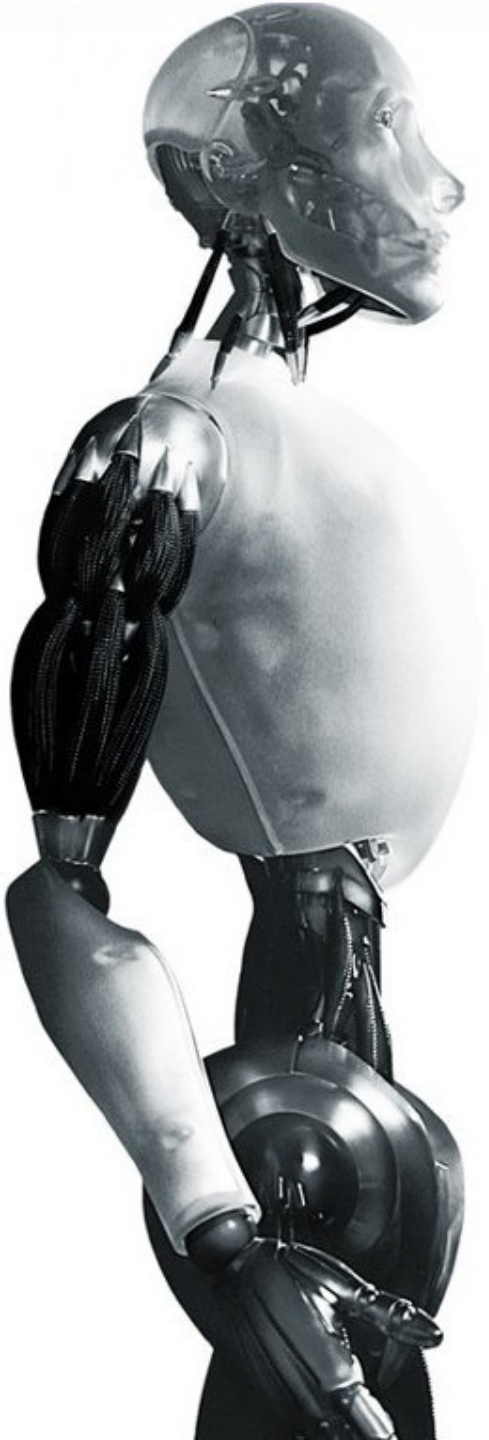


Disclaimer

These slides are intended as presentation aids for the lecture. They contain information that would otherwise be too difficult or time-consuming to reproduce on the board. But they are incomplete, not self-explanatory, and are not always used in the order they appear in this presentation. As a result, these slides should not be used as a script for this course. I recommend you take notes during class, maybe on the slides themselves. It has been shown that taking notes improves learning success.



Robotics

VIP: Jacobian

TU Berlin
Oliver Brock

Reading for this set of slides

- Craig – Intro to Robotics (3rd Edition)
 - Chapter 5 (5.1 – 5.10)

Please note that this set of slides is intended as support for the lecture, not as a stand-alone script. If you want to study for this course, please use these slides in conjunction with the indicated chapters in the text books. The textbooks are available online or in the TUB library (many copies that can be checked out for the entire semester. There are also some aspects of the lectures that will not be covered in the text books but can still be part of the homework or exam. For those It is important that you attend class or ask somebody about what was covered in class.

Kinematics

- Forward: Where is the end-effector based on a given configuration?
- Inverse: Where do the joints have to be to place the end-effector? (We did not cover this yet)
- Now: How does joint **motion** and end-effector **motion** relate?
 - before, we considered static configurations
 - we start to consider linear and angular velocities, as well as static forces

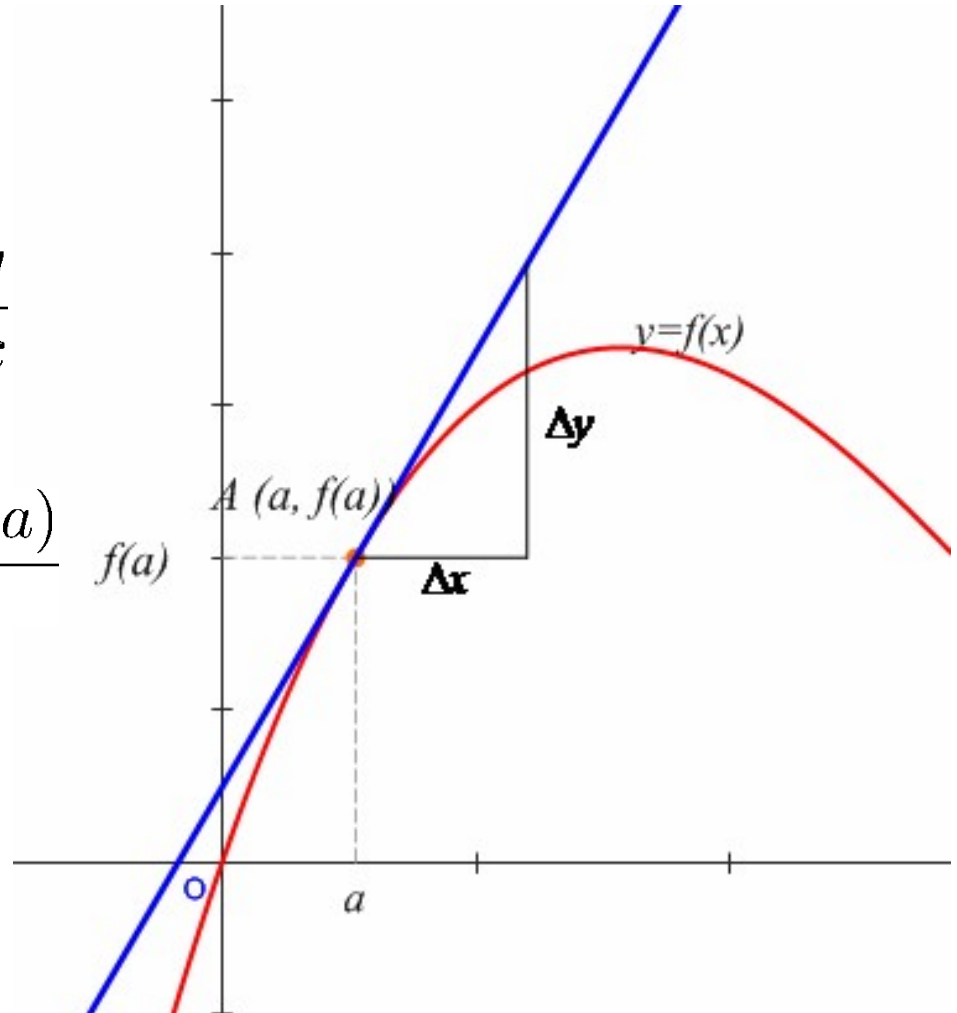
Notation

- To make equations more readable we will adopt the following:
 - Vector: $\mathbf{x}_{(n \times 1)} = (x_1, x_2, \dots, x_n)^T$
 - Matrix: $M_{(n \times m)} = [\mathbf{x}_1 \mathbf{x}_2 \dots \mathbf{x}_m]$
 - Derivative of vectors and matrices: $\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}$
 - We'll introduce more as we go along...

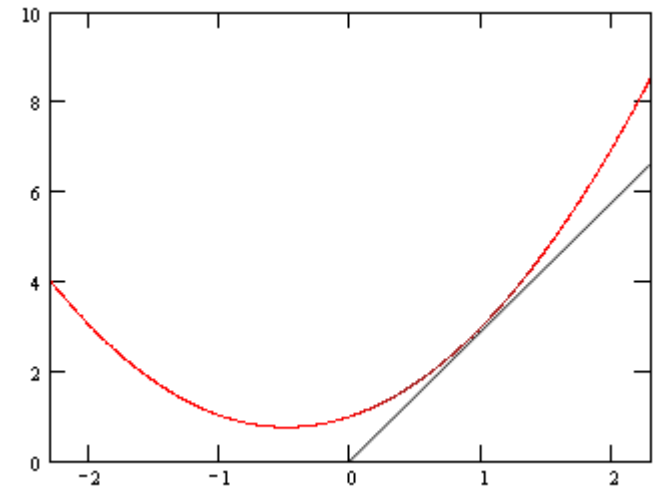
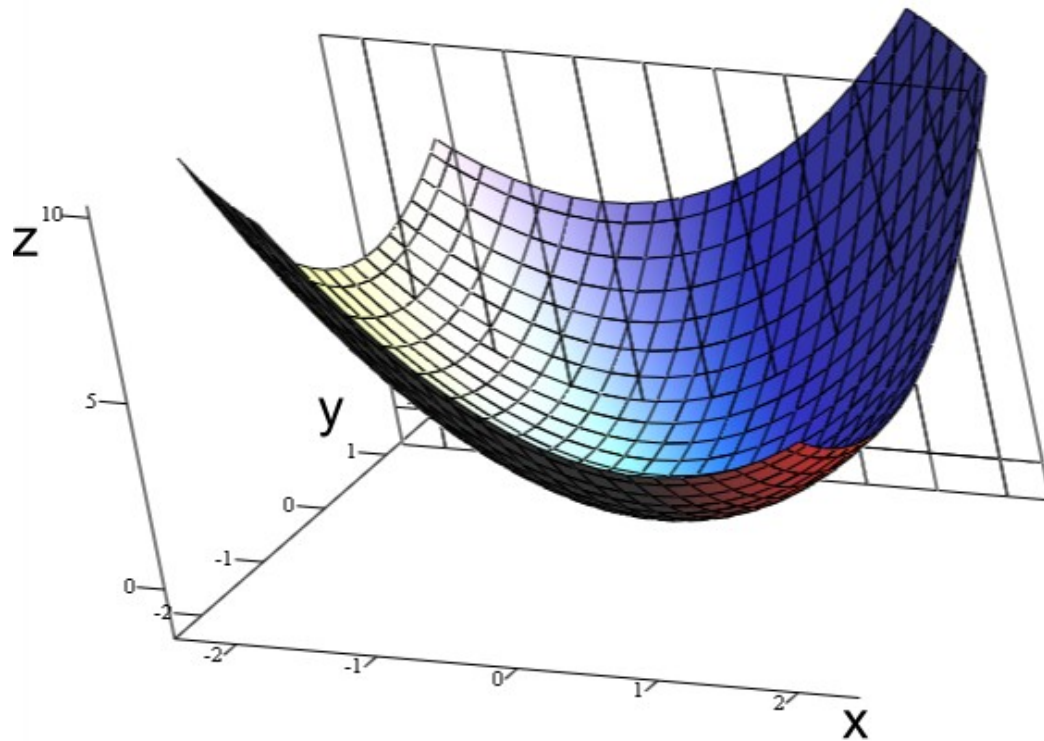
Warm-Up: Derivative

$$m = \frac{\text{change in } y}{\text{change in } x} = \frac{\Delta y}{\Delta x}$$

$$f'(a) = \lim_{h \rightarrow 0} \frac{f(a+h) - f(a)}{h}$$



Warm-Up: Partial Derivative



$$f'(a) = \lim_{h \rightarrow 0} \frac{f(a+h) - f(a)}{h}$$

$$\frac{\partial f}{\partial x_i}(a_1, \dots, a_n) = \lim_{h \rightarrow 0} \frac{f(a_1, \dots, a_i + h, \dots, a_n) - f(a_1, \dots, a_i, \dots, a_n)}{h}$$

