TAMPERE UNIVERSITY OF TECHNOLOGY

Robot Motion Control

Reza Ghabcheloo

IHA 4506 Advanced Robotics

Main References

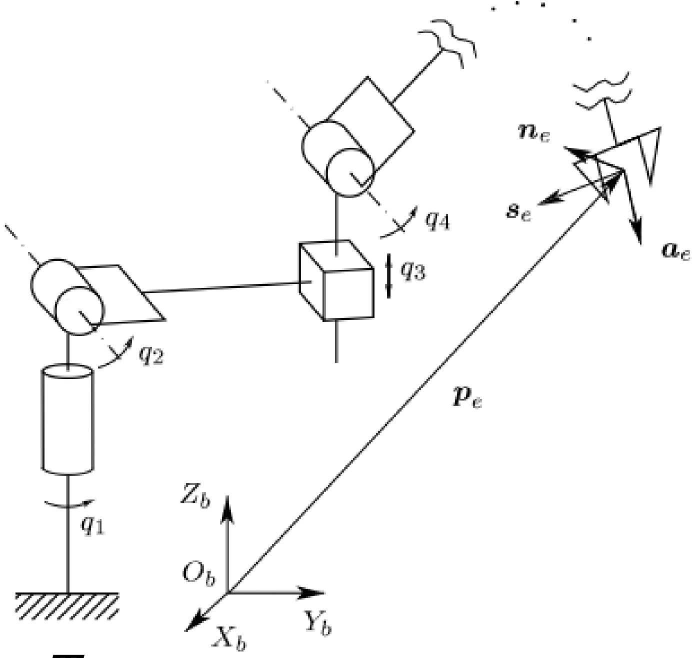
1. Siciliano et al. Robotics: modeling, planning and control, Springer 2009- Chapter 3 and 8

Content

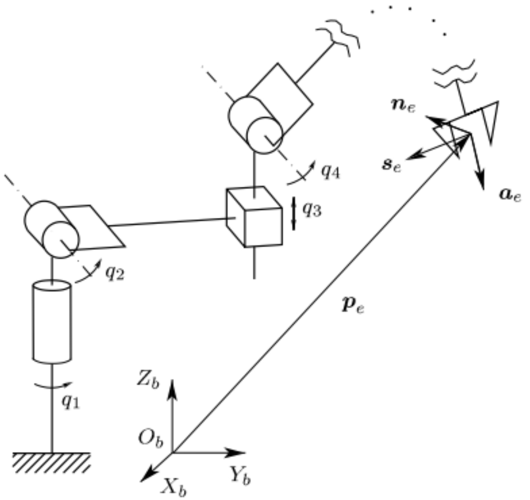
- Robot dynamic models
- Kinematics, Jacobian, Analytical Jacobian and terminologies
- Lyapunov stability and inverse kinematic examples
- Inverse dynamic motion control

Terminology

- Open chain robot
- End effector position p_e
- End effector orientation R_e
- Joint variables $q = (q_1, \dots, q_n)^T$
- Forward kinematic $p_e = p_e(q), R_e = R_e(q)$



Terminology



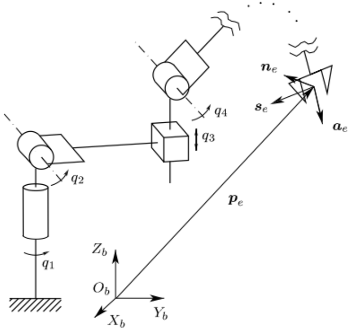
- Joint velocities $\dot{q}$
- EE linear vel $\dot{p_e}$
- EE angular vel $\omega_e, \dot{R}_e = S(\omega_e)R_e$
- Differential kinematics $v_e = \begin{bmatrix} \dot{p_e} \\ \omega_e \end{bmatrix}, v_e = J(q)\dot{q}$
- Geometric Jacobian matrix $J(q)$
- Forward kinematics $x_e = \begin{bmatrix} p_e \\ \phi_e \end{bmatrix} = \begin{bmatrix} k_P(q) \\ k_O(q) \end{bmatrix} = k(q)$

Orientation representation

- Euler angles
- Rotation matrix
- Axis angle
- Quaternions
- …

Modeling / Dynamics

Lagrangian
Newton-Euler



Joint space dynamic model

$$B(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) + f(q,\dot{q}) = \tau - J^T(q)h_e$$

Coriolis and centrifugal effects $C(q,\dot{q})$

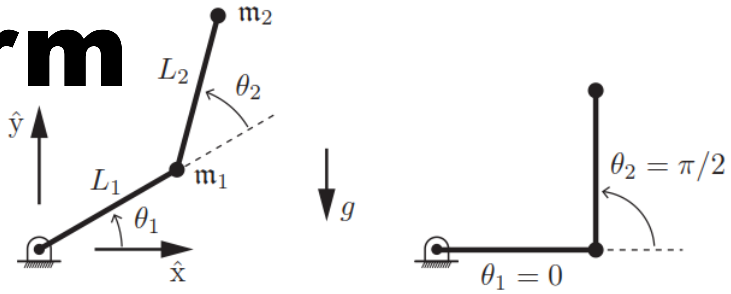
Gravity component $g(q)$

Friction/non-conservative forces. For example $f(q,\dot{q}) = F\dot{q}$

Force and torque exerted by the EE to environment h_e

Actuator torque/forces τ

Two-link planar arm



- $B(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) + f(q,\dot{q}) = \tau - J^T(q)h_e$
- $q = (\theta_1, \theta_2)^T$

$$M(\theta) = \begin{bmatrix} \mathfrak{m}_1 L_1^2 + \mathfrak{m}_2(L_1^2 + 2L_1L_2\cos\theta_2 + L_2^2) & \mathfrak{m}_2(L_1L_2\cos\theta_2 + L_2^2) \\ \mathfrak{m}_2(L_1L_2\cos\theta_2 + L_2^2) & \mathfrak{m}_2L_2^2 \end{bmatrix},$$

$$C(\theta,\dot{\theta})\dot{\theta} = \begin{bmatrix} -\mathfrak{m}_2L_1L_2\sin\theta_2(2\dot{\theta}_1\dot{\theta}_2 + \dot{\theta}_2^2) \\ \mathfrak{m}_2L_1L_2\dot{\theta}_1^2\sin\theta_2 \end{bmatrix},$$

$$g(\theta) = \begin{bmatrix} (\mathfrak{m}_1 + \mathfrak{m}_2)L_1g\cos\theta_1 + \mathfrak{m}_2gL_2\cos(\theta_1+\theta_2) \\ \mathfrak{m}_2gL_2\cos(\theta_1+\theta_2) \end{bmatrix},$$

