Robotic Manipulator II: Control

ENGR 7401 – Differential Kinematics

(Lecture 3)

Prof. Walter Lucia

Lecture Outline

Differential Kinematics

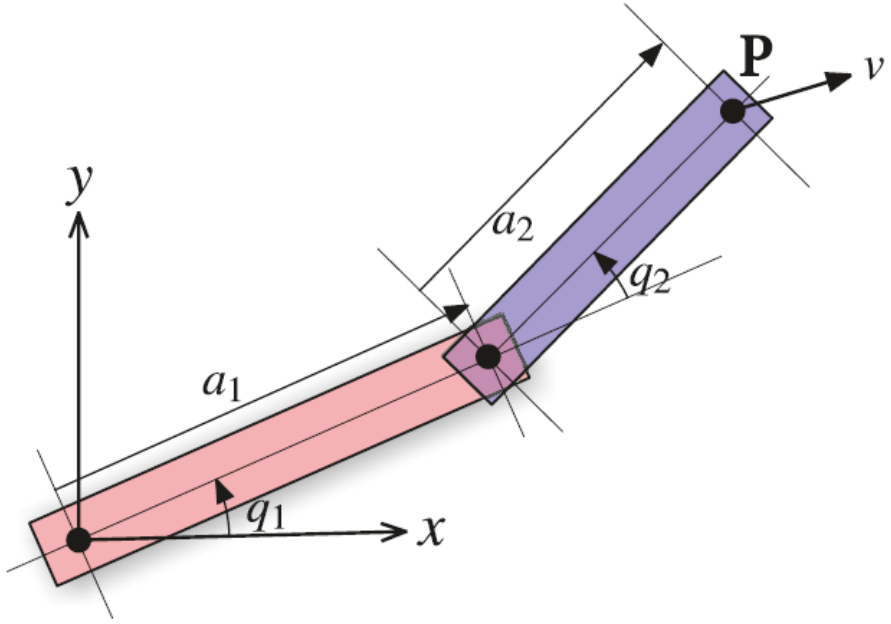
- Analytical Jacobian
- Geometrical Jacobian
- Geometrical vs Analytical Jacobian

Inverse Kinematics with Differential Kinematics

Inverse Kinematics Algorithms

Differential Kinematics Problem

Differential Kinematics



$$\boldsymbol{v}_e = \boldsymbol{J}(\boldsymbol{q})\dot{\boldsymbol{q}}$$

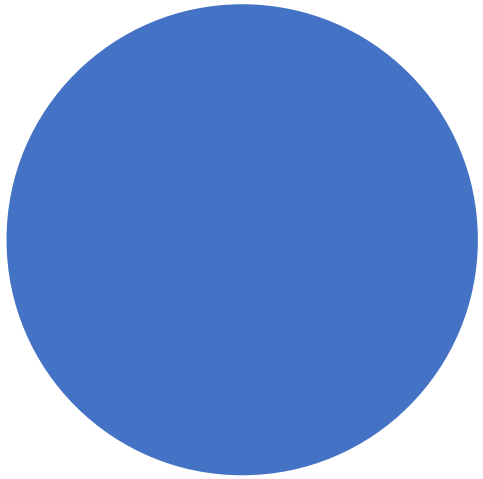
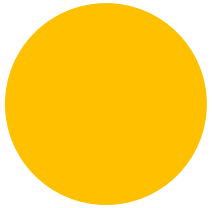
$$\boldsymbol{v}_e = [\,\dot{\boldsymbol{p}}_e^T \quad \boldsymbol{\omega}_e^T\,]^T$$

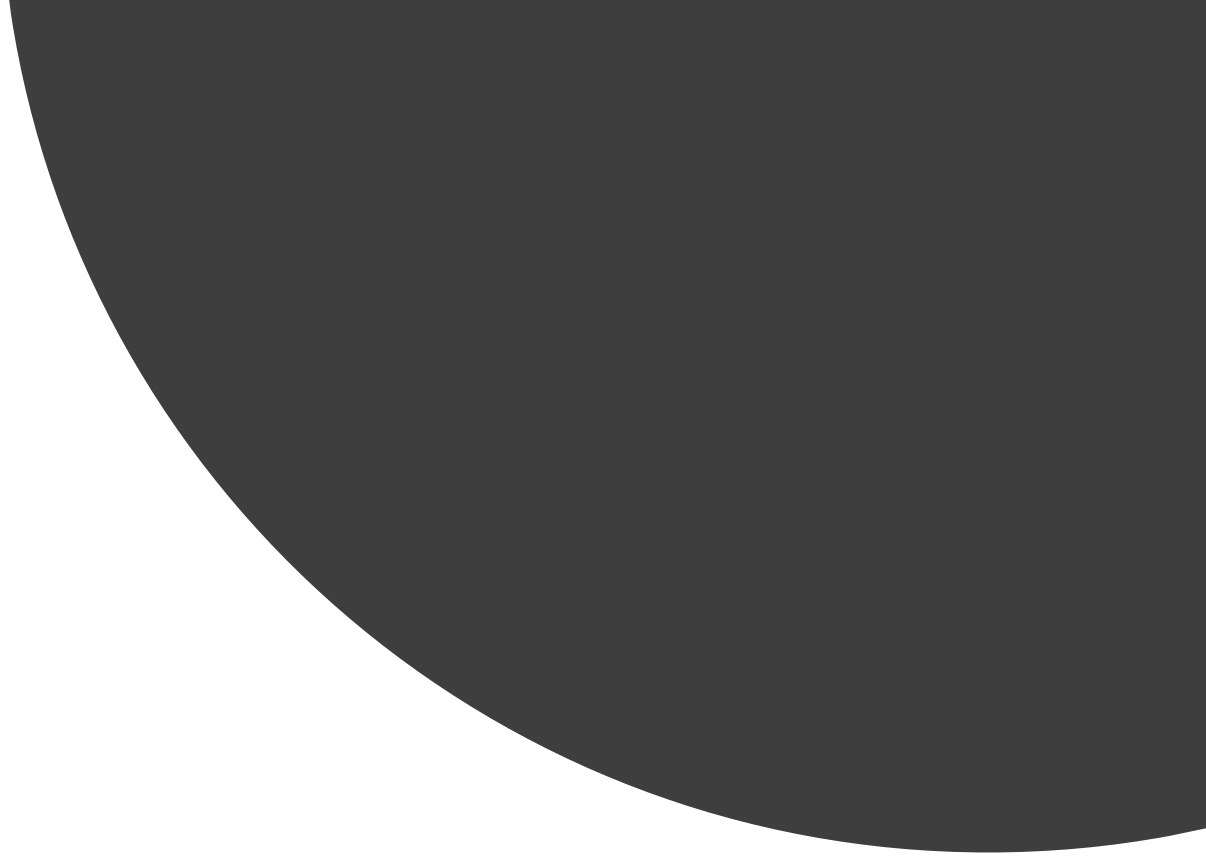
- Differential kinematics gives us the relationship between the joint velocities and the corresponding end-effector linear and angular velocity.

Brief Recap of Direct Kinematics

$$\boldsymbol{x} = \boldsymbol{k}(\boldsymbol{q}) = \begin{bmatrix} \boldsymbol{p}(\boldsymbol{q}) \\ \boldsymbol{\phi}(\boldsymbol{q}) \end{bmatrix}$$

- The vector function **k(q)** is usually a nonlinear function of the joint variables.
- The relation between the joint variables and the end-effector position is usually simple. The relation between the joint variables and the minimal orientation (of the end-effector) cannot be expressed in closed form (in general).
  - First, we need to compute the element of the rotation matrix $n_e(q),\ s_e(q),\ a_e(q)$
  - Then, we determine the Euler angles (the orientation) from the rotation matrix



Analytical Jacobian

Analytical Jacobian

$$\boldsymbol{x} = \boldsymbol{k}(\boldsymbol{q}) = \begin{bmatrix} \boldsymbol{p}(\boldsymbol{q}) \\ \phi(\boldsymbol{q}) \end{bmatrix}$$

- If ϕ is a minimal representation of the orientation (if we have it). Differentiating w.r.t. time we obtain

$$\dot{\boldsymbol{x}} = \begin{bmatrix} \dot{\boldsymbol{p}} \\ \dot{\phi} \end{bmatrix} = \begin{bmatrix} (\partial \boldsymbol{p}(\boldsymbol{q})/\partial \boldsymbol{q})\dot{\boldsymbol{q}} \\ (\partial \phi(\boldsymbol{q})/\partial \boldsymbol{q})\dot{\boldsymbol{q}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{J}_P(\boldsymbol{q}) \\ \boldsymbol{J}_\phi(\boldsymbol{q}) \end{bmatrix} \dot{\boldsymbol{q}} \qquad \dot{\boldsymbol{x}} = \frac{\partial \boldsymbol{k}(\boldsymbol{q})}{\partial \boldsymbol{q}} \dot{\boldsymbol{q}} = \boldsymbol{J}_A(\boldsymbol{q})\dot{\boldsymbol{q}}$$

- $\mathbf{J_A(q)}$ is called Analytical Jacobian

Geometrical Jacobian

Geometrical Jacobian

- If we don't have a minimal representation of the orientation, then we have to look for a so called Geometrical Jacobian

$$\dot{\boldsymbol{p}}_e = \boldsymbol{J}_P(\boldsymbol{q})\dot{\boldsymbol{q}}$$

$$\boldsymbol{\omega}_e = \boldsymbol{J}_O(\boldsymbol{q})\dot{\boldsymbol{q}}$$

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

- **J(q)** is called Geometrical Jacobian
- systematic methods exist to compute the geometrical Jacobian starting from the direct kinematics (homogenous transformation)

