目 录

符号和缩略语说明 3
第 1 章 机械狗的运动模型 1
1.1 机械狗运动模型描述 1
1.2 质心运动 (CoM Motion) 问题表述 2
第 2 章 分层次优化计算 4
2.1 运动方程 4
2.2 接触部分运动约束 4
2.3 接触力和扭矩限制 4
2.4 运动跟随 5
2.5 接触力最小化 5
2.6 根据优化结果计算电机扭矩 6
第 3 章 运动优化 7
3.1 足态保持生成 7
3.2 支撑多边形序列生成 7
3.3 运动优化问题描述 8
3.4 规划初始化 8
3.5 质心运动优化 1 8
3.5.1 质心运动优化的成本函数 8
3.6 等式约束 11
3.7 不等式约束 11
3.8 约束松弛 12
3.9 质心运动优化 2 12
3.9.1 成本函数：最小化运动规划结果的加速度 13
3.9.2 等式约束：运动曲线段节点连续性 14
3.9.3 不等约束：基于 ZMP 14
3.10 落脚点优化：倒立摆模型 16
3.10.1 成本函数 16
3.10.2 不等约束：不可达落脚点 17

第 4 章 如何构建 RL 控制训练模型 18
4.1 强化学习控制 18
4.1.1 强化学习的基本概念 18
4.1.2 如何实现基于深度强化学习的控制器在时间环境中的部署和优化 18
4.2 RL 的构建、训练和部署 19
4.2.1 MDP 的制定 19
4.2.2 RL 模型的训练方式 20
4.2.3 训练平台和部署 20
4.3 RL 的实现和部署 2 20
4.3.1 问题描述 22
4.3.2 教师策略训练 22
4.3.3 观测和行动 23
4.3.4 策略构架 24
4.3.5 奖励函数 25
4.3.6 课程 28
4.3.7 学生策略训练 28
4.3.8 高度采样随机化 29
4.3.9 信念状态寄存器 30
4.3.10 部署 31
第 5 章 Xxx 32
参考文献 33
附录 A 补充内容 35

符号和缩略语说明

κ	传输系数 (Transmission Coefficient)
ν_i	虚频 (Imaginary Frequency)

第 1 章　机械狗的运动模型

1.1　机械狗运动模型描述

机器人与环境接触的机械系统的运动模型描述方程可以描述如下[1]p2：

$$\boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{u}} + \boldsymbol{h}(\boldsymbol{q},\boldsymbol{u}) = \boldsymbol{S}^T\boldsymbol{\tau} + \boldsymbol{J}_S^T\boldsymbol{\lambda} \tag{1-1}$$

这其中 $\boldsymbol{q}$ 是一个描述机器人主体及各个节点的广义位置矢量：

$$\boldsymbol{q} = \begin{bmatrix} {}_I\boldsymbol{r}_{IB} \\ \boldsymbol{q}_{IB} \\ \boldsymbol{q}_j \end{bmatrix} \in SE(3) \times \mathbb{R}^{n_j} \tag{1-2}$$

它里面 ${}_I\boldsymbol{r}_{IB} \in \mathbb{R}^3$ 是机器人主体相对于惯性系的三维位置矢量；$\boldsymbol{q}_{IB} \in SO(3)$ 是机器人主体相对于惯性系的转动描述，用哈密顿单位四元数表示的；$\boldsymbol{q}_j \in \mathbb{R}^{n_j}$ 是一个储存机器人所有节点角度的 n_j 维矢量。

> **Notes 1.1.1.** 不过我还是不太清楚 $\boldsymbol{q}$ 也就是 $SE(3) \times \mathbb{R}^{n_j}$ 到底是一个什么形状的矩阵？单纯看表达的话他应该是一个 $3+4+n_j$ 维的矢量，包含了整个机器人的所有位置有关的状态信息。它似乎不用直接参与运算，因此可以先放一放，重要的是 $SE(3) \times \mathbb{R}^{n_j}$ 的含义到底是什么需要好好理解一下。

这其中 $\boldsymbol{u}$ 是一个描述机器人主体及各个节点的广义速度矢量：

$$\boldsymbol{u} = \begin{bmatrix} {}_I\boldsymbol{v}_B \\ {}_B\boldsymbol{\omega}_{IB} \\ \dot{\boldsymbol{q}}_j \end{bmatrix} \in \mathbb{R}^{n_u} \tag{1-3}$$

它里面的 ${}_I\boldsymbol{v}_B$ 描述了机器人主体相对于惯性系的速度；${}_B\boldsymbol{\omega}_{IB}$ 描述了机器人主体相对于自身的角速度；$\dot{\boldsymbol{q}}_j$ 描述了机器人的各个关节转动的速度。这其中 $\boldsymbol{M}$ 是一个关于机器人整体的质量矩阵，它是一个 $n_u \times n_u$ 的矩阵：

$$\boldsymbol{M} \in \mathbb{R}^{n_u \times n_u} \tag{1-4}$$

这个质量矩阵的具体数值跟机器人的机械系统的状态（各个节点的位置 $\boldsymbol{q}$）相关，可以通过通用的方式计算出它的表达式。实际计算的时候只需要带入 $\boldsymbol{q}$ 的值，就可以计算出 $\boldsymbol{M}$ 矩阵的各个元素具体数值。这其中 $\boldsymbol{h}$ 是一个跟机器人的机械位置

和速度都有关的量，包含了机械系统产生的克里奥利力、离心力和重力的作用，它是一个 n_u 维的矢量：

$$\boldsymbol{h} \in \mathbb{R}^{n_u} \tag{1-5}$$

Notes 1.1.2. 这个 $\boldsymbol{h}$ 的具体计算方式我现在还不是很清楚。

这其中 $\boldsymbol{S}$ 是一个选择矩阵，可以用来选择整个公式中哪些自由度被激活，它是一个 $n_\tau \times n_u$ 的矩阵：

$$\boldsymbol{S} = \begin{bmatrix} 0_{n_\tau \times (n_u - n_\tau)} & \mathbb{I}_{n_\tau \times n_\tau} \end{bmatrix} \in \mathbb{R}^{n_\tau \times n_u} \tag{1-6}$$

它其中包含的参数 n_τ 表示被激活的自由度数量，如果机器人的所有自由度都被激活，则 $n_\tau = n_j$。这其中 $\boldsymbol{\tau}$ 是机器人各个关节的电机提供的扭矩，它是一个 n_j 维的向量：

$$\boldsymbol{\tau} \in \mathbb{R}^{n_j} \tag{1-7}$$

它的节点成员是否产生作用受到 $\boldsymbol{S}^T$ 矩阵的选择。这其中 $\boldsymbol{J}_S$ 是一些列雅可比矩阵的集合：

$$J_S = \begin{bmatrix} J_{C_1}^T & \dots & J_{C_{n_c}}^T \end{bmatrix}^T \in \mathbb{R}^{3n_c \times n_u} \tag{1-8}$$

它是接触点的支撑力 $\boldsymbol{\lambda}$ 向节点力转换的矩雅可比矩阵，包含了 n_c 个雅可比矩阵，n_c 为接触地面的肢体个数。

如果足接触建模为点接触的话，在稳定接触的情况下，每个接触点会产生三个相应的约束条件：

$$\begin{cases} {}_I\boldsymbol{r}_{IC}(t) = const \\ {}_I\dot{\boldsymbol{r}}_{IC} = \boldsymbol{J}_C\boldsymbol{u} = 0 \\ {}_I\ddot{\boldsymbol{r}}_{IC} = \boldsymbol{J}_C\dot{\boldsymbol{u}} + \dot{\boldsymbol{J}}_C\boldsymbol{u} = 0 \end{cases} \tag{1-9}$$

其含义就是在惯性系下看接触点的位置是固定的为定值，并且其速度和加速，也即其一阶导数和二阶导数为零。

1.2　质心运动 (CoM Motion) 问题表述

每一个坐标方向的质心运动规划都被描述成一个系列的五次样条。比如沿着 x 方向的其中第 $i-th$ 曲线可以描述为[2]p7：

$$x(t) = a_{i5}^x t^5 + a_{i4}^x t^4 + a_{i3}^x t^3 + a_{i2}^x t^5 2 + a_{i1}^x t^1 + a_{i0}^x t^0$$

$$
\begin{aligned}
&= \begin{bmatrix} t^5 & t^4 & t^3 & t^2 & t^1 & t^0 \end{bmatrix} \\
&\quad \cdot \begin{bmatrix} a_{i5} & a_{i4} & a_{i3} & a_{i2} & a_{i1} & a_{i0} \end{bmatrix} \\
&= \boldsymbol{\eta}^T(t)\boldsymbol{\alpha}_i^x
\end{aligned} \tag{1-10}
$$

这其中 $t \in [t, t+\Delta t_i]$ 是第 i 个曲线段前所有 $(i-1)$ 个曲线持续时间长度的总和，Δt_i 是第 i 个曲线持续的时长。基于 (2) 的关于位置的表述，可以很容易地将关于速度和加速度的表述写出来：

$$\dot{x}(t) = \dot{\boldsymbol{\eta}}^T(t)\boldsymbol{\alpha}_i^x \tag{1-11}$$

$$\ddot{x}(t) = \ddot{\boldsymbol{\eta}}^T(t)\boldsymbol{\alpha}_i^x \tag{1-12}$$

这其中：

$$\dot{\boldsymbol{\eta}}^T(t) = \begin{bmatrix} 5t^4 & 4t^3 & 3t^2 & 2t^1 & 1 & 0 \end{bmatrix}^T \tag{1-13}$$

$$\ddot{\boldsymbol{\eta}}^T(t) = \begin{bmatrix} 20t^3 & 12t^2 & 6t^t & 2 & 0 & 0 \end{bmatrix}^T \tag{1-14}$$

对于质心在 y, z 方向上的描述是一样的。每一条质心曲线都由 x, y, z 三部分分量 $\boldsymbol{\alpha}_i = \begin{bmatrix} \boldsymbol{\alpha}_i^{x^T} & \boldsymbol{\alpha}_i^{y^T} & \boldsymbol{\alpha}_i^{z^T} \end{bmatrix}^T$ 组成，可以将 n_s 条质心曲线的 $3n_s$ 条曲线参数写到一起，优化的参数矢量可以写成：$\boldsymbol{\alpha} = \begin{bmatrix} \boldsymbol{\alpha}_0^T & \dots & \boldsymbol{\alpha}_i^T & \dots & \boldsymbol{\alpha}_{n_s}^T \end{bmatrix}^T$。这样一来，质心的位置可以表示为：

$$\boldsymbol{p}_{CoM}(t) = \boldsymbol{T}(t)\boldsymbol{\alpha}_i \in \mathbb{R}^3, \quad \boldsymbol{T}(t) = \begin{bmatrix} \boldsymbol{\eta}^T(t) & 0 & 0 \\ 0 & \boldsymbol{\eta}^T(t) & 0 \\ 0 & 0 & \boldsymbol{\eta}^T(t) \end{bmatrix} \tag{1-15}$$

同样质心的速度和位置也可以得到了：

$$\dot{\boldsymbol{p}}_{CoM}(t) = \dot{\boldsymbol{T}}(t)\boldsymbol{\alpha}_i \in \mathbb{R}^3 \tag{1-16}$$

$$\ddot{\boldsymbol{p}}_{CoM}(t) = \ddot{\boldsymbol{T}}(t)\boldsymbol{\alpha}_i \in \mathbb{R}^3 \tag{1-17}$$

第 2 章　分层次优化计算

2.1　运动方程

$\boldsymbol{\xi}_d$ 下面的控制方法采用接触力控制

通过利用选择矩阵 $\boldsymbol{S}^T$ 诱导的分解，我们可以将运动和接触力限制在浮动基系统动力学描述的流形上，有如下式[1]p2：

$$\begin{bmatrix}\boldsymbol{M}_{fb} & -\boldsymbol{J}_{s_{fb}}^T\end{bmatrix}\boldsymbol{\xi}_d = -\boldsymbol{h}_{fb} \tag{2-1}$$

Notes 2.1.1. XXX_{fb}: 下标的意思是浮动的主体-'floating base'

（1）$\boldsymbol{\xi}_d$ 是一个一个 $n_u + n_c$ 行 1 列的向量：$\boldsymbol{\xi}_d = \begin{bmatrix}\boldsymbol{u}_d^T & \boldsymbol{\lambda}_d^T\end{bmatrix} \in \mathbb{R}^{n_u+n_c}$，其中:

① $\boldsymbol{u}_d^T$ 是目标节点的加速度；

② $\boldsymbol{\lambda}_d^T$ 是目标接触力；

（2）$\boldsymbol{M}_{fb}$ 是复合型惯性矩阵的前六行；啥是复合惯性矩阵？

（3）$\boldsymbol{J}_{s_{fb}}^T$ 是雅克比阵的前六行，它将接触力转换到节点的扭矩；

（4）$\boldsymbol{h}_{fb}$ 是非线性项的前六行，包括克里奥利力、离心力和重力项；

2.2　接触部分运动约束

控制器找到的解决方案不应违反 (2) 中定义的接触约束。因此，我们通过设置在接触点施加空加速度:

$$\begin{bmatrix}\boldsymbol{J}_s & 0_{3n_c\times 3n_c}\end{bmatrix}\boldsymbol{\xi}_d = -\dot{\boldsymbol{J}}_s\boldsymbol{u} \tag{2-2}$$

哦 它将 $\boldsymbol{\xi}_d$ 带进去计算后就得到了：$\boldsymbol{J}_s\boldsymbol{\xi}_d = -\dot{\boldsymbol{J}}_s\boldsymbol{u}$ 其实就是公式 (2) 的第二个式子 $\boldsymbol{J}_s\boldsymbol{\xi}_d + \dot{\boldsymbol{J}}_s\boldsymbol{u} = 0$。

2.3　接触力和扭矩限制

$$({}_I\boldsymbol{h} - {}_I\boldsymbol{n}_\mu)^T {}_I\boldsymbol{\lambda}_k \leqslant 0 \tag{2-3}$$

$$-({}_I\boldsymbol{h} + {}_I\boldsymbol{n}_\mu)^T {}_I\boldsymbol{\lambda}_k \leqslant 0 \tag{2-4}$$

$$({}_I\boldsymbol{l} - {}_I\boldsymbol{n}_\mu)^T {}_I\boldsymbol{\lambda}_k \leqslant 0 \tag{2-5}$$

$$-({}_I\boldsymbol{l} + {}_I\boldsymbol{n}_\mu)^T {}_I\boldsymbol{\lambda}_k \leqslant 0 \tag{2-6}$$

