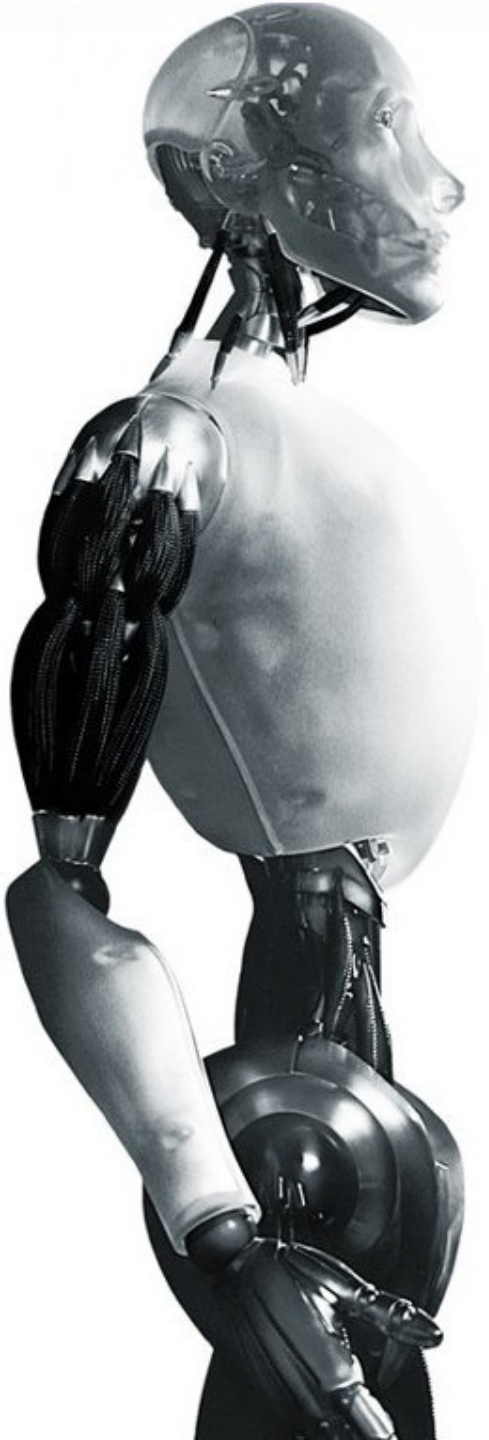


Disclaimer

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

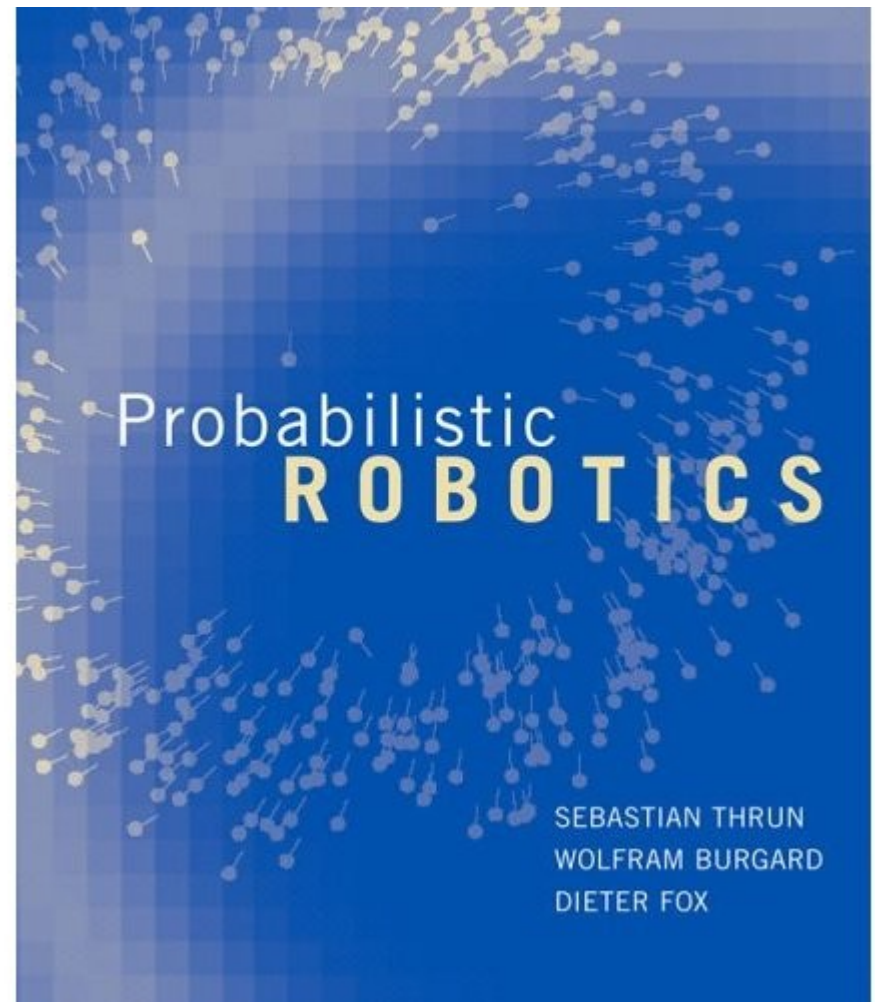
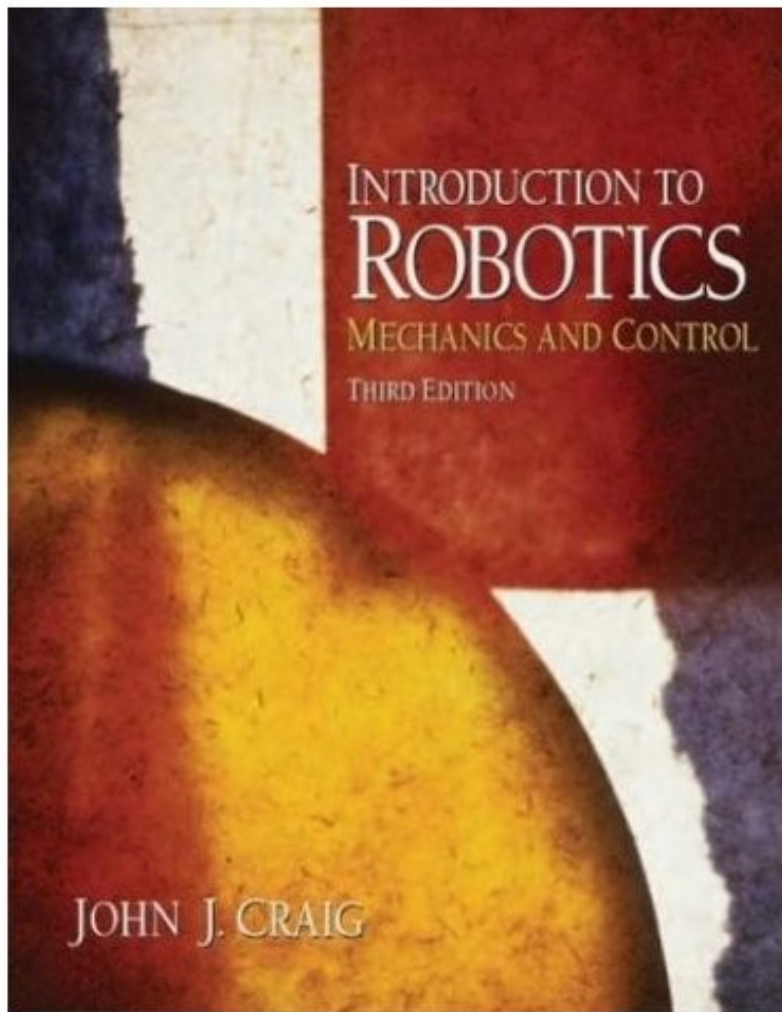
Our First Task: Hold Still!

TU Berlin
Oliver Brock

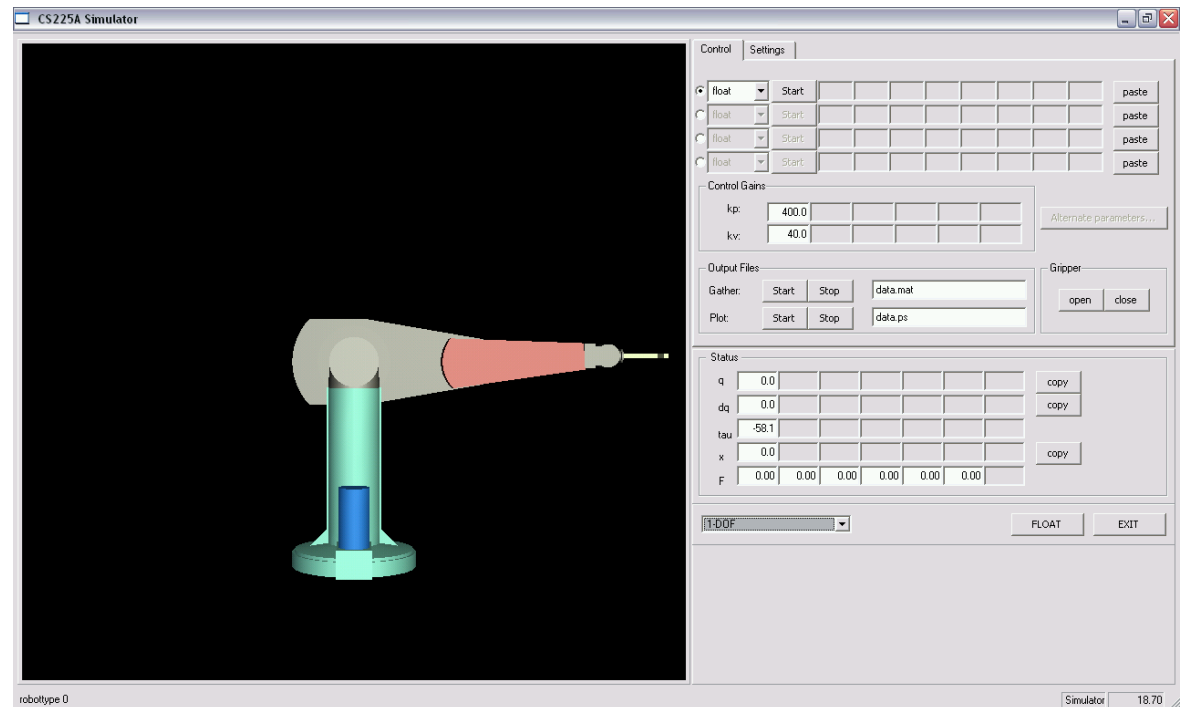
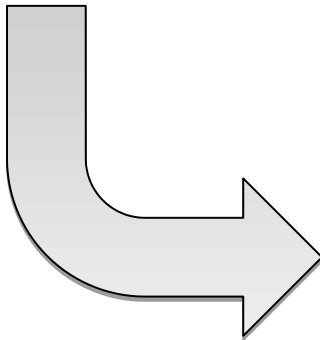
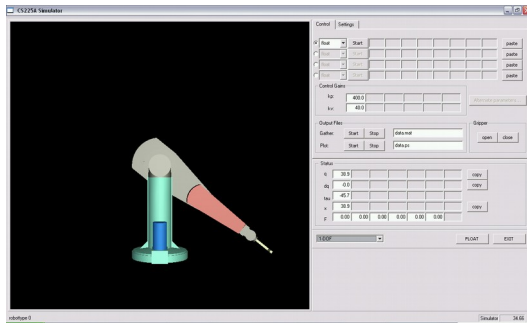
Reading for this set of slides

- Craig – Intro to Robotics (3rd Edition)
 - 1 Introduction
 - 2 Spatial descriptions and transformations (2.1 – 2.9)
 - 3 Manipulator kinematics (3.1 – 3.6)

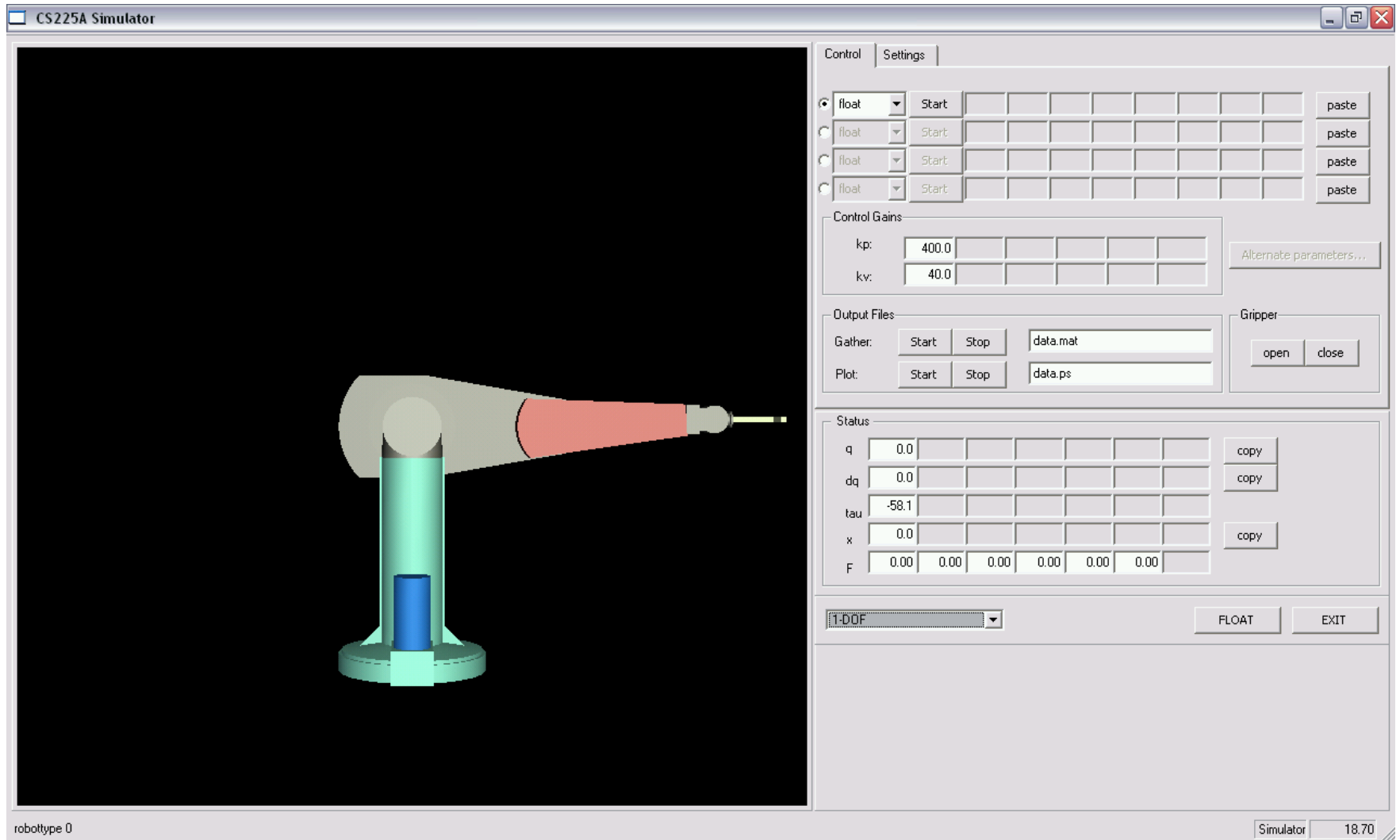
Please note that this set of slides is intended as support for the lecture, not as a stand-alone script. If you want to study for this course, please use these slides in conjunction with the indicated chapters in the text books. The textbooks are available online or in the TUB library (many copies that 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.



