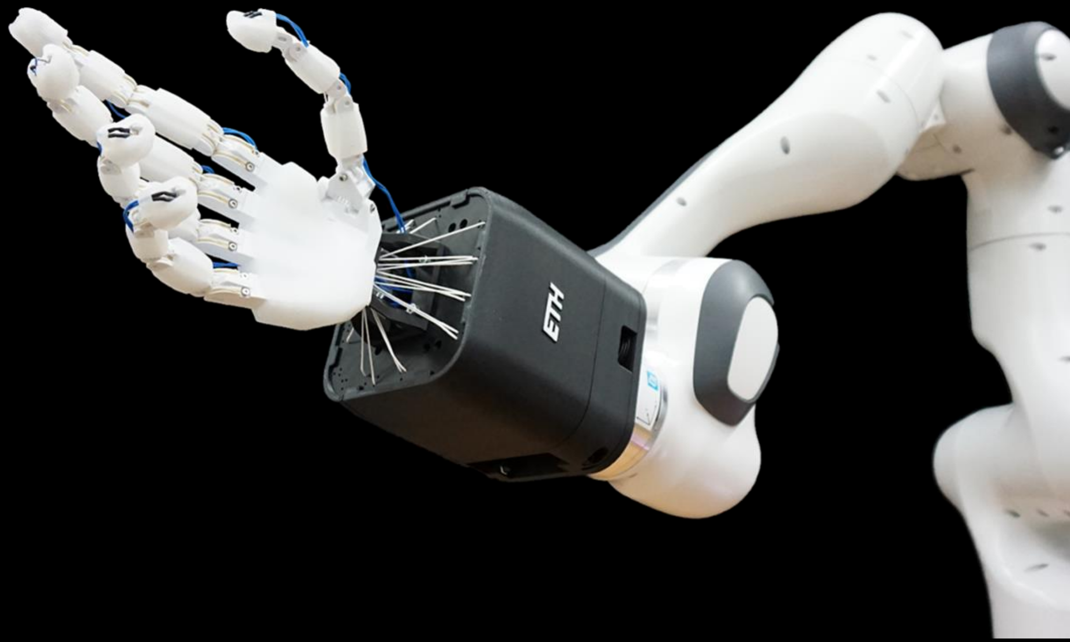
ETH zürich

Kinematics, Dynamics and Control of Robotic Hands

Robert Katzschmann

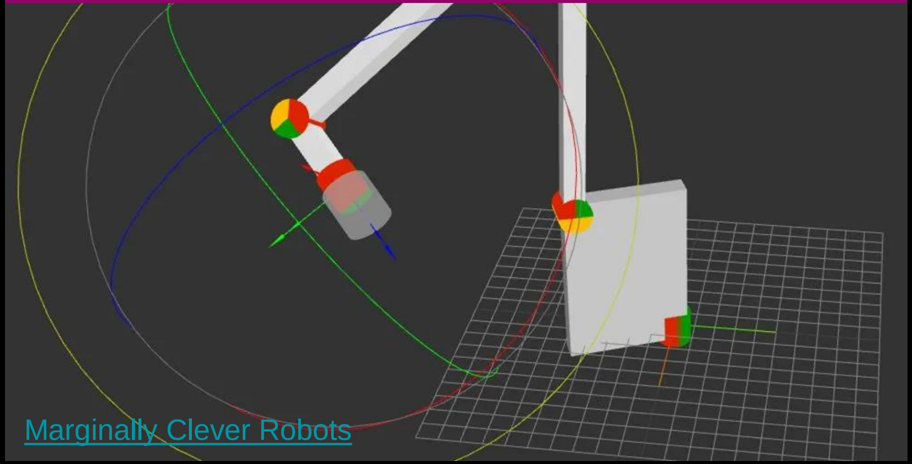
Assistant Professor of Robotics, Soft Robotics Lab



