Robotic Manipulator II: Control

ENGR 7401 – Kinematic Control of Manipulators

(Lecture 4)

Prof. Walter Lucia

Lecture Outline

Dynamical Systems

- Equilibrium Points
- Stability

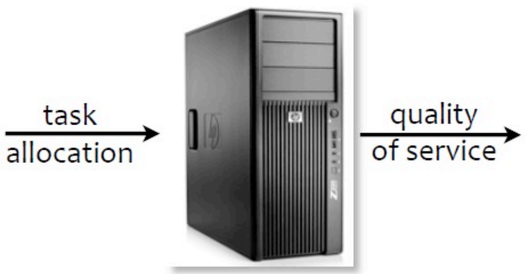
Kinematics Control Algorithms

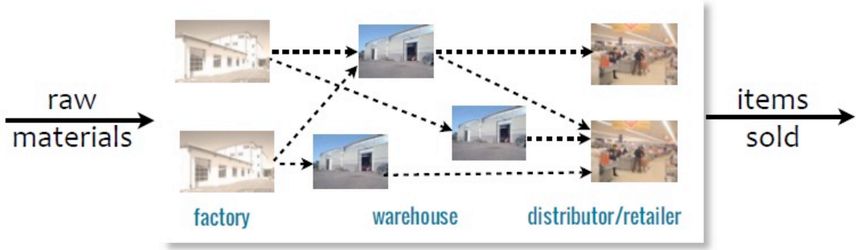
- Jacobian Inverse
- Jacobian Transpose

What is a Dynamical System

What is a Dynamical System

- A dynamical system is an entity that evolves over time. The way the system evolves is called the dynamics of the system:
  - Examples: An engine, a nuclear reactor, a human transporter, a supply chain, a computer server, etc.





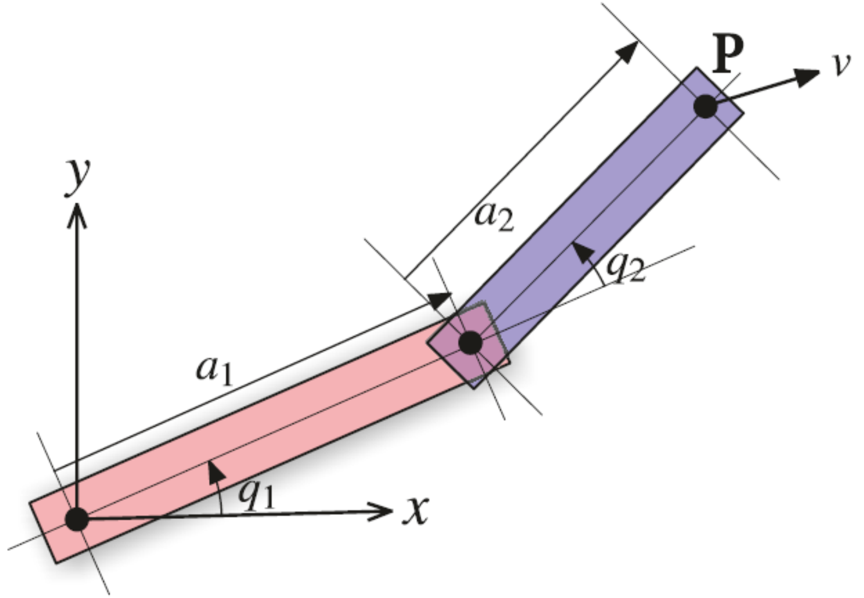
Have we seen any dynamical system so far?

Kinematics model of Manipulators

- Differential kinematics Model

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

- This equations describes how the robot moves according to the joints velocities. The equation tells us that the robot is a dynamical system subject to external excitation (joint velocities in this case)



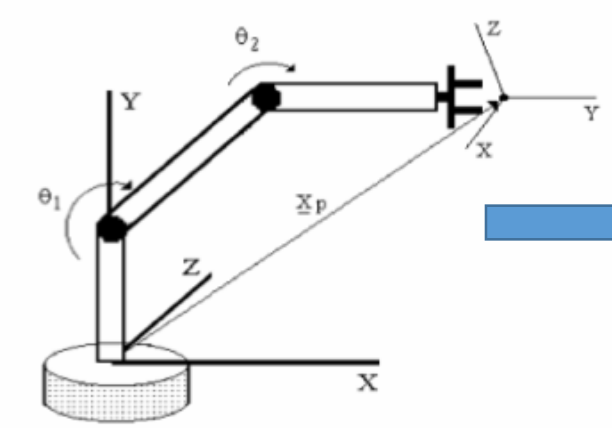
Why is important to have a dynamical model

- Understand the system (“How does X influence Y ?”)
- Simulation (“If I apply X, I obtain Y”)
- Control (“How to make the system behave how we want ?”)

Dynamical model and Direct Chain

- Understand the system (“How does X influence Y ?”)
- Simulation (“If I apply X, I obtain Y”)

$\mathrm{u}(t)$



Joints variables

$\mathrm{q}(t)\ \dot{q}(t)$

Sensors

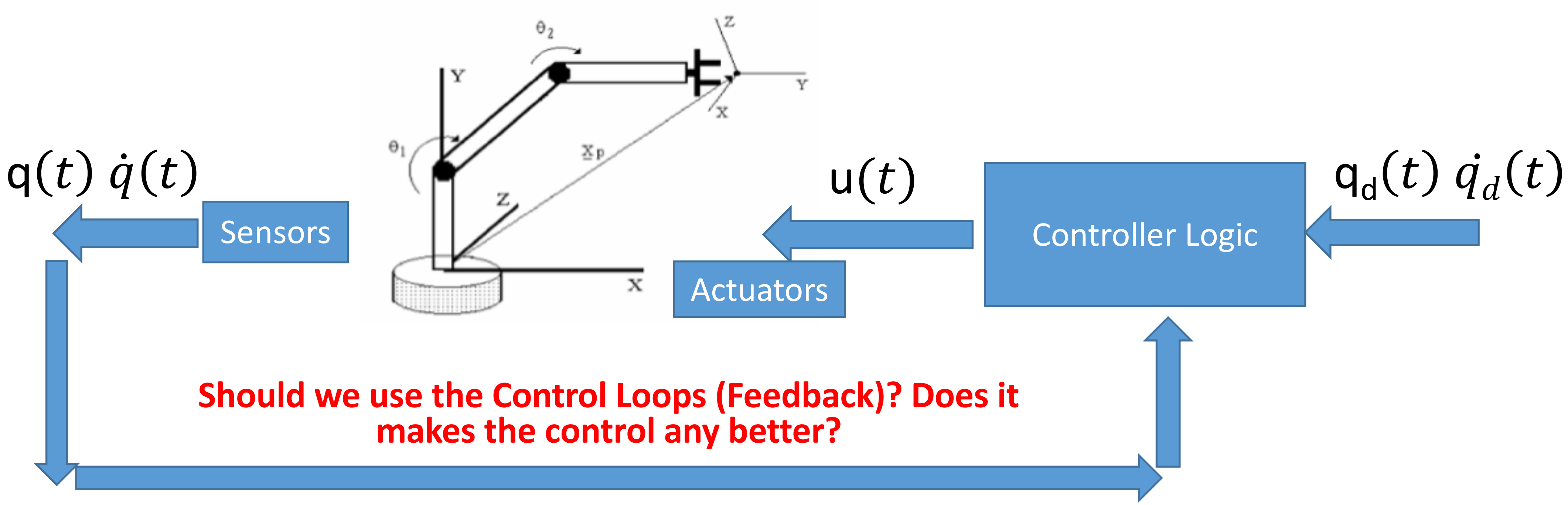
Kinematic Model

End Effector Position and Orientation

$\mathrm{P}, (\gamma, \theta, \phi)$

Dynamical model and Inverse Chain and Control

- Control ("How to make the system behave how we want ?")



Should we use the Control Loops (Feedback)? Does it makes the control any better?

Why is important to have a dynamical model

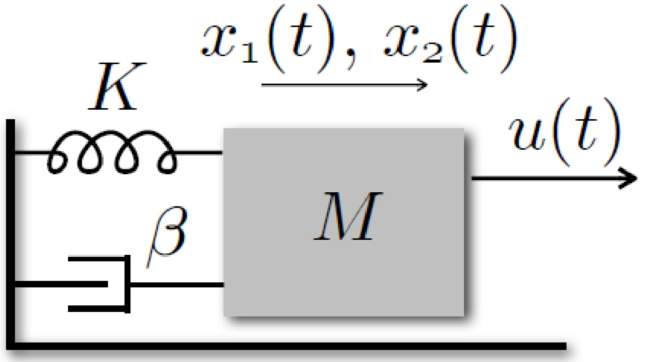
- Control (“How to make the system behave how we want ?”)