Our Task: Hold Still!

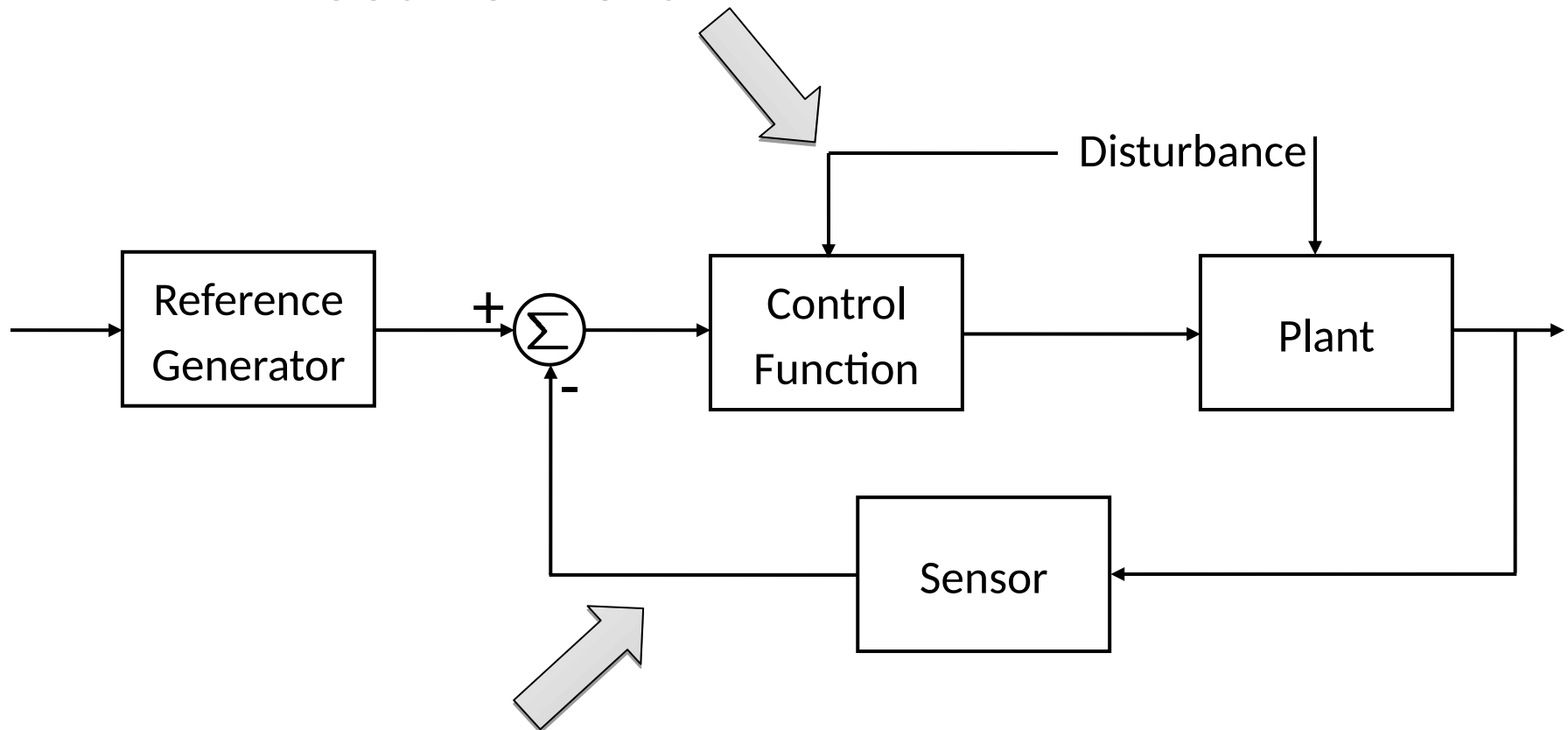


Let's try...



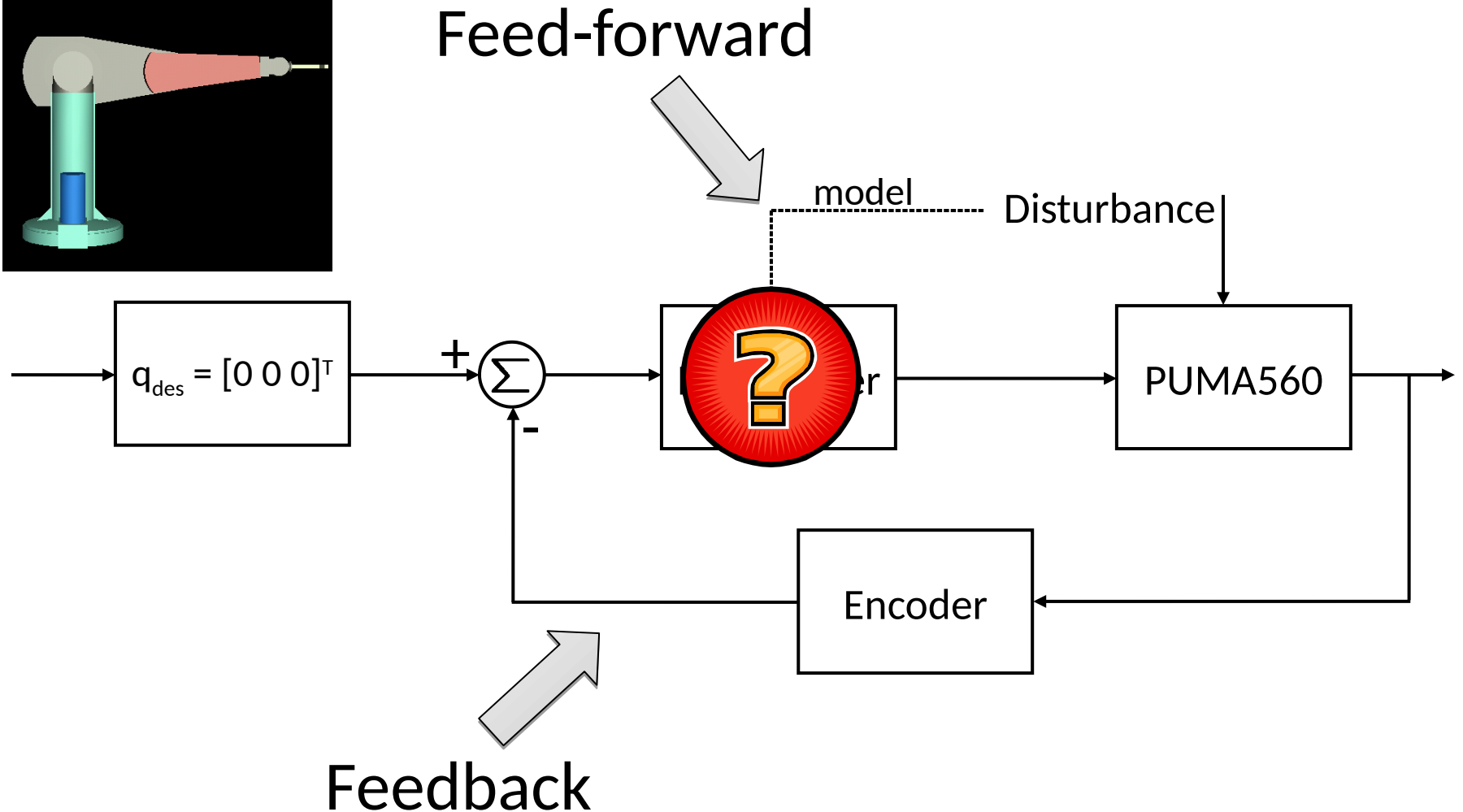
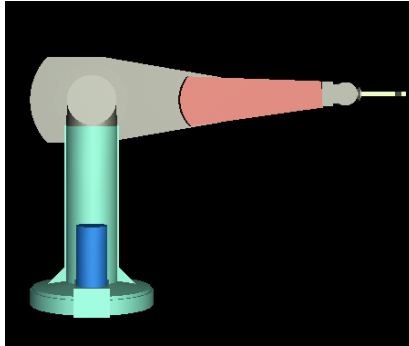
A Simple Controller [Regler]

Feed-forward

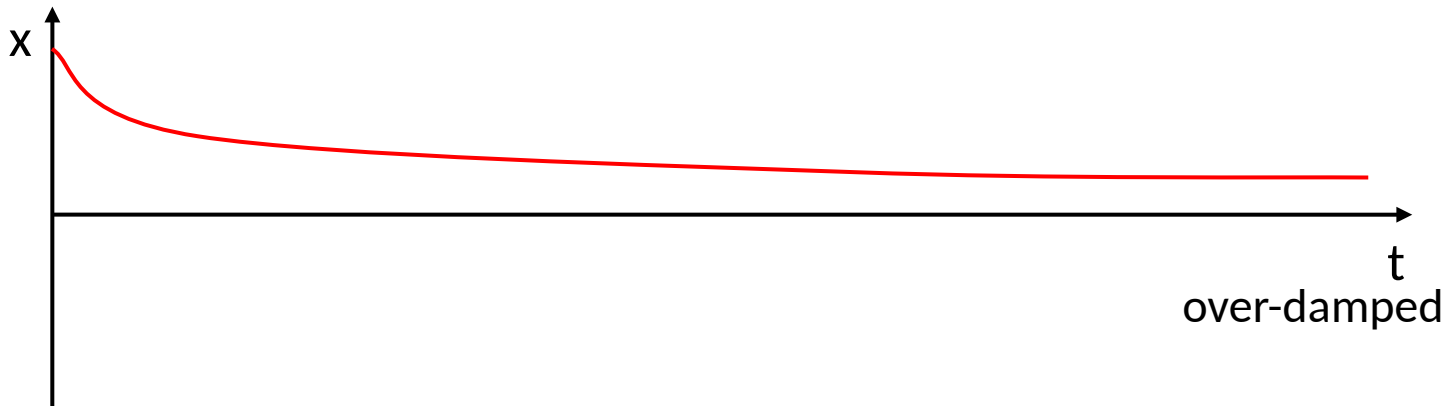
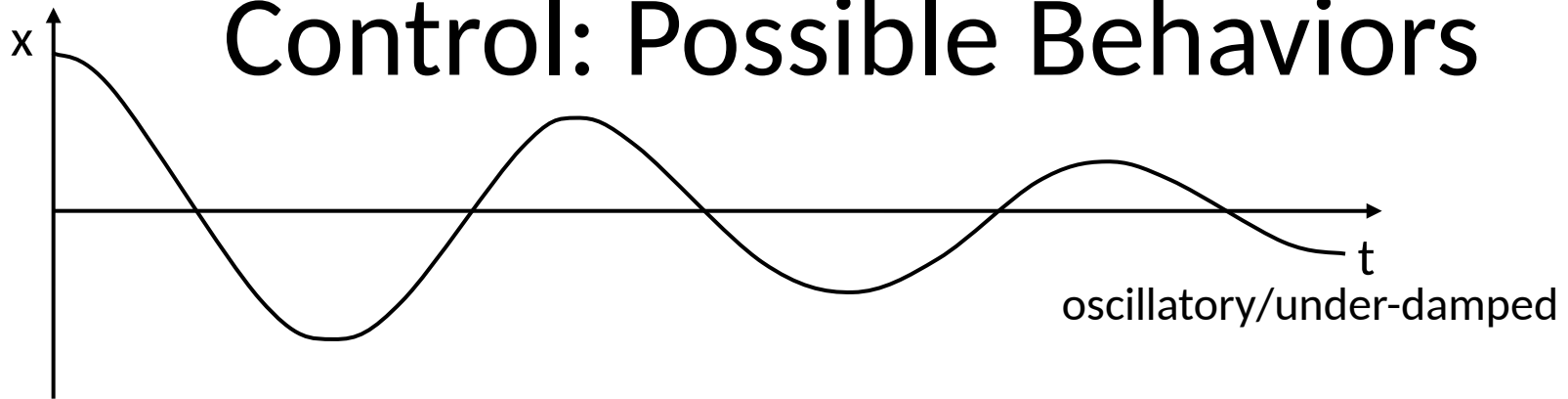


Feedback (we used proportional feedback or P control)
(also: closed loop – as opposed to open loop)