${}_In$ 是接触面的法向量；μ 是摩擦系数；有了这两个再乘上受力 ${}_I\lambda$，就可以得出最大静摩擦力 ${}_I\boldsymbol{n}_\mu^T\lambda$。实际在接触点平行于地面方向的两个分力 ${}_I\boldsymbol{h}$，${}_I\boldsymbol{l}$ 在正反方向上都不应该比这个值大，否则就会发生滑动。这就是公式 5 约束的由来。

$$\boldsymbol{\tau}_{min} - \boldsymbol{h}_j \leqslant \begin{bmatrix}\boldsymbol{M}_j & -\boldsymbol{J}_{s_j}^T\end{bmatrix} \leqslant \boldsymbol{\tau}_{max} - \boldsymbol{h}_j \tag{2-7}$$

这里的 $\boldsymbol{h}_j$ 应该就是科里奥利力那一堆东西。而 $\begin{bmatrix}\boldsymbol{M}_j & -\boldsymbol{J}_{s_j}^T\end{bmatrix}$ 就是加速度力 $\boldsymbol{M}_j\dot{\boldsymbol{u}}_d^T$ 和传递力 $-\boldsymbol{J}_{s_j}^T\boldsymbol{\lambda}_d^T$ 两项的和。它们计算的结果就是电机提供的扭矩力 $\boldsymbol{\tau}$ 克服完 $\boldsymbol{h}(\boldsymbol{q},\boldsymbol{u})$ 剩下的力。

2.4　运动跟随

为了能跟随浮动主体和摆动腿的目标运动。我们通过实现具有前馈参考加速度和运动相关状态反馈状态的操作空间控制器来约束关节加速度。对于主体的线性运动：

$$\begin{bmatrix}{}_C\boldsymbol{J}_P & 0\end{bmatrix}\boldsymbol{\xi}_d = {}_C\ddot{\boldsymbol{r}}_{IB}^d + \boldsymbol{k}_D^{pos}({}_C\dot{\boldsymbol{r}}_{IB}^d - {}_C\boldsymbol{v}_B) + \boldsymbol{k}_P^{pos}({}_C\boldsymbol{r}_{IB}^d - {}_C\boldsymbol{r}_B) \tag{2-8}$$

对于主体的角度运动：

$$\begin{bmatrix}{}_C\boldsymbol{J}_R & 0\end{bmatrix}\boldsymbol{\xi}_d = -\boldsymbol{k}_D^{ang}{}_C\boldsymbol{\omega}_B + \boldsymbol{k}_P^{ang}(\boldsymbol{q}_{CB}^d \Xi \boldsymbol{q}_{CB}) \tag{2-9}$$

* 雅可比矩阵 ${}_C\boldsymbol{J}_P$ 和 ${}_C\boldsymbol{J}_R$ 是与‘控制坐标系 C（这是一个与地形局部估计和机器人航向方向对齐的帧）‘中表达的基相关的平移和旋转雅可比矩阵。

* $\boldsymbol{\Xi}$ 这个算子产生欧拉向量，表示期望姿态 q_{CB}^d 和估计姿态 q_{CB} 之间的相对方向。

* 这里面 $\boldsymbol{k}_P^{pos}, \boldsymbol{k}_D^{pos}, \boldsymbol{k}_P^{ang}, \boldsymbol{k}_D^{ang}$ 是用来控制增益的对角正定矩阵。

* 参考的运动 ${}_C\boldsymbol{r}_{IB}$ 和它的导数是运动规划的结果。

2.5　接触力最小化

可以通过下面的方法将接触力设置为最小值：

$$\begin{bmatrix}0_{3n_c\times n_u} & \mathbb{I}_{3n_c\times 3n_c}\end{bmatrix}\boldsymbol{\xi}_d = 0 \tag{2-10}$$

2.6　根据优化结果计算电机扭矩

如果给定了一个优化的节点运动和接触力，$\boldsymbol{\xi}_d = \begin{bmatrix} \dot{\boldsymbol{u}}_d^T & \boldsymbol{\lambda}_d^T \end{bmatrix}^T$ 我们可以用以下公式计算各个电机的扭矩：

$$\boldsymbol{\tau}_d = \begin{bmatrix} \boldsymbol{M}_j & -\boldsymbol{J}_{s_j}^T \end{bmatrix} \boldsymbol{\xi}_d + \boldsymbol{h}_j$$

其中 $\boldsymbol{M}_j$, $-\boldsymbol{J}_{s_j}^T$, $\boldsymbol{h}_j$ 在公式 (6) 已中定义。

Notes 2.6.1. 因此，所有规划的目的就是给出节点的运动加速度 $\dot{\boldsymbol{u}}_d^T$ 和接触力 $\boldsymbol{\lambda}_d^T$。

第 3 章　运动优化

这部分主要参考文献[1]p3-6。运动框架可以从外部源 (即操作员设备或高级导航规划器) 接收高级速度命令。为了驱动运动到参考速度，为所有腿 (3.1节) 生成立足点，以获得躯干的期望平均速度。此信息以及脚落模式（例如图 4）用于生成一系列支持多边形（3.2节），这些多边形被发送到运动计划优化器（3.3节）。这反过来又将为全身质心的 x, y 坐标产生位置、速度和加速度曲线。

整个计划是在位于惯性框架原点的规划系 P 中计算的，并遵循机器人躯干的偏航旋转。因此，它总是与机器人的航向方向对齐。这允许我们在描述问题时将 x, y 坐标规划的贡献直接对应于对运动的航向 x 和横向 y 的约束。

涉及到的一些坐标系定义：

- I: 惯性系（inertial frame）
- B: 躯体系（free-floating base）
- C: 控制系（control frame）它与从立足点从地形中导出的平面平行，x 轴方向与机器人躯体系重合。
- P: 规划系（plan frame）它的原点和惯性系重合，x 轴方向和机器人躯体系重合。

3.1　足态保持生成

外部高级速度命令用于通过调整参考立足点以特定方向和速度驱动运动，这些立足点是针对每个新控制回路的每条腿计算的。根据腿的接触状态，这些是通过两种不同的方式计算的：当腿接触时，命令速度将被投影到计算的立足点位置，以便平均躯干速度与所需的运动速度相匹配；当腿摆动时，参考立足点以相同的方式计算，但添加了一个速度反馈项，该项用来在机器人受到会导致速度控制误差的外部推力时稳定机器人。相关细节参考文献[3]。（这块儿随后要看看，补充道下面）

3.2　支撑多边形序列生成

我们在相域中（phase domain）为每条腿定义升降事件 ϕ_{lo} 和触下事件 ϕ_{td}。这也定义了所有腿的接触时间表。给定所有腿的触下和升降事件，以及当前脚位置

和所需立足点的集合，我们可以计算一系列支持多边形 (定义为顶点元组和以秒为单位的时间持续时间) 用于运动规划器。

当一个新的运动计划可用时，我们都会执行这样的操作。这样新的解决方案就可以适应接触计划的变化、参考立足点的变化以及高级操作员速度的变化。

为了计算每个多边形，我们从当前阶段 ϕ 开始，并存储接触的脚的 $x-y$ 坐标。我们搜索 $k=1$ 的下一阶段事件 ϕ_k 以获得第一个多边形 $t_0 = t_s(\phi_k - \phi)$ 的持续时间，t_s 是以秒为单位的步幅持续时间。这样，我们就通过顶点及其持续时间在几秒钟内完全定义了一个支持多边形。我们通过将 ϕ 更新为 ϕ_k 并搜索下一阶段相位事件来不断迭代。由于接触计划是周期性的，我们重复这些步骤，直到相位事件 $\phi_0 + 1$，对应于起始相位 ϕ_0。

3.3　运动优化问题描述

等待添加…

$$\underset{\boldsymbol{\xi}}{min.} \quad \frac{1}{2}\boldsymbol{\xi}^T\boldsymbol{Q}\xi + \boldsymbol{c}^T\boldsymbol{\xi} \tag{3-1}$$

$$s.t. \quad \boldsymbol{A}\xi = \boldsymbol{b}, \quad \boldsymbol{D}\xi \leqslant \boldsymbol{f}. \tag{16}$$

3.4　规划初始化

运动计划使用扩展状态（位置、速度和加速度）进行初始化，该扩展状态用作样条序列初始状态的硬约束。这是通过使用一个 *alpha* 滤波器来融合前一次目标位置 ${}_C\boldsymbol{r}_{IB}^{des}$ 和当前测量 ${}_C\boldsymbol{r}_{IB}$ 到的位置实现的：

$${}_C\boldsymbol{r}_{IB}^{ref} = \alpha_C\boldsymbol{r}_{IB}^{des} + (1-\alpha)_C\boldsymbol{r}_{IB} \tag{3-2}$$

其中 $\alpha = 0.5e^{-\boldsymbol{\lambda}t_c}$，其中 $\boldsymbol{\lambda} \in \mathbb{R}$ 是一个用来调节前一次目标位置随其已完成时间长度 t_c 的权重。类似的过滤器也用于设置初始速度，而加速度使用最后可用的参考值进行初始化。

3.5　质心运动优化 1

3.5.1　质心运动优化的成本函数

在我们的问题描述中，在公式3-1中出现的 Hessian 矩阵 $\boldsymbol{Q} \in \mathbb{R}^{12n\times 12n}$ 和线性项 $\boldsymbol{c} \in \mathbb{R}^{12n}$ 都有若干组成部分。我们通过下面式子来惩罚实际位置 $\boldsymbol{p}(t) = \boldsymbol{T}(t)\boldsymbol{\alpha}$

与 $\boldsymbol{p}_r$ 的偏差:

$$\begin{cases} \frac{1}{2}\|\boldsymbol{T\alpha}-\boldsymbol{p}_r\|_{\boldsymbol{W}}^2 \\ =\frac{1}{2}(\boldsymbol{T\alpha}-\boldsymbol{p}_r)^T\boldsymbol{W}(\boldsymbol{T\alpha}-\boldsymbol{p}_r) \\ =\frac{1}{2}\boldsymbol{\alpha}^T\boldsymbol{T}^T\boldsymbol{WT\alpha}-\boldsymbol{p}_r^T\boldsymbol{WT\alpha}+\frac{1}{2}\boldsymbol{p}_r^T\boldsymbol{Wp}_r \end{cases} \tag{3-3}$$

最小化公式3-3中的最小化平方范数相当于用公式3-1的形式求解 QP 问题：

$$\boldsymbol{Q}=\boldsymbol{T}^T\boldsymbol{WT} \quad \boldsymbol{c}=-\boldsymbol{T}^T\boldsymbol{W}^T\boldsymbol{p}_r \tag{3-4}$$

为了惩罚速度参考的偏差，只需在公式3-3中使用 $\dot{\boldsymbol{T}}$ 和 $\dot{\boldsymbol{p}}_r$。可以对加速度偏差进行类似的推理。

3.5.1.1　加速度最小化

如文献[4]p 所示，我们可以通过写作来最小化加速度：

$$\boldsymbol{Q}_k^{acc}=\begin{bmatrix} (400/7)\Delta t_k^7 & 40\Delta t_k^6 & 24\Delta t_k^5 & 10\Delta t_k^4 & \\ 40\Delta t_k^6 & 28.8\Delta t_k^5 & 18\Delta t_k^4 & 8\Delta t_k^3 & \\ 24\Delta t_k^5 & 18\Delta t_k^4 & 12\Delta t_k^3 & 6\Delta t_k^2 & 0_{4\times2} \\ 10\Delta t_k^4 & 8\Delta t_k^3 & 6\Delta t_k^2 & 4\Delta t_k & \\ & 0_{2\times4} & & & 0_{2\times2} \end{bmatrix} \tag{3-5}$$

此时非线性项 $\boldsymbol{c}_k^{acc}=0$。这里请注意，如果这是添加到成本函数的唯一项，则不会有与每个样条的 α_{1k} 和 α_{2k} 系数相关联的成本，从而导致正半定 Hessian 矩阵。当使用诸如 *Active Set one*[5]p 之类的方法时，这是有问题的，该方法要求 Hessian 矩阵是正定的。在这种情况下，可以添加一个正则化项：

$$\boldsymbol{Q}_k^{acc_\rho}=\begin{bmatrix} 0_{4\times4} & 0_{4\times2} \\ 0_{2\times4} & \rho\mathbb{I}——2\times2 \end{bmatrix} \tag{3-6}$$

其中 $\rho=10e^{-8}$，线性项为空。

从文献[5]p 中看得，由于两次求导，α_{1k}, α_{2k} 将都是零。这样整个成本函数就从3-5简化成了3-6，这在对矩阵初始化的时候很有意义因为非零项会被初始化成很小的数 $\rho=10e^{-8}$，而恒零项就可以直接赋值为 0。

3.5.1.2　软最终约束

我们将最终位置 $\boldsymbol{p}_f \in \mathbb{R}^2$ 设置为由计划立足点定义的多边形的中心 $\boldsymbol{p}_f^{ref} \in \mathbb{R}^2$，这是如果机器人支持多边形序列的末尾停止，它将支持机器人的多边形。为了最小化这个范数 $\|\boldsymbol{p}_f-\boldsymbol{p}_f^{ref}\|_{\boldsymbol{W}_f}^2=\|\boldsymbol{A}_f\boldsymbol{s}_f-\boldsymbol{p}_f^{ref}\|_{\boldsymbol{W}_f}^2$，给出如下式子：

$$\boldsymbol{Q}_f = \boldsymbol{A}_f^T \boldsymbol{W}_f \boldsymbol{A} \quad \boldsymbol{c}_f = -\boldsymbol{A}^T \boldsymbol{W}_f \boldsymbol{p}_f^{ref} \tag{3-7}$$

其中 $\boldsymbol{W}_f \in \mathbb{R}^{2\times 2}$ 是一个正权重的对称对角矩阵。为了避免优化器将最终状态放置在远离参考位置的解决方案，我们在位置上添加不等式约束，使其被限制在以参考位置为中心的框中。

3.5.1.3　偏离之前目标的解决方案

由于一旦前一个优化成功，我们就计算一个新的优化，为了避免连续运动计划之间的较大偏差，我们惩罚当前解 $\boldsymbol{\xi}$ 得到的位置、速度和加速度与前一个解 $\boldsymbol{\xi}_{i-1}$ 产生的偏差。

