四足机器人 RL 路线

☑ Week 1：向量 / 矩阵 & Python / Eigen 入门
理论：线性代数基础
- 复习向量、矩阵加减乘、转置
- 理解点乘、叉乘的几何意义
- 理解 L2 范数、Frobenius 范数
- 看完一节线性代数相关资料并做笔记（纸/Markdown）
Python：数值运算练习
- 用 Numpy 实现：矩阵乘法、转置、求逆
- 用 np.linalg.norm 计算向量与矩阵范数
- 用 np.linalg.cond 计算矩阵条件数，并记录几个例子
C++ / Eigen：基础使用
- 配好一个简单的 CMake + Eigen 工程
- 写 demo：定义 Matrix3d、Vector3d，做一次矩阵乘法
- 输出矩阵/向量到控制台，确认编译运行 OK

☑ Week 2：SVD / 最小二乘 / 伪逆
理论：分解 & LS
- 了解 EVD / SVD 的用途（稳定求解 / 病态问题）
- 推一遍最小二乘正规方程（可以看资料+抄一遍）
- 理解伪逆的概念与作用场景（欠定/超定）
Python：拟合小实验
- 构造带噪声的一维数据（y = ax + b + noise）
- 用手算 LS：(A^T A)^-1 A^T b
- 用 np.linalg.lstsq 求解并对比
- 用 SVD np.linalg.svd 求伪逆解一次
C++ / Eigen：LS demo
- 用 Eigen 实现一维 LS 拟合
- 封装成一个 LeastSquaresFitter 小类
- 在 main 里调用并打印拟合参数

☑ Week 3：FK / IK / Jacobian（Python 版）
理论：单腿运动学
- 画出简化单腿（2/3 关节）示意图
- 写出 FK 公式（q → foot pos）
- 理解 IK 几何解 or 数值解
- 理解 Jacobian 定义：v = J dq
Python：单腿模型
- 用 Python 写 FK 函数：fk(q) -> x
- 写简化 IK：给足端位置求关节角（可用数值方法）
- 写 Jacobian 函数：J(q)
- 用数值差分验证：J dq ≈ (x(q+dq)-x(q))/dt
小可视化（可选）：
- 用 matplotlib 简单画出腿在平面位置的若干姿态

☑ Week 4：梯度 + 自动求导 + 数值微分
理论：微积分与优化基础
- 复习导数 / 偏导 / 梯度 / 链式法则
- 理解梯度下降思想（沿负梯度方向走）
- 知道 Hessian 是“二阶导矩阵”（只需概念）
Python：梯度 & PyTorch
- 用 Numpy 对 $f(x,y)=x^2 + y^2$ 做梯度下降（手写更新）
- 用 PyTorch 定义一个 MLP 拟合简单函数（如 sin(x)）
- 打印某次反向传播后的 grad，看看数值
C++：数值求导小工具
- 写一个函数：double numerical_grad(std::function<double(double)>, double x)
- 用差分 (f(x+h)-f(x-h))/(2h) 实现
- 测试对 $f(x)=x^2$, sin(x) 等函数的数值梯度

☑ Week 5：动力学直觉 + 简单系统仿真
理论：刚体动力学基础
- 理解质心、转动惯量
- 理解“力 → 加速度”、“力矩 → 角加速度”
- 看一眼 $M(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) = \tau$ 结构，理解每项物理含义
Python：倒立摆/摆杆仿真
- 写一个单摆或倒立摆的动力学方程
- 用 RK4 数值积分模拟一小段时间
- 画出角度/角速度随时间变化曲线
C++：动力学库调用（可选）
- 如果环境有 pinocchio/RBDL，调用一次 inverse dynamics
- 打印某个姿态下的 M、C、g，建立直觉

☑ Week 6：IMU / Encoder / 足端接触（仿真思路）
理论：传感器基础
- IMU：测量角速度、加速度、姿态的基本概念
- Encoder：位置/速度获取方式
- 足端接触：力阈值 + 速度判据
Python：接触检测模拟
- 生成一段伪 foot pos/vel/force 时间序列
- 写一个简单接触检测函数：is_contact(pos_z, vel_z, force)
- 画 contact=true/false 随时间变化
C++：接触估计类
- 写 class FootContactEstimator：
  - 输入：pos_z, vel_z, force
  - 输出：bool contact
- 在 main 中读几组测试数据，打印结果

☑ Week 7：EKF 概念 + 一维示例
理论：EKF 高层结构
- 理解“预测 + 更新”两步
- 知道状态、控制、观测的含义
- 不追矩阵推导，只理解框架
Python：1D EKF demo
- 状态：位置、速度
- 动力学：匀速运动/匀加速模型
- 观测：位置 + 噪声
- 实现简化版 EKF，画真实/观测/估计对比
C++：EKF 框架类
- 写 class SimpleEKF：
  - 成员：x, P
  - 函数：predict(), update(z)
- 用伪数据跑一段，打印估计结果（先不对复杂系统用）

☑ Week 8：整理机器人基础 C++ 工程
本周目标：把前面 C++ 相关代码整理成可复用工具库
任务清单：
- 建一个 quadruped_utils 工程（CMake 管理）
- 整理类：
  - LegKinematics
  - FootContactEstimator
  - SimpleEKF（即使还未完整）
- 写简单 demo 程序调用这些类，统一风格和命名
- 写一份 README，记录这些工具类的用途