SoftRobotics Laboratory

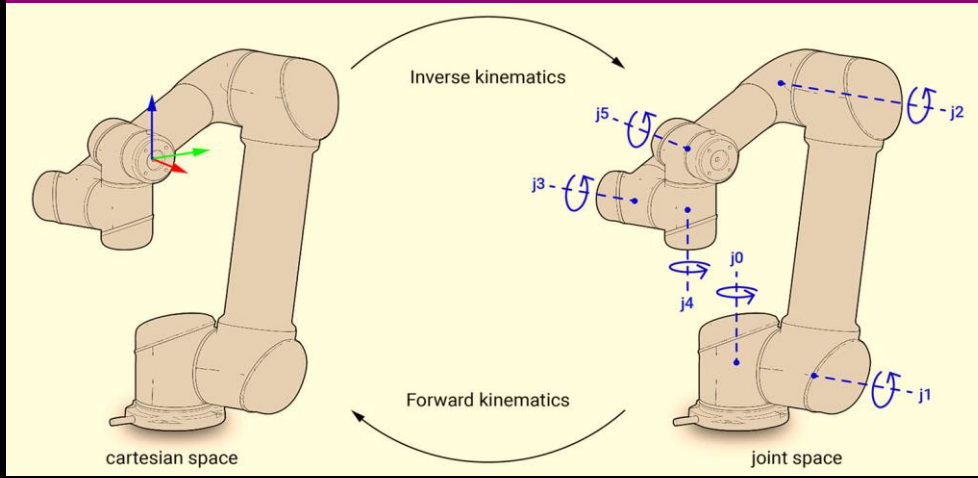
Focus Topics for Today

1. Intro to Robot Kinematics and Dynamics



Marginally Clever Robots

2. Forward and Inverse Kinematics

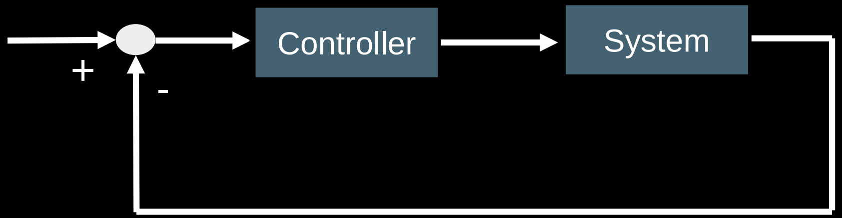


compas_fab

3. Kinematics and Dynamics for hand joints

Mimic Robotics

