本章目录

分布式仿真架构与数据接口

第三章 三维环境建模与仿真

3.1 三维仿真基础与工具选型

3.2 UE场景运行渲染与交互

3.3 多模态传感器数据生成

3.4 分布式仿真架构与数据接口

3.5 场景与载具资源开发

3.6 例程体系与综合实践

3.7 习题案例分析与课程设计

本节 5 个教学单元

3.4.1 仿真架构
教材第3.2.1.1–3.2.1.2节

3.4.2 空间基准
教材第3.2.2.2节；挂载算例补充

3.4.3 时间与传输
教材第3.2.1.1、3.2.1.3节

3.4.4 平台接口与传感器配置
教材第3.2.1.3、3.2.3节

3.4.5 联调排错
教材第3.2.3、3.3.1节

3.4 分布式仿真架构与数据接口

3.4.1 仿真架构：总体框架

资源导入、模型加载与运行数据三条路线

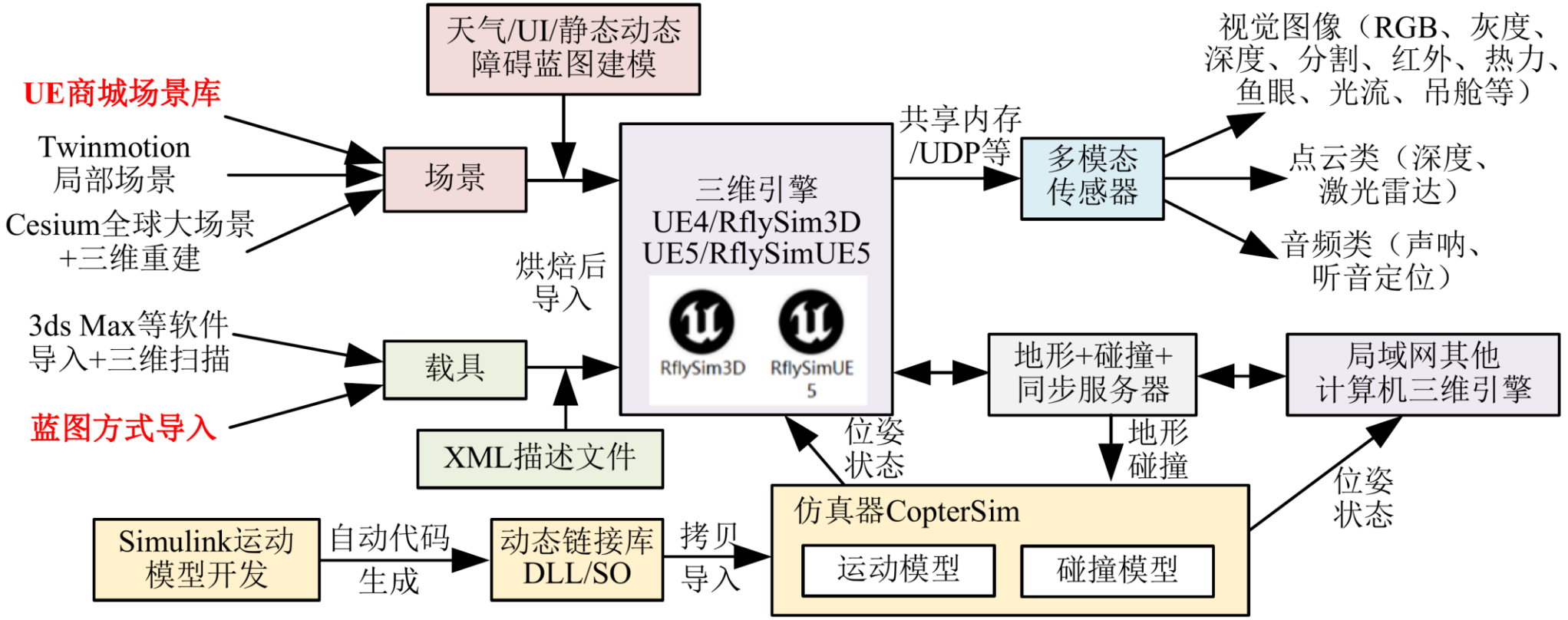


图3.6 RflySim三维仿真总体框架

3.4.1 仿真架构：资源与模型导入

三维外观与运动模型沿两条路径接入平台

- 场景地图：UE原生资源打包加载，Cesium地理数据运行时按需加载。
- 载具外观：网格或蓝图与XML配置进入三维端，按ClassID与DisplayOrder组合索引选模型。
- 运动模型：Simulink生成代码并编译成DLL/SO，由CopterSim在运行时加载。
- DLL是Windows动态链接库；SO是Linux共享库。外观与运动模型分开准备。