记 t_{pre} 是前一个结果给出后消逝的时间长度，我们通过下式惩罚两者的偏差：

$$\|\boldsymbol{p}(\bar{t}) - \boldsymbol{p}_{i-1}(t_{pre} + \bar{t})\|^2_{\boldsymbol{W}_f}, \bar{t} \in [0, t_f] \tag{3-8}$$

其中 $t_f = \sum_{k=0}^{n-1} f_{fk}$ 是之前所有 n 段曲线持续时间的总和。我们使用样本时间 dt 离散化优化范围 $[0, t_f]$。我们采用类似的成本函数来惩罚上一个解决方案获得的速度和加速度的偏差。

3.5.1.4　轨迹正则化

运动计划的持续更新可能会导致躯干相对于参考立足点的运动漂移。这可能是由于控制误差的累积造成的，这改变了解决方案，使运动变得不可行。为了避免这个问题，我们在与参考正则化器路径的偏差上添加了一个成本。这个正则化的路径可以近似表示成曲线段 $\boldsymbol{\pi}(t), \dot{\boldsymbol{\pi}}(t), \ddot{\boldsymbol{\pi}}(t)$，这个路径正则化器的样条系数是从最小化问题设置中获得的，使得：

- $\boldsymbol{\pi}(t)$ 的初始位置与初始化时支撑多边形的中心位置重合，$\dot{\boldsymbol{\pi}}(t), \ddot{\boldsymbol{\pi}}(t)$ 与初始速度和加速度重合；
- $\boldsymbol{\pi}(t), \dot{\boldsymbol{\pi}}(t), \ddot{\boldsymbol{\pi}}(t)$ 的结束状态由3.5.1.2节中定义。
- 加速度最小化
- 通过在样条结点上的状态设置等式约束来平滑地连接样条线

通过对3.5.1.3节中所做的整个运动进行采样，我们惩罚从路径正则化器产生的运动的偏差。

3.6　等式约束

等式约束主要是从规划曲线段的连续性上出发的。 整体上就是要求曲线段的位置、速度、加速度前后连续。

由于整个运动由一系列样条组成，我们设置运动优化问题以确保它们前后相连接。由于初始状态不能被运动规划器修改，我们设置了一个硬性等式约束，使得第一个样条 s_0 的初始状态与计划初始化中的一个集合重合。这个初始化的硬性等式约束可以写作 $\boldsymbol{p}(0)=\boldsymbol{p}^r,\dot{\boldsymbol{p}}(0)=\dot{\boldsymbol{p}}^r,\ddot{\boldsymbol{p}}(0)=\ddot{\boldsymbol{p}}^r,$，于是有：

$$\begin{bmatrix}\boldsymbol{\eta}(0)^T\\ \dot{\boldsymbol{\eta}}(0)^T\\ \ddot{\boldsymbol{\eta}}(0)^T\end{bmatrix}\boldsymbol{\alpha}_0^x=\begin{bmatrix}\boldsymbol{p}_x^r\\ \dot{\boldsymbol{p}}_x^r\\ \ddot{\boldsymbol{p}}_x^r\end{bmatrix} \tag{3-9}$$

为了保证曲线段 $\boldsymbol{s}_k$ 和 $\boldsymbol{s}_{k+1}$ 是连续的，引入如下约束：

$$\begin{bmatrix}\boldsymbol{\eta}(t_{fk})^T & -\boldsymbol{\eta}(0)^T\\ \dot{\boldsymbol{\eta}}(t_{fk})^T & -\dot{\boldsymbol{\eta}}(0)^T\end{bmatrix}\begin{bmatrix}\boldsymbol{\alpha}_k^x\\ \boldsymbol{\alpha}_{k+1}^x\end{bmatrix} \tag{3-10}$$

其中 t_{fk} 表示以秒为单位的曲线 $\boldsymbol{s}_k$ 的持续时间。当连接两个样条的相等连接位于两个不相交的支持多边形之间时，我们不能再保证平滑度或运动，而是必须允许 ZMP 跳跃，这将导致加速度参考的跳跃。尽管这对控制器有负面影响，但它确实允许优化器在遍历不相交的支持多边形时找到解决方案。我们使用文献[]中描述的的分离轴定理（*SAT*）来检查两个支撑多边形是否相交。如果两者相交，那我们再多添加一条约束条件：

$$\begin{bmatrix}\ddot{\boldsymbol{\eta}}(t_{fk})^T & -\ddot{\boldsymbol{\eta}}(0)^T\end{bmatrix}\begin{bmatrix}\boldsymbol{\alpha}_k^x\\ \boldsymbol{\alpha}_{k+1}^x\end{bmatrix}=0 \tag{3-11}$$

3.7　不等式约束

不等约束的主要内容就是将 ZMP 约束到支撑多边形内部。

如 [11] 和 [19] 所示，通过约束 Zero-Moment Point(ZMP) 位于支撑多边形内，可以保证运动过程中的平衡。从运动计划调用时使用计划脚的位置测量的脚配置开始，我们通过使用 [5] 中定义的接触时间表来计算一系列支持多边形及其持续时间。通过始终假设地面上至少有两只脚，四足机器人的支撑多边形可以是一条线、三角形或四边形。

正如 [16] 和 [19] 中所讨论的，ZMP 的位置是质心运动的函数。ZMP 的 x 坐标

位置定义为：

$$x_{zmp} = x_{com} - \frac{z_{com}\ddot{x}_{com}}{g + \ddot{z}_{com}} \tag{3-12}$$

其中 g 是重力加速度。我们通过计算每个支撑多边形的顶点与中心点的极坐标并比较他们的相位来对其顶点进行顺时针排列。这样一来就可以通过添加下式约束来使得 ZMP 点满足要求：

$$ax_{zmp} + by_{zmp} + c \geqslant 0 \tag{3-13}$$

其中，a, b, c 是经过支撑多边形各个边的直线方程的系数。这条针线的法向量是 $\boldsymbol{n} = [a, b]^T$，其方向定义为指向约束多边形内部。

如3.5.1节所述，我们对运动的最终位置 $\boldsymbol{p}_f$ 施加了额外的不等式约束。这是通过在 $\boldsymbol{p}_f$ 上添加如公式3-12形式的四个约束来获得的。

3.8　约束松弛

约束松弛主要涉及三个方面的处理：

- ZMP 约束初始化的问题；
- 支撑多边形的缩小和逐步扩大来适应实际情况；
- 去除持续时间过短的约束多边形情况；

如果约束太紧，优化器可能会失败。首先，初始 ZMP 可能不位于当前支持多边形中。出于这个原因，我们在运动计划的第一个样本上排除了 ZMP 约束。这样优化器仍然能够找到解决方案，我们的实验表明这适用于真实系统。其次，支撑多边形安全裕度可能太高。为了抵消这一点，每当优化失败时，我们都会以固定量迭代地减少边距，并以较慢的速度将其增加以确保优化的成功。最后，我们检查与每个多边形相关的持续时间 t_{fk}。如果它与用于离散化运动并设置约束的样本时间更小或相当，我们从序列中删除支持多边形。

3.9　质心运动优化 2

机械狗的质心运动天然受到多种条件约束。因此，我们将质心运动规划问题描述为一个非线性优化问题，它的目标是在相等约束 $\boldsymbol{c}(\boldsymbol{\xi})$ 和不等约束 $\boldsymbol{h}(\boldsymbol{\xi})$ 的条件下最小化一个通用的成本函数 $\boldsymbol{f}(\boldsymbol{\xi})$。解决这个问题的数值方法被称作为顺序二次规划算法 ((SQP) algorithm)，这个算法要求计算约束条件和成本函数的雅可比和海森矩阵。这个优化算法会以局部运动的踱步周期 τ 为时间间隔，不断地重新计算新的结果。下面的内容将介绍一些用到的成本函数和约束条件.

3.9.1　成本函数：最小化运动规划结果的加速度

通过计算样条段的二次成本可以得到：

$$\boldsymbol{Q}_k^{acc} = \begin{bmatrix} (400/7)\Delta t_k^7 & 40\Delta t_k^6 & 24\Delta t_k^5 & 10\Delta t_k^4 & \\ 40\Delta t_k^6 & 28.8\Delta t_k^5 & 18\Delta t_k^4 & 8\Delta t_k^3 & \\ 24\Delta t_k^5 & 18\Delta t_k^4 & 12\Delta t_k^3 & 6\Delta t_k^2 & 0_{4\times 2} \\ 10\Delta t_k^4 & 8\Delta t_k^3 & 6\Delta t_k^2 & 4\Delta t_k & \\ & 0_{2\times 4} & & & 0_{2\times 2} \end{bmatrix} \tag{3-14}$$

相应的 $\boldsymbol{c}_k^{acc} = 0$。$\Delta t_k$ 是第 k 个曲线段以秒为单位的持续时间。$\boldsymbol{Q}_k^{acc}$ 是通过平方和积分 CoM 在第 k 样条的持续时间上的加速度得到。$\boldsymbol{Q}_k^{acc}$ 被添加为整个成本函数的 Hessian 的子矩阵。

> **Notes 3.9.1.**（$\boldsymbol{Q}_k^{acc}$ 到底是成本函数还是 Hessian 矩阵？它们之间有什么关系？有必要了解一下 Hessian 矩阵去。）

最终期望的位置计算为参考高级线性速度 v_{ref} 和角速度 ω_{ref} 的 z 分量和优化时间间隔 τ 的函数。在以上参数相对于优化时间间隔 τ 变化很缓慢的情况下，最后的位置可以计算为：

$$\boldsymbol{p}_{final} = \boldsymbol{p}_{init} + \boldsymbol{R}(\tau\hat{\boldsymbol{\omega}}_z)(\tau\boldsymbol{v}_{ref}) \tag{3-15}$$

这里面第一项 $\boldsymbol{p}_{init}$ 是前一时刻的位置，第二项是当前时刻的位移 $\tau\boldsymbol{v}_{ref}$ 乘上一个旋转矩阵 $\boldsymbol{R}(\tau\hat{\boldsymbol{\omega}}_z) = \begin{bmatrix} 0 & 0 & \boldsymbol{\omega}_{ref_z} \end{bmatrix}^T$。路径正则化 $\boldsymbol{\pi}$。$\boldsymbol{\pi}$ 的初始位置计算为优化视界 τ 上平均的足迹中心，假设脚要么接触，要么以恒定的速度移动。最终位置是通过参考高级速度计算的，如参考文献（7）所示。

我们将初始速度设置为参考速度 v_{ref}，最后一个速度与运动计划结束时预测的躯干方向对齐。$\boldsymbol{\pi}$ 的初始和最终加速度设置为零。$\boldsymbol{\pi}$ 的初始和最终高度设置为用户指定的参考值。（这个 $\boldsymbol{\pi}$ 到底是个啥？运动规划的高级计算近似结果，比如用传感器得到的轨迹推算出来的结果？）我们通过添加以下形式的成本 $\lambda_{lin}\epsilon_z + \lambda_{quad}\epsilon_z^2$ 来限制沿 z 轴超调。同时对原有描述扩充以下几个约束：

$$\boldsymbol{p}_{CoM}^z(t) - \boldsymbol{\pi}_z(t) \leqslant \epsilon_z \tag{3-16}$$

$$\boldsymbol{p}_{CoM}^z(t) - \boldsymbol{\pi}_z(t) \geqslant -\epsilon_z \tag{3-17}$$

$$\epsilon_z \geqslant 0 \tag{3-18}$$

这里面 ϵ 是一个松弛变量，它会被添加入优化参数 $\boldsymbol{\xi}$ 中。由于时间依赖性，需要对轨迹进行采样，其中每个采样点引入两个额外的不等式约束。（为什么？）

3.9.2　等式约束：运动曲线段节点连续性

对于 x 方向上的第 k 段和第 $k+1$ 段曲线的连续性要求可以写作：

$$\begin{bmatrix} \boldsymbol{\eta}(t_{fk})^T & -\boldsymbol{\eta}(0)^T \\ \dot{\boldsymbol{\eta}}(t_{fk})^T & -\dot{\boldsymbol{\eta}}(0)^T \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha}_k^x \\ \boldsymbol{\alpha}_{k+1}^x \end{bmatrix} = 0 \tag{3-19}$$

这里面 t_{fk} 代表曲线 $\boldsymbol{s}_k$ 以秒为单位的持续时间长度。同样的方法，可以将 y, z 方向上的连续性要求写出来。节点连续性在涉及到悬空自由落体过程时需要被另外对待：起飞时的质心的动力学可以描述为：$\ddot{\boldsymbol{p}}_{CoM} = \boldsymbol{g}$，其中 $\boldsymbol{g}$ 是重力加速度矢量。将上式积分可以得到：

$$\dot{\boldsymbol{p}}(t) = \boldsymbol{g}t + \dot{\boldsymbol{p}}_{CoM}(0) \tag{3-20}$$

$$\boldsymbol{p}(t) = \frac{1}{2}\boldsymbol{g}t^2 + \dot{\boldsymbol{p}}_{CoM}(t) + \boldsymbol{p}_{CoM}(0)(0) \tag{3-21}$$

着陆时的质心动力学可以描述为：

$$\ddot{\boldsymbol{T}}(0)\boldsymbol{\alpha}_{i+1} = \boldsymbol{g} \tag{3-22}$$

$$\dot{\boldsymbol{T}}(0)\boldsymbol{\alpha}_{i+1} = \boldsymbol{g}t_f + \dot{\boldsymbol{T}}(t_i)\boldsymbol{\alpha}_i \tag{3-23}$$

$$\boldsymbol{T}(0)\boldsymbol{\alpha}_{i+1} = \frac{1}{2}\boldsymbol{g}t_f^2 + [\dot{\boldsymbol{T}}(t_i)t_f + \boldsymbol{T}(t_i)]\boldsymbol{\alpha}_i \tag{3-24}$$

这里面 t_i 表示第 i 条曲线段的持续时间。如果第一个或最后一个支持多边形对应于一个完整的飞行阶段，则可以在初始和最终条件下找到类似的替换。

Notes 3.9.2. 这句话的含义是什么？实践上应该怎样应用？

3.9.3　不等约束：基于 ZMP

Notes 3.9.3. 这部分我还是看不太懂，这家伙已经不用购物车模型来近似计算 ZMP 了，还要再去看看 ZMP 的知识。

3.9.3.1　ZMP 的计算

$$\boldsymbol{p}_{ZMP} = \frac{\boldsymbol{n} \times \boldsymbol{M}_O^{gi}}{\boldsymbol{n}^T \boldsymbol{F}^{gi}} \tag{3-25}$$