Homogeneous Transformation Recap

- A direct kinematic equation (homogenous transformation) describes the end-effect pose (position and orientation) as function of the joint variables.

$$\boldsymbol{T}_e(\boldsymbol{q}) = \begin{bmatrix} \boldsymbol{R}_e(\boldsymbol{q}) & \boldsymbol{p}_e(\boldsymbol{q}) \\ \mathbf{0}^T & 1 \end{bmatrix}$$

- Since we are interested in the end-effector linear and angular velocities it is worth to investigate the derivative of the rotation matrix R w.r.t. time

Derivation of a Rotation Matrix

- Let's consider a rotating matrix R=R(t)
- For orthogonality $\boldsymbol{R}(t)\boldsymbol{R}^T(t) = \boldsymbol{I}$
- Differentiating w.r.t. time $\dot{\boldsymbol{R}}(t)\boldsymbol{R}^T(t) + \boldsymbol{R}(t)\dot{\boldsymbol{R}}^T(t) = \boldsymbol{O}.$
- By defining $\boldsymbol{S}(t) = \dot{\boldsymbol{R}}(t)\boldsymbol{R}^T(t)$

$$\boldsymbol{S}(t) + \boldsymbol{S}^T(t) = \boldsymbol{O}$$

- By post multiplying both sides by R(t), we get

$$\dot{\boldsymbol{R}}(t) = \boldsymbol{S}(t)\boldsymbol{R}(t)$$

- The latter express the derivative of R(t) as a function of R(t) itself

Physical Interpretation of the derivative of a Rotation Matrix

$$\dot{\boldsymbol{R}}(t) = \boldsymbol{S}(t)\boldsymbol{R}(t)$$

- If we consider a vector p' and its rotation by means of R(t), $\boldsymbol{p}(t) = \boldsymbol{R}(t)\boldsymbol{p}'$
- The time derivative is $\dot{\boldsymbol{p}}(t) = \dot{\boldsymbol{R}}(t)\boldsymbol{p}'$

which can be rewritten as

$$\dot{\boldsymbol{p}}(t) = \boldsymbol{S}(t)\boldsymbol{R}(t)\boldsymbol{p}'$$

- From the physics we know that if we know the angular velocity with respect to the reference frame

$$\dot{\boldsymbol{p}}(t) = \boldsymbol{\omega}(t) \times \boldsymbol{R}(t)\boldsymbol{p}'.$$

- Therefore $S(t) = \omega \times$ (interpretation as vector product)

Angular Velocity From Physical Interpretation

$$\dot{\boldsymbol{p}}(t) = \boldsymbol{S}(t)\boldsymbol{R}(t)\boldsymbol{p}' \quad \Longleftrightarrow \quad \dot{\boldsymbol{p}}(t) = \boldsymbol{\omega}(t) \times \boldsymbol{R}(t)\boldsymbol{p}'.$$

- If $\boldsymbol{\omega}(t) = [\omega_x \quad \omega_y \quad \omega_z]^T$

- Then S contains the information on the angular velocities, $\boldsymbol{S}(t) = \boldsymbol{S}(\boldsymbol{\omega}(t))$

$$\boldsymbol{S} = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}$$

$$\boldsymbol{S}(t) = \dot{\boldsymbol{R}}(t)\boldsymbol{R}^T(t)$$

Example: Elementary rotation around z

$$\boldsymbol{R}_z(\alpha) = \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

- The matrix $\boldsymbol{S}(t) = \dot{\boldsymbol{R}}(t)\boldsymbol{R}^T(t)$ is

$$\boldsymbol{S}(t) = \begin{bmatrix} -\dot\alpha\sin\alpha & -\dot\alpha\cos\alpha & 0 \\ \dot\alpha\cos\alpha & -\dot\alpha\sin\alpha & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \cos\alpha & \sin\alpha & 0 \\ -\sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & -\dot\alpha & 0 \\ \dot\alpha & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} = \boldsymbol{S}(\boldsymbol{\omega}(t)).$$

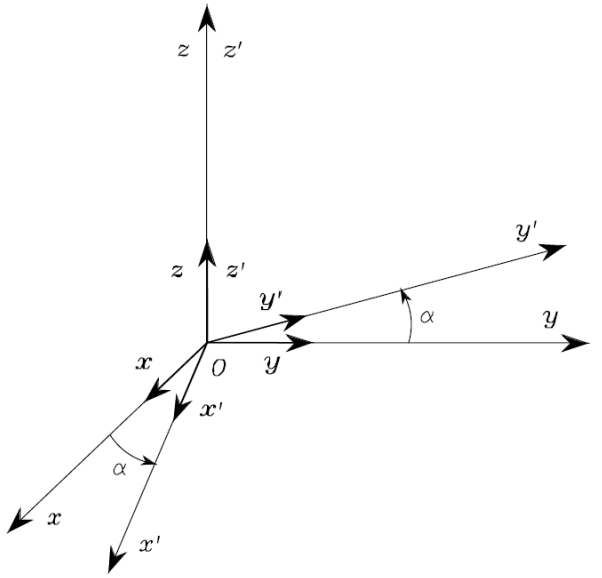


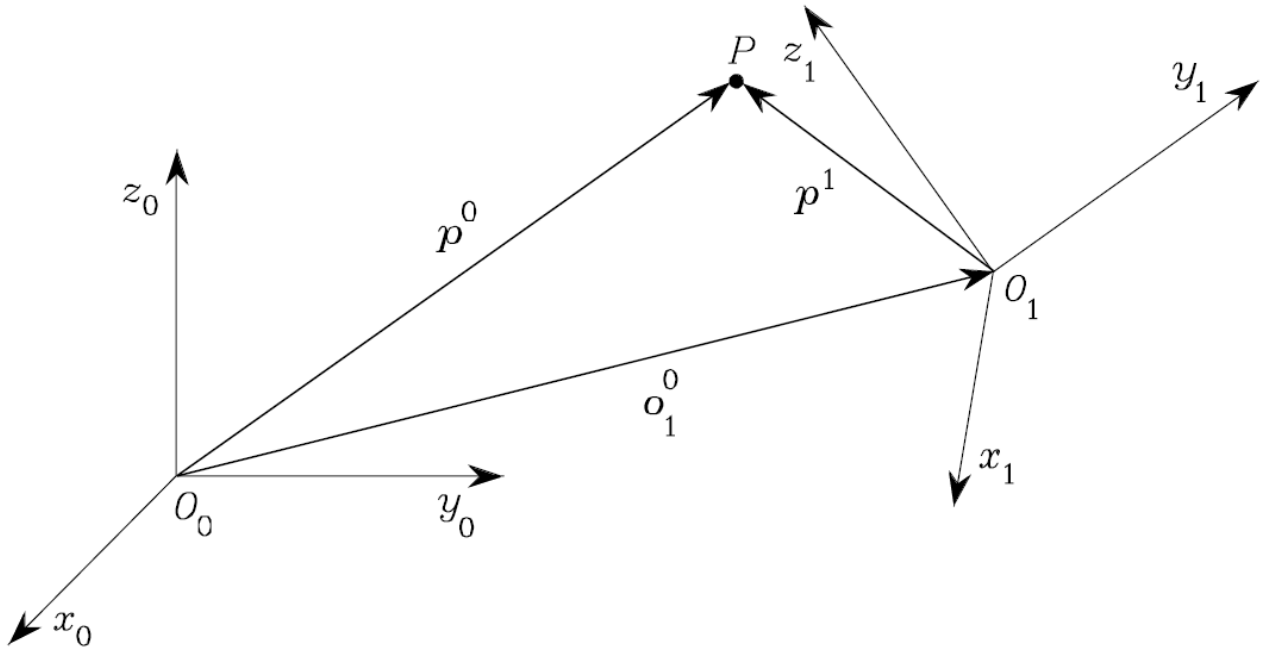
Fig. 2.2. Rotation of frame O–xyz by an angle α about axis z

$$\boldsymbol{S} = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix} \longrightarrow \boldsymbol{\omega} = \begin{bmatrix} 0 & 0 & \dot\alpha \end{bmatrix}^T$$