外观资源预先部署，运动模型由CopterSim在运行时加载。

3.4.1 仿真架构：运行分工

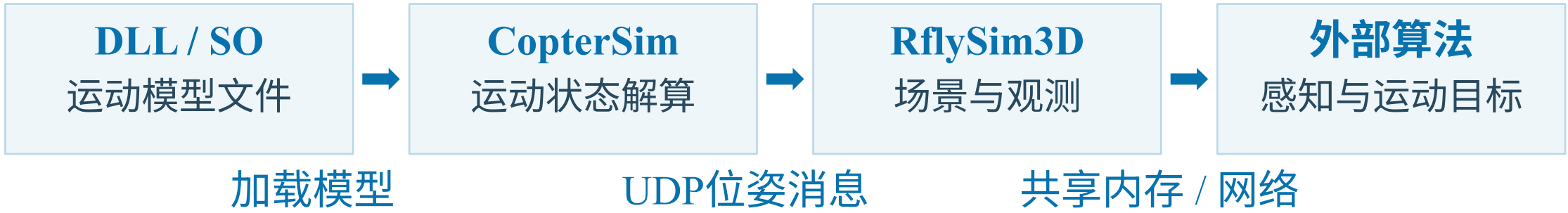
运动由CopterSim解算，三维端负责显示与观测

- CopterSim：加载运动模型 DLL，接收电机或舵面指令，解算位置、姿态、速度与角速度，并处理碰撞响应。
- 三维引擎：RflySim3D（UE4）或 RflySimUE5（UE5）负责场景渲染、几何碰撞检测和传感器成像，不用 UE 物理引擎算刚体运动。
- 同步服务器：把三维端得到的地形高度与碰撞信息回传 CopterSim，使运动解算能“看到”场景。
- 局域网联机：不同计算机上的三维引擎都能接收位姿，并把各自的交互数据回传给仿真器。

CopterSim算运动，三维端算画面与碰撞检测，交互信息经同步服务器回传。

3.4.1 仿真架构：加载与传输

DLL加载与UDP通信连接不同环节

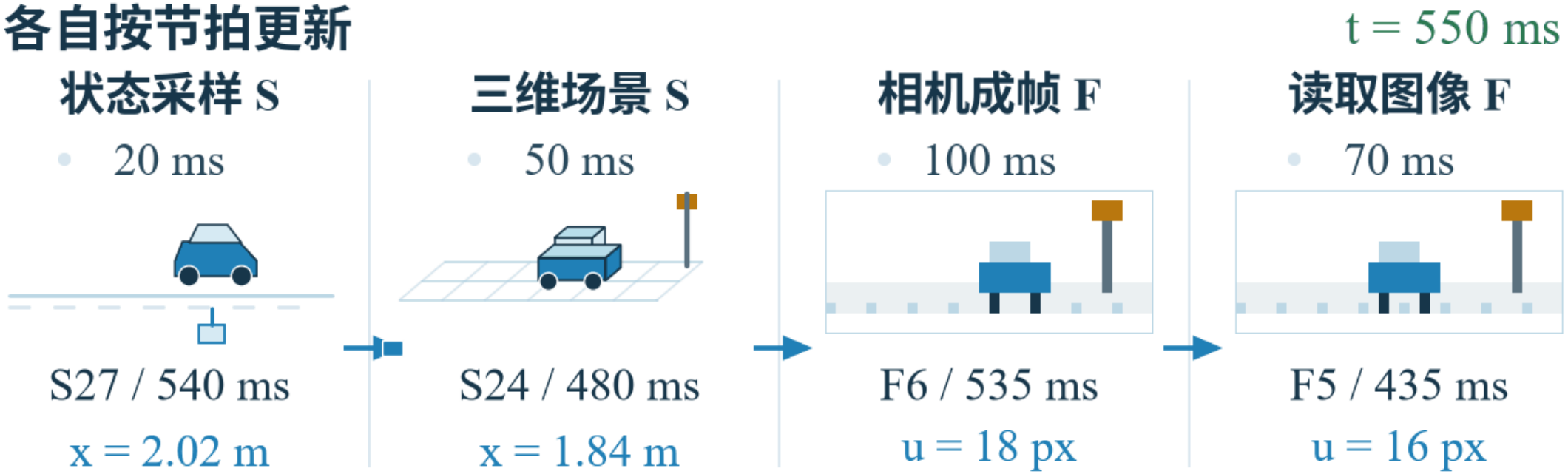


- **动态链接库（DLL）：** CopterSim在进程内调用编译后的运动方程；SO是相应的共享库形式。
- **软件在环路线：** 算法把运动目标送给PX4飞控，飞控控制量作用于运动模型，再产生新的位姿。