Rewriting Forward Kinematics

$$\mathbf{x} = f(\mathbf{q}) = {}^0_nT(\mathbf{q}) = {}^0_1T(q_1) {}^1_2T(q_2) {}^2_3T(q_3) \cdots {}^{n-1}_nT(q_n)$$

$$\begin{aligned}x_1 &= f_1((q_1, q_2, \cdots, q_n)^T) \\x_2 &= f_2((q_1, q_2, \cdots, q_n)^T) \\x_3 &= f_3((q_1, q_2, \cdots, q_n)^T) \\&\vdots \\x_m &= f_m((q_1, q_2, \cdots, q_n)^T)\end{aligned}$$

$$x_i = f_i((q_1, q_2, \cdots, q_n)^T)$$

Infinitesimal Motion

$$\mathbf{x}_{(6 \times 1)} = \begin{pmatrix} \mathbf{x}_p(3 \times 1) \\ \mathbf{x}_r(3 \times 1) \end{pmatrix} \quad \mathbf{q}_{(n \times 1)} = \begin{pmatrix} q_1 \\ q_2 \\ \vdots \\ q_n \end{pmatrix}$$

$$\mathbf{x} = f(\mathbf{q})$$

$$\mathbf{q} = f^{-1}(\mathbf{x})$$

$$\delta \mathbf{x} = g(\delta \mathbf{q})$$

$$g = ?$$

Partial Derivatives of f

Static:

$$\mathbf{x} = f(\mathbf{q}) = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{pmatrix} = \begin{pmatrix} f_1(\mathbf{q}) \\ f_2(\mathbf{q}) \\ \vdots \\ f_m(\mathbf{q}) \end{pmatrix}$$

$$x_i = f_i((q_1, q_2, \dots, q_n)^T)$$

Motion:

$$\delta \mathbf{x} = \dot{f}(\delta \mathbf{q})$$

$$\delta \mathbf{q} = \begin{pmatrix} \delta q_1 \\ \delta q_2 \\ \vdots \\ \delta q_n \end{pmatrix}$$

$$\delta x_1 = \frac{\partial f_1}{\partial q_1} \delta q_1 + \frac{\partial f_1}{\partial q_2} \delta q_2 + \dots + \frac{\partial f_1}{\partial q_n} \delta q_n$$

$$\delta x_2 = \frac{\partial f_2}{\partial q_1} \delta q_1 + \frac{\partial f_2}{\partial q_2} \delta q_2 + \dots + \frac{\partial f_2}{\partial q_n} \delta q_n$$

$$\vdots$$

$$\delta x_m = \frac{\partial f_m}{\partial q_1} \delta q_1 + \frac{\partial f_m}{\partial q_2} \delta q_2 + \dots + \frac{\partial f_m}{\partial q_n} \delta q_n$$

How does the end-effector move?

The Jacobian Matrix

$$\delta x_1 = \frac{\partial f_1}{\partial q_1} \delta q_1 + \frac{\partial f_1}{\partial q_2} \delta q_2 + \cdots + \frac{\partial f_1}{\partial q_n} \delta q_n$$

$$\delta x_2 = \frac{\partial f_2}{\partial q_1} \delta q_1 + \frac{\partial f_2}{\partial q_2} \delta q_2 + \cdots + \frac{\partial f_2}{\partial q_n} \delta q_n$$

$$\vdots$$

$$\delta x_m = \frac{\partial f_m}{\partial q_1} \delta q_1 + \frac{\partial f_m}{\partial q_2} \delta q_2 + \cdots + \frac{\partial f_m}{\partial q_n} \delta q_n$$

$$J = \frac{\partial f(\mathbf{q})}{\partial \mathbf{q}}$$

$$\delta \mathbf{x} = \begin{bmatrix} \frac{\partial f_1}{\partial q_1} & \frac{\partial f_1}{\partial q_2} & \cdots & \frac{\partial f_1}{\partial q_n} \\ \frac{\partial f_2}{\partial q_1} & \frac{\partial f_2}{\partial q_2} & \cdots & \frac{\partial f_2}{\partial q_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial q_1} & \frac{\partial f_m}{\partial q_2} & \cdots & \frac{\partial f_m}{\partial q_n} \end{bmatrix} \delta \mathbf{q} = J_{(m \times n)}(\mathbf{q}) \delta \mathbf{q}$$

Jacobian and Velocities

$$\lim_{\delta t \rightarrow 0} \left(\frac{\delta \mathbf{x}}{\delta t} = J(\mathbf{q}) \frac{\delta \mathbf{q}}{\delta t} \right)$$

$$\dot{\mathbf{x}} = J(\mathbf{q}) \dot{\mathbf{q}}$$

$$\delta \mathbf{x}_{(m \times 1)} = J_{(m \times n)}(\mathbf{q}) \delta \mathbf{q}_{(n \times 1)}$$

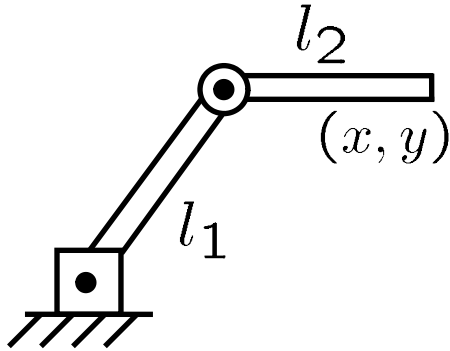
$$\dot{\mathbf{x}}_{(m \times 1)} = J_{(m \times n)}(\mathbf{q}) \dot{\mathbf{q}}_{(n \times 1)}$$

Reminder:

$$\sin'(\theta) = \cos(\theta)$$

$$\cos'(\theta) = -\sin(\theta)$$

Jacobian: Example



$$\mathbf{x} = f(\mathbf{q}) \quad \mathbf{x} = (x, y)^T \quad \mathbf{q} = (\theta_1, \theta_2)^T$$

$$x = f_1(\mathbf{q}) = l_1 c_1 + l_2 c_{12}$$

$$y = f_2(\mathbf{q}) = l_1 s_1 + l_2 s_{12}$$

$$\frac{\partial f_1}{\partial \theta_1} = -(l_1 s_1 + l_2 s_{12})$$

$$\frac{\partial f_1}{\partial \theta_2} = -l_2 s_{12}$$

$$\frac{\partial f_2}{\partial \theta_1} = l_1 c_1 + l_2 c_{12}$$

$$\frac{\partial f_2}{\partial \theta_2} = l_2 c_{12}$$

$$J_{(2 \times 2)}(\mathbf{q}) = \begin{bmatrix} -(l_1 s_1 + l_2 s_{12}) & -l_2 s_{12} \\ l_1 c_1 + l_2 c_{12} & l_2 c_{12} \end{bmatrix}$$

$$\delta \mathbf{x} = J \left(\begin{pmatrix} 0 \\ 0 \end{pmatrix} \right) \begin{pmatrix} \delta \theta_1 \\ \delta \theta_2 \end{pmatrix}$$

$$= \begin{bmatrix} 0 & 0 \\ l_1 + l_2 & l_2 \end{bmatrix} \begin{pmatrix} \delta \theta_1 \\ \delta \theta_2 \end{pmatrix} = \begin{pmatrix} 0 \\ (l_1 + l_2)\delta \theta_1 + l_2 \delta \theta_2 \end{pmatrix}$$

What the Jacobian can do...

$$\dot{\mathbf{x}} = J(\mathbf{q}) \dot{\mathbf{q}}$$

$$\delta \mathbf{x} = J(\mathbf{q}) \delta \mathbf{q}$$

$$\delta \mathbf{q} = J(\mathbf{q})^{-1} \delta \mathbf{x}$$

Jacobian and Forces

work = force · distance

virtual work: virtual displacements with real force (or vice versa)

$$\mathbf{F}_{(6 \times 1)} \delta \mathbf{x} = \tau \delta \mathbf{q}$$

$$\mathbf{F}^T \delta \mathbf{x} = \tau^T \delta \mathbf{q} \quad \text{using matrix notation}$$

$$\mathbf{F}^T J \delta \mathbf{q} = \tau^T \delta \mathbf{q} \quad \text{using } \delta \mathbf{x} = J(\mathbf{q}) \delta \mathbf{q}$$

