Constrained Trajectory Optimization For Complex Robotics Systems

机器人系统的带约束轨迹规划

杨硕

大纲

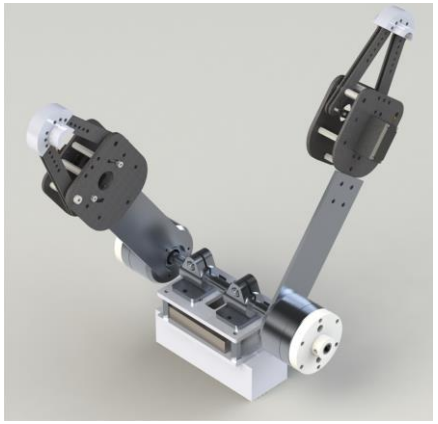
1. 复杂机器人运动轨迹规划的定义和其中的挑战
2. 基础知识
3. Differential Dynamic Programming
4. Direct Collocation
5. Planning As Inference
6. 相关的论文和学习资料

复杂机器人运动轨迹规划的定义和其中的挑战

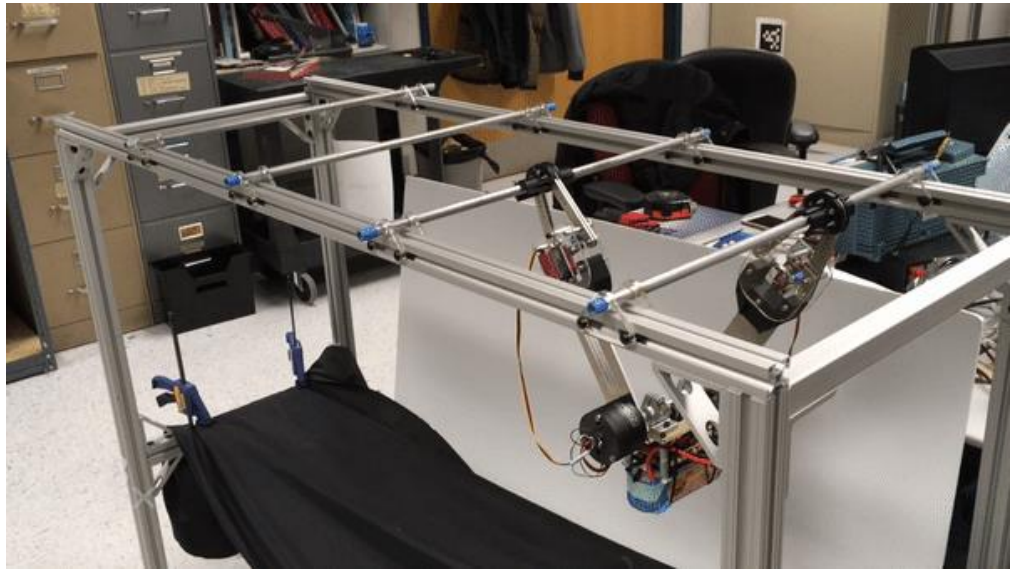
- 复杂的机器人系统往往为高自由度、欠驱动、非线性系统
- 轨迹规划：如何通过单个关节的运动实现机器人特定肢体的位移
- 轨迹控制：对每个关节进行独立的PID控制不一定能够实现整体的轨迹执行目标
- 复杂机器人系统会发生和环境的交互，特别是摩擦力和碰撞
- 轨迹规划和控制都必须通过机载计算机实时进行

例子：欠驱动系统（猴子机器人）

- 平面系统，三个关节，晃动过程中只有两个关节有驱动

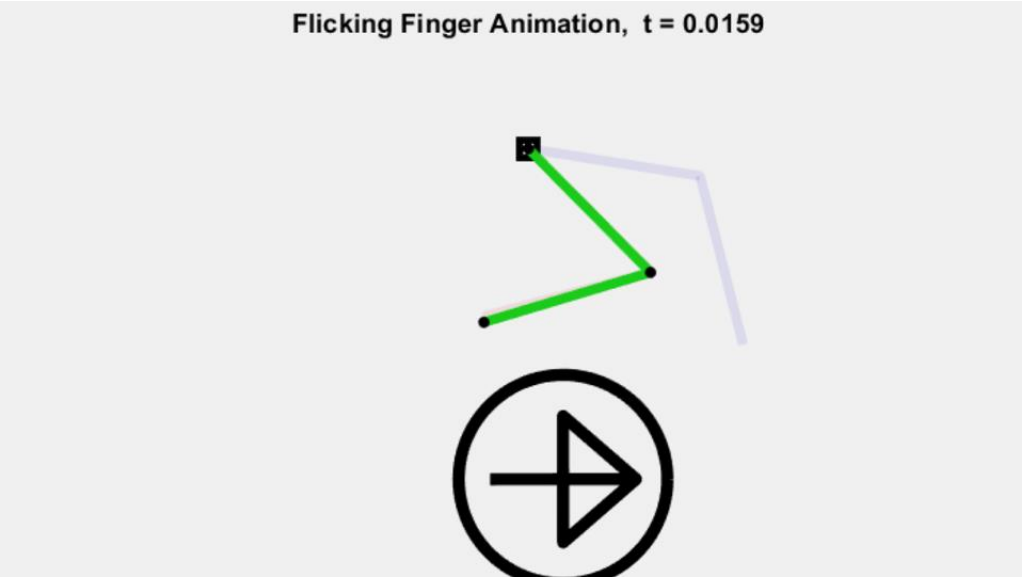


- 对每个关节进行独立的PID控制不一定能够实现整体的轨迹执行目标

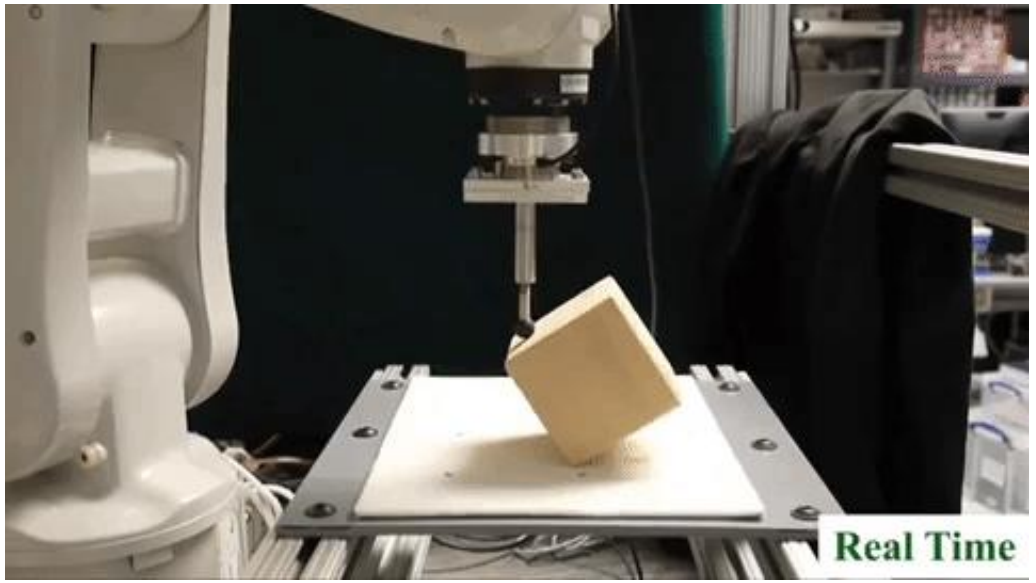


- 需要同时规划轨迹以及考虑欠驱动的轨迹控制器

例子：机械臂控制中的欠驱动和摩擦力



两自由度机械臂，两个关节都有驱动电机。下方是一个可自由旋转的轮子。机械臂和轮子合起来有3个自由度，而只有2个自由度有电机的控制。机械臂只能通过摩擦来转动轮子。



一个更加复杂的实际例子

例子：高自由度系统（人形机器人）

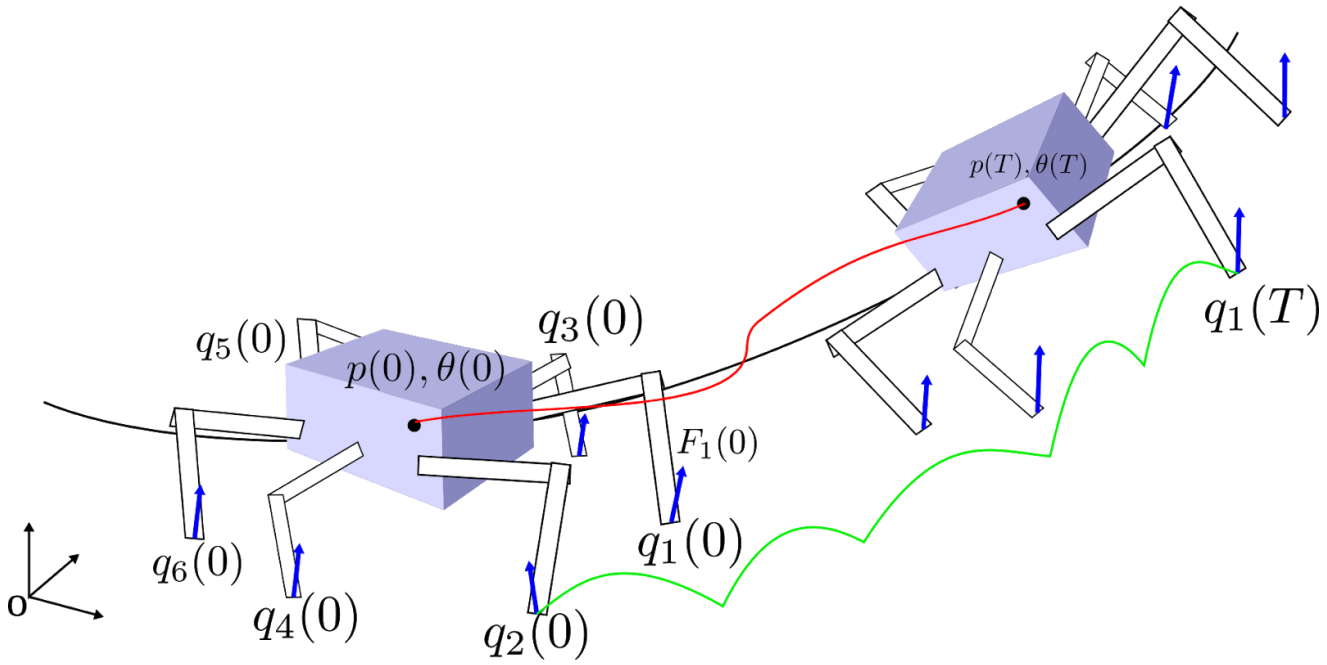
最关注躯干的移动，而躯干的移动只能靠脚和地面的接触力来驱动

"Recent Progress on Atlas, the World’s Most Dynamic Humanoid Robot" - Scott Kuindersma

我的观察和理解：波士顿动力机器人采用Direct Collocation作为机器人的运动规划方法，在线生成轨迹和处理约束

Seminar: Recent Progress on Atlas, the World's Most Dynamic Humanoid Robot - Scott Kuindersma

如何定义轨迹规划



$$x(t) = [p(t) \;\; \theta(t) \;\; \dot{p}(t) \;\; \omega(t) \;\; q_1(t) \;\; q_2(t) \;\; q_3(t) \;\; q_4(t) \;\; q_5(t) \;\; q_6(t)]^T$$

给定一个机器人系统的动力学方程

我们想让机器人从它的状态空间中的A点运动到B点

寻找满足动力学方程和额外的约束的从A点到B点的轨迹，以及输入机器人系统的控制量

轨迹规划中的约束

机器人的关节有位置、速度、力矩的限制

机器人多个关节的运动可能会被限制耦合在一起

机器人一段时间内消耗的能量为定值

对轨迹规划的要求

1. 实时。解算运行2-5秒时长的轨迹的求解速度必须小于0.5秒甚至达到50Hz（这样才能做MPC）
2. 尽量精确地符合约束。所有的等式约束不能有较大的违反值
3. 最好可以解出反馈控制器。反馈控制器可以提高控制稳定性。

三种轨迹规划的方法

Differential Dynamic Programming（微分动态规划）

Direct Collocation（直接配点法）

Planning As Inference （规划即推理）

我们主要关注基于模型的方法

1. 不断进步的精密制造业会提供越来越精确的机器人模型

2. 很多无模型的方法都是以基于模型的方法为基础的

基础知识：
动力学方程、线性化、LQR

基础知识：系统的动力学方程

机器人系统本质是可用经典力学分析的质点系统

定义机器人系统的广义位置（躯干COM的位置、关节角度等）为p，定义广义速度（躯干COM的速度、关节角速度等）为v。

则机器人的状态为$x = \begin{bmatrix} p \\ v \end{bmatrix}$，状态的导数为$\dot{x} = \begin{bmatrix} \dot{p} \\ \dot{v} \end{bmatrix}$。其中$\dot{p} = v$，$\dot{v}$为系统的广义加速度$a$（躯干COM的加速度、关节角加速度等），则也有$\ddot{p} = a$。

作用在机器人上的力、力矩等控制量会改变状态的导数（比如根据$F = ma$,有$\dot{v} = F/m$），将所有的控制量构成向量u，系统的动力学特性可写成

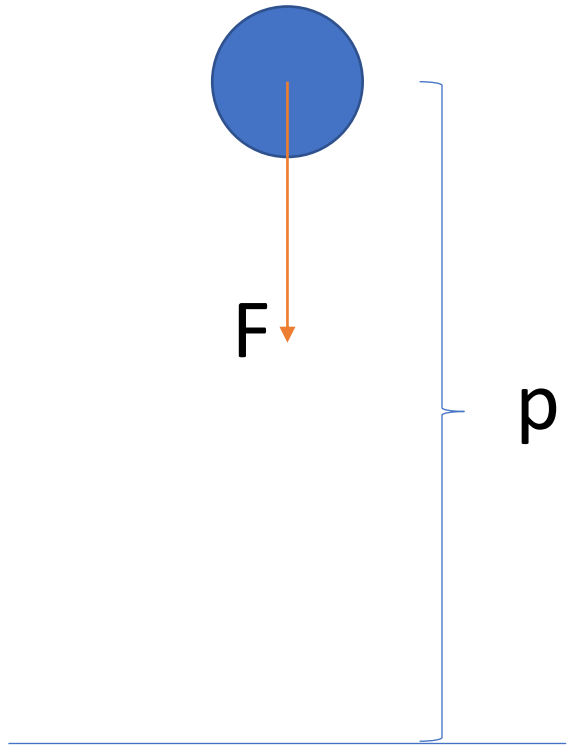
$$\dot{x} = f(x, u)$$

下落的球

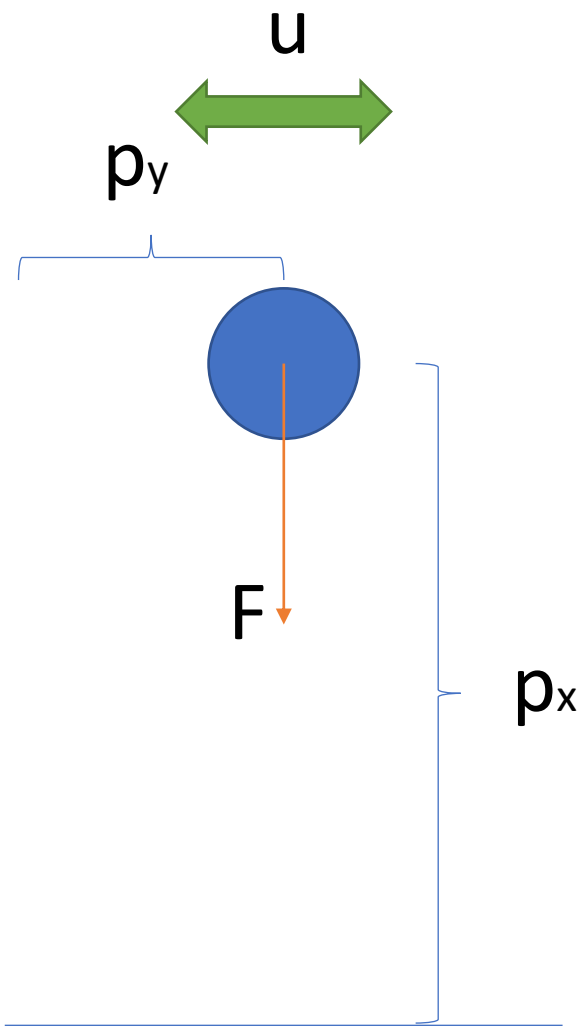
只考虑竖直方向的运动。球的高度为p，速度为$v = \dot{p}$,球受到重力的影响,球上还被施加了一个控制的外力F，所以$\ddot{p} = F - g$。系统的动力学方程可写为

$$\dot{x} = \begin{bmatrix} \dot{p} \\ \ddot{p} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} p \\ v \end{bmatrix} + \begin{bmatrix} 0 \\ -g \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} F$$

$$\dot{x} = f(x, F)$$



下落的球



如果现在我们同时考虑水平方向的运动$p_x, \dot{p}_x$和竖直方向的运动$p_y, \dot{p}_y$，系统的状态为$x = \begin{bmatrix} p_x \\ \dot{p}_x \\ p_y \\ \dot{p}_y \end{bmatrix}$。球上有个神奇的喷气装置可以直接控制水平方向的速度，控制量为u,则

$$\dot{x} = \begin{bmatrix} \dot{p}_x \\ \ddot{p}_x \\ \dot{p}_y \\ \ddot{p}_y \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} p_x \\ \dot{p}_x \\ p_y \\ \dot{p}_y \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ -g \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} F + \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} u$$

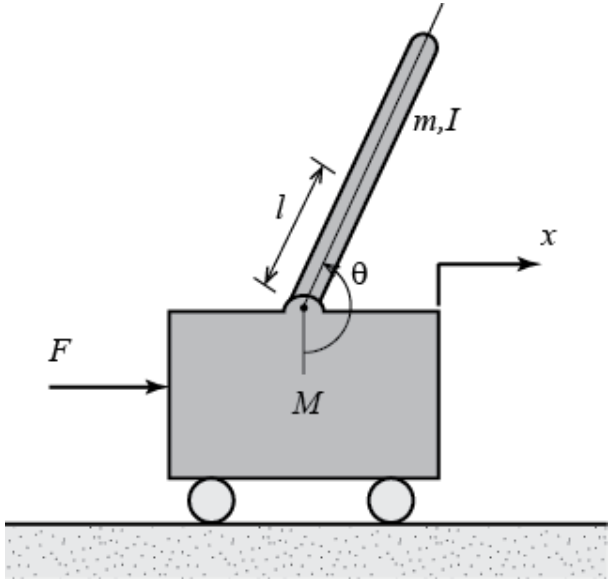
$$\dot{x} = f(x, \begin{bmatrix} F \\ u \end{bmatrix})$$

倒立摆小车模型

定义X $= \begin{bmatrix} x \\ \dot{x} \\ \theta \\ \dot{\theta} \end{bmatrix}$

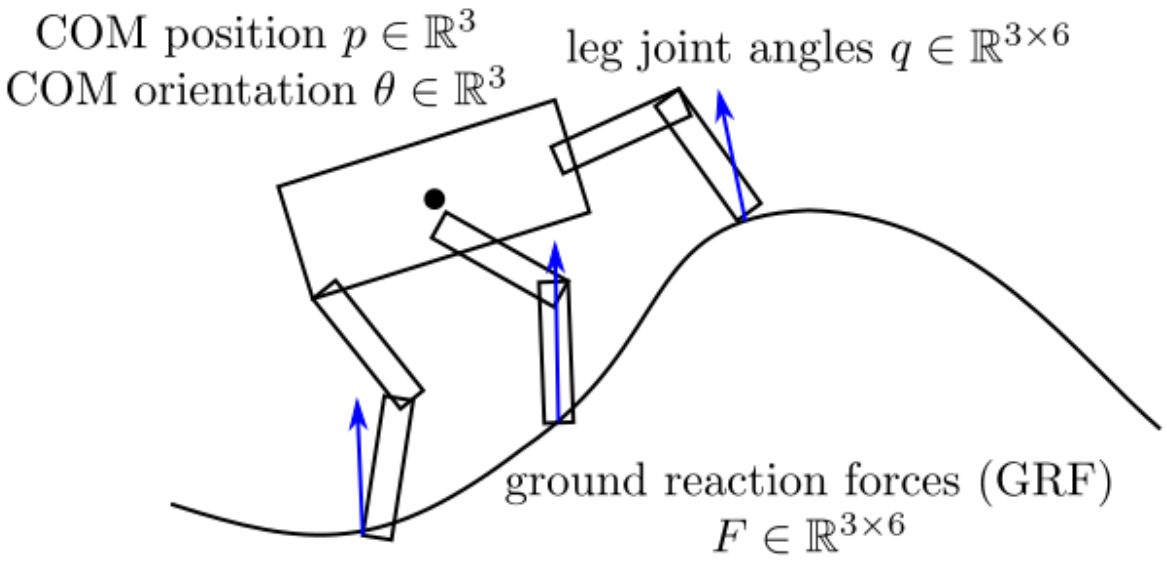
$$(M+m)\ddot{x} + b\dot{x} + ml\ddot{\theta}\cos\theta - ml\dot{\theta}^2\sin\theta = F$$

$$(I+ml^2)\ddot{\theta} + mgl\sin\theta = -ml\ddot{x}\cos\theta$$



- https://ctms.engin.umich.edu/CTMS/index.php?example=InvertedPendulum&section=SystemModeling

一种多足机器人在复杂地形上的动力学模型



System model

$$x = [p \quad \theta \quad \dot{p} \quad \dot{\theta} \quad q]$$

$$u = [F \quad \dot{q}]$$

$$\dot{x} = \begin{bmatrix} \dot{p} \\ \dot{\theta} \\ \ddot{p} \\ \ddot{\theta} \\ \dot{q} \end{bmatrix} = \begin{bmatrix} \dot{p} \\ \dot{\theta} \\ g + \sum F_i \\ I^{-1}(\dot{I} - \omega \times I\omega + \sum r_{f_i} \times F_i) \\ \dot{q} \end{bmatrix}$$

$x(t)$ and $u(t)$ subject to a number of constra

- Dai, Hongkai, Andrés Valenzuela, and Russ Tedrake. "Whole-body motion planning with centroidal dynamics and full kinematics." *2014 IEEE-RAS International Conference on Humanoid Robots*. IEEE, 2014.