动力学库由仿真器加载，位姿消息驱动三维端。

3.4.1 仿真架构：状态与传感器数据链

状态、渲染、采样与读取各有一个核查点



图：状态到图像四段数据链（概念示意）

联调时逐段记录配置、时间戳和数据形状。

3.4.1 仿真架构：主要优势

分布式架构的四项优势

- 降低算力需求：UE 负责显示、传感器渲染与场景几何查询，运动与碰撞响应由 CopterSim 计算，不占用 UE 物理引擎算力。
- 分布式计算：CopterSim 可独立运行，多台计算机分别承担一部分载具，集群规模可随计算机数量扩大。
- 运动模型可定制：在 Simulink 中开发运动模型，用系统辨识优化参数，编译成 DLL 或 SO 后交给 CopterSim。
- 支持大规模仿真：多台计算机协同渲染，分摊大量相机画面的渲染负载（见“渲染负载算例”）。

运动解算与三维渲染分开后，多台计算机可以分摊集群仿真负载。

3.4.1 仿真架构：渲染负载算例

算例：40路1080P图像为什么要多机渲染

- 题设：10架载具、每架4路1920×1080图像，共40路；假定每路30 Hz。
- 单路每秒像素：1920×1080×30≈6221万；40路约24.9亿像素/s。
- 若按RGB每像素3 B原始传输，约7.46 GB/s；这是未压缩算例，不是实测带宽。
- 渲染与传输都可能成为瓶颈；多机分配相机并测每机GPU、链路负载。

$$40 \times 1920 \times 1080 \times 30 \times 3 \approx 7.46 \times 10^{9}\ \mathrm{B/s}$$

40路1080P每秒约25亿像素；是否需多机渲染，应以实测GPU与链路负载判定。

3.4.1 仿真架构：局限与对策

分布式架构的两项局限与应对办法

- 地形与碰撞精度：动力学在 CopterSim 中解算，不用 UE 的 PhysX／Chaos 物理引擎，轮胎形变、泥地打滑等难以模拟，空中任务也须按接地交互与时延要求评估影响。
- 对策一：需要精细地面交互的无人车，改用 RflySim3D 提供的 UE 原生车辆蓝图模型，运动、控制与三维模型都在 UE 中完成。
- 运动同步性：三维端与 CopterSim 运行在不同进程甚至不同计算机上，状态到达有先后和延迟，显示不如单进程方案同步。
- 对策二：数据带时间戳，按时间戳插值，并结合位置、速度、加速度做运动预测和同步滤波（算例见课件3.4.3）。

物理解算移出UE换来扩展性，代价是地面交互精度与跨进程同步。

本章目录

分布式仿真架构与数据接口

第三章 三维环境建模与仿真

3.1 三维仿真基础与工具选型

3.2 UE场景运行渲染与交互

3.3 多模态传感器数据生成

3.4 分布式仿真架构与数据接口

3.5 场景与载具资源开发

3.6 例程体系与综合实践

3.7 习题案例分析与课程设计

本节 5 个教学单元

3.4.1 仿真架构
教材第3.2.1.1–3.2.1.2节

3.4.2 空间基准
教材第3.2.2.2节；挂载算例补充

3.4.3 时间与传输
教材第3.2.1.1、3.2.1.3节

3.4.4 平台接口与传感器配置
教材第3.2.1.3、3.2.3节

3.4.5 联调排错
教材第3.2.3、3.3.1节

3.4.2 空间基准：参考系定义

先定义世界、机体与挂载参考系

- **本地世界系：**选定场景参考点为原点；北东地（NED）以北、东、下为正向，描述对象在环境中的位置。

- **机体系：**前右下（FRD）轴随载具姿态转动。
机体角速度RateB的单位为rad/s。

- **挂载变换：**传感器相对载具具有平移和旋转；求其世界位姿时，先旋转相对位移，再加载具世界位置。

机体轴随姿态旋转，使固定安装偏移改变世界方向。

3.4.2 空间基准：长度单位换算

算例：只把NED米制位置换为厘米

题设：UE 场景内部以厘米计长度；把 NED 位置 [2, 3, −1] m 换成厘米，原点与轴向不变。

北向N	东向E	地向D
2 m → 200 cm	3 m → 300 cm	−1 m → −100 cm

$$p_{\mathbf{NED,cm}} = 100 \times p_{\mathbf{NED,m}} = [200, 300, -100]\ \mathbf{cm}$$