$$\mathbf{F}^T J = \tau^T$$

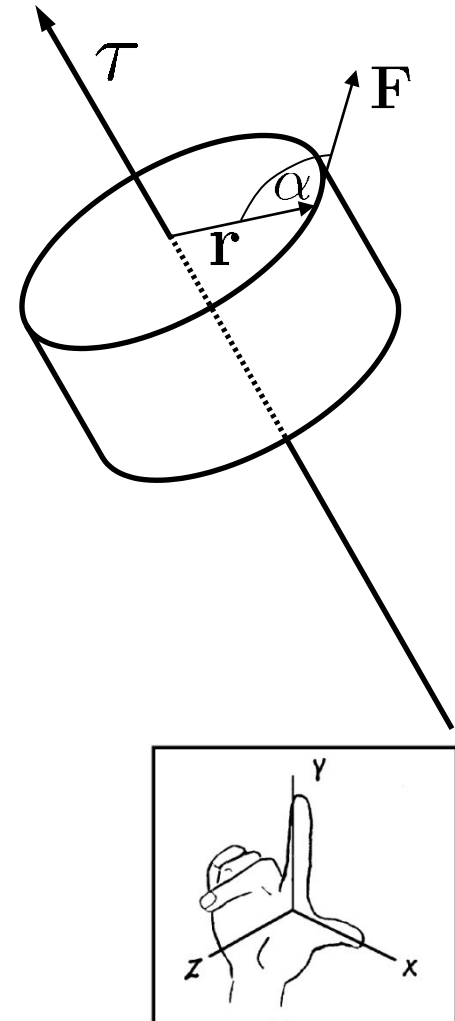
$$\tau = J^T \mathbf{F}$$

Reminder: torque/moment

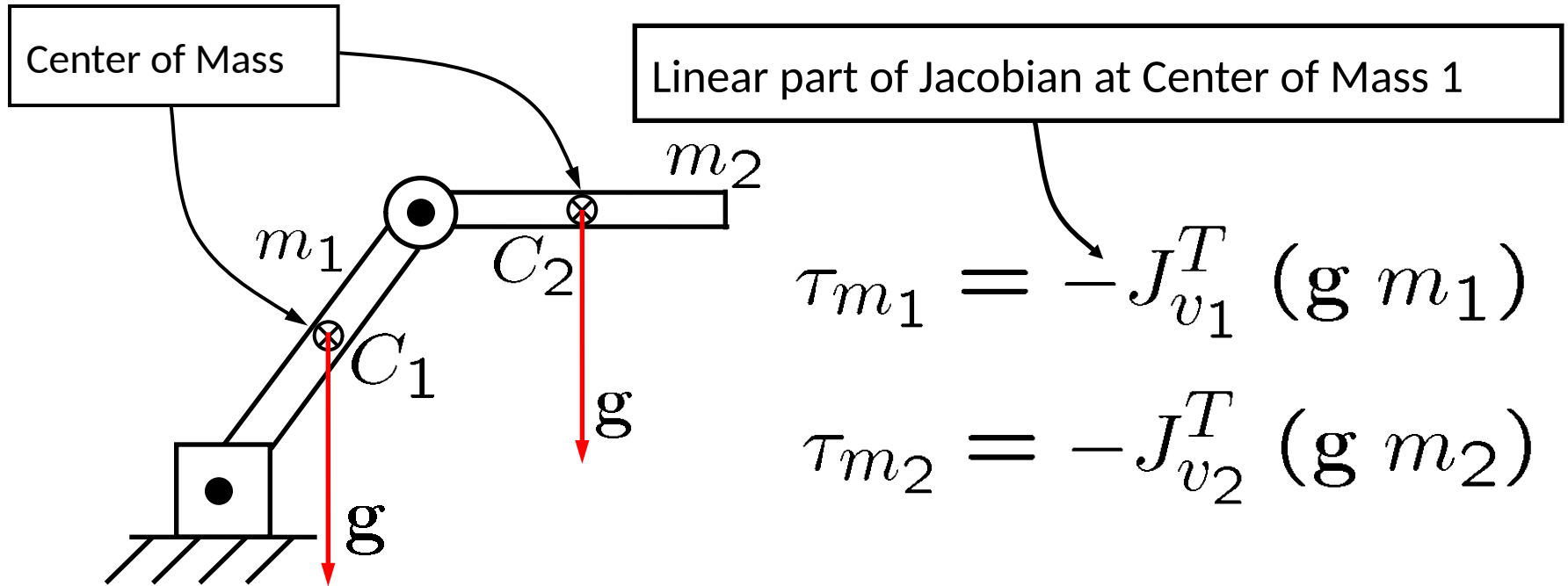
$$\boldsymbol{\tau} = \mathbf{r} \times \mathbf{F}$$

$$\boldsymbol{\tau} \perp \mathbf{F} \quad \boldsymbol{\tau} \perp \mathbf{r}$$

$$\|\boldsymbol{\tau}\| = \|\mathbf{F}\| \|\mathbf{r}\| \sin(\alpha)$$



Gravity Reloaded: With Jacobian

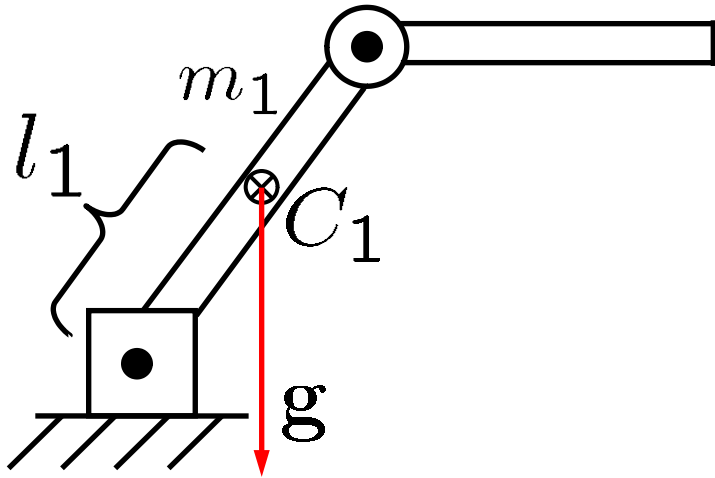


$$\tau_{m_1} = -J_{v_1}^T (g m_1)$$

$$\tau_{m_2} = -J_{v_2}^T (g m_2)$$

$$\mathbf{G} = - \sum_{i=0}^n J_{v_i}^T (g m_i)$$

Gravity Example



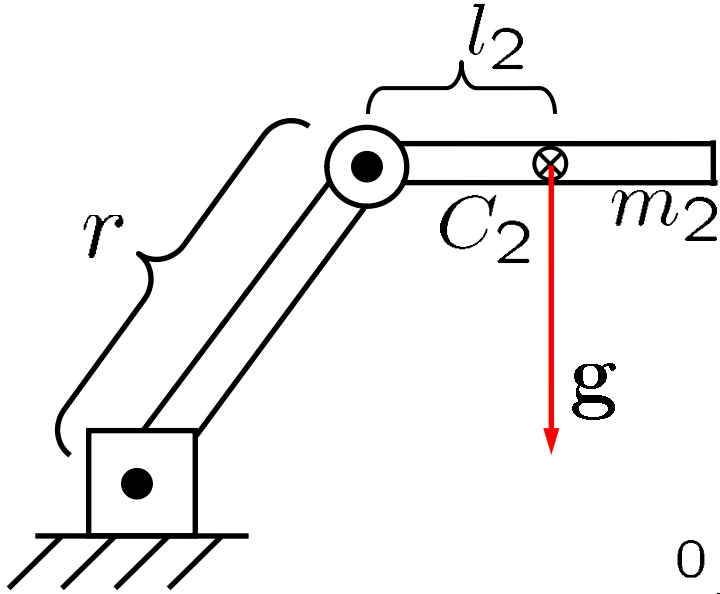
$$\tau_{m_1} = -J_{c_1}^T (g \ m_1)$$

$${}^0\mathbf{p}_{C_1} = \begin{pmatrix} l_1 \ c_1 \\ l_1 \ s_1 \end{pmatrix}$$

$${}^0J_{v_1} = \begin{bmatrix} -l_1 \ s_1 & 0 \\ l_1 \ c_1 & 0 \end{bmatrix}$$

$$\tau_{m_1} = - \begin{bmatrix} -l_1 \ s_1 & l_1 \ c_1 \\ 0 & 0 \end{bmatrix} \begin{pmatrix} 0 \\ -g \ m_1 \end{pmatrix} = \begin{pmatrix} l_1 \ c_1 \ g \ m_1 \\ 0 \end{pmatrix}$$

Gravity Example cont.



$$\tau_{m_2} = -J_{c_2}^T (g m_2)$$

$${}^0\mathbf{p}_{C_2} = \begin{pmatrix} r c_1 + l_2 c_{12} \\ r s_1 + l_2 s_{12} \end{pmatrix}$$

$${}^0J_{v_2} = \begin{bmatrix} -r s_1 - l_2 s_{12} & -l_2 s_{12} \\ r c_1 + l_2 c_{12} & l_2 c_{12} \end{bmatrix}$$

$$\tau_{m_2} = - \begin{bmatrix} -r s_1 - l_2 s_{12} & r c_1 + l_2 c_{12} \\ -l_2 s_{12} & l_2 c_{12} \end{bmatrix} \begin{pmatrix} 0 \\ -g m_2 \end{pmatrix} =$$

$$\begin{pmatrix} (r c_1 + l_2 c_{12}) g m_2 \\ l_2 c_{12} g m_2 \end{pmatrix}$$

What the Jacobian can do...

$$\dot{\mathbf{x}} = J(\mathbf{q}) \dot{\mathbf{q}}$$

$$\delta \mathbf{x} = J(\mathbf{q}) \delta \mathbf{q}$$

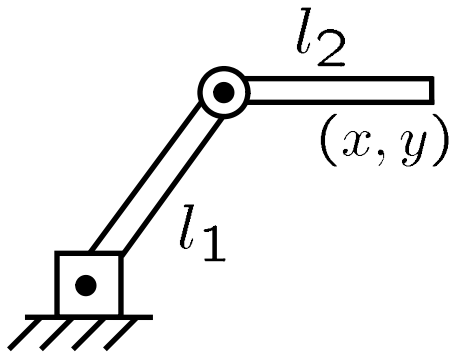
$$\dot{\mathbf{q}} = J(\mathbf{q})^{-1} \dot{\mathbf{x}}$$

$$\delta \mathbf{q} = J(\mathbf{q})^{-1} \delta \mathbf{x}$$

$$\boldsymbol{\tau} = J^T \mathbf{F}$$

My favorite formula!