Geometrical Jacobian Computation

Linear Velocity

$$\boldsymbol{p}^0 = \boldsymbol{o}_1^0 + \boldsymbol{R}_1^0 \boldsymbol{p}^1.$$



- Differentiating w.r.t. time

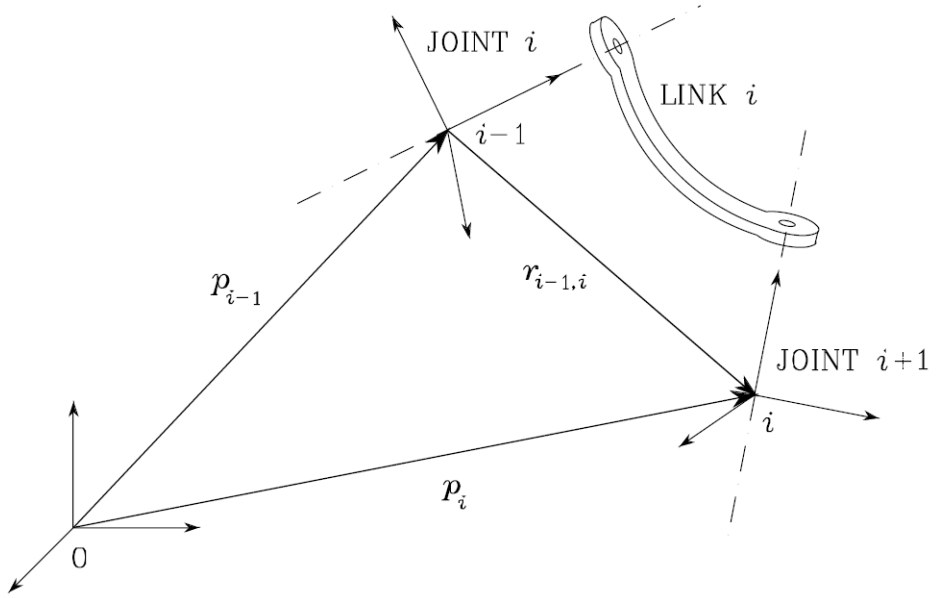
$$\dot{\boldsymbol{p}}^0 = \dot{\boldsymbol{o}}_1^0 + \boldsymbol{R}_1^0 \dot{\boldsymbol{p}}^1 + \dot{\boldsymbol{R}}_1^0 \boldsymbol{p}^1$$

$$\dot{\boldsymbol{p}}^0 = \dot{\boldsymbol{o}}_1^0 + \boldsymbol{R}_1^0 \dot{\boldsymbol{p}}^1 + \boldsymbol{S}(\boldsymbol{\omega}_1^0)\boldsymbol{R}_1^0 \boldsymbol{p}^1 \longleftarrow \boldsymbol{S}(t) = \dot{\boldsymbol{R}}(t)\boldsymbol{R}^T(t)$$

- If we denote $\boldsymbol{r}_1^0 = \boldsymbol{R}_1^0 \boldsymbol{p}^1$ and $\dot{\boldsymbol{p}}^1 = \boldsymbol{0}$ (if p^1 is fixed in the frame O_1) then

$$\dot{\boldsymbol{p}}^0 = \dot{\boldsymbol{o}}_1^0 + \boldsymbol{\omega}_1^0 \times \boldsymbol{r}_1^0$$

Link Velocities



- The linear velocity of Link *i* as a function of the translation and rotational velocities of Link *i-1*

$$\boldsymbol{p}_i = \boldsymbol{p}_{i-1} + \boldsymbol{R}_{i-1}\boldsymbol{r}_{i-1,i}^{i-1}$$

$$\dot{\boldsymbol{p}}_i = \dot{\boldsymbol{p}}_{i-1} + \boldsymbol{R}_{i-1}\dot{\boldsymbol{r}}_{i-1,i}^{i-1} + \boldsymbol{\omega}_{i-1} \times \boldsymbol{R}_{i-1}\boldsymbol{r}_{i-1,i}^{i-1} = \dot{\boldsymbol{p}}_{i-1} + \boldsymbol{v}_{i-1,i} + \boldsymbol{\omega}_{i-1} \times \boldsymbol{r}_{i-1,i}$$

- The rotational velocity of Link *i* as a function of the angular velocity of Link *i-1* and angular velocity of I w.r.t. Link *i-1*

$$\boldsymbol{\omega}_i = \boldsymbol{\omega}_{i-1} + \boldsymbol{R}_{i-1}\boldsymbol{\omega}_{i-1,i}^{i-1} = \boldsymbol{\omega}_{i-1} + \boldsymbol{\omega}_{i-1,i},$$

Link Velocities

Prismatic Joint

$$\boldsymbol{\omega}_i = \boldsymbol{\omega}_{i-1}$$

$$\dot{\boldsymbol{p}}_i = \dot{\boldsymbol{p}}_{i-1} + \dot{d}_i \boldsymbol{z}_{i-1} + \boldsymbol{\omega}_i \times \boldsymbol{r}_{i-1,i}$$

Revolution Joint

$$\boldsymbol{\omega}_i = \boldsymbol{\omega}_{i-1} + \dot{\vartheta}_i \boldsymbol{z}_{i-1}$$

$$\dot{\boldsymbol{p}}_i = \dot{\boldsymbol{p}}_{i-1} + \boldsymbol{\omega}_i \times \boldsymbol{r}_{i-1,i}$$

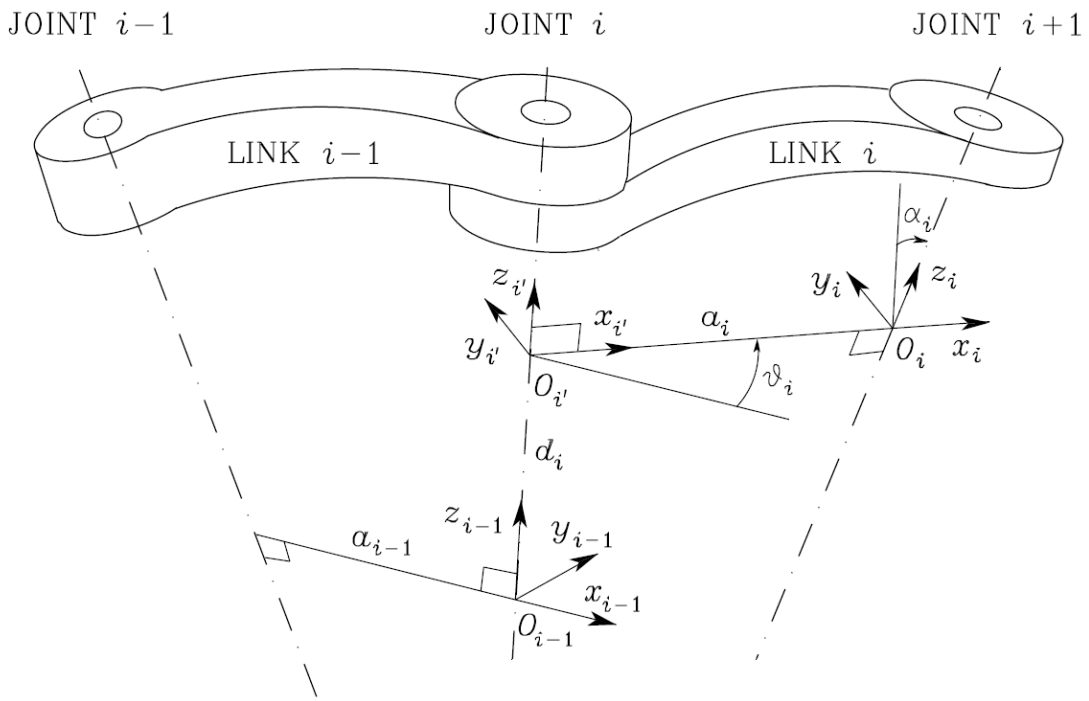
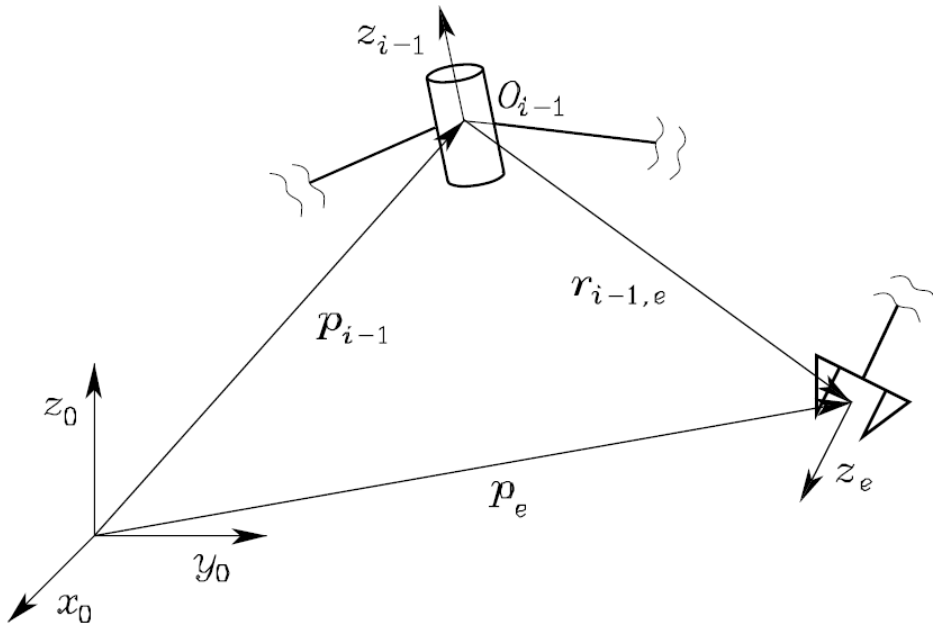


Fig. 2.16. Denavit–Hartenberg kinematic parameters

Geometrical Jacobian (Linear Velocities)

Linear Velocities



- Note that we can write down the derivative of the end-effector position as follows

$$\dot{\boldsymbol{p}}_e = \sum_{i=1}^{n} \frac{\partial \boldsymbol{p}_e}{\partial q_i} \dot{q}_i = \sum_{i=1}^{n} \boldsymbol{J}_{Pi} \dot{q}_i$$