Notable properties/skew symmetric

$$B(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) + f(q,\dot{q}) = \tau - J^T(q)h_e$$

Define $N = \dot{B} - 2C$

1) $\dot{q}^T N \dot{q} = 0$ (valid for all choices of C)

2) Writing C using *Chrostoffel* symbols leads to **skew symmetric** N, that is, $w^T N w = 0, \forall w$

Notable properties/linearity in parameters

$$B(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) + f(q,\dot{q}) = \tau$$
$$Y(q,\dot{q},\ddot{q})\pi = \tau$$

π a vector of dynamic paramters $p \times 1$

$$\boldsymbol{\pi}_i = [\, m_i \;\; m_i\ell_{C_ix} \;\; m_i\ell_{C_iy} \;\; m_i\ell_{C_iz} \;\; \widehat{I}_{ixx} \;\; \widehat{I}_{ixy} \;\; \widehat{I}_{ixz} \;\; \widehat{I}_{iyy} \;\; \widehat{I}_{iyz} \;\; \widehat{I}_{izz} \;\; I_{m_i} \,]^T$$

Usually 13 parameters per joint: masses/inertia, length, friction coefficients

$Y(q,\dot{q},\ddot{q})$ called regressor (contains no parameters)

Modeli ... nal space

Although you do not need to know how to derive these models, you need to know their size, meaning/ properties, expressed frames,

$$x_e = \begin{bmatrix} p_e \\ \phi_e \end{bmatrix} \quad v_e = \begin{bmatrix} \dot{p}_e \\ \omega_e \end{bmatrix}$$

$$B_A(x_e)\ddot{x}_e + C_A(x_e, \dot{x}_e)\dot{x}_e + g_A(x_e) = \gamma_A - h_A$$

$$\boldsymbol{B}_A = (\boldsymbol{J}_A \boldsymbol{B}^{-1} \boldsymbol{J}_A^T)^{-1}$$

$$\boldsymbol{C}_A \dot{\boldsymbol{x}}_e = \boldsymbol{B}_A \boldsymbol{J}_A \boldsymbol{B}^{-1} \boldsymbol{C} \dot{\boldsymbol{q}} - \boldsymbol{B}_A \dot{\boldsymbol{J}}_A \dot{\boldsymbol{q}}$$

$$\boldsymbol{g}_A = \boldsymbol{B}_A \boldsymbol{J}_A \boldsymbol{B}^{-1} \boldsymbol{g},$$

$$\boldsymbol{T}_A^T(\boldsymbol{x}_e)\boldsymbol{\gamma}_e = \boldsymbol{\gamma}_A \qquad \boldsymbol{T}_A^T(\boldsymbol{x}_e)\boldsymbol{h}_e = \boldsymbol{h}_A$$

$$\boldsymbol{v}_e = \begin{bmatrix} \boldsymbol{I} & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{T}(\boldsymbol{\phi}_e) \end{bmatrix} \dot{\boldsymbol{x}}_e = \boldsymbol{T}_A(\boldsymbol{\phi}_e)\dot{\boldsymbol{x}}_e$$

$$\boldsymbol{J} = \boldsymbol{T}_A(\boldsymbol{\phi})\boldsymbol{J}_A \qquad v_e = J\dot{q}$$

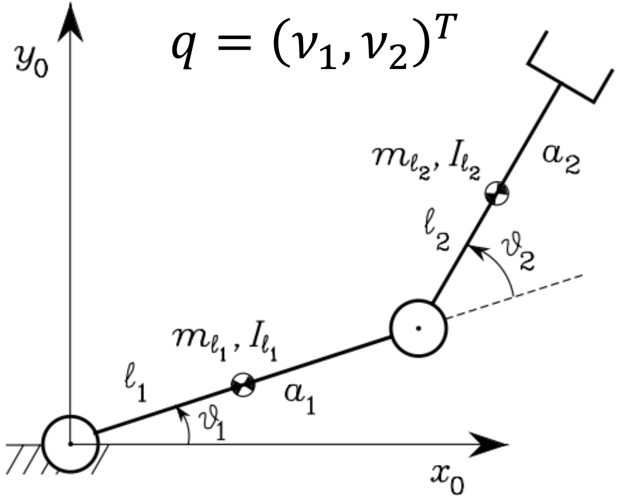
Analytical Jacobian J_A, Geometric Jacobian J,

- Geometric Jacobian used when handling variable with clear physical meanings, while Analytical Jacobian when using differential quantities

Two-link planar arm

- Jacobians up to certain link

$q = (\nu_1, \nu_2)^T$



$$\boldsymbol{J}_P^{(\ell_1)} = \begin{bmatrix} -\ell_1 s_1 & 0 \\ \ell_1 c_1 & 0 \\ 0 & 0 \end{bmatrix} \qquad \boldsymbol{J}_P^{(\ell_2)} = \begin{bmatrix} -a_1 s_1 - \ell_2 s_{12} & -\ell_2 s_{12} \\ a_1 c_1 + \ell_2 c_{12} & \ell_2 c_{12} \\ 0 & 0 \end{bmatrix},$$

$$\boldsymbol{J}_O^{(\ell_1)} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \end{bmatrix} \qquad \boldsymbol{J}_O^{(\ell_2)} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 1 \end{bmatrix}$$

$$\dot{p}_{l1} = J_P^{l_1}\dot{q}, \dot{\omega}_{l1} = J_O^{l_1}\dot{q}, \ldots$$