Kinematic Singularities



The determinant does not change when the reference frame changes

$$\det J(q) = 0 \Rightarrow \text{singularity}$$

$${}^0J = \begin{bmatrix} -(l_1 s_1 + l_2 s_{12}) & -l_2 s_{12} \\ l_1 c_1 + l_2 c_{12} & l_2 c_{12} \end{bmatrix}$$

$${}^1J = {}^0R {}^0J = \begin{bmatrix} c_1 & -s_1 \\ s_1 & c_1 \end{bmatrix} {}^0J = \begin{bmatrix} -l_2 s_2 & -l_2 s_2 \\ l_1 + l_2 c_2 & l_2 c_2 \end{bmatrix}$$

$${}^1J((\theta_1, 0)^T) = \begin{bmatrix} 0 & 0 \\ l_1 + l_2 & l_2 \end{bmatrix}$$

2D Determinant

$$\det \begin{bmatrix} a & b \\ c & d \end{bmatrix} = ad - bc$$

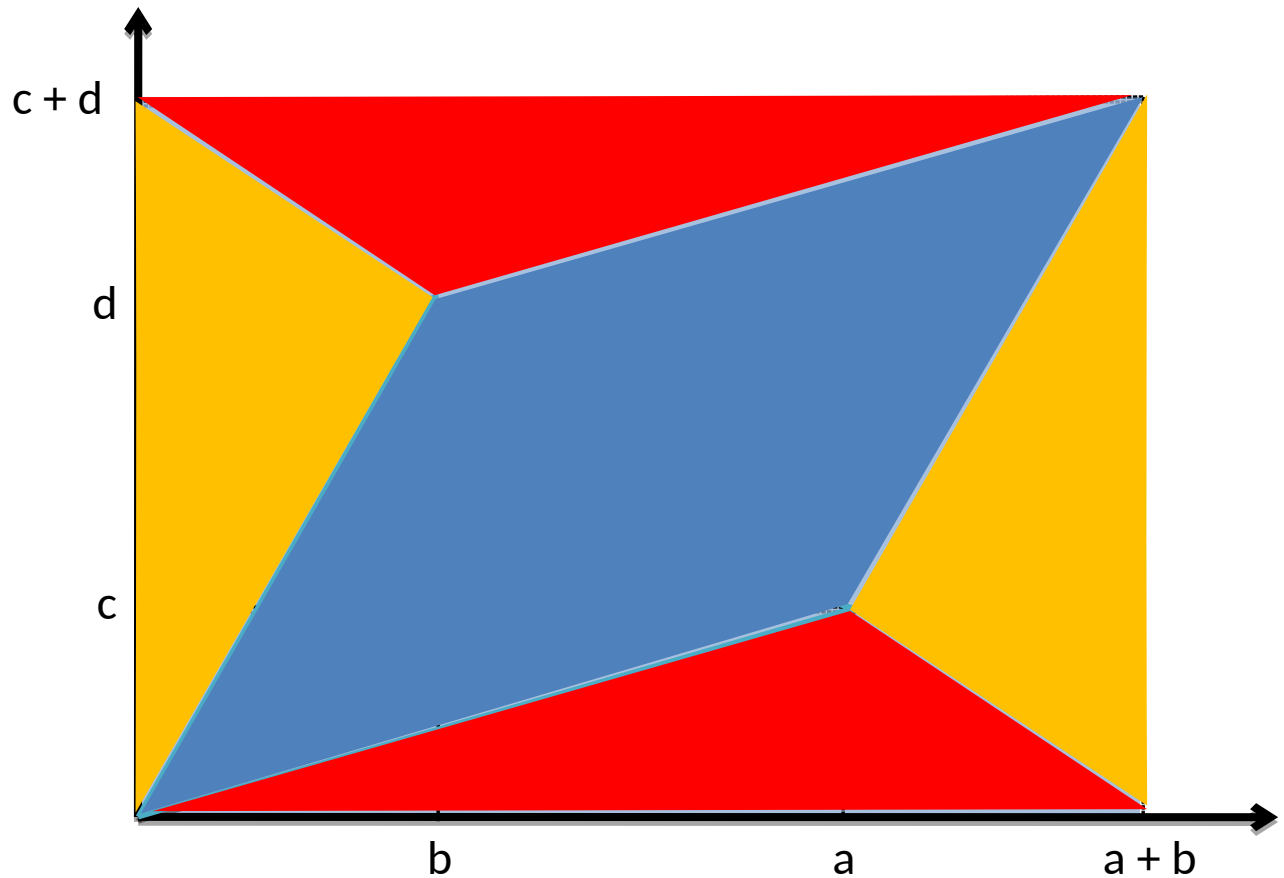
$$= (a + b)(c + d)$$

$$- (a + b)c$$

$$- b(c + d)$$

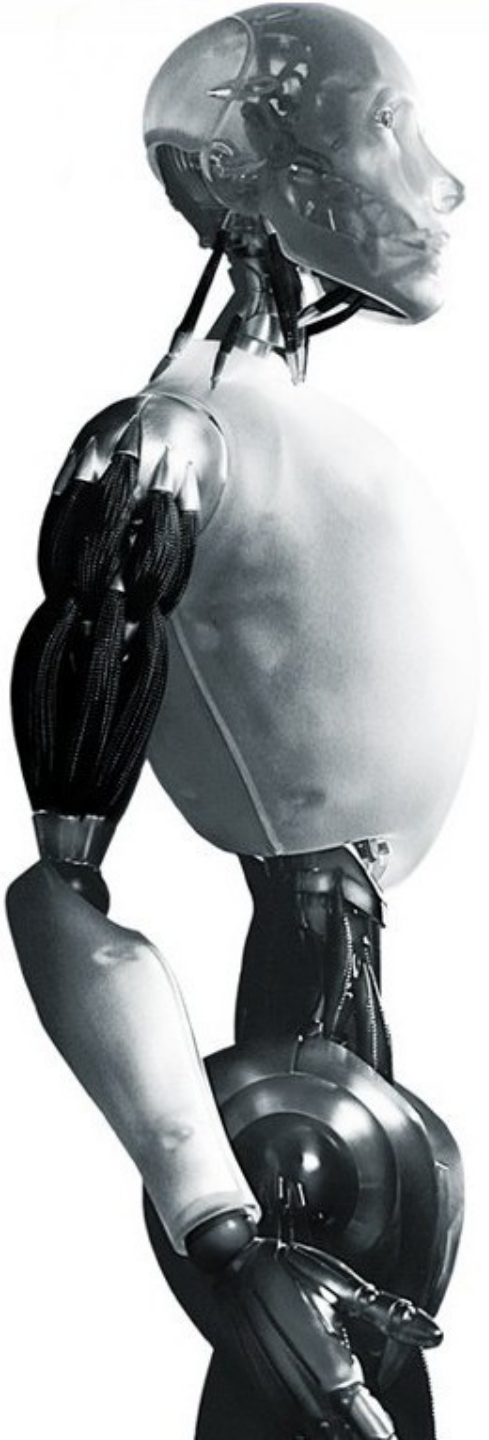
$$= ad + bd - bc - bd$$

$$= ad - bc$$



Kinematic Singularities

- If J loses rank, a singularity is encountered
- The determinant for rectangular (non-square) matrices is not defined (almost not defined)
- If J is rectangular (more columns than rows), the robot is redundant with respect to the task
- Singularity can be detected by determining the determinant or eigenvalues and eigenvectors

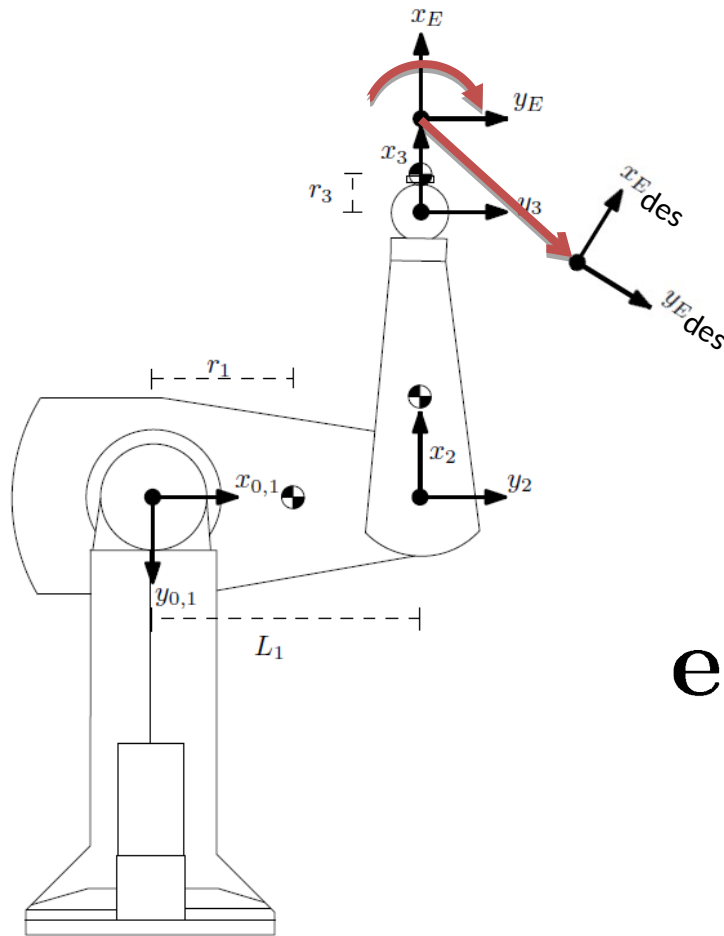


Robotics

First Steps in Operational Space Control

TU Berlin
Oliver Brock

Control of the End-Effector



$$\mathbf{e}_x = \mathbf{x}_{\text{des}} - \mathbf{x}$$

Resolved-Rate Motion Control

$$\dot{\mathbf{q}}^* = J(\mathbf{q})^{-1} \dot{\mathbf{x}}^*$$

$$\mathbf{q}_{t+1}^* = \mathbf{q}_t + \delta t \dot{\mathbf{q}}_t^*$$

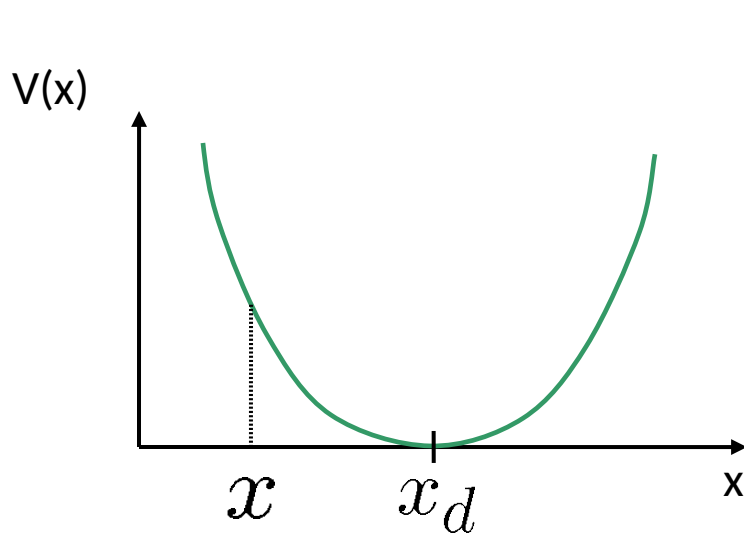
The * indicates a desired quantity as opposed to a measured one

