本章目录

第二章习题与课堂讨论

第二章　实验平台配置

2.1　关键工具介绍

2.2　平台安装与首次仿真

2.3　平台架构与开发流程

2.4　API与SDK开发接口

2.5　例程体系与实验路线

2.6　习题与课堂讨论

本节 2 个教学单元

2.6.1 概念与方法

题1—6：模式、数据流与开发环境

2.6.2 实践与迁移

题7—10：首跑、日志与闭环实验

随后：课堂讨论 → 课后实践 → 本章小结

教材对应：第2.5节习题

2.6 习题与课堂讨论

2.6.1 概念与方法：教材题2

题2：沿整体框架追踪一次控制

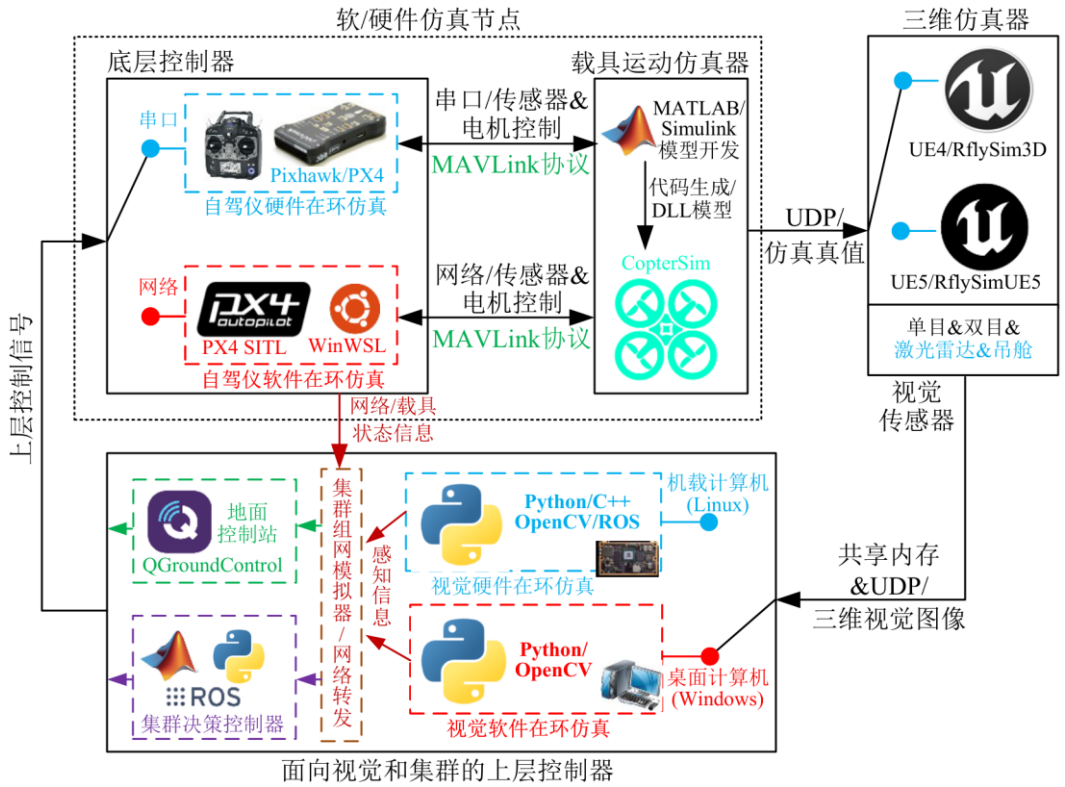


图2.9 RflySim工具链整体框架

- **题设：**本章默认单机SITL；按图2.9说明四个工具的职责。
- **控制与运动：**WinWSL承载PX4；CopterSim计算载具运动。
- **场景与监控：**RflySim3D生成场景与观测；QGC配置、监控载具。
- **追踪接口：**说出模拟测量、执行量、位姿和图像各由谁发出、谁接收。

2.6 习题与课堂讨论

2.6.1 概念与方法：教材题2讲评

不同数据走不同链路

数据链	发送与接收关系	数据含义
底层闭环	PX4 → CopterSim CopterSim → PX4	执行量 模拟传感测量
视觉反馈	CopterSim → RflySim3D RflySim3D → Python	位姿等仿真真值 图像等观测数据
上层与监控	QGC／Python ↔ CopterSim ↔ PX4	配置、运动目标与状态

接口：底层使用MAVLink；位姿走UDP；图像可走共享内存或UDP。

执行量与模拟测量闭合底层回路，位姿与图像连接上层视觉反馈。

2.6 习题与课堂讨论

2.6.1 概念与方法：教材题1

题1：什么问题需要真实控制板

同一四旋翼模型：甲组修改位置控制算法；乙组检查Pixhawk板上计算节拍。分别先选哪种模式?