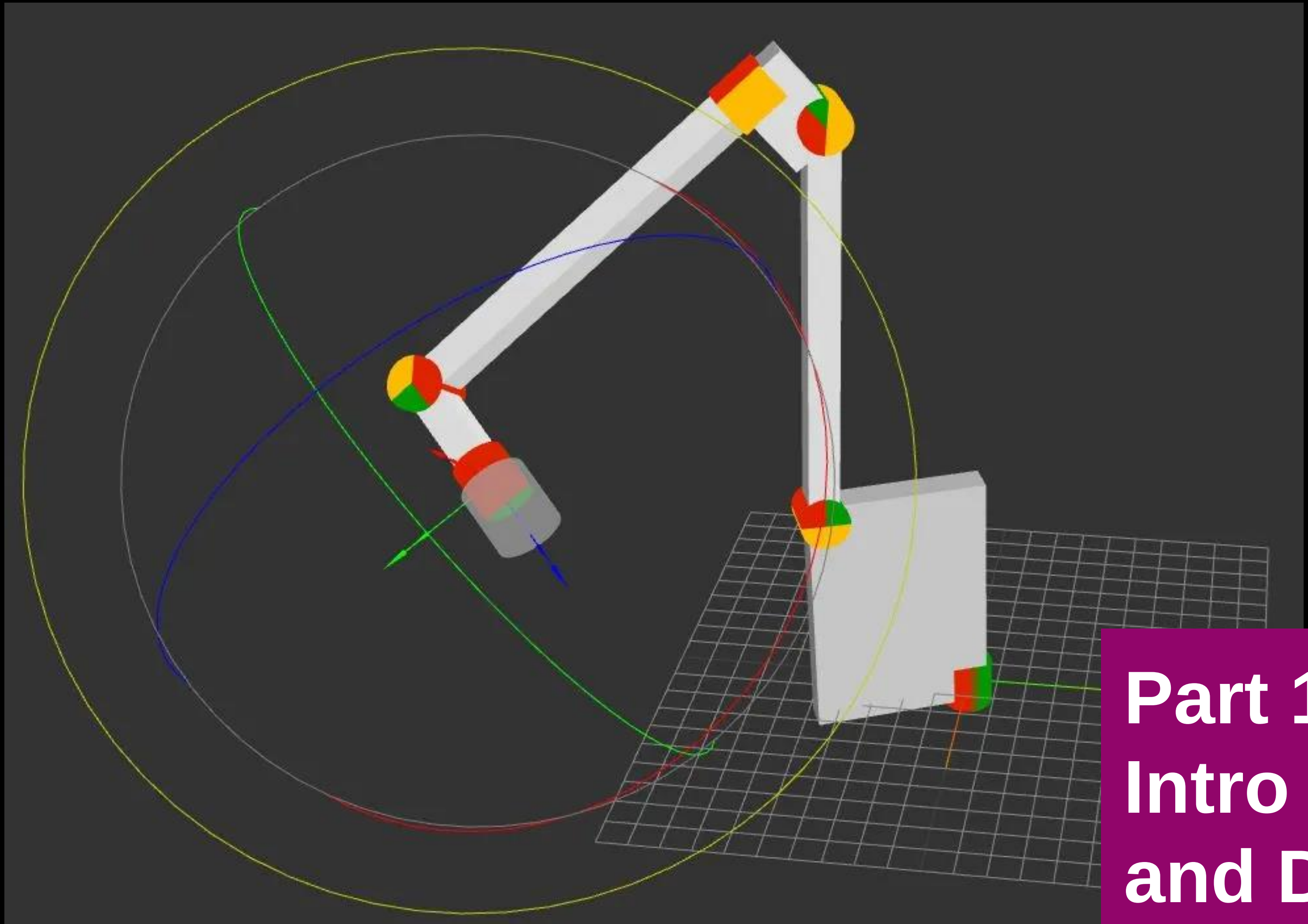
4. Control



5. Challenges

ETH zürich

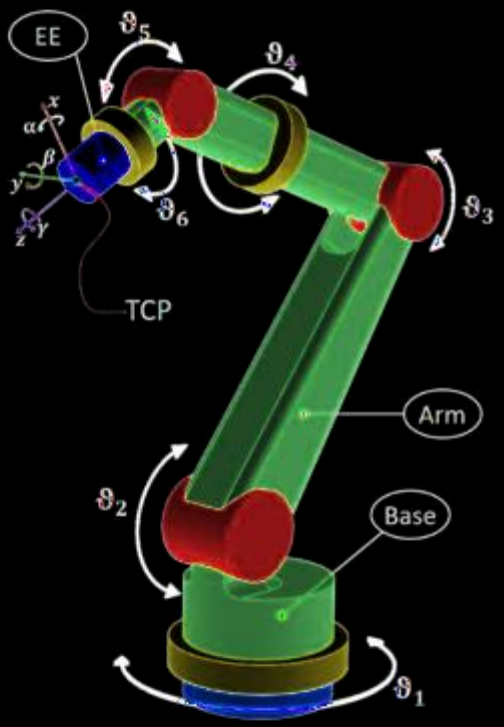
SoftRobotics Laboratory



Marginally Clever Robots

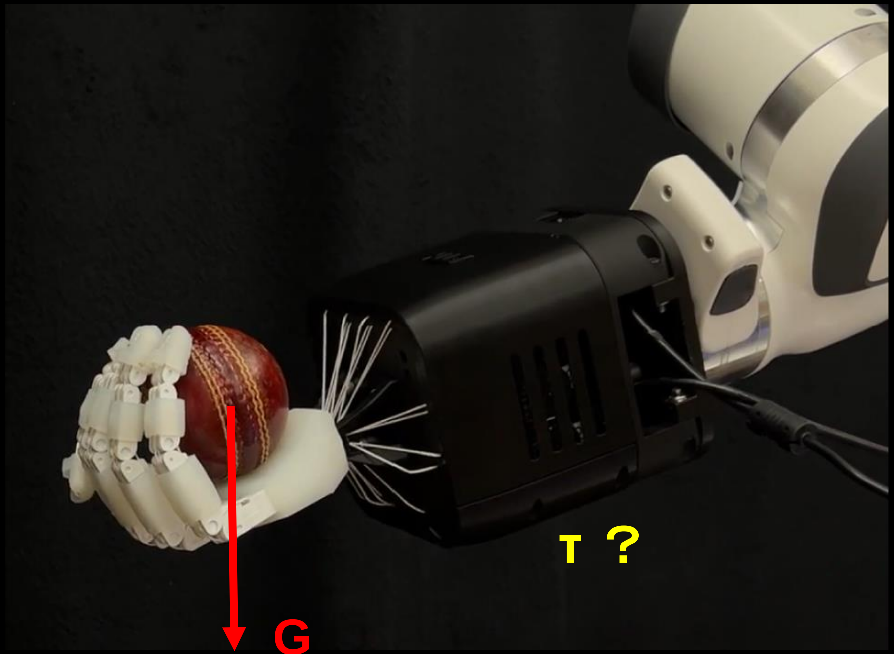
Part 1: Intro to Robot Kinematics and Dynamics

Robot Kinematics and Dynamics



researchgate.net

Kinematics