- This tells us that we can express the linear velocity as the sum of the single joints contributions

$$\dot{q}_i \boldsymbol{J}_{Pi}$$

Prisma Joint $(q_i = d_i)$

- For linear velocity, we know that

$$\boldsymbol{v}_{i-1,i} = \dot{d}_i \boldsymbol{z}_{i-1}$$

therefore,

$$\dot{q}_i \boldsymbol{J}_{Pi} = \dot{d}_i \boldsymbol{z}_{i-1} \quad \boldsymbol{J}_{Pi} = \boldsymbol{z}_{i-1}$$

- No contribution to the angular velocity

Revolute Joint $(q_i = \vartheta_i)$

- For linear velocity, we know that

$$\boldsymbol{v}_{i-1,i} = \boldsymbol{\omega}_{i-1,i} \times \boldsymbol{r}_{i-1,i}$$

therefore,

$$\dot{q}_i \boldsymbol{J}_{Pi} = \boldsymbol{\omega}_{i-1,i} \times \boldsymbol{r}_{i-1,e} = \dot{\vartheta}_i \boldsymbol{z}_{i-1} \times (\boldsymbol{p}_e - \boldsymbol{p}_{i-1})$$

$$\boldsymbol{J}_{Pi} = \boldsymbol{z}_{i-1} \times (\boldsymbol{p}_e - \boldsymbol{p}_{i-1})$$

- For the angular velocity, we know that

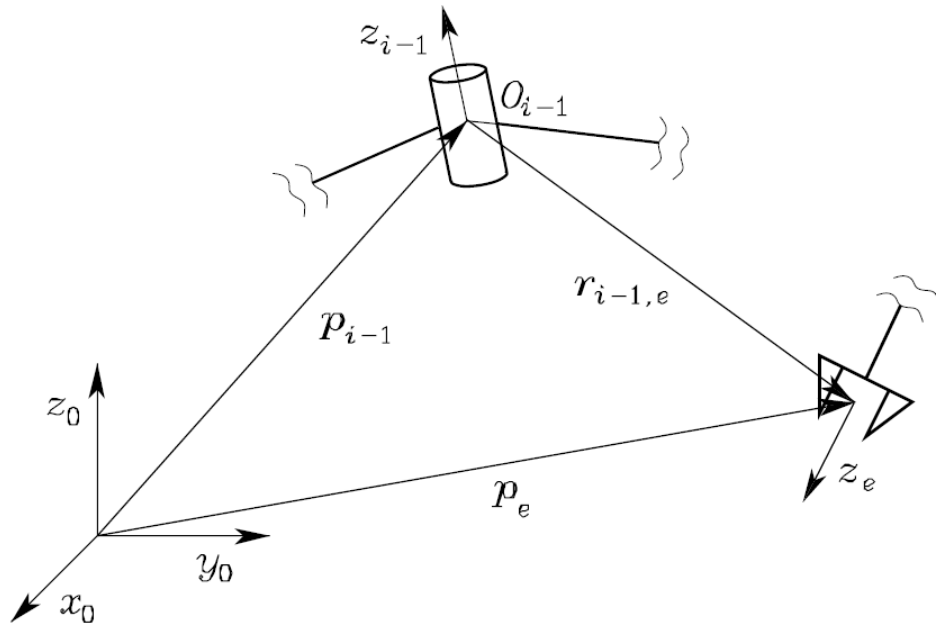
$$\boldsymbol{\omega}_{i-1,i} = \dot{\vartheta}_i \boldsymbol{z}_{i-1},$$

- Therefore

$$\boldsymbol{J}_{Oi} = \boldsymbol{z}_{i-1}$$

Geometrical Jacobian (Angular Velocities)

Angular Velocities



- In view of

$$\boldsymbol{\omega}_i = \boldsymbol{\omega}_{i-1} + \boldsymbol{R}_{i-1}\boldsymbol{\omega}_{i-1,i}^{i-1} = \boldsymbol{\omega}_{i-1} + \boldsymbol{\omega}_{i-1,i}$$

we know that

$$\boldsymbol{\omega}_e = \boldsymbol{\omega}_n = \sum_{i=1}^{n} \boldsymbol{\omega}_{i-1,i} = \sum_{i=1}^{n} \boldsymbol{J}_{Oi}\dot{q}_i,$$

Prismatic Joint $(q_i = d_i)$

- No contribution to the angular velocity

$$\boldsymbol{J}_{Oi} = \mathbf{0}$$

Revolute Joint $(q_i = \vartheta_i)$

- For the angular velocity, we know that

$$\boldsymbol{\omega}_{i-1,i} = \dot{\vartheta}_i \boldsymbol{z}_{i-1},$$

- Therefore

$$\dot{q}_i \boldsymbol{J}_{Oi} = \dot{\vartheta}_i \boldsymbol{z}_{i-1} \qquad \boldsymbol{J}_{Oi} = \boldsymbol{z}_{i-1}$$

Geometrical Jacobian (putting all together)

$$\boldsymbol{J} = \begin{bmatrix} \boldsymbol{\jmath}_{P1} & & \boldsymbol{\jmath}_{Pn} \\ & \cdots & \\ \boldsymbol{\jmath}_{O1} & & \boldsymbol{\jmath}_{On} \end{bmatrix}$$

$$\begin{bmatrix} \boldsymbol{\jmath}_{Pi} \\ \boldsymbol{\jmath}_{Oi} \end{bmatrix} = \begin{cases} \begin{bmatrix} \boldsymbol{z}_{i-1} \\ \boldsymbol{0} \end{bmatrix} & \text{for a } \textit{prismatic} \text{ joint} \\ \begin{bmatrix} \boldsymbol{z}_{i-1} \times (\boldsymbol{p}_e - \boldsymbol{p}_{i-1}) \\ \boldsymbol{z}_{i-1} \end{bmatrix} & \text{for a } \textit{revolute} \text{ joint.} \end{cases}$$

Geometrical Jacobian

$$\begin{bmatrix} \boldsymbol{J}_{Pi} \\ \boldsymbol{J}_{Oi} \end{bmatrix} = \begin{cases} \begin{bmatrix} \boldsymbol{z}_{i-1} \\ \boldsymbol{0} \end{bmatrix} & \text{for a } \textit{prismatic} \text{ joint} \\ \begin{bmatrix} \boldsymbol{z}_{i-1} \times (\boldsymbol{p}_e - \boldsymbol{p}_{i-1}) \\ \boldsymbol{z}_{i-1} \end{bmatrix} & \text{for a } \textit{revolute} \text{ joint.} \end{cases}$$

- To compute the geometrical Jacobian the following information are needed:

  - $\boldsymbol{z}_{i-1}$ is given by the third column of the rotation matrix $\boldsymbol{R}_{i-1}^{0}$, i.e.,

$$\boldsymbol{z}_{i-1} = \boldsymbol{R}_{1}^{0}(q_1) \dots \boldsymbol{R}_{i-1}^{i-2}(q_{i-1})\boldsymbol{z}_0$$

    where $\boldsymbol{z}_0 = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T$ allows the selection of the third column.

Geometrical Jacobian

$$\begin{bmatrix} \boldsymbol{J}_{Pi} \\ \boldsymbol{J}_{Oi} \end{bmatrix} = \begin{cases} \begin{bmatrix} \boldsymbol{z}_{i-1} \\ \boldsymbol{0} \end{bmatrix} & \text{for a } \textit{prismatic} \text{ joint} \\ \begin{bmatrix} \boldsymbol{z}_{i-1} \times (\boldsymbol{p}_e - \boldsymbol{p}_{i-1}) \\ \boldsymbol{z}_{i-1} \end{bmatrix} & \text{for a } \textit{revolute} \text{ joint.} \end{cases}$$

- $\boldsymbol{p}_e$ is given by the first three elements of the fourth column of the transformation matrix $\boldsymbol{T}_e^0$, i.e., by expressing $\widetilde{\boldsymbol{p}}_e$ in the (4×1) homogeneous form

$$\widetilde{\boldsymbol{p}}_e = \boldsymbol{A}_1^0(q_1) \dots \boldsymbol{A}_n^{n-1}(q_n)\widetilde{\boldsymbol{p}}_0 \qquad (3.32)$$

  where $\widetilde{\boldsymbol{p}}_0 = [\,0 \quad 0 \quad 0 \quad 1\,]^T$ allows the selection of the fourth column.