- There is not a generic magic solution that will tell us how to design the control system.
- We need to analyze the model we have (its properties) to understand how to design the control logic
  - This is true for robotic manipulators
  - This applies, in general, to any kind of dynamical system

Dynamical Systems Classification

Example of Dynamical System

$$x_1(t) = x(t), \quad x_2(t) = \dot{x}(t)$$



- Mass-Spring-Damper System

$$\begin{cases} \dot{x}_1(t) = x_2(t) \\ M\dot{x}_2(t) = u - \beta x_2(t) - K x_1(t) \end{cases}$$

$$y(t) = x_1(t)$$

$$\begin{cases} \frac{dx_1(t)}{dt} = x_2(t) \\ \frac{dx_2(t)}{dt} = -\frac{\beta}{M} x_2(t) - \frac{K}{M} x_1(t) + \frac{1}{M} u(t) \end{cases}$$

$$x(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}$$

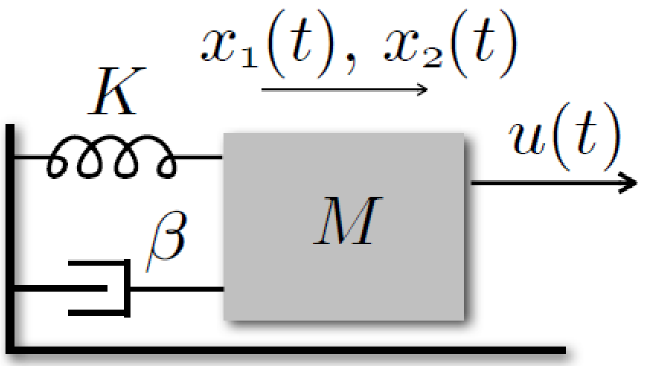
$$\dot{x}(t) = \underbrace{\begin{bmatrix} 0 & 1 \\ -\frac{K}{M} & -\frac{\beta}{M} \end{bmatrix}}_{A} x(t) + \underbrace{\begin{bmatrix} 0 \\ \frac{1}{M} \end{bmatrix}}_{B} u(t)$$

$$y(t) = \underbrace{\begin{bmatrix} 1 & 0 \end{bmatrix}}_{C} x(t)$$

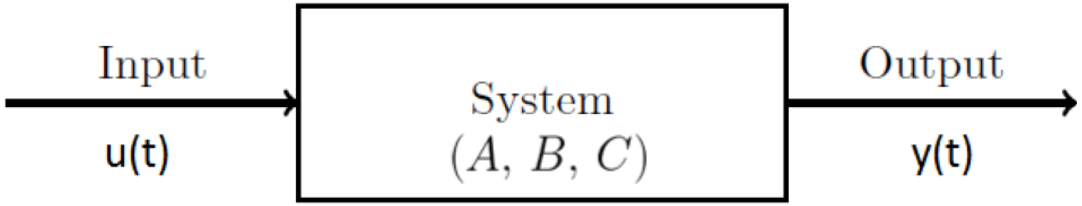
Example of Dynamical System

$$x_1(t) = x(t), \quad x_2(t) = \dot{x}(t)$$

- Mass-Spring-Damper System



$$\dot{x}(t) = \underbrace{\begin{bmatrix} 0 & 1 \\ -\frac{K}{M} & -\frac{\beta}{M} \end{bmatrix}}_{A} x(t) + \underbrace{\begin{bmatrix} 0 \\ \frac{1}{M} \end{bmatrix}}_{B} u(t)$$



$$y(t) = \underbrace{\begin{bmatrix} 1 & 0 \end{bmatrix}}_{C} x(t)$$

- This system is described by a dynamical model that is a system of Ordinary Linear Differential Equations

Some Classes of Dynamical Systems

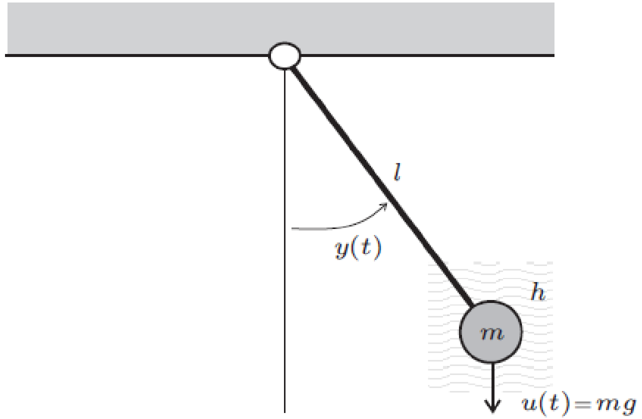
- **Linear Time-Invariant** (e.g. the Mass-Spring Damper System)

$$\begin{cases} \dot{x}(t) & = & Ax(t) + Bu(t) \\ y(t) & = & Cx(t) + Du(t) \end{cases}$$

- **Non-Linear Time-Invariant**
  - e.g. Inverted Pendulum

$$\begin{cases} \dot{x}(t) & = & f(x(t), u(t)) \\ y(t) & = & g(x(t), u(t)) \end{cases}$$

$x_1 = y$ and $x_2 = \dot{y}$



$$\begin{cases} \dot{x}_1 & = & x_2 \\ \dot{x}_2 & = & -\frac{g}{l}\sin x_1 - Hx_2, \qquad H \triangleq \frac{h}{ml^2} \end{cases}$$

Kinematics of a Manipulator

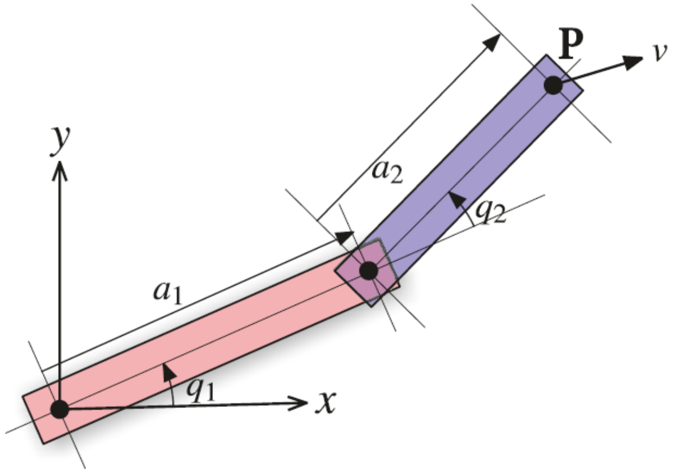
- Differential Kinematics $v(t) = J(q(t))\dot{q}(t)$

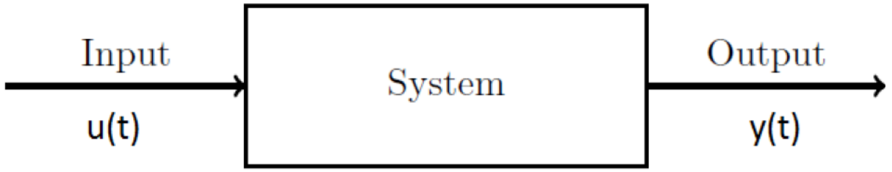
If $x_1(t) = q(t), \quad x_2(t) = p(t), \quad u(t) = \dot{q}(t)$

- It can be re-written as

$$\begin{cases} \dot{x}_1(t) = u(t) \\ \dot{x}_2(t) = J(x_1(t))u(t) \end{cases}$$

- Is a Linear or Non-Linear Model?

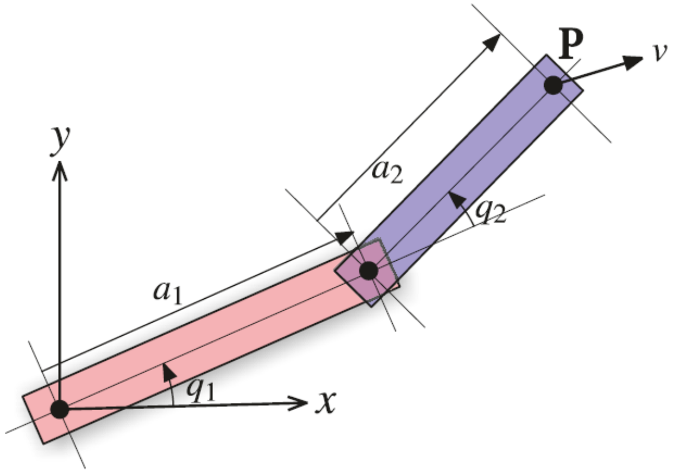




$$\dot{x}(t) = Ax(t) + Bu(t) \qquad \dot{x}(t) = f(x(t), u(t))$$

Kinematics of a Manipulator

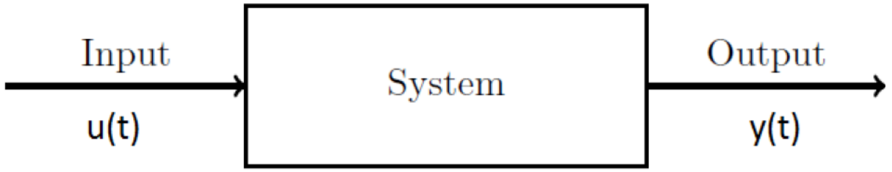
- Differential Kinematics $v(t) = J(q(t))\dot{q}(t)$



If $x_1(t) = q(t), \quad x_2(t) = p(t), \quad u(t) = \dot{q}(t)$

- It can be re-written as

$$\begin{cases} \dot{x}_1(t) = u(t) \\ \dot{x}_2(t) = J(x_1(t))u(t) \end{cases}$$



- Is a Linear or Non-Linear Model?