这里面 $\boldsymbol{M}_O^{gi}$ 和 $\boldsymbol{F}^{gi}$ 是重力-惯性力螺旋的组成部分，它们的计算方式如下：

$$\boldsymbol{M}_O^{gi} = m \cdot \boldsymbol{p}_{CoM} \times (\boldsymbol{g} - \ddot{\boldsymbol{p}}_{CoM}) - \dot{\boldsymbol{L}} \tag{3-26}$$

$$\boldsymbol{F}^{gi} = m \cdot \boldsymbol{g} - \dot{\boldsymbol{P}} \tag{3-27}$$

这里面 $m, \boldsymbol{P}, \boldsymbol{L}$ 分别是质心的质量、线性动量和角动量。因为我们将不会针对转动及其延伸进行优化，因此下面的计算中角动量近似为零 $\dot{\boldsymbol{L}} = 0$。

3.9.3.2　引入的不等约束

基本要求是 ZMP 点药被约束在机械狗的支撑多边形内部，这给出以下形式的不等约束：

$$\boldsymbol{h}_{ZMP} = \boldsymbol{d}^T \boldsymbol{p}_{ZMP} + r \geqslant 0 \tag{3-28}$$

这其中 $\boldsymbol{d}^T = \begin{bmatrix} p & q & 0 \end{bmatrix}$ 和 r 是描述支撑多边形的边的系数。接下来将 (12) 代入 (14) 后可以得到：

$$\begin{aligned} &\boldsymbol{d}^T \boldsymbol{S}(\boldsymbol{n}) \boldsymbol{M}_O^{gi} + r \boldsymbol{n}^T \boldsymbol{F}^{gi} \\ =&\boldsymbol{d}^T \boldsymbol{S}(\boldsymbol{n})[m \cdot \boldsymbol{p}_{CoM} \times (\boldsymbol{g} - \ddot{\boldsymbol{p}}_{CoM}) - \dot{\boldsymbol{L}}] + r \boldsymbol{n}^T (m \cdot \boldsymbol{g} - \dot{\boldsymbol{P}}) \\ \overset{\dot{\boldsymbol{L}} \approx 0 | \dot{\boldsymbol{P}} = m \ddot{\boldsymbol{p}}_{CoM}}{=}&m \boldsymbol{d}^T \boldsymbol{S}(\boldsymbol{n}) \boldsymbol{p}_{CoM} \times (\boldsymbol{g} - \ddot{\boldsymbol{p}}_{CoM}) + m r \boldsymbol{n}^T (\boldsymbol{g} - \ddot{\boldsymbol{p}}_{CoM}) \geqslant 0 \\ \rightarrow&\boldsymbol{d}^T \boldsymbol{S}(\boldsymbol{n}) \boldsymbol{S}(\boldsymbol{p}_{CoM}) (\boldsymbol{g} - \ddot{\boldsymbol{p}}_{CoM}) + r \boldsymbol{n}^T (\boldsymbol{g} - \ddot{\boldsymbol{p}}_{CoM}) \geqslant 0 \end{aligned} \tag{3-29}$$

这里面定义了一个斜对称矩阵 $\boldsymbol{S}(a)$ 用来实现将矩阵叉乘转化为点乘的效果（这个方法优惠什么标准依据？）。它通过计算一个可以实现以下效果的方程得到：$\boldsymbol{S}(\boldsymbol{a})\boldsymbol{b} = \boldsymbol{a} \times \boldsymbol{b}$。用 (15) 也可以计算出 (14) 相对于质心位置 $\boldsymbol{p}_{CoM}$ 和质心加速度 $\ddot{\boldsymbol{p}}_{CoM}$ 的梯度（为什么要求这个梯度？怎么求得？）：

$$\nabla \boldsymbol{h}_{ZMP} = \begin{bmatrix} \Gamma \cdot (\ddot{\boldsymbol{p}}_{CoM} - \boldsymbol{g}) \\ -\Gamma \cdot \boldsymbol{p}_{CoM} - r \boldsymbol{n} \end{bmatrix} \tag{3-30}$$

这里面定义 $\boldsymbol{\Gamma} = \boldsymbol{S}(\boldsymbol{S}^T(\boldsymbol{n})\boldsymbol{d})$。同样，它的 Hessian 矩阵计算为：

$$\nabla^2 \boldsymbol{h}_{ZMP} = \begin{bmatrix} 0 & \boldsymbol{\Gamma}^T \\ -\boldsymbol{\Gamma}^T & 0 \end{bmatrix} \tag{3-31}$$

这个 Hessian 矩阵是一个反对称矩阵，因此可以从优化问题中去除。（为什么呢？Hessian 矩阵怎么计算的也得去了解一下。）

Notes 3.9.4. 这部分不是很明白。为什么对问题描述添加那样的项就可以软化不等约束？具体的表现项是与软化参数之间的对应什么？

我们软化了初始 n_{ineq} 个样本的不等式约束，其中 n_{ineq} 是用户设置的调整

参数。为了实现这一点，我们在优化参数 $\boldsymbol{\xi}$ 中添加松弛变量 ξ_{ineq}。这样一来，问题的表述会增加 $\lambda_{lin}\xi_{ineq} + \lambda_{quad}\xi_{ineq}^2$，同时添加下面两个不等约束 $\boldsymbol{c}_{ineq} \geqslant -\xi_{ineq}$, $\xi_{ineq} \geqslant 0$，其中 $\boldsymbol{c}_{ineq}$ 是如 (15) 中描述的前 n_{ineq} 个约束条件。从物理的角度来看，松弛相当于一个可变大小的支撑多边形，不能小于标称多边形。

3.9.3.3　分配一个新的规划：搜索算法

下面讨论一下如何将一个新得到的运动规划添加到一个之前已经在执行的运动规划上，避免两者的冲突。为此，我们首先存储求解器优化所需的计算时间 t_c。我们使用它作为初始猜测来搜索最接近当前测量值的运动规划中的位置，以便过渡到新计划是一个平滑的。为此，我们通过写作来解决线性搜索问题：

$$t = arg\,min_{\boldsymbol{W}} ||\boldsymbol{p}(t) - \boldsymbol{p}_{meas}||_2^2 \tag{3-32}$$

这里面 $\boldsymbol{W}$ 是一个正定权重矩阵 $\boldsymbol{p}_{meas}$ 是完成优化动作后的测量得到的质心位置。

3.10　落脚点优化：倒立摆模型

建立如下二次规划问题：

$$\min_{\boldsymbol{\xi}} \quad \frac{1}{2}\boldsymbol{\xi}^T\boldsymbol{Q}\boldsymbol{\xi} + \boldsymbol{c}^T\boldsymbol{\xi} \quad s.t. \quad \boldsymbol{D}\boldsymbol{\xi} \leqslant \boldsymbol{f} \tag{3-33}$$

这里面 $\boldsymbol{\xi} \in \mathbb{R}^{2n_{feet}}$ 是裸足点 $\boldsymbol{p}_{f_i}$ 的 x, y 方向分量，其中 $i = 1, \ldots, n_{feet}$，$n_{feet} = 4$ 是机器所有脚的总数。与 CoM 运动规划器所做的类似，我们并行优化主控制回路。因此，每当新的优化准备好时，我们都会更新立足点计划。

3.10.1　成本函数

相对于默认的战力姿势配置，用户可以自定义一个落脚点位置。这个可以解释为落脚点优化问题的一个正则化项。

Notes 3.10.1.（正则化的概念需要熟悉一下？）

为了跟踪默认落脚点的位置 $\boldsymbol{p}_{ref_i}$ 我们为成本函数 (19) 添加下述表述：

$$\boldsymbol{p}_{ref_i} = \boldsymbol{W}_{def}, \quad \boldsymbol{c}_{def_i} = -\boldsymbol{W}_{def}^T\boldsymbol{p}_{ref_i} \tag{3-34}$$

默认立足点的选择将影响当所有腿与环境接触时足迹的程度。虽然这可能会使小跑步态对干扰更加健壮，但它会在较慢的步行步态中产生更广泛的横向运动。（不知所云？）在实践中，我们在 ANYmal 的实验中可靠地工作的默认立足点位置计算为臀部到地形的垂直投影。

为了跟踪驱动整个运动框架的平均高级速度，我们惩罚与立足点位置的偏差。这些是在优化视界的持续时间内实现恒定速度来计算的。为了避免参考立足点中的跳跃，我们还为当前解决方案和先前计算的解决方案之间的距离设置了成本。

最后，我们为成本函数添加一个稳定项，它是一个倒立摆模型的函数。如参考文献 10 中所述，这个函数用 $\omega(\boldsymbol{v}_ref - \boldsymbol{v}_{hip_k}\sqrt{h/g})$ 计算第 k 个落脚点。这其中，ω 是一个正的权重标量，$\boldsymbol{v}_ref$ 是高级速度参考，$\boldsymbol{b}_{hips}$ 是第 k 个臀部的速度，h 是第 k 个臀部到地面的高度，g 是加速度常量。

Notes 3.10.2. 这引导了去关注倒立摆模型的建模？

3.10.2　不等约束：不可达落脚点

为了避免计算违反腿运动学扩展的立足点，我们通过在每个立足点的可行位置上添加不等式约束来利用 QP 设置。我们通过考虑腿的最大可伸展长度和测量的臀部与地面的高度关系来完成这个约束。将臀部的位置投影到地形平面上记作 $\boldsymbol{h}_0$，我们设置了描述一个多边形的不等式，该多边形具有均匀分布在 $\boldsymbol{h}_0$ 附近的 n_p 个顶点。多边形的各个顶点到臀部投影在地形的点 $\boldsymbol{h}_0$ 的距离计算为：

$$\sqrt{l_{max}^2 - h_i^2} \tag{3-35}$$

Notes 3.10.3. 这就是一个直角三角形，斜边长度为腿的最大伸展长度 l_{max}，高为臀部到地形的投影高度 h_i，计算落脚点的可行范围，也就是底边长度。

第 4 章　如何构建 RL 控制训练模型

在这部分将阐述关于如何使用强化学习来进行运动控制的内容。相比于传统的基于模型的控制方法，该方法通过在模拟环境中训练控制策略，可以更好地适应复杂的非线性系统动力学和环境不确定性，并且能够实时响应用户命令和环境变化。在这里我们主要关注如何使用强化学习来训练机械狗类型的足式和轮式机器人控制器。

4.1　强化学习控制

本部分内容的探究起点是这篇文献的内容：*Control of Wheeled-Legged Quadrupeds Using Deep Reinforcement Learning*[6]

4.1.1　强化学习的基本概念

在强化学习中，控制问题被建模为一个马尔可夫决策过程（Markov Decision Process）。MDP 是一个 RL 中常用的用于随机控制过程建模的数学框架，它定义了一个包含状态空间、动作空间、奖励函数和状态转移函数的元组，描述了一个决策过程的基本组成部分。在 MDP 中，每个时间步骤代理从周围环境中观察到某个状态 $s_t \in \mathcal{S}$，并输出一个动作 $a_t \in \mathcal{A}$，接着环境通过状态转移函数 $p(s_{t+1}|s_t, a_t)$ 演化到新的状态 s_{t+1}，并且根据奖励函数给出相应的奖励 $r_t \in \mathcal{R}: \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ 和对更新后的环境状态的观察结果作为下一周期的 s_t。代理可以根据 θ 参数化的随机策略 $\pi_\theta(a_t|s_t)$ 来采取行动。RL 通过与环境交互来更新参数 θ 以最大化累计折扣奖励（cumulative discounted rewards）$\mathbb{E}[\sum_{t=k}^{\infty} \gamma^t r_t]$，其中 k 是当前时间步长（timestep），γ 是折扣因子（discount factor）。

4.1.2　如何实现基于深度强化学习的控制器在时间环境中的部署和优化

（1）在仿真环境中训练控制器：使用深度强化学习算法在仿真环境中训练控制器，使其能够完成所需的任务。在训练过程中，可以使用特权训练方法来提高训练效率和性能。

（2）部署控制器到实际环境中：将训练好的控制器部署到实际环境中，例如机器人或移动设备。在实际环境中，控制器将接收传感器数据并输出动作命令。

（3）优化控制器：在实际环境中，可以使用在线学习方法来进一步优化控制

器的性能。例如，可以使用模型预测控制方法来对控制器进行在线微调。

需要注意的是，在实际环境中，传感器数据可能会受到噪声和不确定性的影响，因此需要设计鲁棒性强的控制器来应对这些问题。此外，还需要考虑实际环境中的安全性和可靠性问题。

4.2　RL 的构建、训练和部署

4.2.1　MDP 的制定

一个 MDP 是由一个元组定义，它包含状态空间 $\mathcal{S}$、动作空间 $\mathcal{A}$、奖励函数 $r_t(s_t|s_{t+1})$、转移函数 $p(s_{t+1}|s_t,a_t)$。这套定义方式和文献类似[7]p 。这个策略的环境观察部分包括周围地面的高度扫描结果、来自 IMU 的本体姿态信息、来自各个关节点的编码器信息。运动被定义为一个 21 维的向量，包括步态参数偏移、关节点位置偏移和轮子速度控制指令。奖励函数由表4-1定义。

表 4-1　Reward functions. (The velocities are defined in the base frame)

Reward terms	
Linear velocity	$1.5\exp(-3(v^{xy}-\hat{v}^{xy})^2)$
Angular velocity	$1.0\exp(-3(\omega^{z}-\hat{\omega}^{z})^2)$
Base motion penalty	$0.8\exp(-(v^{z})^2)+\exp(-(\omega^{x,y})^2)$
Torque penalty	$-10^{-6}\|\tau\|^2$
Joint speed penalty	$-0.05\sum_{i=0}^{12}\max(\|\dot{\phi}_i\|-\dot{\phi}_{jslim},0)^2$
Action smoothness penalty	$-0.05\|a_{t-2}-2a_{t-1}+a_t\|$
Symbols	
$\dot{\phi}_{jslim}$	Maximum joint speed
τ	Vector of joint torques
f_c	Foot contact state
$\hat{(\cdot)}$	Traget quantity
$(\cdot)_g$	Sub-goal quantity
$\mathbb{1}_{(condition)}(\cdot)$	Indicator function