负D在原点上方；按米接收的接口仍传 [2, 3, −1]，不能先乘100。

长度换算只改变数值比例，不改变参考系。

3.4.2 空间基准：轴向变换

算例：明确目标轴向后再转换位置

题设：两系原点重合，目标X朝北、Y朝东、Z向上，单位cm。

源NED位置
[2, 3, −1] m

→

目标世界位置
[200, 300, 100] cm

$$[X, Y, Z] = 100 \times [N, E, -D]$$

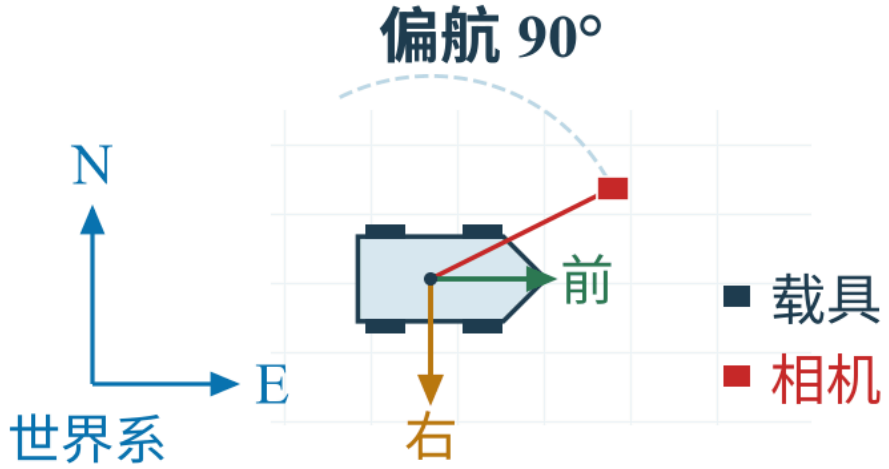
Z向上而D向下，所以第三分量反号；欧拉角须另作姿态变换。

轴向变换与长度换算共同决定目标坐标。

3.4.2 空间基准：相机挂载变换

算例：配置中的相机安装位置怎样映射到世界

题设：机头朝东（偏航90°），载具位于[2, 3, −1] m；0号相机SensorPosXYZ＝[0.3, −0.15, 0] m，即前0.3 m、左0.15 m。