非线性系统的线性化

选择泰勒展开的参考点为x_0, u_0 对于函数$f(x,u)$，在点x_0, u_0处通过一阶泰勒展开获得它的线性化函数为

$$f(x,u) = f(x_0,u_0) + \left.\frac{\partial f}{\partial x}\right|_{x=x_0}(x-x_0) + \left.\frac{\partial f}{\partial u}\right|_{u=u_0}(u-u_0)$$

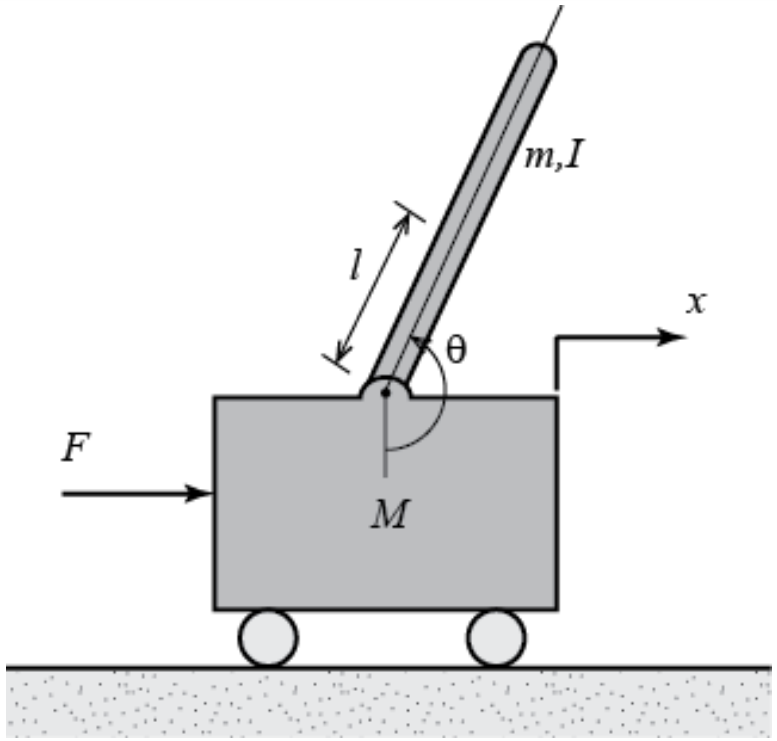
如果我们定义$x = x_0 + \delta x$和$u = u_0 + \delta u$, 则$\dot{x} = \dot{x}_0 + \delta\dot{x}$,由此可以获得

$$\delta\dot{x} = \left.\frac{\partial f}{\partial x}\right|_{x=x_0}\delta x + \left.\frac{\partial f}{\partial u}\right|_{u=u_0}\delta u$$

如果我们把$\delta x = x - x_0$ 当做我们要研究的状态x, $\delta u = u - u_0$为控制量u，然后令$A = \left.\frac{\partial f}{\partial x}\right|_{x=x_0}$且$B = \left.\frac{\partial f}{\partial u}\right|_{u=u_0}$，则

$$\dot{\delta x} = A\delta x + B\delta u$$

倒立摆小车模型



我们在$X_0 = [0 \quad 0 \quad \pi \quad 0]^T$处线性化，将$\theta$在$\pi$周围小量近似，取$\theta = \pi + \phi$，

$$(I + ml^2)\ddot{\phi} - mgl\phi = ml\ddot{x}$$
$$(M + m)\ddot{x} + b\dot{x} - ml\ddot{\phi} = F$$

可整理成

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\phi} \\ \ddot{\phi} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & \frac{-(I+ml^2)b}{I(M+m)+Mml^2} & \frac{m^2gl^2}{I(M+m)+Mml^2} & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{-mlb}{I(M+m)+Mml^2} & \frac{mgl(M+m)}{I(M+m)+Mml^2} & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \phi \\ \dot{\phi} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{I+ml^2}{I(M+m)+Mml^2} \\ 0 \\ \frac{ml}{I(M+m)+Mml^2} \end{bmatrix} u$$

注意这里我们把δX直接写成了x或者ϕ, 另外δu就等于F。

- https://ctms.engin.umich.edu/CTMS/index.php?example=InvertedPendulum&section=SystemModeling

线性二次型调节器

如果我们把$\delta x = x - x_0$当做我们要研究的状态x, $\delta u = u - u_0$为控制量u，然后令$A = \left.\frac{\partial f}{\partial x}\right|_{x=x_0}$且$B = \left.\frac{\partial f}{\partial u}\right|_{u=u_0}$，则

$$\dot{\delta x} = A\delta x + B\delta u$$

我们希望通过控制δu,使得δx在无穷时间后趋于0，且x和u都不偏离x_0和u_0太多（假设$0 = f(x_0, u_0)$）。

如何用数学表达这个问题并求解?

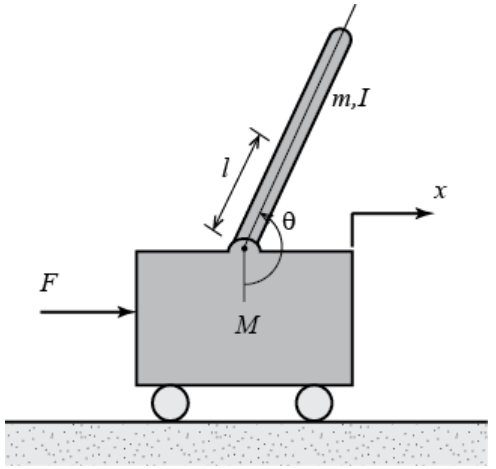
线性二次型调节器

如果我们把$\delta x = x - x_0$ 当做我们要研究的状态x, $\delta u = u - u_0$为控制量u，然后令$A = \left.\frac{\partial f}{\partial x}\right|_{x=x_0}$ 且$B = \left.\frac{\partial f}{\partial u}\right|_{u=u_0}$，则

$$\dot{\delta x} = A\delta x + B\delta u$$

我们希望通过控制δu,使得δx在无穷时间后趋于0，且x和u都不偏离x_0和u_0太多（假设$0 = f(x_0, u_0)$）。

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\phi} \\ \ddot{\phi} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & \frac{-(I+ml^2)b}{I(M+m)+Mml^2} & \frac{m^2gl^2}{I(M+m)+Mml^2} & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{-mlb}{I(M+m)+Mml^2} & \frac{mgl(M+m)}{I(M+m)+Mml^2} & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \phi \\ \dot{\phi} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{I+ml^2}{I(M+m)+Mml^2} \\ 0 \\ \frac{ml}{I(M+m)+Mml^2} \end{bmatrix} u$$



线性二次型调节器

这个问题最基本的表达形式是线性二次型调节器（LQR, linear quadratic regulator），控制论中最经典的问题。

首先我们很不负责任地扔掉δ符号（所有的控制论的书都这么干，这样其实非常容易让人迷惑）。

$$\dot{x} = Ax + Bu$$

定义关于x和u的代价函数$J = \int_0^\infty [x^T Q x + u^T R u] dt$，其中$Q$和$R$是半正定或者正定矩阵。

我们希望通过以下的优化问题找到控制量u

$$\min_u \quad J = \int_0^\infty [x^T Q x + u^T R u] dt \tag{1}$$

$$s.t. \quad \dot{x} = Ax + Bu \tag{2}$$

这就是著名的连续时间无穷时域线性二次型调节器（Continuous time infinite horizon LQR）。

连续时间无穷时域线性二次型调节器

$$\min_{u} \quad J = \int_0^{\infty} [x^T Q x + u^T R u] dt$$
$$s.t. \quad \dot{x} = Ax + Bu$$

这个优化问题的解有许多种推导方法，最核心的原理来自哈密尔顿-雅克比-贝尔曼方程(HJB Equation)。其中关键的感性理解是当u取最优的值的时候,J可以写成一个关于x的二次型函数

$$J^*(x) = x^T S x, \quad S\text{为一个特定的半正定矩阵}$$

这个形式被称为后续代价（cost-to-go）函数，代表从某个x出发直到无穷时间这期间总共的代价。S是如下方程（algebraic Riccati equation）的解

$$0 = SA + A^T S - SBR^{-1}B^T S + Q$$

从J的二次型表示可以推出最优的u为

$$u^* = -R^{-1}B^T S x = -Kx$$

连续时间有穷时域线性二次型调节器

$$\min_u \quad J = \int_0^{\infty} [x^T Q x + u^T R u] dt$$
$$s.t. \quad \dot{x} = Ax + Bu$$

$$\min_u \quad J = \int_0^{T} [x^T Q x + u^T R u] dt$$
$$s.t. \quad \dot{x} = Ax + Bu$$

有穷时域问题比无穷时域问题困难很多

有无限长时间读完PhD vs 要在给定时间段内读完PhD

连续时间有穷时域线性二次型调节器

$$\min_u \quad J = \int_0^\infty [x^T Q x + u^T R u] dt$$
$$s.t. \quad \dot{x} = Ax + Bu$$

后续代价（cost-to-go）函数不显式依赖时间

$$J^*(x) = x^T S x$$

需要求解algebraic Riccati equation

$$0 = SA + A^T S - SBR^{-1}B^T S + Q$$

最优的u为

$$u^* = -R^{-1}B^T S x = -Kx$$

$$\min_u \quad J = \int_0^T [x^T Q x + u^T R u] dt$$
$$s.t. \quad \dot{x} = Ax + Bu$$

后续代价（cost-to-go）函数显式依赖时间

$$J^*(x,t) = x^T S(t) x$$

需要求解differential Riccati equation

$$-\dot{S}(t) = S(t)A + A^T S(t) - S(t)BR^{-1}B^T S(t) + Q$$

最优的u为

$$u^* = -R^{-1}B^T S(t) x = -K(t)x$$

离散时间有穷时域线性二次型调节器

Differential Riccati equation 基本只能通过计算机程序进行离散化求解

$$-\dot{S}(t) = S(t)A + A^T S(t) - S(t)BR^{-1}B^T S(t) + Q$$

我们把从$t = 0$到$t = T$的时间分成N个时刻，每个时刻之间的时间差为$\Delta t = T/(N-1)$，然后取$S(N) = Q$, Riccati equation可被离散化成

$$S[n-1] = Q + A^T S[n]A - A^T S[n]B(R + B^T S[n]B)^{-1}B^T S[n]A$$

由于我们已经计算出$u^* = -R^{-1}B^T S(t)x = -K(t)x$，所以如果离散化计算S，那么我们相应地也要对$x$和$u$进行离散化，对于$\dot{x} = Ax + Bu$，它的离散化方程为

$$x[n+1] = (A + I\Delta t)x[n] + B\Delta t u[n]$$

然后很多控制理论的教科书就直接把$(A + I\Delta t)$和$B\Delta t$又直接写成了A和B。

$$x[n+1] = Ax[n] + Bu[n]$$

离散时间有穷时域线性二次型调节器

离散化以后的整个问题变成

$$\begin{aligned} \min_{u} \quad & J = \sum_{n=0}^{T} x[n]^T Q x[n] + u[n]^T R u[n] \\ s.t. \quad & x[n+1] = Ax[n] + Bu[n] \end{aligned}$$

我们从$S[N] = Q$开始反向求解Riccati difference equation

$$S[n-1] = Q + A^T S[n] A - A^T S[n] B (R + B^T S[n] B)^{-1} B^T S[n] A$$

然后每一个时刻的最优控制为

$$u[n]^* = -R^{-1} B^T S[n] x[n] = -K[n] x[n]$$

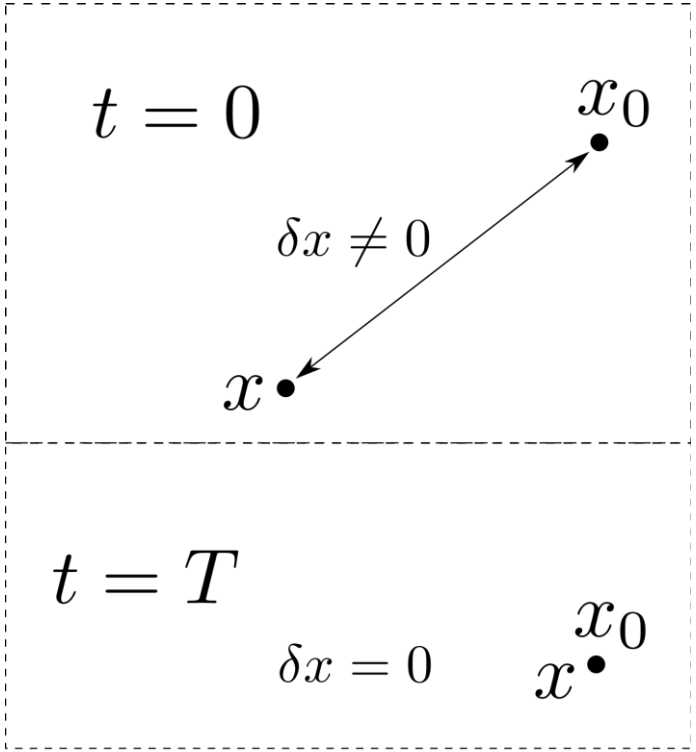
离散时间有穷时域线性二次型调节器

总结一下刚才进行线性化和离散化的过程

把$\delta x = x - x_0$ 当做我们要研究的状态x, $\delta u = u - u_0$为控制量u，然后令$A = \left.\frac{\partial f}{\partial x}\right|_{x=x_0}$ 且$B = \left.\frac{\partial f}{\partial u}\right|_{u=u_0}$，则

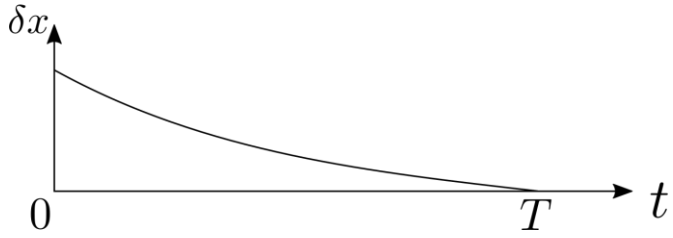
$$\dot{\delta x} = A\delta x + B\delta u$$

我们希望通过控制δu,使得δx在T时间后趋于0，且x和u都不偏离x_0和u_0太多



把δ符号带着，再写一下离散时间有穷时域LQR

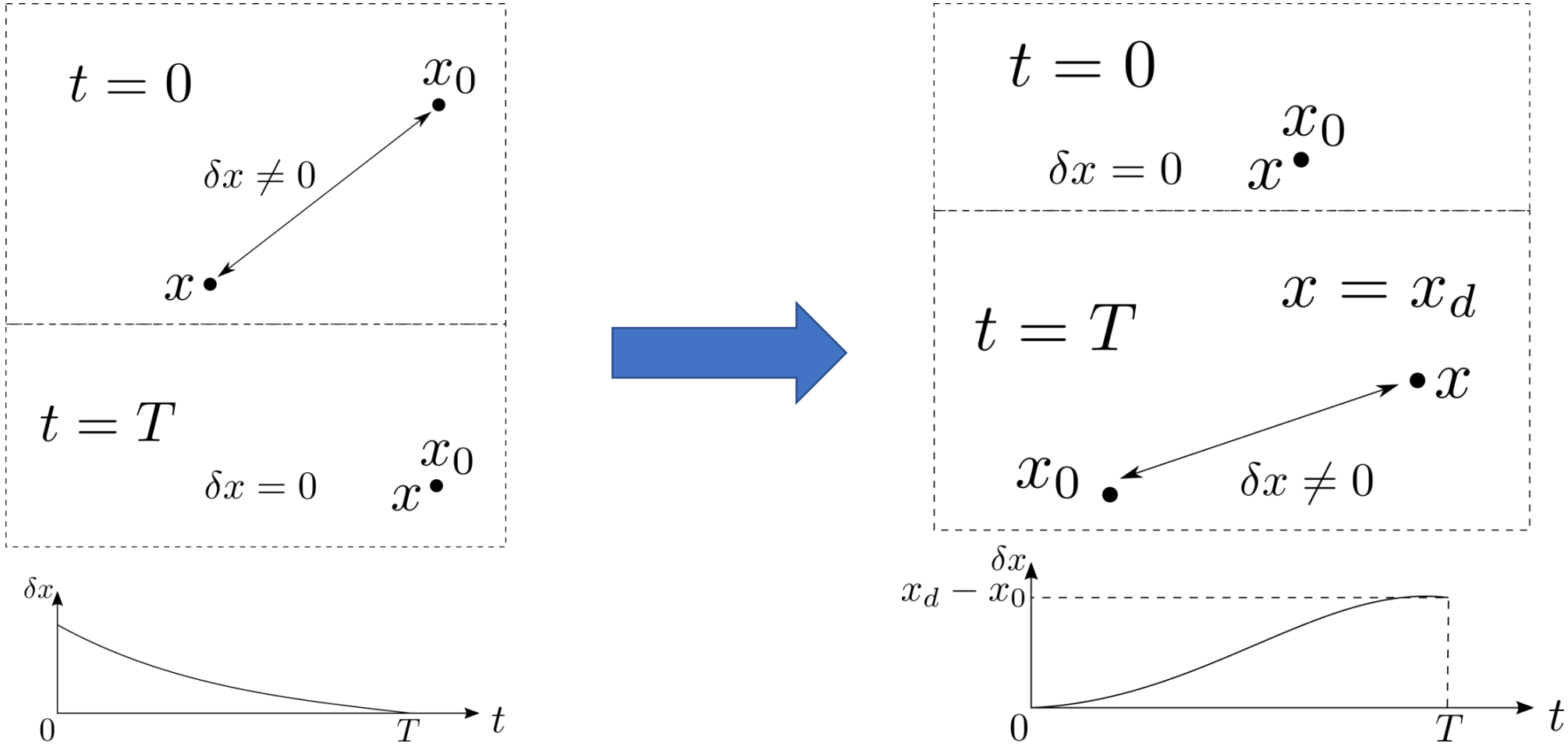
$$\begin{aligned} \min_{\delta u} \quad & J = \sum_{n=0}^{N} \delta x[n]^T Q \delta x[n] + \delta u[n]^T R \delta u[n] \\ s.t. \quad & \delta x[n+1] = A\delta x[n] + B\delta u[n] \\ & \delta x(0) = \delta x_0 \end{aligned}$$



Differential Dynamic Programming
微分动态规划

离散线性二次型调节器和轨迹规划

我们基于上一节的离散线性二次型调节器定义一个类似的问题



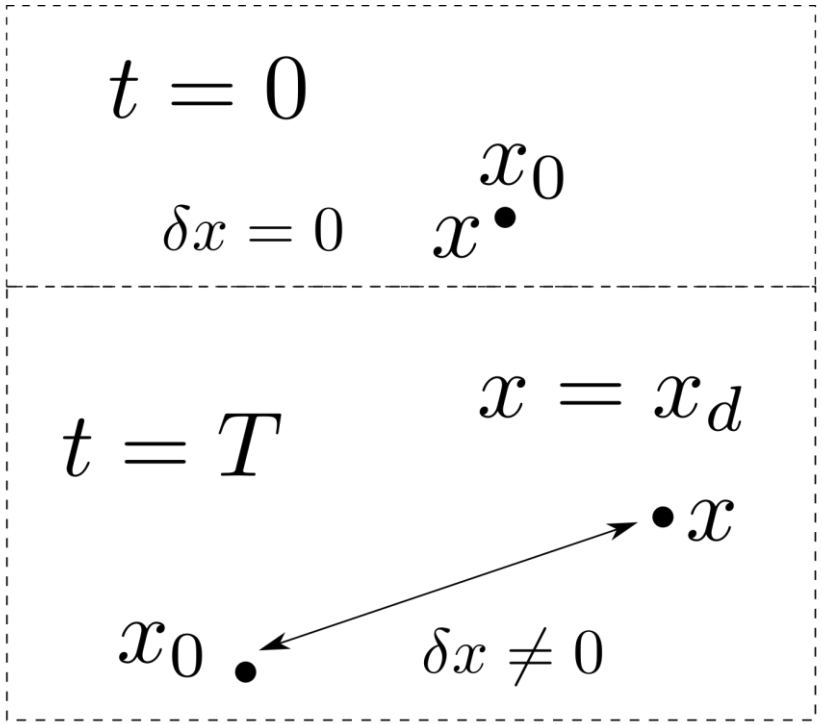
离散线性二次型调节器和轨迹规划

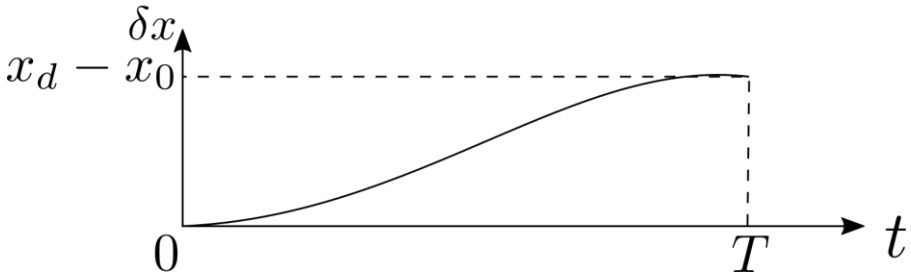
我们基于上一节的离散线性二次型调节器定义一个类似的问题

$$\begin{aligned}\min_{\delta u} \quad & J = \sum_{n=0}^{N} \delta x[n]^T Q \delta x[n] + \delta u[n]^T R \delta u[n] \\ s.t. \quad & \delta x[n+1] = A\delta x[n] + B\delta u[n] \\ & \delta x(0) = \delta x_0\end{aligned}$$

$$\begin{aligned}\min_{\delta u} \quad & J = (\delta x[N] - \delta x_d)^T Q_f (\delta x[N] - \delta x_d) + \sum_{n=0}^{N-1} \delta x[n]^T Q \delta x[n] + \delta u[n]^T R \delta u[n] \\ s.t. \quad & \delta x[n+1] = A\delta x[n] + B\delta u[n] \\ & \delta x(0) = 0\end{aligned}$$