P Control in Operational Space

Idea: apply force proportional to error

$$f = -\boxed{k_p}e$$

position gain



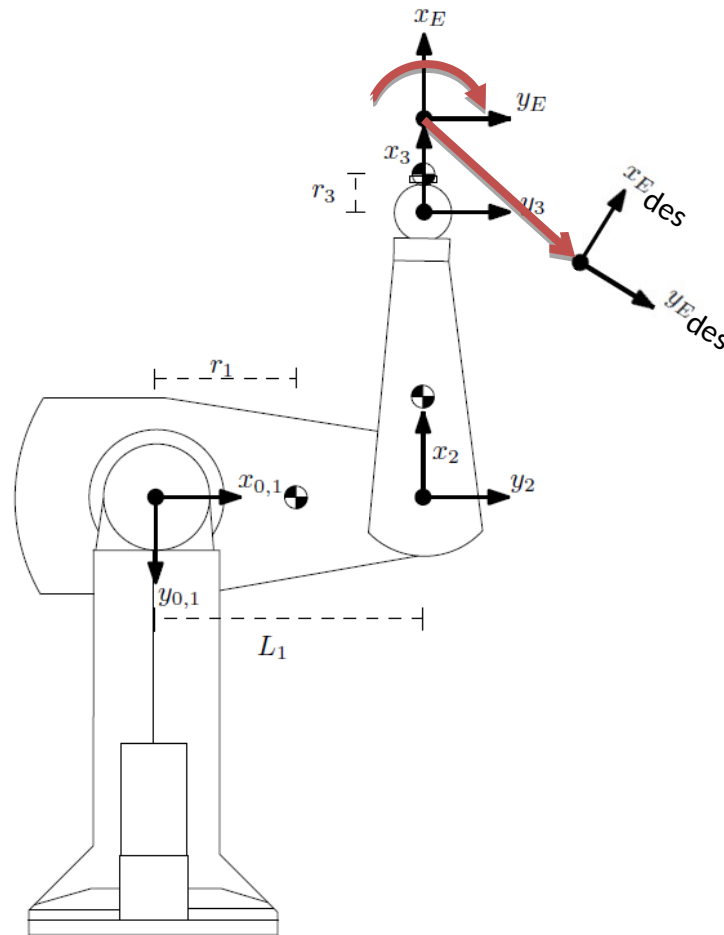
$$V(\mathbf{x}) = \frac{1}{2}k_p e^2$$

$$\mathbf{F} = -\nabla V(\mathbf{x}) = -\frac{\partial V}{\partial \mathbf{x}}$$

First Operational Space Control

$$\tau = \mathbf{k} J^T \mathbf{F} + G(\mathbf{q})$$

Trajectory Generation





Robotics

More on Jacobians

TU Berlin

Oliver Brock

Different Jacobians

- So far we computed the Jacobian
 - wrt/ end-effector
 - for a given end-effector **representation** \mathbf{x}
- Jacobian can be computed for
 - any point
 - any representation
- We want a uniquely defined Jacobian!

Basic Jacobian

$$\begin{pmatrix} v_x \\ v_y \\ v_z \\ \omega_x \\ \omega_y \\ \omega_z \end{pmatrix} = \begin{pmatrix} \mathbf{v}_{(3 \times 1)} \\ \boldsymbol{\omega}_{(3 \times 1)} \end{pmatrix} = J_{0(6 \times n)}(\mathbf{q}) \dot{\mathbf{q}}_{(n \times 1)}$$

Sometimes we will drop the (\mathbf{q})

$$\dot{\mathbf{x}} = \begin{pmatrix} \dot{\mathbf{x}}_p \\ \dot{\mathbf{x}}_r \end{pmatrix} = \begin{bmatrix} E_p(\mathbf{x}_p) & 0 \\ 0 & E_r(\mathbf{x}_r) \end{bmatrix} \begin{pmatrix} \mathbf{v} \\ \boldsymbol{\omega} \end{pmatrix}$$

$$E_p(\mathbf{x}_p) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad E_r(\mathbf{x}_r) = \begin{bmatrix} -\frac{s\alpha c\beta}{s\beta} & \frac{c\alpha c\beta}{s\beta} & 1 \\ c\alpha & s\alpha & 0 \\ \frac{s\alpha}{s\beta} & -\frac{c\alpha}{s\beta} & 0 \end{bmatrix}$$

e.g., to convert from $\boldsymbol{\omega}$ to α, β, γ Euler

Relationship between J_x and J_0

$$J_0(6 \times n) = \begin{bmatrix} J_v(3 \times n) \\ J_\omega(3 \times n) \end{bmatrix}$$

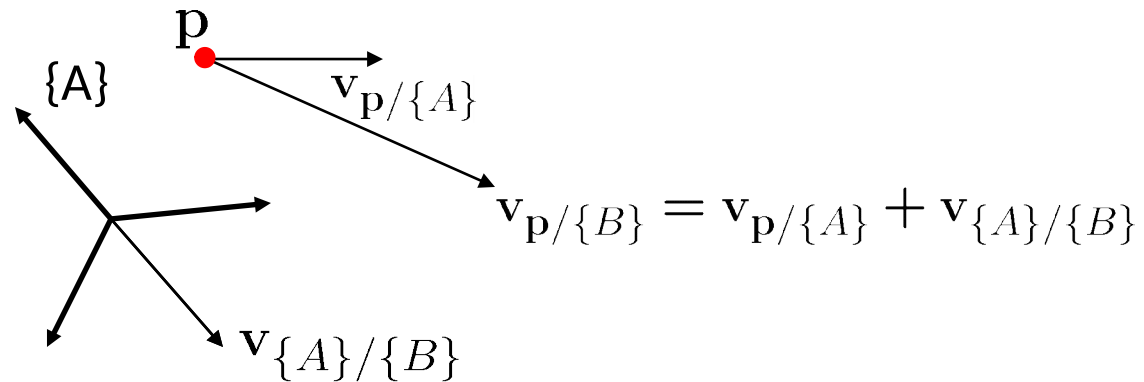
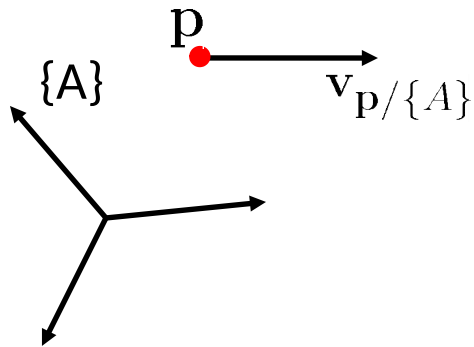
$$\dot{\mathbf{x}} = \begin{pmatrix} \dot{\mathbf{x}}_p \\ \dot{\mathbf{x}}_r \end{pmatrix} = \begin{pmatrix} E_p(\mathbf{x}_p) \mathbf{v} \\ E_r(\mathbf{x}_r) \boldsymbol{\omega} \end{pmatrix} = \begin{pmatrix} E_p(\mathbf{x}_p) J_v \dot{\mathbf{q}} \\ E_r(\mathbf{x}_r) J_\omega \dot{\mathbf{q}} \end{pmatrix}$$

$$J_x = \begin{bmatrix} J_p \\ J_r \end{bmatrix} = \begin{bmatrix} E_p & 0 \\ 0 & E_r \end{bmatrix} \begin{bmatrix} J_v \\ J_\omega \end{bmatrix}$$

Next: Motion of articulated bodies

- Linear motion = translation
- Angular motion = rotation
- Linear and angular motion
- Columns of Jacobian and joint motion
- Explicit form of computing Jacobian

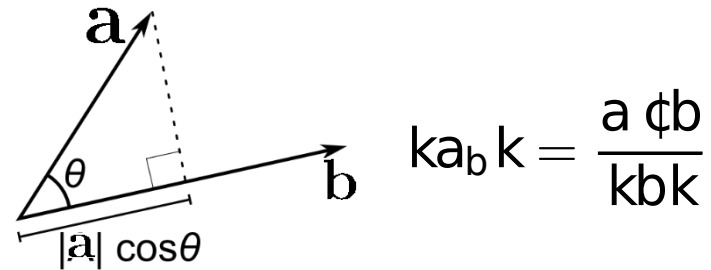
Linear Motion



Sidebar: dot / cross product

dot product

$$\mathbf{a} \cdot \mathbf{b} = \|\mathbf{a}\| \|\mathbf{b}\| \cos \phi$$

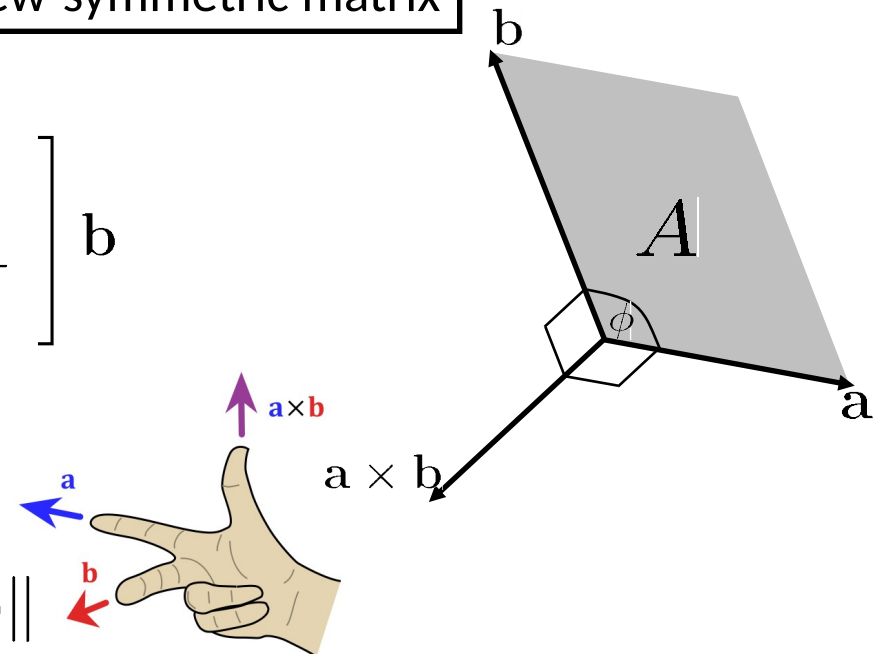


cross product

$$\mathbf{a} \times \mathbf{b} = \hat{\mathbf{a}} \mathbf{b} = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix} \mathbf{b}$$

cross product operator

skew-symmetric matrix



$$A = \|\mathbf{a}\| \|\mathbf{b}\| \sin \phi = \|\mathbf{a} \times \mathbf{b}\|$$