图：相机随车转向，世界轴不动

$$p_{\mathbf{c}} = p_{\mathbf{b}} + Rr$$

p_{b} / p_{c}：载具 / 相机世界位置
R：机体系到世界系的旋转矩阵
r：机体系中的相机安装偏移

$$Rr = [0.15, 0.3, 0]\ \mathrm{m}$$
$$p_c = [2.15, 3.3, -1]\ \mathrm{m}$$

挂载偏移先随载具旋转，再加到世界位置。

本章目录

分布式仿真架构与数据接口

第三章 三维环境建模与仿真

3.1 三维仿真基础与工具选型

3.2 UE场景运行渲染与交互

3.3 多模态传感器数据生成

3.4 分布式仿真架构与数据接口

3.5 场景与载具资源开发

3.6 例程体系与综合实践

3.7 习题案例分析与课程设计

本节 5 个教学单元

3.4.1 仿真架构
教材第3.2.1.1–3.2.1.2节

3.4.2 空间基准
教材第3.2.2.2节；挂载算例补充

3.4.3 时间与传输
教材第3.2.1.1、3.2.1.3节

3.4.4 平台接口与传感器配置
教材第3.2.1.3、3.2.3节

3.4.5 联调排错
教材第3.2.3、3.3.1节

3.4.3 时间与传输：频率与时间域

频率描述节奏，时间域定义时刻

环节	渲染显示	相机采样	控制计算
题设频率	60 Hz	30 Hz	100 Hz
对应周期	16.7 ms	33.3 ms	10 ms

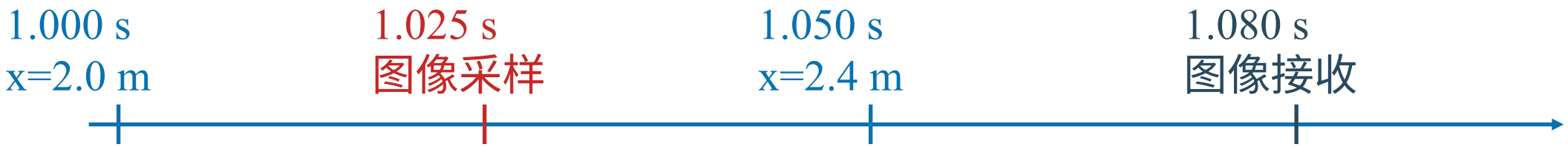
- **仿真时间：**按模拟步推进，暂停或加速时可偏离真实时间；本机耗时用单调时钟测量。
- **时间映射：**校准时钟偏移和快慢，把状态与图像映射到同一时间域，再比较时间戳。

统一时间原点与时钟尺度后，时间戳才可比较。

3.4.3 时间与传输：插值配对

算例：按采样时刻插值得到对应状态

题设：时间已同域，采样时刻两侧状态均已取得；区间内近似匀速。



$$\alpha = \frac{1.025 - 1.000}{1.050 - 1.000} = 0.5$$

$$x(1.025) = (1 - \alpha) \times 2.0 + \alpha \times 2.4 = 2.2\ \mathbf{m}$$

采样到接收为55 ms，包含生成和等待；配对用采样时刻的2.2 m。

接收时刻用于衡量延迟，采样时刻用于匹配状态。

3.4.3 时间与传输：运动预测

算例：后一个状态未到时按速度外推

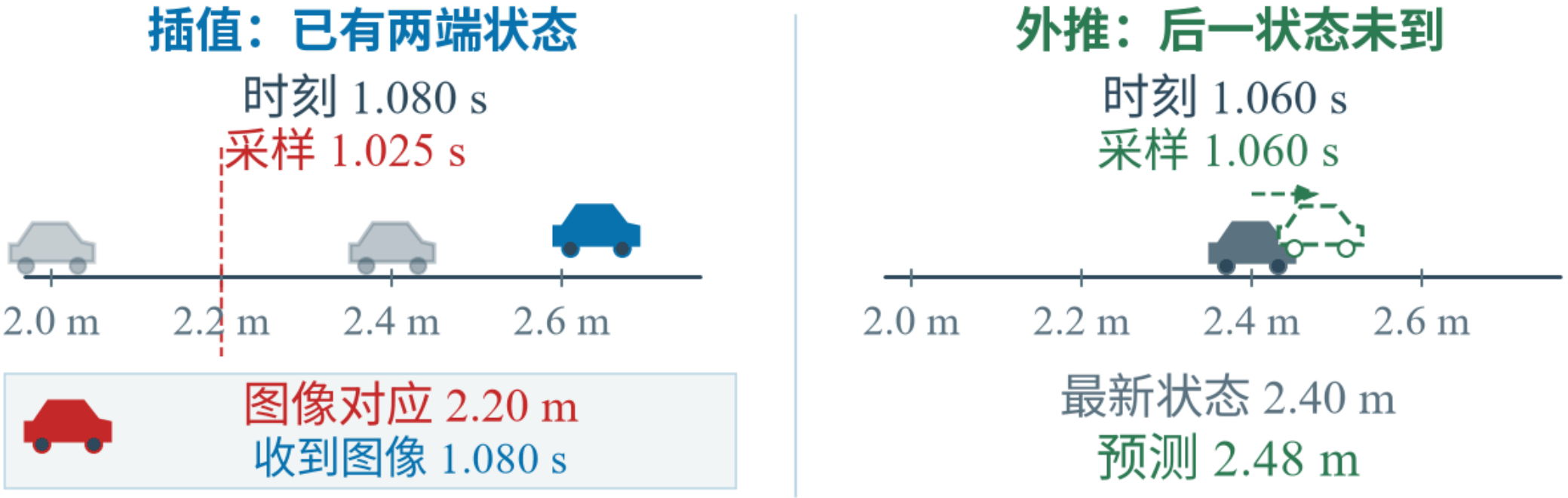
- 题设：x(1.000 s)=2.0 m，x(1.050 s)=2.4 m；图像采样于1.060 s，后一状态尚未到达。
- 前一区间估计速度：(2.4−2.0)/0.050=8 m/s。

$$\hat{x}(1.060) = 2.4 + \frac{2.4 - 2.0}{0.050} \times 0.010 = 2.48\ \text{m}$$

- 外推：$\hat{x}(1.060)=2.4+8\times0.010=2.48$ m；直接用2.4 m会滞后0.08 m。
- 外推是估计，不是测量；后续状态到达时应重新校正。