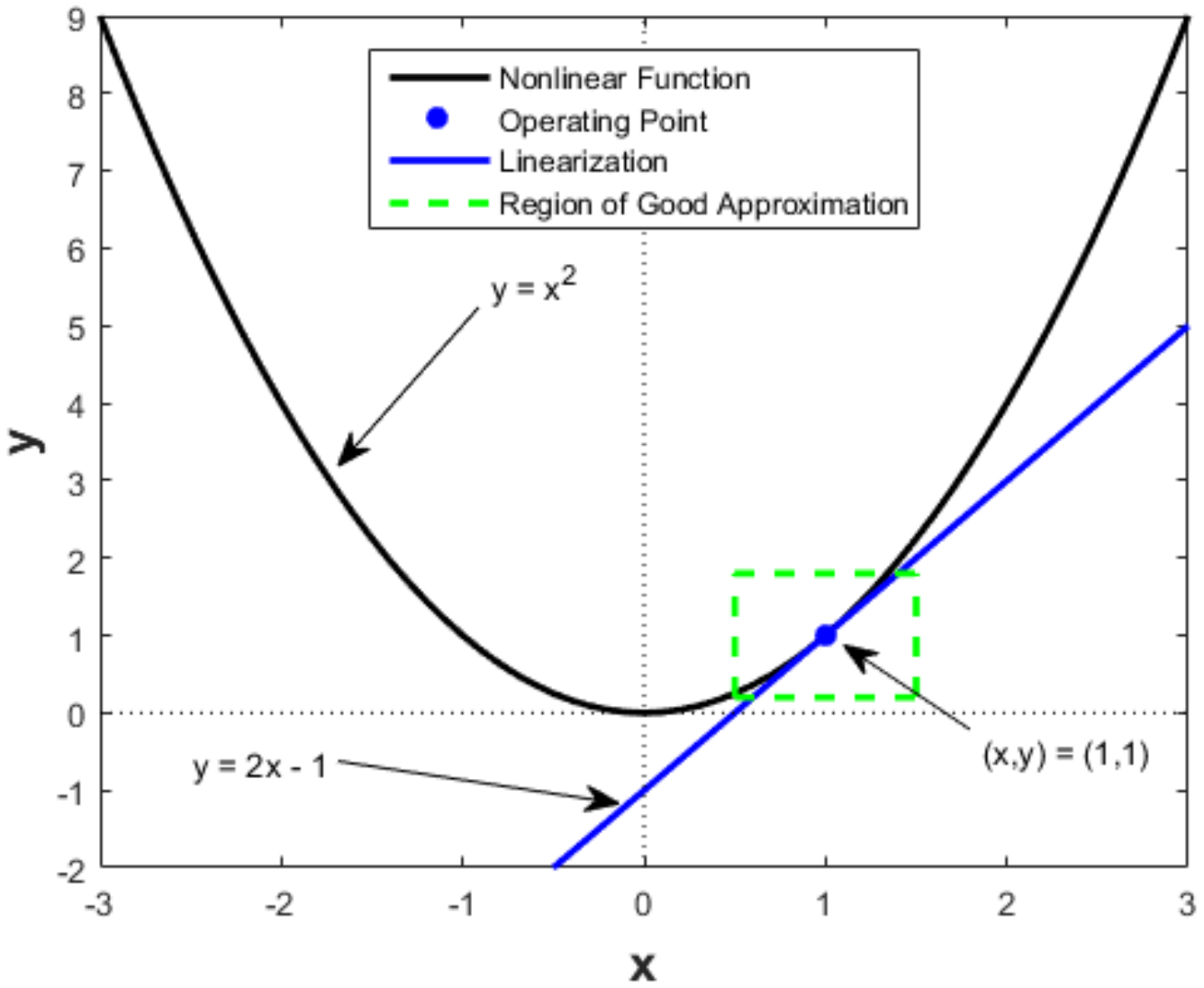
其中$\delta x_d) = x_d - x_0$, Q_f 是远大于Q的正定矩阵





线性化的精度

- 非线性系统的线性化精度依赖于参考点
- 偏离参考点太远则线性化的方程会不准确



https://www.mathworks.com/help/slcontrol/ug/linearizing-nonlinear-models.html

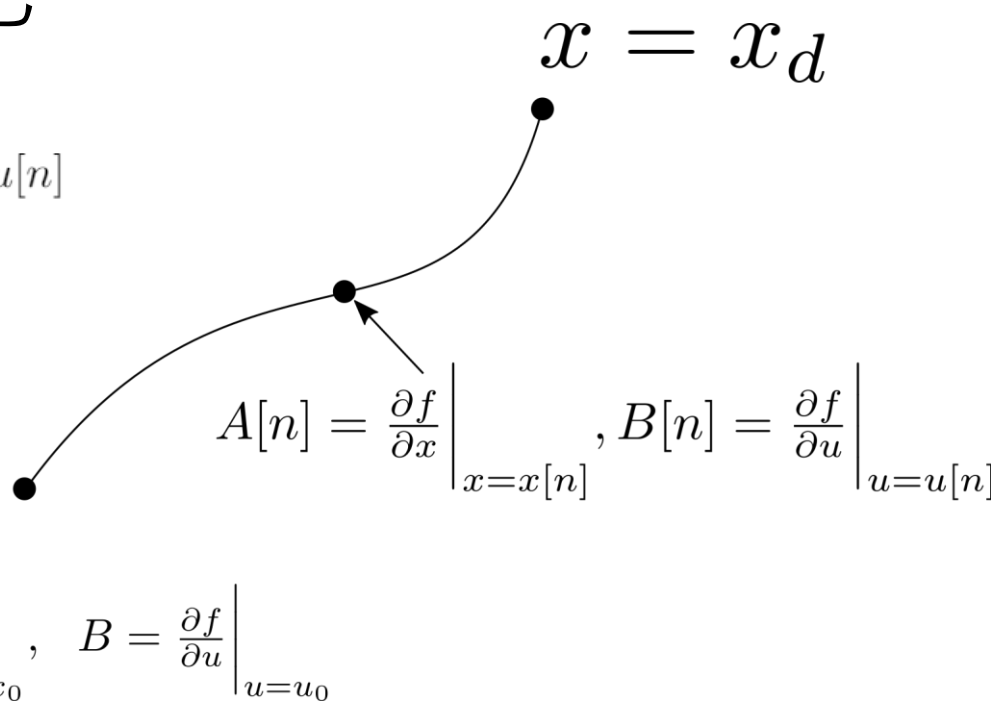
需要沿轨迹进行重复线性化

$$\min_{\delta u} \quad J = (\delta x[N] - \delta x_d)^T Q_f (\delta x[N] - \delta x_d) + \sum_{n=0}^{N-1} \delta x[n]^T Q \delta x[n] + \delta u[n]^T R \delta u[n]$$

$$s.t. \quad \delta x[n+1] = A\delta x[n] + B\delta u[n]$$

$$\delta x(0) = 0$$

$$A = \frac{\partial f}{\partial x}\bigg|_{x=x_0}, \quad B = \frac{\partial f}{\partial u}\bigg|_{u=u_0}$$



$$\min_{\delta u} \quad J = (\delta x[N] - \delta x_d)^T Q_f (\delta x[N] - \delta x_d) + \sum_{n=0}^{N-1} \delta x[n]^T Q \delta x[n] + \delta u[n]^T R \delta u[n]$$

$$s.t. \quad \delta x[n+1] = A[n]\delta x[n] + B[n]\delta u[n]$$

$$\delta x(0) = 0$$

$$A[n] = \frac{\partial f}{\partial x}\bigg|_{x=x[n]}, B[n] = \frac{\partial f}{\partial u}\bigg|_{u=u[n]}$$

三个基本等价的术语

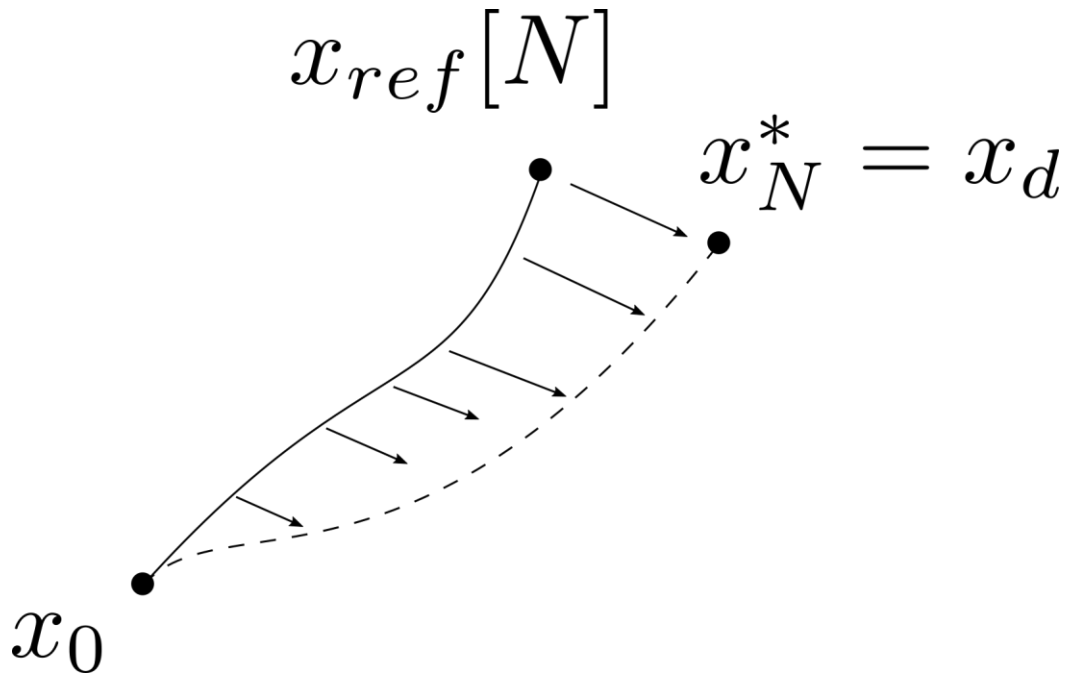
- Differential Dynamic Programming (DDP)
- Sequential Linear Quadratic (SLQ)
- **Iterative Linear Quadratic Regular (iLQR)**

iLQR

对于一个非线性系统，我们希望规划它的轨迹使得它从某个起始状态运动到某个目标状态，并且我们希望得到反馈控制器

我们首先根据经验和对模型的理解设定一个初始控制器，然后用这个控制器生成初始的轨迹

接着重复进行沿轨迹线性化、解LQR、用LQR的解更新初始的控制器的过程



Mayne, David. "A second-order gradient method for determining optimal trajectories of non-linear discrete-time systems." *International Journal of Control* 3.1 (1966): 85-95.

Li, Weiwei, and Emanuel Todorov. "Iterative linear quadratic regulator design for nonlinear biological movement systems." *ICINCO (1)*. 2004.

iLQR

初始控制器$u[n] = u_{ff}[n] + K[n]x[n]$, 非线性的系统动力学方程$\dot{x} = f(x, u)$, 初始状态$x(0) = x_0$, 代价函数$J = (x - x_N^*)^T Q_f (x - x_N^*) + \sum x^T Q x + u^T R u$.

生成初始轨迹：用$u[n]$从x_0开始计算系统动力学方程N步

$$x_{ref}[0], \quad x_{ref}[1], \quad \dots \quad x_{ref}[N]$$

$$u_{ref}[0], \quad u_{ref}[1], \quad \dots \quad u_{ref}[N]$$

循环：沿标称轨迹线性化系统动力学方程$A[n] = \frac{\partial f}{\partial x}\Big|_{x=x_{ref}(n)}, B[n] = \frac{\partial f}{\partial u}\Big|_{u=u_{ref}(n)}$.

构建局部LQR问题：定义$\bar{x} = x - x_{ref}$, $\bar{u} = u - u_{ref}$, $\bar{x}_N^* = x_N^* - x_{ref}(N)$.

$$\min_u \quad \bar{J} = (\bar{x}(N) - \bar{x}_N^*)^T Q_f (\bar{x}(N) - \bar{x}_N^*) + \sum_{i=0}^{N-1} \bar{x}^T Q \bar{x} + \bar{u}^T R \bar{u}$$

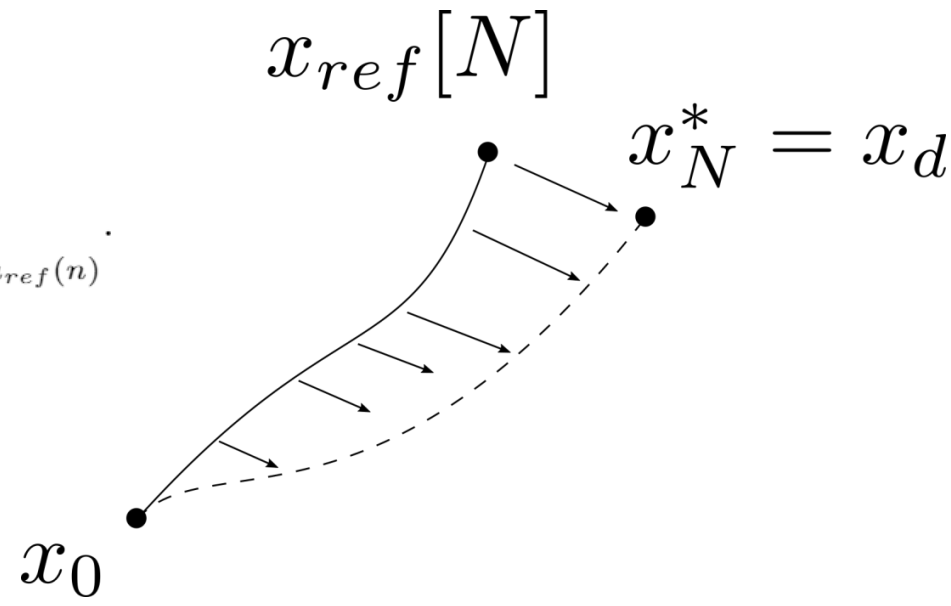
$$\text{s.t.} \quad \dot{\bar{x}}(n) = A(n)\bar{x}(n) + B(n)\bar{u}(n), \bar{x}(0) = 0$$

局部LQR问题的解为

$$\bar{u}(n) = -R^{-1}B^T \mathbf{S}_{xx}(n)\bar{x}(n) - R^{-1}B^T \mathbf{s}_x(n) = \bar{K}(n)\bar{x}(n) + \bar{u}_{ff}(n)$$

更新初始控制器

$$u'[n] = u_{ref}[n] + \alpha \bar{u}_{ff}[n] + \bar{K}[n](x[n] - x_{ref}(n))$$



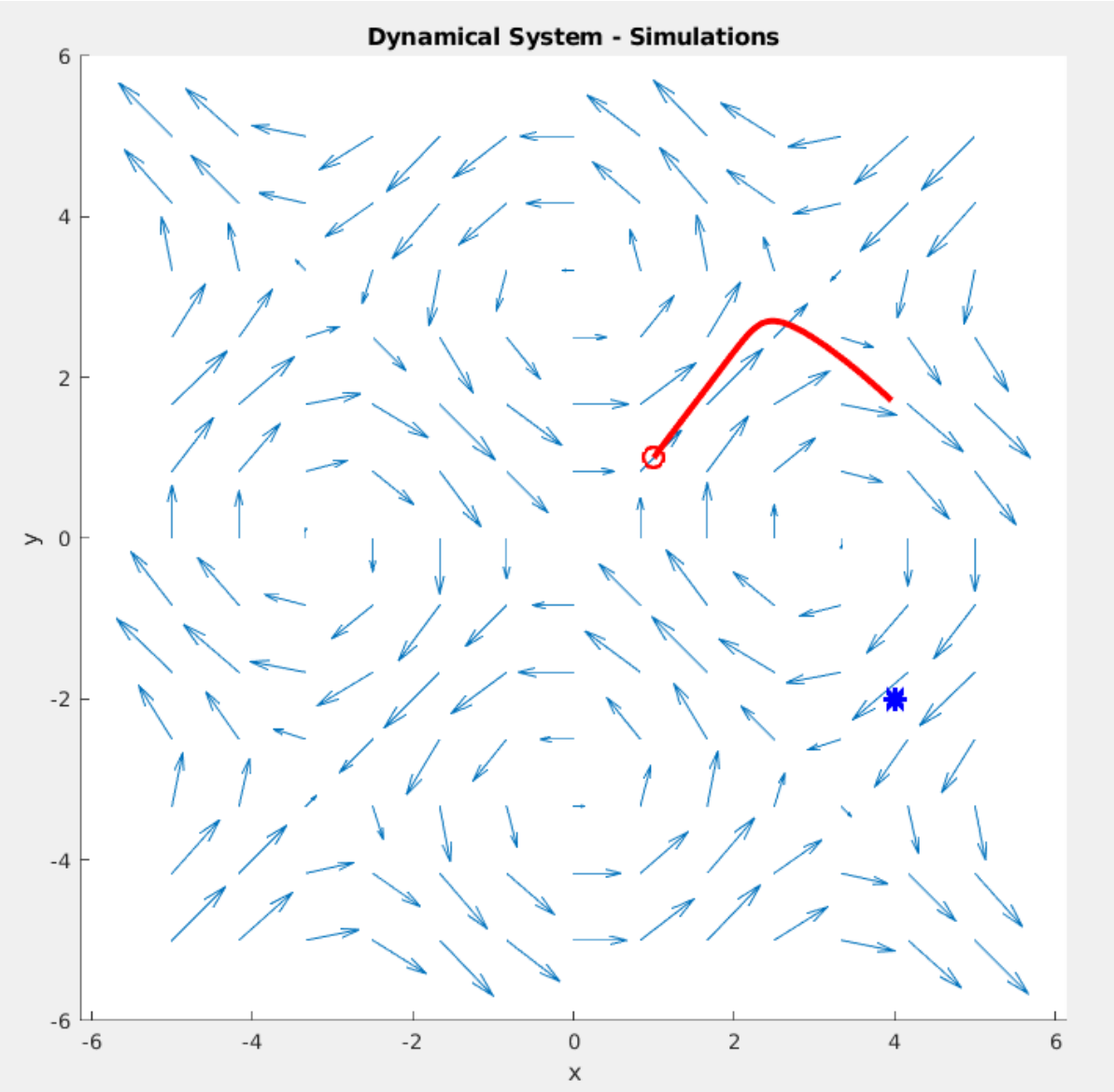
iLQR 简单的例子

State dimenison 2, control dimension 1.

$$\begin{bmatrix}\dot{x}\\ \dot{y}\end{bmatrix} = \begin{bmatrix}sin(y)+1+cos(u)\\ cos(x)+sin(u)\end{bmatrix}$$