Rotational Motion

Angular Velocity

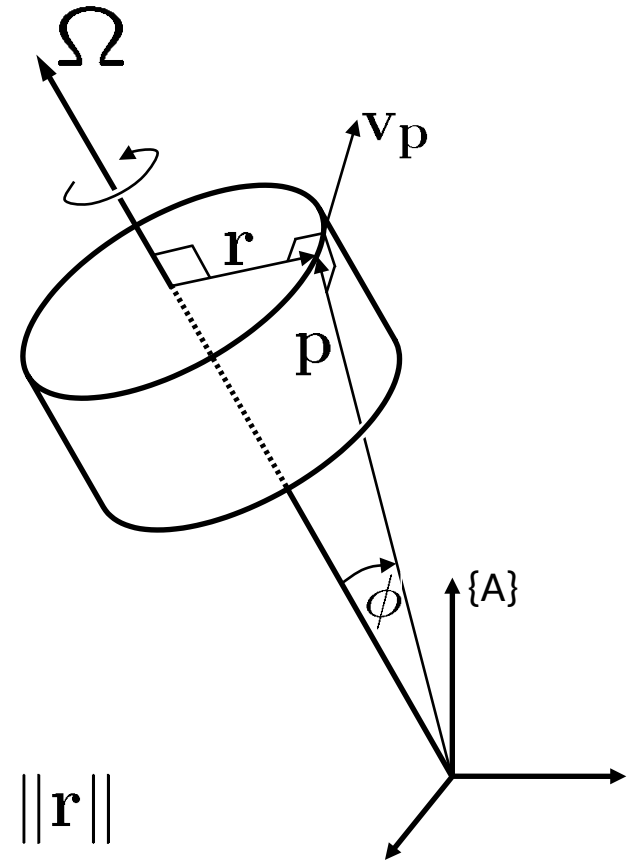
$$\mathbf{v}_p = \boldsymbol{\Omega} \times \mathbf{p}$$

$$\|\mathbf{v}_p\| = \|\boldsymbol{\Omega}\| \|\mathbf{p}\| \sin \phi = \|\boldsymbol{\Omega}\| \|\mathbf{r}\|$$

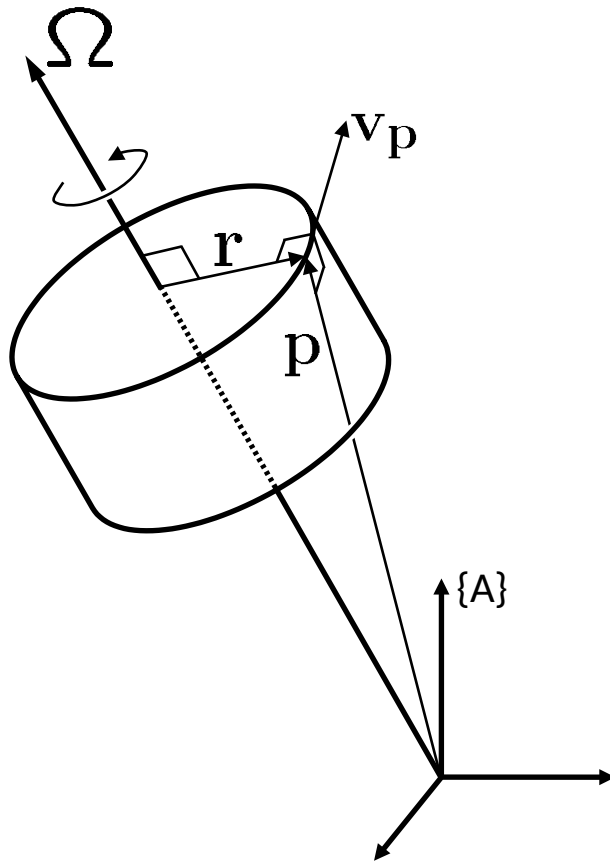
$$\|\boldsymbol{\Omega}\| = \frac{\|\mathbf{v}_p\|}{\|\mathbf{r}\|}$$

$$\boldsymbol{\Omega} = \frac{\mathbf{r} \times \mathbf{v}_p}{\|\mathbf{r}\|^2}$$

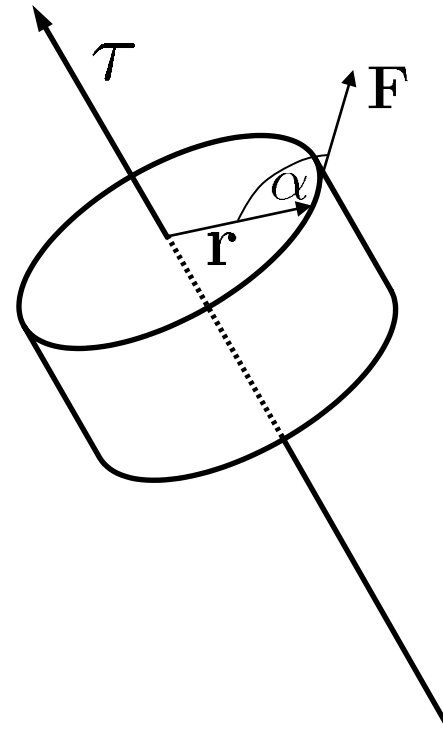
$$\|\mathbf{v}_p\| = \|\boldsymbol{\Omega}\| \|\mathbf{r}\|$$



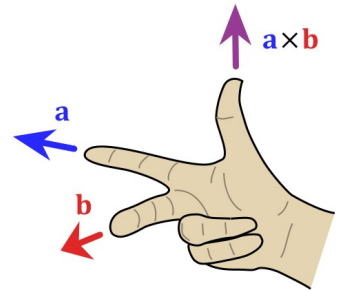
Side by Side



$$\mathbf{v}_p = \Omega \times \mathbf{p}$$



$$\tau = \mathbf{r} \times \mathbf{F}$$

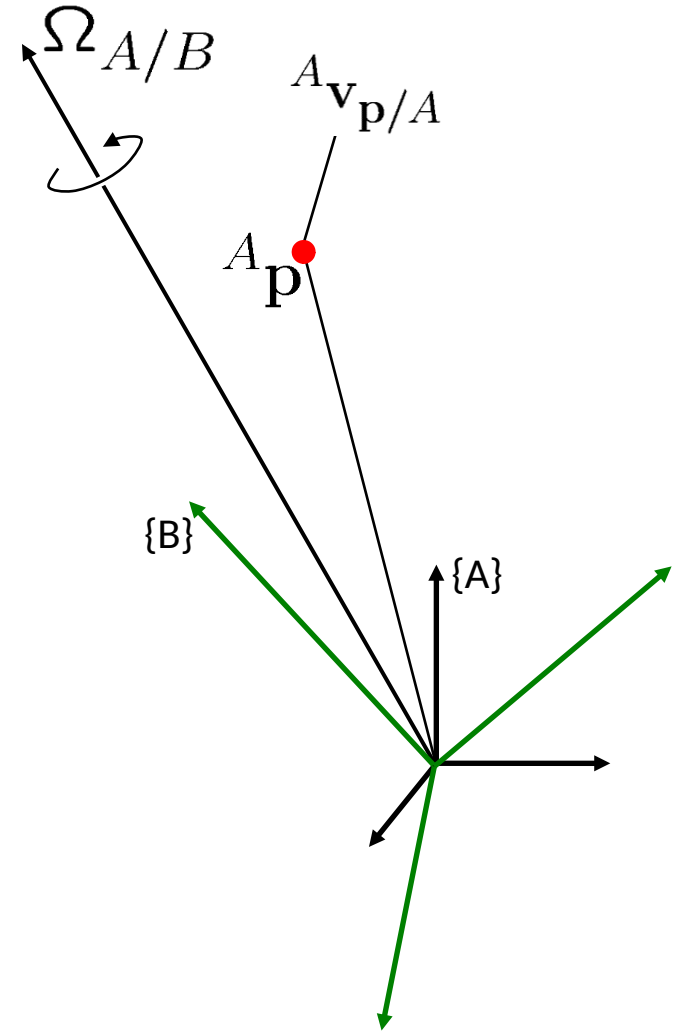


Velocity Between Rotating Frames

Considering rotation between {A} and {B}:

$${}^A\mathbf{v}_p = {}^A\boldsymbol{\Omega}_{A/B} \times {}^A\mathbf{p}$$

$${}^B\mathbf{v}_p = {}^B_A R {}^A\boldsymbol{\Omega}_{A/B} \times {}^B\mathbf{p}$$



Linear and Angular Motion

From before:
$${}^B\mathbf{v}_p = \underbrace{{}^B R^A \Omega_{A/B}}_{{}^B \Omega_{A/B}} \times {}^B \mathbf{p}$$