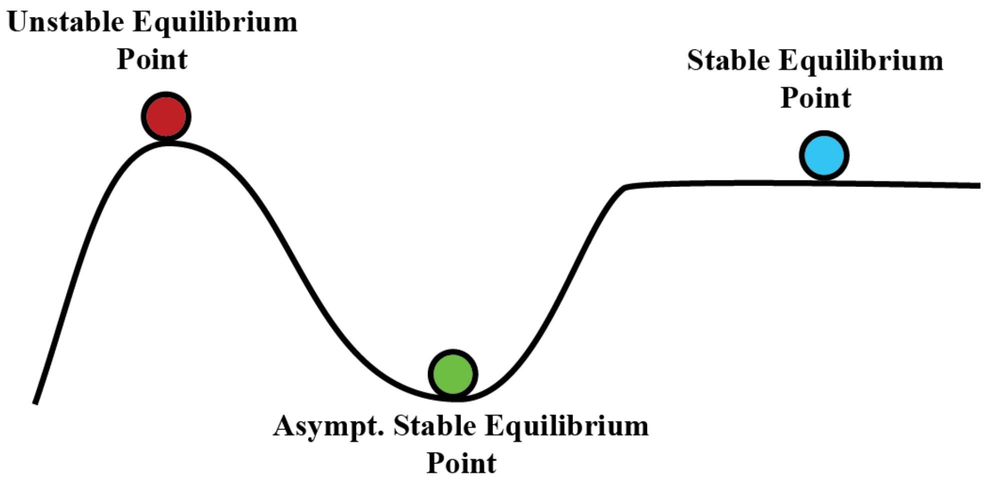
$$\dot{x}(t) = Ax(t) + Bu(t)$$

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

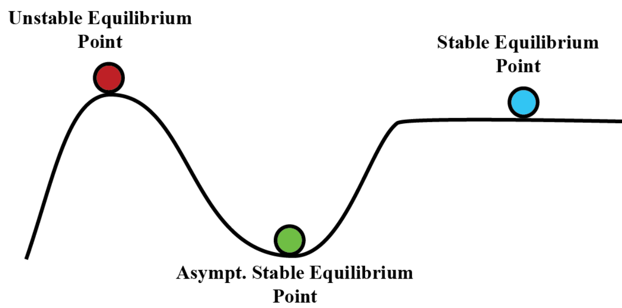
Stability of Dynamical Systems

Equilibrium Points and Stability

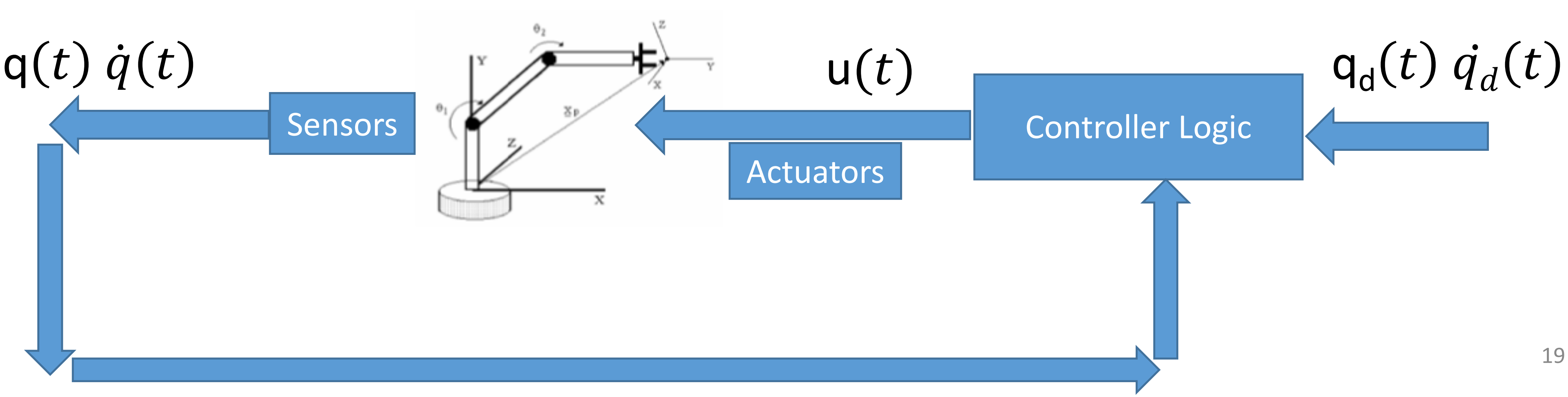
- By looking at the physics of the system (the ball) is easy to understand:
  - The equilibrium points (where the ball does not move)
  - How good the equilibrium is in a neighborhood of the equilibrium (locally)
    - Stable equilibrium: if I move the ball (by a little) w.r.t. the equilibrium, the ball is not going far away from the equilibrium
      - Stable (denoted with Marginally Stable or Simple Stable): if the ball remains in a neighborhood
      - Asymptotically Stable: if the ball comes back exactly to the equilibrium
    - Unstable equilibrium: small movements make the ball never come back to the equilibrium



Robot Manipulator: Equilibrium Point and Stability



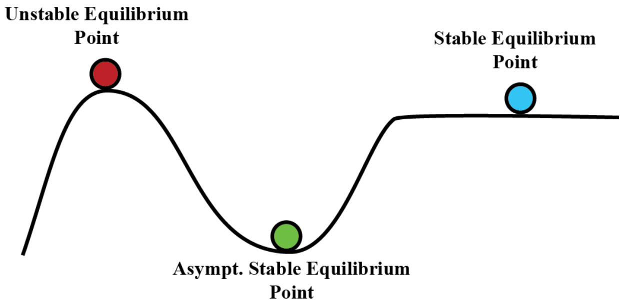
- Why do we care of equilibrium and stability for a Robotic Manipulator?
- Think about what we want to achieve in this system.



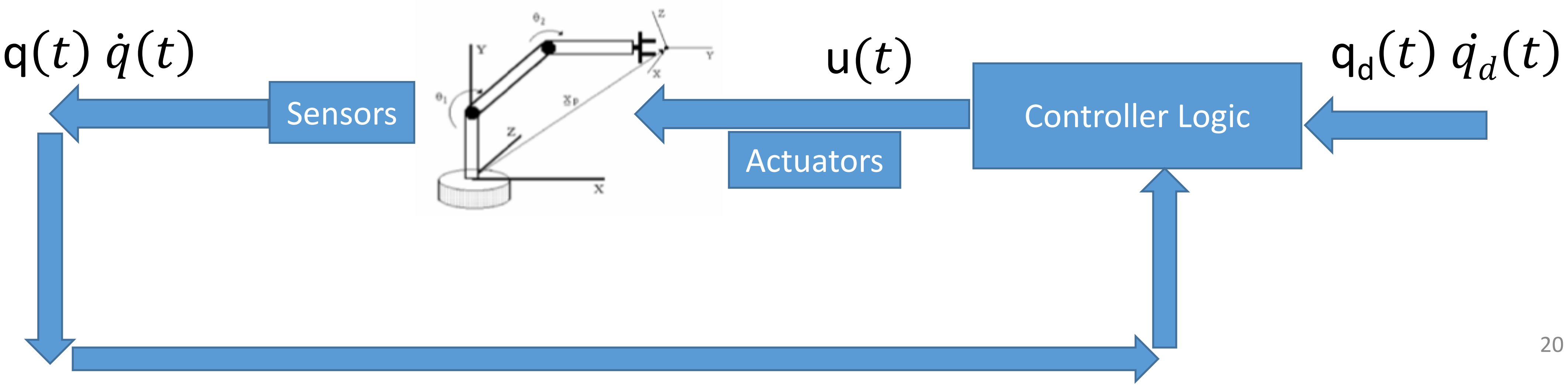
Robot Manipulator: Equilibrium Point and Stability

- If I define the error system

$$e(t) = q_d(t) - q(t)$$

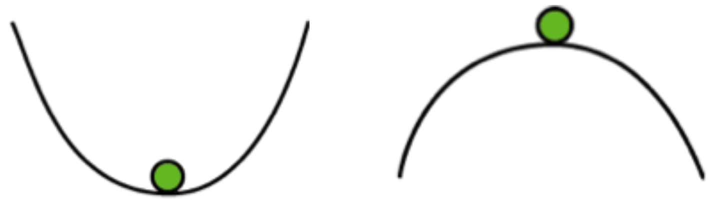


  - We want e(t)=0 to be an equilibrium point, right?
  - For any perturbation, we want the error to come back to 0, right?.
- We want e=0 to be an asymptotically stable equilibrium point.



