Lecture 16 ME 221 11/22/22

Last time: Trajectory Generation

$q(t) = a_0 + a_1 t + a_2 t^2 \dots$

Given (you control)

$q(t_0) = q_0$

$\dot{q}(t_0) = \dot{q}_0$

$q(t_f) = q_f$

$\dot{q}(t_f) = \dot{q}_f$

$\vdots$

constraints

We solve the trajectory problem with polynomials:

parameters

$$\begin{bmatrix} q_0 \\ \dot{q}_0 \\ \vdots \\ \vdots \end{bmatrix} = \begin{bmatrix} 1 & t & t^2 & \cdots \\ 0 & 1 & 2t & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} \begin{bmatrix} a_0 \\ \vdots \\ a_n \end{bmatrix}$$

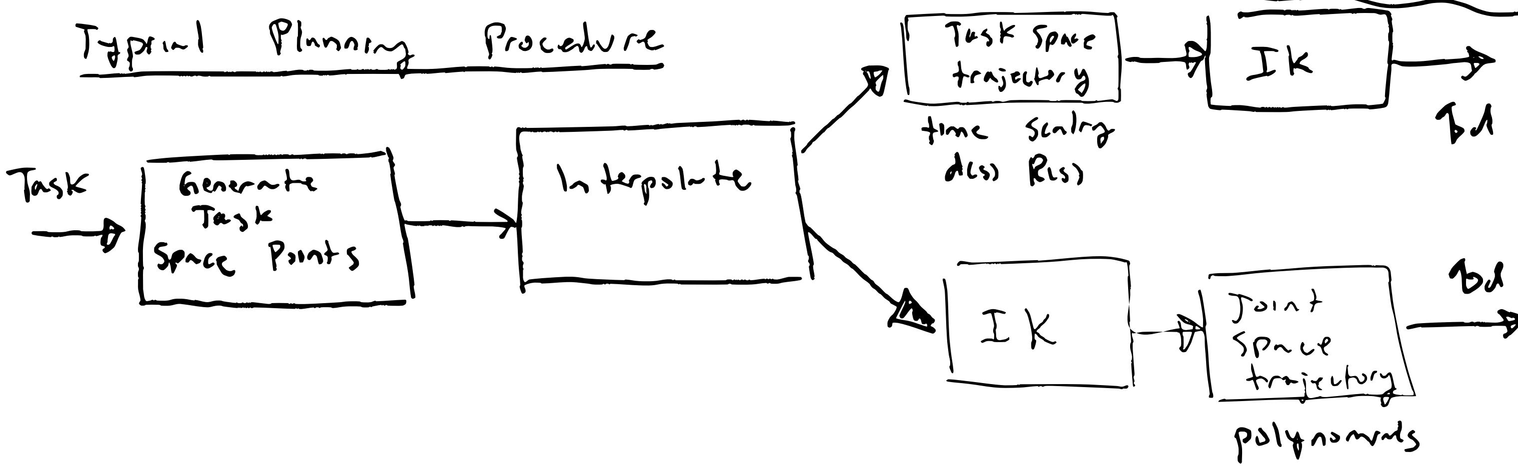
Lynch & Park Ch. 11

Springer handbook on Robotics Ch. 8

Today:

- outline basic trajectory Planning
- Control Architecture

Typical Planning Procedure



Feedforward Control

Q: If you have a perfect model of the robot, how can you use this info to help in control?

Let's use "tilde" to denote our model:

$$\underbrace{\tau = \tilde{M}(q)\ddot{q} + \tilde{C}(q,\dot{q})\dot{q} + \tilde{g}(q)}_{\text{our model!}}$$

$\tilde{M} \neq M$ (unless model is perfect!)