- $\boldsymbol{p}_{i-1}$ is given by the first three elements of the fourth column of the transformation matrix $\boldsymbol{T}_{i-1}^0$, i.e., it can be extracted from

$$\widetilde{\boldsymbol{p}}_{i-1} = \boldsymbol{A}_1^0(q_1) \dots \boldsymbol{A}_{i-1}^{i-2}(q_{i-1})\widetilde{\boldsymbol{p}}_0. \qquad (3.33)$$

Example 1

3-Links Planar Arm

$$\begin{bmatrix} \boldsymbol{J}_{Pi} \\ \boldsymbol{J}_{Oi} \end{bmatrix} = \begin{cases} \begin{bmatrix} \boldsymbol{z}_{i-1} \\ \boldsymbol{0} \end{bmatrix} & \text{for a } \textit{prismatic} \text{ joint} \\ \begin{bmatrix} \boldsymbol{z}_{i-1} \times (\boldsymbol{p}_e - \boldsymbol{p}_{i-1}) \\ \boldsymbol{z}_{i-1} \end{bmatrix} & \text{for a } \textit{revolute} \text{ joint.} \end{cases}$$

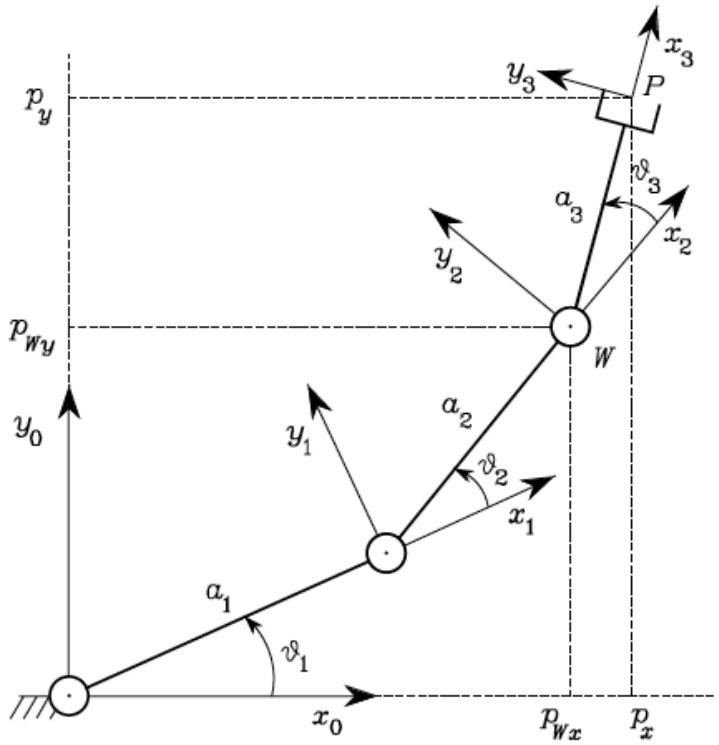


Fig. 2.20. Three-link planar arm

$$\boldsymbol{J}(\boldsymbol{q}) = \begin{bmatrix} \boldsymbol{z}_0 \times (\boldsymbol{p}_3 - \boldsymbol{p}_0) & \boldsymbol{z}_1 \times (\boldsymbol{p}_3 - \boldsymbol{p}_1) & \boldsymbol{z}_2 \times (\boldsymbol{p}_3 - \boldsymbol{p}_2) \\ \boldsymbol{z}_0 & \boldsymbol{z}_1 & \boldsymbol{z}_2 \end{bmatrix}$$

$$\boldsymbol{p}_0 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \quad \boldsymbol{p}_1 = \begin{bmatrix} a_1 c_1 \\ a_1 s_1 \\ 0 \end{bmatrix} \quad \boldsymbol{p}_2 = \begin{bmatrix} a_1 c_1 + a_2 c_{12} \\ a_1 s_1 + a_2 s_{12} \\ 0 \end{bmatrix} \quad \boldsymbol{p}_3 = \begin{bmatrix} a_1 c_1 + a_2 c_{12} + a_3 c_{123} \\ a_1 s_1 + a_2 s_{12} + a_3 s_{123} \\ 0 \end{bmatrix}$$

$$\boldsymbol{z}_0 = \boldsymbol{z}_1 = \boldsymbol{z}_2 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

Recap: Cross Product Operator

$$\mathbf{u} \times \mathbf{v} = \begin{vmatrix} \mathbf{i} & \mathbf{j} & \mathbf{k} \\ u_1 & u_2 & u_3 \\ v_1 & v_2 & v_3 \end{vmatrix}$$

$$\mathbf{u} \times \mathbf{v} = \begin{vmatrix} u_2 & u_3 \\ v_2 & v_3 \end{vmatrix} \mathbf{i} - \begin{vmatrix} u_1 & u_3 \\ v_1 & v_3 \end{vmatrix} \mathbf{j} + \begin{vmatrix} u_1 & u_2 \\ v_1 & v_2 \end{vmatrix} \mathbf{k}$$

$$\begin{aligned} \mathbf{u} \times \mathbf{v} &= (u_2 v_3 \mathbf{i} + u_3 v_1 \mathbf{j} + u_1 v_2 \mathbf{k}) - (u_3 v_2 \mathbf{i} + u_1 v_3 \mathbf{j} + u_2 v_1 \mathbf{k}) \\ &= (u_2 v_3 - u_3 v_2)\mathbf{i} + (u_3 v_1 - u_1 v_3)\mathbf{j} + (u_1 v_2 - u_2 v_1)\mathbf{k}. \end{aligned}$$

Be careful, the cross product is anticommunative: $\mathbf{a} \times \mathbf{b} = -(\mathbf{b} \times \mathbf{a})$,

3-Links Planar Arm – Differential Kinematics

$$\boldsymbol{J} = \begin{bmatrix} -a_1 s_1 - a_2 s_{12} - a_3 s_{123} & -a_2 s_{12} - a_3 s_{123} & -a_3 s_{123} \\ a_1 c_1 + a_2 c_{12} + a_3 c_{123} & a_2 c_{12} + a_3 c_{123} & a_3 c_{123} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

- Physically, why only 3 rows are non-null?

Example 2

Anthropomorphic Arm - Differential Kinematic

$$\boldsymbol{J} = \begin{bmatrix} -s_1(a_2c_2 + a_3c_{23}) & -c_1(a_2s_2 + a_3s_{23}) & -a_3c_1s_{23} \\ c_1(a_2c_2 + a_3c_{23}) & -s_1(a_2s_2 + a_3s_{23}) & -a_3s_1s_{23} \\ 0 & a_2c_2 + a_3c_{23} & a_3c_{23} \\ 0 & s_1 & s_1 \\ 0 & -c_1 & -c_1 \\ 1 & 0 & 0 \end{bmatrix}$$

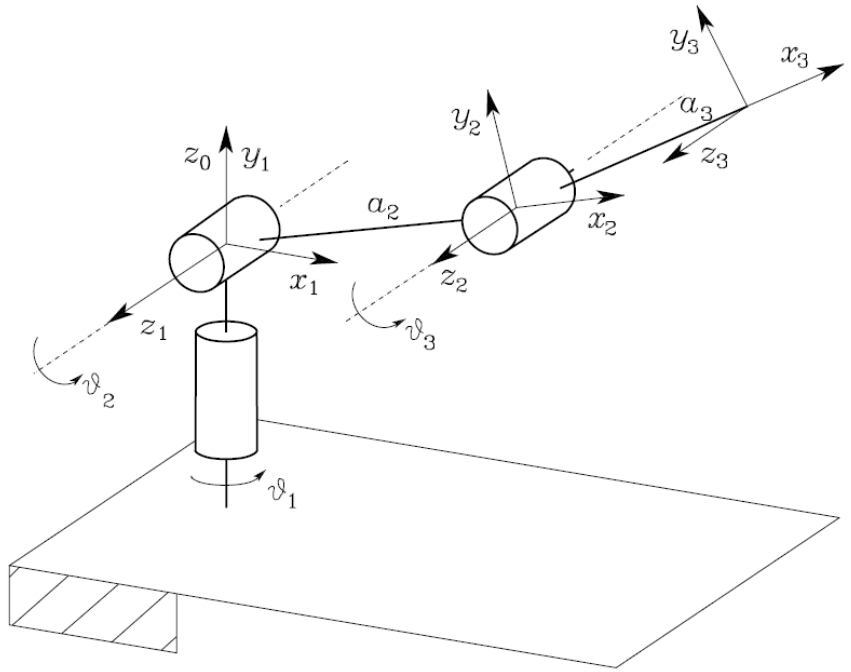
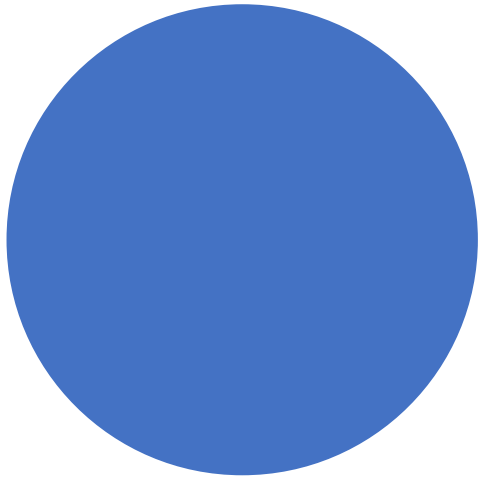
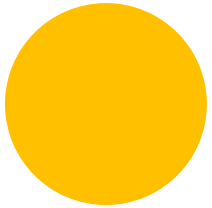


Fig. 2.23. Anthropomorphic arm

- J at most has Rank 3. We cannot have 6 independent velocities. The structure does not allow an arbitrary angular velocity

Geometrical vs Analytical Jacobian

Geometrical and Analytical Jacobian

- The geometrical Jacobian is computed without considering a specific minimal orientation representation.

$$v_e = \begin{bmatrix} \dot{p}_e \\ \omega_e \end{bmatrix} = J(q)\dot{q}$$

- The analytical Jacobian is computed w.r.t. a specific minimal orientation representation (e.g. Euler angles)

$$x = k(q) = \begin{bmatrix} p(q) \\ \phi(q) \end{bmatrix} \qquad \dot{x} = \frac{\partial k(q)}{\partial q}\dot{q} = J_A(q)\dot{q}$$

