

Motor control

Gregor Schöner

Motor control

- is about the processes of bringing about the physical movement of an arm (robot or human)

- this entails

 - the mechanical dynamics of an arm

 - control principles

 - actuators

Resources

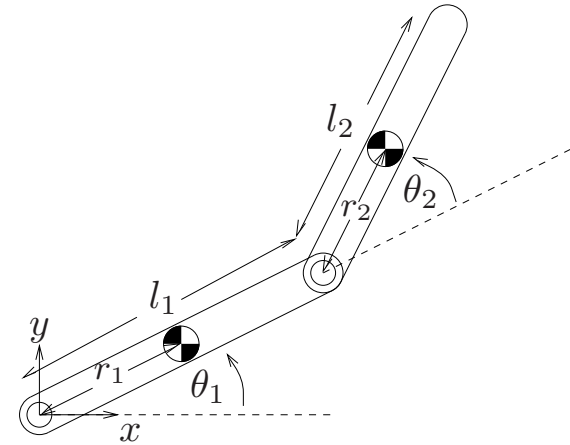
- R M Murray, Z Li, S S. Sastry: A mathematical introduction to robotic manipulation. CRC Press, 1994
- K M Lynch, F C Park: Modern Robotics: Mechanics, Planning, and Control. Cambridge University Press, 2017
- online version of both available...

Newton's law

- for a mass, m , described by a variable, x , in an inertial frame: $m\ddot{x} = f(x, t)$ where f is a force
- in non-inertial frames, e.g. rotating or accelerating frames:
 - centripetal forces
 - Coriolis forces

Rigid bodies: constraints

- constraints reduce the effective numbers of degrees of freedom...

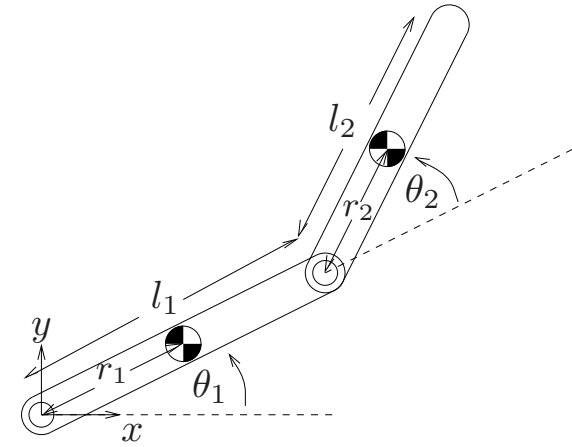


$$F_i = m_i \ddot{r}_i \quad r_i \in \mathbb{R}^3, i = 1, \dots, n.$$

$$g_j(r_1, \dots, r_n) = 0 \quad j = 1, \dots, k.$$

Rigid bodies: constraints

- generalized coordinates capture the remaining, free degrees of freedom



$$r_i = f_i(q_1, \dots, q_m) \\ i = 1, \dots, n$$



$$g_j(r_1, \dots, r_n) = 0 \\ j = 1, \dots, k.$$

Lagrangian mechanics

- The Lagrangian framework makes it possible to capture dynamics in generalized coordinates that reflect constraints

$$L(q, \dot{q}) = T(q, \dot{q}) - V(q),$$

- Lagrange function L = kinetic-potential energy

Lagrangian mechanics

- Least action principle: The integral of L over time=action is minimal $\delta A = \delta \int L(q, \dot{q}, t) dt = 0$

Euler-Lagrange equation

- $\delta A = \int \left(\frac{\partial L}{\partial q} \delta q + \frac{\partial L}{\partial \dot{q}} \delta \dot{q} \right) dt = 0$

- with $\delta \dot{q} = d\delta q/dt$

- and with partial integration

- $\delta A = \left[\frac{\partial L}{\partial \dot{q}} \delta q \right] + \int \left(\frac{\partial L}{\partial q} - \frac{d}{dt} \frac{\partial L}{\partial \dot{q}} \right) \delta q dt = 0$

- first term vanishes: no variation at start/end points

Euler-Lagrange equation

■ $\Rightarrow \frac{d}{dt} \frac{\partial L}{\partial \dot{q}} - \frac{\partial L}{\partial q} = 0$

■ ...plus generalized external forces, γ

■ $\frac{d}{dt} \frac{\partial L}{\partial \dot{q}} - \frac{\partial L}{\partial q} = \gamma$

■ in component form:

$$\frac{d}{dt} \frac{\partial L}{\partial \dot{q}_i} - \frac{\partial L}{\partial q_i} = \Upsilon_i \quad i = 1, \dots, m,$$

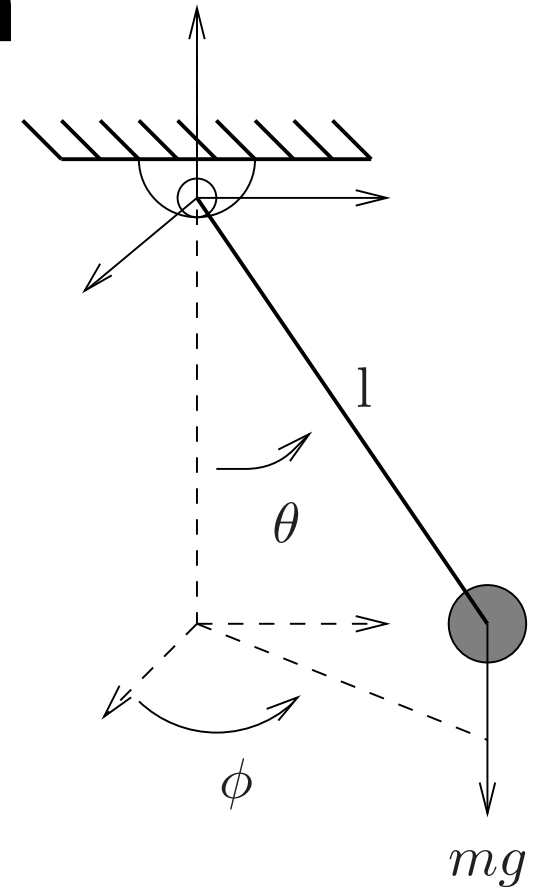
Example: pendulum

■ generalized coordinates: θ, ϕ

■
$$T = \frac{1}{2}ml^2 \|\dot{r}\|^2 = \frac{1}{2}ml^2 \left(\dot{\theta}^2 + (1 - \cos^2 \theta) \dot{\phi}^2 \right)$$