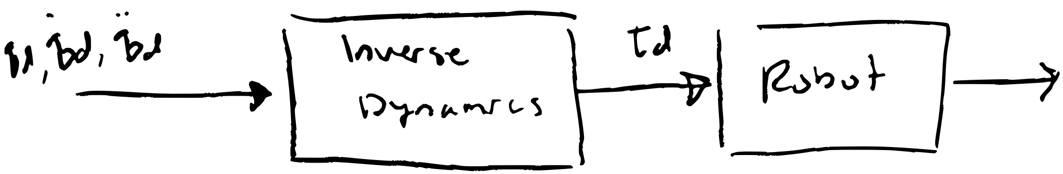
Why not use model to generate τ?

$$\tau_d = \tilde{M}(q_d)\ddot{q}_d + \tilde{C}(q_d, \dot{q}_d)\dot{q}_d + g(q_d)$$

Inverse Dynamics!

$$\tau_d = \tilde{M}(q_d)\,\ddot{q}_d + \tilde{C}(q_d, \dot{q}_d)\,\dot{q}_d + g(q_d)$$

τ_d : the torque required to drive $\begin{cases} q \to q_d \\ \dot{q} \to \dot{q}_d \\ \ddot{q} \to \ddot{q}_d \end{cases}$



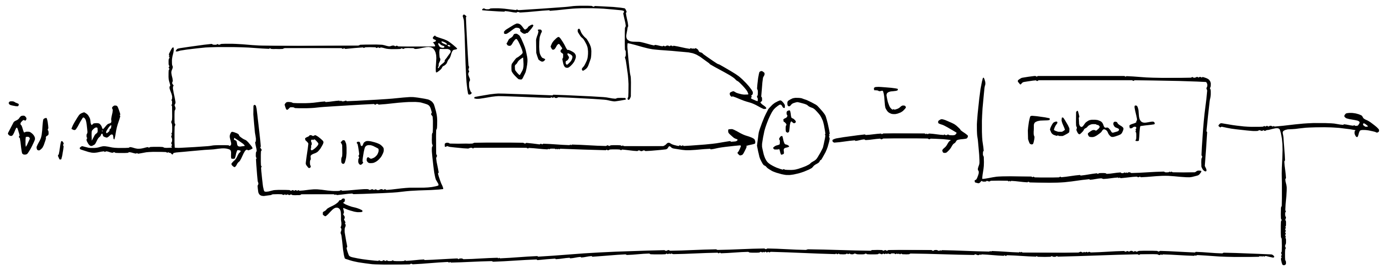
Comments

1. a perfect model would result in perfect tracking of q_d
2. Impossible to have a perfect model
3. Inverse Dynamics can be computationally expensive.

We don't have to "feed forward" all the dynamics. For example, gravity compensation is very common as a feedforward signal:

$$\tau_g = \tilde{g}(q)$$

$$\tau_{PID} = K_P e + K_D \dot{e} + K_I \int e\,dt$$



The computed Torque Controller (Feedback linearization)

idea: use model to *cancel* the dynamics of the robot to make it easier to control.

Model: $\tau = \tilde{M}(q)\ddot{q} + \tilde{C}(q,\dot{q})\dot{q} + \tilde{g}(q)$

PID Error Dynamics: $\ddot{e} + k_d\dot{e} + k_p e + k_I \int e\,dt = 0$

we want $e \to 0$, as $t \to \infty$

$\ddot{e} = \ddot{q}_d - \ddot{q}$

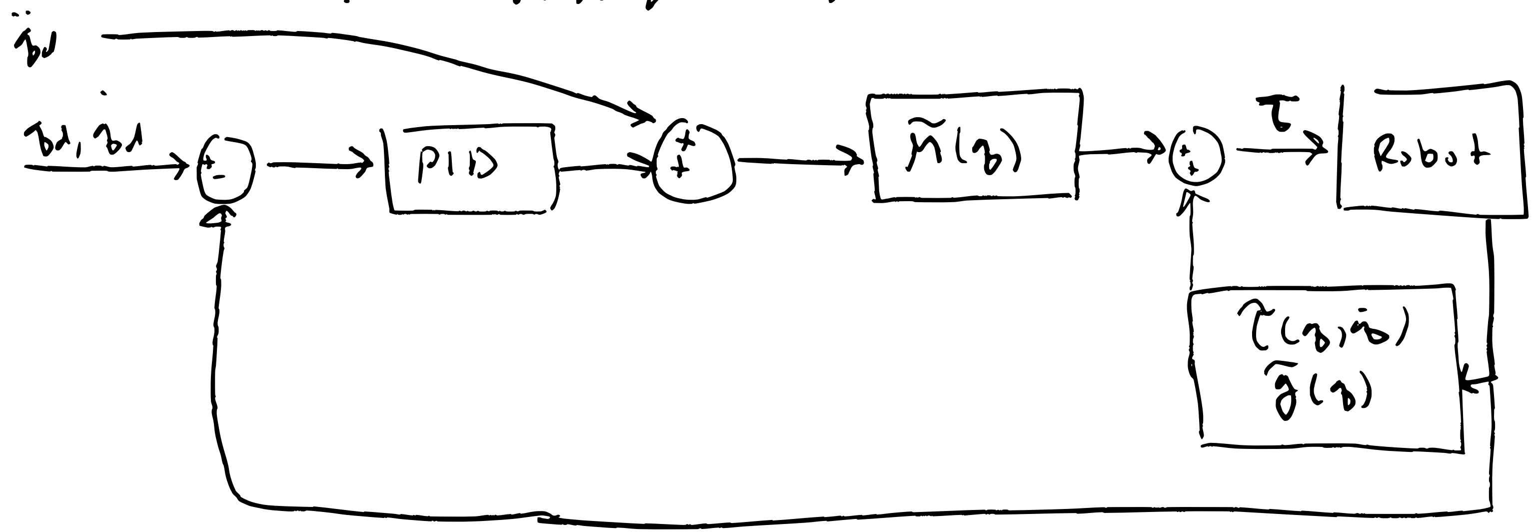
$\ddot{q}_d - \ddot{q} + k_d\dot{e} + k_p e + k_I \int e\,dt = 0$