Geometrical vs Analytical Jacobian

- The two Jacobian are in general different because the angular velocity, in genera is not equal to the derivative of the minimal representation of the orientation (e.g. Euler Angles derivative)

$$\mathbf{J}_A \neq \mathbf{J} \qquad \dot{\omega} \neq \dot{\phi}$$

- However, a relationship between the two angular velocity exist. Give the Euler angles ZYZ, the relation is

$$\boldsymbol{\omega}_e = \boldsymbol{T}(\boldsymbol{\phi}_e)\dot{\boldsymbol{\phi}}_e \qquad \boldsymbol{T} = \begin{bmatrix} 0 & -s_\varphi & c_\varphi s_\vartheta \\ 0 & c_\varphi & s_\varphi s_\vartheta \\ 1 & 0 & c_\vartheta \end{bmatrix}$$

Geometrical vs Analytical Jacobian

- Therefore, give an analytical Jacobian we can compute the geometrical Jacobian as

$$\boldsymbol{J} = \boldsymbol{T}_A(\boldsymbol{\phi})\boldsymbol{J}_A$$

$$\begin{bmatrix} \boldsymbol{I} & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{T}(\boldsymbol{\phi}_e) \end{bmatrix}$$

- The inverse is possible except for the singular configurations where the relation is not invertible $\vartheta = 0, \pi$
- Such angles are **representation singularities** of $\boldsymbol{\phi}_e$

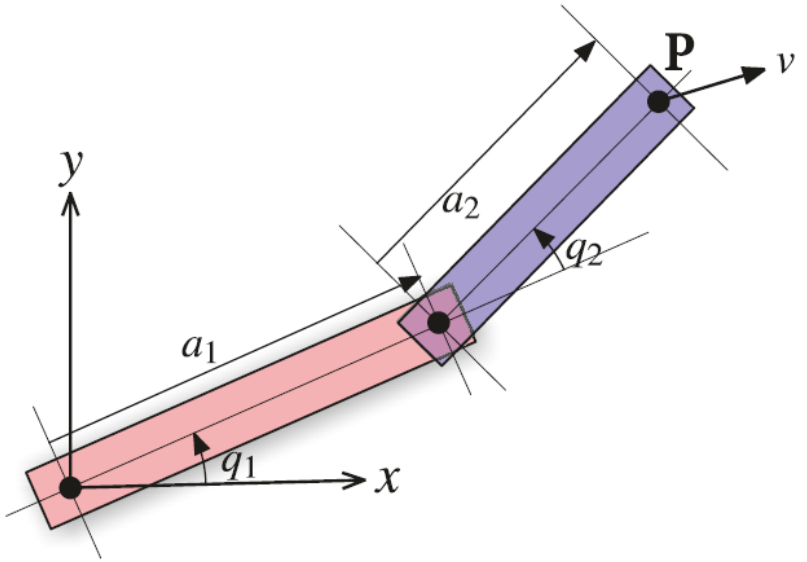
Geometrical vs Analytical Jacobian

- The geometric Jacobian will be adopted whenever it is necessary to refer to angular velocities of clear physical meaning

- The analytical Jacobian will be adopted whenever it is necessary to refer to differential quantities of variables defined in the operational space.

Kinematic Singularities

Kinematic Singularities



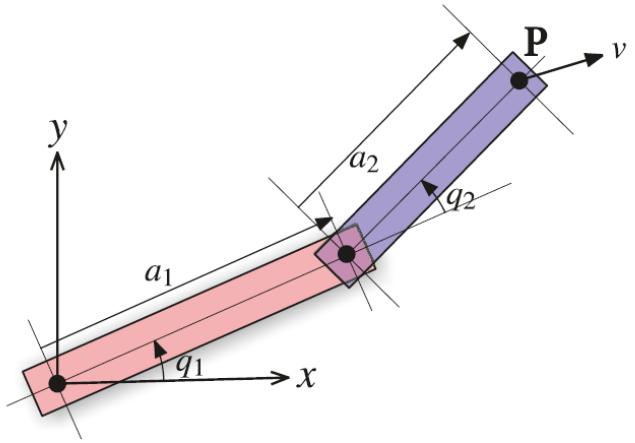
- The Jacobian defines the following mapping

$$\boldsymbol{v}_e = \boldsymbol{J}(\boldsymbol{q})\dot{\boldsymbol{q}} \qquad \boldsymbol{v}_e = [\,\dot{\boldsymbol{p}}_e^T \quad \boldsymbol{\omega}_e^T\,]^T$$

- The relation is a function of the current joint variables
- Values of **q** that make **J(q)** rank deficient are called **kinematic singularities**

Kinematic Singularities: Interpretations

$$\boldsymbol{v}_e = \boldsymbol{J}(\boldsymbol{q})\dot{\boldsymbol{q}}$$



- Interpretations:
  - Reduced Mobility configurations
  - Possible infinite solution to inverse kinematics
  - In the proximity of singularity, small velocities in the operational space may cause large velocities in the joint space

- Where we can find singularities
  - Boundary: either outstretched or retracted
  - Internal: It happens when two of more axes of motion are aligned or particular end-effector configuration

Kinematic Singularities: 2-links planar robot

- If just the linear velocities are of interest the Jacobian is

$$\boldsymbol{J} = \begin{bmatrix} -a_1 s_1 - a_2 s_{12} & -a_2 s_{12} \\ a_1 c_1 + a_2 c_{12} & a_2 c_{12} \end{bmatrix}$$

- How we find singularities:

$$\det(\boldsymbol{J}) = a_1 a_2 s_2$$

- Singularities (null determinant configurations)

$$\vartheta_2 = 0 \qquad \vartheta_2 = \pi$$

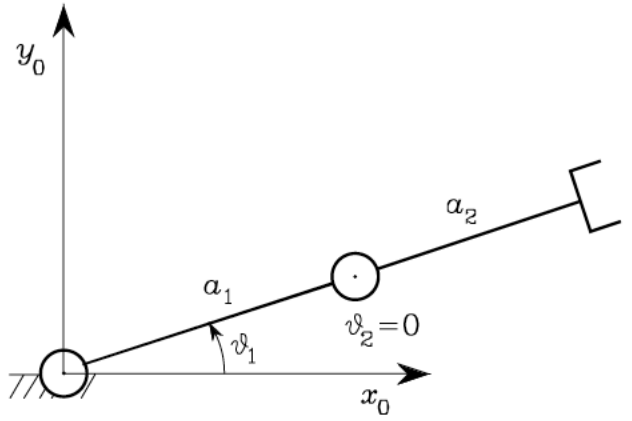


Fig. 3.3. Two-link planar arm at a boundary singularity

Inverse Kinematics with Differential Kinematics

Recap From the Previous Lecture

Recap: Inverse Kinematics Problem:

- The equations to solve are in general nonlinear, and thus it is not always possible to find a closed-form solution .
- Multiple solutions may exist.
- Infinite solutions may exist, e.g., in the case of a kinematically redundant manipulator.

Recap: Solution of the Inverse Kinematics Problem

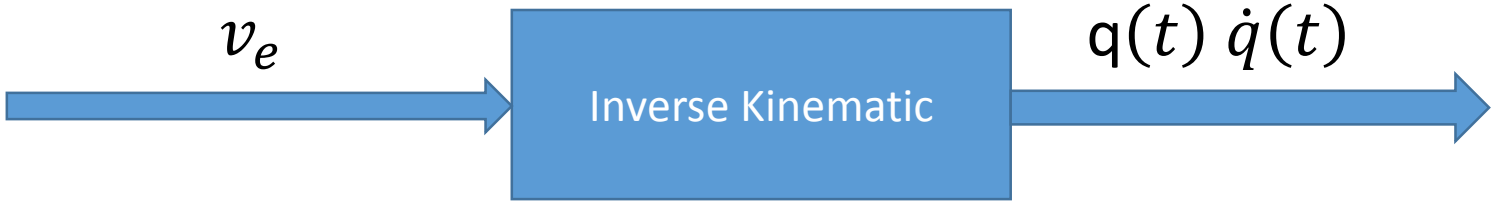
- Closed Form Solution
  - Requires algebraic intuition to find those significant equations containing the unknowns
  - We can find all the admissible solutions (the entire set of solutions)
  - Exists for Standard and Simple robot Configurations

- Numerical Solution
  - In all those cases when it is difficult to find closed-form solutions.
  - Have the advantage of being applicable to any kinematic structure, but in general they do not allow computation of all admissible solutions (it will find one solution belonging to the set admissible solutions (which is not known))

Inverse Differential Kinematics

$$\boldsymbol{v}_e = \boldsymbol{J}(\boldsymbol{q})\dot{\boldsymbol{q}}$$

- The differential kinematics equations represent an instantaneous linear mapping between the joint variables and the operational velocity space
- Can we exploit this linear mapping to compute inverse kinematics?