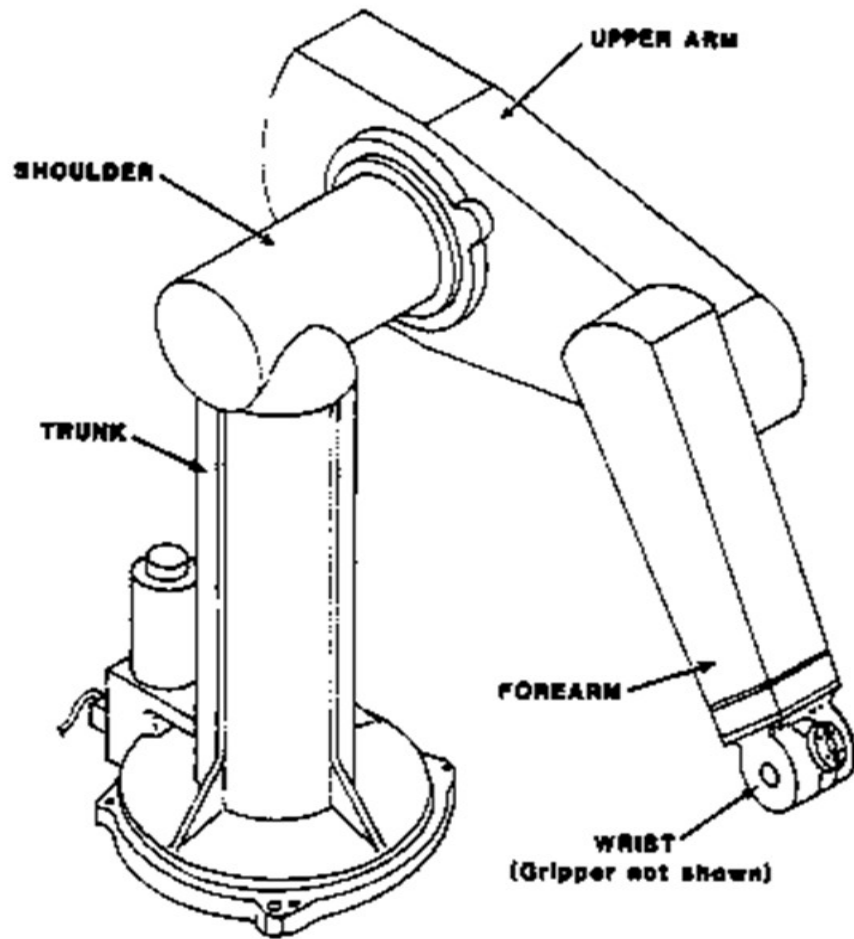
A Proportional Controller



Control: Possible Behaviors

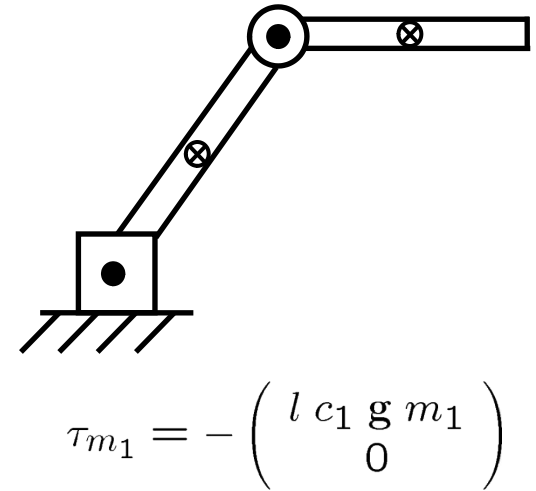


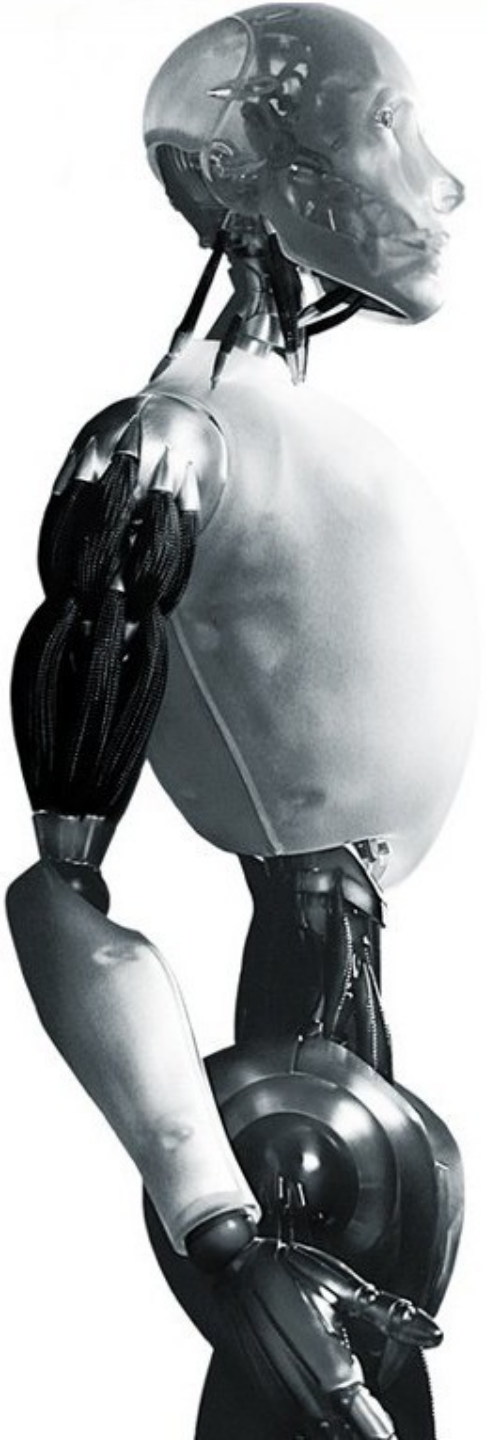
Possible Disturbances



What we need to solve the problem

- Fixed parameters of the robot
 - Kinematic
 - Dynamic
- Changing parameters
- Then we can:
 - Compensate gravity (feed-forward)
 - Reject error (feedback, P-controller)
- But: what happens when the robot moves?



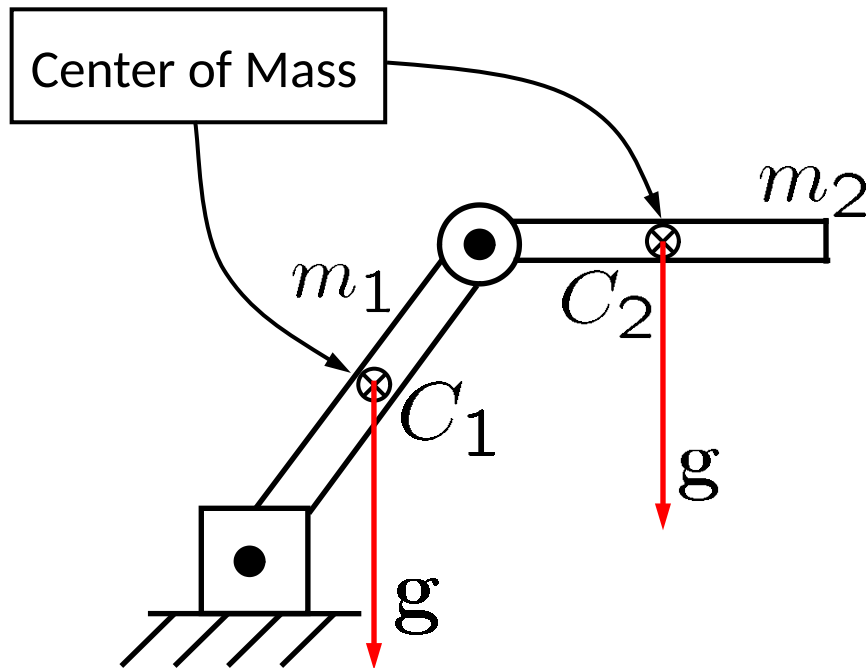


Robotics

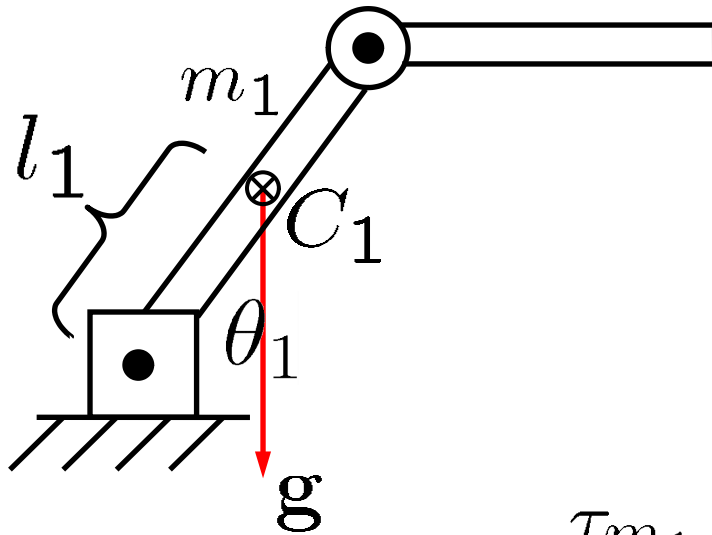
First Steps in Kinematics, Gravity Compensation

TU Berlin
Oliver Brock

Gravity

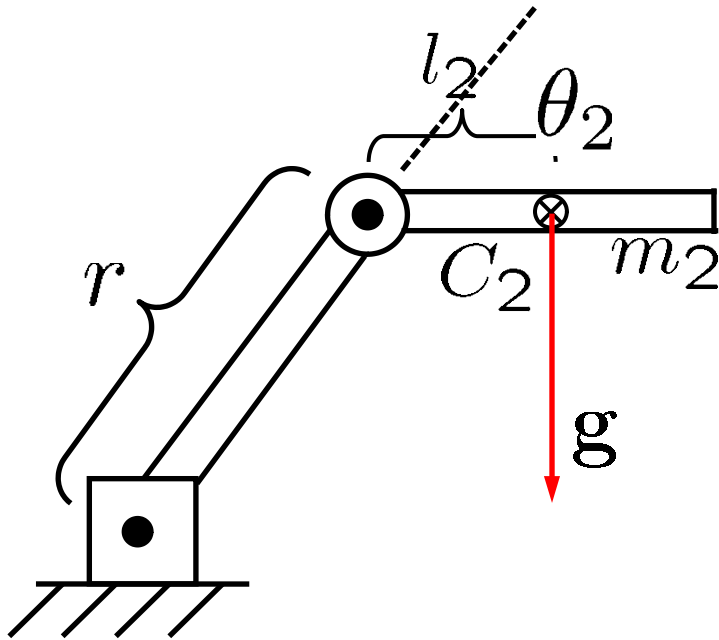


Gravity: Link 1



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

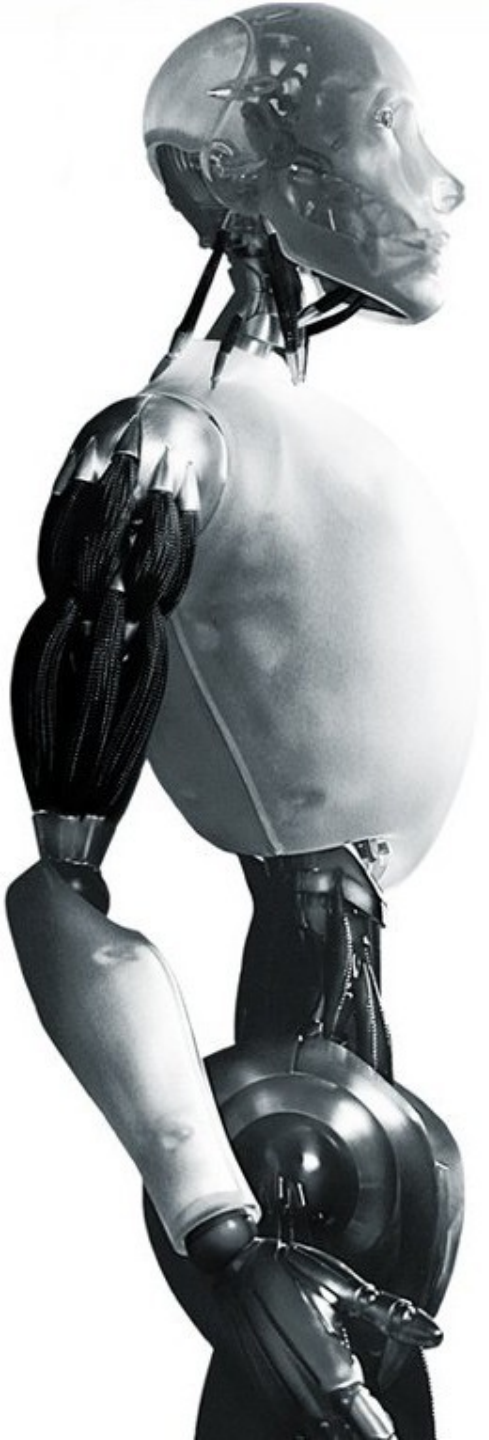
Gravity: Link 2



$$c_{12} = \cos(\theta_1 + \theta_2)$$

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

$$\tau_g = \tau_{m_1} + \tau_{m_2} = - \begin{pmatrix} (r c_1 + l_2 c_{12}) m_2 + l_1 c_1 m_1 \\ l_2 c_{12} m_2 \end{pmatrix} g$$