Initial state $(1,1)$, target state $(4,-2)$.
Start with a initial open loop controller

$$u(t) = -\sin(0.4t-0.1)+0.6\cos(0.9t+0.1)$$



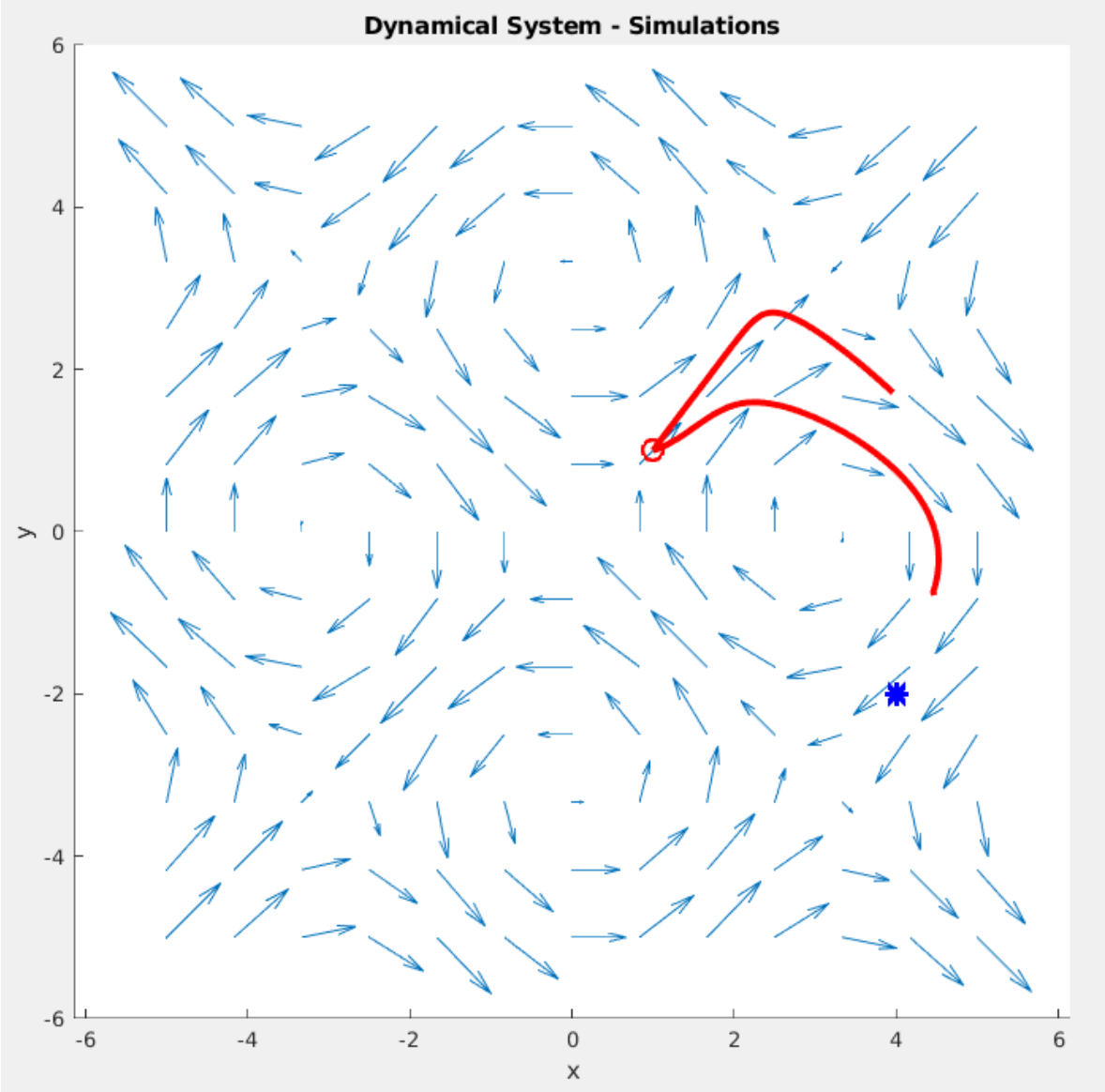
iLQR 简单的例子

State dimenison 2, control dimension 1.

$$\begin{bmatrix}\dot{x}\\ \dot{y}\end{bmatrix} = \begin{bmatrix}sin(y)+1+cos(u)\\ cos(x)+sin(u)\end{bmatrix}$$

Initial state $(1,1)$, target state $(4,-2)$.
Start with a initial open loop controller

$$u(t) = -\sin(0.4t-0.1)+0.6\cos(0.9t+0.1)$$



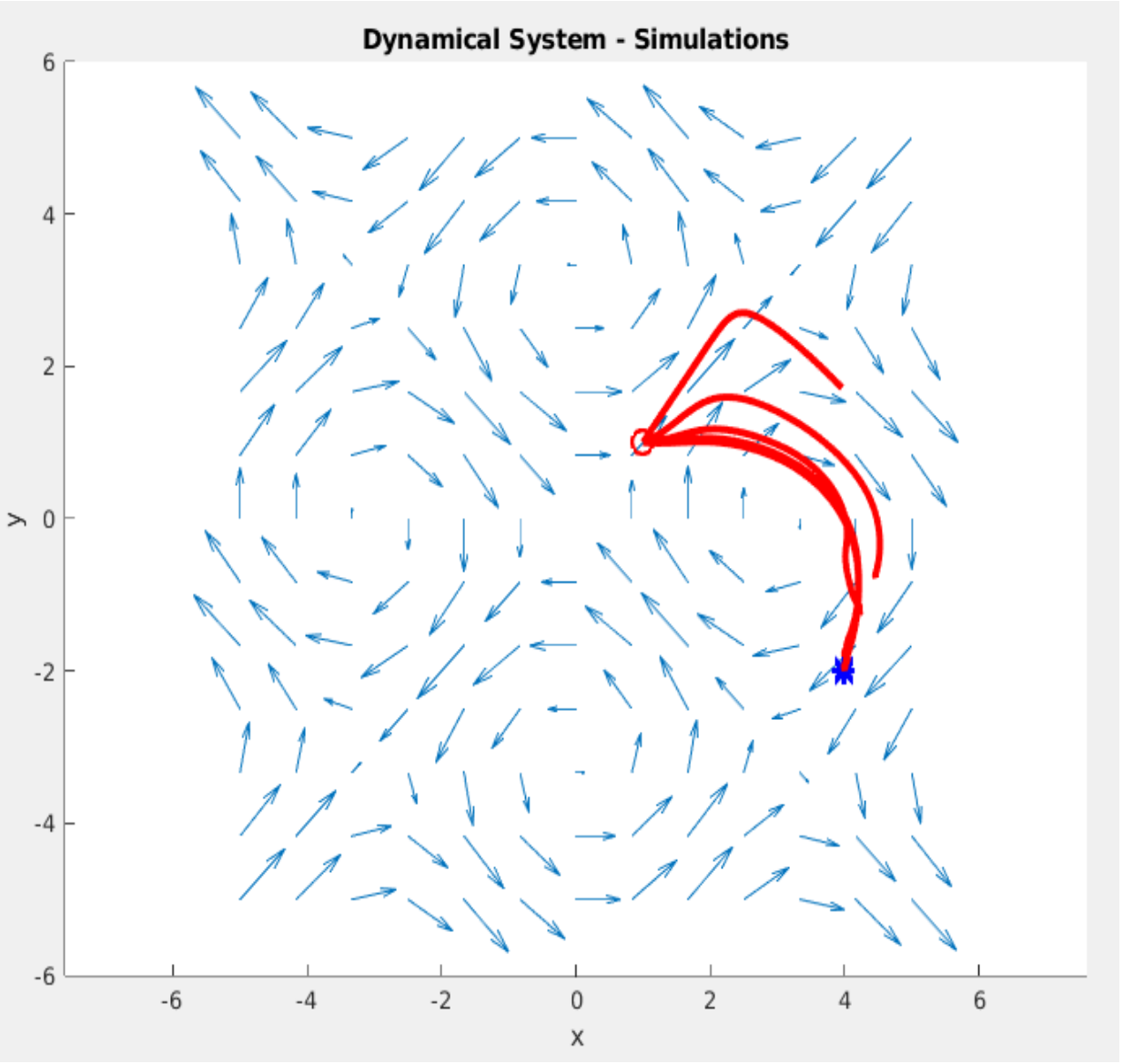
iLQR 简单的例子

State dimenison 2, control dimension 1.

$$\begin{bmatrix}\dot{x}\\ \dot{y}\end{bmatrix} = \begin{bmatrix}sin(y)+1+cos(u)\\ cos(x)+sin(u)\end{bmatrix}$$

Initial state $(1,1)$, target state $(4,-2)$.
Start with a initial open loop controller

$$u(t) = -\sin(0.4t-0.1)+0.6\cos(0.9t+0.1)$$



iLQR 猴子机器人

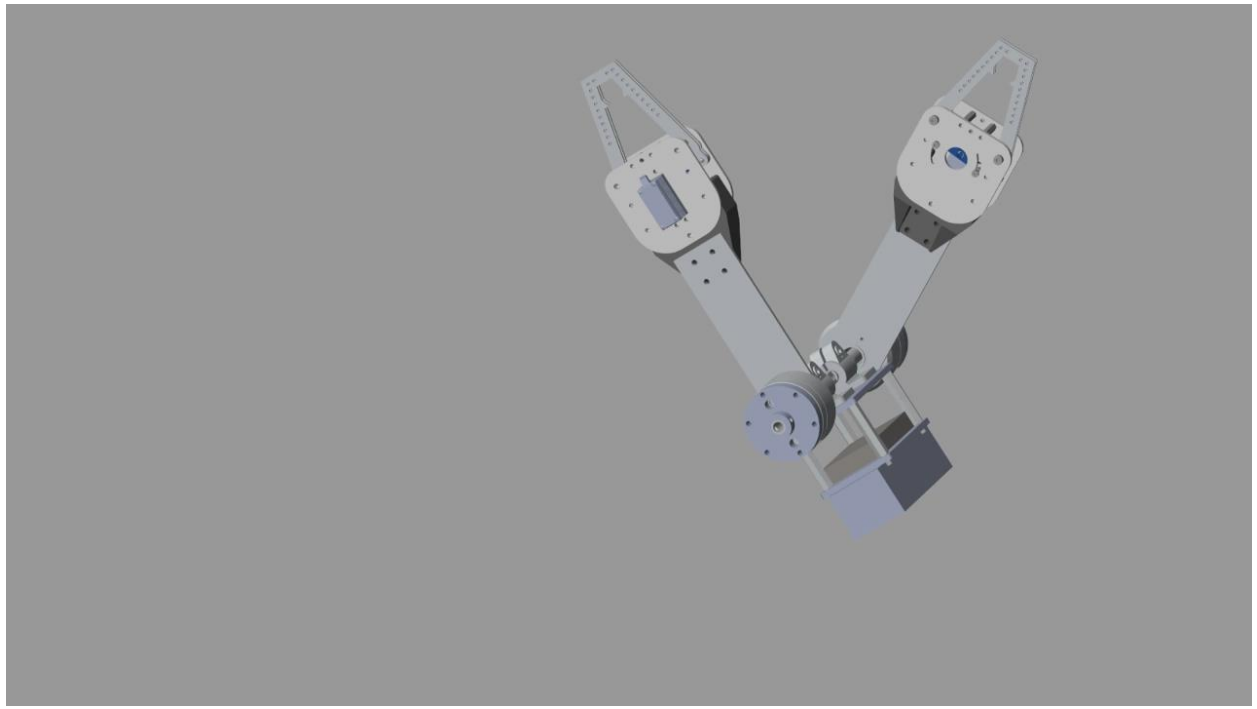
The dynamic model of the robot is

$$M(q)\ddot{q} + C(q,\dot{q}) + G(q) = Bu$$

where

$$q = \begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \\ \ddot{q}_3 \end{bmatrix} \qquad B = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \qquad u = \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix}$$

$$\begin{aligned} \min_{u} \quad & (x_T - x_T^*)^T Q f (x_T - x_T^*) + \sum_{i=0}^{T} u_i^T R u_i \\ \text{subject to} \quad & x_{i+1} = f(x_i, u_i) \ \ \forall i = 0...T \\ & x0 = x0 \end{aligned}$$



iLQR 猴子机器人

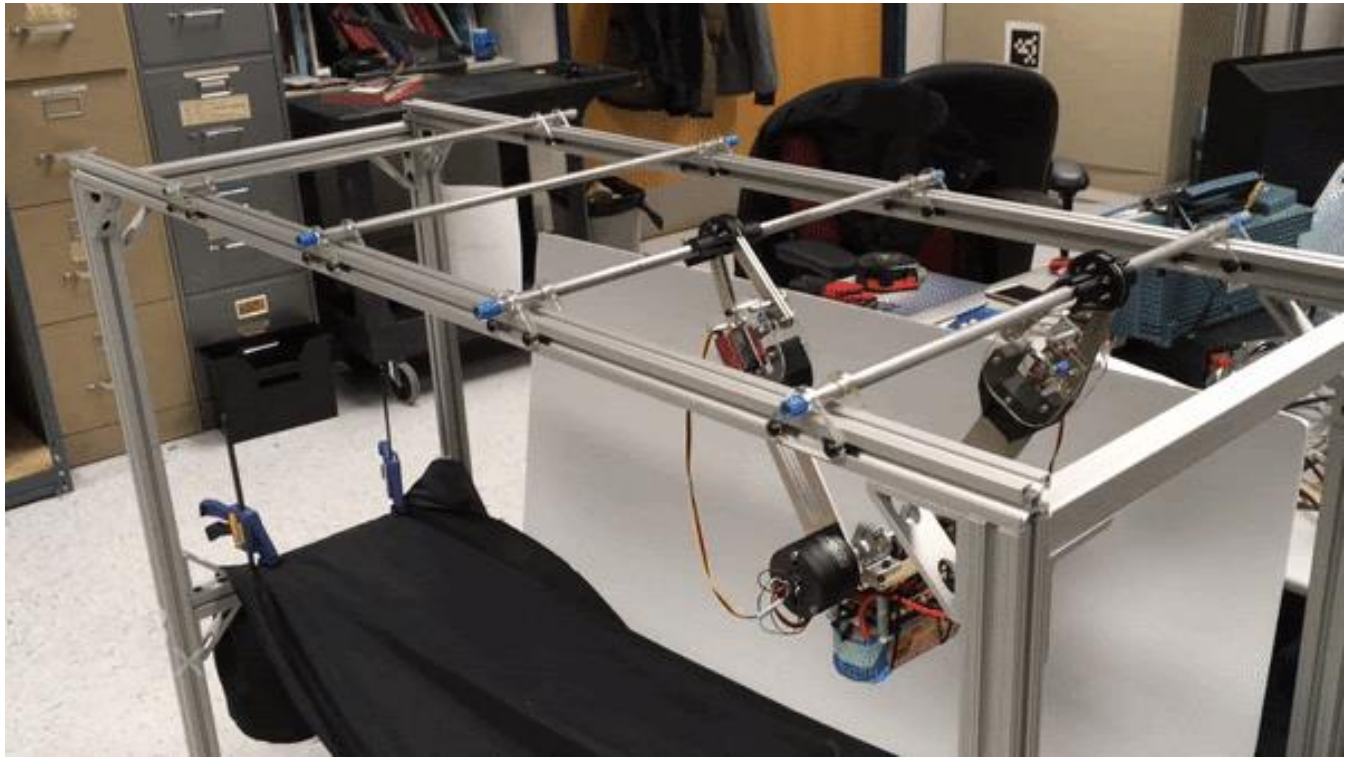
The dynamic model of the robot is

$$M(q)\ddot{q} + C(q, \dot{q}) + G(q) = Bu$$

where

$$q = \begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \\ \ddot{q}_3 \end{bmatrix} \qquad B = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \qquad u = \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix}$$

$$\begin{aligned} \min_{u} \quad & (x_T - x_T^*)^T Qf (x_T - x_T^*) + \sum_{i=0}^{T} u_i^T R u_i \\ \text{subject to} \quad & x_{i+1} = f(x_i, u_i) \ \ \forall i = 0...T \\ & x0 = x0 \end{aligned}$$



微分动态规划

优点

- 能获得最优轨迹，也能获得最优的反馈控制器
- 局部LQR问题的求解可以并行化[1]能达到非常高的求解速度
- 据说（我自己并没有实验验证过），比其他方法有更好的数值精度

[1] Farshidian, Farbod, et al. "Real-time motion planning of legged robots: A model predictive control approach." *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*. IEEE, 2017.

微分动态规划

缺点：

- 需要较好的初始化轨迹
- 比较难考虑额外的约束

在微分动态规划中考虑约束

许多已有的工作在微分动态规划的基础上考虑额外的等式约束或者不等式约束。

通常来说如果想要保持和无约束的微分动态规划相同的计算量和精度，算法需要较大幅度的修改以使得解满足这些额外的约束

通常有两种方法：
1. 在局部LQR求解中考虑线性化后的约束
2. 局部LQR为无约束问题，在用局部LQR的解更新标称控制器的时候考虑约束

[1] Tassa, Yuval, Nicolas Mansard, and Emo Todorov. "Control-limited differential dynamic programming." *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014.

[2] Farshidian, Farbod, et al. "Real-time motion planning of legged robots: A model predictive control approach." *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*. IEEE, 2017.

LQR & 等式约束的LQR

- 微分线性规划的核心是构造局部LQR问题
- 处理额外的等式约束需要将局部LQR问题拓展为等式约束LQR问题（EC-LQR）
- 不等式约束可以通过一些数值优化的方法来化为等式约束
- 等式约束使得Riccati equation的构造加倍困难

$$
\begin{aligned}
\min_{\delta u} \quad & J = \sum_{n=0}^{N} \delta x[n]^T Q \delta x[n] + \delta u[n]^T R \delta u[n] \\
s.t. \quad & \delta x[n+1] = A[n]\delta x[n] + B[n]\delta u[n] \\
& \delta x(0) = \delta x_0
\end{aligned}
$$

$$
\begin{aligned}
\min_{\delta u} \quad & J = \sum_{n=0}^{N} \delta x[n]^T Q \delta x[n] + \delta u[n]^T R \delta u[n] \\
s.t. \quad & \delta x[n+1] = A[n]\delta x[n] + B[n]\delta u[n] \\
& \delta x(0) = \delta x_0 \\
& C[n]x[n] + D[n]u[n] = 0
\end{aligned}
$$

Direct Collocation
直接配点法

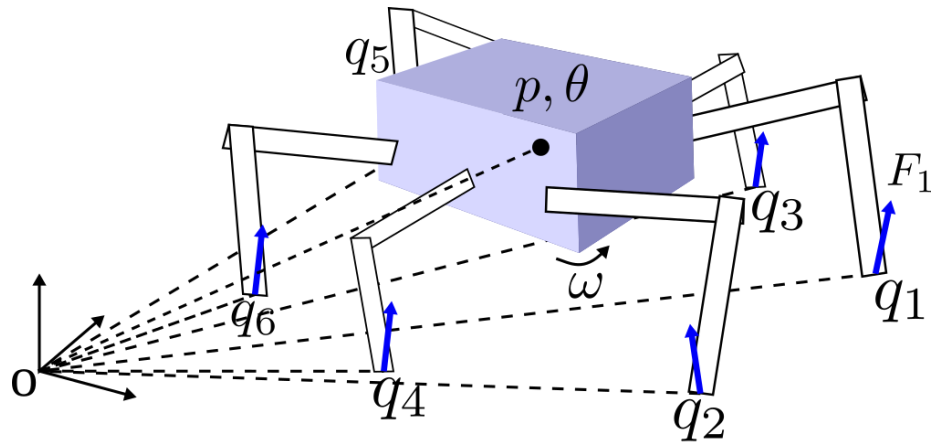
Direct Collocation 直接配点法

- 放弃获得反馈控制器
- 将轨迹上每一时刻的状态和控制量看做一个非线性优化问题的决策变量（decision variable）
- 通过成熟的非线性优化领域的技术来处理约束

$$Y = [x_1, ..., x_N, u_1, ..., u_N]$$

$$\begin{aligned} \min_{Y} \quad & J(Y) \\ \text{s.t.} \quad & G(Y) \leq 0 \\ & H(Y) = 0 \\ & x_1 = x_s, \quad x_N = x_d \end{aligned}$$

直接配点法 – 多足机器人



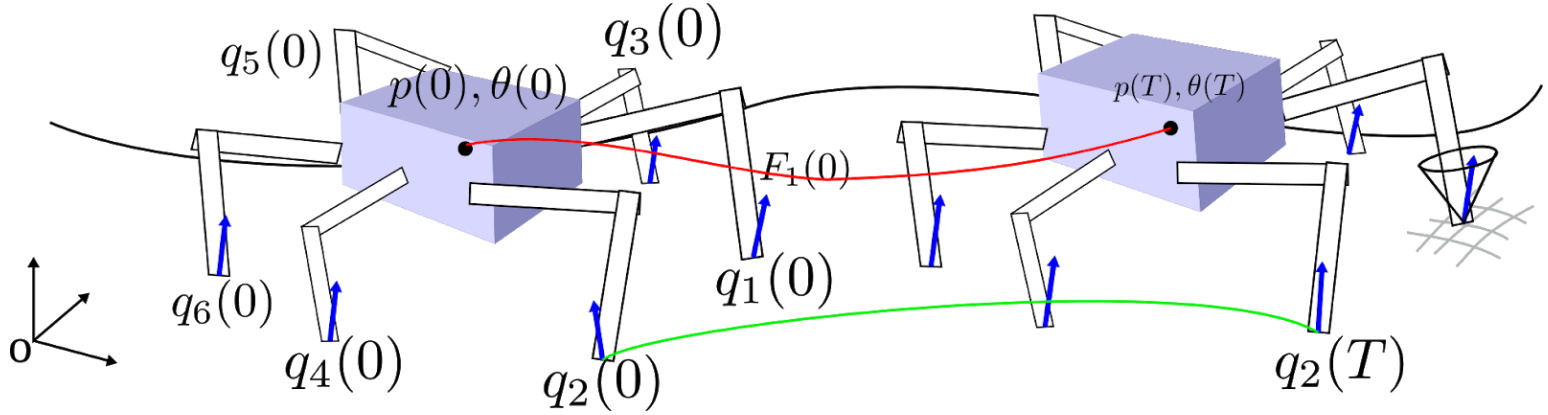
$$x = [p \quad \theta \quad \dot{p} \quad \omega \quad q_1 \quad q_2 \quad q_3 \quad q_4 \quad q_5 \quad q_6]^T$$

$$u = [F_1 \ldots F_6 \quad \dot{q}_1 \ldots \dot{q}_6]^T$$

- n is the number of legs
- $p, \dot{p} \in \mathbb{R}^3$ are the COM position and velocity in world frame
- $\theta \in \mathbb{R}^3$ is the COM orientation in world frame
- ω is the angular velocity of the robot chassis
- $q_i \in \mathbb{R}^3, i = 1 \ldots n$ are foot positions of the robot in world frame
- $\dot{q}_i \in \mathbb{R}^3, i = 1 \ldots n$ are foot velocities
- $F_i \in \mathbb{R}^3, i = 1 \ldots n$ are ground reaction forces in world frame

- Winkler, Alexander W., et al. "Gait and trajectory optimization for legged systems through phase-based end-effector parameterization." *IEEE Robotics and Automation Letters* 3.3 (2018): 1560-1567.