Inverse Differential Kinematics

$$\boldsymbol{v}_e = \boldsymbol{J}(\boldsymbol{q})\dot{\boldsymbol{q}}$$

- If n=r (robot not intrinsically redundant) and **J(q)** invertible (no singularities), the joints velocities can be obtained as

$$\dot{\boldsymbol{q}} = \boldsymbol{J}^{-1}(\boldsymbol{q})\boldsymbol{v}_e$$

- If we integrate both side of the last equation and we know the initial robot posture (**q**(0)) then we obtain

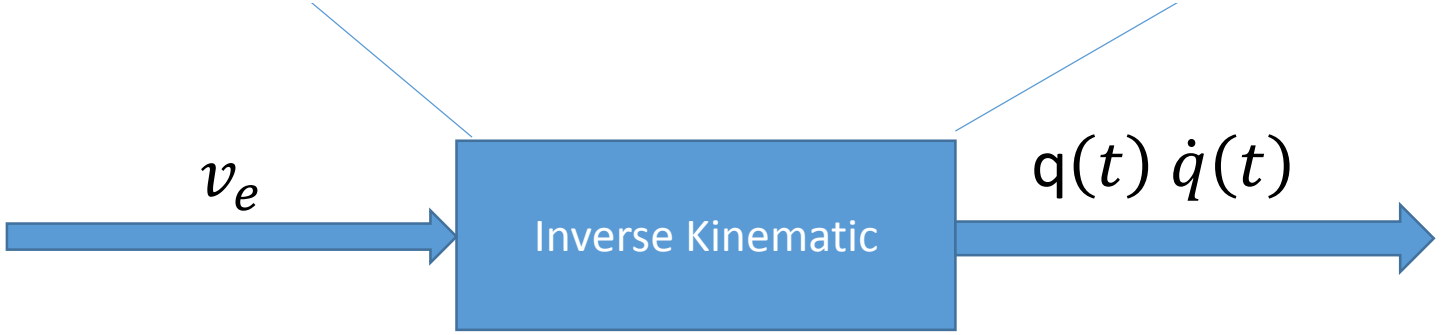
$$\boldsymbol{q}(t) = \int_0^t \dot{\boldsymbol{q}}(\varsigma)d\varsigma + \boldsymbol{q}(0)$$

Inverse Kinematics: Numerical Computation

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

- We can numerically compute the integral by resorting to the Euler integration method (Δt is the sampling time: $t_{k+1} = t_k + \Delta t$)

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



Inverse Kinematics with Differential Kinematics

- **Can we always apply this technique?**

Inverse Kinematics with Differential Kinematics

- Can we always apply this technique?

- **We have to see what happens when the Jacobian is not invertible**

Inverse Kinematics Redundant Manipulators

$$\boldsymbol{v}_e = \boldsymbol{J}(\boldsymbol{q})\dot{\boldsymbol{q}}$$

- (r<n), and the Jacobian has more columns than rows.
- Infinite solutions exist to the inverse differential kinematics

- The problem is that we don’t have an unique solution. We need to pick one solution among all the available ones!

Inverse Kinematics Redundant Manipulators

$$\boldsymbol{v}_e = \boldsymbol{J}(\boldsymbol{q})\dot{\boldsymbol{q}}$$

- Idea: Why we don't pick the solution that minimize a given cost (I want to have an optimization problem that will tell me which one should to pick!)
- Optimization problem:

$$\dot{\mathbf{q}}^* = \arg\min_{\dot{\mathbf{q}}} g(\dot{\mathbf{q}}) \quad s.t.$$
$$\mathbf{v}_e = \mathbf{J}\dot{\mathbf{q}}$$

Inverse Kinematics Redundant Manipulators

$$\boldsymbol{v}_e = \boldsymbol{J}(\boldsymbol{q})\dot{\boldsymbol{q}}$$

$$\dot{\mathbf{q}}^* = \arg\min_{\dot{\mathbf{q}}} g(\dot{\mathbf{q}}) \quad s.t.$$
$$\mathbf{v}_e = \mathbf{J}\dot{\mathbf{q}}$$

- One way to do it: Pick the cost function *g* to minimize the the norm of the joints velocities

$$g(\dot{\boldsymbol{q}}) = \frac{1}{2}\dot{\boldsymbol{q}}^T \boldsymbol{W}\dot{\boldsymbol{q}}$$

- If **W=I** and **J** has full-rank, then the solution to the inverse differential kinematics is (**Least Mean Square** (LMS) solution):

$$\dot{\boldsymbol{q}} = \boldsymbol{J}^{\dagger}\boldsymbol{v}_e \qquad \boldsymbol{J}^{\dagger} = \boldsymbol{J}^T(\boldsymbol{J}\boldsymbol{J}^T)^{-1}$$

- $\boldsymbol{J}^{\dagger}$ is known as the right pseudo inverse of J

Inverse Kinematics: J not full rank

$$\boldsymbol{v}_e = \boldsymbol{J}(\boldsymbol{q})\dot{\boldsymbol{q}}$$

- In this case, the manipulator is at a singular configuration and ***J(q)*** contains linearly dependent equations.
- Two case are possible:
  1. $\boldsymbol{v}_e \in \mathcal{R}(\boldsymbol{J})$ In this case we can still compute the solution by extracting the linear independent equations
  2. Otherwise, the inverse problem has no solution.
- It is important to notice that we have also problems around the singularities points because small velocities in the operational space correspond to huge velocities in the joint space (**why?**)

Inverse Kinematics: general solution (all cases)

$$\dot{\mathbf{q}} = \mathbf{J}^* \mathbf{v}_e$$

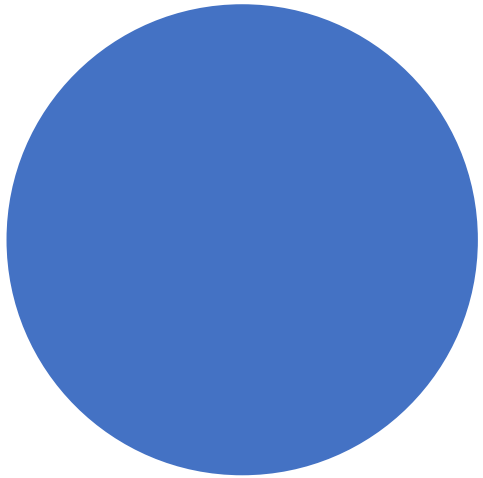
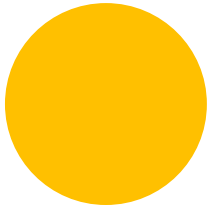
$$\boldsymbol{J}^\star = \boldsymbol{J}^T(\boldsymbol{J}\boldsymbol{J}^T + k^2\boldsymbol{I})^{-1}$$

$$\boldsymbol{q}(t_{k+1}) = \boldsymbol{q}(t_k) + \dot{\boldsymbol{q}}(t_k)\Delta t$$

- A general solution suitable for all cases is called **Damped-Least Squares (DLS) inverse**
- Basically, it comes out when we define an optimization problem where the cost function is the following (optimal ways to pick the k value exist)

$$g''(\dot{\boldsymbol{q}}) = \frac{1}{2}(\boldsymbol{v}_e - \boldsymbol{J}\dot{\boldsymbol{q}})^T(\boldsymbol{v}_e - \boldsymbol{J}\dot{\boldsymbol{q}}) + \frac{1}{2}k^2\dot{\boldsymbol{q}}^T\dot{\boldsymbol{q}}$$

1. render the inversion problem better conditioned (when we are close to singularities)
2. allow a finite inversion error to be tolerated (no exact solution for singularity configurations.)

Inverse Kinematic Algorithms

Inverse Kinematics Algorithms

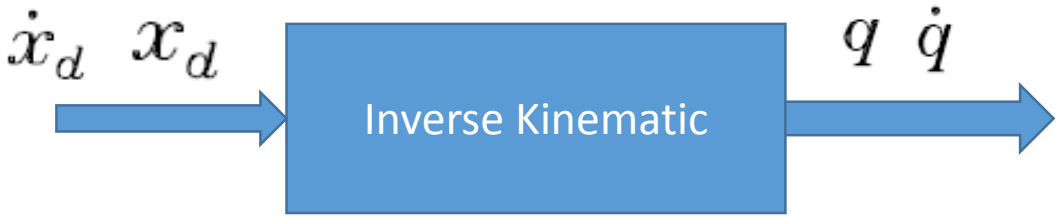
- We have seen, that we can obtain inverse kinematics as follows

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

- However, this represents a discrete approximation that brings with it some drift errors.
- This means that the outcome (end-effector pose) might differ from the desired one
- With this formula we don't have any way to correct the error (if any), it is like open-loop control!!

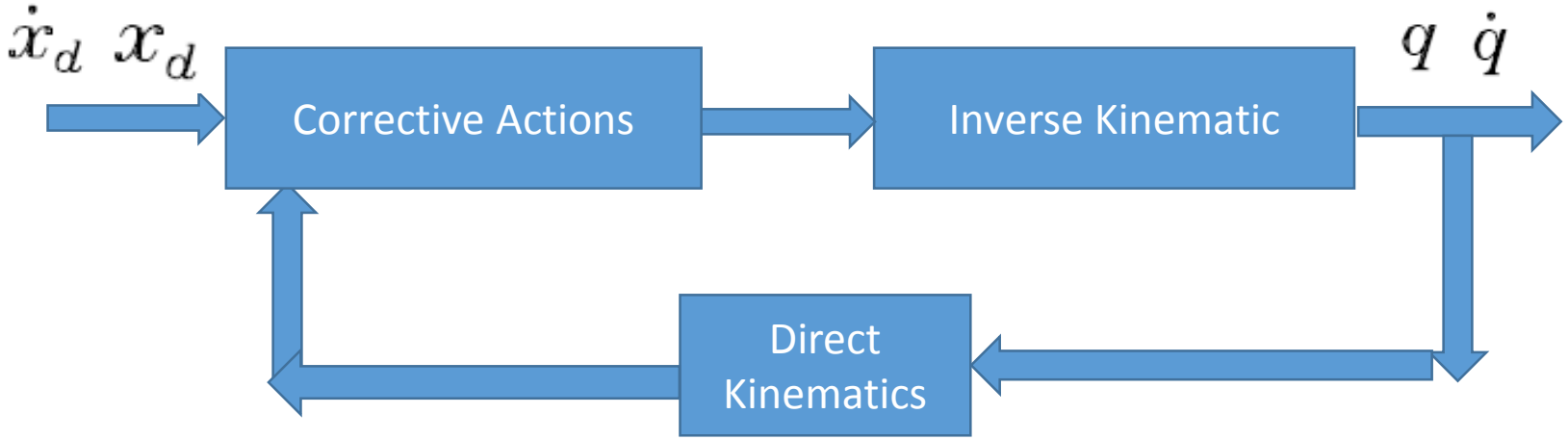
Improved Inverse Kinematics Algorithms

- Can we use feedback control idea to improve the inverse kinematics computation?