Linear Systems: Equilibrium Points and Stability

$$\dot{x}(t) = Ax(t) + Bu(t)$$



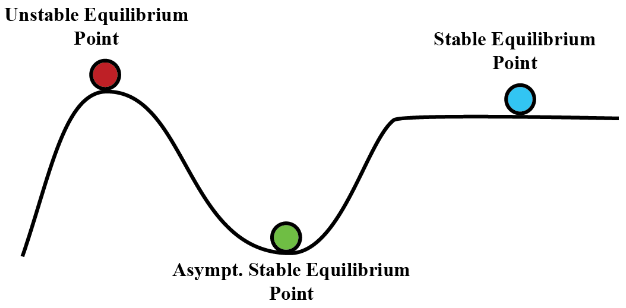
- Equilibrium points:
  - $(\bar{x}, \bar{u})$ such that $A\bar{x} + B\bar{u} = 0$
  - (0,0) is an evident equilibrium
  - Stability: All the equilibria share the same type of stability (system property)

Theorem 9.1 *Let* $\lambda_1, \ldots, \lambda_m$, $m \leq n$ *be the eingenvalues of* $A \in \mathbb{R}^{n \times n}$. *The system* $\dot{x} = Ax + Bu$ *is:*

- *Asymptotically stable if and only if (iff)* $\Re[\lambda_i] < 0$, $\forall i = 1, \ldots, m$
- *Stable or Marginally Stable if* $\Re[\lambda_i] \leq 0$, $\forall i$ *and the eigenvalues with null real part,* $\Re[\lambda_i] = 0$, *have equal algebraic and geometric multiplicity*
- *Unstable if* $\exists\, i$ *such that* $\Re[\lambda_i] > 0$ *or* $\exists\, \Re[\lambda_i] = 0$ *with algebraic multiplicity different from the geometric multiplicity.*

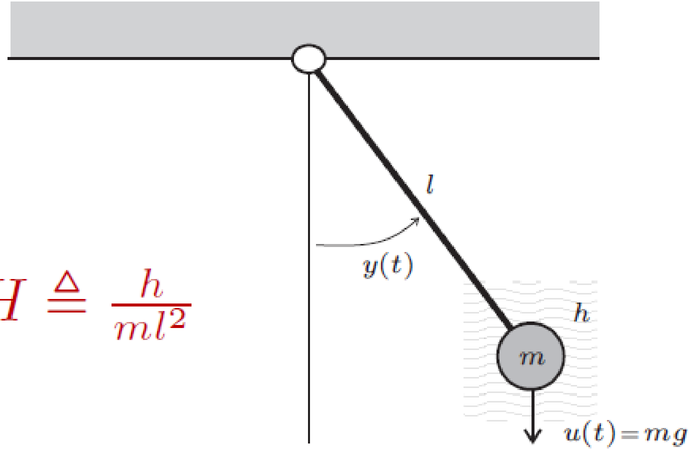
Non Linear Systems: Equilibrium Points and Stability

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



- Equilibrium points:
  - $(\bar{x}, \bar{u})$ such that $f(\bar{x}, \bar{u}) = 0$
  - No evident equilibrium: they can be 0,1,..many…infinitely many
- Stability: It is hard to prove. For different equilibria I can have different stability. Stability is not a system property but an equilibrium property (local)
- How to prove stability of equilibrium points:
  - Lyapunov indirect method
  - Lyapunov direct method

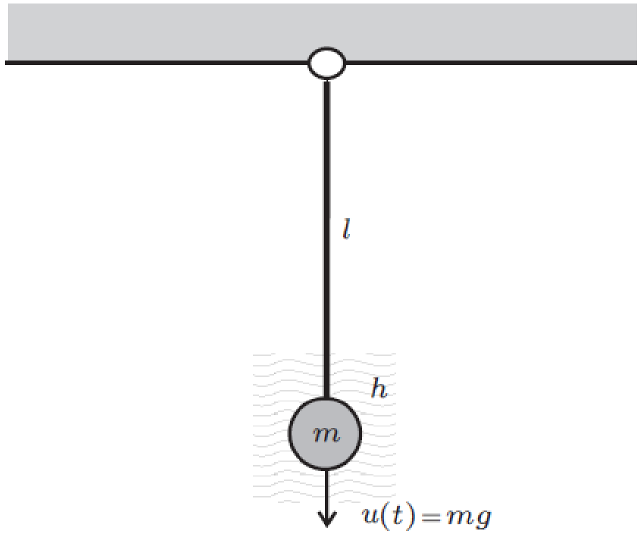
Example: Inverted Pendulum



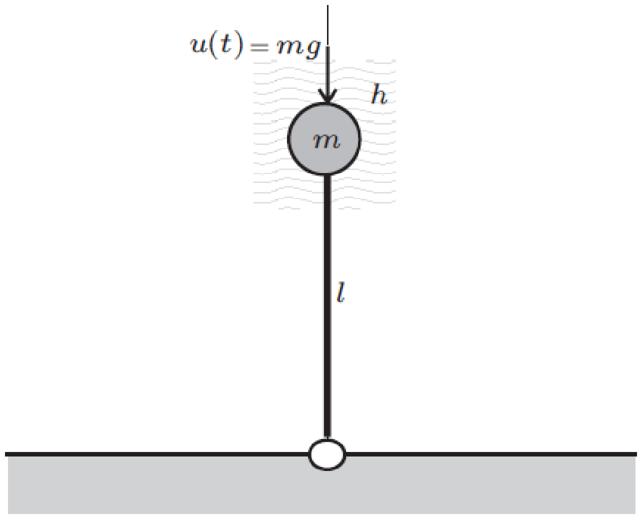
$$\begin{cases} \dot{x}_1 & = & x_2 \\ \dot{x}_2 & = & -\frac{g}{l}\sin x_1 - Hx_2, \qquad H \triangleq \frac{h}{ml^2} \end{cases}$$

- Equilibrium Points:

$$\begin{bmatrix} x_{2r} \\ -\frac{g}{l}\sin x_{1r} - Hx_{2r} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \Rightarrow \begin{cases} x_{2r} = 0 \\ x_{1r} = \pm k\pi, \ k = 0, 1, \ldots \end{cases}$$



$x_{2r} = 0, x_{1r} = 0, \pm 2\pi, \ldots$

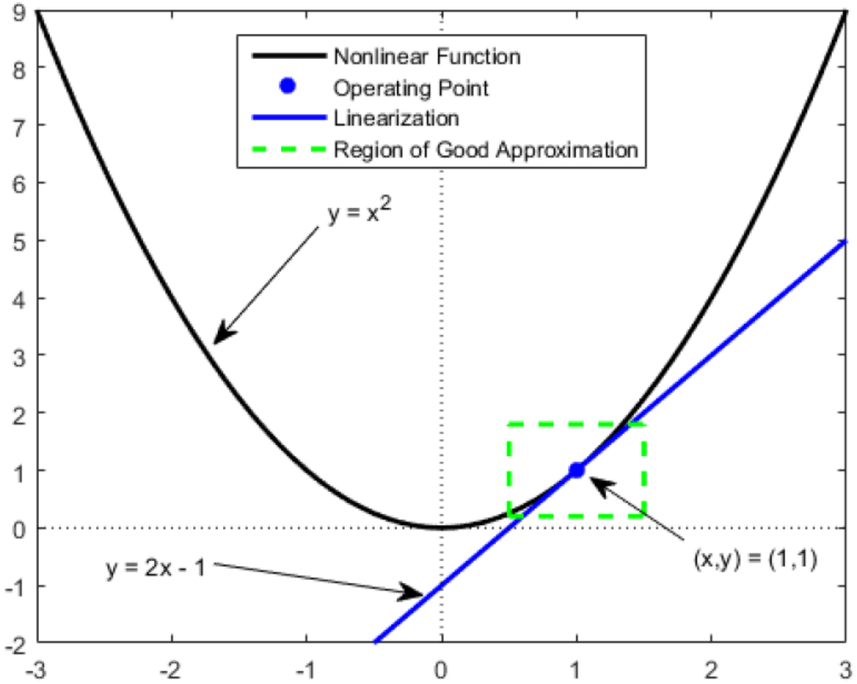


$x_{2r} = 0, x_{1r} = 0, \pm\pi, \pm 3\pi, \ldots$

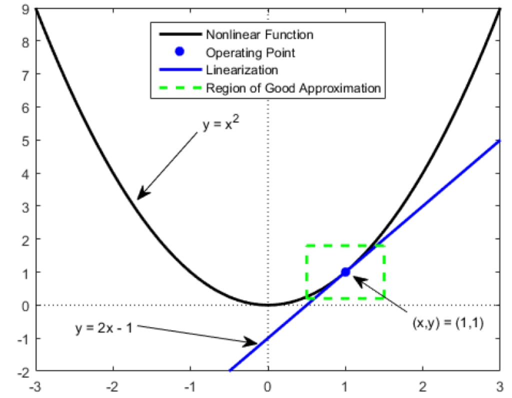
Lyapunov Indirect Method

Idea: Linearization of Non-Linear Systems:

- We know how to check stability for Linear Systems
- We can approximate the non-Linear system as a linear system around the equilibrium of interest
- E.g.