Toshimitsu et al. (2023) https://srl-ethz.github.io/get-ball-rolling/

Dynamics

Simulation
reaction to certain actuator commands

Control
invert of simulation, want to get somewhere, what to command?

Design
how are the loads distributed?

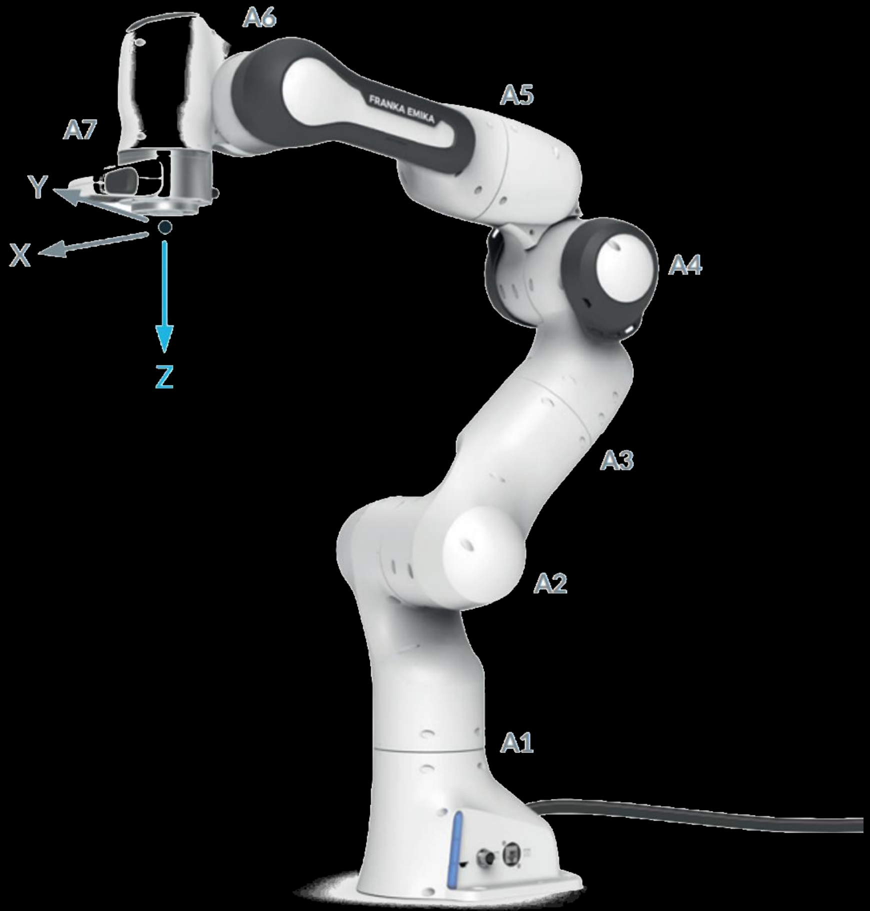
Optimization
what dimension should I have?