状态转换遵循刚性的动力学模拟，当机器人主体接触地面或到达关节扭矩和速度限制时，该训练集终止。

4.2.2　RL 模型的训练方式

我们遵循 Lee 等人[8]的特权训练方法。我们首先使用 on-policy RL 算法[9]训练具有无噪声模拟状态 (特权信息) 的教师策略。换句话说，我们用额外的模拟状态附加给 s_t，包括地面摩擦系数、地面反作用力以及每个连杆的接触状态。特权信息使教师策略能够学习地形自适应行为。由于无噪声和准确的模拟状态，教师策略快速收敛到高性能策略。在桌面 PC 上使用 PPO 算法[9]，教师策略训练大约需要 8 小时。然后我们为实际部署训练一个学生策略。学生策略在没有特权信息和观察噪声的情况下模仿老师。通过模仿学习[10]，它学习从嘈杂的真实世界观察序列构建世界的内部表示。以这种方式训练的策略已被证明在具有高干扰和噪声观测的真实世界环境中更具适应性和鲁棒性[8]。

4.2.3　训练平台和部署

训练采用的平台有 Raisim[6,11]仿真器、英伟达的 Isaac[12]仿真器。

Notes 4.2.1. 2023-10-06 18:19:57，感觉强化学习者部署起来也不是件容易得事情呀。后面关于运动控制建模的方式就先不看了。重点关注用强化学习做控制要关注的内容。

4.3　RL 的实现和部署 2[7]p10-12

整个神经网络的训练是在仿真环境中完成的，然后采用 *zero-shot sim-to-real* 的转换部署到实际的额机器人上。整个方法分为三个阶段，如图4-1所示。

（1）首先，使用 RL 训练教师策略，以在随机生成的具有随机干扰的地形上遵循随机目标速度。该策略可以访问特权信息，例如无噪声地形测量、地面摩擦和引入的扰动。

（2）在第二阶段，训练学生策略重现教师策略的动作，而不使用这种特权信息。学生策略构造一个信念状态来使用循环编码器捕获未观察到的信息，并根据该信念状态输出一个动作。在训练期间，我们利用两个损失：行为克隆损失和重建损失。行为克隆损失旨在模仿教师策略。重新构造损失鼓励编码器产生信息丰富的内部表示。

（3）最后，我们将学习到的学生策略转移到物理机器人上，并将其与机载传感器在现实世界中部署。机器人通过整合来自板传感器的深度数据和从构建的高程图中采样高度读数来构建高程图，以形成策略的外部感知输入。这种外部感知输入与本体感觉数据相结合，并提供给神经网络，该网络产生执行器命令。板上

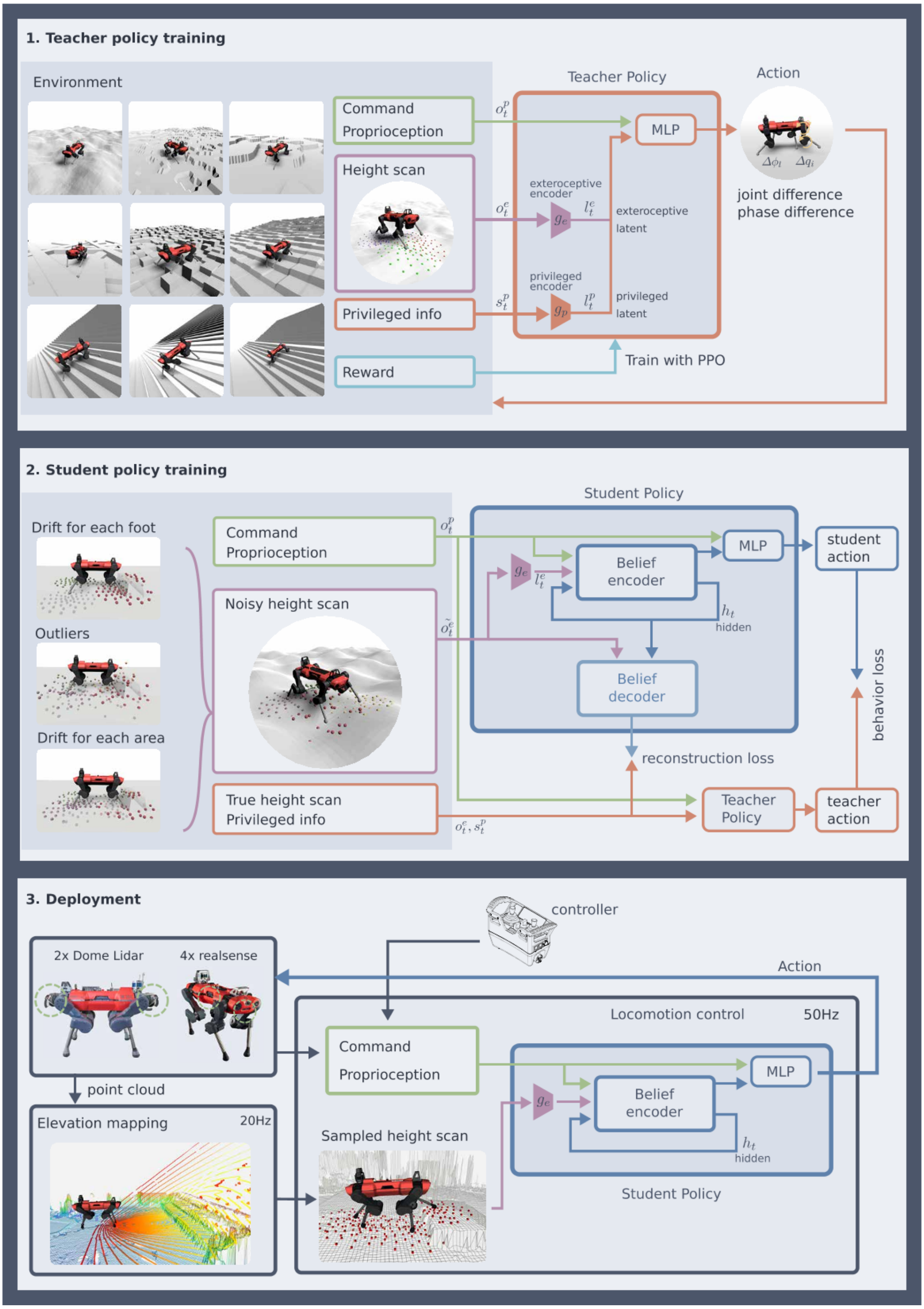

图 4-1　RL implementation process[7]p9 .

传感器和自身构建等高图融合

4.3.1　问题描述

我们在离散时间动力学中制定了我们的控制问题，其中环境完全由时间步 t 的状态 s_t 定义。该策略实施一个动作 a_t 并且从环境中获得一个观测结果 o_t，这个测量结果来自观测模型 $\mathcal{O}(o_t|s_t,a_t)$。接着环境以 $P(s_{t+1}|s_t,a_t)$ 的概率转移到下一个状态 s_{t+1} 并给出一个奖励 r_{t+1}。

当所有状态都可以被观测的实况下，也即 $o_t = s_t$ 时，整个问题可以被看做一个马尔可夫决策过程 *Markov decision process(MDP)*。然而，当存在不可观察性的信息时，例如外力或完整的地形信息，动力学被建模为部分可观察的马尔可夫决策过程 *partially observable Markov decision process(POMDP)*。

RL 的目标是找到一个能够使得未来轨迹的预期折扣奖励最大化的策略 $\boldsymbol{\pi}^*$，以使得：

$$\boldsymbol{\pi}^* = \underset{a}{argmax} E[\sum_{t=0}^{\infty} \gamma^t r_t] \tag{4-1}$$

已经开发了许多 RL 算法来解决完全可观察的 MDP 问题，并且易于用于训练。然而，POMDP 问题的情况更具挑战性，因为状态不能完全观察到。这通常通过从历史的观察结果 $o_0,\cdots,o_t$ 中构建一个信念状态 b_t*belief state* 以尝试构建完全状态的方式来解决。在深度 RL 中，这通常是通过堆叠一系列先前的观察结果[13]或使用可以压缩过去信息的架构来完成的，例如循环神经网络 (RNN)[14]或时间卷积网络[8,15]。

从头开始训练一个天真地处理序列数据的复杂的神经网络策略可能很耗时[8]。因此，我们使用特权学习（45），我们首先训练一个具有特权信息的教师策略，然后通过监督学习将教师策略提炼为学生策略[16]。

（1）训练环境：

（2）地形：

（3）域随机化：

（4）片段终止条件：

4.3.2　教师策略训练

在训练的第一阶段，我们的目标是找到一个可以访问完美、特权信息的最佳参考控制策略，并使 ANYmal 在随机生成的地形上遵循所需的命令速度。命令需求随机生成并构成一个向量 $\boldsymbol{v}_des \in \mathbb{R}^3 = (v_x, v_y, w)$，其中 v_x, v_y 分别表示在机器

人自身坐标系下的纵向速度和横向速度，w 表示自转速度。

我们采用近端策略优化 *proximal policy optimization(PPO)*[9]p 来训练教师策略。教师策略被建模为一个高斯策略，$a_t \sim \mathcal{N}(\pi_\theta(o_t = s_t), \sigma I)$，其中 π_θ 由用 θ 参数化的多层感知器 *multilayer perceptron(MLP)* 实现，σ 表示每个动作之间的方差。

4.3.3　观测和行动

教师策略的观察定义为 $o_t^{teacher} = (o_t^p, o_t^e, s_t^p)$，其中 o_t^p 表示本体感觉观察 *proprioception observation*，s_t^e 表示外部观察 *exteroceptive observation*，s_t^p 表示特权状态 *privileged state*。

- o_t^p 包含本体速度、转动、节点位置和速度历史、动作历史、每条腿的相位；
- o_t^e 是每只脚周围高度样本的向量，包括五种不同的半径；
- s_t^p 包括接触状态、接触力、接触法线、摩擦系数、大腿和小腿接触状态、施加到身体上外部力和力矩、摆动阶段持续时间；

我们的动作空间受到中心模式生成器的启发[8]p。每条腿 $l = 1, 2, 3, 4$ 保存一个相位变量 ϕ_l 并定义了基于相位的标称轨迹。这个标称轨迹是一个脚尖的步进运动，我们使用逆运动学来计算每个关节执行器 $i = 1, \cdots, 12$ 的标称关节目标 $q_i(\phi_l)$。来自策略的动作是相差 $\Delta\phi_l$ 和关节位置目标残差 Δq_i。

更详细相关内容看相关附件的 S5。

观测向量定义如4-2表所示。本体感知包括命令、节点、身体信息、腿部相位信息。中央模式生成器 *central pattern generator(CPG)* 的相位信息包含 $\Delta\phi_l, \cos\phi_l, \sin\phi_l$ 和每条腿 l 的基础频率。对于外部感知我们采用每个脚周围的高度采样来代替局部高程图。圆形采样模式包括每只脚周围的 $\{6, 8, 10, 12, 16\}$ 个点，半径分别为 $\{0.08, 0.16, 0.26, 0.36, 0.48\}$m。

运动被定义为 $\langle \Delta\phi_l, \Delta q_i \rangle$，其中 $\Delta\phi_l$ 和 Δq_i 分别表示每个条腿（$l \in \{legs\}$）的相位偏移和节点目标位置残差（$i \in \{1, \cdots, 12\}$）。我们有一个标称轨迹 $\boldsymbol{p}(\phi) : \mathbb{R} \longrightarrow \mathbb{R}^3$，它将各个 ϕ_l 映射到目标脚位置，随着 ϕ 在 $[0, 2\pi)$ 范围内生成周期性步进运动。从动作中，每条腿 l 的节点目标位置用逆运动学 $IK(\cdot)$ 和基础相位频率 $\Delta\phi_0$ 定义为 $q_{i\in l}^{target} = IK(\boldsymbol{p}(\phi_l + \Delta\phi_l + \Delta\phi_0)) + \Delta q_{i\in l}$。

标称足迹定义如下。如果相位是处于上摆动期间（$0 \leqslant \phi_l \leqslant \pi/2$）则：

$$\boldsymbol{p}_l(\phi_l) = \langle x_l^n, y_l^n, z_l^n + 0.2 \cdot (-2t_l^3 + 3t_l^2) \rangle, where\ t_l = 2/\pi \cdot \phi_l \qquad (4\text{-}2)$$

$\{x, y, z\}_l^n$ 是默认姿态配置处的标称脚位置。三次 Hermite 样条在 $\phi_l = 0$ 处连接 $z = z_l^n$ 在 $\phi_l = \pi/2$ 处连接 $z = z_l^n + 0.2$。

在下摆动期间（$\pi/2 < \phi_l \leqslant \pi$），足高计算如下：

$$\boldsymbol{p}_l(\phi_l) = \langle x_l^n, y_l^n, z_l^n + 0.2 \cdot (2t_l^3 - 3t_l^2 + 1) \rangle, where\ t_l = 2/\pi \cdot \phi_l - 1 \qquad (4\text{-}3)$$

它与前面的函数对称。在驻立阶段（$\pi < \phi_l \leqslant 2\pi$），$\boldsymbol{p}_l(\phi_l) = \langle x_l^n, y_l^n, z_l^n \rangle$。

表 4-2　Observations. Proprioception is used for both teacher and student training. Exteroception is given in the form of height samples. The privileged information is used only for teacher training.