Check (7.16) to (7.29) for further details

Direct dynamics (Simulation)

Given $\tau(t), h_e(t), q(0), \dot{q}(0)$, what is $q(t)$

- $B\ddot{q} + C\dot{q} + g + f = \tau - J^T h_e$
- $\tau_q(q, \dot{q}, h_e) = C(q, \dot{q})\dot{q} + g(q) + f(q, \dot{q}) + J^T(q)h_e$
- $\ddot{q} = B^{-1}(q)(\tau - \tau_q)$
- Example: a Simulink block with (appropriate solver)
  – input $q, \dot{q}, h_e, \tau$
  – output $\ddot{q}$ → integrate twice to produce $q, \dot{q}$

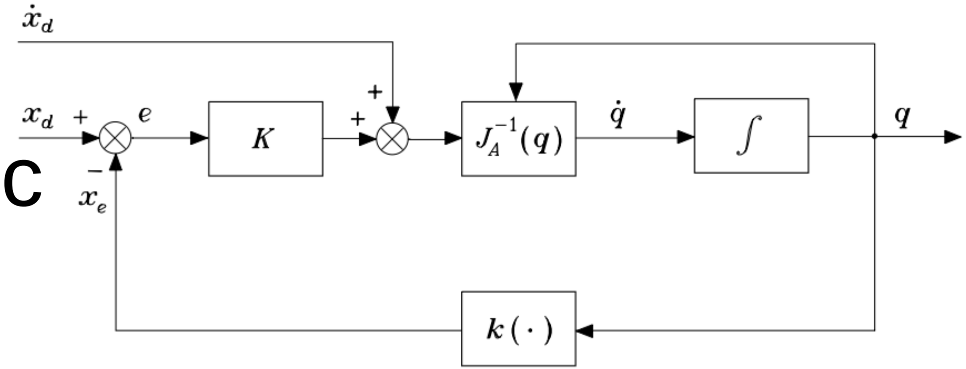
Inverse dynamic

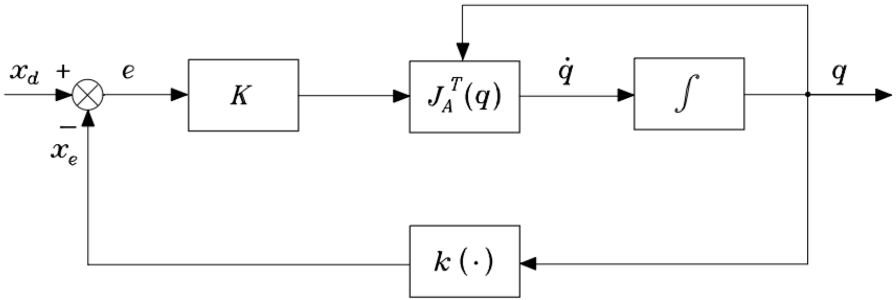
Given $q(t), h_e(t)$, what is $\tau(t)$

- $B\ddot{q} + C\dot{q} + g + f = \tau - J^T h_e$
- Used for trajectory generation and robot control
- Usually: $x_e(t) \xrightarrow{\text{Inverse Kin.}} q(t) \xrightarrow{\text{Inverse Dyn.}} \tau(t)$
- O(n^2): Direct dynamics
- O(n): Inverse dynamics

