10/10/2021 Lecture 13

Last time :

- Lagrangian Mechanics

$$\mathcal{L} = T - V$$

$$\cancel{\frac{d}{dt}\,\dot{\mathcal{L}}} \qquad \frac{d}{dt}\frac{\partial \mathcal{L}}{\partial \dot{q}} - \frac{\partial \mathcal{L}}{\partial q} = f$$

- simulating pendulum

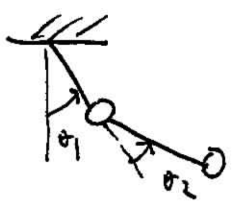
  PD-control

Today :

- a little more on Lagrangian
- basic control theory
- Matlab

Multi-link Lagrangian :

$I\omega$



$$\mathcal{L} = \sum T_i - \sum V_i$$

$$\frac{d}{dt}\begin{bmatrix} \frac{\partial \mathcal{L}}{\partial \dot{q}_1} \\ \vdots \\ \frac{\partial \mathcal{L}}{\partial \dot{q}_n} \end{bmatrix} - \begin{bmatrix} \frac{\partial \mathcal{L}}{\partial q_1} \\ \vdots \\ \frac{\partial \mathcal{L}}{\partial q_n} \end{bmatrix} = \begin{bmatrix} \tau_1 \\ \vdots \\ \tau_n \end{bmatrix}$$

Joint Space Control

- Goal $\lim_{t \to \infty} \vec{q}(t) = \vec{q}_d$

- error $\vec{e}(t) = \vec{q}_d - \vec{q}(t)$

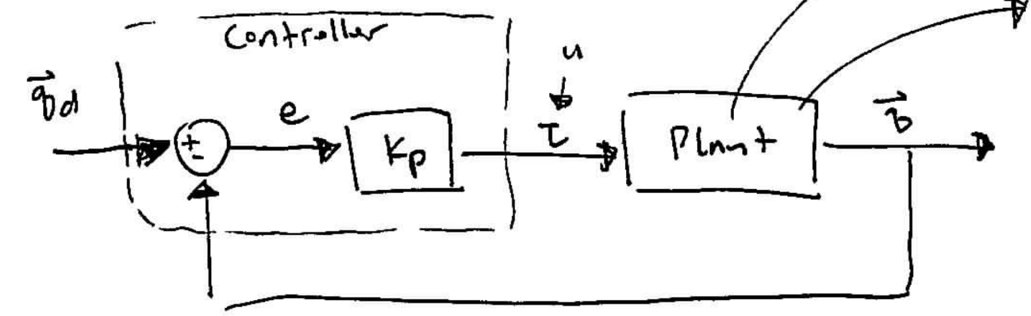
- regulation problem

$$\lim_{t \to \infty} \vec{e}(t) = 0$$

Proportional Control

Robot

Forward Dynamics



$$u = \begin{bmatrix} K_{p_1} & & 0 \\ & \ddots & \\ 0 & & K_{p_n} \end{bmatrix} \begin{bmatrix} q_{d_1} - q_1 \\ \vdots \\ q_{d_n} - q_n \end{bmatrix} \Longrightarrow K_p \vec{e}$$

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) = \tau - J^T(q) F_{ext}$$

$$\ddot{q} = \frac{1}{M(q)}\left[-C(q,\dot{q})\dot{q} - g(q) - J^T F_{ext} + \tau\right]$$

How can we simulate this in Matlab?

$$\bar{q} = \begin{bmatrix} q \\ \dot{q} \end{bmatrix} = \begin{matrix} q_1 \\ \vdots \\ q_n \end{matrix}$$

System of 1st ODE's

$$\dot{\bar{q}} = \begin{bmatrix} \cancel{\dot{q}} \quad \bar{q}(\overbrace{(1:n)+n}^{n+1:2n}) \\ \underbrace{\frac{1}{M(q)}\left[-C(q,\dot{q})\dot{q} - g(q) - J^T F_{ext} + \tau\right]} \end{bmatrix}$$

$$\ddot{q} = \text{forwardDynamics}(q, \dot{q}, \tau, F_{ext})$$

$$\bar{q} = \begin{bmatrix} q_1 \\ \vdots \\ q_n \\ \dot{q}_1 \\ \vdots \\ \dot{q}_n \end{bmatrix} \qquad \bar{q}(1:n) = \begin{bmatrix} q_1 \\ \vdots \\ q_n \end{bmatrix}$$

$$\bar{q}((1:n)+n) = \begin{bmatrix} \dot{q}_1 \\ \vdots \\ \dot{q}_n \end{bmatrix}$$