☑ Week 9：RL 基础 + CartPole
理论：RL 基本概念
- MDP：S, A, R, P, γ 的含义
- Value / Q / Policy 的区别
- Actor-Critic 的直觉
Python：CartPole 小实验
- 用 Gym + Python 实现 Q-learning 或 DQN 或 A2C
- 跑通训练，让 CartPole 站稳
- 画 reward 随训练步数的曲线
C++：环境抽象（可选）
- 写一个抽象类 Env：reset(), step(action)
- 实现一个最简单的 1D 环境，练习接口设计

☑ Week 10–11：PPO 实战（连续控制）
这两周任务比较重，可以按自己节奏分散到 10/11 两周完成。
理论：PPO & Advantage
- 看 PPO 公式：clip loss / value loss / entropy

- 理解 advantage（GAE 只需直觉）
Python：PPO 实现
- 用 PyTorch 实现 PPO 框架：
    - policy/value 网络
    - rollout 收集
    - 计算 advantage
    - 更新参数
- 选一个连续动作环境（如 Pendulum）训练成功
- 调试学习率、clip 参数、entropy 系数，看收敛情况变化
- 记录一版“自己可用的 PPO 模板”代码
C++：日志工具
- 写一个 C++ 日志类，把数据（时间、状态、命令）存成 CSV
- 用 Python 画出这些 CSV 的曲线

☑ Week 12：把 RL 思路映射到四足任务
理论：四足 RL 任务结构梳理
- 列出当前四足 RL 的：
    - obs 含哪些量（姿态、速度、foot 等）
    - action 是什么（期望关节角/foot force/增量等）
    - reward 分为哪几类（姿态/速度/foot/能耗/平滑等）
Python：阅读现有训练代码 + 注释
- 通读目前在用的四足 PPO 训练脚本
- 在代码里加中文注释，标出：
    - obs 构造
    - reward 各部分
    - action 如何映射到电机/关节
C++：Sim2Real 接口草图
- 写头文件草稿：
    - struct RLObservation
    - struct RLAction
- 设计 ROS/LCM 消息或内部接口大致结构

☑ Week 13–14：Gait / Phase / Duty Factor
理论：腿足步态
- 理解 Trot / Pace / Bound 的相位关系
- 理解 Phase / Duty Factor 概念
- 了解 stance/swing 对稳定性的影响
Python：gait generator
- 写函数：get_leg_phase(gait_type, t)
- 输出 4 条腿在一个周期内 swing/stance 序列
- 用 matplotlib 画四条腿的“条形图步态图”
C++：GaitScheduler
- 实现 class GaitScheduler：
    - 输入：时间 t 或相位 ϕ
    - 输出：每腿 swing/stance 标志

- 在一个伪控制循环中调用，打印结果检查

☑ Week 15：Foot clearance 轨迹
理论：足端轨迹
- 认识典型足端 z(t) 曲线：起步、抬脚、落地
- 明白落地速度过大易引发冲击/抖动
Python：z(t) 设计
- 为 swing 相设计 z(t)：
  - 起始 0，高度 h，周期 T，峰值在 0.5T
  - 可用三次或五次多项式
- 画出 z(t)、dz/dt 曲线，控制落地速度不要太大
C++：FootTrajectoryGenerator
- class FootTrajectoryGenerator：
  - 输入：相位 Φ、步长、步高
  - 输出：foot pos/vel
- 在 main 中跑一个周期，打印/保存用于可视化

☑ Week 16：Foot Placement & 奖励几何直觉
理论：Raibert 落脚点
- 理解公式结构：x_foot = x_com + k * v 类似关系
- 知道如何基于速度误差调整落脚点
Python：落脚点模拟
- 写 compute_foot_target(v_des, v_now) 函数
- 不同 v_des/v_now 下，画出 foot target 分布
- 思考两种情况：
  - 加速：落点应该偏前
  - 减速：落点应该偏后
C++：FootPlacementPlanner
- 实现 C++ 版 FootPlacementPlanner 接口
- 集成到 gait + foot trajectory 结构体中（先在仿真逻辑级别）。

☑ Week 17－18：仿真环境定制（Isaac Gym / Legged-Gym）
理论：仿真系统结构
- 理解 Isaac Gym / Legged-Gym 中：
  - env 数量
  - root_state_tensor/dof_state_tensor
  - 每步仿真流程（step → obs → act → reward）
Python：环境走读 & 改造
- 跑通完整训练一次（可短一点）
- 通读环境定义代码，理解：
  - obs 构造
  - reward 项
  - action → 电机命令映射
- 尝试修改一两个 reward 项，看行为变化

- 尝试修改 obs（如加/减某个量），观察训练表现变化

C++：规划 Sim2Real 对接

- 根据仿真中 obs/action 定义，整理出真机侧应提供/接收的字段（姿态、速度、foot contact 等）

☑ Week 19：Domain Randomization（随机化）

理论：Sim2Real gap & 随机化

- 理解随机化作用：对抗建模误差（质量、摩擦、噪声）
- 分析当前系统中可能的误差来源