- **甲组选SITL：**PX4在计算机中运行，经网络连模型，同一输入可反复对照算法。
- **乙组选HITL：**PX4在Pixhawk上运行，经USB串口连模型，检查板上计算节拍。
- **共同模拟对象：**载具运动与传感反馈由模型生成；

电机负载、供电波动和振动属于实物条件。

- **验证阶段：**开发初期用SITL快速迭代，部署前用HITL联调，最后做整机试验。

SITL验证软件闭环；HITL增加目标控制硬件，载具仍由模型模拟。

2.6.1 概念与方法：教材题3

题3：6C代码与固件生成环境

编译命令：px4_fmu-v6c_default（Pixhawk 6C）；烧录另需6C实物与USB数据线

- **环境匹配：**先用PX4Switch选支持6C的源码版本与编译目标；
版本已兼容时，可用PX4CMD('px4_fmu-v6c_default')仅改目标。
- **原生固件：**WinWSL按该命令编译PX4源码，生成6C用的.px4固件。
- **模型代码：**Simulink控制器用SimulinkPSP模块经uORB读传感、发控制，自动生成C/C++模块px4_simulink_app。
- **编入烧录：**px4_simulink_app随PX4编入6C固件；USB连接后用PX4Upload上传，再核对版本与心跳。

模型代码实现控制运算，6C固件还集成板级驱动与启动程序。

2.6 习题与课堂讨论

2.6.1 概念与方法：教材题4

题4：WSL1与WSL2的差异和默认选择

比较项	WSL1	WSL2
运行机制	翻译系统调用，内存占用低	虚拟机运行完整Linux内核
文件读写	读写Windows盘符较快	本系统内快，跨系统读写慢
网络通信	共用Windows网络，互联简单	Win11 22H2起可用镜像网络
容器与GPU	不支持Docker与GPU加速	支持Docker与GPU，内存较高
默认选择	RflySim默认：编译与SITL	Docker、GPU等高阶需求

- **切换与确认：** RflyTools中WslSwitch2输入1或2；uname -a含WSL2即为WSL2。

基础首跑沿用已验证环境；需要Linux内核和容器后端时选择WSL2。

2.6.1 概念与方法：教材题5

题5：MAVROS怎样连接两端



- **协议与桥接：**MAVLink是自驾仪侧消息协议；ROS 1/2用话题和服务组织节点（ROS 2底层经DDS）；MAVROS在两者之间双向转换。

- **状态上行（绿）：**/mavros/state的mode给出实际模式，armed给出解锁状态。

- **目标下行（蓝）：**发布/mavros/setpoint_position/local；/mavros/set_mode仅请求切换，实际结果回读mode。

2.6.1 概念与方法：教材题6

题6：为何用Docker，怎样生成ARM64镜像

- **为何用Docker：**镜像把算法连同库和版本一起打包，统一用户态依赖，少配环境、少依赖冲突；容器共享宿主内核，比虚拟机轻。
- **跨架构构建：**Dockerfile（构建步骤清单）选ARM64基础镜像与依赖，x86主机借QEMU模拟、ARM节点或交叉编译生成ARM64程序。

```
docker buildx build --platform linux/arm64 \
 -t course-demo:arm64 --load .
```