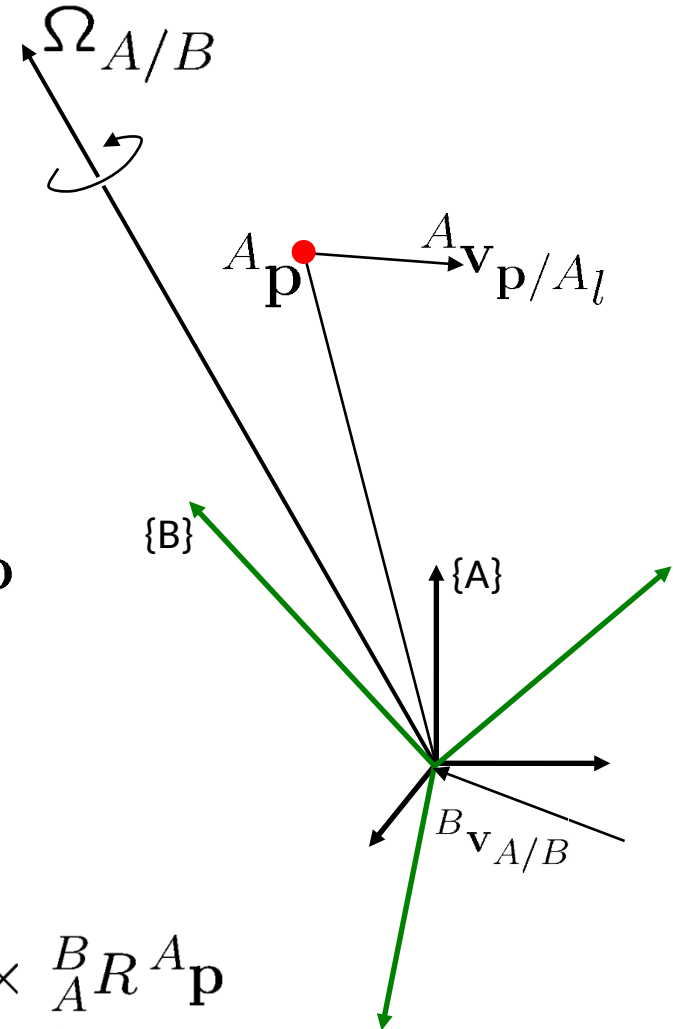
Adding linear motion of \mathbf{p} (possibly the result of linear and angular motion in $\{A\}$):

$${}^B\mathbf{v}_p = {}^B({}^A\mathbf{v}_{p/A_l}) + {}^B\Omega_{A/B} \times {}^B\mathbf{p}$$

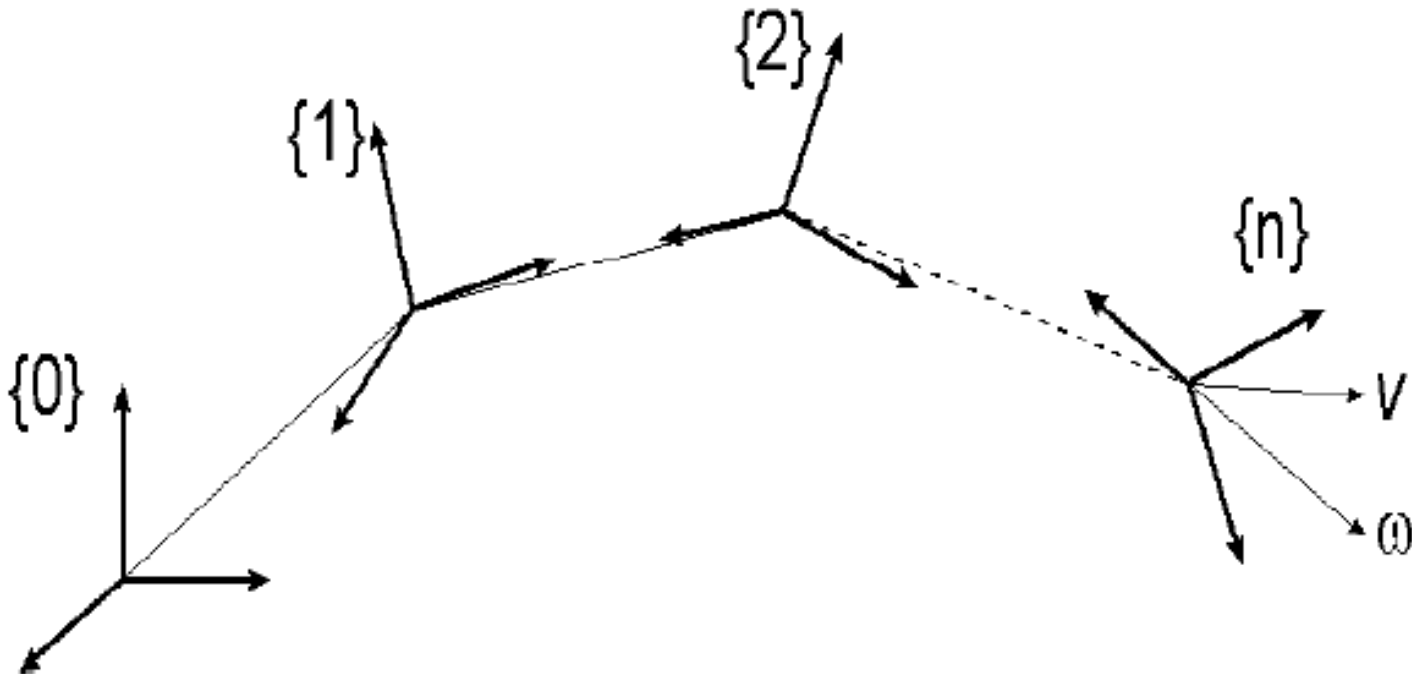
$${}^B\mathbf{v}_p = {}^B R^A \mathbf{v}_{p/A_l} + {}^B\Omega_{A/B} \times {}^B R^A \mathbf{p}$$

Adding linear motion between frames A and B:

$${}^B\mathbf{v}_p = {}^B\mathbf{v}_{A/B} + {}^B R^A \mathbf{v}_{p/A_l} + {}^B\Omega_{A/B} \times {}^B R^A \mathbf{p}$$



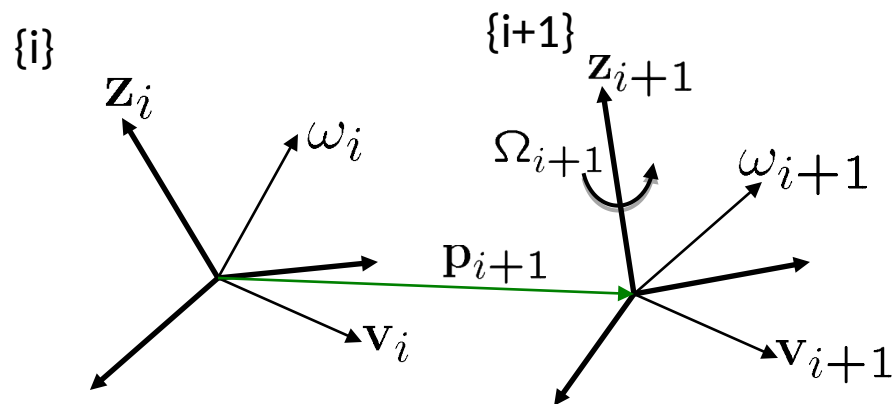
Velocity Propagation



Velocity Propagation

$$\omega_{i+1} = \omega_i + \Omega_{i+1}$$

$$\Omega_{i+1} = \dot{\theta}_{i+1} \mathbf{z}_{i+1}$$



$$\mathbf{v}_{i+1} = \mathbf{v}_i + \omega_i \times \mathbf{p}_{i+1} + \dot{d}_{i+1} \mathbf{z}_{i+1}$$

$$\begin{pmatrix} {}^0\mathbf{v} \\ {}^0\omega \end{pmatrix} = \begin{bmatrix} {}^0_n R & 0 \\ 0 & {}^0_n R \end{bmatrix} \begin{pmatrix} {}^n\mathbf{v} \\ {}^n\omega \end{pmatrix}$$

Huh?

- Velocity propagation gives us end-effector velocities based on joint velocities
- That is exactly what the Jacobian does!
- Velocity propagation only involved simple operations – no differentiation of the forward kinematics!
- Can we use velocity propagation to easily compute the Jacobian?

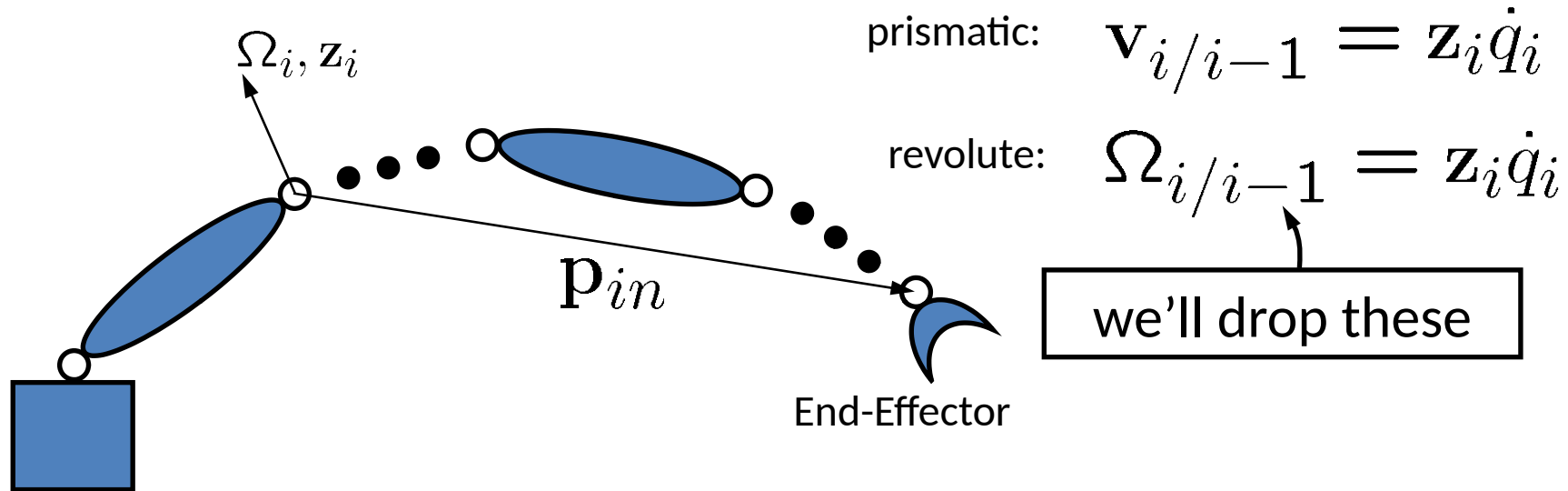
q revisited: Joint Coordinates

$$q_i = \bar{\epsilon}\theta_i + \epsilon d_i$$

$$\epsilon = \begin{cases} 0 & \text{for revolute joints} \\ 1 & \text{for prismatic joints} \end{cases}$$