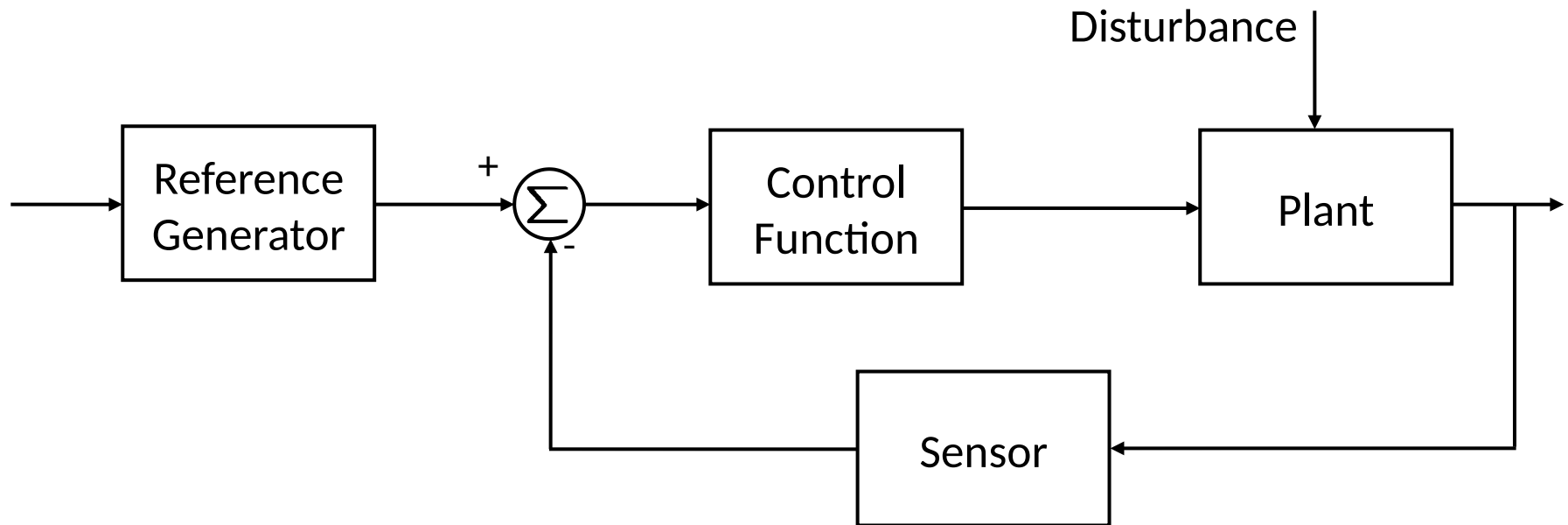
Robotics

First Steps in Control

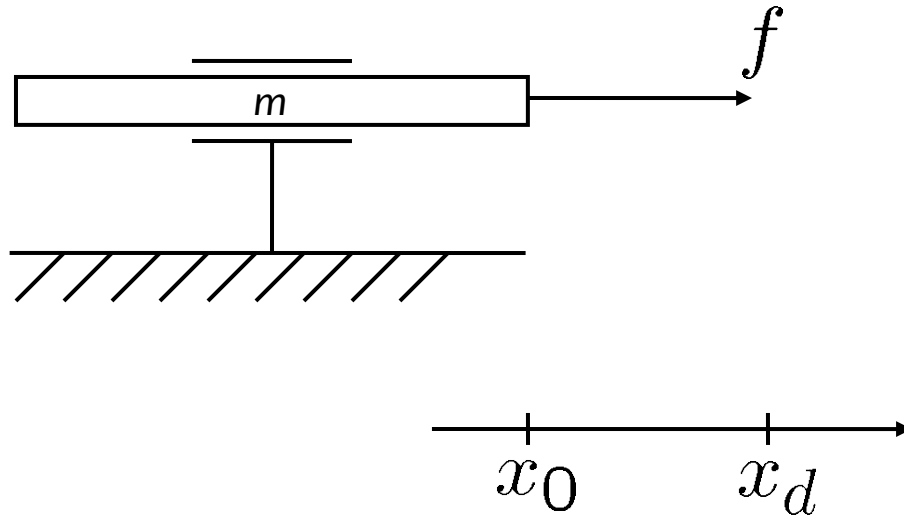
TU Berlin
Oliver Brock

Control

- **Control** is the process of causing a *system variable* to conform to some desired value, called a *reference value*.



Simplest Case: Prismatic Joint

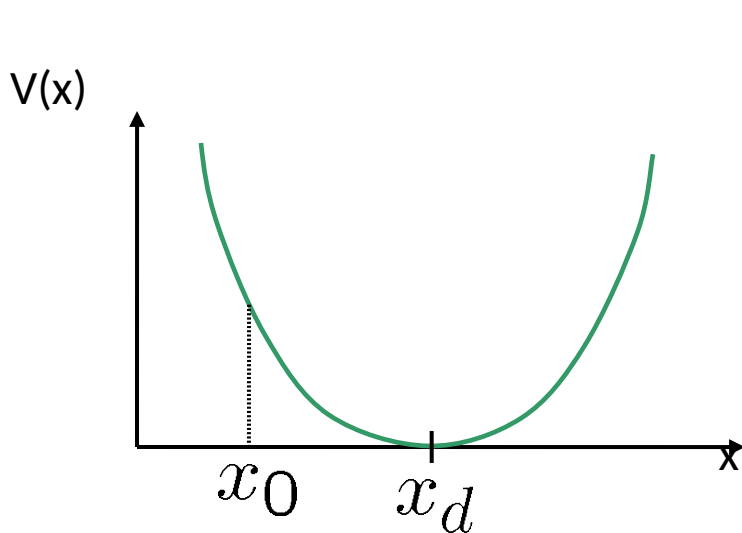


Proportional Control

Idea: apply force proportional to error

$$f = -\boxed{k_p}(x - x_d)$$

position gain



$$V(x) = \frac{1}{2}k_p(x - x_d)^2$$

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

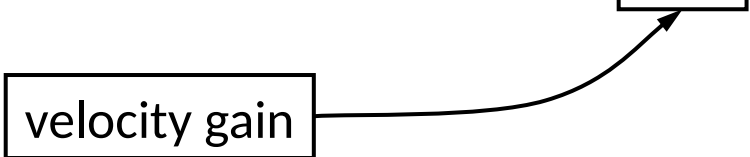
Introduction of Dissipation

Idea: apply force opposing velocity

this leads to dissipation (or dampening)

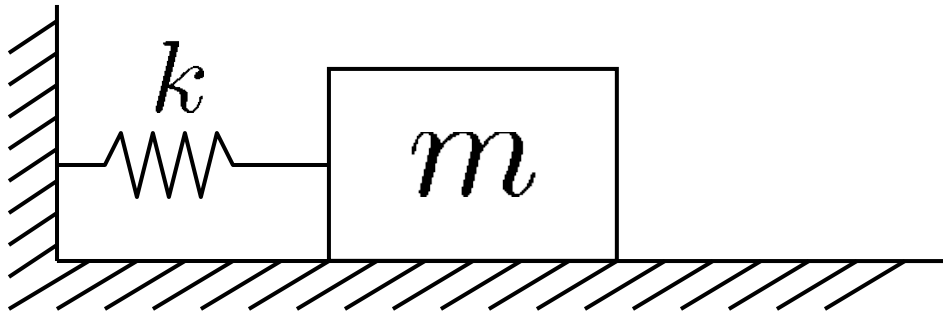
$$f = -k_p(x - x_d) - \boxed{k_v} \dot{x}$$

velocity gain

A diagram consisting of a rectangular box containing the text "velocity gain". An arrow originates from the right side of this box and curves upwards and to the right, pointing towards the boxed term k_v in the equation $f = -k_p(x - x_d) - \boxed{k_v} \dot{x}$.

Physics will give us answers!

Conservative System

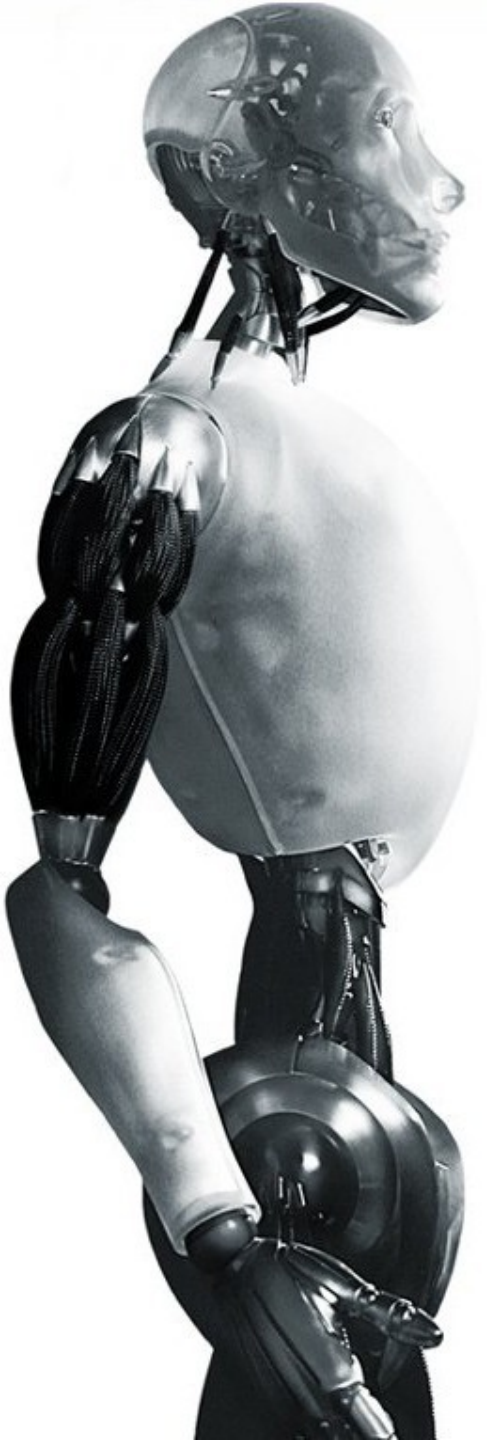


$$K = \frac{1}{2} m \dot{x}^2$$

$$f = -k x$$

$$V = \frac{1}{2} k x^2$$

$$m\ddot{x} + kx = 0$$



Robotics

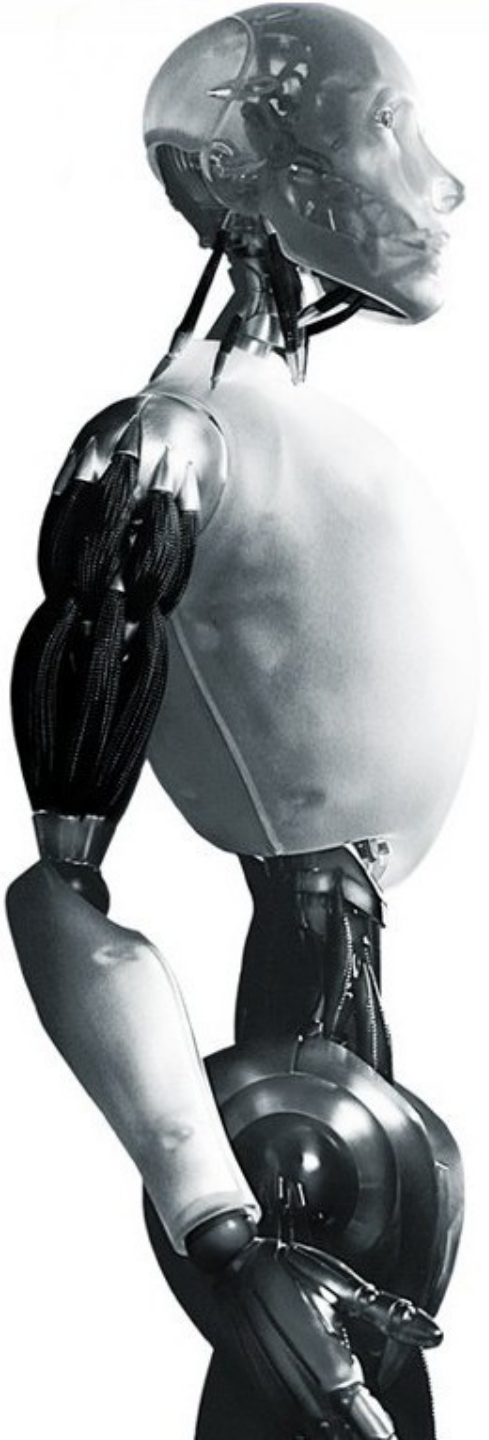
Task 2: Track Circle with End-Effector

TU Berlin
Oliver Brock



What do we need now?

1. A way to specify a trajectory (task) for the end-effector
2. A control law to move the end-effector along the specified trajectory
3. But first: a way to determine the joint angles, given a desired pose of the end-effector – or: *a way to determine how to move the joints so that the error is reduced!*
4. To improve performance of the controller: dynamic compensation (feed-forward model of the robot's dynamics)

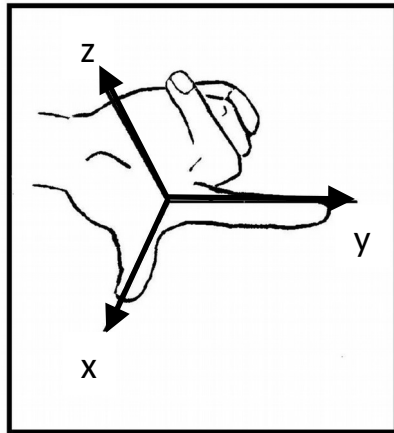


Robotics

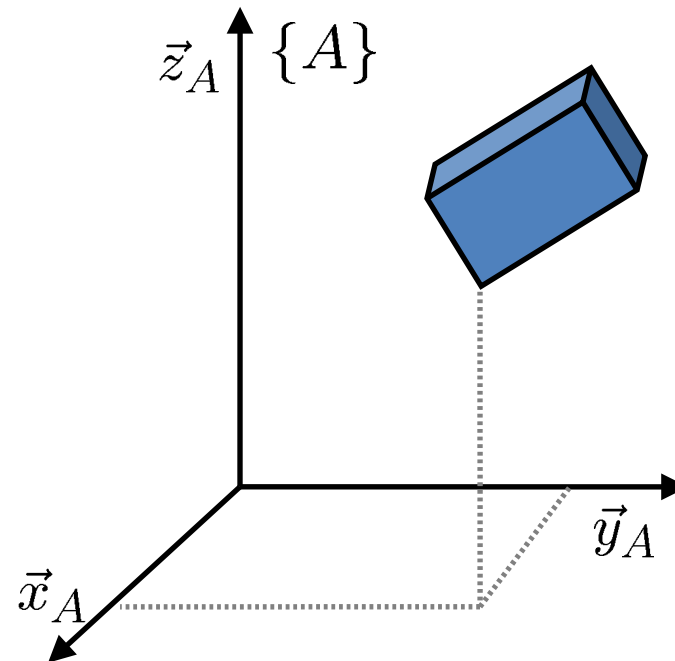
Where is the End-Effector?
Homogeneous Transforms, Spatial Representations