- **部署验收：**--load只载入本机，镜像经仓库或导出文件送到ARM64目标机；GPU与外设靠驱动和设备映射；核对输出与时延并存日志。

跨架构构建：https://docs.docker.com/build/building/multi-platform/

镜像封装用户态依赖，ARM64程序仍依赖宿主内核与驱动。

本章目录

第二章习题与课堂讨论

第二章　实验平台配置

2.1　关键工具介绍

2.2　平台安装与首次仿真

2.3　平台架构与开发流程

2.4　API与SDK开发接口

2.5　例程体系与实验路线

2.6　习题与课堂讨论

本节 2 个教学单元

2.6.1 概念与方法

题1—6：模式、数据流与开发环境

2.6.2 实践与迁移

题7—10：首跑、日志与闭环实验

随后：课堂讨论 → 课后实践 → 本章小结

教材对应：第2.5节习题

2.6.2 实践与迁移：教材题7

题7：安装结束后还要检查什么

已知：安装器正常退出、TestWSL成功，但没有初始化和起降记录。
能否判定基本验证完成？

- **判断：**不能。TestWSL只说明WSL可用，初始化和起降记录才证明闭环已接通。
- **安装前：**Win10 1903及以上或Win11，纯英文路径，安装盘至少100 GB可用；EnableWSL.bat启用WSL后重启。
- **一键安装：**首次用默认配置，记录安装路径、编译命令、固件版本和编译器。
- **完整首跑：**SITLRun输入1，等三维窗口显示EKF 3DFixed；QGC在Hold模式下Takeoff，完成起降、上锁并正常退出。

环境检测确认程序条件，起降记录体现测量、控制与运动的闭环。

2.6.2 实践与迁移：教材题8

题8：SITL四阶段与飞行日志

起飞		悬停		前飞		降落
离地3 m	➡	保持10 s	➡	前移5 m 航向不变	➡	接地并上锁

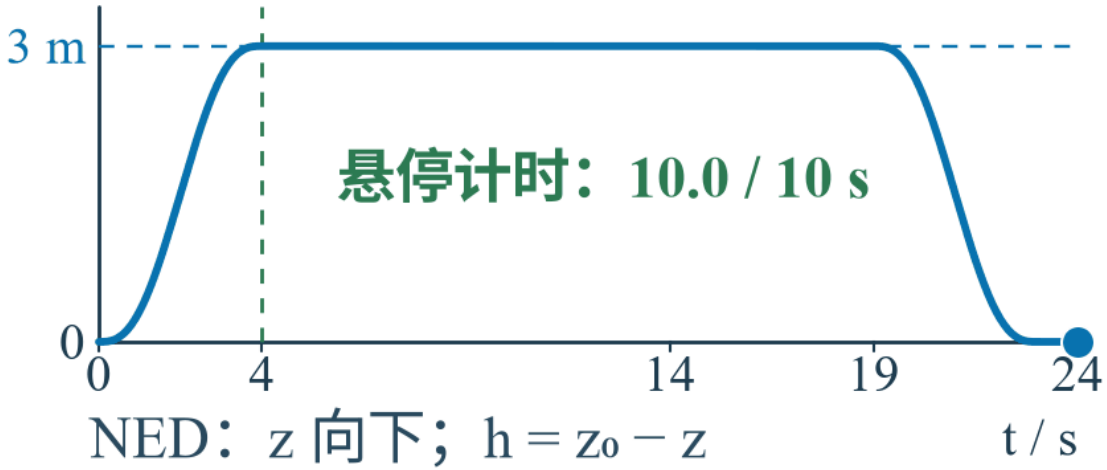
- **状态条件：**平坦场景，NED原点固定在起点；到达3 m后才开始计悬停10 s。
- **数据选择：**按e13_Log-Get记录本次CopterSim真值CSV，PX4内部量取同次飞行日志（ULog）；两者按时间对齐。
- **完成判据：**z向下为正，离地高度$h = z_0 - z$（z_0为起点值）；前飞5 m取水平位移在起始航向上的投影，航向不变，接地上锁。