3.4.3 时间与传输：采样时刻的状态

图像晚到时，仍要找回采样时刻的位置



图：实线车辆表示场景或已知状态，虚线车辆表示外推估计

有两端状态就插值；缺少后一状态时，先按运动模型预测。

3.4.3 时间与传输：原始图像数据率

算例：RGB数据率由像素数与频率相乘

题设：640×480、RGB每通道8 bit、30 Hz，不计压缩与协议开销。

单帧像素	×	每像素	×	每秒帧数
640 × 480		3 × 1 B		30

$$640 \times 480 \times 3 \times 30 = 27{,}648{,}000\ \mathbf{B/s}$$

约26.4 MiB/s，或221 Mb/s；压缩节省字节，但增加编解码时间。

原始数据率给出量级，编码与传输还产生额外代价。

3.4.3 时间与传输：图像传输方式

同机用共享内存，跨机用UDP传图像

- 本机共享内存：生产端和消费端共用内存，需核对写完标志、时间戳和数组形状。
- 跨机UDP：图像可能先编码或分包，接收端须按当前SDK例程重组、解码并识别丢包。
- 未压缩图像字节率=宽×高×通道数×每通道字节数×帧率；网络协议另有开销。
- SendProtocol/sendMode文档与实现版本有差异；先核当前配置和代码，不在此页硬编码编号含义。

共享内存省去网络传输，UDP能跨计算机，但要分包并承受丢包。

本章目录

分布式仿真架构与数据接口

第三章 三维环境建模与仿真

3.1 三维仿真基础与工具选型

3.2 UE场景运行渲染与交互

3.3 多模态传感器数据生成

3.4 分布式仿真架构与数据接口

3.5 场景与载具资源开发

3.6 例程体系与综合实践

3.7 习题案例分析与课程设计

本节 5 个教学单元

3.4.1 仿真架构
教材第3.2.1.1–3.2.1.2节

3.4.2 空间基准
教材第3.2.2.2节；挂载算例补充

3.4.3 时间与传输
教材第3.2.1.1、3.2.1.3节

3.4.4 平台接口与传感器配置
教材第3.2.1.3、3.2.3节

3.4.5 联调排错
教材第3.2.3、3.3.1节

3.4.4 平台接口与传感器配置：二次开发

Python、Simulink与蓝图三类二次开发接口

- API规定调用方式，SDK提供实现、示例与文档。
- Python：通过控制接口向三维端发送命令/位姿并读取返回，用于自动化控制。
- Simulink：以结构体或模块接口交互，用于模型与三维显示联动。
- 蓝图：在三维端内部扩展对象行为，外部命令触发自定义动作。

Python与Simulink在三维端外发命令，蓝图在三维端内实现自定义动作。

3.4.4 平台接口与传感器配置：传感器类型

RflySim3D按类型编号提供图像、点云和声音传感器

- 图像｜1彩色、2深度、3灰度、4分割、8鱼眼｜免费版可用
- 付费图像｜9吊舱、40红外灰度、41红外伪彩色｜仅付费版
- 点云｜20/21激光雷达（本体/世界坐标）、22/23 Livox式扫描、7深度点云｜20–23免费版可用；7仅付费版
- 其他｜5测距、10光流、30简易视觉、31简易雷达、6声音｜6声音仅付费版，其余免费版可用

先按任务选传感器类型，再核对版本条件，避免配置了用不了的类型。

3.4.4 平台接口与传感器配置：配置实例

传感器编号、载体编号与数组下标各有职责

TypeID: 1　　SeqID: 0

TargetCopter: 1　TargetMountType: 0

DataWidth: 640　DataHeight: 480

CameraFOV: 90　DataCheckFreq: 30

- **配置单位：** CameraFOV=90为90°视场；DataCheckFreq=30请求30 Hz更新。实际接收频率还受渲染与传输影响。
- **安装关系：** 模式0随载具运动；SensorPosXYZ设置相对中心的偏移。TargetCopter=1绑定载具1；SeqID为传感器编号，i为列表下标。