Observation type	Input	Dimensionality
Proprioception	command	3
	body orientation	3
	body velocity	6
	joint position	12
	joint velocity	12
	joint position history (3 time steps)	36
	joint velocity history (2 time steps)	24
	joint target history (2 time steps)	24
	CPG phase information	13
Exteroception	height samples	208
Privileged info.	contact states	4
	contact forces	12
	contact normals	12
	friction coefficients	4
	thigh and shank contact	8
	external forces and torques	6
	airtime	4

4.3.4　策略构架

我们将教师策略 π_θ 建模为一个 MLP。它包括三个 MLP 组成部分：外部感知编码器、特权编码器、主网络，如图4-1所示。

（1）外部感知编码器 g_e 接收来自 o_t^e 的信息，然后输出一个小些的潜在表示

$$l_t^e = g_e(o_t^e)$$

（2）特权编码器 g_p 接收来自特权状态 s_t^p 的信息，然后输出一个潜在表示

$$l_t^{priv} = g_p(s_t^p)$$

（3）

这些编码器将每个输入压缩为更紧凑的表示，并使得学生策略能更方便地重用一些教师策略组件。

更详细相关内容看相关附件的 S6。

策略网络由多层 MLP 组成。用基于 MLP 的编码器 (g_e, g_p)，将高度采样结果首先编码成为一个 $24 \times 4 = 96$ 维的潜在向量；将特权信息编码成为一个 24 维的潜在变量。每个编码器有两层分别是 $\{80, 60\}$ 和 $\{64, 32\}$ 的隐藏单元。高度样本首先分别针对每只脚分别输入到编码器中，然后连接成一个特征向量。然后将这些特征与本体感受观察连接起来，并馈送到另一个具有三个隐藏层 $\{256, 160, 128\}$ 的 MLP。所有 MLP 的激活函数为 LeakyReLU (72)。

我们使用具有外部感受门的 GRU 作为信念编码器（图4-2C）。GRU 由 2 个堆叠的层组成，每个层有 50 个隐藏单元。信念编码器和外部感知门 g_b, g_a 用于计算 $96 + 24 = 120$ 维信念状态 b_t 和 96 维注意力向量 α。每个编码器有两个隐藏层，每个隐藏层有 $\{64, 64\}$ 和 $\{64, 64\}$ 个隐藏单元。过滤后的外部感受信息 $l_t^e \odot \alpha$ 被添加到 $g_b(b_t')$，使用零填充来匹配维度差异。

4.3.5　奖励函数

针对速度控制命令的跟随，我们定义正奖励；针对违反约束的情况，我们定义负奖励。指令跟随奖励定义如下：

$$\boldsymbol{r}_{command} = \begin{cases} 1.0, & if \boldsymbol{v}_{des} \cdot \boldsymbol{v} > |\boldsymbol{v}_{des}| \\ \exp(-(\boldsymbol{v}_{des} \cdot \boldsymbol{v})^2), & otherwise \end{cases} \tag{4-4}$$

其中 $\boldsymbol{v}_{des} \in \mathbb{R}^2$ 是所需的水平速度，$\boldsymbol{v} \in \mathbb{R}^2$ 是身体坐标系下当前身体速度。同样的奖励机制也被应用与转动速度情况。

我们惩罚与期望速度正交的速度分量以及横摇、俯仰和偏航周围的身体速度。此外，我们使用整形奖励进行身体方向、关节扭矩、关节速度、关节加速度和脚滑以及小腿和膝盖碰撞。身体方向奖励用于避免身体的奇怪姿势。联合相关奖励术语用于避免过于激进的运动。脚滑和碰撞奖励术语用于避免它们。我们通过在模拟中查看策略的行为来调整奖励术语。除了遍历性能外，我们还检查了运动的平滑度。

更详细相关内容看相关附件的 S7。

奖励函数定义为：

$$r = 0.75(r_{lv} + r_{av} + r_{lvo}) + r_b + 0.003r_{fc} + 0.1r_{co} + 0.001r_j + 0.08r_{jc} + 0.003r_s + 1.0 \cdot 10^{-6}r_\tau + 0.0 \tag{4-5}$$

各项的具体定义如下：

（1）线速度奖励 *Linear Velocity Reward*(r_{lv})：这项鼓励策略去跟随需要的水平（x, y plane）速度指令：

$$r_{lv} = \begin{cases} \exp(-|\boldsymbol{v}|^2), & if|\boldsymbol{v}_{des}| = 0 \\ 1.0, & elseif\boldsymbol{v}_{des} \cdot \boldsymbol{v} > |\boldsymbol{v}_{des}| \\ \exp(-(\boldsymbol{v}_{des} \cdot \boldsymbol{v} - |\boldsymbol{v}_{des}|)^2), & otherwise \end{cases} \tag{4-6}$$

其中 $\boldsymbol{v}_{des} \in \mathbb{R}^2$ 是目标水平速度，$\boldsymbol{v} \in \mathbb{R}^2$ 是身体坐标系下当前身体的速度。

（2）角速度奖励 *Angular Velocity Reward*(r_{av})：这一项鼓励策略去跟随需要的转动速度指令：

$$r_{av} = \begin{cases} \exp(-w_z^2), & if w_{des} = 0 \\ 1.0, & elseif w_{des} \cdot \boldsymbol{w}_z > w_{des} \\ \exp(-(w_{des} \cdot w_z - |w_{des}|)^2), & otherwise \end{cases} \tag{4-7}$$

（3）线性正交速度奖励 *Linear Orthogonal Velocity Reward*(r_{lvo})：这项惩罚与目标速度指令正交的速度：

$$r_{lvo} = \exp(-3.0|\boldsymbol{v}_o|^2), where\ \boldsymbol{v}_o = \boldsymbol{v} - (\boldsymbol{v}_{des} \cdot \boldsymbol{v})\boldsymbol{v}_{des} \tag{4-8}$$

（4）身体运动奖励 *Body Motion Reward*(r_b)：这项惩罚身体速度中不符合指令要求的部分：

$$r_{bm} = -1.25v_z^2 - 0.4|\omega_x| - 0.4|\omega_y| \tag{4-9}$$

（5）脚感奖励 *Foot Clearance Reward*(r_{fc})：当一条腿处于摆动相位时，比如 $\phi_i \in [0, \pi)$，机器人应该将这条腿上对应的脚抬起到比障碍物高的程度。但是，为了防止机器人做出不必要的抬高间隙，我们通过给出惩罚 r_{fcl} 来正则化腿轨迹。$H_{sample,l}$ 是第 l 只脚的高度采样。这样，间隙成本定义为：

$$f_{fcl} = \begin{cases} -1.0, & if\ \max(H_{sample,l}) < -0.2 \\ 0.0, & otherwise \end{cases} \tag{4-10}$$

$$r_{fc} = \sum_{l=1}^{4} r_{fcl} \tag{4-11}$$

注意高度样本是相对于脚高度进行采样，因此 -0.2 表示地形高度比脚低 0.2m；脚比采样的地形高度高 0.2m。

（6）小腿和膝关节碰撞奖励 *Shank and Knee Collision Reward*(r_{co})：我们希望

惩罚脚以外的地形和机器人部件之间的不良接触，以避免硬件损坏：

$$r_{co} = \begin{cases} -c_k, & if\, shank or knee is in collision \\ 0.0, & otherwise \end{cases}$$

这里 c_k 是课程因子，它单调增加并收敛到 1。

（7）节点运动奖励 *Joint Motion Reward(r_j)*：该项惩罚关节速度和加速度以避免振动：

$$r_s = -c_k \sum_{i=1}^{12} (0.01\dot{q}_i^2 + \ddot{q}_i^2) \tag{4-12}$$

其中 $\dot{q}_i, \ddot{q}_i$ 分别是节点的速度和加速度。

（8）节点约束奖励 *Joint Constraint Reward(r_{jc})*：该项在联合空间中引入了一个软约束。为了避免相反方向的膝关节翻转，我们对超过阈值的惩罚：

$$r_{jc,i} = \begin{cases} -(q_i - q_{i,th})^2, & if\ q_i > q_{i,th} \\ 0.0, & otherwise \end{cases} \tag{4-13}$$

$$r_{jc} = \sum_{i=1}^{12} r_{jc,i} \tag{4-14}$$

(4-15)

其中 $q_{i,th}$ 是第 i 个节点的阈值。我们只设置膝关节的阈值。

（9）目标平滑奖励 *Target Smoothness Reward(r_s)*：通过对目标位置一阶和二阶有限差分导数的大小进行惩罚，使生成的足部轨迹变得更平滑:

$$r_s = c_k \sum_{i=1}^{12} ((q_{i,t}^{des} - q_{i,t-1}^{des})^2 + (q_{i,t}^{des} - 2q_{i,t-1}^{des} + q_{i,t-2}^{des})^2) \tag{4-16}$$

其中 $q_{i,t}^{des}$ 是 t 步时间里第 i 个节点的目标位置。

（10）扭矩奖励 *Torque Reward(r_τ)*：我们惩罚节点的扭矩里节省能耗（$\tau \propto electriccurrent$）：

$$r_\tau = -c_k \sum_{i=1}^{12} \tau_i^2 \tag{4-17}$$

其中 τ_i 是执行器网络计算出的第 i 个节点的扭矩。

（11）滑动奖励 *Slip Reward(r_{slip})*：我们惩罚与地面接触的脚的速度以减少滑

动情况：

$$r_{slip} = -c_k \sum_{l \in \{footincontact\}} v_{f,l^2} \tag{4-18}$$

其中 v_{f,l^2} 是与地面接触了的第 l 只脚的速度。

4.3.6　课程

随着策略性能的提高，我们使用两个课程来提高难度。一个课程使用自适应方法[8]p 调整地形难度，另一个改变元素，如奖励或使用逻辑函数[17]p 应用干扰。

对于地形课程，粒子滤波更新地形参数，使它们仍然具有挑战性，但在策略训练期间的任何时候都可以实现[8]p 。

第二个课程将域随机化的幅度和一些奖励项（关节速度、关节加速度、方向、滑移和大腿和小腿接触）乘以单调递增且渐近趋势为 1 的因子：

$$c_{k+1} = (c_k)^d$$

其中 c_k 是第 k 次迭代的课程因子，$d \in (0,1)$ 是收敛率。

4.3.7　学生策略训练

在我们训练好一个可以在特权信息的帮助下穿越各种地形教师策略后，我们就可以将其提炼成一个学生策略，该策略只能访问真实 robot 上可用的信息。我们使用与教师策略相同的训练环境，但在学生高度样本观察中添加额外的噪声：$o_t^{student} = (o_t^p, n(o_t^e))$，其中 $n(o_t^e)$ 是一个用于高度样本输入的噪声模型。该噪声模型模拟了现场部署过程中经常遇到的外部感觉的不同失败案例，具体如下。

当外部感觉中存在较大的噪声时，它变得不可观察；因此，动力学被认为是 POMDP。此外，由于缺乏直接测量的传感器，特权状态是不可观察的。因此，该策略需要考虑顺序相关性来估计不可观察的状态。我们建议使用循环信念状态编码器来组合外部感知和本体感觉的序列，以估计不可观察的状态作为信念状态。

学生策略由循环信念状态编码器和 MLP 组成，如图4-1所示。我们用 h_t 表示循环网络的隐藏状态。信念状态编码器接收 $o_t^{sutdent}, h_t$ 并输出一个潜在向量 b_t，它表示信念状态。我们的目标是将信念状态 b_t 与编码所有运动相关信息的教师策略的特征向量（l_t^e, l_t^{priv}）进行匹配。接下来我们将 o_t^p 和 b_t 传入 MLP，由它计算出最终的动作。MLP 结构与教师策略相同，这样我们就可以重用教师策略的学习权重来初始化学生网络并加快训练速度。

学生策略的训练通过最小化两个损失以有监督的方式进行训练：行为克隆损失 *behavior cloning loss* 和重建损失 *reconstruction loss*。

• 克隆损失定义为给定相同状态和指令的学生动作和教师动作之间的平方距离。

• 重建损失定义为无噪声高度样本和特权信息（o_t^e, s_t^p）及其与信念状态 b_t 的重建之间的平方距离。

我们通过推出学生策略来生成样本，以提高鲁棒性[10,18]p。

4.3.8　高度采样随机化

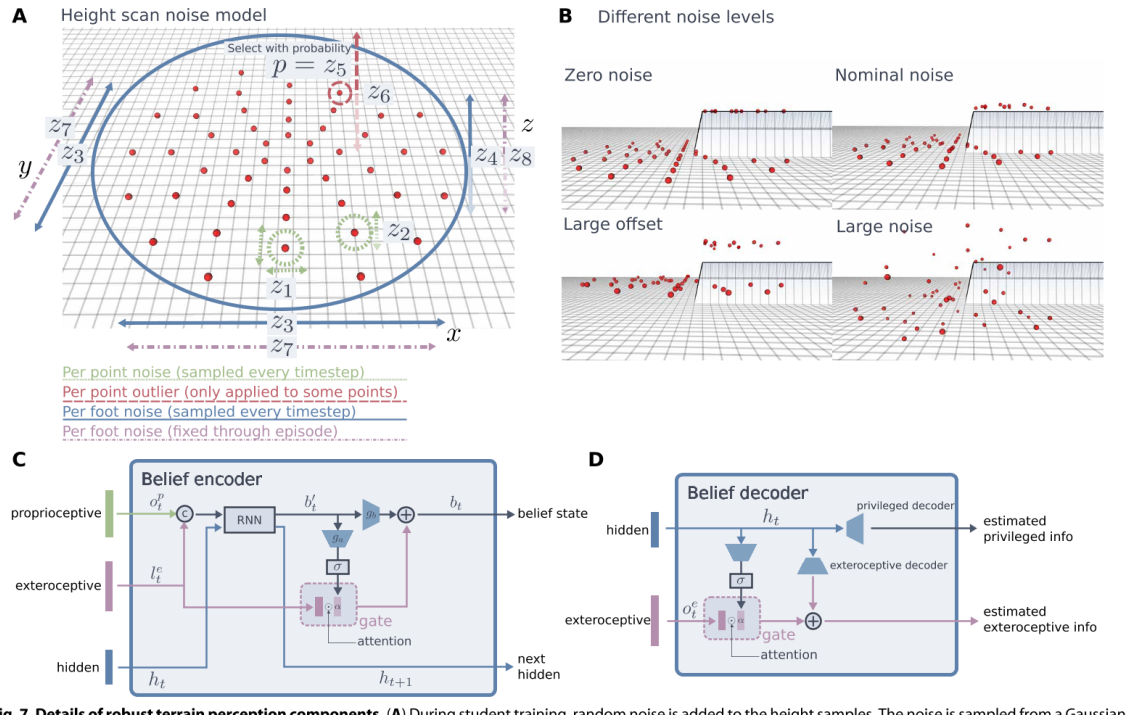

图 4-2　Robust terrain perception[7]p10.