Close Loop Inverse Kinematics Algorithms

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



Closed Loop Kinematics Algorithm (if the Jacobian is invertible)

- Let's consider the operational space error

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

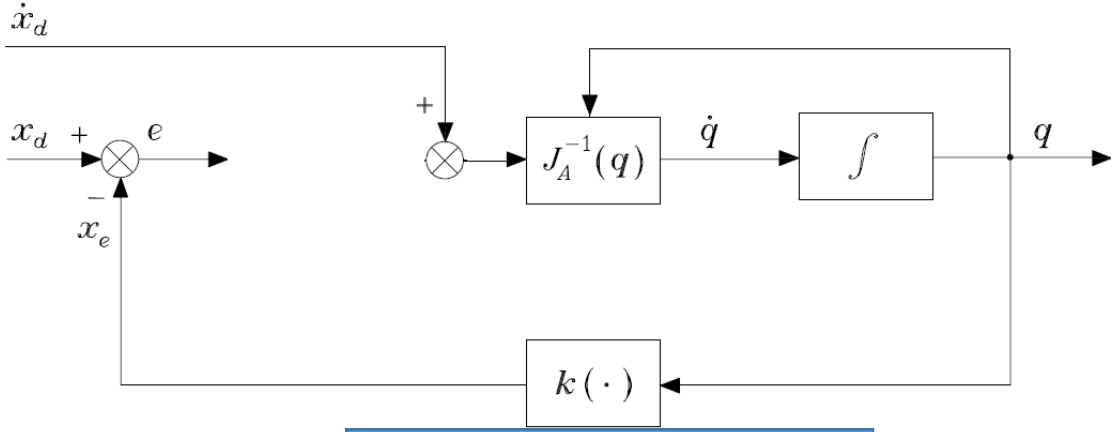
- Its derivative is $\dot{\boldsymbol{e}} = \dot{\boldsymbol{x}}_d - \dot{\boldsymbol{x}}_e$. By using the analytical Jacobian we have

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

- 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, we have

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

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



Closed Loop Kinematics Algorithm (if the Jacobian is invertible)

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

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

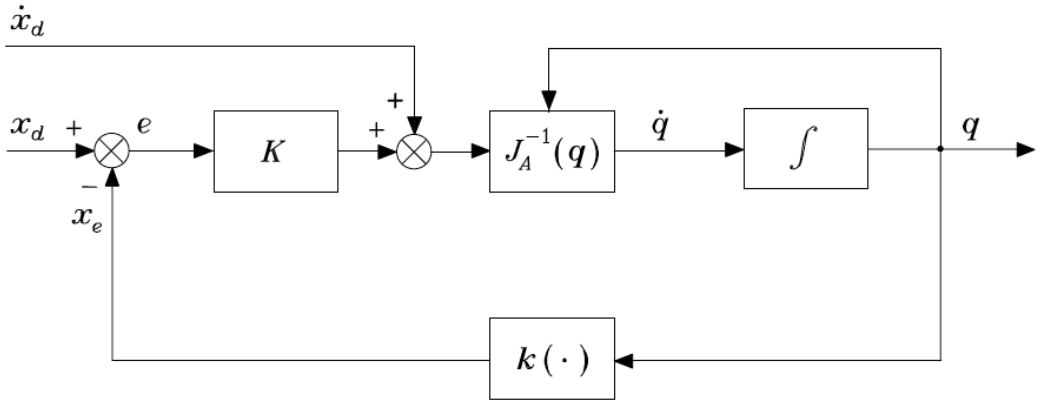


Fig. 3.11. Inverse kinematics algorithm with Jacobian inverse

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

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

Improved Inverse Kinematics Algorithm (without inversion of the Jacobian)

- It is possible to prove (Lyapunov theorem) 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 computation only of direct kinematics function is needed (no inversions need to be computed)

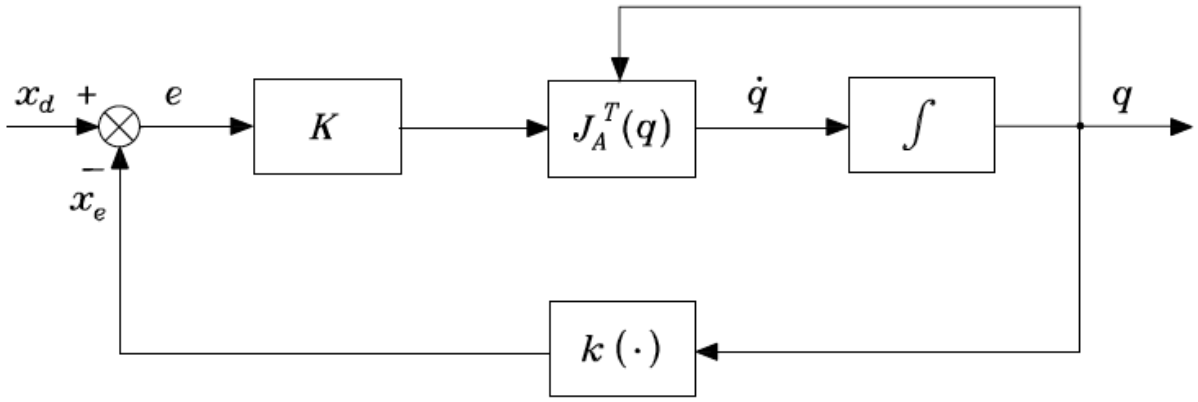


Fig. 3.12. Block scheme of the inverse kinematics algorithm with Jacobian transpose

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

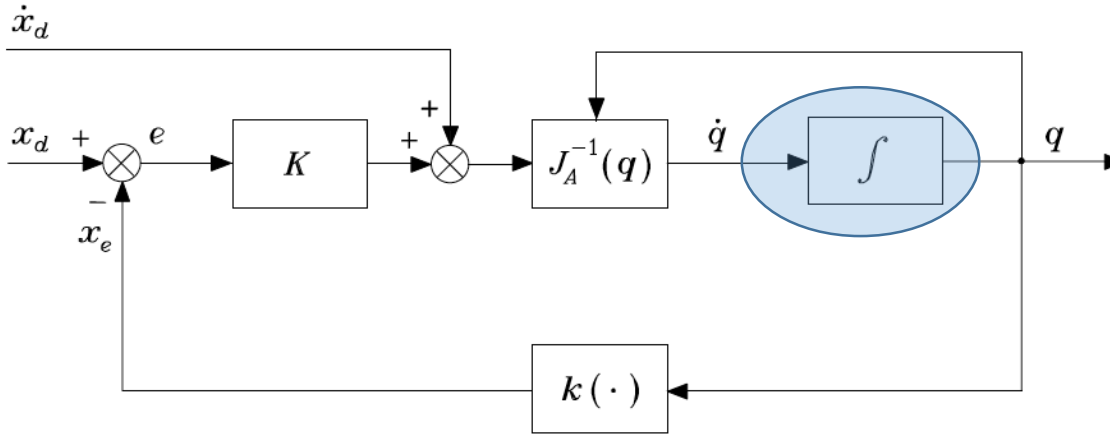


Fig. 3.11. Inverse kinematics algorithm with Jacobian inverse

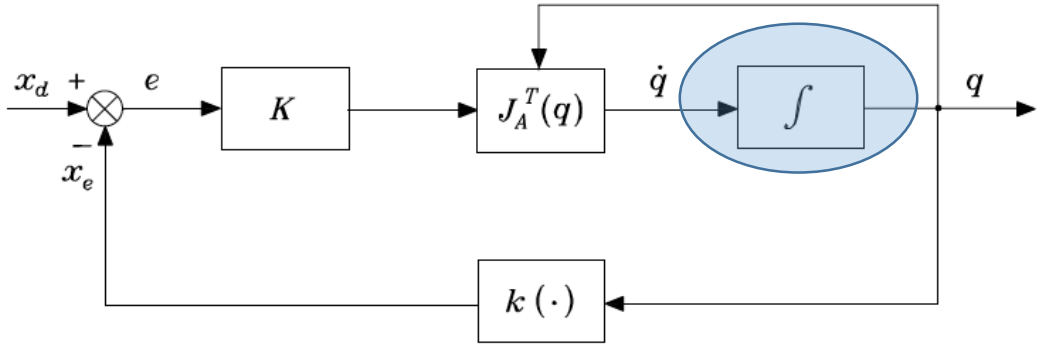


Fig. 3.12. Block scheme of the inverse kinematics algorithm with Jacobian transpose

THANK YOU!