TypeID选数据类型，SeqID标传感器，TargetCopter指定挂载对象。

3.4.4 平台接口与传感器配置：显示状态消息

一条显示状态消息有哪些字段

ue.sendUE4PosFull(1001, 3, [0]*8, [1,0,0], [2,3,-1], [0,0,0], [0,0,1.5708], 0)

字段	本例取值	含义与单位
PosE	[2, 3, −1]	北东地（NED）位置，m
VelE	[1, 0, 0]	NED 速度，m/s
AngEuler	[0, 0, 1.5708]	滚转、俯仰、偏航，rad

copterID=1001为独立算例实例；vehicleType=3选四旋翼；windowID=0选窗口。

对象身份、参考系与单位须一同约定，位姿才有确定含义。

3.4.4 平台接口与传感器配置：实例生命周期

实例ID决定更新和删除的目标



图：用 RflyDelVehicles 删除 1000、1001 号载具（录屏）

同一ID再次接收位姿，
更新对应对象。

先停止这些ID的发送，
再执行删除命令。

RflyDelVehicles "1000,1001"

持续发送同一ID的位姿，会使已删除的对象重新创建。

3.4.4 平台接口与传感器配置：相机状态与像素

相机状态结构与像素数组走两条路线

状态查询

```
ue.reqCamCoptObj(0, 0)
ue.initUE4MsgRec()
ue.getCamCoptObj(0, 0)
```

CameraData：标识、位姿与宽高，不含像素。
目标：本例查询SeqID=0的相机。
接收：未收到状态时返回0，先核请求与接收。

图像采集

```
vis.jsonLoad()
vis.sendReqToUE4()
vis.startImgCap()
```

vis.Img[i]：按配置列表项读取像素。
SeqID：标识相机，单独核对列表下标。
新帧：hasData[i]可读缓存；再核采样时间戳。

3.4.4 平台接口与传感器配置：接收数组

一帧图像同时携带像素和采样信息

```
img = vis.Img[i]
shape, dtype = img.shape, img.dtype
sensor_id = vis.VisSensor[i].SeqID
sample_time = vis.timeStmp[i]
```

- **生成链：** Scene Capture 2D（场景采集）→ Render Target（图像缓冲），再经编码和接收成为数组。
- **字段定义：** vis为采集对象，i为列表下标，SeqID标识传感器。代码用于认识字段；连续采集时需将同帧图像和时间戳成对保存。

显示颜色来自数组解释，采样信息说明这帧数据属于谁、何时产生。

3.4.4 平台接口与传感器配置：数组形式

各类图像数组的形状、元素类型与通道顺序

观测	数组形式	如何解释
彩色/分割	H×W×3；uint8；BGR	元素顺序依次是蓝、绿、红
16位PNG深度	H×W；uint16	每个整数保留深度原码
原始深度传输	H×W×1；uint16	末维为1，可统一为二维

深度原码： uint16保存整数编码；接收时不自动换算为米。
换算方法：先按无效码生成掩码，再按量化步长换算距离。
参数含义：步长与无效码由配套协议规定；课件3.3.2“深度原码换算”给出示例。

BGR决定颜色索引顺序，uint16深度需要距离编码而不是颜色解释。

3.4.4 平台接口与传感器配置：消息方向与端口

具体端口区分命令、状态和图像

数据方向	地址与端口示例	用途
Python → 三维窗口0	127.0.0.1 : 20010	位姿、控制台命令
三维端 → 状态接收器	224.0.0.10 : 20006	相机/对象状态
三维端 → 图像接收机	192.168.1.20 : 10001	**首项1：** UDP编码图像

跨机UDP示例：SendProtocol = [1,192,168,1,20,10001,0,0]