Inverse Kinematics Asymptotic solution

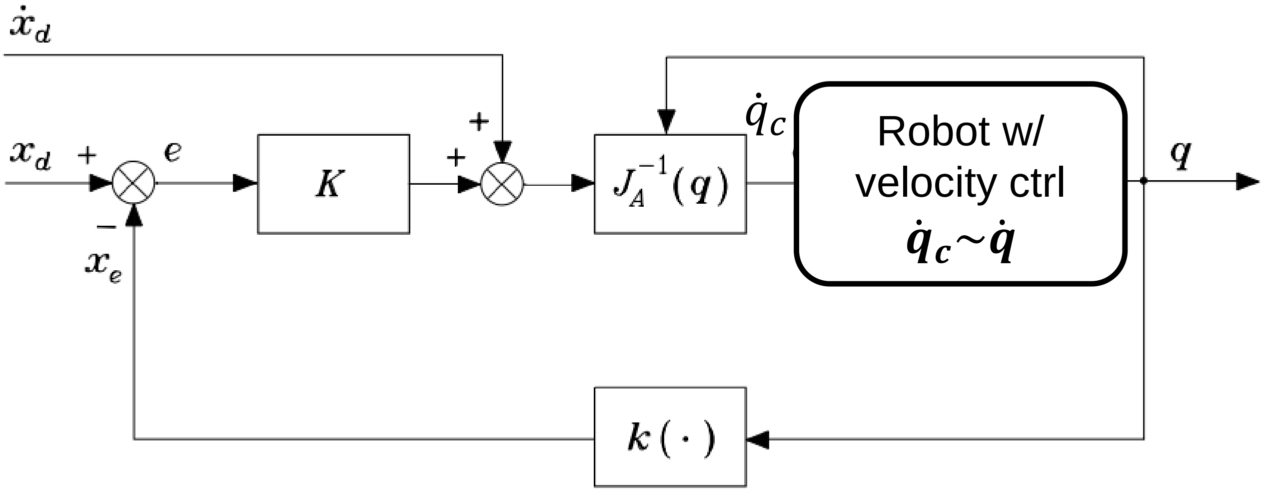
assuming J_A is square and symmetric



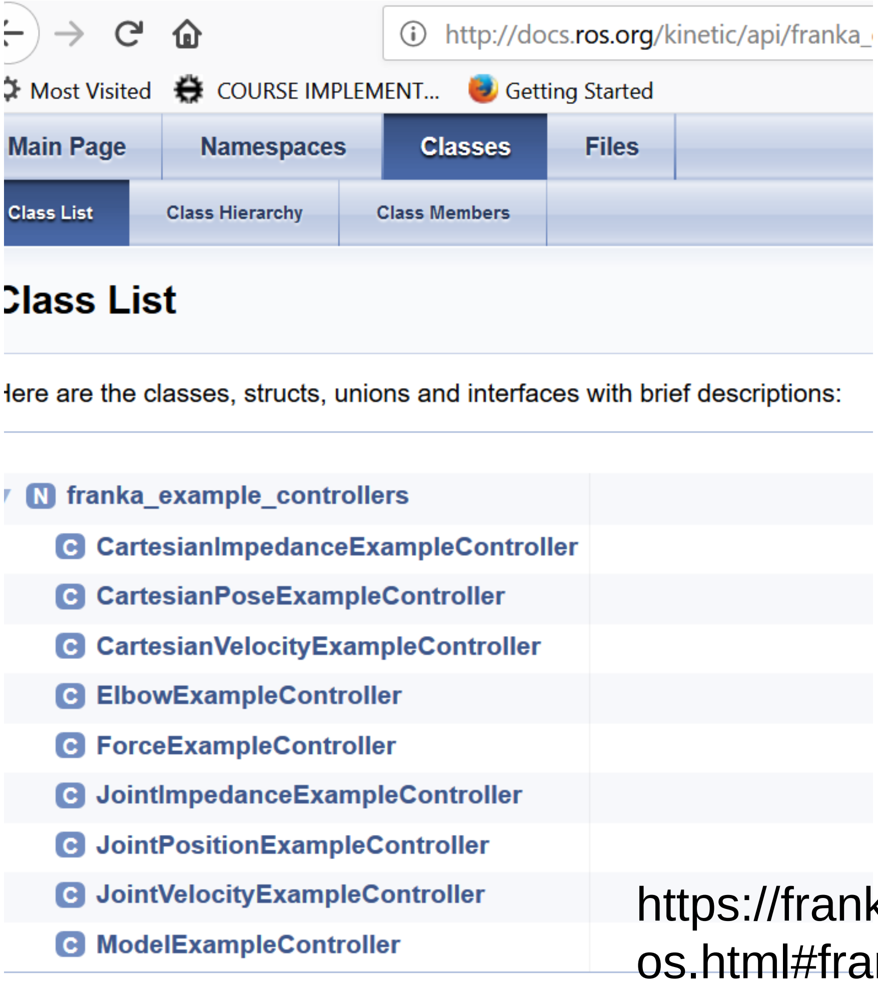


$k(.)$ is direct kinematic function

Kinematic control

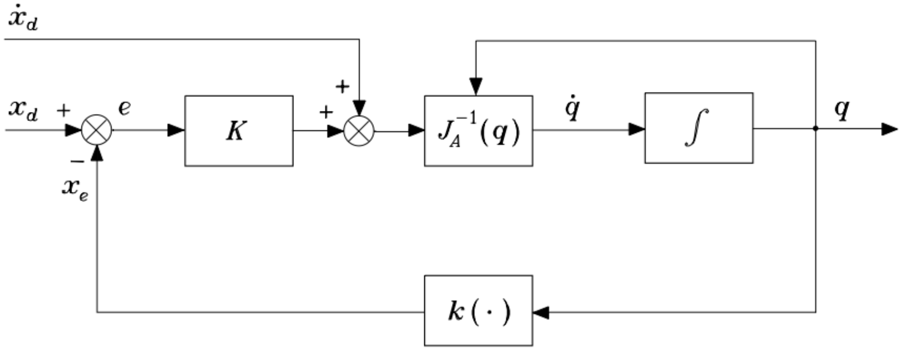


Franka Panda



https://frankaemika.github.io/docs/franka_ros.html#franka-example-controllers

Proof



- $e = x_d - x_e$
- $\dot{q} = J_A^{-1}(\dot{x}_d + Ke) \rightarrow \dot{e} + Ke = 0, K > 0$

Exponential stable, that is, $e \rightarrow 0$ or remains zero if it is initially zero.

NOTE: computation of x_d and $\dot{x}_d$ for orientation may be involved depending on the choice of orientation representation and corresponding error

Stability
Lyapunov direct method

- Given a nonlinear system $\dot{e} = f(e)$, with $f(0) = 0$ (*equilibrium*)
- If there is a function $V(e)$, such that

$$V(e) > 0, e \neq 0$$

$$V(e) = 0 \Leftrightarrow e = 0 \quad \& \quad f(e).\nabla V(e) < 0, \forall e \neq 0$$

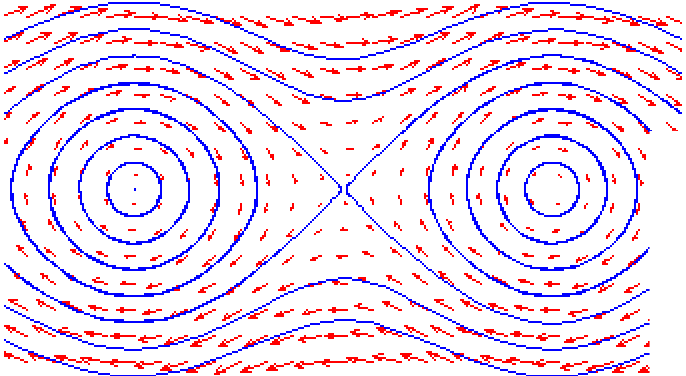
$$V(e) \to \infty, \|e\| \to \infty$$

- Then origin ($e = 0$) is globally asymptotically stable equilibrium of $\dot{e} = f(e)$