直接配点法
多足机器人



把机器人在两点之间的运动轨迹全都看做决策变量

$$x_n = [p(n) \;\; \theta(n) \;\; \dot{p}(n) \;\; \omega(n) \;\; q_1(n) \;\; q_2(n) \;\; q_3(n) \;\; q_4(n) \;\; q_5(n) \;\; q_6(n)]^T$$

$$u_n = [F_1(n) \ldots F_6(n) \;\; \dot{q}_1(n) \ldots \dot{q}_6(n)]^T, n = 1 \ldots N$$

构建非线性优化问题，将运动的约束表示为优化问题的等式约束或者不等式约束

$$\begin{aligned}
\min_{Y} \quad & J(Y) \\
\text{s.t.} \quad & x_1 = x(0), \;\; x_N = x(T) \\
& C_{i,n}F_{i,n} \leq d_{i,n}, \text{ for } i = 1 \ldots 6, n = 1 \ldots N \\
& q_i(n) = constant, \dot{q}_i(n) = 0, \text{for stance legs} \\
& F_i(n) = 0, \text{for swing legs} \\
& x_{n+1} = x_n + \dot{x}_n \Delta t, \text{ for } n = 1 \ldots N-1
\end{aligned}$$

直接配点法 - 多足机器人 - 约束介绍1

机器人的起始姿态和终止姿态是给定的

这两个姿态由其他的基于地形的优化算法得到

$$x_n = [p(n) \;\; \theta(n) \;\; \dot{p}(n) \;\; \omega(n) \;\; q_1(n) \;\; q_2(n) \;\; q_3(n) \;\; q_4(n) \;\; q_5(n) \;\; q_6(n)]^T$$

$$u_n = [F_1(n) \ldots F_6(n) \;\; \dot{q}_1(n) \ldots \dot{q}_6(n)]^T, n = 1 \ldots N$$

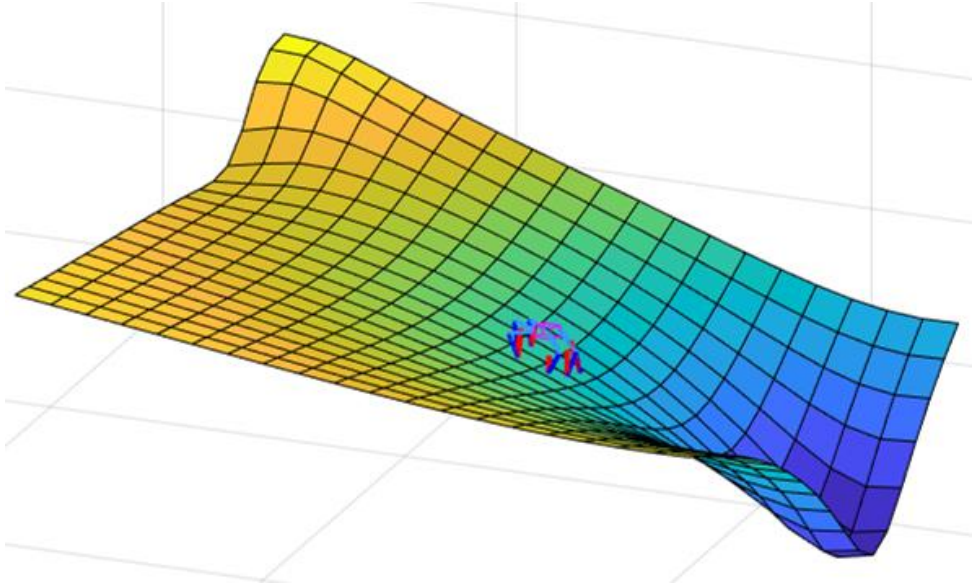
$$Y = [x_1, \ldots, x_N, u_1, \ldots, u_N]$$

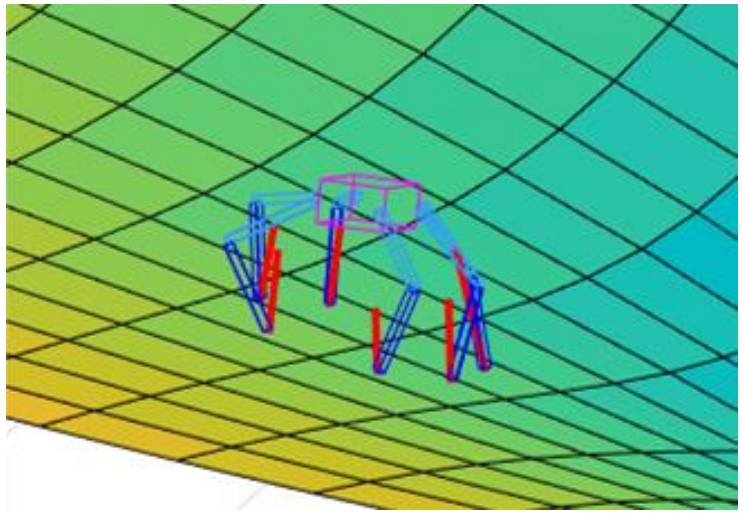
$$\min_{Y} \quad J(Y)$$

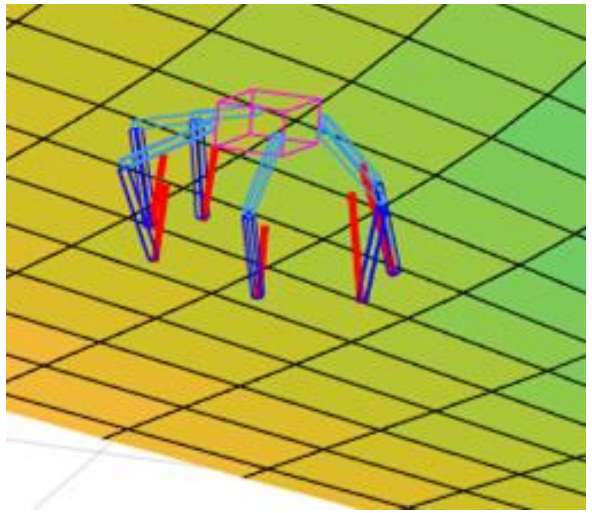
$$\text{s.t.} \quad x_1 = x(0), \;\; x_N = x(T)$$

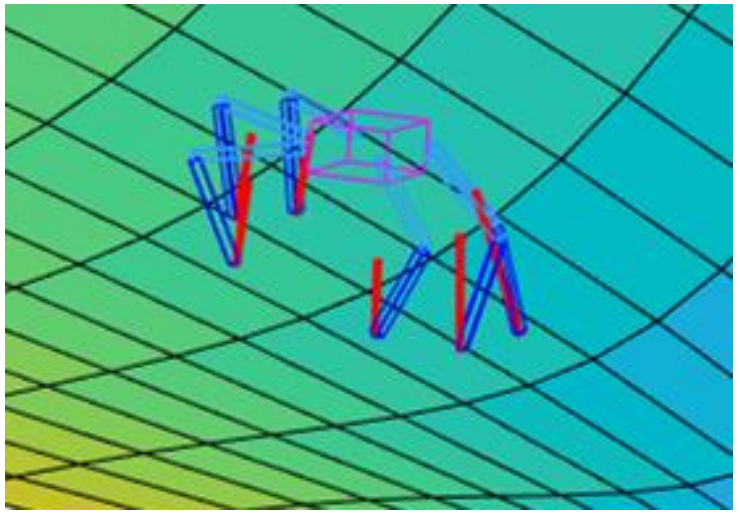
起始和终止姿态的选择

给定机器人周围的地形图，优化在地形图上的站立姿态









在地形图上不同位置的站姿

直接配点法 - 多足机器人 - 约束介绍1

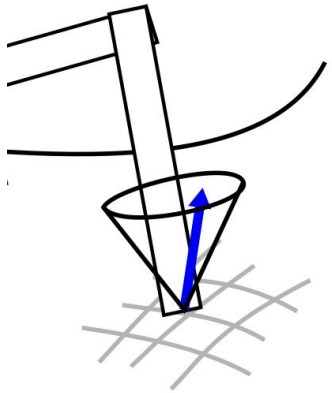
足尖受到的地面作用力（或者说机器人施加给地面的推力）必须处在一个摩擦锥内，这样才能避免足端打滑

$$Y = [x_1, \ldots, x_N, u_1, \ldots, u_N]$$

$$\min_{Y} \quad J(Y)$$

$$\text{s.t.} \quad x_1 = x(0), \quad x_N = x(T)$$

$$C_{i,n} F_{i,n} \leq d_{i,n}, \text{for } i = 1 \ldots 6, n = 1 \ldots N$$



直接配点法 - 多足机器人 - 约束介绍3

机器人运动过程中，足在swing和stance两个模式间切换

Swing mode：足在空中移动，足上的受力必为0

Stance mode：足支撑在地面上，足的运动速度必为0

这两个模式导致了足在不同时刻受到不同的约束

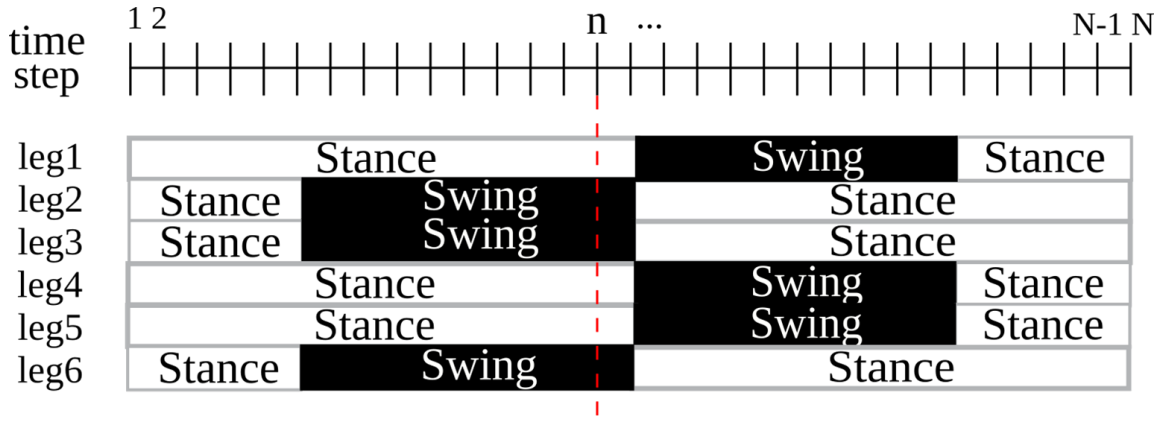
$$\min_{Y} \quad J(Y)$$

$$\text{s.t.} \quad x_1 = x(0), \; x_N = x(T)$$

$$C_{i,n}F_{i,n} \leq d_{i,n}, \text{ for } i = 1 \ldots 6, n = 1 \ldots N$$

$$\dot{q}_i(n) = 0, \text{for stance legs}$$

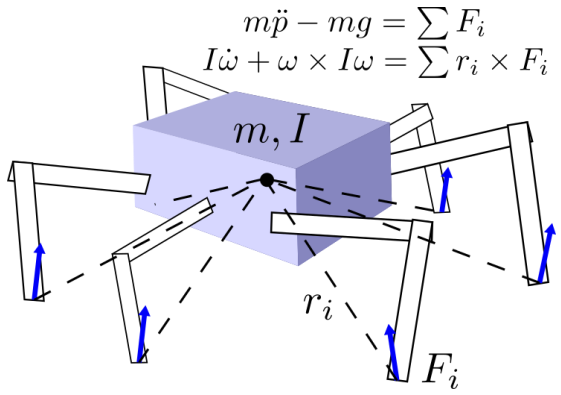
$$F_i(n) = 0, \text{for swing legs}$$



For stance leg i, $q_i(n) = constant$ and $\dot{q}_i(n) = 0$.
For swing leg j, $F_i(n) = 0$.

直接配点法 - 多足机器人 – 约束介绍3

- 直接配点法中最重要的约束
- 对于运动轨迹上相邻的两个点，两者的差必然等于动力学方程在两个时刻之间的积分量



System dynamics

$$\dot{x} = \begin{bmatrix} \dot{p} \\ \dot{\theta} \\ \ddot{p} \\ \dot{\omega} \\ \dot{q}_i \end{bmatrix} = \begin{bmatrix} \dot{p} \\ B\omega \\ g + \frac{1}{m}\sum c_i F_i \\ I^{-1}(-\omega \times I\omega + \sum r_{f_i}(p, q_i) \times c_i F_i) \\ \dot{q}_i \end{bmatrix} = f(x, u)$$

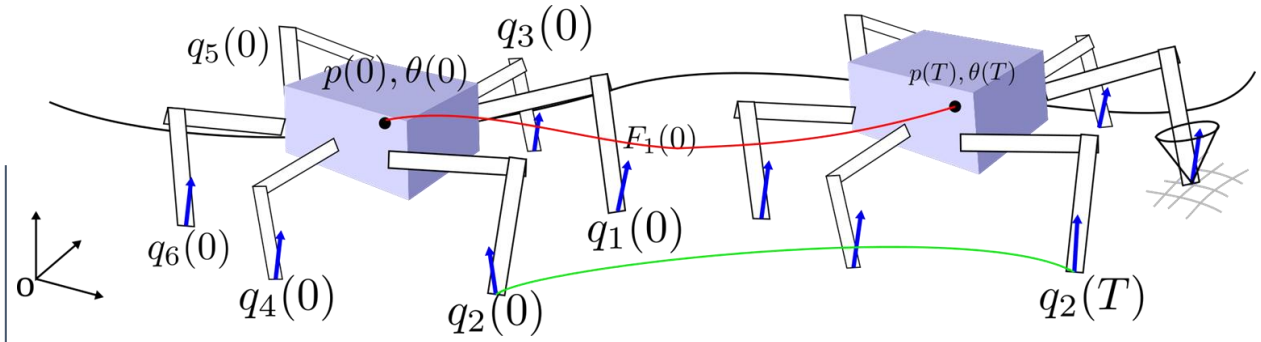
$$x(n+1) = x(n) + \dot{x}_n \Delta t$$

Dai, Hongkai, Andrés Valenzuela, and Russ Tedrake. "Whole-body motion planning with centroidal dynamics and full kinematics." *2014 IEEE-RAS International Conference on Humanoid Robots*. IEEE, 2014.

直接配点法 –完整的问题

这个问题只作为概念介绍，实际不能用这个形式来求解

对于六足机器人，当 N=200时，决策变量的维度为13200



$$x_n = [p(n) \;\; \theta(n) \;\; \dot{p}(n) \;\; \omega(n) \;\; q_1(n) \;\; q_2(n) \;\; q_3(n) \;\; q_4(n) \;\; q_5(n) \;\; q_6(n)]^T$$

$$u_n = [F_1(n) \ldots F_6(n) \;\; \dot{q}_1(n) \ldots \dot{q}_6(n)]^T, n = 1 \ldots N$$

$$Y = [x_1, ..., x_N, u_1, ..., u_N]^T$$

$$\begin{aligned} \min_{Y} \quad & J(Y) \\ \text{s.t.} \quad & x_1 = x(0), \;\; x_N = x(T) \\ & C_{i,n}F_{i,n} \leq d_{i,n}, \text{ for } i = 1 \ldots 6, n = 1 \ldots N \\ & q_i(n) = constant, \dot{q}_i(n) = 0, \text{for stance legs} \\ & F_i(n) = 0, \text{for swing legs} \\ & x_{n+1} = x_n + \dot{x}_n \Delta t, \text{ for } n = 1 \ldots N-1 \end{aligned}$$

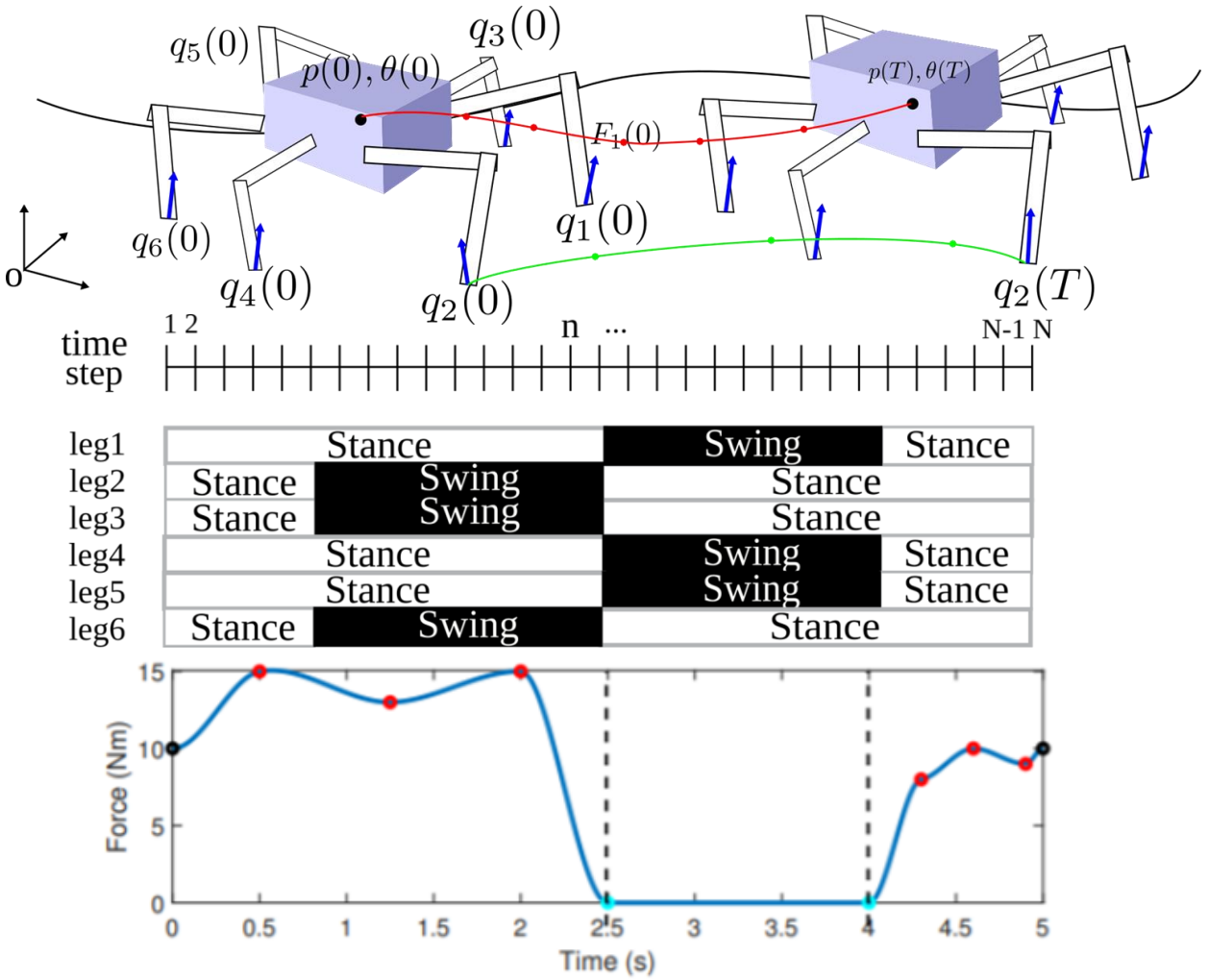
分段直接配点法 Phase Based Direct Collocation

把轨迹表示成分段多项式，决策变量不是轨迹上的点，而是多项式的系数

大大减少问题的维度

自然地保证了运动的平滑性

自然地加入了足在swing和stance mode的约束

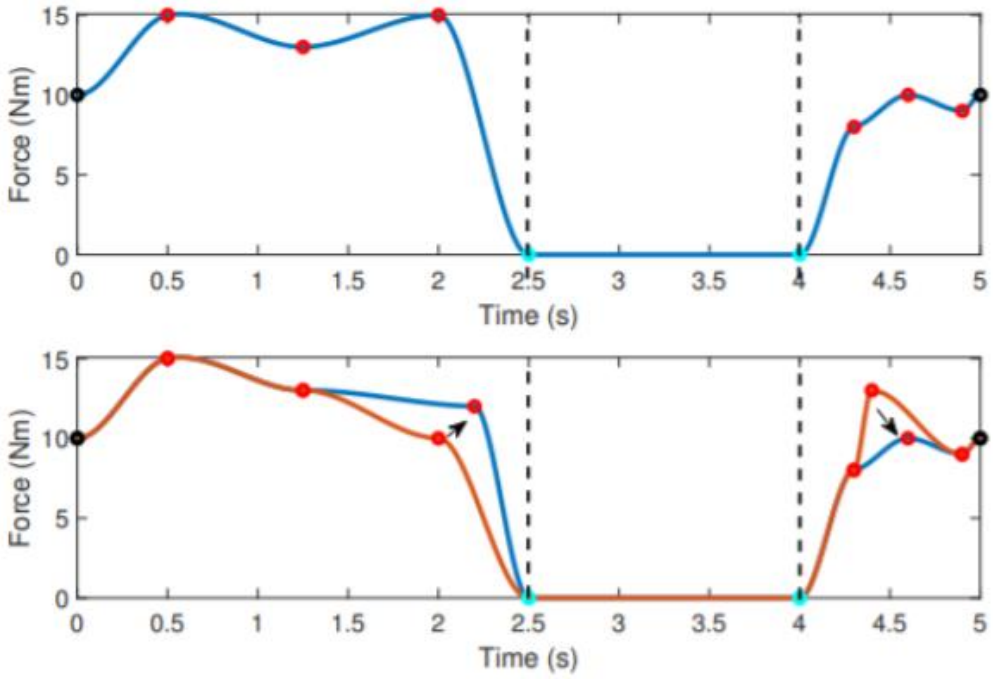


Winkler, Alexander W., et al. "Gait and trajectory optimization for legged systems through phase-based end-effector parameterization." *IEEE Robotics and Automation Letters* 3.3 (2018): 1560-1567.