* $\boxed{\ddot{q} = \ddot{q}_d + k_d\dot{e} + k_p e + k_I \int e\,dt}$

use this signal to generate torque with our model!!

$$\ddot{q} = \ddot{q}_d + K_d\dot{e} + K_p e + K_I\int e\,dt$$

$$\tau = \tilde{M}(q)\left[\ddot{q}_d + K_p e + K_I\int e\,dt + K_D\dot{e}\right] + \tilde{C}(q,\dot{q})\dot{q} + g(q)$$



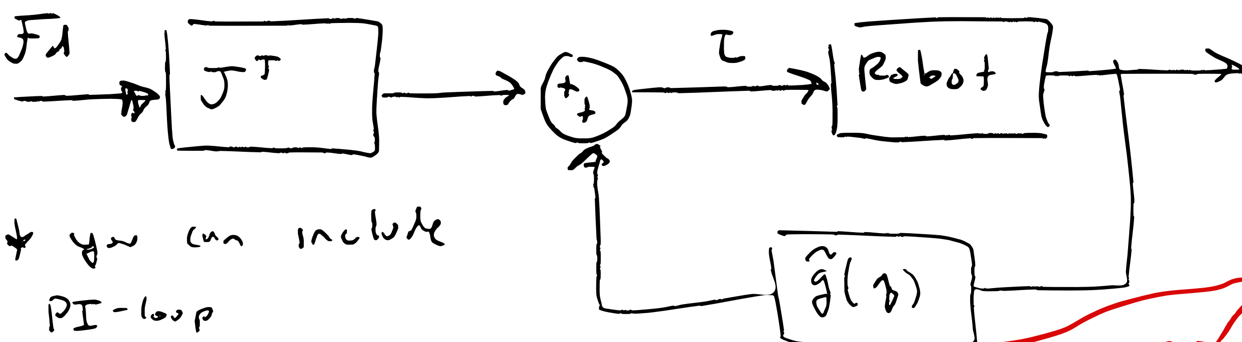
One of the most used controllers in application

Force Control

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) = \tau$$

F_d : desired end-effector twist $\quad F_d = \begin{pmatrix} F_x \\ F_y \\ F_z \\ \tau_x \\ \tau_y \\ \tau_z \end{pmatrix}$

$$\tau = \tilde{g}(q) + \underbrace{J^T(q)\, F_d}_{\tau \text{ need to get } F_d}$$



* you can include PI-loop

sensor?

$$\tau = \tilde{g}(q) + J^T(q)\left[K_p(F_d - F_{ext}) + K_I \int F_d - F_{ext}\, dt \right]$$

Task Space Control (PD-controller)

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) = \tau$$

lets start by define error signal:

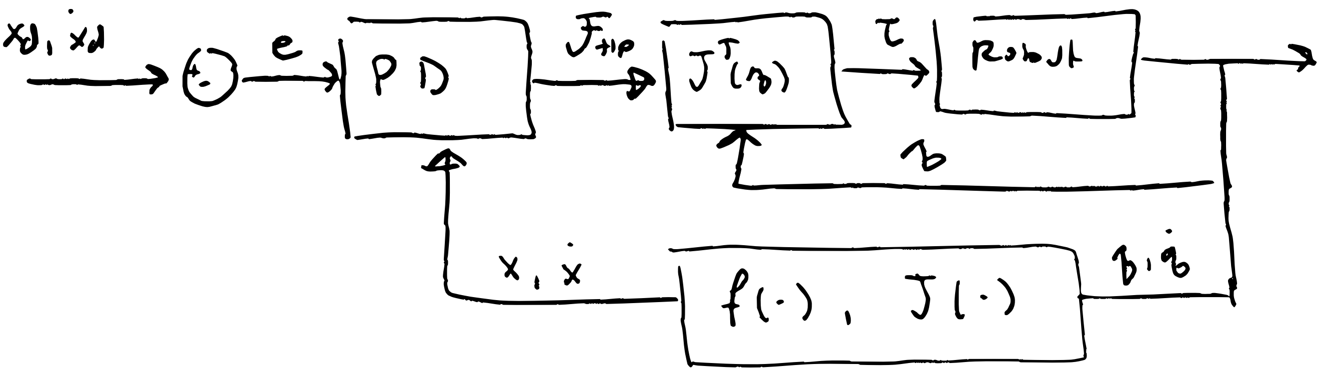
$$\underset{\substack{\uparrow \\ \text{desired} \\ \text{end-effect} \\ \text{position}}}{e = x_d} - \underset{\substack{\uparrow \\ \text{actual} \\ \text{end-effector} \\ \text{position.}}}{x} \longrightarrow e = x_d - \overbrace{f(q)}^{\text{forward kinematics}}$$

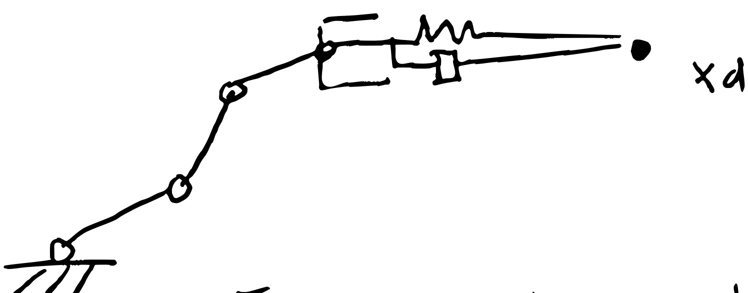
$$\dot{e} = \dot{x}_d - J(q)\dot{q}$$

idea: let's generate a tip force proportional to the error

$$F_{tip} = \underbrace{K_p\left(x_d - f(q)\right) + K_d\left(\dot{x}_d - J(q)\dot{q}\right)}_{\text{PD-controller based on end-effector error.}}$$

$$\tau = J^T(q)\left[K_p\left(x_d - f(q)\right) + K_d\left(\dot{x}_d - J(q)\dot{q}\right)\right]$$





This can be used as an Impedance Controller.

Task Space Dynamics

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) = \tau$$

$$\dot{x} = J(q)\dot{q}$$

$$\ddot{x} = J\ddot{q} + \dot{J}\dot{q} \Rightarrow \ddot{q} = J^{-1}x - J^{-1}\dot{J}\dot{q}$$

$$\begin{cases} \ddot{q} = J^{-1}x - J^{-1}\dot{J}J^{-1}\dot{x} \\ \dot{q} = J^{-1}\dot{x} \end{cases}$$

$$M\left(J^{-1}x - J^{-1}\dot{J}J^{-1}\dot{x}\right) + CJ^{-1}\dot{x} + g = \tau$$

$\times \; J^{-T}$

$$\left(J^{-T}MJ^{-1}\right)\ddot{x} - \left(J^{-T}MJ^{-1}\dot{J}J^{-1}\right)\dot{x} + \left(J^{-T}CJ^{-1}\right)\dot{x} + J^{-T}g = J^{-T}\tau$$

$$\Lambda(q)\ddot{x} + \Gamma(q,\dot{q})\dot{x} + \eta(q) = F$$

You can use __any__ methods previously discussed for task space control.