Example: $V(e) = e^T e$

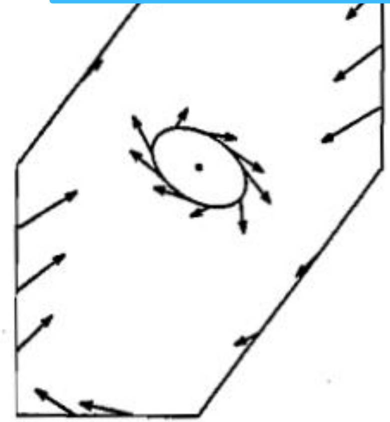
- *There are several variations. I will present the simplest form to give the idea. Check H Khalil Nonlinear Systems*
- *Appendix C.3.*

Intuition behind Lyapunov stability

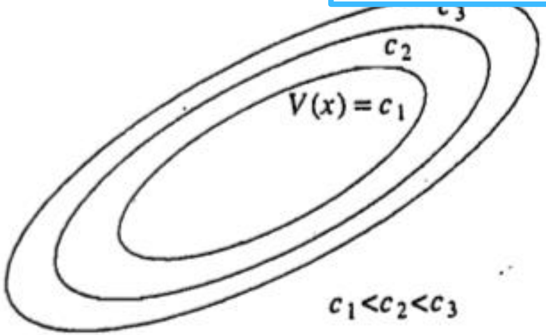


$f(e)$
Frictionless pendulum vector field

If at the boundaries $f(e).\nabla V(e) < 0$, then system trajectories cannot leave the boundary



$V(e) > 0$
Closed level curves



Example

- $\dot{e} = -Ke,\ \text{with}\ K > 0, K = K^T, \left(e = 0 \Rightarrow \dot{e} = 0\right)$

$$V(e) = \frac{1}{2}e^T e \rightarrow \dot{V} = e^T \dot{e} = -e^T K e < 0$$

- $\dot{e} = Ae$ under what conditions this is stable?

Example $\dot{e} = Ae$

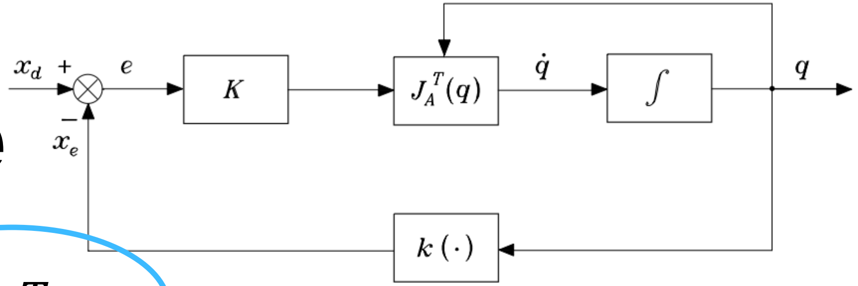
- $V(e) = \frac{1}{2} e^T P e, P > 0$

$\dot{V} = \frac{1}{2} e^T P \dot{e} + \frac{1}{2} \dot{e}^T P e = \frac{1}{2} e^T (PA + A^T P) e = -\frac{1}{2} e^T Q e$, if $Q > 0$

- $PA + A^T P = -Q$ the famous Lyapunov equations. Usually we do not know P.
- There are solutions for this: given A and $Q > 0$ find $P > 0$

P exists if A stable (all eigenvalues have negative real part)

Examle, J_A^T case



- $\dot{e} = \dot{x}_d - \dot{x}_e = \dot{x}_d - J_A\dot{q}$, $\dot{q} = J_A^T K e$,
- $V(e) = \frac{1}{2} e^T K e \Rightarrow \dot{V} = e^T K \dot{e}$

The gradient method for solving inverse kinematic

- Simple case of $\dot{x}_d = 0$, or $\dot{e} = -J_A J_A^T K e$

$\dot{V} = -e^T K J_A J_A^T K e < 0 \Rightarrow \mathrm{e} \to 0$

- When $\dot{x}_d \neq 0$, there will be some error
- We could start with choosing V and then derive $\dot{q} = J_A^T K e$

Lyapunov positive definiteness

- State vector (x_1, x_2), with equilibrium $(0,0)$
- Mass-spring-damper $\dot{x} = v, \dot{v} = -\frac{k}{m}x - \frac{b}{m}v$
- Positive definite (a Lyapunov function)?

$V = x_1^2$?

$V = x_1^2 + x_1 x_2 + x_2^2$?

$V = (1 - \cos x_1) + x_2^2$?

Dynamic motion control

- High gear ration and low speeds and accelerations: independent joint control
- We will skip the details of decentralized control, there are lots of *art* there.
- Straight to centralized control

Simple case of PD control + Gravity compensation

- $B(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) + f(q,\dot{q}) = u$

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

- $\tilde{q} = q_d - q$

q_d constant for now

- Error state $(\tilde{q},\dot{q})$
- Objective $(\tilde{q},\dot{q}) \rightarrow 0$

Lyapunov direct method

$$V(\dot{q},\widetilde{q}) = \frac{1}{2}\dot{q}^T B(q)\dot{q} + \frac{1}{2}\widetilde{q}^T K_P \widetilde{q} > 0 \qquad \forall \dot{q},\widetilde{q} \neq 0$$

$$\dot{V} = \dot{q}^T B(q)\ddot{q} + \frac{1}{2}\dot{q}^T \dot{B}(q)\dot{q} - \dot{q}^T K_P \widetilde{q}.$$

$$\dot{V} = \frac{1}{2}\dot{q}^T\left(\dot{B}(q) - 2C(q,\dot{q})\right)\dot{q} - \dot{q}^T F\dot{q} + \dot{q}^T\left(u - g(q) - K_P\widetilde{q}\right)$$