TU Berlin
Oliver Brock

Object Position and Orientation



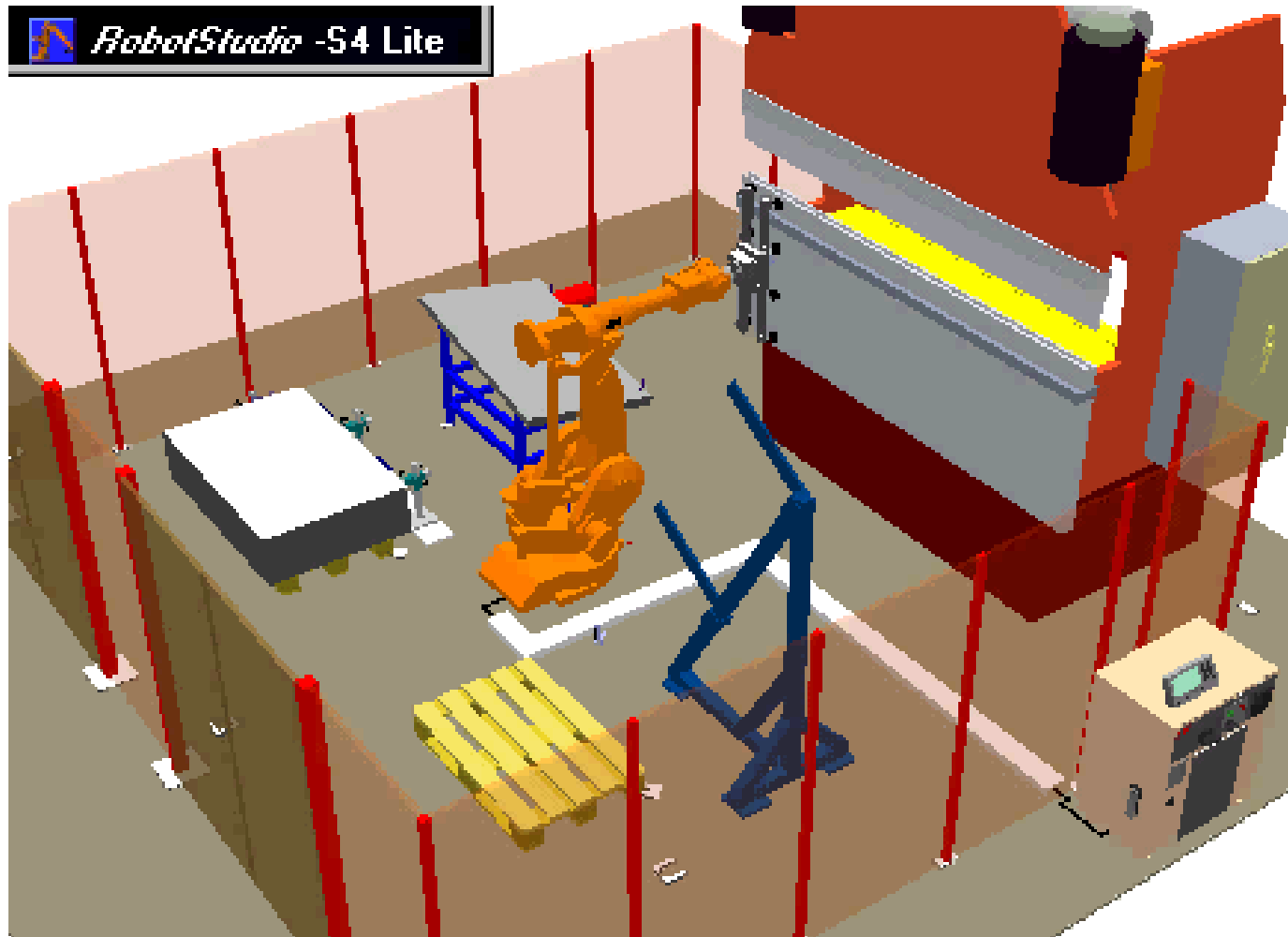
right-handed coordinate system



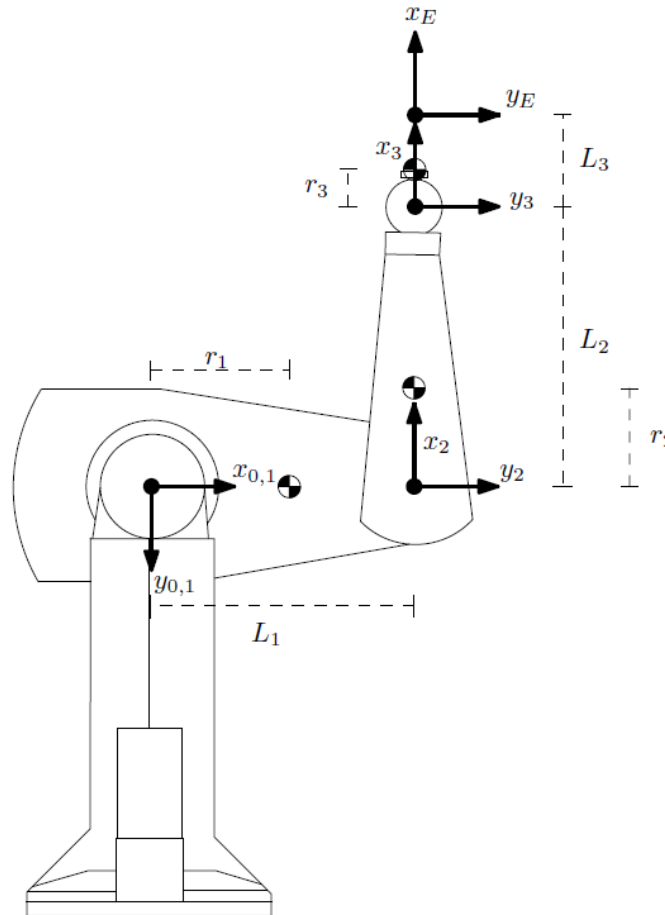
Kinematics

- Webster
 - Kinematics: branch of dynamics that deals with aspects of motion apart from considerations of mass and force
 - Dynamics: branch of mechanics that deals with forces and their relation primarily to the motion but sometimes also to the equilibrium of bodies
 - Mechanics: branch of physical science that deals with energy and forces and their effect on bodies
- Here: relationship between joint motion and motion of rigid bodies (links) without regard for the forces that cause it

Where is the end-effector?



Forward Kinematics



$$\mathbf{x} = f(\mathbf{q}) = {}^0_E T(\mathbf{q}) = {}^0_1 T(q_1) {}^1_2 T(q_2) {}^2_3 T(q_3) {}^3_E T$$

Configuration / Degrees of Freedom

- A configuration \mathbf{q} is a minimal set of parameters required to uniquely specify the position and orientation (pose) of every point on the robot.

$$\mathbf{q} = \begin{pmatrix} q_1 \\ q_2 \\ \vdots \\ q_n \end{pmatrix} \quad q_i = \begin{cases} \theta_i & \text{if dof } i \text{ is revolute} \\ d_i & \text{if dof } i \text{ is prismatic} \end{cases}$$

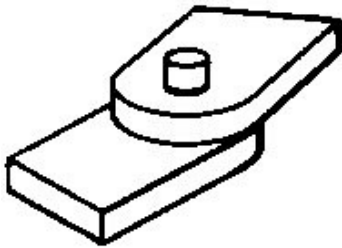
End-effector Pose in Operational Space

- A pose \mathbf{x} is a (not necessarily minimal) set of parameters required to uniquely specify the position and orientation (pose) of the robot's end-effector. The choice of \mathbf{x} depends on the task.

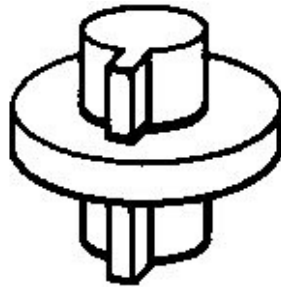
Example:

$$\mathbf{x} = \begin{pmatrix} x \\ y \\ z \\ \alpha \\ \beta \\ \gamma \end{pmatrix}$$

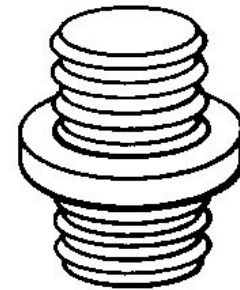
Types of Joints



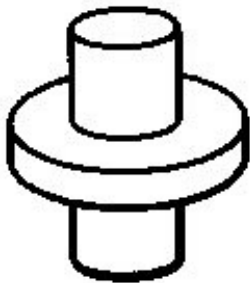
Revolute: 1 dof



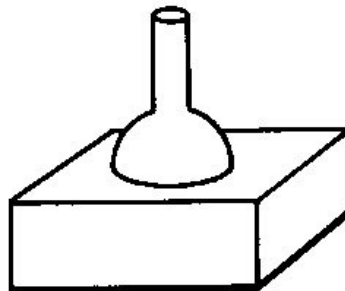
Prismatic: 1 dof



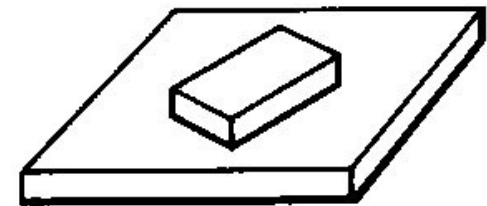
Screw: 1 dof



Cylindrical: 2 dof



Spherical: 3 dof

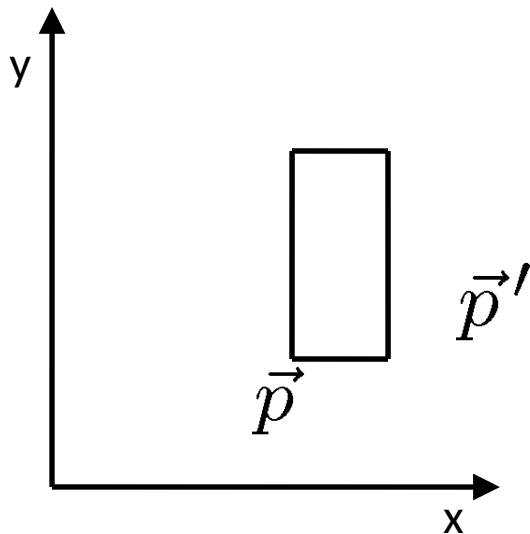


Planar: 3 dof

Translation

$$\vec{p} = \begin{pmatrix} p_x \\ p_y \end{pmatrix}$$

$$\vec{t} = \begin{pmatrix} t_x \\ t_y \end{pmatrix}$$



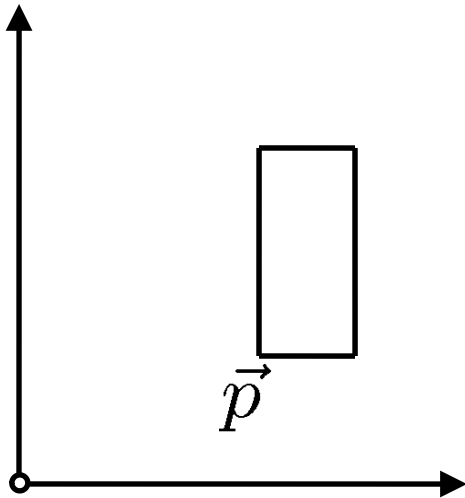
$$\vec{p}' = \vec{p} + \vec{t} = \begin{pmatrix} p_x + t_x \\ p_y + t_y \end{pmatrix}$$

Global Reference Coordinate System = World Frame

Rotation



$$\vec{p} = \begin{pmatrix} p_x \\ p_y \end{pmatrix}$$



$$\vec{p}' = ?$$

Deriving the Rotation Matrix

$$p_x = r \cdot \cos \phi$$

$$p_y = r \cdot \sin \phi$$

$$p'_x = r \cdot \cos(\theta + \phi)$$

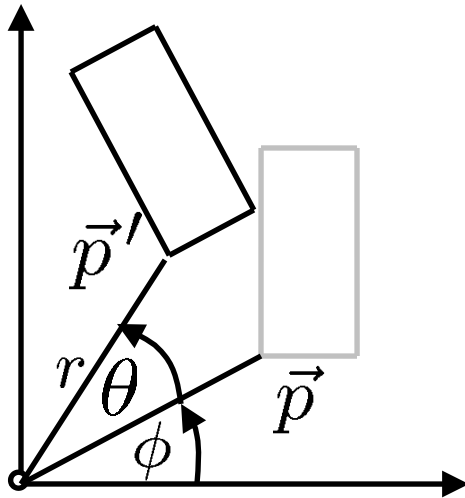
$$= r \cdot \cos \phi \cdot \cos \theta - r \cdot \sin \phi \cdot \sin \theta$$

$$= p_x \cdot \cos \theta - p_y \cdot \sin \theta$$

$$p'_y = r \cdot \sin(\theta + \phi)$$

$$= r \cdot \cos \phi \cdot \sin \theta + r \cdot \sin \phi \cdot \cos \theta$$

$$= p_x \sin \theta + p_y \cos \theta$$



$$\vec{p}' = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \cdot \vec{p}$$

$$\vec{p}' = R(\theta) \cdot \vec{p}$$

What you should know...

$$\cos(\theta_1 + \theta_2) = \cos \theta_1 \cos \theta_2 - \sin \theta_1 \sin \theta_2$$

$$\sin(\theta_1 + \theta_2) = \cos \theta_1 \sin \theta_2 + \sin \theta_1 \cos \theta_2$$

$$\cos(\theta_1 - \theta_2) = \cos \theta_1 \cos \theta_2 + \sin \theta_1 \sin \theta_2$$

$$\sin(\theta_1 - \theta_2) = \sin \theta_1 \cos \theta_2 - \cos \theta_1 \sin \theta_2$$

$$\sin \theta = -\sin(-\theta) = -\cos\left(\theta + \frac{\pi}{2}\right) = \cos\left(\theta - \frac{\pi}{2}\right)$$

$$\cos \theta = \cos(-\theta) = \sin\left(\theta + \frac{\pi}{2}\right) = -\sin\left(\theta - \frac{\pi}{2}\right)$$

$$\cos^2 \theta + \sin^2 \theta = 1$$

Properties of Rotation Matrices

- Orthogonal
- Normal
- Determinant = 1
- $R^T = R^{-1}$

These properties hold for n-dimensional matrices!