■
$$V = -mgl \cos \theta,$$

■
$$L(q, \dot{q}) = \frac{1}{2}ml^2 \left(\dot{\theta}^2 + (1 - \cos^2 \theta) \dot{\phi}^2 \right) + mgl \cos \theta$$



position relative to base $r(\theta, \phi) = \begin{bmatrix} l \sin \theta \cos \phi \\ l \sin \theta \sin \phi \\ -l \cos \theta \end{bmatrix}$

Example: pendulum


$$\frac{d}{dt} \frac{\partial L}{\partial \dot{\theta}} = \frac{d}{dt} (ml^2 \dot{\theta}) = ml^2 \ddot{\theta}$$
$$\frac{\partial L}{\partial \theta} = ml^2 \sin \theta \cos \theta \dot{\phi}^2 - mgl \sin \theta$$

$$\frac{d}{dt} \frac{\partial L}{\partial \dot{\phi}} = \frac{d}{dt} (ml^2 \sin^2 \theta \dot{\phi}) = ml^2 \sin^2 \theta \ddot{\phi} + 2ml^2 \sin \theta \cos \theta \dot{\theta} \dot{\phi}$$
$$\frac{\partial L}{\partial \phi} = 0$$


$$\begin{bmatrix} ml^2 & 0 \\ 0 & ml^2 \sin^2 \theta \end{bmatrix} \begin{bmatrix} \ddot{\theta} \\ \ddot{\phi} \end{bmatrix} + \begin{bmatrix} -ml^2 \sin \theta \cos \theta \dot{\phi}^2 \\ 2ml^2 \sin \theta \cos \theta \dot{\theta} \dot{\phi} \end{bmatrix} + \begin{bmatrix} mgl \sin \theta \\ 0 \end{bmatrix} = 0.$$

inertial

centrifugal
(Coriolis)

gravitational

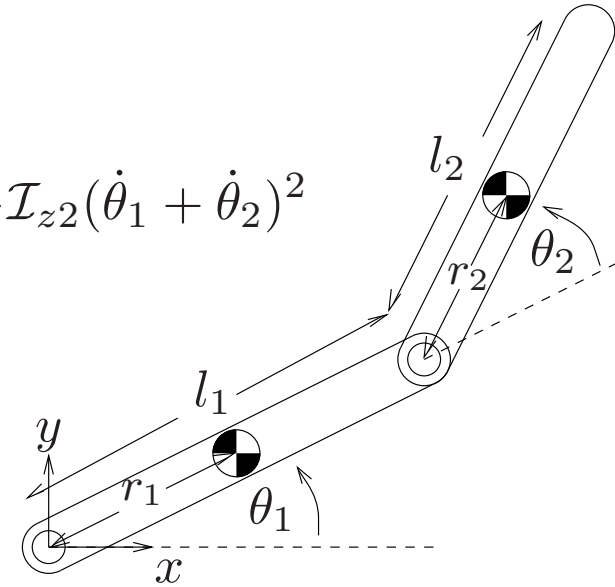
Example: two-link planar robot

■ generalized coordinates: θ_1, θ_2

■
$$T(\theta, \dot{\theta}) = \frac{1}{2}m_1(\dot{x}_1^2 + \dot{y}_1^2) + \frac{1}{2}\mathcal{I}_{z1}\dot{\theta}_1^2 + \frac{1}{2}m_2(\dot{x}_2^2 + \dot{y}_2^2) + \frac{1}{2}\mathcal{I}_{z2}(\dot{\theta}_1 + \dot{\theta}_2)^2$$

$$= \frac{1}{2} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix}^T \begin{bmatrix} \alpha + 2\beta c_2 & \delta + \beta c_2 \\ \delta + \beta c_2 & \delta \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix},$$

■ where $s_i = \sin(\theta_i)$, $c_i = \cos(\theta_i)$



$$\begin{bmatrix} \alpha + 2\beta c_2 & \delta + \beta c_2 \\ \delta + \beta c_2 & \delta \end{bmatrix} \begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{bmatrix} + \begin{bmatrix} -\beta s_2 \dot{\theta}_2 & -\beta s_2 (\dot{\theta}_1 + \dot{\theta}_2) \\ \beta s_2 \dot{\theta}_1 & 0 \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix} = \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix}$$

inertial

centrifugal/Coriolis

active
torques

Open-chain manipulator

$$M(\theta)\ddot{\theta} + C(\theta, \dot{\theta})\dot{\theta} + N(\theta, \dot{\theta}) = \tau$$

inertial

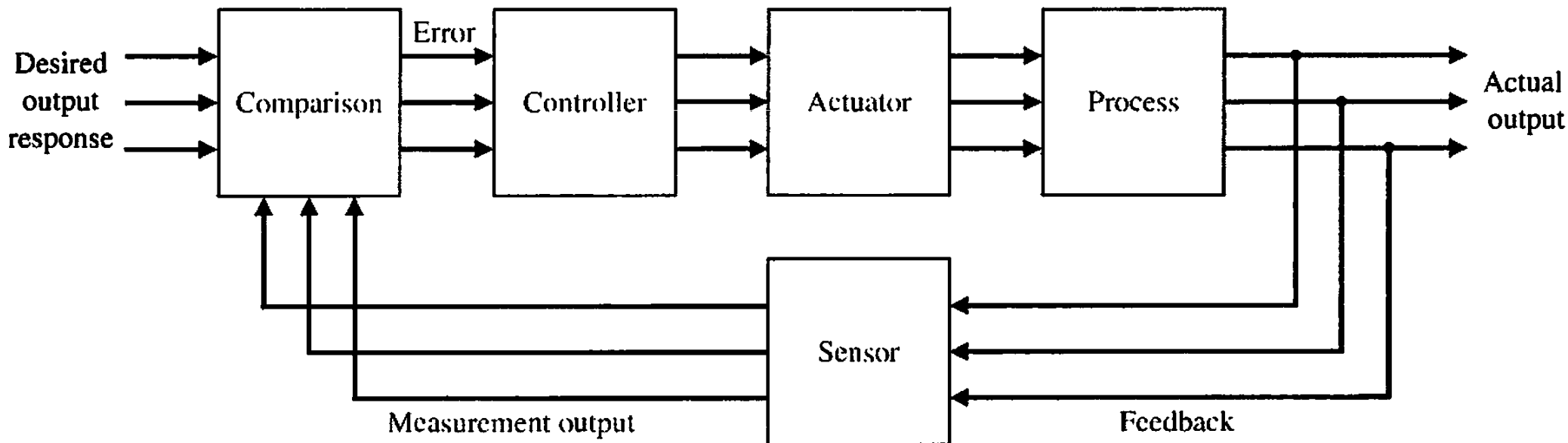
centrifugal/
Coriolis

gravitational

active
torques

Control systems

- robotic motion as a special case of control



[Dorf, Bishop, 2011]