任务阶段、运动响应和同一次运行的日志应能按时间相互对应。

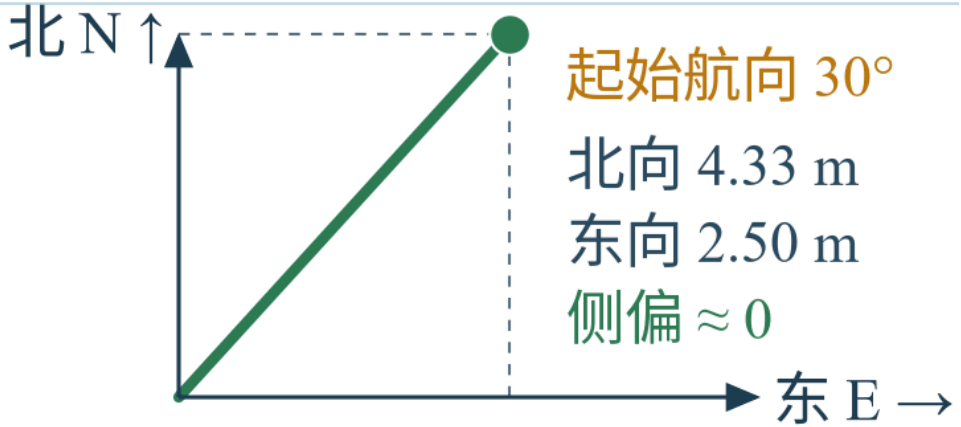
2.6.2 实践与迁移：教材题8讲评

四阶段任务怎样按时间判读

高度 h(t)：到达 3 m 才开始计时



沿起始航向前飞 5 m



起飞	悬停 10 s	前飞 5 m	降落 / 上锁

图：四阶段任务的时间轴判读（示意曲线，非仿真数据）

悬停从到达3 m起计；前飞5 m沿起始航向量测（非实测）。

2.6.2 实践与迁移：教材题9

题9：把同类任务迁到HITL



- **硬件连线：**Pixhawk经USB数据线接仿真电脑；拆桨，断开电机等执行机构。
- **固件与模式：**QGC刷入匹配固件并设为HITL模式；运行HITLRun并输入串口号，等初始化完成。
- **起降验收：**QGC中起飞、悬停、降落并记录；接地上锁后退出，恢复改动配置。

HITL增加真实自驾仪计算与通信，起飞和降落仍由虚拟载具完成。

2.6.2 实践与迁移：教材题10

题10：构建视觉参与的控制闭环

图像生成 → **图像处理** → **运动控制**

按VisCreate相机配置出图 → Python求目标误差 → Offboard发速度目标

载具运动后，三维场景生成下一帧观测

- **开始：** SITL初始化，取图与外部控制（Offboard）分别测通后起飞。
- **反馈：** 横向偏差定偏航速率，纵向偏差定升降速度，面积未达阈值时前进。
- **结束与异常：** 完成或超时后降落，接地上锁再停通信；新帧无目标可限时搜索，帧超时判为失图，先发保持目标覆盖旧指令。

图像误差改变运动目标，运动再改变新观测，形成视觉反馈。

2.6.2 实践与迁移：教材题10讲评

控制是否用到了图像

ex = p_i[0] - width / 2; yawrate = K_yawrate * ex

题设：前视相机，图宽640像素，横坐标向右；K_yawrate=0.00375，前右下机体系。

给定观测	横向误差	本式计算的偏航角速率
A：p_i[0]=400（偏右80像素）	ex = +80像素	+0.30 rad/s，机头右转
B：p_i[0]=240（偏左80像素）	ex = −80像素	−0.30 rad/s，机头左转