在学生训练过程中，我们使用参数化噪声模型 $n(\sigma_t^e | o_t^e, z), z \in \mathbb{R}^{8\times4}$ 将随机噪声注入到高度样本中。我们在对高度进行采样时应用了两种不同类型的测量噪声，如图4-2A 所示：

（1）横向移动扫描点。

（2）扰动高度值。

每个噪声值都是从高斯分布中采样的，噪声参数 z 定义方差。这两种类型的噪声都应用于三个不同的范围，所有这些都有自己的噪声方差：每个扫描点、每只脚和每一集。每个扫描点和每只脚的噪声值在每个时间步重新采样，而所有扫描点的表观轮廓噪声保持不变。

此外，我们定义了三个具有相关噪声参数 z 的映射条件来模拟不断变化的地图质量和误差源，如图4-2B 所示。

（1）名义噪声假设常规操作期间具有良好的地图质量。

（2）大偏移量噪声来模拟由于姿态估计漂移或可变形地形造成的地图偏移。

（3）幅度较大的噪声来模拟由于遮挡或映射失败导致完全缺乏地形信息的情况。

这三个映射条件在每个训练集的开头以 60%、30% 和 10% 的比例选择。

最后，我们将每个训练地形划分为单元格，并向高度样本添加附加偏移量，具体取决于它采样的单元格。这模拟了不同地形特征的区域之间的转换，如植被和深度雪。参数向量 z 也是学习课程的一部分，其幅度随训练持续时间线性增加。

更详细相关内容看相关附件的 S8。

在学生训练期间，我们随机化每只脚周围绘制的高度样本（图4-2A）。我们扰动每个样本的位置，并将噪声添加到测量的高度值中，如下所示。

$$x_p = r_p \cos(\theta_p) + \epsilon_{px} + \epsilon_{fx} + w_x \tag{4-19}$$

$$y_p = r_p \sin(\theta_p) + \epsilon_{py} + \epsilon_{fy} + w_y \tag{4-20}$$

$$h_p = h(x_p, y_p) + \epsilon_{pz} + \epsilon_{fz} + w_z + \epsilon_{outlier} \tag{4-21}$$

其中 $h(x_p, y_p)$ 表示在点 (x_p, y_p) 处的高度，r_p 是 p 点的径向距离，θ_p 是 p 在脚周围的极坐标中的方位角。$\epsilon_{px}, \epsilon_{py}, \epsilon_{pz}$ 表示每个时间步长里各个点的采样噪声。$\epsilon_{fx}, \epsilon_{fy}, \epsilon_{fz}$ 表示每个时间步长里各个脚的采样噪声。w_x, w_y, w_z 表示每个片段里各个脚的采样噪声。$\epsilon_{outlier}$ 是间歇性添加的大噪声来模拟异常值。

使用参数 z 从正态分布中采样每个噪声。$\epsilon_{px}, \epsilon py \sim \mathcal{N}(0, z_0), \epsilon pz \sim \mathcal{N}(0, z_1), \epsilon_{fx}, \epsilon fy \sim \mathcal{N}(0, z_2), \epsilon fz \sim \mathcal{N}(0, z_3), \epsilon_{outlier} \sim \mathcal{N}(0, z_4)$ 概率分别为 $p = z_5, w_x, w_y \sim \mathcal{N}(0, z_6), w_z \sim \mathcal{N}(0, z_7)$。

我们为学生训练定义三种情况：*nominal, offset, noisy*。每种情况的参数 z 定义如下：

$$z_{nominal} = \langle 0.004, 0.005, 0.01, 0.04, 0.03, 0.05, 0.1 \rangle \tag{4-22}$$

$$z_{offset} = \langle 0.004, 0.005, 0.01, 0.1c_{sk}, 0.1c_{sk}, 0.02, 0.1 \rangle \tag{4-23}$$

$$z_{noisy} = \langle 0.004, 0.1c_{sk}, 0.1c_{sk}, 0.3c_{sk}, 0.3c_{sk}, 0.3c_{sk}, 0.1 \rangle \tag{4-24}$$

其中 c_{sk} 是学生课程因子，会随着训练片段增加而不断线性增加。我们在轨迹的开头和中间随机选择其中一个条件。选择的概率分别是 60%、30% 和 10%。

4.3.9　信念状态寄存器

循环信念状态编码器编码不能直接观察到的状态。为了整合本体感受和外感受数据，我们引入了一个门控编码器，如图 7C 所示，灵感来自门控 RNN 模型[19-20]ᵽ和多模态信息融合 (4-66)。

信念状态编码器学习使用一个可变的门控因子来控制外部感知信息通过的量。首先，内部感知 s_t^p、从含噪声观测提取的外部感知 $l_t^e = g_e(\tilde{o}_t^e)$ 以及隐藏状态 s_t 被 RNN 模型编码成为一个中间信念状态 $b_t t'$。它控制最终进入 b_t 的外部感知信息量：

$$b_{t'}, h_{t+1} = RNN(o_t^p, l_t^e, h_t) \tag{4-25}$$

$$\alpha = \sigma(g_a(b_{t'})) \tag{4-26}$$

$$b_t = g_b(b_{t'}) + l_t^e \odot \alpha \tag{4-27}$$

这里 g_a, g_b 是全连接的神经网络，$\sigma(\cdot)$ 是 sigmoid 函数。

解码器使用相同的门，用于重建特权信息和高度样本（图 7D）。这用于计算重建损失，它鼓励信念状态捕获有关环境的真实信息。我们使用 GR[19]p 作为我们的 RNN 架构。

门结构有效性的评估见第 S9 节。

4.3.10　部署

我们在 PyTorch[21]p 中训练策略，并在没有任何微调的情况下部署在机器人 zero-shot 上。我们通过估计机器人的姿态，并相应地从传感器中调节点云读数，构建了一个以机器人为中心的 2.5D 高程图刷新率为 20 赫兹。该策略以 50Hz 运行，并从最新的高程图中映射采样高度；如果查询位置没有可用的地图信息，则填充随机采样的值。

我们开发了一个高程映射管道，用于在图形处理单元上快速地形映射，以并行化点云处理。我们遵循与 Fankhauser 等人[22]p 使用的类似方法，以卡尔曼滤波的方式更新地图，另外按漂移补偿 *drift compensation* 和光线投射 *ray casting* 以获得更吻合的地图。这种快速映射实现对于保持快速处理速率和跟上我们的控制器实现的快速运动速度至关重要。

第 5 章　Xxx

参考文献

[1] BELLICOSO C, JENELTEN F, FANKHAUSER P, et al. Dynamic locomotion and whole-body control for quadrupedal robots[C/OL]//2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). 2017. http://dx.doi.org/10.1109/iros.2017.8206174.

[2] BELLICOSO C D, JENELTEN F, GEHRING C, et al. Dynamic Locomotion Through Online Nonlinear Motion Optimization for Quadrupedal Robots[J/OL]. IEEE Robotics and Automation Letters, 2018: 2261–2268. http://dx.doi.org/10.1109/lra.2018.2794620.

[3] GEHRING C, COROS S, HUTTER M, et al. An Optimization-based Approach to Controlling Agile Motions for a Quadruped Robot[Z]. 2016.

[4] KALAKRISHNAN M, BUCHLI J, PASTOR P, et al. Fast, robust quadruped locomotion over challenging terrain[C/OL]//2010 IEEE International Conference on Robotics and Automation. 2010. http://dx.doi.org/10.1109/robot.2010.5509805.

[5] GOLDFARB D, IDNANI A. A numerically stable dual method for solving strictly convex quadratic programs[J/OL]. Mathematical Programming, 1983, 27(1): 1–33. http://dx.doi.org/10.1007/bf02591962.

[6] LEE J, BJELONIC M, HUTTER M. Control of Wheeled-Legged Quadrupeds Using Deep Reinforcement Learning[M/OL]. 2023: 119–127. http://dx.doi.org/10.1007/978-3-031-15226-9_14.

[7] MIKI T, LEE J, HWANGBO J, et al. Learning robust perceptive locomotion for quadrupedal robots in the wild[J/OL]. Science Robotics, 2022(VOL. 7, NO. 62). DOI: 10.1126/scirobotics.abk2822.

[8] LEE J, HWANGBO J, WELLHAUSEN L, et al. Learning Quadrupedal Locomotion over Challenging Terrain[J/OL]. Science Robotics, 2020. http://dx.doi.org/10.1126/scirobotics.abc5986.

[9] SCHULMAN J, WOLSKI F, DHARIWAL P, et al. Proximal Policy Optimization Algorithms [A]. 2017.

[10] ROSS S, GORDON G, BAGNELL J. A Reduction of Imitation Learning and Structured Prediction to No-Regret Online Learning[A]. 2010.

[11] HWANGBO J, LEE J, HUTTER M. Per-Contact Iteration Method for Solving Contact Dynamics[J/OL]. IEEE Robotics and Automation Letters, 2018: 895–902. http://dx.doi.org/10.1109/lra.2018.2792536.

[12] RUDIN N, HOELLER D, REIST P, et al. Learning to Walk in Minutes Using Massively Parallel Deep Reinforcement Learning[A]. 2021.

[13] MNIH V, KAVUKCUOGLU K, SILVER D, et al. Playing Atari with Deep Reinforcement Learning[A]. 2013.

[14] ZHU P, LI X, POUPART P, et al. On Improving Deep Reinforcement Learning for POMDPs [A]. 2017.

[15] BAI S, KOLTER J, KOLTUN V. An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling[A]. 2018.

[16] CHEN D, ZHOU B, KOLTUN V, et al. Learning by Cheating[J]. Conference on Robot Learning,Conference on Robot Learning, 2019.

[17] HWANGBO J, LEE J, DOSOVITSKIY A, et al. Learning agile and dynamic motor skills for legged robots[J/OL]. Science Robotics, 2019. http://dx.doi.org/10.1126/scirobotics.aau5872.

[18] CZARNECKI W, PASCANU R, OSINDERO S, et al. Distilling Policy Distillation[A]. 2019.

[19] CHO K, VAN MERRIENBOER B, GULCEHRE C, et al. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation[C/OL]//Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). 2014. http://dx.doi.org/10.3115/v1/d14-1179.

[20] HOCHREITER S, SCHMIDHUBER J. Long short-term memory[J/OL]. Neural Computation, 1997: 1735–1780. http://dx.doi.org/10.1162/neco.1997.9.8.1735.

[21] PASZKE A, GROSS S, MASSA F, et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library: Article No.: 721[M/OL]. 2019: Pages 8026-8037. DOI: https://dl.acm.org/doi/10.5555/3454287.3455008.

[22] FANKHAUSER P, BLOESCH M, HUTTER M. Probabilistic Terrain Mapping for Mobile Robots With Uncertain Localization[J/OL]. IEEE Robotics and Automation Letters, 2018: 3019–3026. http://dx.doi.org/10.1109/lra.2018.2849506.

附录 A　补充内容

在附录里面我会写一些控制相关的一些数学概念的理解。

A.1　拉格朗日乘子法

在机械狗的运动规划中，机械狗质心的运动是通过求解空间中 x,y,z 方向上的五次曲线得到的。这个五次曲线是在一些列约束条件下的最优运动结果。这个结果是通过求解 QP(二次优化) 问题得到的。实际上所谓的 QP 问题的一种形式就是之前高数上很熟悉的‘拉格朗日乘子法（Lagrange Multiplier）’。下面就对这部分知识进行总结，这部分内容主要参考这个博客。

A.1.1　优化问题

通常我们需要求解的优化问题有如下几类：

（1）无约束优化问题，可以写为：

$$minf(x);$$

（2）有等式约束的优化问题，可以写为：

$$minf(x);$$
$$s.t.\quad h_i(x)=0, i=1,\cdots,n;$$

（3）有等式和不等式约束的优化问题，可以写为：

$$minf(x);$$
$$s.t.\quad g_i \leqslant 0, i=1,\cdots,n;$$
$$and\quad h_i(x)=0, i=1,\cdots,n;$$

对于第一类优化问题，常使用的求解方式是 *Fermat* 定理，即使用求取 $f(x)$ 的导数，然后令其为零，可以求得候选的最优值，再在这些候选值中验证；如果是凸函数，可以保证是最优解。

对于第二类优化问题，常使用的求解方式是拉格朗日乘子法，即把等式约束 $h_i(x)$ 用一个系数与 $f(x)$ 写为一个式子，称为‘拉格朗日函数’，而系数称为拉格朗日乘子。通过拉格朗日函数对各个变量求导，令其为零，可以求得候选值集合，然后验证求得最优值。

对于第三类优化问题，常使用的求解方式是 *KKT* 条件。同样地，把所有等式、不等式约束条件与 $f(x)$ 写成一个式子，也叫拉格朗日函数，系数也称为拉格朗日乘子。通过一些条件可以求出最优值的必要条件，这个条件称为 KKT 条件。

A.1.2　拉格朗日乘子法

对于等式约束，我们可以通过一个拉格朗日系数 a 把等式约束和目标函数组合成一个拉格朗日函数：

$$\boldsymbol{L(a,x)=f(x)+ah(x)} \tag{A-1}$$

其中 $\boldsymbol{a,h(x)}$ 分别为一行向量和列向量。最优值可以通过对拉格朗日函数 $\boldsymbol{L(a,x)}$ 的各个参数求导取零，联立等式进行求取。它要求：

（1）拉格朗日函数对 $\boldsymbol{a}$ 的偏导为零，即 $\frac{\partial \boldsymbol{L(a,x)}}{\partial \boldsymbol{a}}=0$；

（2）拉格朗日函数对 $\boldsymbol{x}$ 的偏导为零，即 $\frac{\partial \boldsymbol{L(a,x)}}{\partial \boldsymbol{x}}=0$；

通过联立求解上面两个条件等式可以求得候选的最优值结果 $\boldsymbol{a,x}$ 的候选值，然后验证 $\boldsymbol{x}$ 求得最优值（如果是如函数可以保证是最优解）。

A.1.3　KKT 条件

对于含有不等式约束的优化问题，常用的方法是 KKT 条件同样像等式约束一样将所有的不等约束也添加到拉格朗日函数中谢伟一个式子：

$$\boldsymbol{L(a,b,x)=f(x)+ag(x)+bh(x)} \tag{A-2}$$

KKT 条件指出最优值必须满足一下三个条件：

（1）拉格朗日函数对 x 求导为零，即 $\frac{\boldsymbol{L(a,b,x)}}{\partial \boldsymbol{x}}$；

（2）等式约束自身为零，即 $\boldsymbol{h(x)=0}$；

（3）不等约束及其系数乘积为零，即 $\boldsymbol{ag(x)=0}$；

求取这三个等式之后就能得到候选最优值。其中第三个式子非常有趣，因为 $\boldsymbol{g(x)\leqslant 0}$，如果要满足这个等式，必须 $\boldsymbol{a=0}$ 或者 $\boldsymbol{g(x)=0}$. 这是 *SVM* 的很多重要性质的来源，如支持向量的概念。

A.2　二次规划问题

二次规划 (QP, Quadratic Programming) 定义：目标函数为二次函数，约束条件为线性约束，属于最简单的一种非线性规划。这部分内容主要参考这个博客。

A.2.1　等式约束的 QP 问题

一个标准的等式约束 QP 问题模型如下：

$$\begin{aligned} min \quad & \frac{1}{2}\boldsymbol{x}^T\boldsymbol{H}\boldsymbol{x}+\boldsymbol{g}^T\boldsymbol{x} \\ s.t. \quad & \boldsymbol{a}_i^T\boldsymbol{x}=b_i, \quad i=1,2,\cdots,m \end{aligned} \tag{A-3}$$