Control systems

■ $\dot{x} = f(t, x, u) \qquad y = \eta(t, x, u)$

■ state of process/actuator x

■ output, y

■ control signal, u

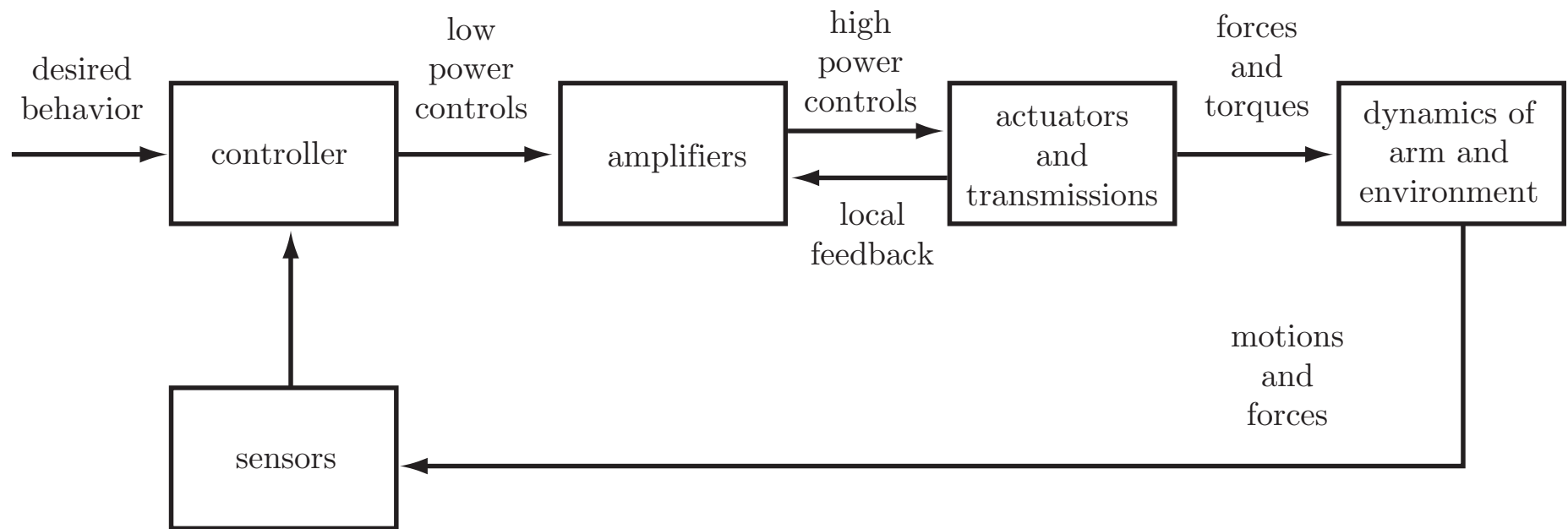
Control systems

■ $\dot{x} = f(t, x, u) \qquad y = \eta(t, x, u)$

■ control law: u as a function of y (or $\hat{y}$), desired response, y_d

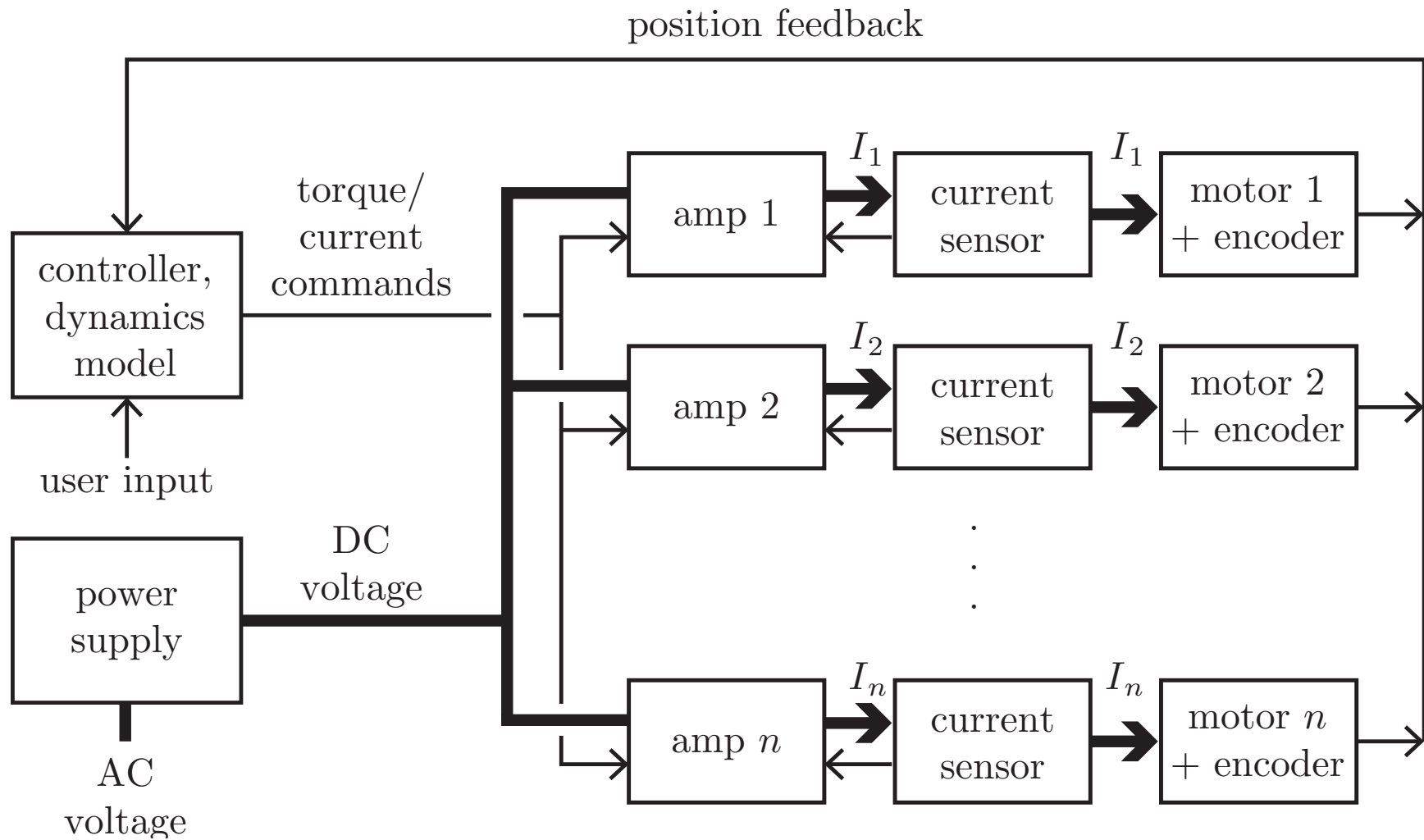
■ disturbances modeled stochastically

Robotic control