Linearization of Non-Linear Systems:



- Non-linear System $\dot{x}(t) \quad = \quad f(x(t), u(t))$
- Approximated Linear System:

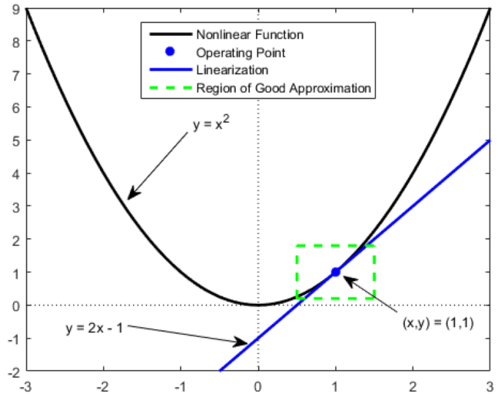
Let (x_r, u_r) be an equilibrium, $f(x_r, u_r) = 0$

$$\Delta u(t) \triangleq u(t) - u_r$$

$$\Delta x(t) \triangleq x(t) - x_r$$

$$\dot{x}(t) \approx \underbrace{\frac{\partial f}{\partial x}(x_r, u_r)}_{A} \Delta x(t) + \underbrace{\frac{\partial f}{\partial u}(x_r, u_r)}_{B} \Delta u(t)$$

$$\dot{\tilde{x}}(t) = A\tilde{x}(t) + B\tilde{u}(t)$$



Lyapunov Indirect Method

- Non-linear System $\dot{x}(t) = f(x(t))$
- Equilibrium Point $x = 0$
- Approximated Linear System around the equilibrium : $\dot{\tilde{x}}(t) = A\tilde{x}(t)$

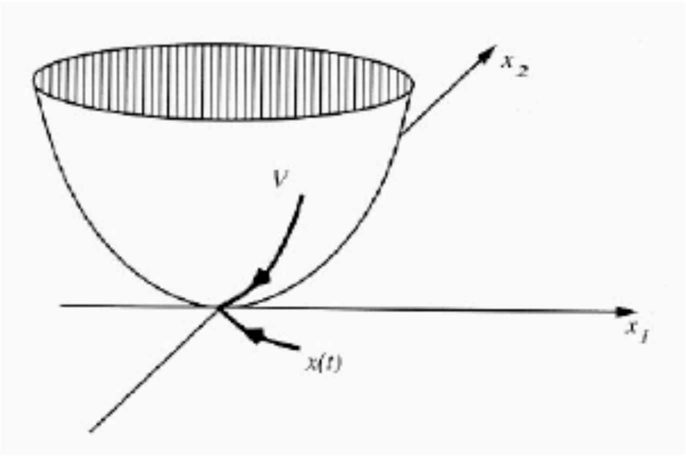
- *If $\dot{\tilde{x}} = A\tilde{x}$ is asymptotically stable in $\tilde{x} = 0$, then $x = 0$ is also an asymptotically stable equilibrium for the nonlinear system $\dot{x} = f(x)$*
- *If $\dot{\tilde{x}} = A\tilde{x}$ is unstable in $x = 0$, then $x = 0$ is also an unstable equilibrium for the nonlinear system $\dot{x} = f(x)$*
- *If $\dot{\tilde{x}} = A\tilde{x}$ is marginally stable in $x = 0$, nothing can be said about the stability of $x = 0$ for the nonlinear system $\dot{x} = f(x)$*

Lyapunov Direct Method

Lyapunov Direct Method: Idea

- If the energy of a system dissipates over time, the system asymptotically reaches a minimum-energy configuration

- If we can prove that a system reaches a minimum-energy configuration we have proved system stablility



Lyapunov Direct Method: Theorem

Theorem

Given the nonlinear system $\dot{x} = f(x)$, $f(0) = 0$, let $V : \mathbb{R}^n \mapsto \mathbb{R}$ be positive definite in a ball B_ϵ around the origin, $\epsilon > 0$, $V \in C^1(\mathbb{R})$. If the function

$$\dot{V}(x) = \nabla V(x)'\dot{x} = \nabla V(x)' f(x)$$

is negative definite on B_ϵ, then the origin is an asymptotically stable equilibrium point with **domain of attraction** B_ϵ ($\lim_{t\to+\infty} x(t) = 0$ for all $x(0) \in B_\epsilon$). If $\dot{V}(x)$ is only negative semi-definite on B_ϵ, then the the origin is a stable equilibrium point.

$$\dot{V}(x) = \frac{d}{dt}V(x) = \sum_{i=1}^{N} \frac{\partial V(x)}{\partial x_i}\frac{d}{dt}x_i = \left[\frac{\partial V(x)}{\partial x_1}, \ldots, \frac{\partial V(x)}{\partial x_n}\right]\dot{x}$$
$$= \Delta V(x)' f(x)$$

Example 1

- Let's consider the following non-linear system where the origin is clearly an equilibrium point

$$\begin{cases} \dot{x}_1 & = x_1(x_1^2 + x_2^2 - 2) - 4x_1x_2^2 \\ \dot{x}_2 & = 4x_1^2x_2 + x_2(x_1^2 + x_2^2 - 2) \end{cases}$$

- If we use the indirect Lyapunov Method

$$\frac{\partial f(0,0)}{\partial x} = \left[\begin{bmatrix} 3x_1^2 - 3x_2^2 - 2 & -6x_1x_2 \\ 10x_1x_2 & 5x_1^2 + 3x_2^2 - 2 \end{bmatrix}\right]\Bigg|_{x=0} = \begin{bmatrix} -2 & 0 \\ 0 & -2 \end{bmatrix}$$

- We know that the origin is asymptotically stable (locally).

Example 1

$$\begin{cases} \dot{x}_1 &= x_1(x_1^2 + x_2^2 - 2) - 4x_1x_2^2 \\ \dot{x}_2 &= 4x_1^2x_2 + x_2(x_1^2 + x_2^2 - 2) \end{cases}$$

- Let's consider the candidate Lyapunov function $V(x_1, x_2) = x_1^2 + x_2^2$
- If we compute its derivative along the system trajectory

$$\dot{V}(x_1, x_2) = 2(x_1^2 + x_2^2)(x_1^2 + x_2^2 - 2)$$

- It is easy to check that $\dot{V}(x_1, x_2)$ is negative definite if $\|x\|_2^2 = x_1^2 + x_2^2 < 2$
- Since for any B_ϵ with $0 < \epsilon < \sqrt{2}$ the hypotheses of Lyapunov's theorem are satisfied, $x = 0$ is an asymptotically stable equilibrium
- Any B_ϵ with $0 < \epsilon < \sqrt{2}$ is a domain of attraction

Example 2

- Let's consider the simple system: $\dot{x} = -x^3$
- Indirect Method is useless in this case

$$\frac{\partial f(0,0)}{\partial x} = -3 \cdot 0^2 = 0$$

- If we use the Lyapunov function:

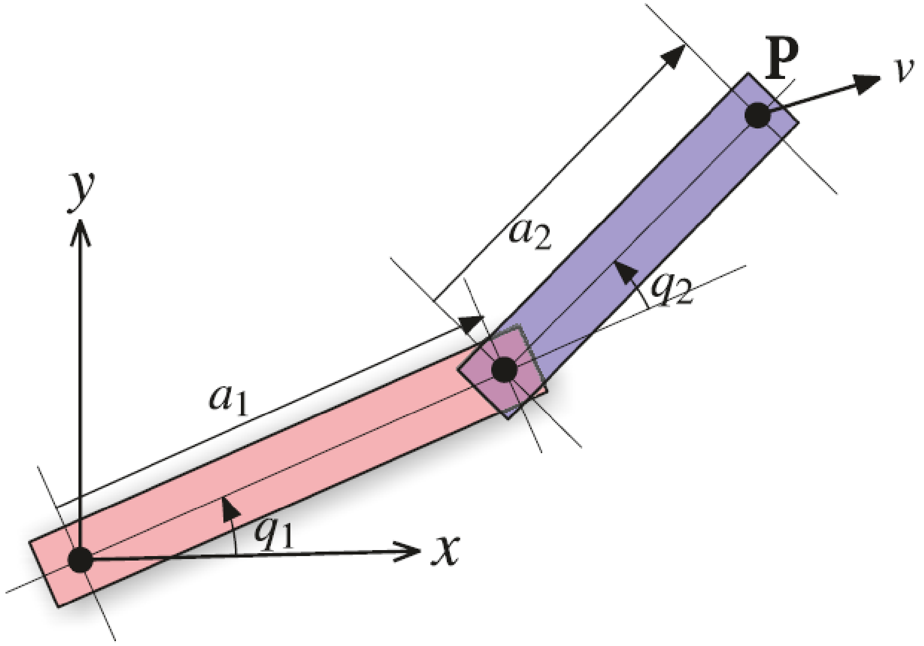
$$V = x^2 \qquad\qquad \dot{V} = -2x^4$$

- We can conclude that x=0 is a global asymptotically stable equilibrium point

Kinematics Control Algorithms

Differential Kinematics

$$\boldsymbol{v}_e = \begin{bmatrix} \dot{\boldsymbol{p}}_e \\ \boldsymbol{\omega}_e \end{bmatrix} = \boldsymbol{J}(\boldsymbol{q})\dot{\boldsymbol{q}}$$



Inverse Kinematics Algorithms

$$\dot{\boldsymbol{q}} = \boldsymbol{J}^{-1}(\boldsymbol{q})\boldsymbol{v}_e \qquad\qquad \boldsymbol{q}(t) = \int_0^t \dot{\boldsymbol{q}}(\varsigma)d\varsigma + \boldsymbol{q}(0)$$

- We have seen, that we can obtain inverse kinematics as follows

$$\dot{\boldsymbol{q}} = \boldsymbol{J}^{-1}(\boldsymbol{q})\boldsymbol{v}_e \qquad\qquad \boldsymbol{q}(t_{k+1}) = \boldsymbol{q}(t_k) + \boldsymbol{J}^{-1}(\boldsymbol{q}(t_k))\boldsymbol{v}_e(t_k)\Delta t$$

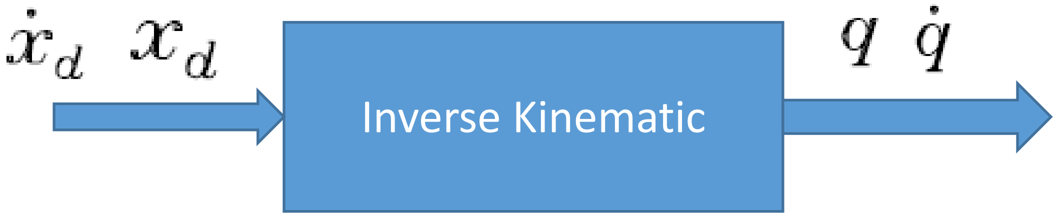
- However, this represents a discrete approximation of the integral that brings with it some drift errors.
- This means that the outcome (end-effector pose) might differ from the desired one

Improve Inverse Kinematics Algorithms?

- With this formula we don't have any way to correct the error (if any)

$$\boldsymbol{q}(t_{k+1}) = \boldsymbol{q}(t_k) + \boldsymbol{J}^{-1}(\boldsymbol{q}(t_k))\boldsymbol{v}_e(t_k)\Delta t$$

- Let's see why we can't correct errors from a Control System perspective



Open-Loop Kinematics Algorithm (if the Jacobian is invertible)

- Let's consider the operational space error

$$\boldsymbol{e} = \boldsymbol{x}_d - \boldsymbol{x}_e$$

- Its derivative is $\dot{\boldsymbol{e}} = \dot{\boldsymbol{x}}_d - \dot{\boldsymbol{x}}_e$. By using the analytical Jacobian we have

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

- If we consider the simple inversion $\dot{\boldsymbol{q}} = \boldsymbol{J}_A^{-1}(\boldsymbol{q})(\dot{\boldsymbol{x}}_d)$ and we plug it in the previous equation, we have

$$\dot{\boldsymbol{e}} = \boldsymbol{0}$$

- So, we don't have control on *e(t).*

  The system is stable but not Asympt. Stable

Open-Loop Kinematics Algorithm (if the Jacobian is invertible)

- Let's consider the operational space error

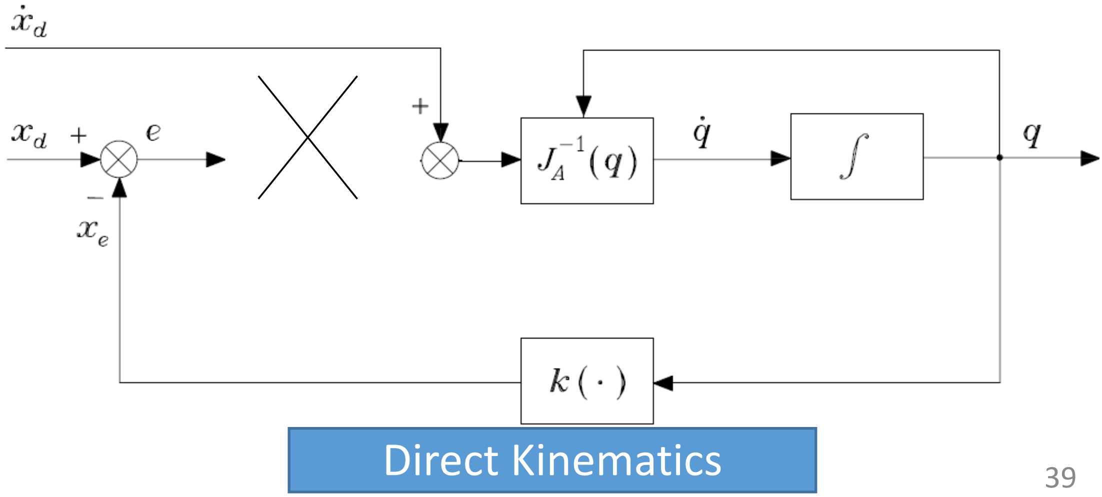
$$\boldsymbol{e} = \boldsymbol{x}_d - \boldsymbol{x}_e$$

- Its derivative is $\dot{\boldsymbol{e}} = \dot{\boldsymbol{x}}_d - \dot{\boldsymbol{x}}_e$. By using the analytical Jacobian we have

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

- If we consider the simple inversion $\dot{\boldsymbol{q}} = \boldsymbol{J}_A^{-1}(\boldsymbol{q})(\dot{\boldsymbol{x}}_d)$ and we plug it in the previous equation, we have

$$\dot{\boldsymbol{e}} = \boldsymbol{0}$$

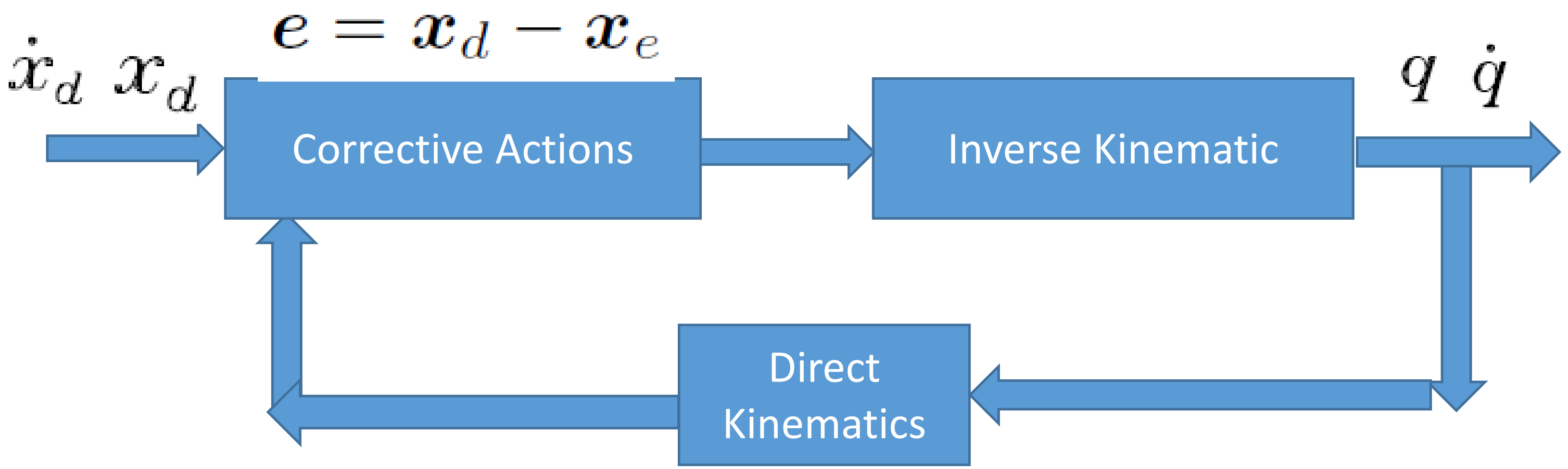


- So, we don't have control on *e(t).*

  The system is stable but not Asympt. Stable

Close Loop Inverse Kinematics Algorithms

- Can we use feedback control to improve the inverse kinematics computation?
- Yes, Feedback Scheme (like feedback control)!!