- **对照记录：**目标位置、最终发送目标和载具状态使用可对应的时间。
- **完整反馈：**控制方向随观测改变，运动后获得新观测，并最终完成降落。

观测、最终发送目标与后续响应按时间对应，才能解释视觉反馈。

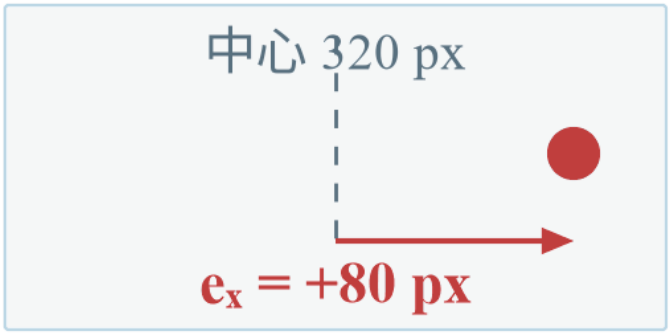
2.6.2 实践与迁移：教材题10讲评

偏航怎样改变下一帧图像

A：目标偏右，机头右转

观测 → 误差 → 指令 → 转向 → 新观测

相机画面



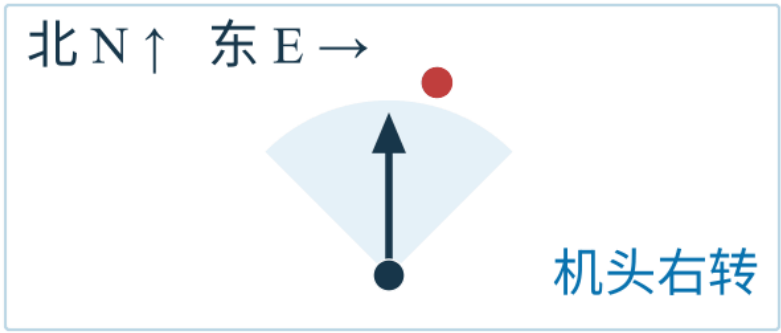
偏航指令

$$yawrate = K \times e_x$$

$$0.00375 \times (+80)$$

+0.30 rad/s

俯视：正偏航顺时针



横向误差幅值随时间减小



图：比例偏航使目标逐帧靠近画面中心（几何示意）

一次看准还不够，持续观测与修正才能把目标保持在视野中。

课堂讨论

一环正常时，下一步先查哪一环

已知现象	目前能判断	优先检查
三维画面正常 没有有效载具状态	场景正在显示	状态接收、连接对象 及自驾仪初始化
遥测正常 目标指令未被接受	状态链路有数据	模式、目标格式 坐标与持续发送
图像更新 控制量不随观测变	相机正在输出	检测结果、误差 与最终发送目标

回答：选择一个情境，给出首查点、理由和要补充的记录。

课后实践

课后选择与提交内容

- **全体作答：**十题概念与方案完整回答。
- **动手基础：**题7、题8；其余按条件选做。
- **提交记录：**任务、版本、配置和改动；截图与原始日志；判据、结果和异常处置。

入口：例程广场 → 第2章 → 实验说明



图：第二章例程入口

课后实践

题3—6对应的例程

题号	例程或工具（实际名称）	用途
题3	1.BasicExps\e18_PX4VersionSwitch、e9_Build-Firmware 1.BasicExps\e7_Code-Generation、e10_Firmware-Upload	切换命令与编译 代码生成与烧录
题4	桌面RflyTools\WslSwitch2 1.RflySimIntro\2.AdvExps\e13.WinWSL2-GPU	切换WSL版本 WSL2与GPU环境
题5	0.ApiExps\e10_Mavros\3.Mavros2Offboard	ROS 2外部控制
题6	0.ApiExps\e11_docker	Docker镜像构建

目录在[安装目录]\RflySimAPIs\下；1.BasicExps、0.ApiExps前省略2.RflySimUsage\。

题3烧录需6C；题4镜像网络需Win11 22H2+；题5需ROS 2，题6需WSL2。

课后实践

题7—10对应的例程

题号	例程或工具（实际名称）	用途
题7	桌面RflyTools\SITLRun	软件在环首跑
题9	桌面RflyTools\HITLRun	硬件在环仿真
题8	1.BasicExps\e13_Log-Get	记录真值CSV
	1.BasicExps\e4_Log-Reads-Python38Env	Python读取ULog
题10	1.BasicExps\e19.VisCreateUsage	视觉传感器配置
	1.RflySimIntro\1.BasicExps\e7_RunPythonProject	红球视觉跟踪

目录在[安装目录]\RflySimAPIs\下；以1.BasicExps开头的省略了2.RflySimUsage\。

题9需要Pixhawk板卡，其余三题用单机软件在环即可完成。

本章小结

第二章小结：平台、接口与验证

- **工具职责：**运行环境承载程序；
运动模型、自驾仪与三维场景分别计算状态、控制和观测。
- **配置选择：**按任务选模式、开发工具与目标硬件，让模型、代码和接口匹配。
- **验证方法：**固定比较条件，观察输入、目标与响应，用原始数据解释修改效果。
- **易错点：**TestWSL只查WSL，首跑以起降记录为准；
实际模式以回读mode为准；--load只载入本机镜像库。
- **后续学习：**第三章学习三维环境建模与传感器生成，把场景和观测接入闭环。

平台配置服务具体任务，数据闭环与可重复结果支撑后续开发。