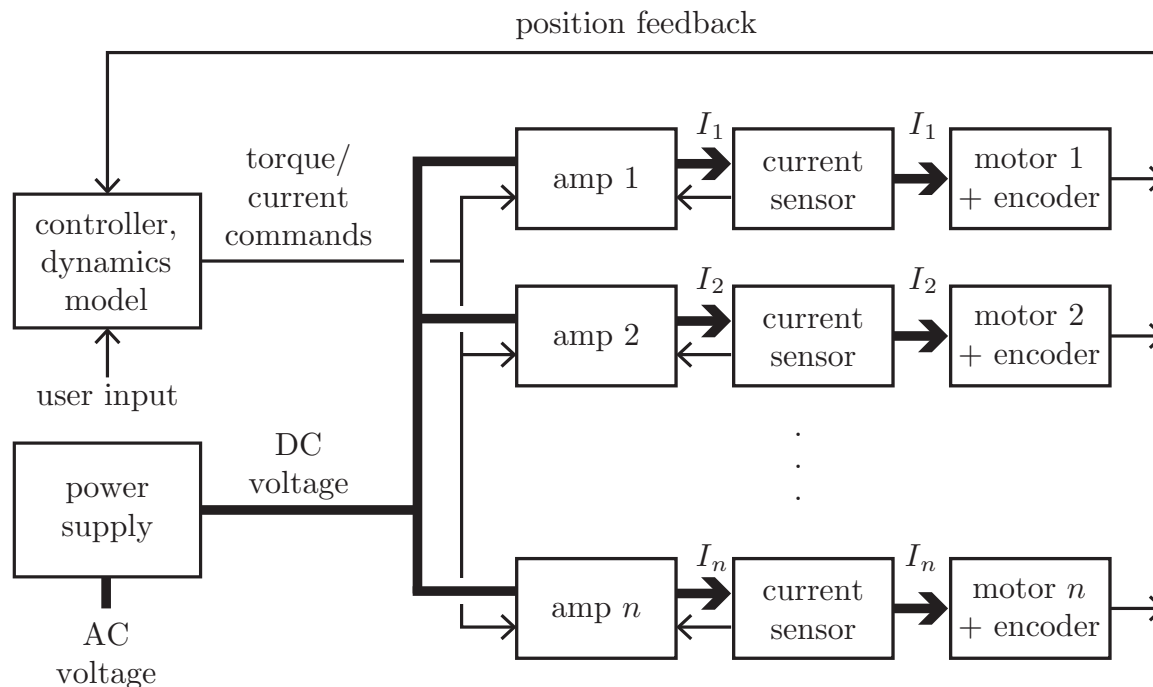
(a)

Robotic control



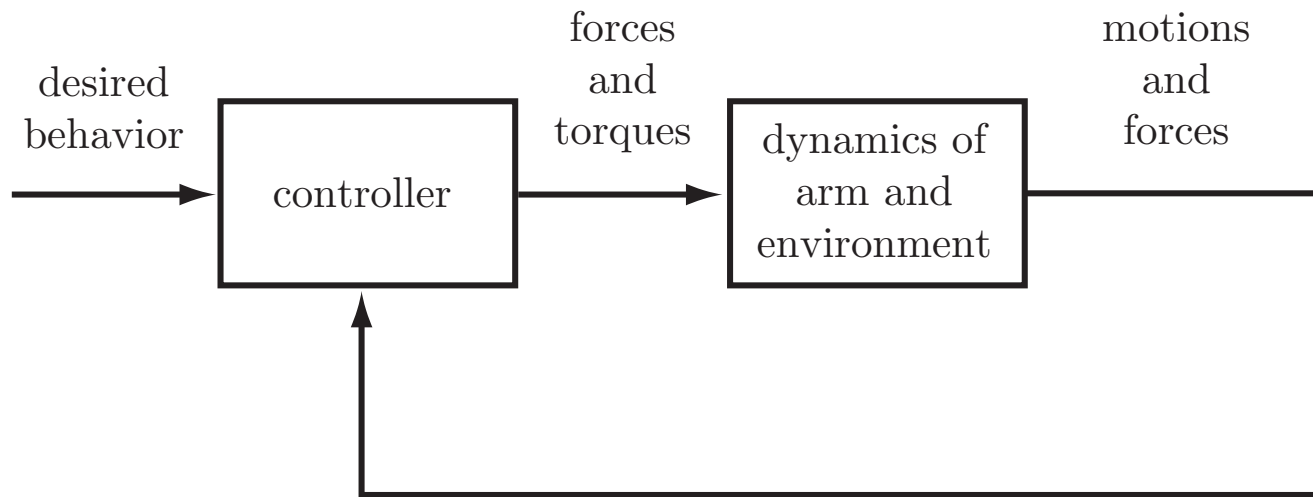
Robotic control

- actuators enable commanding a torque by commanding a current... in good approximation
- => control signal: torque



Robotic control

- $\dot{x} = f(t, x, u)$
- state variable $x(t)$ = output: kinematic state of robot
- desired trajectory: $x_d(t)$ (from motion planning)
- control signal: u = torques



