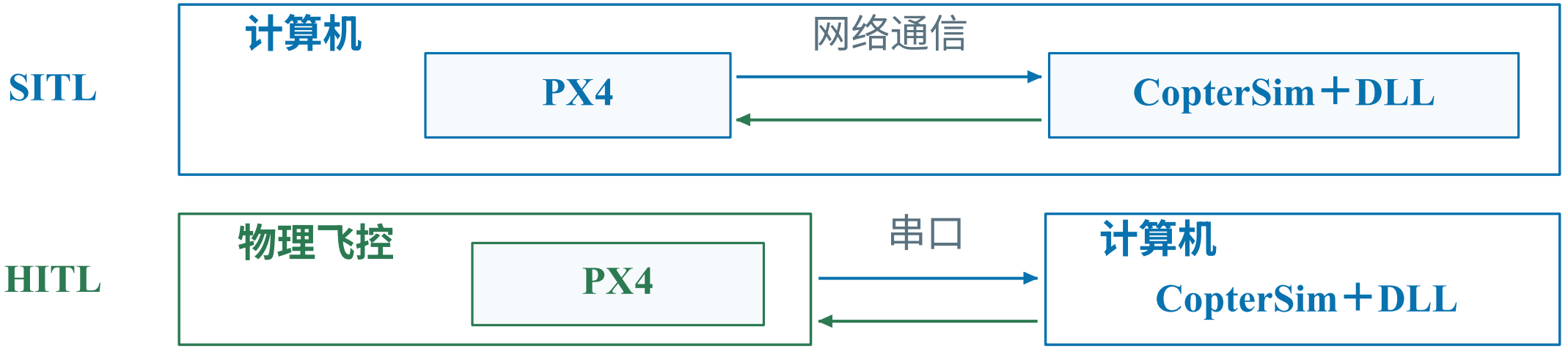
4.9.6 封装、部署与加载：M0例程速查

M0部署例程：入口、步骤、判据与常见失败

- **目录与入口：**先读Readme.pdf，入口为SLX与Exp1_MinModelTemp_SITL.bat；[安装目录]/RflySimAPIs/4.RflySimModel/1.BasicExps/e0_MinModelTemp/
- **步骤：**打开SLX执行Build → 运行GenerateModelDLLFile得到Exp1_MinModelTemp.dll → 以管理员身份运行SITL批处理，提示框输入1。
- **副作用：**脚本把DLL复制到CopterSim/external/model，并结束已有CopterSim进程；运行前保存其他实验。
- **判据：**RflySim3D显示“CopterSim/PX4 EKF 3D Fixed:1/1”，CopterSim提示GPS 3D fixed与EKF初始化完成，再在QGC起飞。
- **常见失败：**未配置mex编译器导致编译失败；“无法加载库文件”多因PX4PSP工具箱未更新；仍运行旧DLL时核对哈希。

4.9.6 封装、部署与加载：在环方式

本章如何使用SITL与HITL



图：动力学仍在计算机，控制器改变运行位置

- 本章以现成控制器核对模型与接口；控制律设计留到第五章。
- SITL便于全班复现；HITL还需固件、串口和硬件时间配套。

先在共同软件环境验证模型，再按硬件条件扩展通信与实时检查。

本章目录

建模工具、代码生成与模型部署

第四章 载具运动建模与仿真

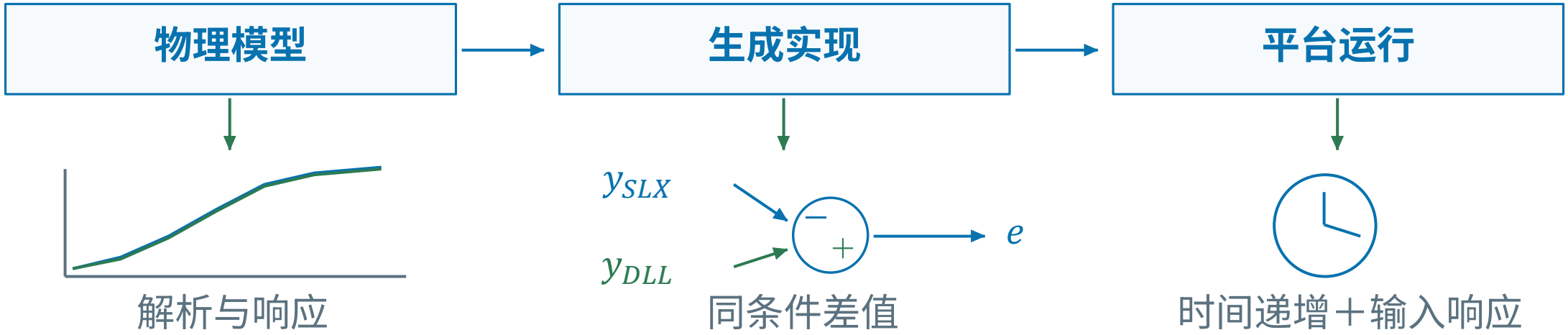
4.1 运动建模基础与运动学
4.2 刚体动力学与运动求解
4.3 动力单元、受力与观测建模
4.4 配平、线性化与动态系统基础
4.5 多旋翼模型逐步编程与仿真
4.6 参数测量与静态模型辨识
4.7 动态系统辨识、验证与模型修正
4.8 典型无人载具建模与扩展
4.9 建模工具、代码生成与模型部署
4.10 API与SDK建模接口
4.11 例程体系与综合实践
4.12 习题案例分析与课程设计