- **地址方向：**命令发向三维端，图像返回算法端；127.0.0.1只指本机。

3.4.4 平台接口与传感器配置：结果判读

按画面形状、打印字段和常见故障判读结果

传感器类型、配置与输出编码

https://rflysim.com/doc/zh/4.DevRef/vis-cfg.html

图像采集接口与调用示例

https://rflysim.com/doc/zh/4.DevRef/pysdk/vision/viscap.html

取图入口：先运行GetCamObjDemo.bat，再由Python38Run.bat启动GetCamObjDemoWithCam.py。

GetCamObjDemo.py仅查询状态，不采集像素。

预期：Img0、Img1两路画面，彩色数组为(480,640,3)。

判读：若打印两个640，检查DataWidth是否重复；无图像则检查取图请求与接收。

先解释观测如何生成，再用数组、配置和时间信息组织实验。

本章目录

分布式仿真架构与数据接口

第三章 三维环境建模与仿真

3.1 三维仿真基础与工具选型

3.2 UE场景运行渲染与交互

3.3 多模态传感器数据生成

3.4 分布式仿真架构与数据接口

3.5 场景与载具资源开发

3.6 例程体系与综合实践

3.7 习题案例分析与课程设计

本节 5 个教学单元

3.4.1 仿真架构
教材第3.2.1.1–3.2.1.2节

3.4.2 空间基准
教材第3.2.2.2节；挂载算例补充

3.4.3 时间与传输
教材第3.2.1.1、3.2.1.3节

3.4.4 平台接口与传感器配置
教材第3.2.1.3、3.2.3节

3.4.5 联调排错
教材第3.2.3、3.3.1节

3.4.5 联调排错：字段诊断

诊断实例：日志宽高为何与配置不一致

例程GetCamObjDemoWithCam.py末尾的打印语句

```
print('Camera: ', Cam.SeqID, Cam.PosUE,
      Cam.DataWidth, Cam.DataWidth)
```

- **字段证据：**Config中DataWidth=640、DataHeight=480；日志重复读取宽度，因此出现640、640。
- **正确解释：**CameraData给状态元数据；图像尺寸另由vis.Img[i].shape判定，预期BGR数组为(480,640,3)。
- **定位分支：**状态存在而图像缺失，查看取图请求与hasData；数组完整而处理滞后，再比较图像时间戳与处理时刻。

先定位被打印的字段，再判断数据链路是否异常。

3.4.5 联调排错：分段定位

从请求到读取，逐段找出数据断在哪里

- 请求：确认取图请求已经发出，并记录返回值和目标窗口/传感器编号。
- 生成：检查三维端是否按当前配置创建传感器，采样时间戳是否推进。
- 回传：核对共享内存或UDP接收地址、端口、分包状态与防火墙。
- 读取：看hasData、数组形状及新旧时间戳；每次只变一个条件。

每段用本段的输出作证据，一次只改一个条件，才能找到断点。

3.4.5 联调排错：接口学习资源

理解消息字段并追踪数据传输

读取显示状态字段：核对PosE单位与对象查询返回结构

https://rflysim.com/doc/zh/4.DevRef/pysdk/ue/ue4ctrl.html

区分图像传输方式：追踪共享内存、UDP分包与返回地址

https://rflysim.com/doc/zh/4.DevRef/vis-comm.html

理解对象生命周期：比较停止更新、按ID删除与重新创建

https://rflysim.com/doc/zh/4.DevRef/3d-keys.html

消息字段约定对象、单位与时间，传输配置决定数据到达哪个接收端。

课堂练习

课堂练习：挂载、带宽与排错三道小题

- 题设规定机体系前右下、世界系北东地：机头朝东，机体在[2,3,−1] m；相机相对机体[0.3,0.15,0] m，求世界位置。
- 带宽：两台640×480、30 Hz的8位RGB相机，未压缩每秒多少字节？每帧按60,000 B净荷上限拆几包？
- 排错：状态正常、图像hasData[0]为假，按请求、生成、回传、读取四段列证据。

收到数据只是起点，还要核对空间含义、传输量级与链路完整性。

参考要点

三道小题的参考要点与自查判据

- 挂载：右轴朝南，旋转偏移为[−0.15,0.3,0] m，故相机在[1.85,3.3,−1] m；这是题设坐标系，不代替SensorPosXYZ版本核验。
- 带宽：一台27,648,000 B/s，两台55,296,000 B/s；单帧921,600 B，按60,000 B净荷需16包。
- 排错：先查请求和配置，再查接收方式、地址端口与分包，最后看缓存标志和时间戳。

数值题先查旋转方向与单位，排错题先查请求再查回传。

本节小结

对象、空间与时间三个约定把数据对应起来

- 架构分工：运动模型 DLL 在 CopterSim 中解算，三维端负责渲染、碰撞检测与传感器数据，外部程序经 UDP 发命令、收数据。
- 空间约定：先确认参考系与单位；安装偏移先随姿态旋转再加到载具位置，米与厘米、向下与向上分步换算。
- 时间约定：配对用采样时刻，延迟看到达时刻；后一状态未到时外推只是估计，要用新状态校正。
- 易错点：还在发送的 ID 删了会被重建；hasData 为真不代表新帧；日志字段先核对打印语句再判断链路。

对象编号、坐标单位和采样时刻一致，跨程序数据才能对应。