(b)

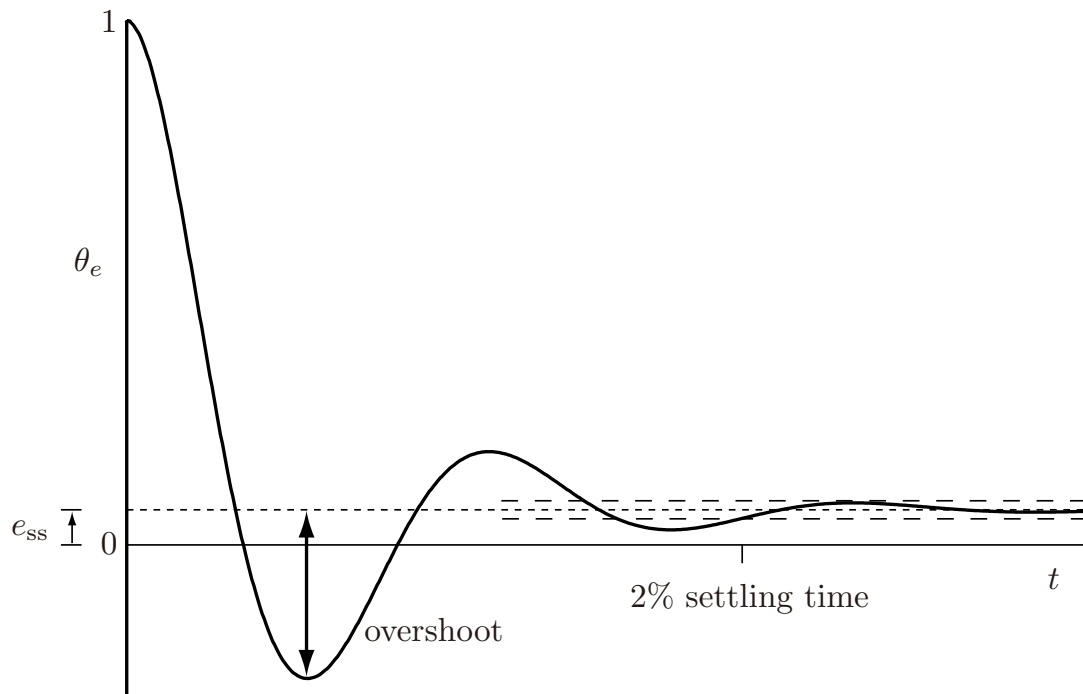
[Lunch, Park, 2017]

Robotic control

- theoretical core of robotic control theory:
- devise control laws that lead to stable control
- (approximate these numerically on hardware and computers)

Robotic control

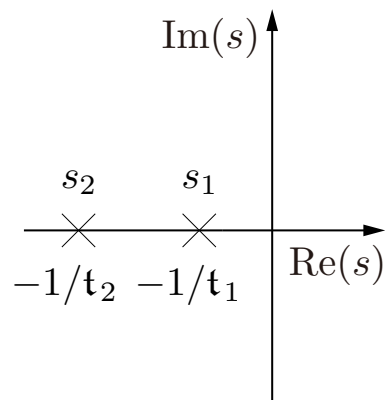
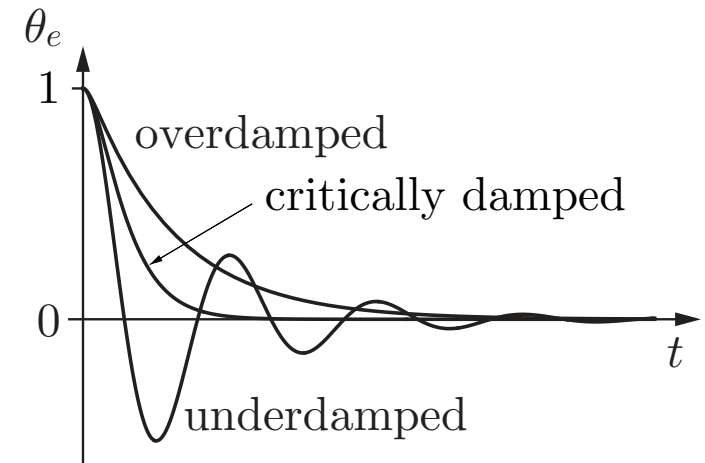
- task: generate joint torques that produce a desired motion... $\theta_d(t)$
- $\Leftrightarrow$ make error: $e(t) = \theta(t) - \theta_d(t)$ small



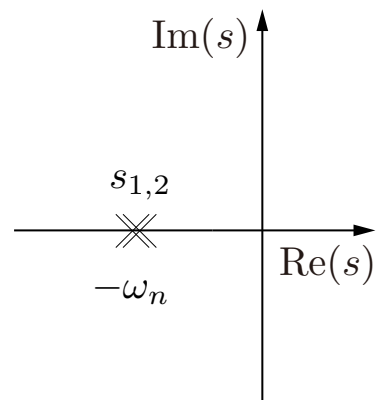
for a constant
desired state

Toy example

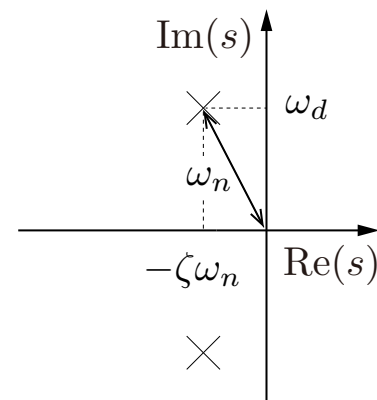
analysis by Eigenvalues s



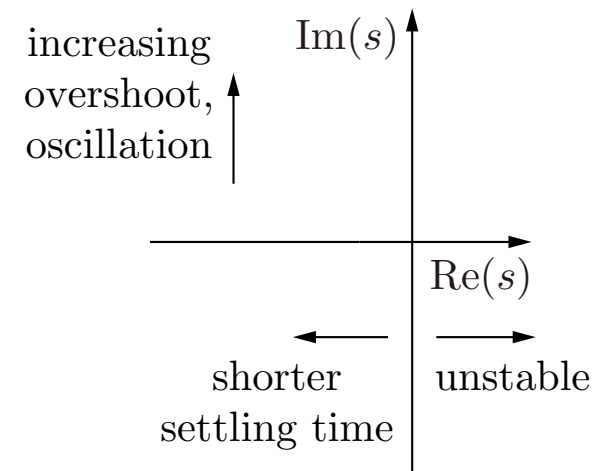
overdamped ($\zeta > 1$)



critically damped ($\zeta = 1$)