本节 7 个教学单元

4.9.1 工具与运行链	教材4.2.1、4.2.2.3
4.9.2 MuJoCo多体建模（拓展）	多关节与接触力
4.9.3 最小模板与可重复工程	教材4.2.2.2、4.2.3
4.9.4 数值时间与调度	步长、多速率、墙钟
4.9.5 从SLX生成C++	教材4.2.3、4.2.6
4.9.6 封装、部署与加载	教材4.2.3、4.2.4
4.9.7 实现一致性与排错	SLX/DLL同条件比较

4.9.7 实现一致性与排错：成功的不同层次

三层成功分别需要什么证据



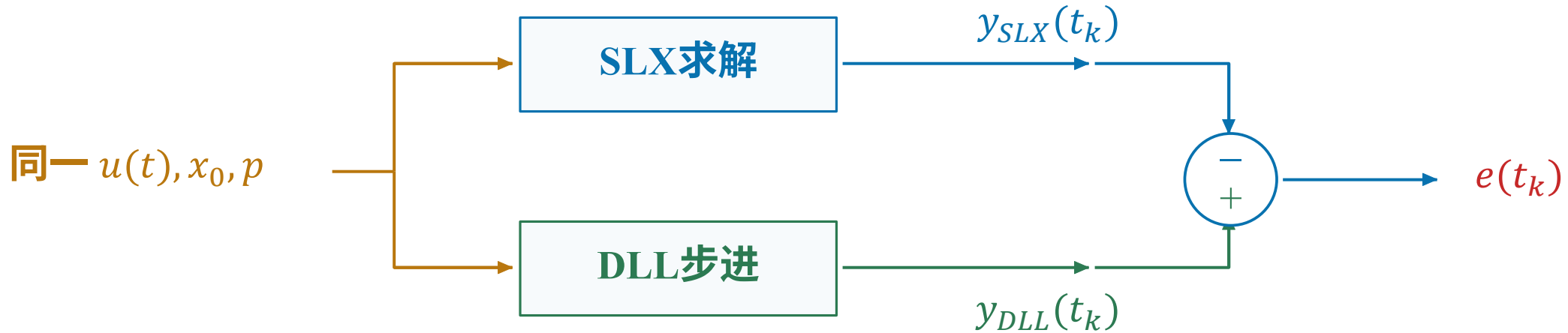
图：在各阶段挂接相应证据

- **物理模型：**解析响应核对受力与运动。
- **生成实现：**条件与系统边界相同，再比较SLX与DLL差值。
- **平台运行：**查递增模型时间，并确认状态响应已知输入。

- 与实物的精度另用课件4.6与4.7的独立数据验证。

4.9.7 实现一致性与排错：同边界对照

给SLX与DLL完全相同的试验



图：共享试验条件的两条平行计算路径

- 参数、初态、输入及环境相同，按同一仿真时刻读取同一量。
- 综合DLL含控制器，须先补同边界入口，再与裸SLX对比。

只有系统边界与条件相同，输出差异才可用于检查实现。

4.9.7 实现一致性与排错：时间与误差

先对齐数据，再计算差值

- 把两份日志的仿真时间单独保存，再选共同记录时刻；需要插值时说明方法，阶跃附近尤其检查输入切换对齐。
- 对每个输出分别计算差值和最大绝对误差，单位沿用原物理量；位置误差、角速度误差不混成没有单位的总分。
- 差值形状指向来源：固定偏移先查初态，阶跃附近尖峰先查采样对齐，随时间累积先查步长与参数；同求解器同步长时差值预期在舍入量级。

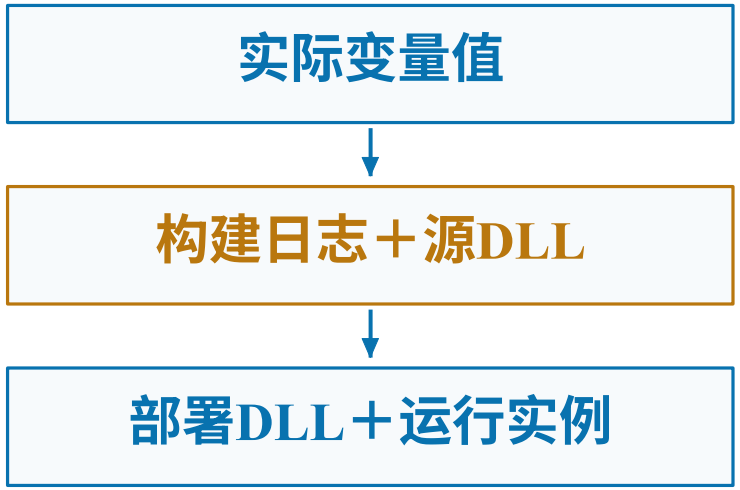
$$e_y(t_k) = y_{\mathrm{DLL}}(t_k) - y_{\mathrm{SLX}}(t_k)$$