Closed Loop Kinematics Algorithm (if the Jacobian is invertible)

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

- A smarter option is $\dot{\boldsymbol{q}} = \boldsymbol{J}_A^{-1}(\boldsymbol{q})(\dot{\boldsymbol{x}}_d + \boldsymbol{K}\boldsymbol{e})$, obtaining

$$\dot{\boldsymbol{e}} + \boldsymbol{K}\boldsymbol{e} = \mathbf{0}$$

- Now we have control on *e(t)* through K

- If K (usually diagonal for decoupling reasoning) is positive defined, then the error system is asymptotically stable and goes to zero.

Closed Loop Kinematics Algorithm (if the Jacobian is invertible)

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

- A smarter option is $\dot{\boldsymbol{q}} = \boldsymbol{J}_A^{-1}(\boldsymbol{q})(\dot{\boldsymbol{x}}_d + \boldsymbol{K}\boldsymbol{e})$, obtaining

$$\dot{\boldsymbol{e}} + \boldsymbol{K}\boldsymbol{e} = \boldsymbol{0}$$

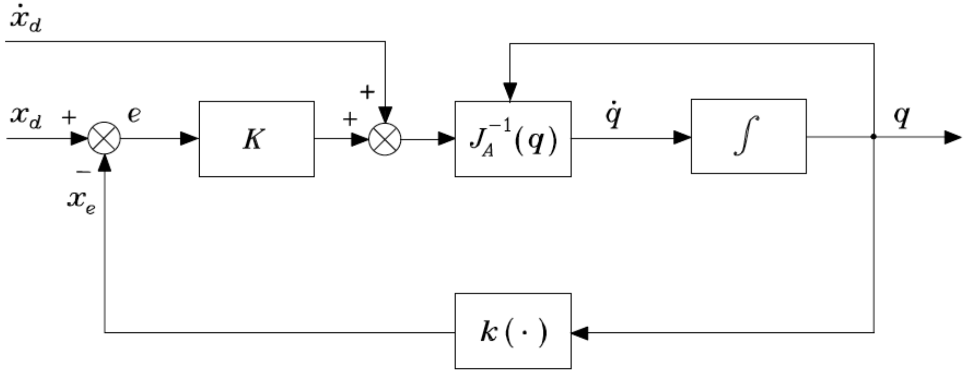


Fig. 3.11. Inverse kinematics algorithm with Jacobian inverse

- Now we have control on *e(t)* through K

- If K (usually diagonal for decoupling reasoning) is positive defined, then the error system is asymptotically stable and goes to zero.

Closed Loop Kinematics Algorithm (if the Jacobian is invertible)

- Properties:
  - As K increase, the error goes to zero faster.
  - The scheme ensure that the error is asymptotically converging to zero regardless of the imposed trajectory
  - If $\dot{x}_d$ is zero, then we have a scheme to numerically compute the inverse kinematics. (It corresponds to the Newton Method for solving a system of nonlinear equations.)

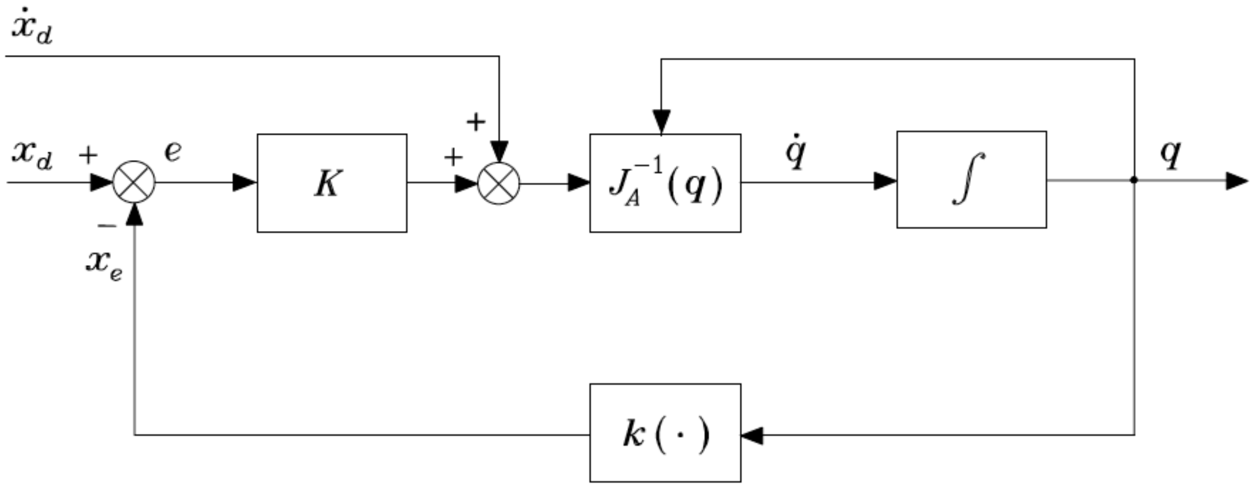


Fig. 3.11. Inverse kinematics algorithm with Jacobian inverse

Closed Loop Kinematics Algorithm (if the Jacobian is invertible)

- Problems:
  - We need to invert the Jacobian.
  - The Jacobian might not be invertible.

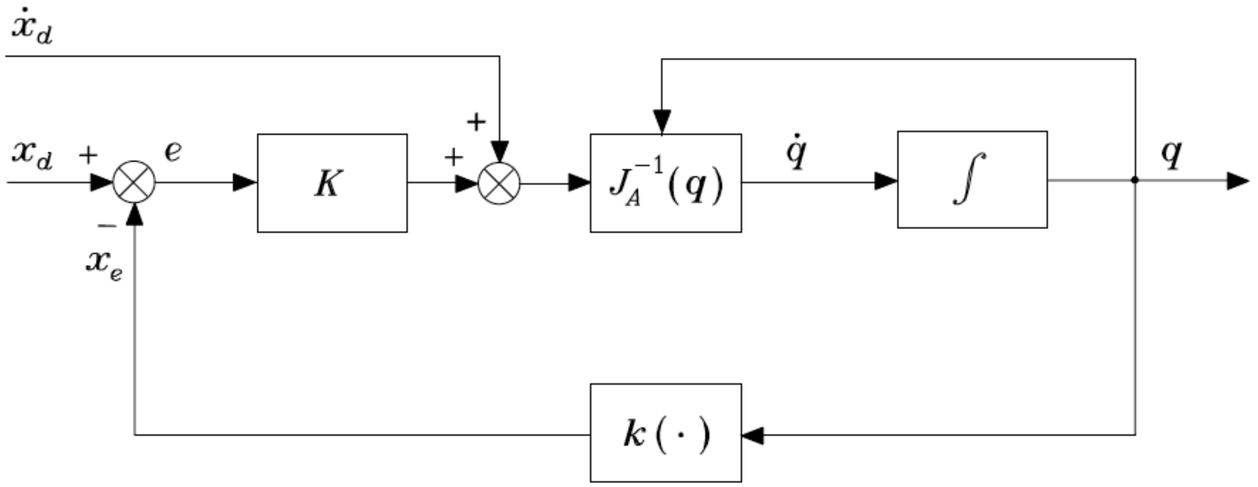


Fig. 3.11. Inverse kinematics algorithm with Jacobian inverse

Improved Inverse Kinematics Algorithm (without inversion of the Jacobian)

- Can we find a control scheme that does not need to compute the inverse of the Jacobian?
- Before, we used the Jacobian inverse to cancel the nonlinear component (the direct kinematics) of the control scheme so obtaining a linear system.
- Can we keep the non-linear part (not canceling it) and use something else instead of the Jacobian inverse to make the error system asymptotically stable?