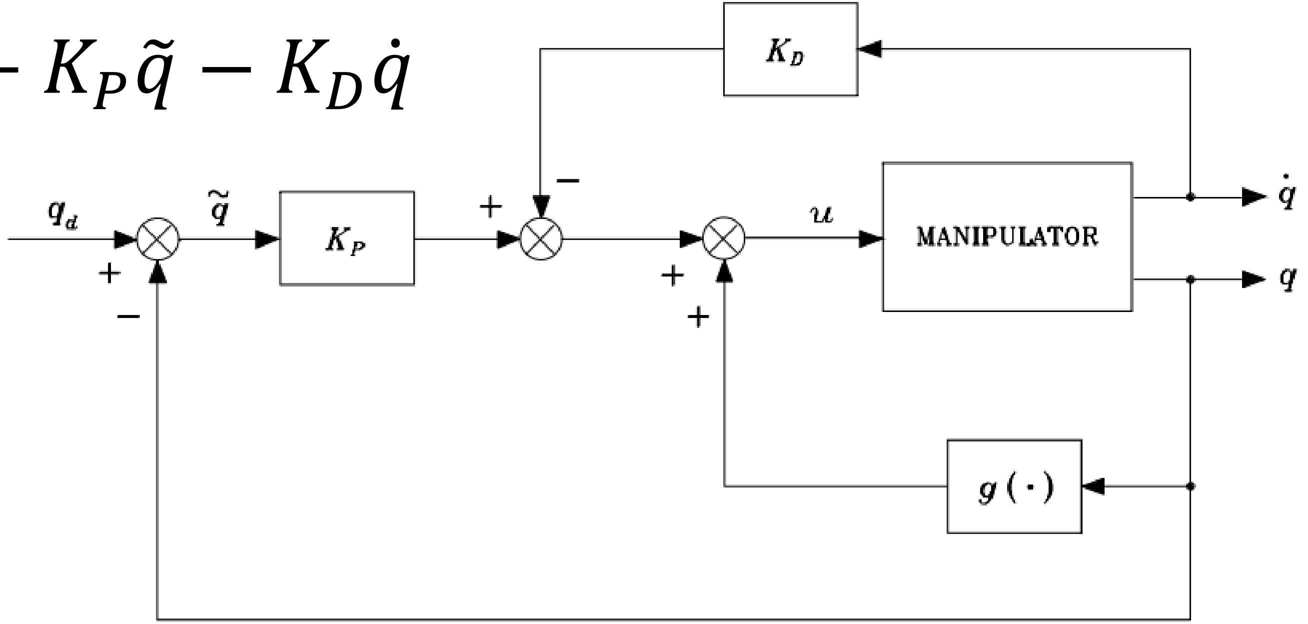
Choose u such that to make the $\dot{V}$ negative

$$u = g(q) + K_P\widetilde{q} - K_D\dot{q}$$

$$\dot{V} = -\dot{q}^T(F + K_D)\dot{q}$$

PD control + gravity compensation

$$u = g(q) + K_P\tilde{q} - K_D\dot{q}$$



Inverse dynamics control

Robot model

$$B(q)\ddot{q} + n(q,\dot{q}) = u, \qquad n(q,\dot{q}) = C(q,\dot{q})\dot{q} + F\dot{q} + g(q).$$