同一数组行号可能对应不同时间，误差必须按物理时刻计算。

4.9.7 实现一致性与排错：参数未生效

参数改了而结果没变怎么查

- **改参数却无变化：**依次读取模型实际参数、构建日志、源与部署DLL哈希，再确认运行实例。
- 方向错误先查电机通道、旋向与机架；固定位置偏差先查原点与初态，差值累积再查时间和参数。
- 综合DLL的速度目标先经内部控制器，响应不同于裸动力学输入；课件4.10按接口约定逐项确定边界。



同一构建哈希贯穿部署

参数改了却无响应，要先查工况，再查运行中的模型。

4.9.7 实现一致性与排错：依赖与编译失败

缺库与编译器问题怎样分层处理

无法加载 'pixhawk_slib_adv/CopterForceModel' 引用的库 'pixhawk_slib_adv'。
组件: Simulink | 类别: Block diagram 错误

错误信息：e0_MinModelTemp的Readme.pdf常见问题Q2

- 错误行指向pixhawk_slib_adv库中的模块无法加载：先核对RflySim工具箱路径与版本（Readme称多因工具箱未更新），而不是调整物理参数。
- 若模型更新通过而C++编译失败，再检查编译器选择与完整构建日志；修复后从失败步骤重新验证并保存结果。

读取第一条有效错误，把模型依赖与构建环境分开排查。

4.9.7 实现一致性与排错：状态不更新

能显示却没有新状态怎么查

- 显示窗口存在只说明程序已经打开；检查真值时间是否递增、位置是否响应输入，并核对读取的载具编号。
- 直接DLL链路若没有调用步进函数，读取输出可能一直返回初态；网络链路则要确认目标实例和接收数据正在更新。
- 接口返回零数组也可能是函数缺失的回退结果；把导出支持、输入生效和时间增量一起验证，才能确认取得真实状态。

画面、缓存和模型时间要分别检查，才能确认仿真正被推进。

4.9.7 实现一致性与排错：开发交付

本节交付：一份可追溯的模型实验包

- 接续课件4.5与4.11的离线基准，保存M0来源、参数、初态、输入与三层日志，写明解析对照的物理条件。验收：另一人按记录能复算同一曲线。

- 保存求解设置、步长对照与误差计算；完成生成的机位同时提交构建记录、DLL标识、启动配置和加载证据。验收：部署DLL哈希与构建记录一致，模型时间递增。

- 使用发布DLL完成加载时标明产物来源；缺条件的路径写明所缺条件，物理分析与接口证据分别提交。验收：每条结论都能指到对应日志或文件。

模型、输入、日志和判断一起交付，才能复查完整开发过程。

课堂练习

用本节的判断方法解释四个现象

- **练习1：**教学副本把质量从1.515 kg改为1.8 kg，重新部署后让PX4悬停，高度曲线几乎不变。改动生效了吗？先看哪个量？

- **练习2：**SLX与DLL同条件对比，误差分别是全程约0.2 m的固定偏移、t=2 s输入阶跃处的单个尖峰、随时间线性增大。各先查什么？

- **练习3：**脚本把1500写入inPWMs前四个通道，飞机会怎样？M0悬停时的归一化输入大约是多少？

- **练习4：**用综合DLL的位置响应与裸M0的SLX输出对比，偏差很大。能否据此判定代码生成出错？

参考要点

先判工况与系统边界，再逐层查证据

- **练习1：** PX4闭环会补偿质量，悬停高度可以不变；改看电机输入：按M0映射、取g=9.81 m/s²，悬停输入应由约0.53升到0.58（教学算例）。输入也没变，再依次查InitFcn覆盖、构建日志、源与部署DLL哈希、运行实例。
- **练习2：** 固定偏移先查初态与原点；阶跃处尖峰先查采样与切换时刻是否对齐；线性增大先查是否按行号相减造成时间错位，再查步长与参数。
- **练习3：** 1500被限幅成0.95，四桨总推力约45.5 N，约为重力14.9 N的3倍，飞机急速上升；悬停只需约0.53（教学算例）。
- **练习4：** 不能。综合DLL含控制器，输入是速度目标，系统边界与裸SLX不同；先补相同根输入的测试包装，再按时间对齐比较。

本节小结

让模型从SLX可靠地走到部署

- **方法：**先冻结参数、初态、输入与模型时间；再沿SLX → C++生成类 → 平台封装DLL → 宿主加载的链路，每层留下可复查的证据。
- **关键判断：**Build得到程序文件，Run得到轨迹；部署生效看DLL哈希与递增的模型时间；实现一致性看同边界、按时刻对齐的差值。
- **易错点（输入与参数）：**把1500微秒脉宽写入归一化的inPWMs；在工作区改质量，又被InitFcn覆盖。
- **易错点（部署）：**目录里的旧DLL没有被替换；把Run当成Build，以为已经生成新代码。
- **易错点（比较）：**按数组行号相减；拿含控制器的综合DLL直接对比裸SLX。