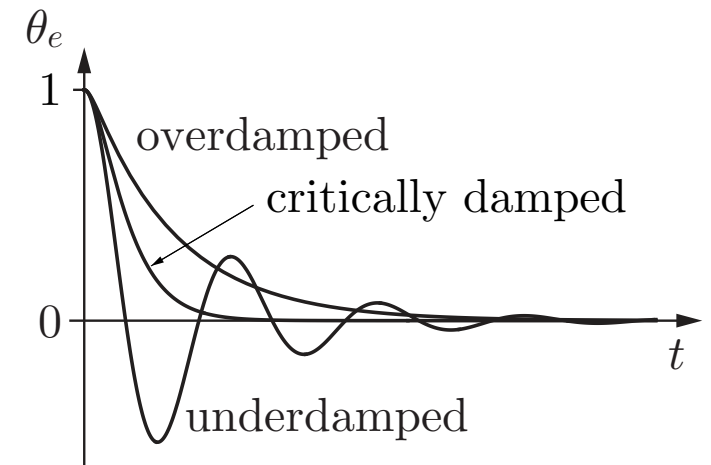
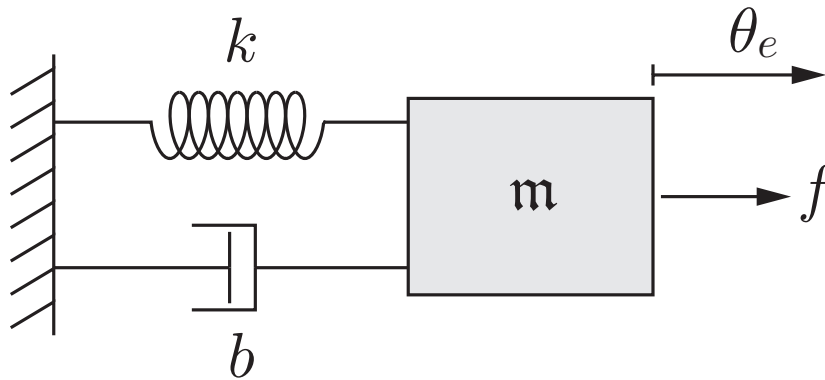


underdamped ($\zeta < 1$)