分段直接配点法
Phase Based Direct Collocation

- 每一个轨迹由多项式的起始和终止节点表示

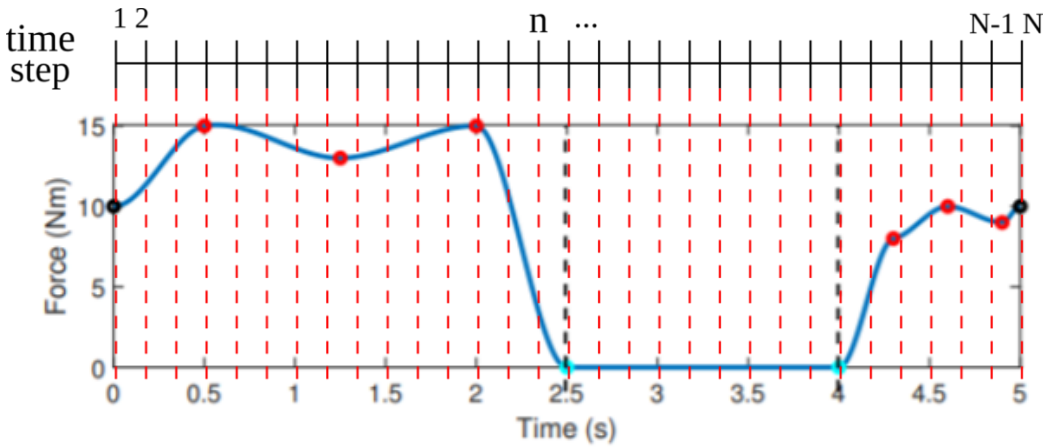
- 优化过程中的决策变量为这些多项式的节点



分段直接配点法 Phase Based Direct Collocation

系统动力学方程的约束只在 N个collocation points上添加

collocation points 的值通过Hermite interpolation得到

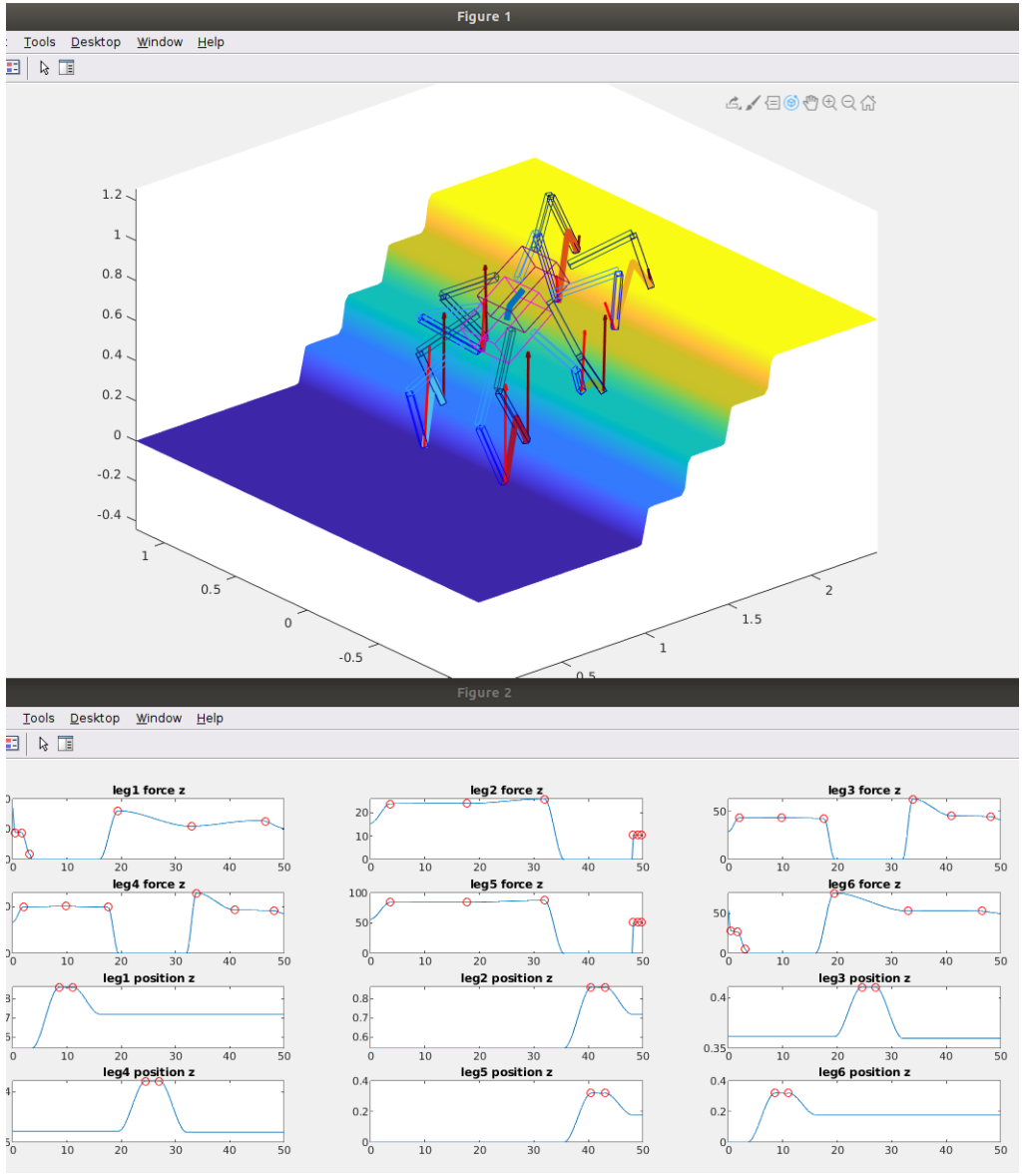


对一条腿i, 在collocation point n, 地面作用力为$F_i(n)$

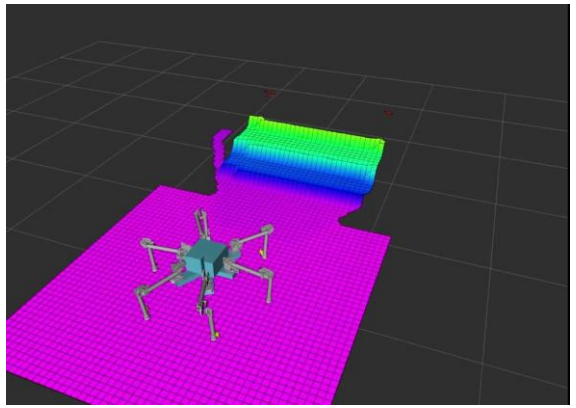
分段直接配点法 Phase Based Direct Collocation

- 在六足机器人中的应用
- 决策变量维度 - 516
- 约束维度 - 180

- 计算时间: 0.1s – 0.4s



系统集成



Elevation Mapping

机器人状态估计

地形图

机器人的姿态等信息

GUI (Rviz) → 生成局部目标 → 选择局部目标上的站姿 → 轨迹规划 → 轨迹控制

用户在地图上点击全局目标点

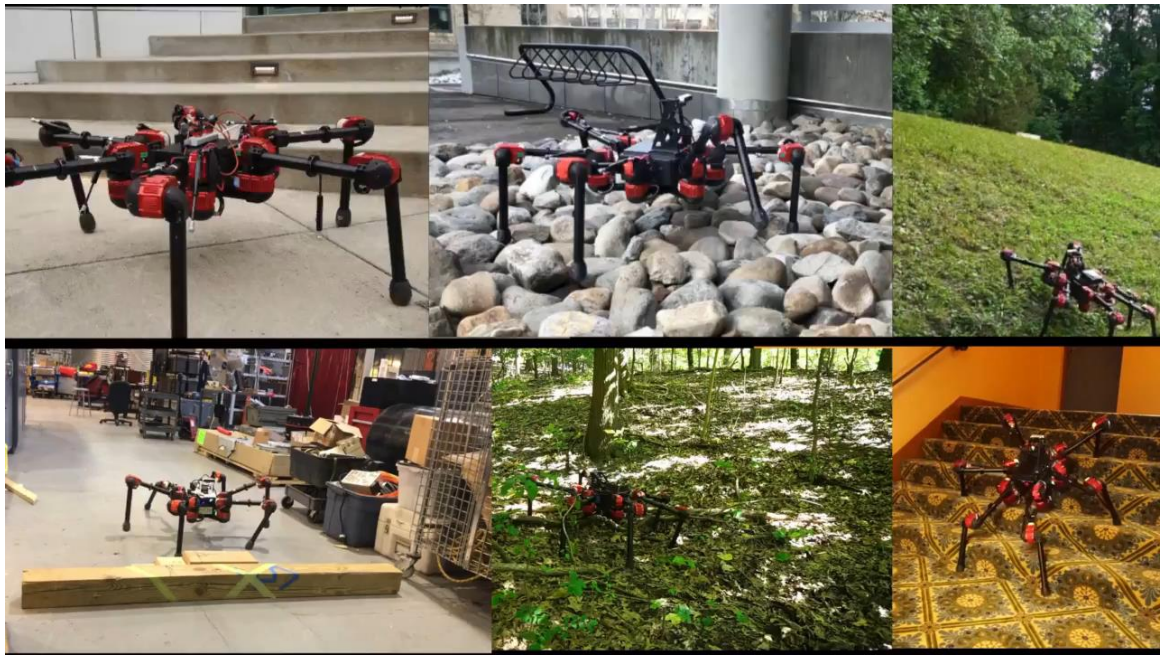
发送目标位置

进行地形图上的站姿优化

直接配点法轨迹规划

将机器人轨迹转化成关节角的控制指令

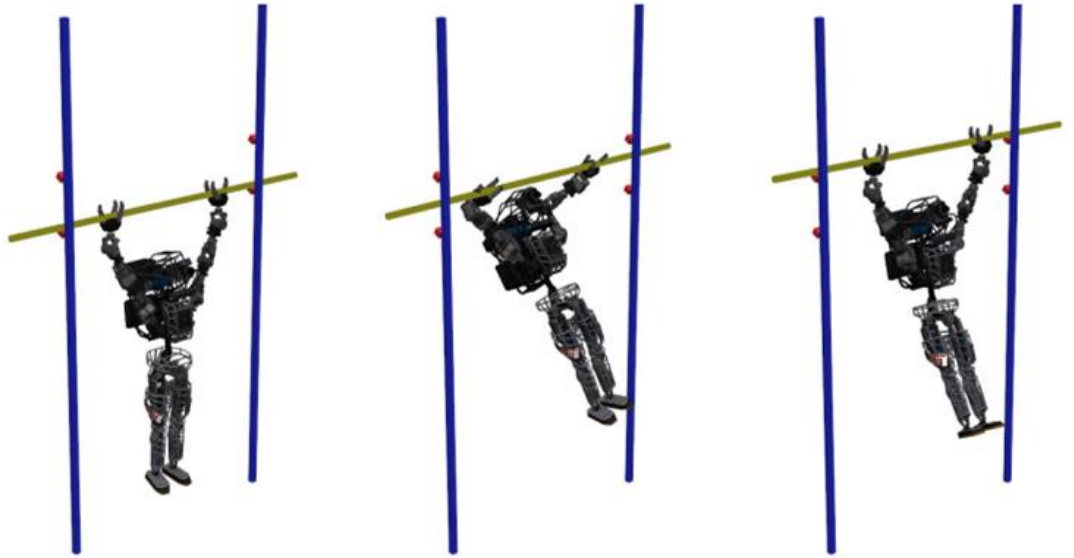
系统集成 – 在复杂地形上的实时运动轨迹优化



直接配点法

优点:

可以处理任意高精度的系统动力学方程



(a) t=0　(b) t=0.8　(c) t=1.41

Dai, Hongkai, Andrés Valenzuela, and Russ Tedrake. "Whole-body motion planning with centroidal dynamics and full kinematics." *2014 IEEE-RAS International Conference on Humanoid Robots*. IEEE, 2014.

直接配点法

优点:

可以处理非常复杂的约束

$$\underset{\{h,x_0,\ldots,x_N,u_1,\ldots,u_N,\lambda_1,\ldots,\lambda_N\}}{\text{minimize}} g_f(x_N) + h\sum_{k=1}^{N} g(x_{k-1},u_k)$$

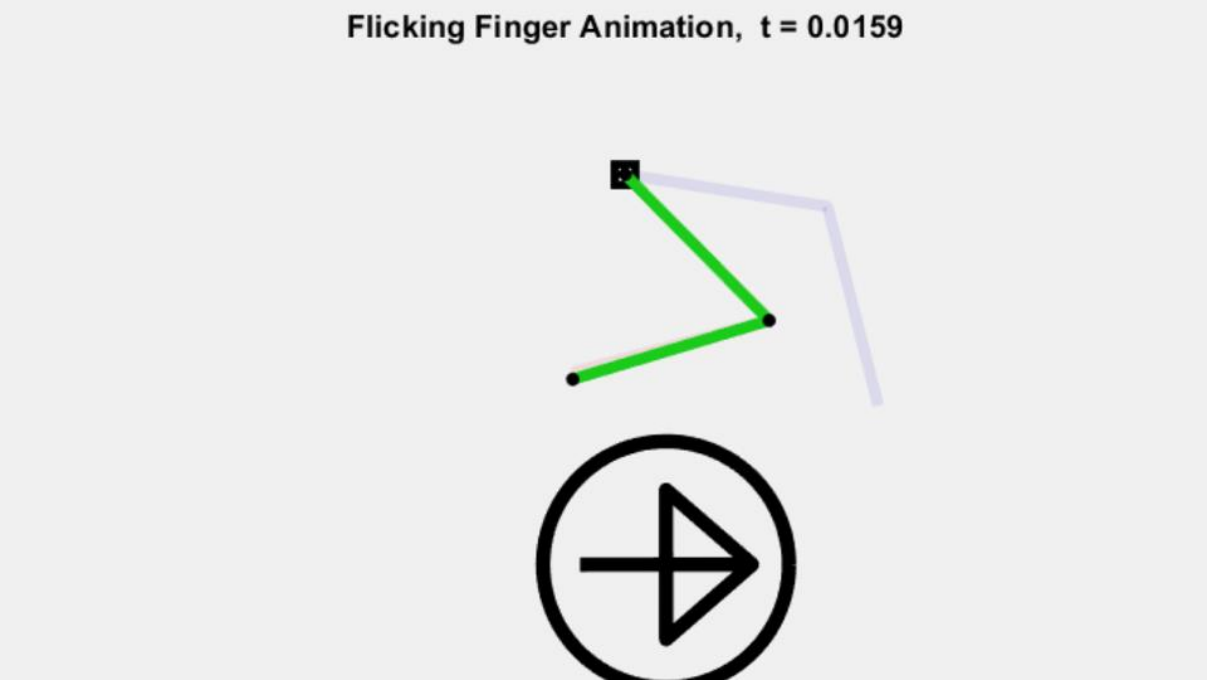
$$\begin{aligned}
q_k - q_{k+1} + h\dot{q}_{k+1} &= 0 \\
H_{k+1}(\dot{q}_{k+1} - \dot{q}_k) + h\left(C_{k+1} + G_{k+1} - B_{k+1}u_{k+1} - J_{k+1}^T\lambda_{k+1}\right) &= 0 \\
\phi(q_k) &\geq 0 \\
\lambda_{k,z}, \lambda_{k,x}^+, \lambda_{k,x}^-, \gamma_k &\geq 0 \\
\mu\lambda_{k,z} - \lambda_{k,x}^+ - \lambda_{k,x}^- &\geq 0 \\
\gamma_k + \psi(q_k,\dot{q}_k) &\geq 0 \\
\gamma_k - \psi(q_k,\dot{q}_k) &\geq 0 \\
\phi(q_k)^T\lambda_{k,z} &= 0 \\
\left(\mu\lambda_{k,z} - \lambda_{k,x}^+ - \lambda_{k,x}^-\right)^T\gamma_k &= 0 \\
(\gamma_k + \psi(q_k,\dot{q}_k))^T\lambda_{k,x}^+ &= 0 \\
(\gamma_k - \psi(q_k,\dot{q}_k))^T\lambda_{k,x}^- &= 0 \\
\phi(q_k) = q_{max} - q_k &\geq 0 \\
-\lambda_k &\geq 0 \\
\phi(q_k)^T\lambda_k &= 0.
\end{aligned}$$

Posa, Michael, Cecilia Cantu, and Russ Tedrake. "A direct method for trajectory optimization of rigid bodies through contact." *The International Journal of Robotics Research* 33.1 (2014): 69-81.

直接配点法

优点:

可以处理非常复杂的约束



Posa, Michael, Cecilia Cantu, and Russ Tedrake. "A direct method for trajectory optimization of rigid bodies through contact." *The International Journal of Robotics Research* 33.1 (2014): 69-81.