$$\bar{\epsilon} = 1 - \epsilon$$

End-Effector Velocities




$$\mathbf{v} = \sum_{i=1}^n [\epsilon_i \mathbf{v}_i + \bar{\epsilon}_i (\Omega_i \times \mathbf{p}_{in})]$$

$$\omega = \sum_{i=1}^n \bar{\epsilon}_i \mathbf{z}_i \dot{q}_i$$

Linear End-Effector Velocities

$$\begin{aligned}
 \dot{\mathbf{x}}_p = \mathbf{v} &= \sum_{i=1}^n [\epsilon_i \mathbf{v}_i + \bar{\epsilon}_i (\Omega_i \times \mathbf{p}_{in})] \\
 &= (\epsilon_1 \mathbf{z}_1 + \bar{\epsilon}_1 (\mathbf{z}_1 \times \mathbf{p}_{1n})) \dot{q}_1 + \\
 &\quad (\epsilon_2 \mathbf{z}_2 + \bar{\epsilon}_2 (\mathbf{z}_2 \times \mathbf{p}_{2n})) \dot{q}_2 + \\
 &\quad \vdots \\
 &\quad (\epsilon_n \mathbf{z}_n + \bar{\epsilon}_n (\mathbf{z}_n \times \mathbf{p}_{nn})) \dot{q}_n \\
 &= J_v \dot{\mathbf{q}}
 \end{aligned}$$

i -th column of J_v



$$\mathbf{v}_{i/i-1} = \mathbf{z}_i \dot{q}_i$$

$$\Omega_{i/i-1} = \mathbf{z}_i \dot{q}_i$$

Angular End-Effector Velocities

$$\begin{aligned}\dot{\mathbf{x}}_r &= \boldsymbol{\omega} = \sum_{i=1}^n \bar{\mathbf{e}}_i \mathbf{z}_i \dot{q}_i \\ &= [\bar{\mathbf{e}}_1 \mathbf{z}_1] \dot{q}_1 + \\ &\quad [\bar{\mathbf{e}}_2 \mathbf{z}_2] \dot{q}_2 + \\ &\quad \vdots \\ &\quad [\bar{\mathbf{e}}_n \mathbf{z}_n] \dot{q}_n \\ &= J_\omega \dot{\mathbf{q}}\end{aligned}$$

i -th column of J_ω



Jacobian Computation

$$J = \begin{bmatrix} \frac{\partial \mathbf{x}_p}{\partial q_1} & \frac{\partial \mathbf{x}_p}{\partial q_2} & \cdots & \frac{\partial \mathbf{x}_p}{\partial q_n} \\ \bar{\epsilon}_1 \mathbf{z}_1 & \bar{\epsilon}_2 \mathbf{z}_2 & \cdots & \bar{\epsilon}_n \mathbf{z}_n \end{bmatrix}$$

$${}^0J = \begin{bmatrix} \frac{\partial {}^0\mathbf{x}_p}{\partial q_1} & \frac{\partial {}^0\mathbf{x}_p}{\partial q_2} & \cdots & \frac{\partial {}^0\mathbf{x}_p}{\partial q_n} \\ \bar{\epsilon}_1 {}^0\mathbf{z}_1 & \bar{\epsilon}_2 {}^0\mathbf{z}_2 & \cdots & \bar{\epsilon}_n {}^0\mathbf{z}_n \end{bmatrix}$$

$$= \begin{bmatrix} \frac{\partial {}^0\mathbf{x}_p}{\partial q_1} & \frac{\partial {}^0\mathbf{x}_p}{\partial q_2} & \cdots & \frac{\partial {}^0\mathbf{x}_p}{\partial q_n} \\ \bar{\epsilon}_1 ({}^0_1R\mathbf{z}) & \bar{\epsilon}_2 ({}^0_2R\mathbf{z}) & \cdots & \bar{\epsilon}_n ({}^0_nR\mathbf{z}) \end{bmatrix}$$

Jacobian Computation

$$J = \begin{bmatrix} (\epsilon_1 \mathbf{z}_1 + \bar{\epsilon}_1 \mathbf{z}_1 \times \mathbf{p}_{1n}) & \cdots & (\epsilon_{n-1} \mathbf{z}_{n-1} + \bar{\epsilon}_{n-1} \mathbf{z}_{n-1} \times \mathbf{p}_{(n-1)n}) & \epsilon_n \mathbf{z}_n \\ \bar{\epsilon}_1 \mathbf{z}_1 & \cdots & \bar{\epsilon}_{n-1} \mathbf{z}_{n-1} & \bar{\epsilon}_n \mathbf{z}_n \end{bmatrix}$$

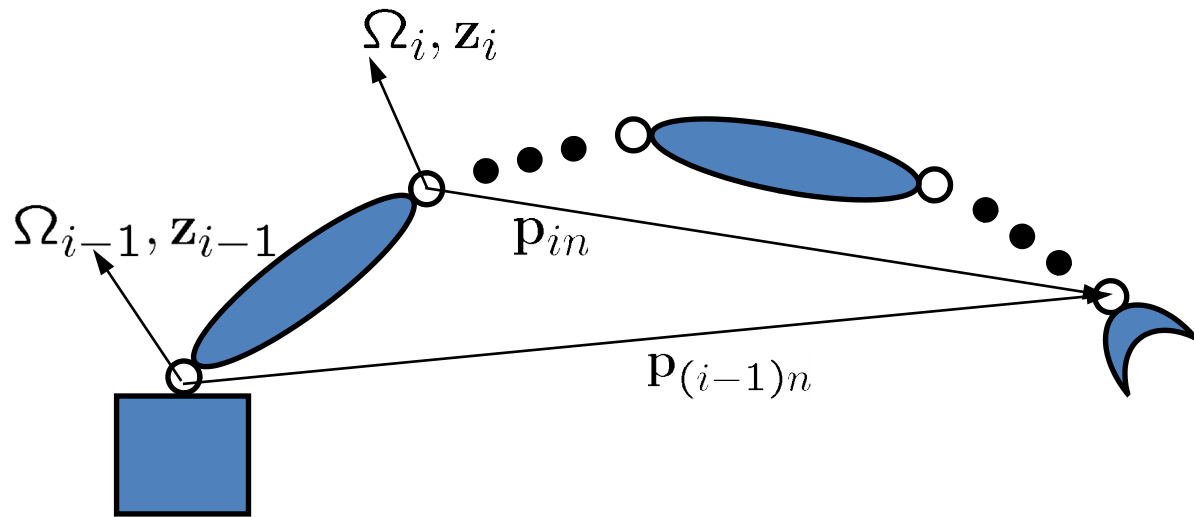
$${}^0J = \begin{bmatrix} {}^0_1R (\epsilon_1 \mathbf{z}_1 + \bar{\epsilon}_1 \mathbf{z}_1 \times \mathbf{p}_{1n}) & \cdots & {}^0_{n-1}R (\epsilon_{n-1} \mathbf{z}_{n-1} + \bar{\epsilon}_{n-1} \mathbf{z}_{n-1} \times \mathbf{p}_{(n-1)n}) & {}^0_nR \epsilon_n \mathbf{z}_n \\ {}^0_1R \bar{\epsilon}_1 \mathbf{z}_1 & \cdots & {}^0_{n-1}R \bar{\epsilon}_{n-1} \mathbf{z}_{n-1} & {}^0_nR \bar{\epsilon}_n \mathbf{z}_n \end{bmatrix}$$

Example: Jacobian for n revolute joint robot

$${}^0J_{allrev} = \begin{bmatrix} ({}^0\mathbf{z}_1 \times {}^0\mathbf{p}_{1n}) & ({}^0\mathbf{z}_2 \times {}^0\mathbf{p}_{2n}) & \cdots & {}^0\mathbf{z}_n \\ {}^0\mathbf{z}_1 & {}^0\mathbf{z}_2 & \cdots & {}^0\mathbf{z}_n \end{bmatrix}$$

All quantities are known from forward kinematics!!!

A Word about Reference Frames



Jacobian in Different Frames

$${}^n J = \begin{bmatrix} \frac{\partial \mathbf{x}_p}{\partial q_1} & \frac{\partial \mathbf{x}_p}{\partial q_2} & \cdots & \frac{\partial \mathbf{x}_p}{\partial q_n} \\ \bar{\epsilon}_1 \mathbf{z}_1 & \bar{\epsilon}_2 \mathbf{z}_2 & \cdots & \bar{\epsilon}_n \mathbf{z}_n \end{bmatrix}$$

$${}^0 J = \begin{bmatrix} \frac{\partial {}^0 \mathbf{x}_p}{\partial q_1} & \frac{\partial {}^0 \mathbf{x}_p}{\partial q_2} & \cdots & \frac{\partial {}^0 \mathbf{x}_p}{\partial q_n} \\ \bar{\epsilon}_1 {}^0 \mathbf{z}_1 & \bar{\epsilon}_2 {}^0 \mathbf{z}_2 & \cdots & \bar{\epsilon}_n {}^0 \mathbf{z}_n \end{bmatrix}$$

So far...

- Mathematically motivated we saw the Jacobian for
 - infinitesimal displacements
 - velocities
 - forces
- Some more Jacobian definitions
- Physical understanding

Moving J Between Frames

$${}^B J = \begin{bmatrix} {}^B_A R & 0 \\ 0 & {}^B_A R \end{bmatrix} {}^A J$$