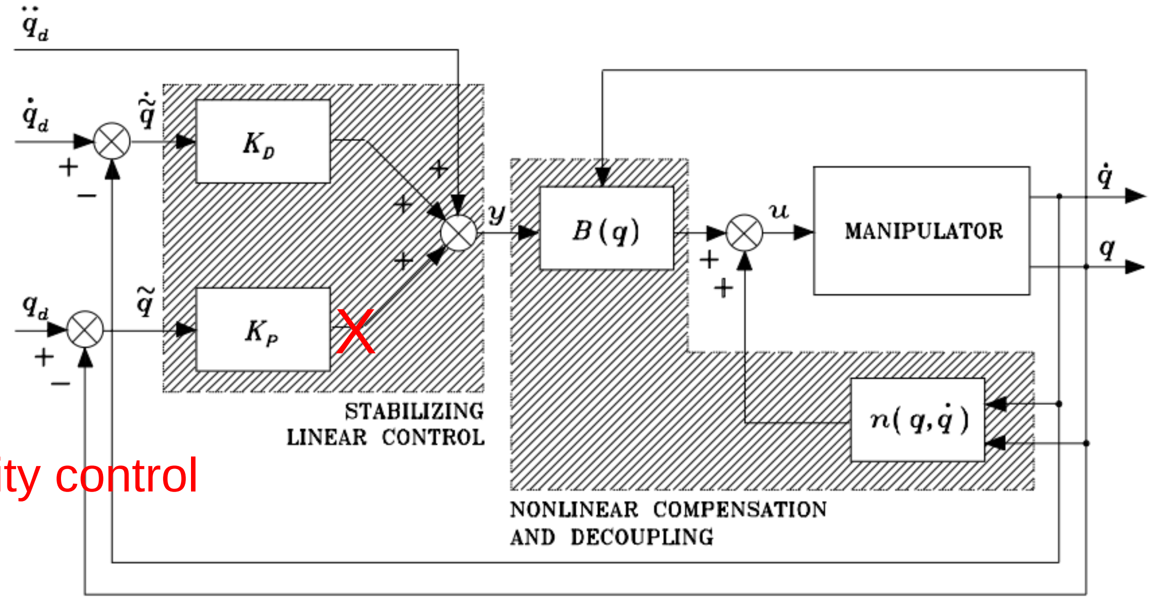
Feedback linearization control loop

$$u = B(q)y + n(q,\dot{q})$$

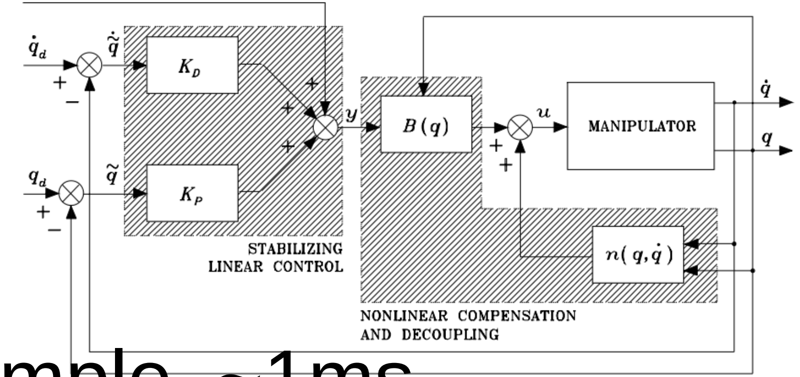
Closed-loop equivalent linear system $\ddot{q} = y$