Direct Collocation

缺点

- 只能解出运动轨迹，不能获得反馈控制器
- 非线性优化算法对于有些问题可能非常低效


Planning As Inference

Planning As Inference 规划即推理

近年来发展的新轨迹规划方法

是现代应用深度增强学习进行机器人控制的主要理论基础之一

将轨迹规划看做最大后验概率估计问题

建立最优控制和概率估计之间的联系

理解LQR和直接配点法能帮助理解规划即推理

将直接配点法的概念推广为条件概率分布

直接配点法的决策变量为状态和控制轨迹上的点

$$Y = [x_1, ..., x_N, u_1, ..., u_N]$$

把这些点都看做是随机变量

$$Y = [x_1, ..., x_N, u_1, ..., u_N]$$

需要给直接配点法一个初始值

给随机变量初始的概率分布

$$p(x_1, ..., x_N, u_1, ..., u_N)$$

系统动力学方程为等式约束

$$x_{n+1} - Ax_n + Bu_n = 0$$

系统动力学方程约束作为一个概率事件

$$p(\mathcal{E}|x_{n+1}, x_n, u_n) = \mathcal{N}(x_{n+1} - Ax_n + Bu_n; 0, \Sigma)$$

$$\mathcal{N}(\theta; \mu, \Sigma) = \frac{1}{\sqrt{|2\pi\Sigma|}} \exp\{-\frac{1}{2}\|\theta - \mu\|_\Sigma^2\}$$

轨迹规划作为最大后验概率估计

定义所有的随机变量为$Y = [x_1, ..., x_N, u_1, ..., u_N]$，定义所有的约束为$\mathcal{C} = [\mathcal{E}_1, ..., \mathcal{E}_K]$ (K个约束)。Y的最大后验概率估计为

$$Y^{MAP} = p(Y, \mathcal{C}) \propto p(Y|\mathcal{C}) \propto p(\mathcal{C}|Y)p(Y)$$

最大后验概率估计的基础
是条件概率和贝叶斯定理

$$P(A \mid B) = \frac{P(B \mid A) \cdot P(A)}{P(B)}$$

A, B	= events
$P(A\|B)$	= probability of A given B is true
$P(B\|A)$	= probability of B given A is true
$P(A), P(B)$	= the independent probabilities of A and B

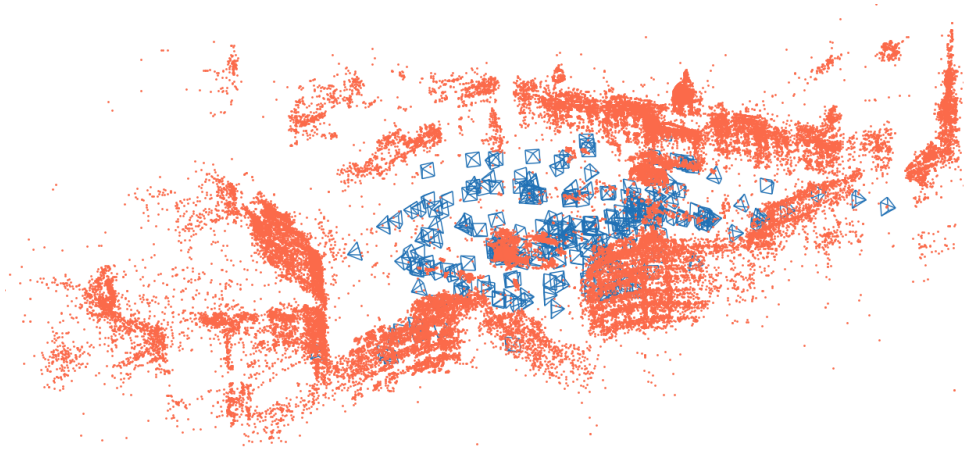
因子图和最大后验概率估计

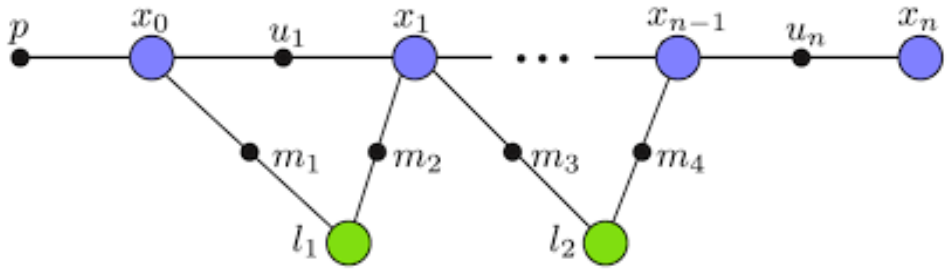
- 对于比较一般性的概率分布，最大后验概率估计必须要通过统计学的方法获得
- 但是如果概率分布都为高斯分布，我们可以用均值和方差来参数化高斯分布，从而得到概率分布的解析表达式
- 因子图（factor graph）是一种常用的处理多个随机变量间的高斯分布的工具

因子图和最大后验概率估计

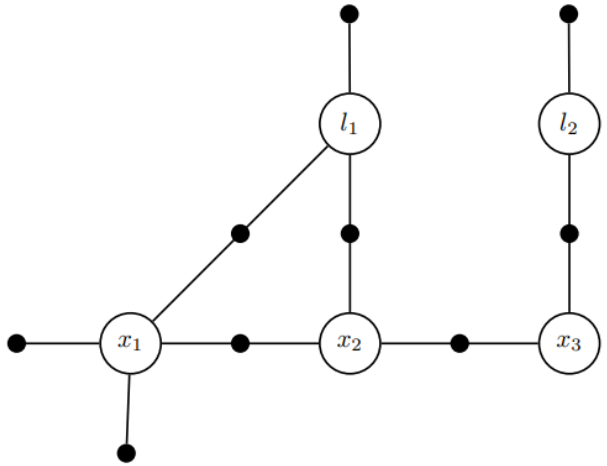
- 因子图广泛应用在SLAM领域

- 机器人的运动轨迹为随机变量分布
- 机器人的传感器提供对概率分布的约束
- 将因子图转化成最大后验概率估计问题，
- 根据传感器的观测值找到运动轨迹的后验概率分布





因子图和最大后验概率估计



圆圈代表随机变量 X
圆点代表因子

每一个因子都形如$\phi_i(X_i) \propto \exp\left\{ -\frac{1}{2}\|A_i X_i - b_i\|^2_{\Sigma_i} \right\}$

因为每个因子都是条件概率，所以所有因子总共的概率分布为因子的积

$$\phi(X) = \prod \phi_i(X_i)$$

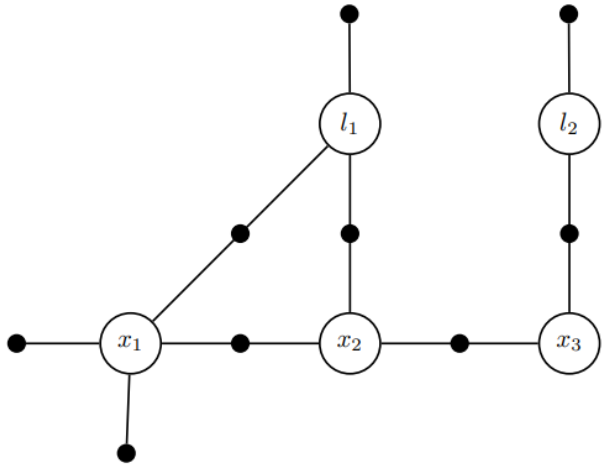
所以

$$X^{MAP} = \arg\max_X \ \phi(X) = \arg\min_X \ -2\log(\prod \phi_i(X_i))$$
$$= \arg\min_X \ \sum_i \|A_i X_i - b_i\|^2_{\Sigma_i} = \arg\min_X \ \|AX - b\|^2_{\Sigma}$$

最大后验概率问题被转化成一个矩阵的最小二乘问题

Dellaert, Frank, and Michael Kaess. "Factor graphs for robot perception." *Foundations and Trends® in Robotics* 6.1-2 (2017): 1-139.

因子图和稀疏矩阵最小二乘优化



圆圈代表随机变量 X
圆点代表因子

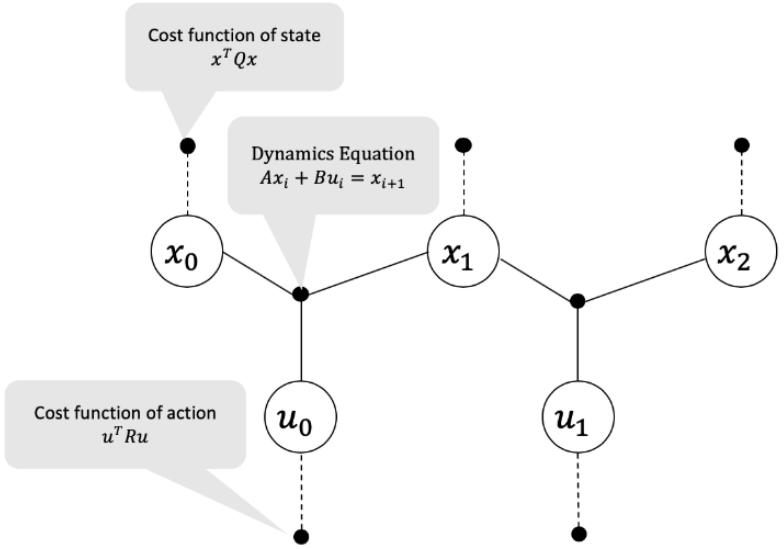
这个最小二乘问题通常用矩阵的QR分解来快速求解

$$\min_X \|AX-b\|_\Sigma^2 = \min_X Q^T\|AX-b\|_\Sigma^2 = \min_X \|RX-d\|_\Sigma^2 + \|e\|_\Sigma^2$$

结合因子图的稀疏性，QR分解可以只在A矩阵的子矩阵上进行，这样可以利用矩阵的稀疏性进一步提高计算速度。这种方法叫做变量消除（variable elimination）。

Dellaert, Frank, and Michael Kaess. "Factor graphs for robot perception." *Foundations and Trends® in Robotics* 6.1-2 (2017): 1-139.

因子图和最优控制



圆圈代表随机变量 X
圆点代表因子

$$\min_X \|AX-b\|_\Sigma^2 = \min_X Q^T\|AX-b\|_\Sigma^2 = \min_X \|RX-d\|_\Sigma^2 + \|e\|_\Sigma^2$$

将因子图应用于最优控制的时候，有两点需要考虑：

1. 如何获得最优反馈控制器？
   - **QR分解后的R矩阵和d向量中会包含最优控制器**

2. 系统动力学方程和额外的约束如果是需要严格遵守的，是否还能用因子图？
   - **可以，把他们当做协方差无限小的概率分布，然后把上述矩阵最小二乘问题当做无穷权重最小二乘问题求解**

Dellaert, Frank, and Michael Kaess. "Factor graphs for robot perception." *Foundations and Trends® in Robotics* 6.1-2 (2017): 1-139.

Gulliksson, Mårten. "On the modified Gram-Schmidt algorithm for weighted and constrained linear least squares problems." *BIT Numerical Mathematics* 35.4 (1995): 453-468.

用因子图表示LQR问题

Let

$$\mathcal{N}(\theta;\mu,\Sigma)=\frac{1}{\sqrt{|2\pi\Sigma|}}\exp\{-\frac{1}{2}\|\theta-\mu\|_{\Sigma}^{2}\}$$

Define random variables $\mathcal{E}_i$ as "constraint factor",

$$p(\mathcal{E}_2|\mathbf{x}_0,\mathbf{u}_0,\mathbf{x}_1)=\mathcal{N}(\mathbf{x}_1-A\mathbf{x}_0-B\mathbf{u}_0;0,\Sigma)$$

Define random variables $\mathcal{F}_i$ as "state cost factor",

$$p(\mathcal{F}_i|\mathbf{x}_j)=\mathcal{N}(\mathbf{x}_j;0,Q)$$

Define random variables $\mathcal{G}_i$ as "control cost factor",

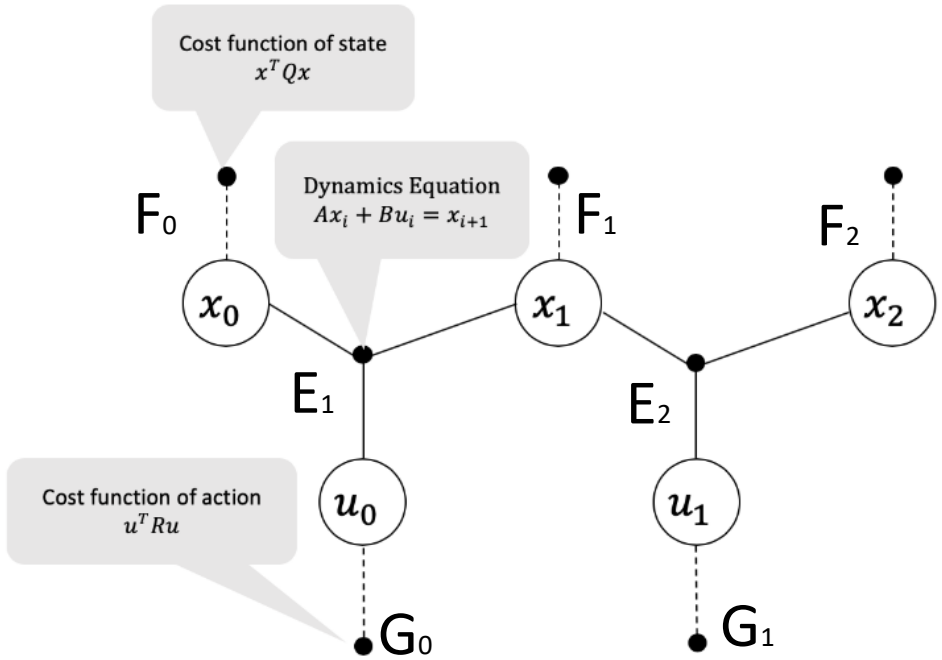
$$p(\mathcal{G}_i|\mathbf{u}_j)=\mathcal{N}(\mathbf{u}_j;0,R)$$

Probability

$$\begin{aligned}&p(\mathbf{x}_0,\mathbf{u}_0,\mathbf{x}_1,\mathbf{u}_1,\mathbf{x}_2,\mathcal{E}_1,\mathcal{E}_2,\mathcal{F}_0,\mathcal{F}_1,\mathcal{F}_2,\mathcal{G}_0,\mathcal{G}_1)\\=&p(\mathcal{F}_0|\mathbf{x}_0)p(\mathcal{F}_1|\mathbf{x}_1)p(\mathcal{E}_1|\mathbf{x}_0,\mathbf{u}_0,\mathbf{x}_1)p(\mathcal{E}_2|\mathbf{x}_1,\mathbf{u}_1,\mathbf{x}_2)\\&p(\mathcal{G}_0|\mathbf{u}_0)p(\mathcal{G}_1|\mathbf{u}_1)p(\mathbf{x}_0,\mathbf{u}_0,\mathbf{x}_1,\mathbf{u}_1,\mathbf{x}_2)\end{aligned}$$

Factor graph represents

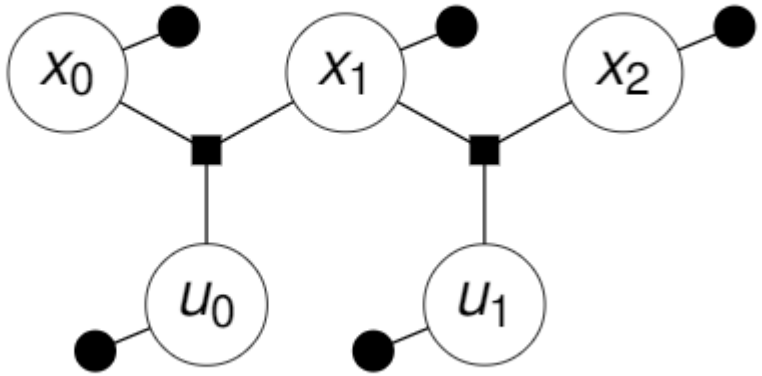
$$p(\mathcal{F}_0|\mathbf{x}_0)p(\mathcal{F}_1|\mathbf{x}_1)p(\mathcal{E}_1|\mathbf{x}_0,\mathbf{u}_0,\mathbf{x}_1)p(\mathcal{E}_2|\mathbf{x}_1,\mathbf{u}_1,\mathbf{x}_2)p(\mathcal{G}_0|\mathbf{u}_0)p(\mathcal{G}_1|\mathbf{u}_1)$$



通过因子图求解LQR

$$\min_{\mathbf{u}} x_T^T Q_{xx_T} x_T + \sum_{t=0}^{T-1} (x_t^T Q_{xx_t} x_t + u_t^T Q_{uu_t} u_t)$$

$$\text{s.t.} \quad x_{t+1} = F_{x_t} x_t + F_{u_t} u_t$$

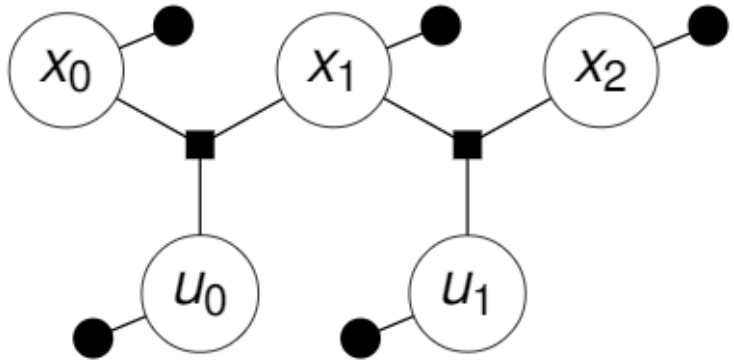


通过因子图求解LQR

$$\min_{\mathbf{u}} x_0^T Q_{xx_t} x_0 + x_1^T Q_{xx_t} x_1 + x_2^T Q_{xx_T} x_2 + u_0^T Q_{uu_t} u_0 + u_1^T Q_{uu_t} u_1$$

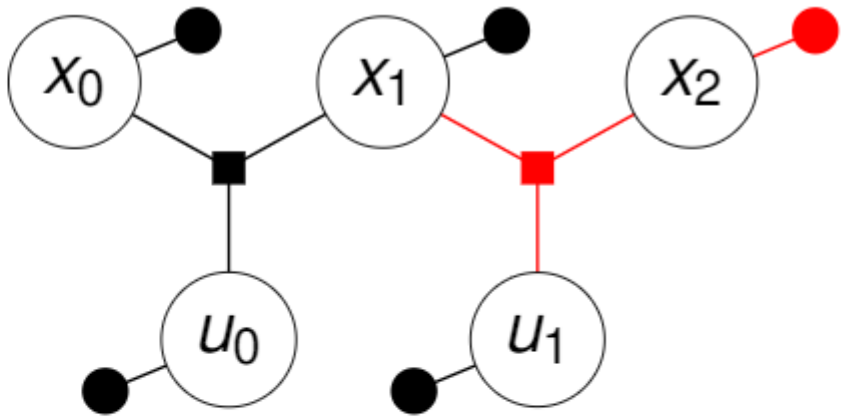
$$\text{s.t.} \quad x_2 = F_{x_t} x_1 + F_{u_t} u_1$$

$$x_1 = F_{x_t} x_0 + F_{u_t} u_0$$



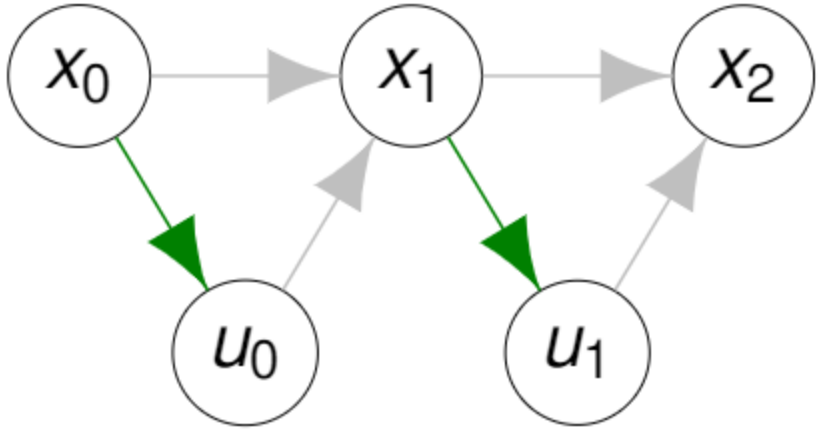
通过因子图求解LQR – 因子图的矩阵表示

$$
\begin{array}{c|cccccc|c}
W & x_2 & u_1 & x_1 & u_0 & x_0 & \\
\hline
I & Q_{xx_T}^{1/2} & 0 & 0 & 0 & 0 & 0 \\
\infty & I & -F_{u_1} & -F_{x_1} & 0 & 0 & 0 \\
I & 0 & Q_{uu_t}^{1/2} & 0 & 0 & 0 & 0 \\
I & 0 & 0 & Q_{xx_t}^{1/2} & 0 & 0 & 0 \\
\infty & 0 & 0 & I & -F_{u_1} & -F_{x_1} & 0 \\
I & 0 & 0 & 0 & Q_{uu_t}^{1/2} & 0 & 0 \\
I & 0 & 0 & 0 & 0 & Q_{xx_t}^{1/2} & 0
\end{array}
$$



通过因子图求解LQR – 因子图的变量消除

$$
\begin{array}{c|ccccc|c}
w & x_2 & u_1 & x_1 & u_0 & x_0 & \\
\hline
\infty & I & -F_{u_1} & -F_{x_1} & 0 & 0 & 0 \\
R_1 & 0 & I & K_1 & 0 & 0 & 0 \\
\infty & 0 & 0 & I & -F_{u_1} & -F_{x_1} & 0 \\
R_0 & 0 & 0 & 0 & I & K_0 & 0 \\
I & 0 & 0 & 0 & 0 & P_0 & 0
\end{array}
$$



反馈控制器出现在变量消除后的因子图中

用因子图表示的EC-LQR

$$\min_{x(n),u(n)} \quad J = x_N^T Q x_N + \sum_{n=0}^{N-1} x(n)^T Q x(n) + u(n)^T R u(n)$$

$$\text{s.t.} \quad x(0) = x_0$$
$$x(n+1) = A(n)x(n) + B(n)u(n)$$
$$C(l)x(l) + D(l)u(l) + r(l) = 0$$
$$G(k)x(k) + h(k) = 0$$

Define random variables $\mathcal{H}_i$ and $\mathcal{I}_i$ as "equality constraint factor"

$$p(\mathcal{H}_i|\mathbf{x}_j, \mathbf{u}_j) = \mathcal{N}(C\mathbf{x}_j + D\mathbf{u}_j; -r, \Sigma)$$