Homogeneous Transformations

$$\vec{p}' = \vec{p} + \vec{t}$$



$$\vec{p}' = R(\theta) \cdot \vec{p}$$



$$\vec{p}' = R(\theta) \cdot \vec{p} + \vec{t}$$



Rotation first!

$$\begin{pmatrix} p'_x \\ p'_y \\ 1 \end{pmatrix} = \begin{bmatrix} R(\theta) & t_x \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{pmatrix} p_x \\ p_y \\ 1 \end{pmatrix}$$

3D Homogeneous Transforms

$$\begin{pmatrix} p'_x \\ p'_y \\ p'_z \\ 1 \end{pmatrix} = \begin{bmatrix} R(\theta) & \begin{matrix} t_x \\ t_y \\ t_z \end{matrix} \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{pmatrix} p_x \\ p_y \\ p_z \\ 1 \end{pmatrix}$$

$$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \theta & -\sin \theta \\ 0 & \sin \theta & \cos \theta \end{bmatrix} \quad R_y(\theta) = \begin{bmatrix} \cos \theta & 0 & \sin \theta \\ 0 & 1 & 0 \\ -\sin \theta & 0 & \cos \theta \end{bmatrix}$$

$$R_z(\theta) = \begin{bmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

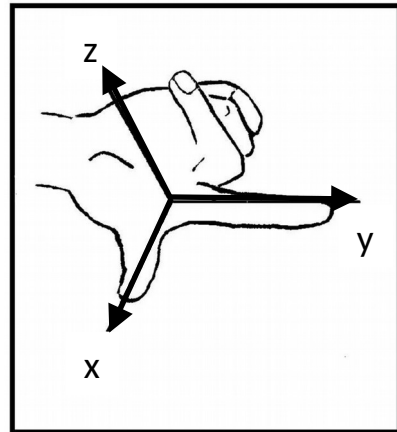
These Transformations:

- Scaling
- Sheering
- Rotation
- Translation
- and any combination of them

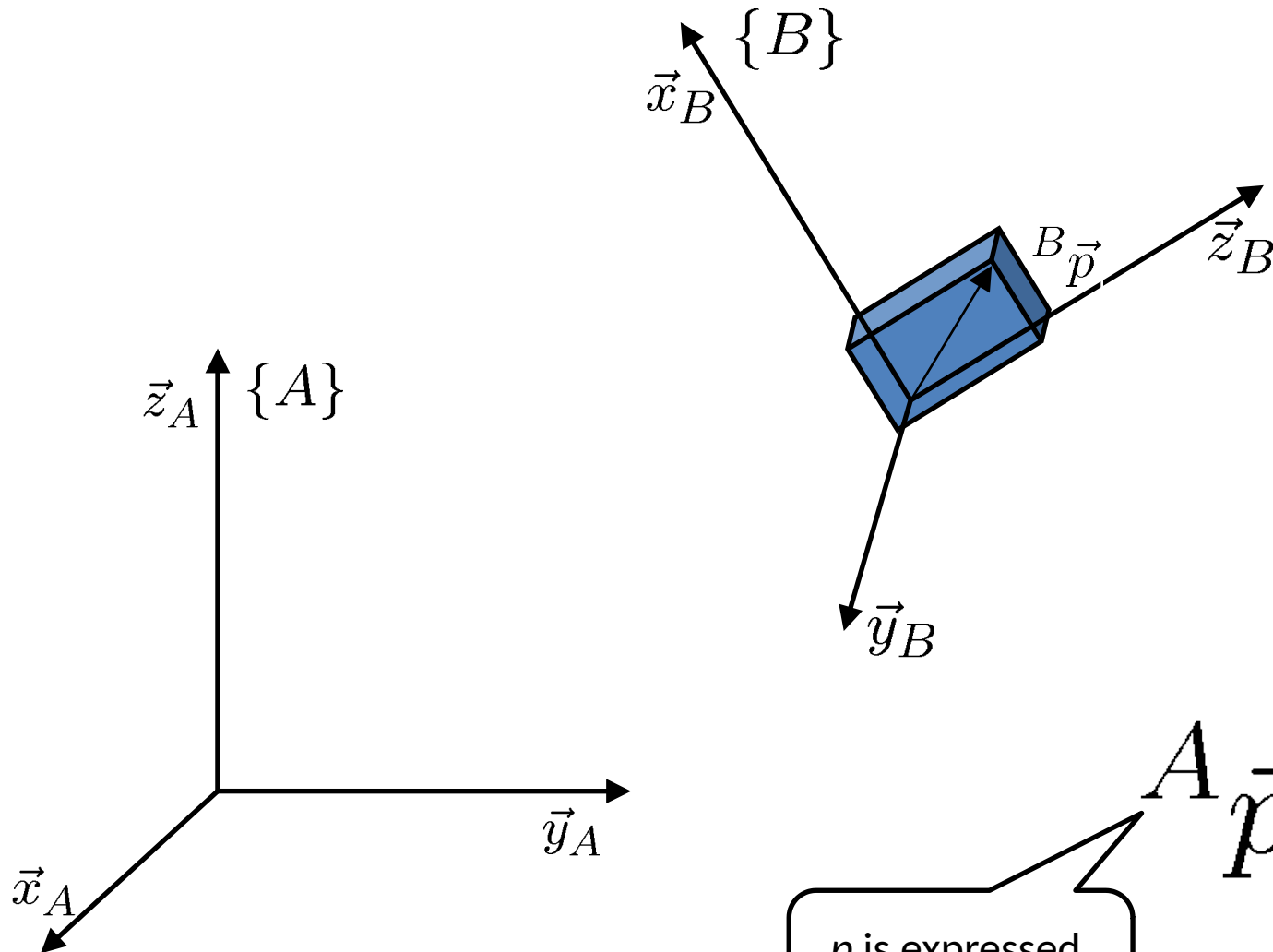
...are affine!

Affine transformations preserve collinearity, parallelism, and ratios of distances.

Frames



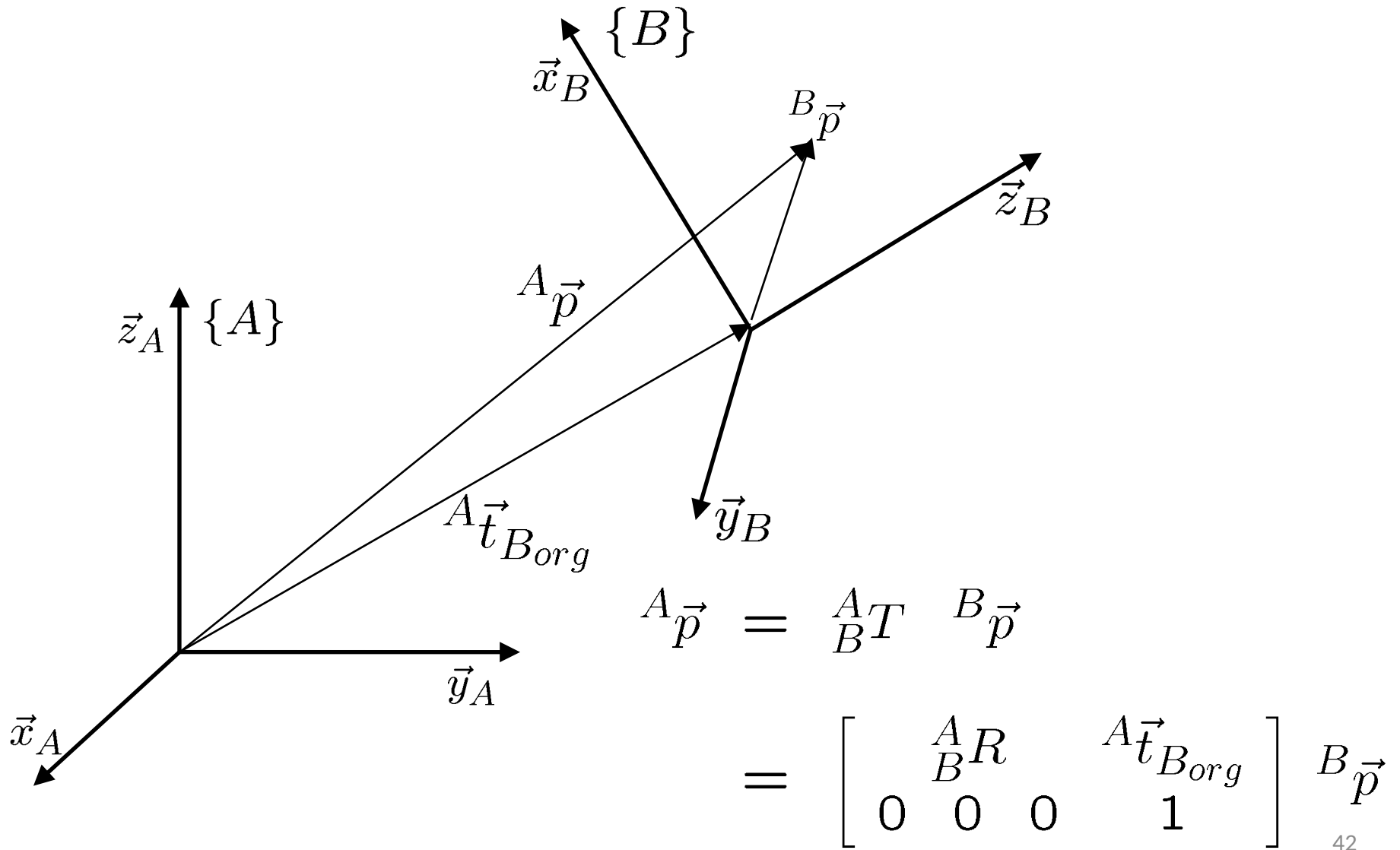
right-handed coordinate system



$${}^A\vec{p} = ?$$

p is expressed
in this frame

HT to Map between Frames

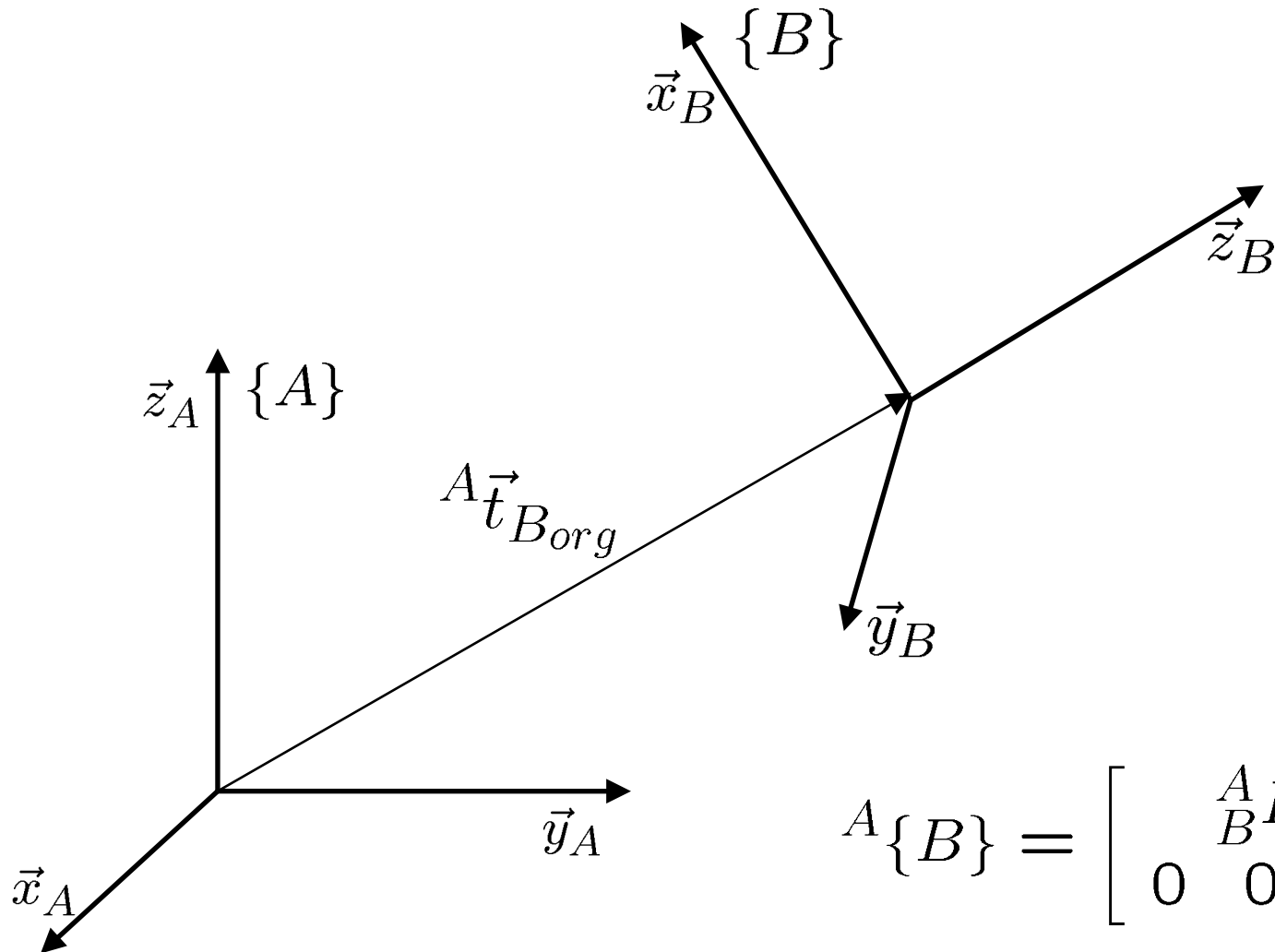


How to compute R?

$$\begin{aligned} {}^A_B R &= \begin{bmatrix} {}^A\vec{x}_B & {}^A\vec{y}_B & {}^A\vec{z}_B \end{bmatrix} \\ &= \begin{bmatrix} \vec{x}_B\vec{x}_A & \vec{y}_B\vec{x}_A & \vec{z}_B\vec{x}_A \\ \vec{x}_B\vec{y}_A & \vec{y}_B\vec{y}_A & \vec{z}_B\vec{y}_A \\ \vec{x}_B\vec{z}_A & \vec{y}_B\vec{z}_A & \vec{z}_B\vec{z}_A \end{bmatrix} \end{aligned}$$