→velocity control

$$\ddot{\tilde{q}} + K_D\dot{\tilde{q}} + \cancel{K_P\tilde{q}} = 0$$

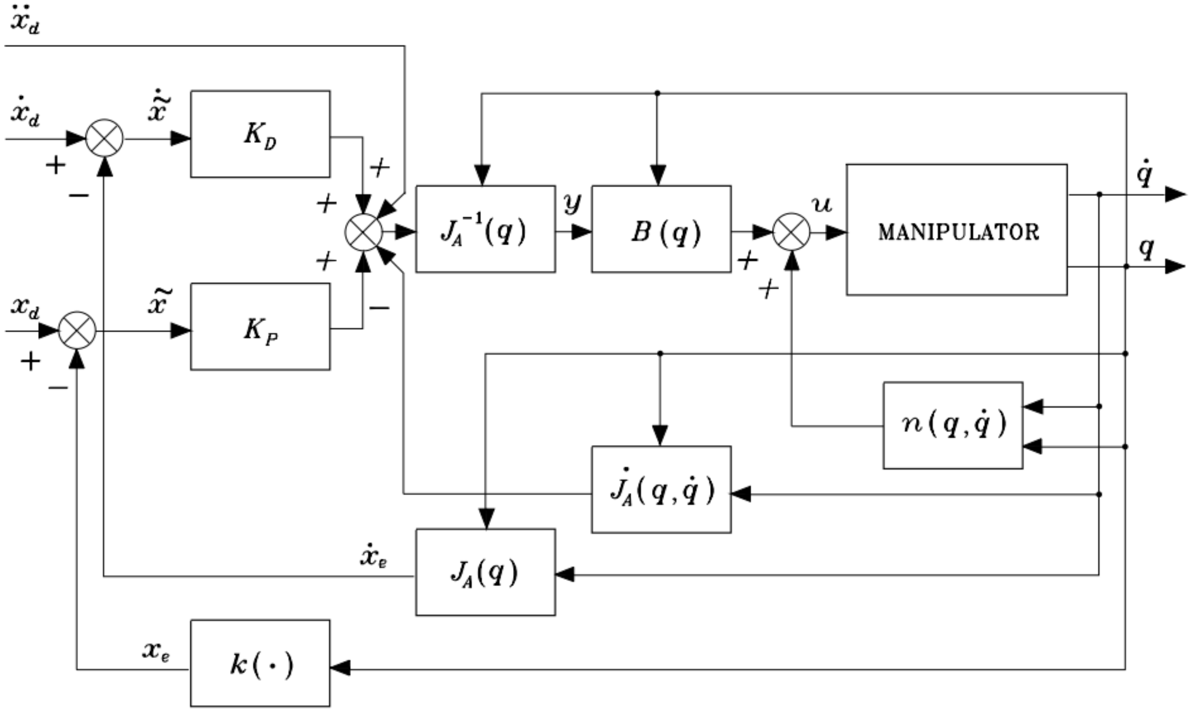


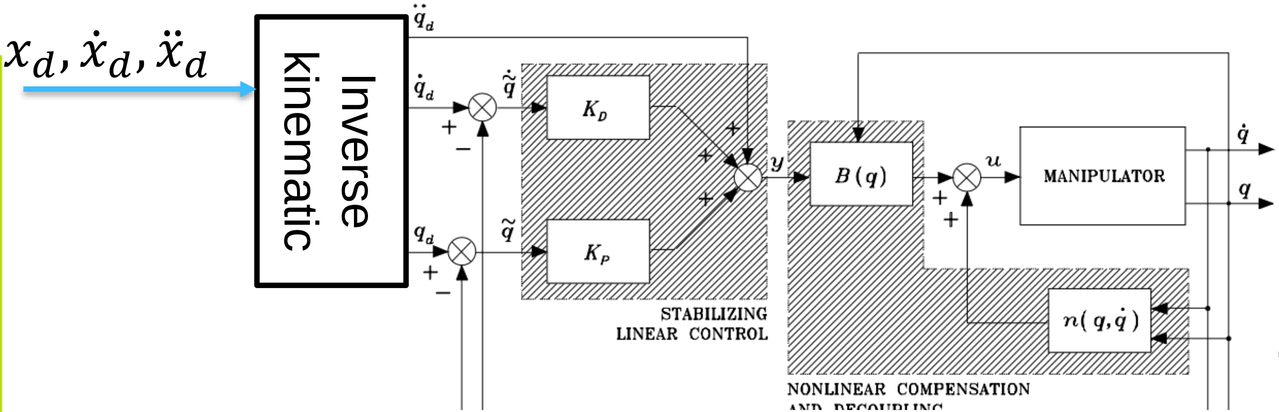
Inverse dynamics control

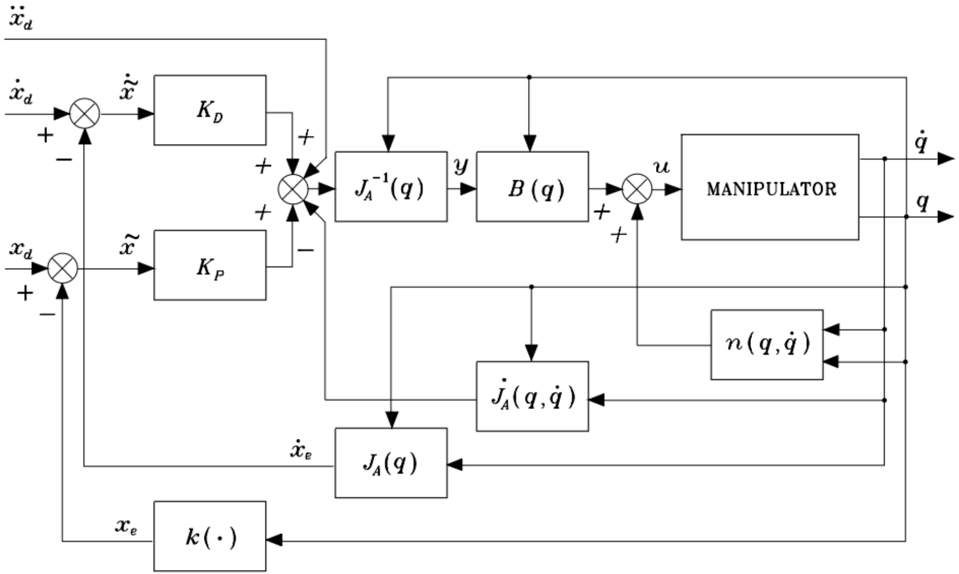


- Exact knowledge of $B(q)$ and $n(q, \dot{q})$
- To be computed every sample, ~1ms
- Robust and adaptive control technics have been developed to address parameter uncertainty and unmodeled dynamics

Inverse dynamics, task space





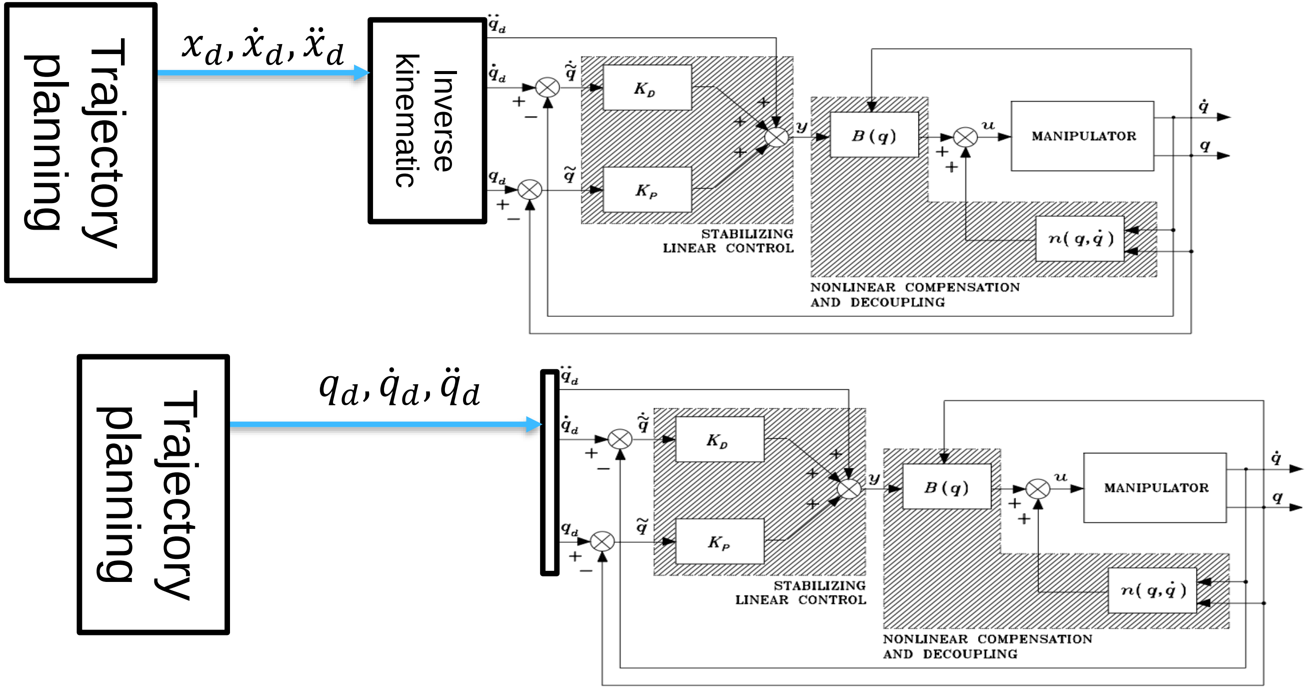


Two different ways to control the robot, given task space trajectories

Compare this with Kinematic control!

Trajectory planning

Input to the algorithm
Initial $x_d(0), \dot{x}_d(0), ..$
Goal $x_d(1), \dot{x}_d(1), ..$
Robot constraints
Obstacle map
…



Trajectory planning

- Chapter 4, Reza’s video lectures
- Sample based planning: RRT, etc + smoothing
- Learning from Demonstration (Behavior Cloning)
- Reflexxes lib: A good example what a reactive motion generator need to take into account

Topics

- Singularities
- Redundancies

Exercise

1. Experiment with “writing your own controller”

https://frankaemika.github.io/docs/franka_ros.html#writing-your-own-controller

2. Implement kinematic controller

2. Implement the reactive obstacle avoidance