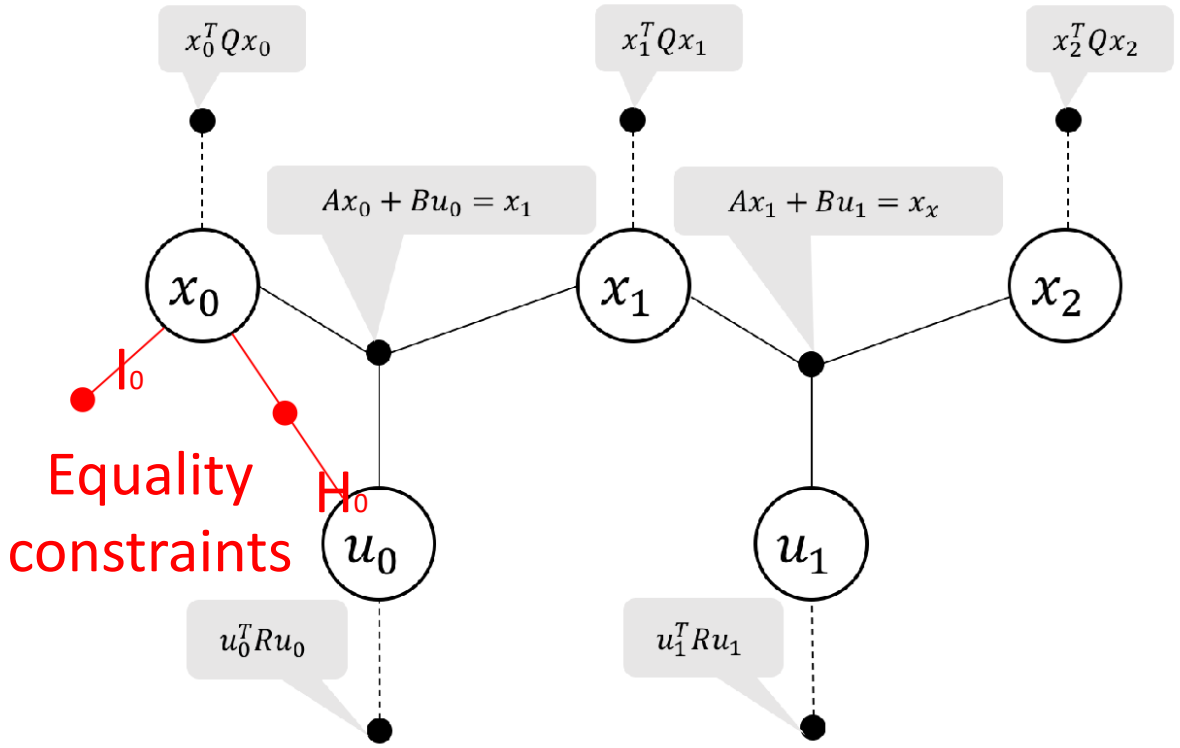
$$p(\mathcal{I}_i|\mathbf{x}_j) = \mathcal{N}(G\mathbf{x}_j; -h, \Sigma)$$

Probability with additional constraints

$$\begin{aligned} & p(\mathbf{x}_0, \mathbf{u}_0, \mathbf{x}_1, \mathbf{u}_1, \mathbf{x}_2, \mathcal{E}_1, \mathcal{E}_2, \mathcal{F}_0, \mathcal{F}_1, \mathcal{F}_2, \mathcal{G}_0, \mathcal{G}_1, \mathcal{H}_0, \mathcal{I}_0) \\ =& p(\mathcal{F}_0|\mathbf{x}_0)p(\mathcal{F}_1|\mathbf{x}_1)p(\mathcal{E}_1|\mathbf{x}_0, \mathbf{u}_0, \mathbf{x}_1)p(\mathcal{E}_2|\mathbf{x}_1, \mathbf{u}_1, \mathbf{x}_2) \\ & p(\mathcal{G}_0|\mathbf{u}_0)p(\mathcal{G}_1|\mathbf{u}_1)p(\mathcal{H}_0|\mathbf{x}_0, \mathbf{u}_0)p(\mathcal{I}_0|\mathbf{x}_0) \\ & p(\mathbf{x}_0, \mathbf{u}_0, \mathbf{x}_1, \mathbf{u}_1, \mathbf{x}_2) \end{aligned}$$

Factor graph represents

$$p(\mathcal{F}_0|\mathbf{x}_0)p(\mathcal{F}_1|\mathbf{x}_1)p(\mathcal{E}_1|\mathbf{x}_0, \mathbf{u}_0, \mathbf{x}_1)p(\mathcal{E}_2|\mathbf{x}_1, \mathbf{u}_1, \mathbf{x}_2)p(\mathcal{G}_0|\mathbf{u}_0)p(\mathcal{G}_1|\mathbf{u}_1)p(\mathcal{H}_0|\mathbf{x}_0, \mathbf{u}_0)p(\mathcal{I}_0|\mathbf{x}_0)$$



通过因子图求解EC-LQR

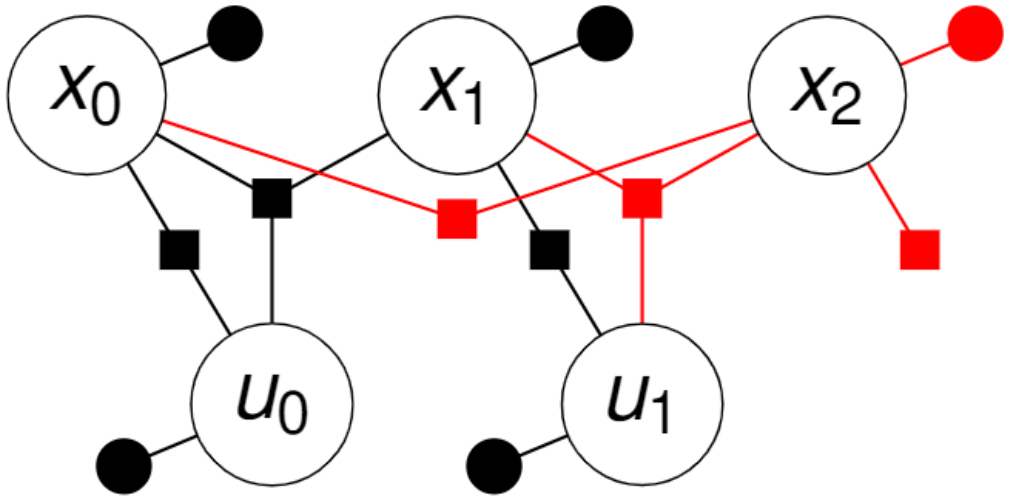
$$\min_{\mathbf{u}} x_T^T Q_{xx_T} x_T + \sum_{t=0}^{T-1} (x_t^T Q_{xx_t} x_t + u_t^T Q_{uu_t} u_t)$$

$$\text{s.t.} \quad x_{t+1} = F_{x_t} x_t + F_{u_t} u_t$$

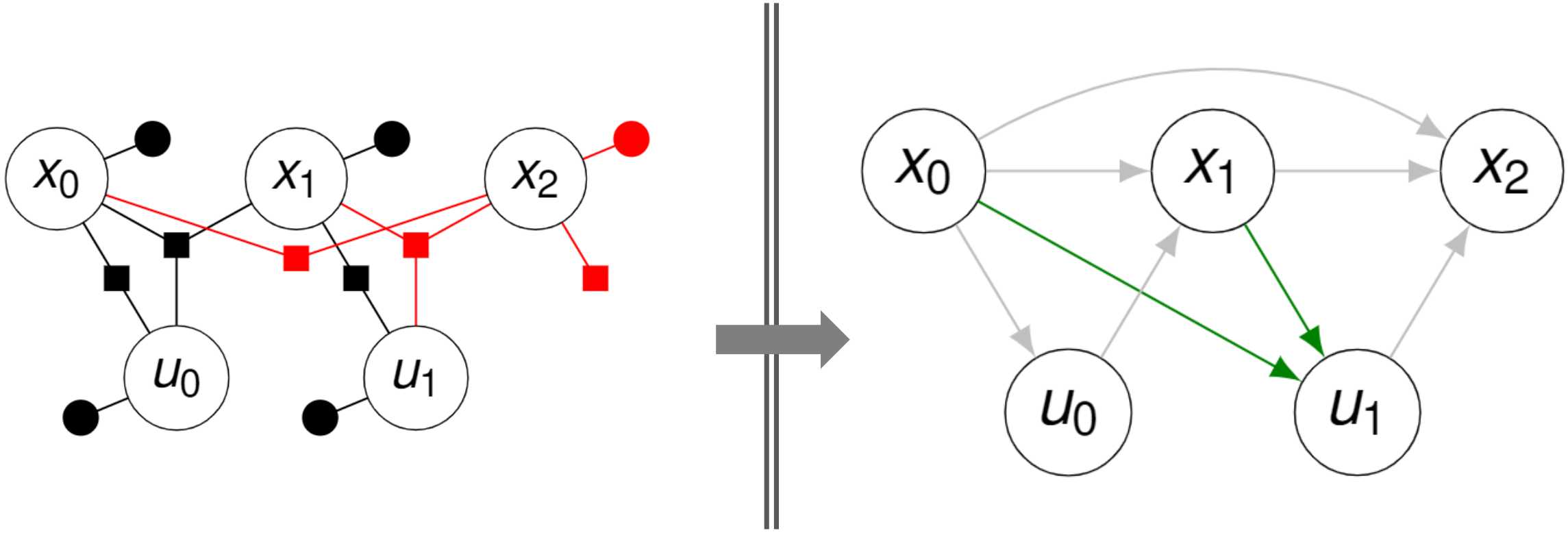
$$G_{x_t} x_t + G_{u_t} u_t + g_{l_t} = 0$$

$$G_{x_T} x_T + g_{l_T} = 0$$

$$S_0 x_0 + S_2 x_2 + s = 0$$



通过因子图求解EC-LQR



其他已有的等式约束线性二次型调节器研究

$$\min_{u(n),\ x(n)} \quad J = \psi(x(N)) + \sum_{n=0}^{N-1} L(x(n), u(n), n) \tag{1}$$

$$\text{subject to} \quad x(n+1) = A(n)x(n) + B(n)u(n), \tag{2}$$

$$n = 0, \ldots, N-1; \ x(0) = x_0$$

$$C(l)x(l) + D(l)u(l) + r(l) = 0 \quad l \in \mathcal{C}_{xu} \tag{3}$$

$$G(k)x(k) + h(k) = 0 \quad k \in \mathcal{C}_x \tag{4}$$

(Laine2019) Laine, Forrest, and Claire Tomlin. "Efficient computation of feedback control for equality-constrained lqr." *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019.

Giftthaler, Markus, and Jonas Buchli. "A projection approach to equality constrained iterative linear quadratic optimal control." *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*. IEEE, 2017.

Farshidian, Farbod, et al. "An efficient optimal planning and control framework for quadrupedal locomotion." *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017.

Rodriguez, Luis A., and Athanasios Sideris. "A sequential linear quadratic approach for constrained nonlinear optimal control." *Proceedings of the 2011 American Control Conference*. IEEE, 2011.

(Sideris2010) Sideris, Athanasios, and Luis A. Rodriguez. "A riccati approach to equality constrained linear quadratic optimal control." *Proceedings of the 2010 American Control Conference*. IEEE, 2010.

将EC-LQR应用于足式机器人

系统动力学方程

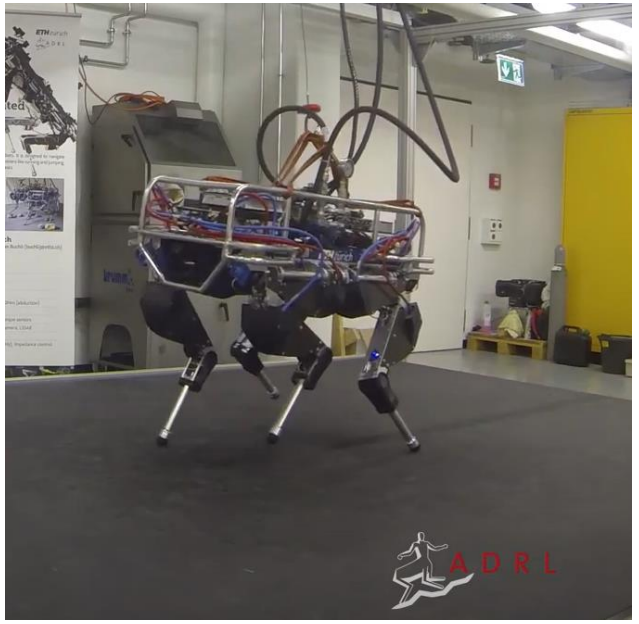
$$\dot{\mathbf{x}} = \mathbf{f}_i(\mathbf{x}, \mathbf{u})$$

$$\begin{cases} \dot{\theta} = \mathbf{T}(\theta)\left(\omega - {}_B\mathbf{J}^{\omega}_{com}(\mathbf{q})\,\mathbf{u}\right) \\ \dot{\mathbf{p}} = \mathbf{R}(\theta)\,\mathbf{v} \\ \dot{\omega} = \mathbf{I}^{-1}(\mathbf{q})\left(\dot{\mathbf{I}}(\mathbf{q},\mathbf{u}) - \omega \times \mathbf{I}(\mathbf{q})\omega + \sum_{i=1}^{4} \mathbf{r}_{f_i}(\mathbf{q}) \times \lambda_{f_i}\right) \\ \dot{\mathbf{v}} = \mathbf{g}(\theta) + \frac{1}{m}\sum_{i=1}^{4} \lambda_{f_i} \\ \dot{\mathbf{q}} = \mathbf{u} \end{cases}$$

足端等式约束

$$\mathbf{g}_{1i}(\mathbf{x}, \mathbf{u}, t) = 0$$

$$\begin{cases} \mathbf{u}_{f_i}(\mathbf{q},\mathbf{u}) = \mathbf{0} & \text{if } i \text{ is a stance leg} \\ \mathbf{u}_{f_i}(\mathbf{q},\mathbf{u}) \cdot \hat{n} = c(t), \quad \lambda_{f_i} = \mathbf{0} & \text{if } i \text{ is a swing leg} \end{cases}$$



Farshidian, Farbod, et al. "Real-time motion planning of legged robots: A model predictive control approach." *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*. IEEE, 2017.

将EC-LQR应用于足式机器人

$$
\begin{aligned}
\underset{\mathbf{u}(\cdot)}{\text{minimize}} \quad & \sum_{i=0}^{I-1} \Phi_i(\mathbf{x}(t_{i+1})) + \int_{t_i}^{t_{i+1}} L_i(\mathbf{x},\mathbf{u},t)dt \\
\text{subject to} \quad & \dot{\mathbf{x}} = \mathbf{f}_i(\mathbf{x},\mathbf{u}), \quad \mathbf{x}(s_0) = \mathbf{x}_0, \quad \mathbf{x}(t_i^-) = \mathbf{x}(t_i^+) \\
& \mathbf{g}_{1i}(\mathbf{x},\mathbf{u},t) = 0, \ \mathbf{g}_{2i}(\mathbf{x},t) = 0,
\end{aligned}
$$

$$
\begin{aligned}
\widetilde{J} &= \sum_{i=0}^{I-1} \widetilde{\Phi}_i(\delta\mathbf{x}(t_{i+1})) + \int_{t_i}^{t_{i+1}} \widetilde{L}_i(\delta\mathbf{x},\delta\mathbf{u},t)dt \\
\widetilde{\Phi}_i(\delta\mathbf{x}) &= q_{f,i} + \mathbf{q}_{f,i}^\top \delta\mathbf{x} + \frac{1}{2}\delta\mathbf{x}^\top \mathbf{Q}_{f,i}\delta\mathbf{x} \\
\widetilde{L}_i(\mathbf{x},\mathbf{u},t) &= q_i(t) + \mathbf{q}_i(t)^\top \delta\mathbf{x} + \mathbf{r}_i(t)^\top \delta\mathbf{u} + \delta\mathbf{x}^\top \mathbf{P}_i(t)\delta\mathbf{u} \\
&\quad + \frac{1}{2}\delta\mathbf{x}^\top \mathbf{Q}_i(t)\delta\mathbf{x} + \frac{1}{2}\delta\mathbf{u}^\top \mathbf{R}_i(t)\delta\mathbf{u} \\
\delta\dot{\mathbf{x}} &= \mathbf{A}_i(t)\delta\mathbf{x} + \mathbf{B}_i(t)\delta\mathbf{u} \\
\mathbf{C}_i(t)\delta\mathbf{x} &+ \mathbf{D}_i(t)\delta\mathbf{u} + \mathbf{e}_i(t) = \mathbf{0} \\
\mathbf{F}_i(t)\delta\mathbf{x} &+ \mathbf{h}_i(t) = \mathbf{0}
\end{aligned}
$$

使用微分动态规划，线性化后的局部LQR问题是EC-LQR问题

Farshidian, Farbod, et al. "Real-time motion planning of legged robots: A model predictive control approach." *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*. IEEE, 2017.

规划即推理

目前的研究工作比较初步

其他已有的工作没有太多考虑约束，主要讨论将概率分布推广到无模型、非高斯分布的情况

我在进行有模型、高斯分布下如何考虑更多不同类型的约束的研究

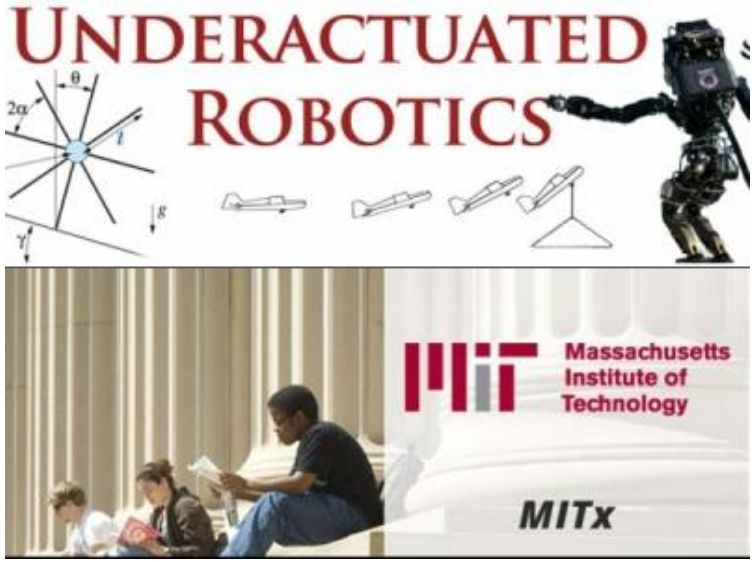
运动轨迹规划的方法对比

	Direct Collocation	**DDP**	**Planning As Inference**
Feedback policy?	No	Yes	Yes
Constraint handling?	Yes	Partial	Partial, better than DDP
Uncertainty handling?	No	No	Yes

学习资料分享

Direct Collocation

- MIT Professor Russ Tedrake's notes [1] is the Bible of trajectory optimization.



- Boston Dynamics engineer Matthew Kelly's tutorial [2] is also pretty good.

[1] Russ Tedrake. *Underactuated Robotics: Algorithms for Walking, Running, Swimming, Flying, and Manipulation (Course Notes for MIT 6.832).* Downloaded on 2020-09-22 from http://underactuated.mit.edu/

[2] Kelly, Matthew. "An introduction to trajectory optimization: How to do your own direct collocation." *SIAM Review* 59.4 (2017): 849-904.

Differential Dynamic Programming

- Tassa, Yuval, Nicolas Mansard, and Emo Todorov. "Control-limited differential dynamic programming." *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014.

- https://studywolf.wordpress.com/2016/02/03/the-iterative-linear-quadratic-regulator-method/

Factor Graph SLAM

- Dellaert, Frank. *Factor graphs and GTSAM: A hands-on introduction*. Georgia Institute of Technology, 2012.

- Dellaert, Frank, and Michael Kaess. "Factor graphs for robot perception." *Foundations and Trends® in Robotics* 6.1-2 (2017): 1-139.

Equality Constrained Linear Optimal Control With Factor Graphs

Shuo Yang[1], Gerry Chen[2], Yetong Zhang[2], Frank Dellaert[2], and Howie Choset[1]

***Abstract*— This paper presents a novel factor graph-based approach to solve the discrete-time finite-horizon Linear Quadratic Regulator problem subject to auxiliary linear equality constraints within and across time steps. We represent such optimal control problems using constrained factor graphs and optimize the factor graphs to obtain the optimal trajectory and the feedback control policies using the variable elimination algorithm with a modified Gram-Schmidt process. We prove that our approach has the same order of computational complexity as the state-of-the-art dynamic programming approach. Furthermore, current dynamic programming approaches can only handle equality constraints between variables at the same time step, but ours can handle equality constraints among any combination of variables at any time step while maintaining linear complexity with respect to trajectory length. Our approach can be used to efficiently generate trajectories and feedback control policies to achieve periodic motion or repetitive manipulation.**

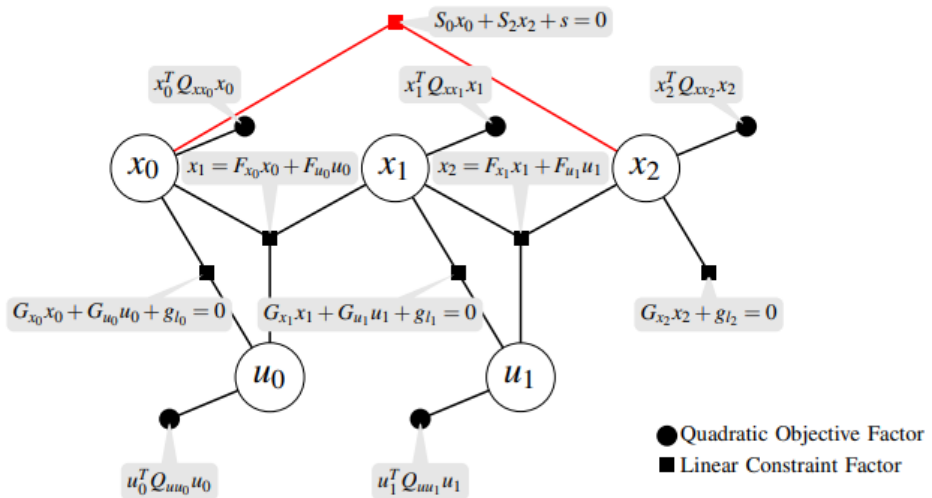


Fig. 1: The factor graph representation of an Equality Constrained Linear Quadratic Regular (EC-LQR) problem. Circles with letters are states or controls. Filled squares and circles represent objectives and constraints that involve the state or controls to which they are connected. The red square

总结

- 动力学建模和线性二次型调节器
- 三种运动轨迹规划方法
- 学习资料分享