$\text{dynamics} = @(t, q, \tau, F_{ext})$

$1 \times 2n$ vector

$[\, q((1:n)+n)) ;$

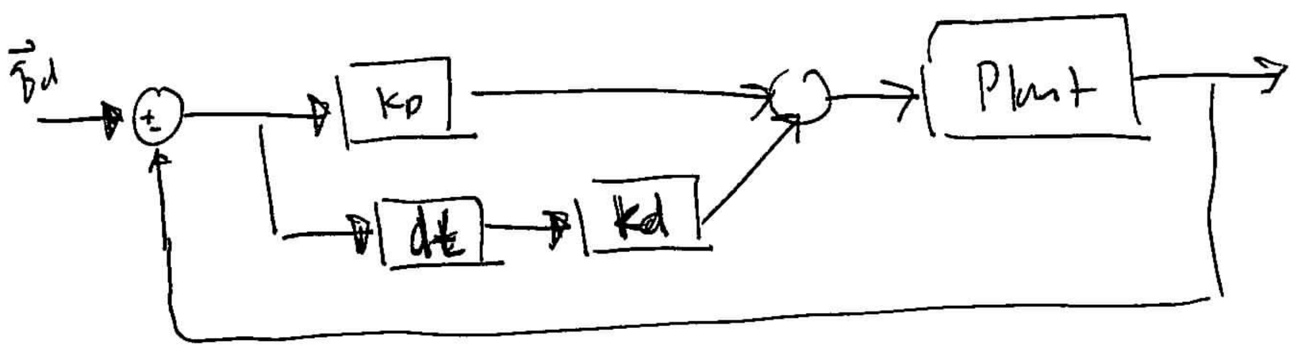
$\text{forwardDynamics}(\cdots)]$

$[t, q] = \text{ode23tb}(@(t,q)\ \text{dynamics}(t, q, \cdots), t, q_0)$

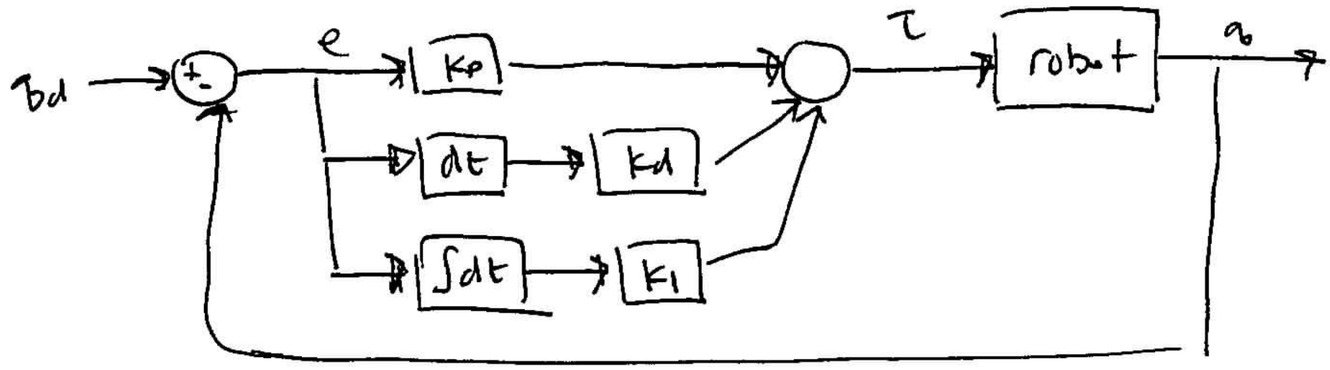
$\cancel{\text{Tau} = @(t, q, K_p)}$

$\text{Tau} = @(t, q, q_d, K_p) \quad (q_d - q(1:n)) \cdot K_p$

$[t, q] = \text{ode23tb}(@(t,q)\ \text{dynamics}(t, q, \tau(t, q, q_d, k_p) \cdots$



$\text{Tau} = @(t, q, q_d, K_p, K_d) \quad [q_d - q(1:n)] \cdot K_p$
$+ \left[-q((1:n)+n) \right] \cdot K_d$



How can we simulate this?

$$\bar{e} = \int_0^t e \, dt$$

then

$$\dot{e} = ? \quad e$$

$$\frac{d}{dt}\begin{bmatrix} \bar{q} \\ \bar{e} \end{bmatrix} = \begin{bmatrix} \bar{q}((1:n)+n) \\ \text{forwardDynamics}(\ldots) \\ q_d - q(1:n) \end{bmatrix}$$

~~tau@~~ tau = @(t, q, qd, Kp, Kd, Ki)

$(q_d - q(1:n)) \cdot K_p$

$(-q((1:n)+n)) \cdot K_d$

$(q((1:n)+2n)) \cdot K_I$