Improved Inverse Kinematics Algorithm

- It is possible to prove that the the following control scheme ensures convergence to zero of the error signal (if x_d does not change in time); If x_d change in time, the error is limited (not zero) and become smaller as K increase.
- The notable feature of this scheme is that only the direct kinematics function is needed

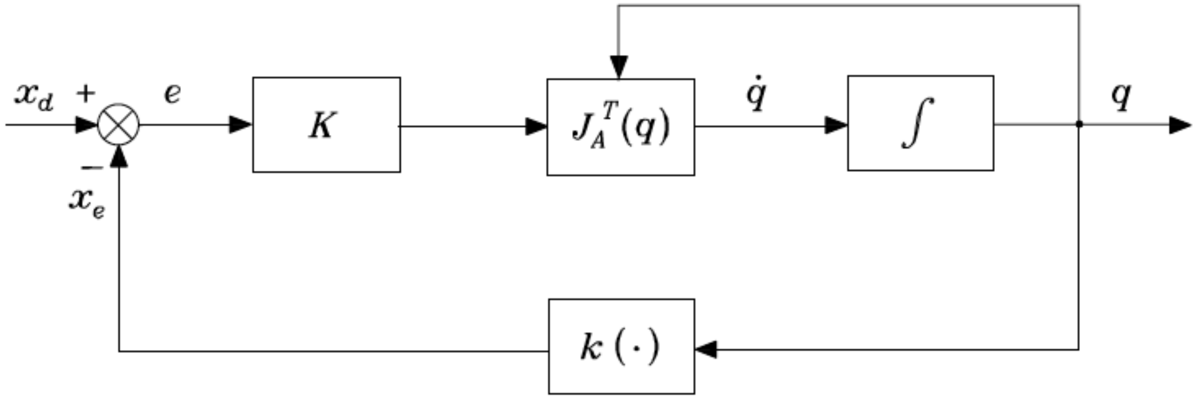


Fig. 3.12. Block scheme of the inverse kinematics algorithm with Jacobian transpose

Improved Inverse Kinematics Algorithm: Proof

- We have a non-linear system. To prove stability we can use the Direct Lyapunov method.
- Let consider the candidate Lyapunov function

$$V(e) = \frac{1}{2} e^T \mathbf{K} \mathbf{e}$$

- And compute its derivative w.r.t. time $\dot{V} = e^T \mathbf{K} \dot{x}_d - e^T \mathbf{K} \dot{x}_e$
- Since $\dot{x}_e = J_A(q)\dot{q}$
- We have

$$\dot{V} = e^T \mathbf{K} \dot{x}_d - e^T \mathbf{K} J_A(q)\dot{q}$$

Improved Inverse Kinematics Algorithm: Proof

$$\dot{V} = e^T \mathbf{K} \dot{x}_d - e^T \mathbf{K} J_A(q) \dot{q}$$

- To prove asymptotically stability we need to find a way to make the derivative definite negative.
- If we pick $\dot{q} = J_A^T(q)\mathbf{K}e$
- We obtain

$$\dot{V} = e^T \mathbf{K} \dot{x}_d - e^T \mathbf{K} J_A(q) J_A^T(q) \mathbf{K} e$$

- Here, if $\dot{x}_d(t) = 0$ $\quad \dot{V} = -e^T \mathbf{K} J_A(q) J_A^T(q) \mathbf{K} e$
  - The derivative is definite negative if J_A has full rank
  - The derivative is semi-definite negative if J_A has not full rank
- The not full rank case happens only for not not-reachable positons, so is not a problem

Improved Inverse Kinematics Algorithm

$$\dot{q} = J_A^T(q)\mathbf{K}e$$

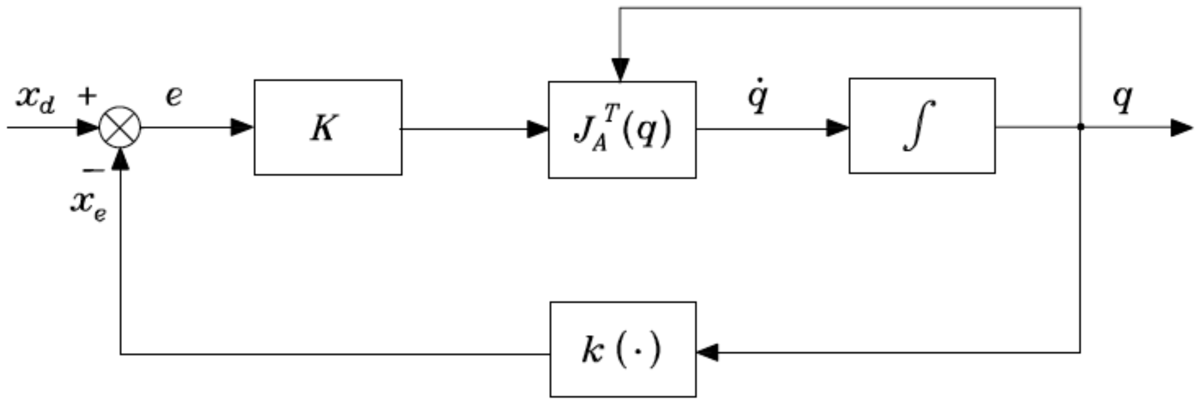


Fig. 3.12. Block scheme of the inverse kinematics algorithm with Jacobian transpose

- If $\dot{x}_d(t) \neq 0$ it is possible to prove that the error is norm-bounded and it become smaller as **K** increases

Kinematic Controller of a Manipulator

- If we look at the integrators as a simplified model of the robot, we can interpret this control scheme as a kinematic controller.
- This is true if there is a low-level joint controllers (local servos) capable to impose any desired joint velocity

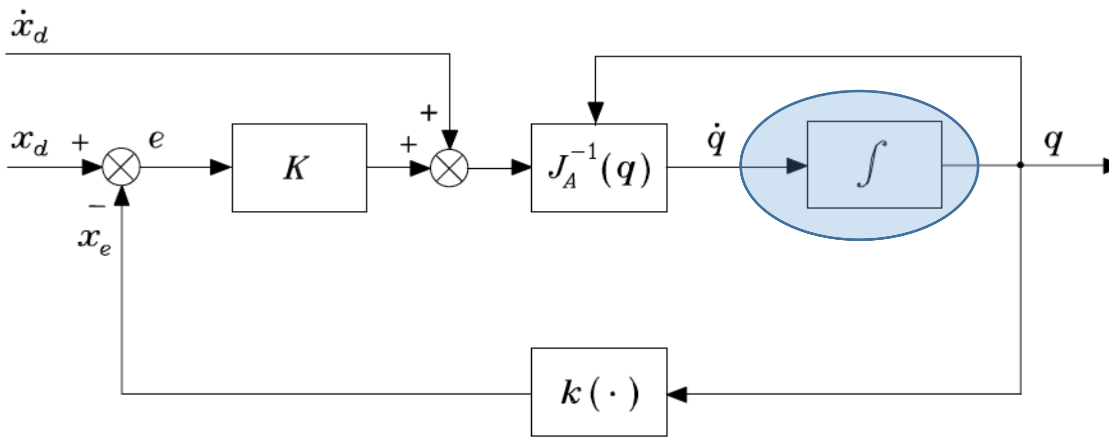


Fig. 3.11. Inverse kinematics algorithm with Jacobian inverse

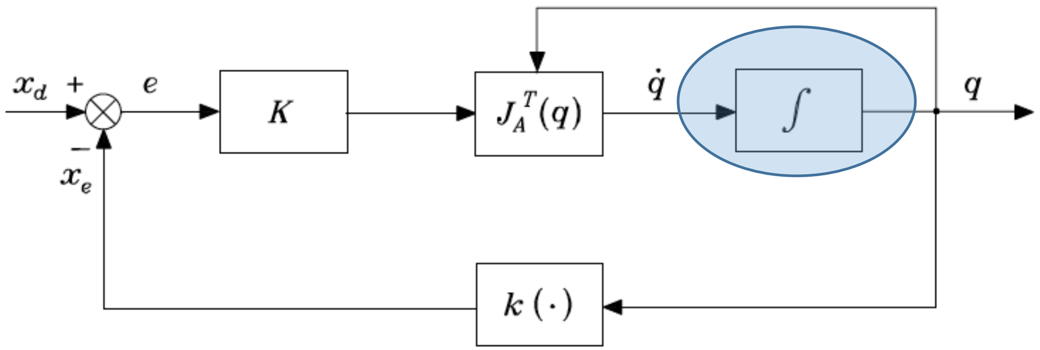


Fig. 3.12. Block scheme of the inverse kinematics algorithm with Jacobian transpose

THANK YOU!