其中 $\boldsymbol{H}$ 是由二阶导构成的 Hessian 矩阵，$\boldsymbol{g}^T\boldsymbol{x}$ 是由梯度构成的 Jacobi 矩阵，这里的向量都是列向量，$\boldsymbol{g}^T\boldsymbol{x}$ 表示转置成行向量。

其对应的拉格朗日函数为:

$$\boldsymbol{L}(\boldsymbol{x},\boldsymbol{\lambda})=\frac{1}{2}\boldsymbol{x}^T\boldsymbol{H}\boldsymbol{x}+\boldsymbol{g}^T\boldsymbol{x}+\sum_{i=1}^{m}\lambda_i(\boldsymbol{a}_i^T\boldsymbol{x}-b_i) \tag{A-4}$$

满足 KKT 条件，即满足：

$$\begin{cases} \frac{\partial \boldsymbol{L}}{\partial \boldsymbol{x}}=\boldsymbol{H}\boldsymbol{x}+\boldsymbol{g}+\sum_{i=1}^{m}\lambda_i\boldsymbol{a}_i=0 \\ \frac{\partial \boldsymbol{L}}{\partial \lambda_i}=\boldsymbol{a}_i^T\boldsymbol{x}-b_i=0, \quad i=1,2,\cdots,m \end{cases} \tag{A-5}$$

将其写成矩阵形式：

$$\boldsymbol{F}(\boldsymbol{x},\boldsymbol{\lambda})=\begin{bmatrix} \boldsymbol{H}\boldsymbol{x}+\boldsymbol{g}+\boldsymbol{\lambda}\boldsymbol{a} \\ \boldsymbol{a}^T\boldsymbol{x}-\boldsymbol{b} \end{bmatrix}=0 \tag{A-6}$$

其中

$$\boldsymbol{\lambda}=\begin{bmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_m \end{bmatrix}, \quad \boldsymbol{a}^T=\begin{bmatrix} \boldsymbol{a}_1^T \\ \boldsymbol{a}_2^T \\ \vdots \\ \boldsymbol{a}_m^T \end{bmatrix}, \quad \boldsymbol{b}=\begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix} \tag{A-7}$$

这是一个线性方程组，易于求解，KKT 方程组的解 $\boldsymbol{x}^*,\boldsymbol{\lambda}^*$，即为优化模型的解。

A.2.2　不等式约束的 QP 问题

一个标准的等式不等式联合约束 QP 问题模型如下：

$$\begin{aligned} min \quad & \frac{1}{2}\boldsymbol{x}^T\boldsymbol{H}\boldsymbol{x}+\boldsymbol{g}^T\boldsymbol{x} \\ s.t. \quad & \boldsymbol{a}_i^T\boldsymbol{x}=b_i, \quad i\in\mathbb{E} \end{aligned} \tag{A-8}$$

$$\boldsymbol{h}_j^T\boldsymbol{x}\leqslant t_j, \quad j\in\mathbb{I} \tag{A-9}$$

其中 $\boldsymbol{H}$ 是由二阶导构成的 Hessian 矩阵，$\boldsymbol{g}^T\boldsymbol{x}$ 是由梯度构成的 Jacobi 矩阵，这里的向量都是列向量，$\boldsymbol{g}^T\boldsymbol{x}$ 表示转置成行向量。$i\in\mathbb{E}$ 表示 m 个等式约束集合，$i\in\mathbb{I}$ 表示 n 个不等式约束集合。下面介绍两种不等式约束条件下的 QP 问题求解方案。

A.2.2.1　内点法

模型的拉格朗日函数为：

$$\boldsymbol{L}(\boldsymbol{x},\boldsymbol{\lambda},\boldsymbol{\mu})=\frac{1}{2}\boldsymbol{x}^T\boldsymbol{H}\boldsymbol{x}+\boldsymbol{g}^T\boldsymbol{x}+\sum_{i=1}^{m}\lambda_i(\boldsymbol{a}_i^T\boldsymbol{x}-b_i)+\sum_{j=1}^{n}\mu_j(\boldsymbol{h}_j^T\boldsymbol{x}-t_j) \tag{A-10}$$

Notes A.2.1. 内点法详细可以在接下来的内容中了解后补充一下。

以原始对偶内点法为例，加入少量 $\boldsymbol{\tau}$ 扰动后 KKT 条件为：

$$\begin{cases}\frac{\partial \boldsymbol{L}}{\partial \boldsymbol{x}}=\boldsymbol{H}\boldsymbol{x}+\boldsymbol{g}+\sum_{i=1}^{m}\lambda_i\boldsymbol{a}_i+\sum_{j=1}^{n}\mu_j\boldsymbol{h}_j=0\\ \frac{\partial \boldsymbol{L}}{\partial \lambda_i}=\boldsymbol{a}_i^T\boldsymbol{x}-b_i=0,\quad i=1,2,\cdots,m\\ \boldsymbol{h}_j^T\boldsymbol{x}\leqslant t_j\\ \mu_j(\boldsymbol{h}_j^T\boldsymbol{x}-t_j)+\tau_j=0\\ \mu_j\geqslant 0,\quad j=1,2,\cdots,n\end{cases} \tag{A-11}$$

为了更快地检查解是否在约束空间内，我们在不等式方程组引入了松弛变量 $s_j=t_j-\boldsymbol{h}_j^T\boldsymbol{x}$，因为判断 $s_j\leqslant 0$ 要比判断 $\boldsymbol{h}_j^T\boldsymbol{x}\leqslant t_j$ 方便的多，这样一来上式变为：

$$\begin{cases}\frac{\partial \boldsymbol{L}}{\partial \boldsymbol{x}}=\boldsymbol{H}\boldsymbol{x}+\boldsymbol{g}+\sum_{i=1}^{m}\lambda_i\boldsymbol{a}_i+\sum_{j=1}^{n}\mu_j\boldsymbol{h}_j=0\\ \frac{\partial \boldsymbol{L}}{\partial \lambda_i}=\boldsymbol{a}_i^T\boldsymbol{x}-b_i=0,\quad i=1,2,\cdots,m\\ \boldsymbol{h}_j^T\boldsymbol{x}+s_j=t_j\\ \mu_j(\boldsymbol{h}_j^T\boldsymbol{x}-t_j)+\tau_j=0\rightarrow\mu_j s_j-\tau_j=0\\ \mu_j,s_j\geqslant 0,\quad j=1,2,\cdots,n\end{cases} \tag{A-12}$$

将其写成矩阵形式：

$$\boldsymbol{F}(\boldsymbol{x},\boldsymbol{s},\boldsymbol{\lambda},\boldsymbol{\mu})=\begin{bmatrix}\boldsymbol{H}\boldsymbol{x}+\boldsymbol{g}+\boldsymbol{\lambda}\boldsymbol{a}+\boldsymbol{\mu}\boldsymbol{h}\\ \boldsymbol{a}^T\boldsymbol{x}-\boldsymbol{b}\\ \boldsymbol{h}^T\boldsymbol{x}+\boldsymbol{s}-\boldsymbol{t}\\ \boldsymbol{\mu}\boldsymbol{s}-\boldsymbol{\tau}1\end{bmatrix}=0 \tag{A-13}$$

Notes A.2.2. 不知道这里添加的扰动 τ 后面的那个 1 是什么意思？难道是笔误？

理论上第一行是关于 $\boldsymbol{x}$ 的偏导项，第二项和第三项分别是关于等式约束和不等式约束引入的拉格朗日乘子的偏导项。最后一项 $\boldsymbol{\mu s}-\boldsymbol{\tau}1=0$ 的含义不是很清楚是什么意思？这里面 $\boldsymbol{\mu}$ 是引入的拉格朗日乘子，$\boldsymbol{s}$ 是引入的松弛变量，$\boldsymbol{\tau}$ 是少量扰动。

其中：

$$\boldsymbol{\lambda}=\begin{bmatrix}\lambda_1 & 0 & \cdots & 0\\ 0 & \lambda_2 & \cdots & 0\\ \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \cdots & \lambda_m\end{bmatrix},\quad \boldsymbol{a}^T=\begin{bmatrix}\boldsymbol{a}_1^T\\ \boldsymbol{a}_2^T\\ \vdots\\ \boldsymbol{a}_m^T\end{bmatrix},\quad \boldsymbol{b}=\begin{bmatrix}b_1\\ b_2\\ \vdots\\ b_m\end{bmatrix},$$

$$\boldsymbol{\mu}=\begin{bmatrix}\mu & 0 & \cdots & 0\\ 0 & \mu_2 & \cdots & 0\\ \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \cdots & \mu_m\end{bmatrix},\quad \boldsymbol{h}^T=\begin{bmatrix}\boldsymbol{h}_1^T\\ \boldsymbol{h}_2^T\\ \vdots\\ \boldsymbol{h}_n^T\end{bmatrix},\quad \boldsymbol{t}=\begin{bmatrix}t_1\\ t_2\\ \vdots\\ t_n\end{bmatrix},\quad \boldsymbol{s}=\begin{bmatrix}s_1\\ s_2\\ \vdots\\ s_n\end{bmatrix} \tag{A-14}$$

牛顿解法方程组：

$$\boldsymbol{F}(\boldsymbol{x}_k,\boldsymbol{s}_k,\boldsymbol{\lambda}_k,\boldsymbol{\mu}_k)+\boldsymbol{F}'(\Delta\boldsymbol{x}_k,\Delta\boldsymbol{s}_k,\Delta\boldsymbol{\lambda}_k,\Delta\boldsymbol{\mu}_k)=0 \tag{A-15}$$

也即：

$$\begin{bmatrix}\boldsymbol{H} & 0 & \boldsymbol{a}^T & \boldsymbol{h}^T\\ \boldsymbol{h}^T & I & 0 & 0\\ \boldsymbol{a}^T & 0 & 0 & 0\\ 0 & \boldsymbol{\mu}_k & 0 & \boldsymbol{s}^T\end{bmatrix}\begin{bmatrix}\Delta\boldsymbol{x}_k\\ \Delta\boldsymbol{s}_k\\ \Delta\boldsymbol{\lambda}_k\\ \Delta\boldsymbol{\mu}_k\end{bmatrix}=-\begin{bmatrix}\boldsymbol{H}\boldsymbol{x}_k+\boldsymbol{g}+\boldsymbol{\lambda}_k\boldsymbol{a}+\boldsymbol{\mu}_k\boldsymbol{h}\\ \boldsymbol{h}^T\boldsymbol{x}_k+\boldsymbol{s}_k-\boldsymbol{t}\\ \boldsymbol{a}^T\boldsymbol{x}_k-\boldsymbol{b}\\ \boldsymbol{\mu}_k\boldsymbol{s}_k-\boldsymbol{\tau}_k1\end{bmatrix} \tag{A-16}$$

得到一组 $\Delta\boldsymbol{x}_k,\Delta\boldsymbol{s}_k,\Delta\boldsymbol{\lambda}_k,\Delta\boldsymbol{\mu}_k$ 然后更新变量 $\boldsymbol{x}_k,\boldsymbol{s}_k,\boldsymbol{\lambda}_k,\boldsymbol{\mu}_{k+1}=(\boldsymbol{x}_k,\boldsymbol{s}_k,\boldsymbol{\lambda}_k,\boldsymbol{\mu}_k)+\boldsymbol{\alpha}\Delta\boldsymbol{x}_k,\Delta\boldsymbol{s}_k,\Delta\boldsymbol{\lambda}_k,\Delta\boldsymbol{\mu}_k$。同时更新 $\boldsymbol{\tau}_{k+1}=-\boldsymbol{\sigma}\sum_{j=1}^{n}\boldsymbol{\mu}_{j,k}\boldsymbol{s}_{j,k},\quad \boldsymbol{\sigma}\in[0,1]$ 进入第 $k+1$ 次迭代直到方程组的解，并且满足 $\boldsymbol{\tau}_k\leqslant\boldsymbol{\epsilon}$。

Notes A.2.3. 这下终于清楚论文里面所谓的二次规划问题具体在说什么了。刚巨额这个过程放在编程上，想要实现也不是个容易的事情。除此之外，实现了的接口用起来对里面的各个参数选择也是个有挑战大问题，可能要花费较长的时间来调整。2023-10-05 23:13:53，竟然这么快又 11 点多了！好在我下午还是回来了，不然这花功夫的笔记进度要延后许久呀。

A.2.2.2　积极集法

积极集法属于图形法在 QP 问题上的扩展，其通过求解有限个等式约束 QP 问题来解决一般约束下的 QP 模型。当不等式约束条件不多时，也是一种高效的求解 QP 算法。积极集法首先将 QP 中所有的不等式约束视为等式约束。把不等式约束直接转成等式约束当然是存在问题的，不等式约束存在有效和无效两种情况，而有效无效很容易通过该不等式对应的拉格朗日乘子进行判断。不等式约束的互补松弛条件告诉我们，不等式对应的拉格朗日乘子应当满足 $\lambda_i \geqslant 0$。

Notes A.2.4. $\lambda_i \geqslant 0$ 这个松弛条件哪里提到的？得查一查。

在第 k 次迭代开始时，我们首先检查当前迭代点 $\boldsymbol{x}_k$ 是否为当前工作集 $\mathbb{W}_{\mathbb{k}}$（有效不等式约束集合）下的最优点。如果不是，我们就通过求解一个等式约束的 QP 命题来得到一个前进方向 p。在计算 p 的时候，只关注 $\mathbb{W}_{\mathbb{k}}$ 中的不等式约束并将其转化为等式约束，而忽略其他不等式约束。令 $\boldsymbol{d} = \boldsymbol{x} - \boldsymbol{x}_k$，带入原命题得：

$$\begin{aligned} min \quad & \frac{1}{2}(\boldsymbol{d} + \boldsymbol{x}_k)^T \boldsymbol{H}(\boldsymbol{d} + \boldsymbol{x}_k) + \boldsymbol{g}^T(\boldsymbol{d} + \boldsymbol{x}_k) \\ s.t. \quad & \boldsymbol{a}_i^T(\boldsymbol{d} + \boldsymbol{x}_k) = b_i, \quad i \in \mathbb{W}_{\mathbb{k}} \end{aligned} \tag{A-17}$$

Notes A.2.5. 这块暂时不继续花时间了，理解问题就可以了。另外也感觉这部分讲的不是很清楚。

A.3　RL 相关资料

这篇博客介绍了很多 RL 控制相关的文献，包括我读过的一些文献，没读过的后面可以一一下载下来读一下。