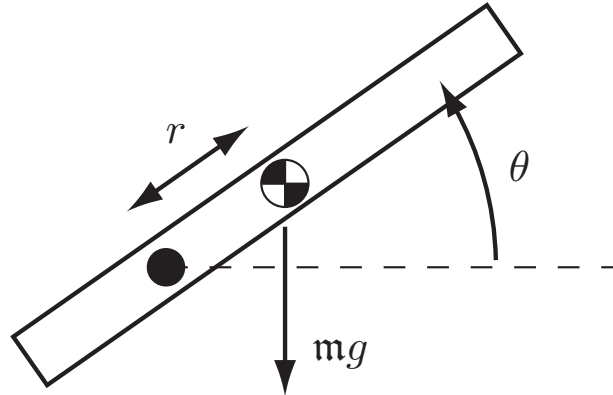
Toy example

- linear mass spring model
 $m\ddot{e}(t) + b\dot{e}(t) + ke(t) = 0$



Motion control single joint

■ $\tau = M\ddot{\theta} + mgr \cos(\theta) + b\dot{\theta}$



Motion control single joint

■ $\tau = M\ddot{\theta} + mgr \cos(\theta) + b\dot{\theta}$

■ feedback PID controller

■ $\tau = K_p\theta_e + K_d\dot{\theta}_e + K_i \int \theta(t')dt'$

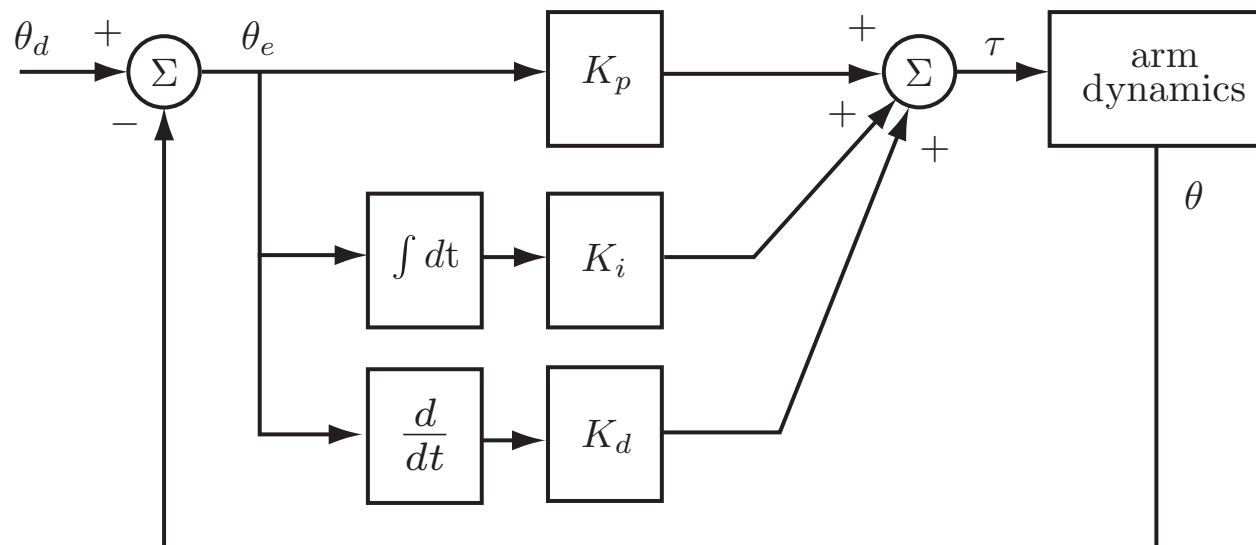


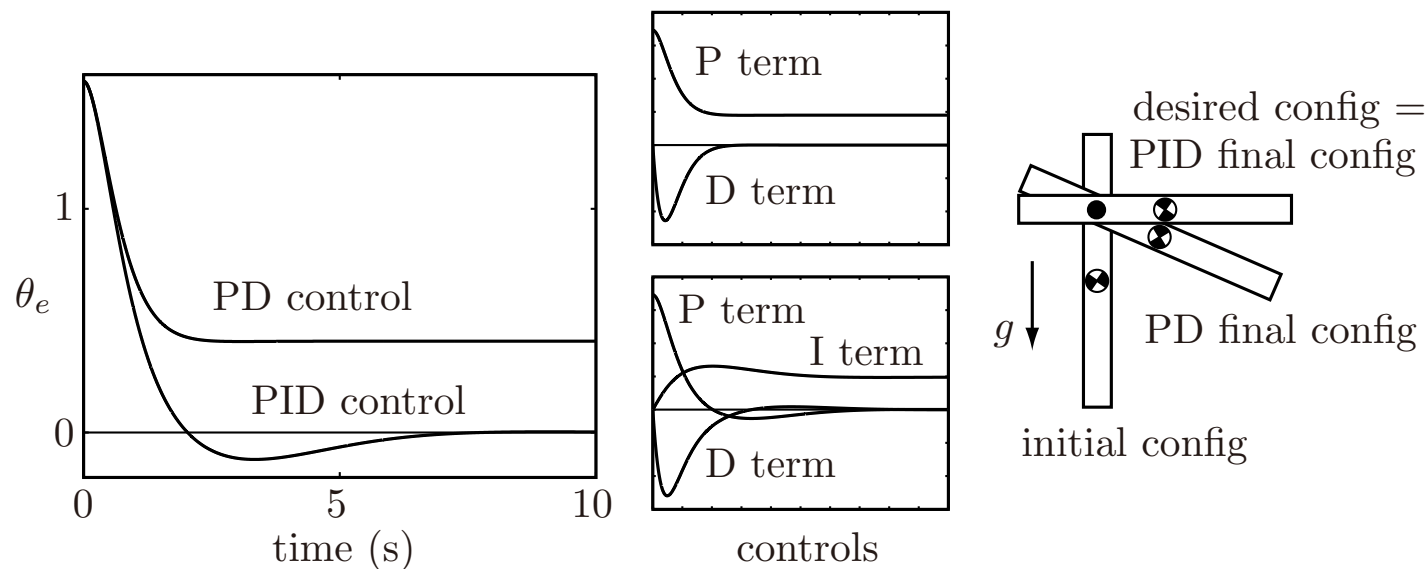
Figure 11.12: Block diagram of a PID controller.

Motion control single joint

■ $\tau = M\ddot{\theta} + mgr \cos(\theta) + b\dot{\theta}$

■ **feedback** PID controller

■ $\tau = K_p\theta_e + K_d\dot{\theta}_e + K_i \int \theta(t')dt'$



Motion control single joint

- $\tau = M\ddot{\theta} + mgr \cos(\theta) + b\dot{\theta} = M\ddot{\theta} + h(\theta, \dot{\theta})$

- **feedforward** controller

- has model of the dynamics:

- $\tau = \tilde{M}\ddot{\theta} + \tilde{h}(\theta, \dot{\theta})$

- compute forward torque

- $\tau(t) = \tilde{M}(\theta_d(t))\ddot{\theta}_d(t) + \tilde{h}(\theta_d, \dot{\theta}_d)$

- if model exact: $\ddot{\theta} \approx \ddot{\theta}_d$

Motion control single joint

■ feedforward controller

■ if model wrong..

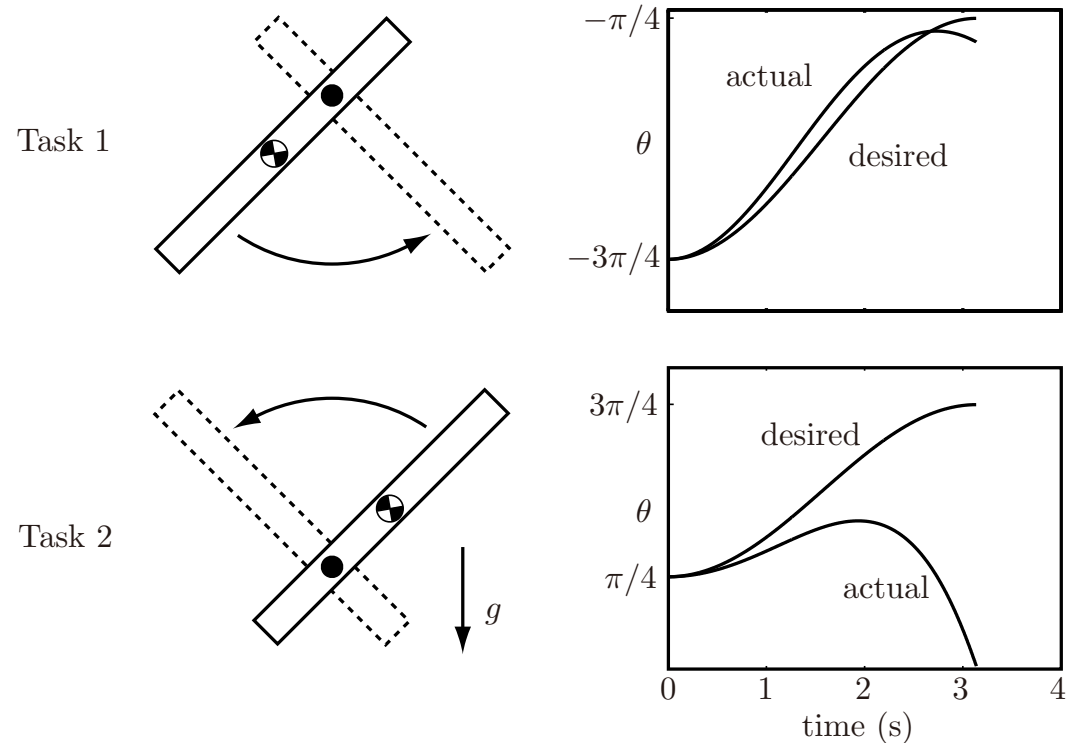


Figure 11.17: Results of feedforward control with an incorrect model: $\tilde{r} = 0.08$ m, but $r = 0.1$ m. The desired trajectory in Task 1 is $\theta_d(t) = -\pi/2 - (\pi/4)\cos(t)$ for $0 \leq t \leq \pi$. The desired trajectory for Task 2 is $\theta_d(t) = \pi/2 - (\pi/4)\cos(t)$, $0 \leq t \leq \pi$.