Direction Cosines

HT as Frame Descriptions



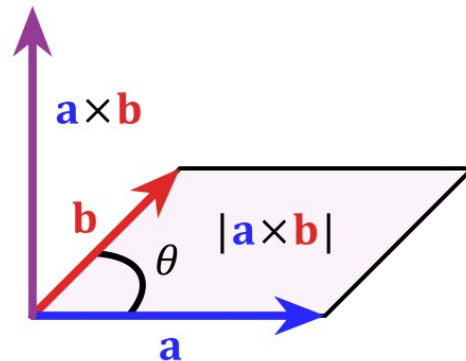
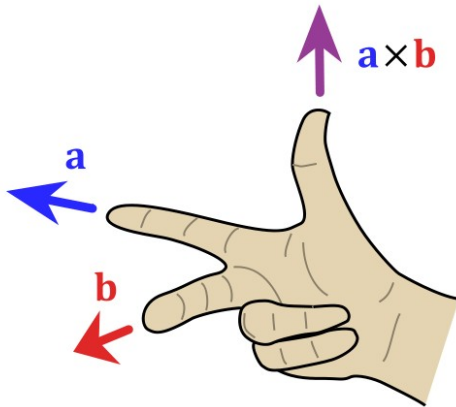
$${}^A\{B\} = \begin{bmatrix} {}^A_B R & {}^A\vec{t}_{B_{org}} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Side Bar: Cross Product

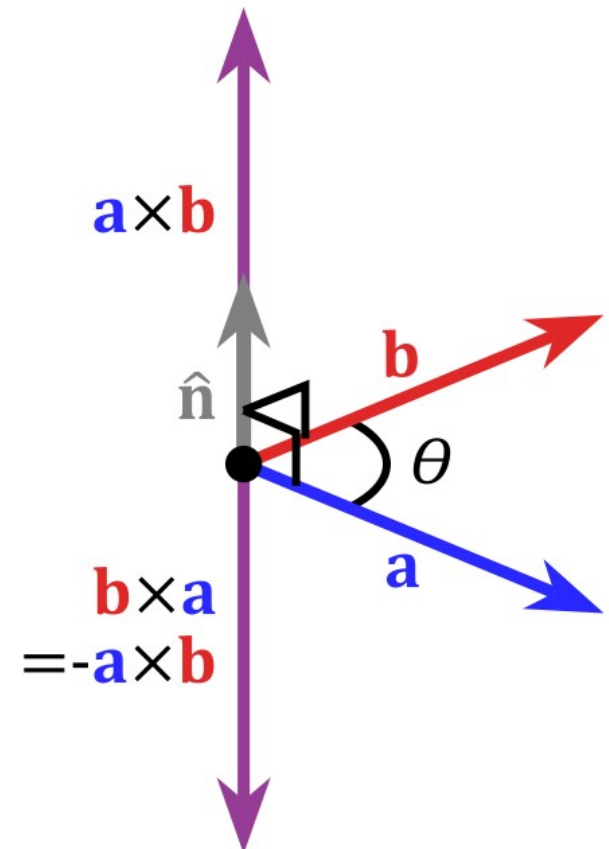
$$\mathbf{a} \times \mathbf{b} = |\mathbf{a}| |\mathbf{b}| \sin \theta \hat{\mathbf{n}}$$

In three dimensions:

$$\mathbf{a} \times \mathbf{b} = \begin{bmatrix} a_y b_z - a_z b_y \\ a_z b_x - a_x b_z \\ a_x b_y - a_y b_x \end{bmatrix}$$



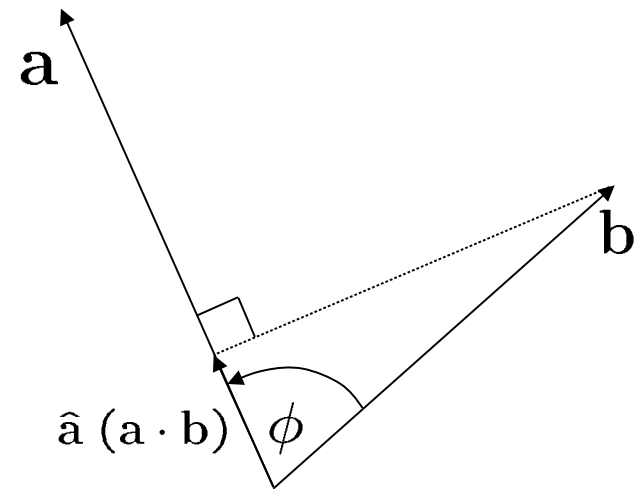
$$A = |\mathbf{a} \times \mathbf{b}| = |\mathbf{a}| |\mathbf{b}| \sin \theta$$



Sidebar: Dot Product

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

$$= \sum_{i=1}^n a_i b_i$$



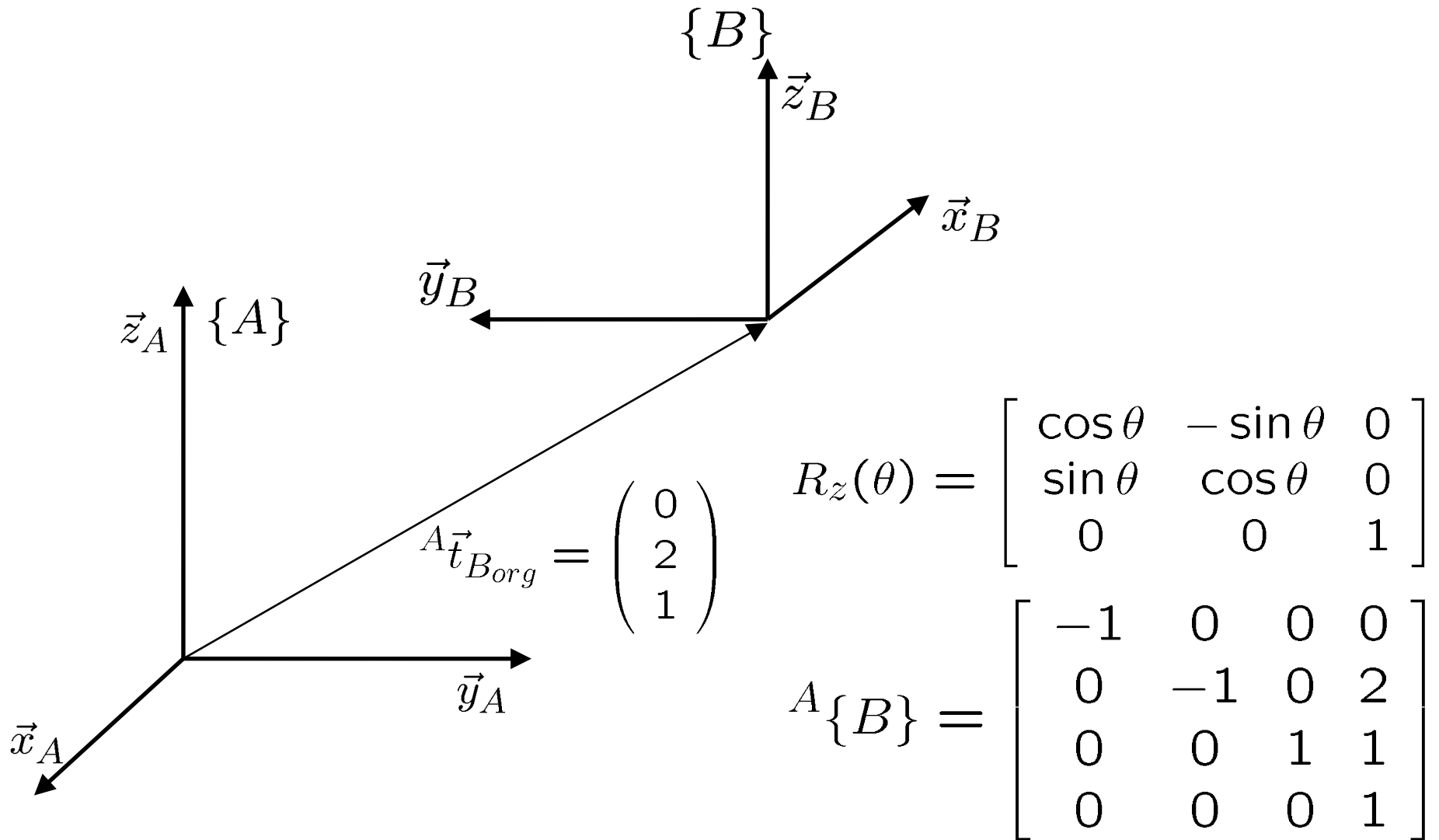
$$\hat{\mathbf{a}} = \frac{\mathbf{a}}{\|\mathbf{a}\|}$$

Three Uses for HTs

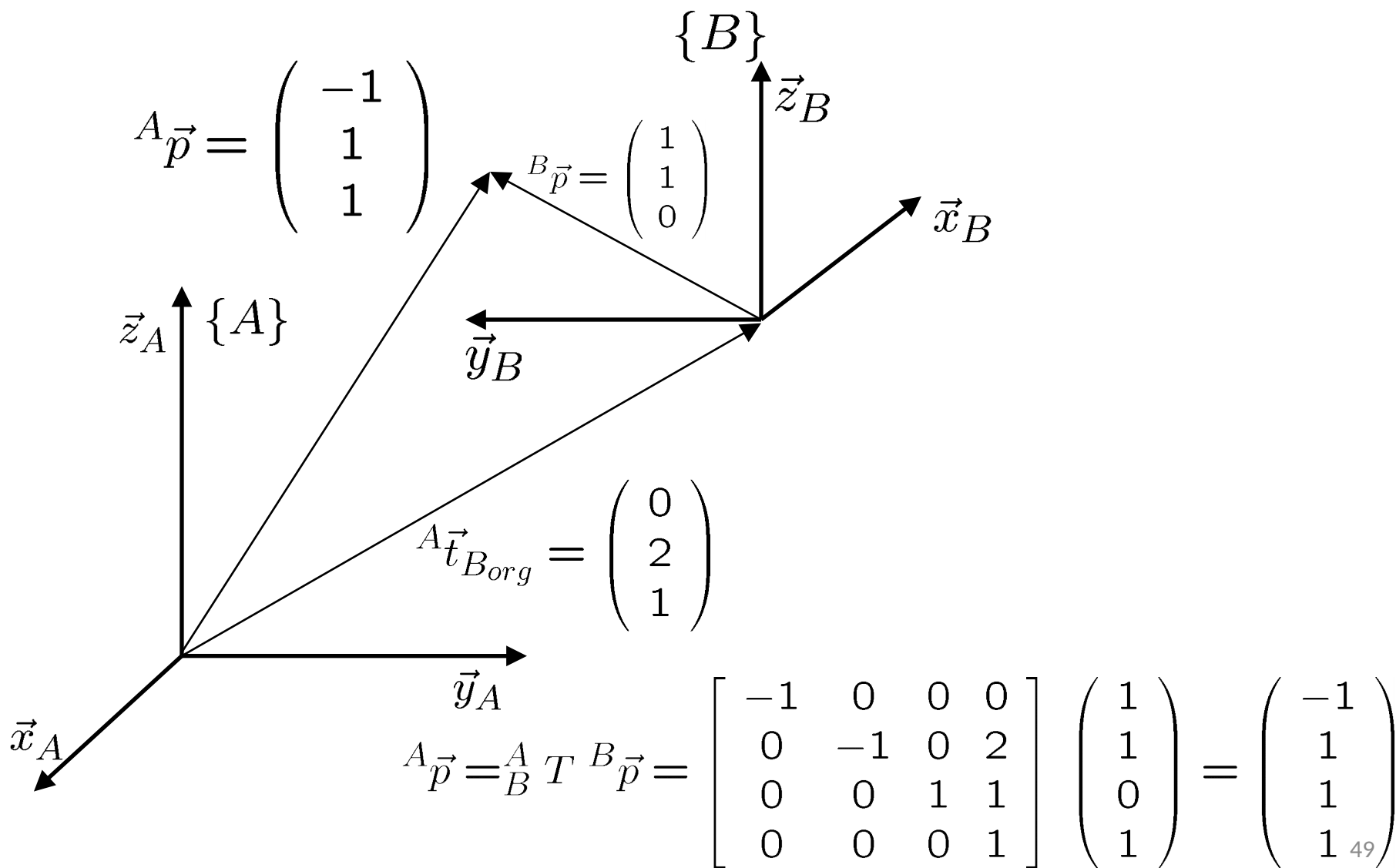
1. Frame Description
2. Mapping between frames
3. Transform operator

We will be somewhat loose about vectors and their homogeneous counterparts.

HT: Describing a Frame

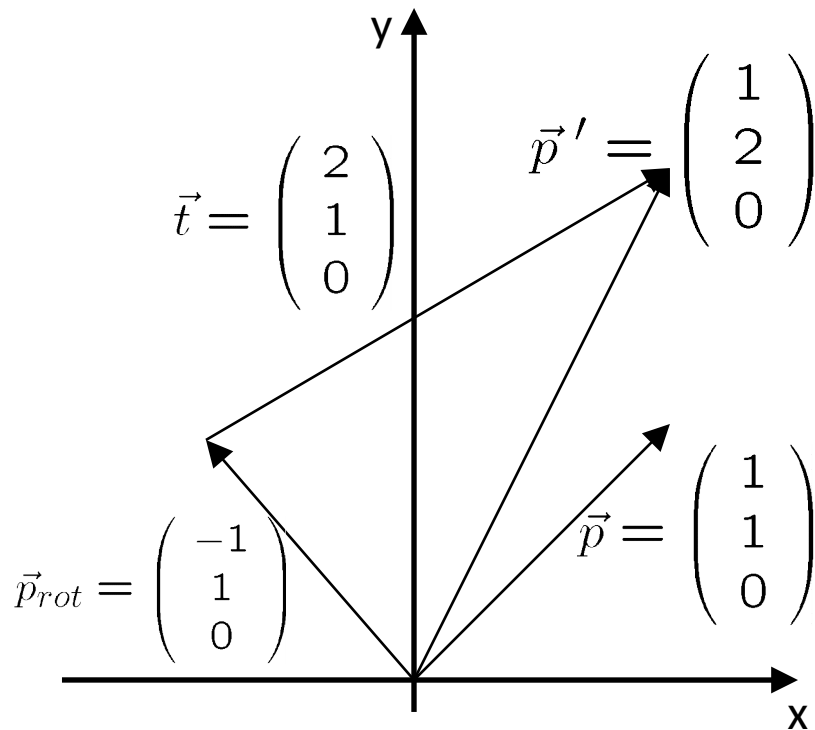


HT: Mapping between Frames



HT: Transforming a point

Rotate by $\pi/2$ about z and translate by (2,1,0)



$$\begin{aligned}\vec{p}' &= \begin{bmatrix} \cos \theta & -\sin \theta & 0 & 2 \\ \sin \theta & \cos \theta & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{pmatrix} 1 \\ 1 \\ 0 \\ 1 \end{pmatrix} \\ &= \begin{bmatrix} 0 & -1 & 0 & 2 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{pmatrix} 1 \\ 1 \\ 0 \\ 1 \end{pmatrix} \\ &= \begin{pmatrix} 1 \\ 2 \\ 0 \\ 1 \end{pmatrix}\end{aligned}$$

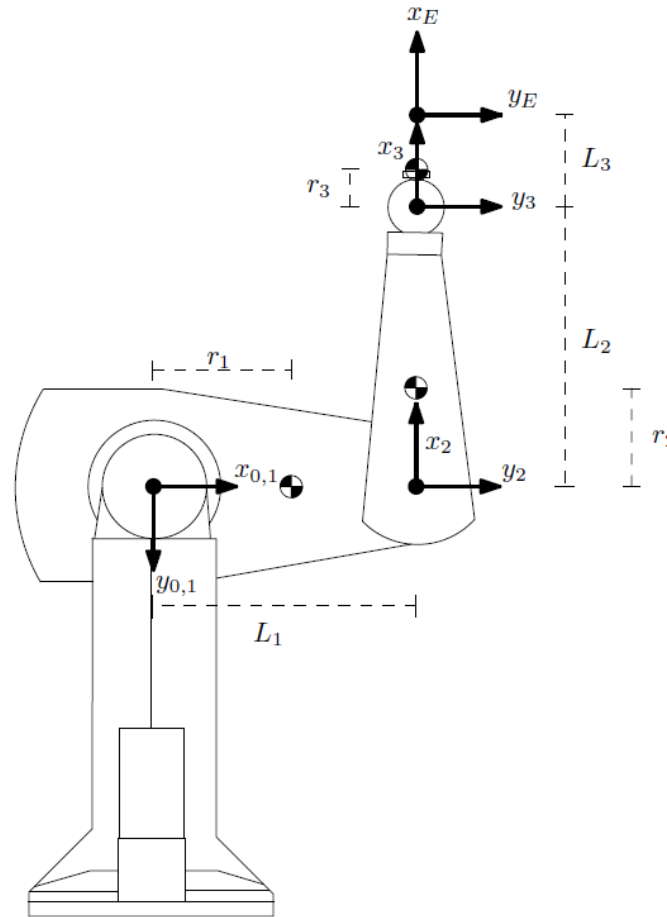
Computing with HTs

$$\begin{matrix} A \\ F \end{matrix} T = \begin{matrix} A \\ B \end{matrix} T \begin{matrix} B \\ C \end{matrix} T \begin{matrix} C \\ D \end{matrix} T \begin{matrix} D \\ E \end{matrix} T \begin{matrix} E \\ F \end{matrix} T$$

$$A \vec{p} = \begin{matrix} A \\ B \end{matrix} T \begin{matrix} B \\ C \end{matrix} T C \vec{p}$$

Matrix multiplication is NOT commutative!
ORDER MATTERS !!!

We Now Have Forward Kinematics!



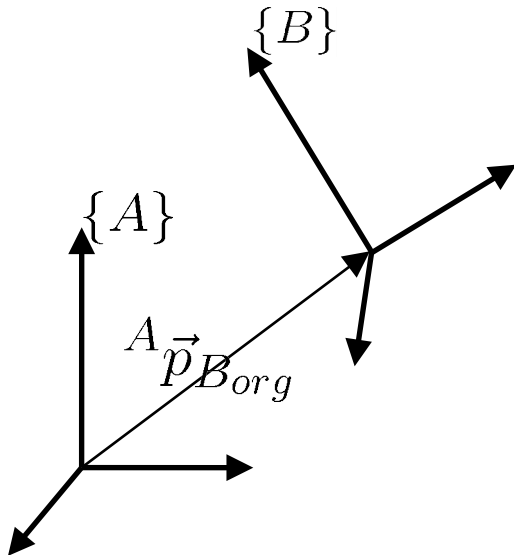
$$\mathbf{x} = f(\mathbf{q}) = {}^0_E T(\mathbf{q}) = {}^0_1 T(q_1) {}^1_2 T(q_2) {}^2_3 T(q_3) {}^3_E T$$

Inverting ${}^A_B T$!

$${}^A \vec{p} = {}^A_B T {}^B \vec{p} = \begin{bmatrix} {}^A_B R & {}^A \vec{p}_{B_{org}} \\ 0 & 1 \end{bmatrix} {}^B \vec{p} \qquad {}^B_A R = {}^A_B R^T$$

$${}^B ({}^A \vec{p}_{B_{org}}) = {}^B_A R {}^A \vec{p}_{B_{org}} + {}^B \vec{p}_{A_{org}}$$

$${}^B \vec{p}_{A_{org}} = -{}^B_A R {}^A \vec{p}_{B_{org}} = -{}^A_B R^T {}^A \vec{p}_{B_{org}}$$



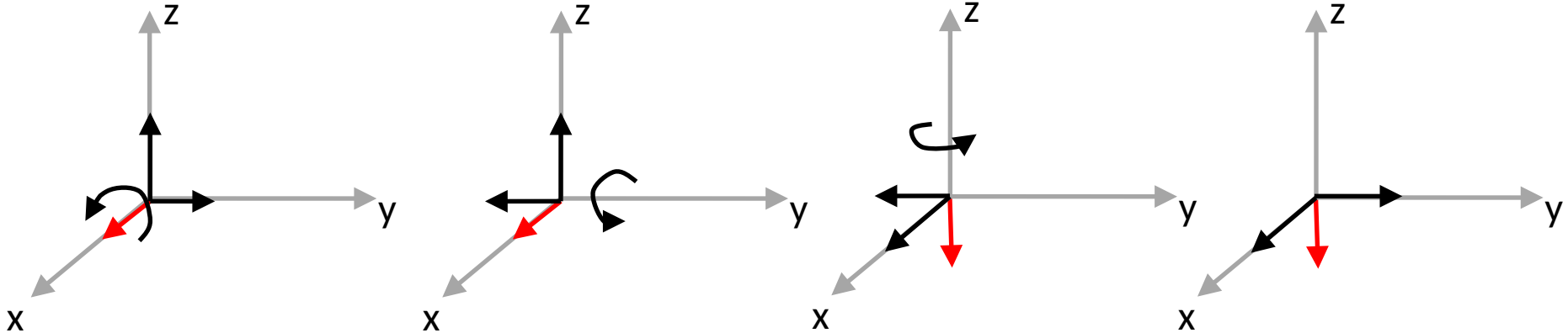
$${}^B_A T = \begin{bmatrix} {}^A_B R^T & -{}^A_B R^T {}^A \vec{p}_{B_{org}} \\ 0 & 1 \end{bmatrix}$$

Review

- Affine transformations
- Homogeneous transforms
 - describe frames
 - map between frames
 - transform point
- Operations with HTs
 - concatenation
 - inversion

X-Y-Z Fixed Angles (roll, pitch, yaw)

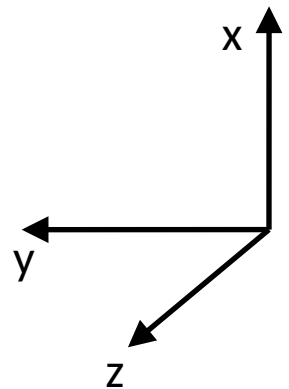
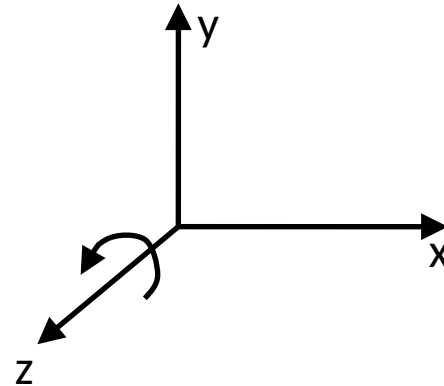
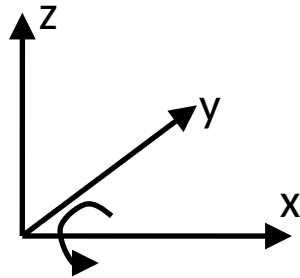
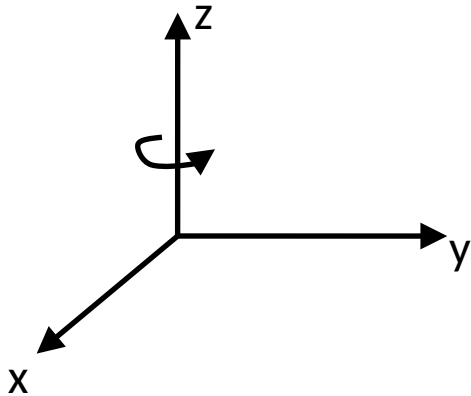
$(\pi/2, \pi/2, \pi/2)$



Other combination of axes exist.

Z-X-Z Euler Angles

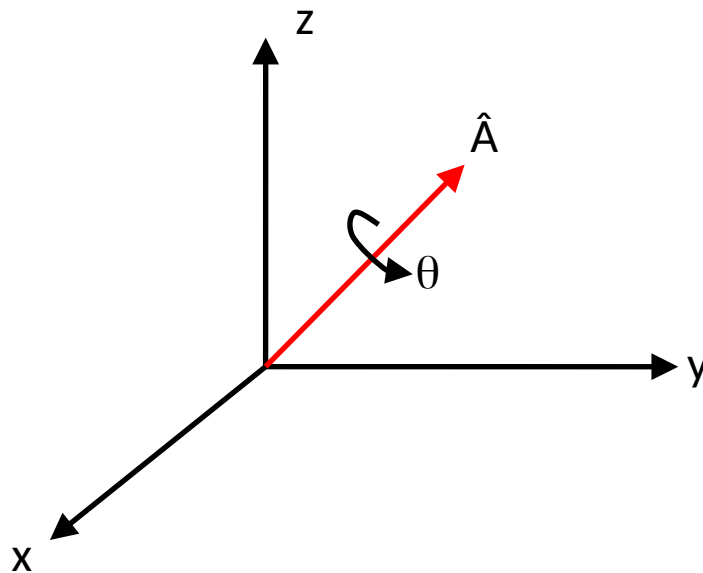
$(\pi/2, \pi/2, \pi/2)$



Other combination of axes exist.

Arbitrary Axis

- Direction of vector indicates axis \hat{A}
- $|\hat{A}| = 1$, so only two numbers are required
- Magnitude given by angle θ



Conversions

- Exist between all representations
- For example, XYZ fixed angle to rotation matrix:

$$R_{XYZ}(\alpha, \beta, \gamma) = \begin{bmatrix} c\gamma c\beta & c\gamma s\beta s\alpha - s\gamma c\alpha & c\gamma s\beta c\alpha + s\gamma s\alpha \\ s\gamma c\beta & s\gamma s\beta s\alpha + c\gamma c\alpha & s\gamma s\beta c\alpha - c\gamma s\alpha \\ -s\beta & c\beta s\alpha & c\beta c\alpha \end{bmatrix}$$

Line Vector vs. Free Vector

$$A \vec{f}_{p \in B}$$

Force Vector

depends on line of action and point of application, in addition to magnitude and direction

Velocity Vector

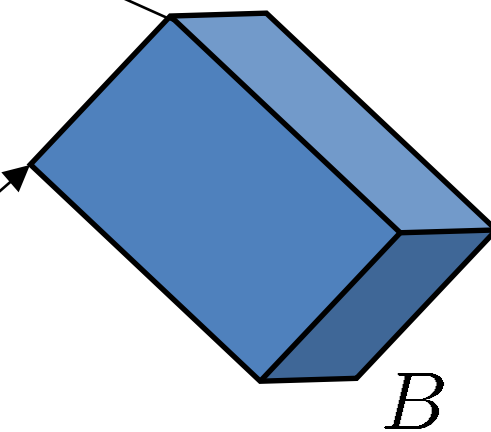
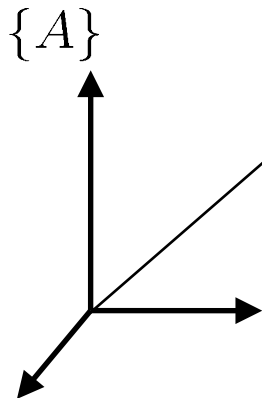
only depends on magnitude and direction

$$A \vec{v}_B$$

$$A \vec{t}_{B_0}$$

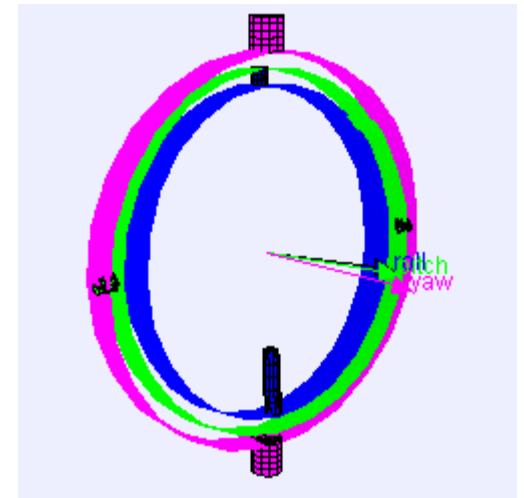
Position Vector

depends on line of action and point of application, in addition to magnitude and direction



Problems with Rotation

- Rotation matrices are used most often, but
 - numerical error buildup
 - interpolation
 - complexity of multiplication
- Angle representations address
 - numerical error buildup
- Arbitrary axis representation introduces
 - zero rotation problem



Quaternions

- Address:
 - numerical error buildup
 - interpolation
 - complexity of multiplication (among themselves)
- But:
 - eliminate homogeneity
 - are inefficient when many points are rotated
(then we convert them into a matrix before performing the rotation)