Motion control single joint

- combined **feedforward** and **feedback** PID controller ...

- $$\tau = \tilde{M}(\theta) \left(\ddot{\theta}_d + K_p \theta_e + K_d \dot{\theta}_e + K_i \int \theta(t') dt' \right) + \tilde{h}(\theta, \dot{\theta})$$

- = inverse dynamics or computed torque controller

Control of multi-joint arm

- generate joint torques that produce a desired motion... θ_d
- error $\theta_e = \theta - \theta_d$
- PD control $\tau = K_p\theta_e + K_e\dot{\theta}_d + K_i \int \theta_e(t')dt'$
- => controlling joints independently

$$M(\theta)\ddot{\theta} + C(\theta, \dot{\theta})\dot{\theta} + N(\theta, \dot{\theta}) = \tau$$

Control of multi-joint arm

- there are many more sophisticated models that compensate for interaction torques/ inertial coupling... e.g. **computed torque control (inverse dynamics)**

- $$\tau = \underbrace{M(\theta)\ddot{\theta}_d + C\dot{\theta} + N}_{\tau_{ff}} + \underbrace{M(\theta)(-K_v\dot{e} - K_p e)}_{\tau_{fb}}.$$

$$M(\theta)\ddot{\theta} + C(\theta, \dot{\theta})\dot{\theta} + N(\theta, \dot{\theta}) = \tau$$

$$\Rightarrow (\ddot{\theta} - \ddot{\theta}_d) = \ddot{e} = -K_v\dot{e} - K_p e$$

Control of multi-joint arm

- ... computed torque control (inverse dynamics)

- but: computational effort can be considerable... simplification.. only compensate for gravity...

- $$\tau = K_p \theta_e + K_e \dot{\theta}_d + K_i \int \theta_e(t') dt' + \tilde{N}(\theta)$$

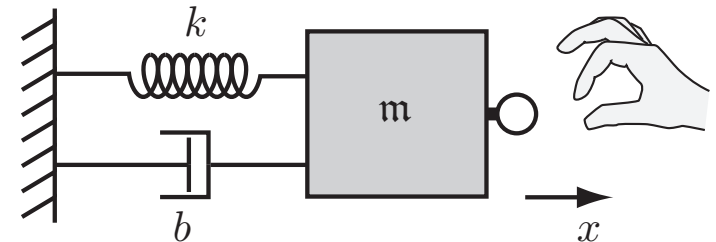
$$M(\theta)\ddot{\theta} + C(\theta, \dot{\theta})\dot{\theta} + N(\theta, \dot{\theta}) = \tau$$

Problem: contact forces

- as soon as the robot arm makes contact, a host of problems arise from the contact forces and their effect on the arm and controller...
- need compliance... resisting to a well-defined degree
- => impedance control... research frontier

Impedance

- to control movement well.. need a very stiff arm and “stiff” controller (high gain K_x)
- to control force/limit force (e.g. for interaction with surfaces or humans) you need a relatively soft arm and soft controller
- design system to give hand, x , a **desired impedance**: m , b , k in
- $m\ddot{x} + b\dot{x} + kx = f$
- where f is force applied..



Operational space formulation

- Euler-Langrage in end-effector space

- $$\Lambda(\mathbf{x})\ddot{\mathbf{x}} + \mu(\mathbf{x}, \dot{\mathbf{x}}) + \mathbf{p}(\mathbf{x}) = \mathbf{F}$$

- with $\mathbf{F}$ forces acting on the end-effector

- equivalent dynamics in joint space

- $$\mathbf{A}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{b}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{g}(\mathbf{q}) = \mathbf{\Gamma}$$

- with joint torques $\mathbf{\Gamma} = \mathbf{J}^T(\mathbf{q})\mathbf{F}$

[Khatib, 1987]

Impedance control

■ Hogan 1985...

■
$$\tau = J^T(\theta) \left(\tilde{\Lambda}(\theta) \ddot{x} + \tilde{\eta}(\theta, \dot{x}) - (M\ddot{x} + B\dot{x} + Kx) \right)$$

Link to movement planning

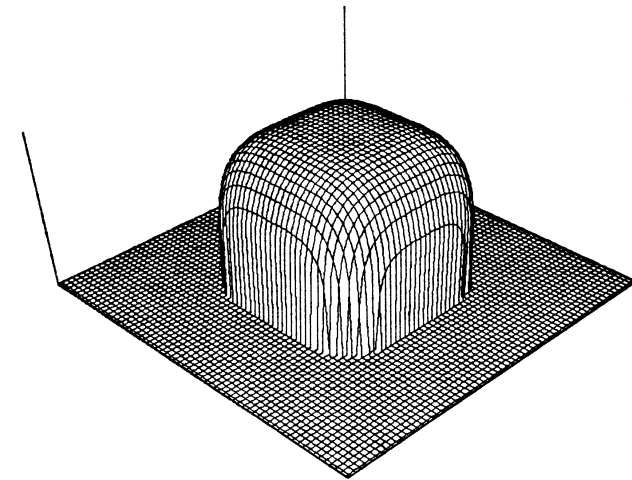
- where does “desired trajectory” come from?
- typically from end-effector level movement planning
 - then add an inverse kinematic...
 - which can be problematic
- alternative: planning and control in end-effector space

Operational space formulation

- in end-effector space add constraints as contributions to the “virtual forces”

- $$\begin{aligned} F_{x_d}^* &= -\text{grad}[U_{x_d}(x)], \\ F_o^* &= -\text{grad}[U_o(x)]. \end{aligned}$$

- $$\Lambda(x)\ddot{x} + \mu(x, \dot{x}) + p(x) = F$$



Optimal control

- given a plant $\dot{x} = f(x, u)$
- find a control signal $u(t)$
- that moves the state from an initial position $x_i(0)$ to a terminal position $x_f(t_f)$ within the time t_f
- a (difficult) planning problem!
- minimize a cost function to find such a signal

How does the human (or other animal) movement system generate movement?

- mechanics:... biomechanics
- actuator: muscle
- control? feedback loops