Actuation
torque, speed, powder etc.

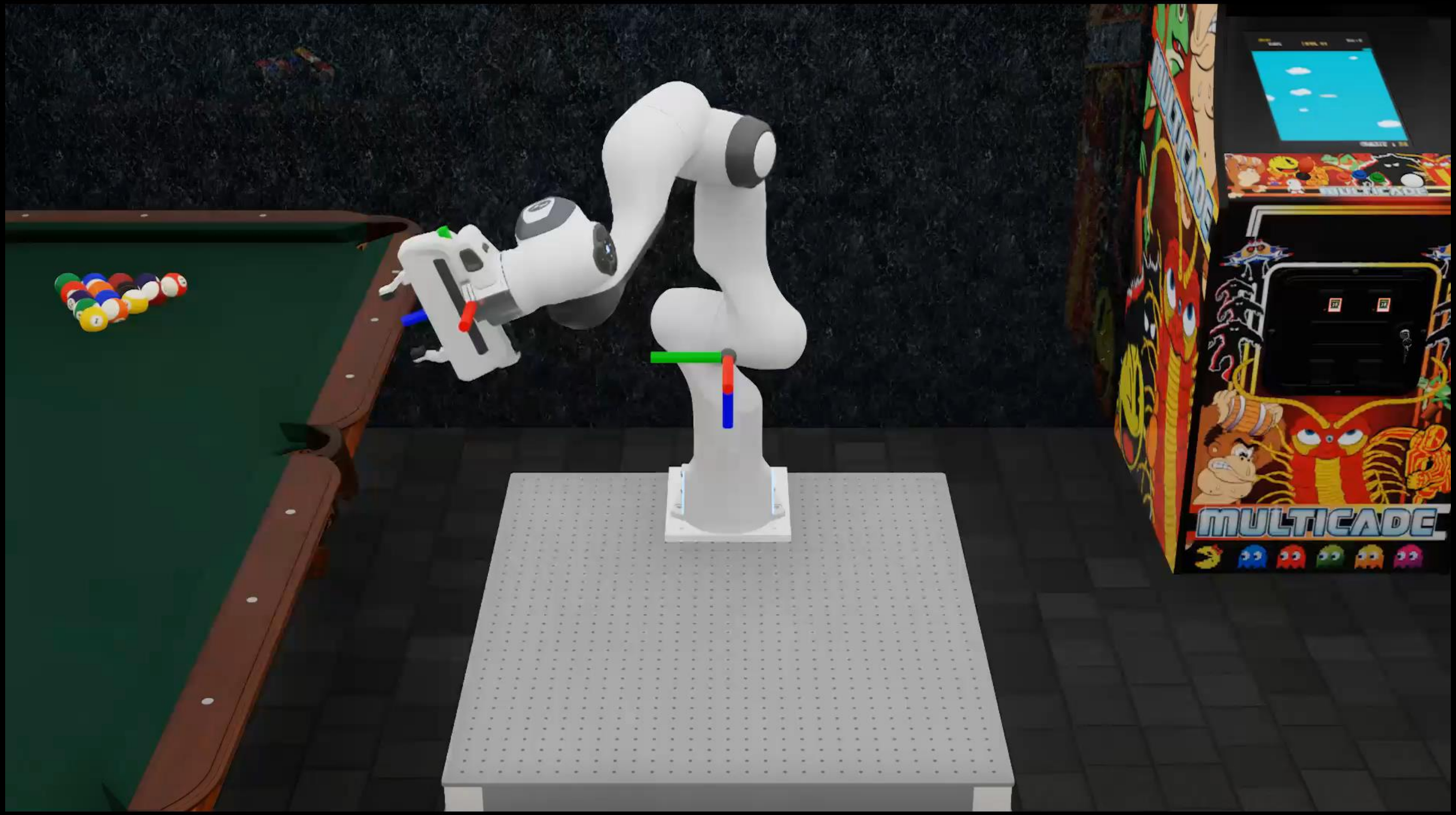
ETH zürich

SoftRobotics Laboratory

Robotic Arm



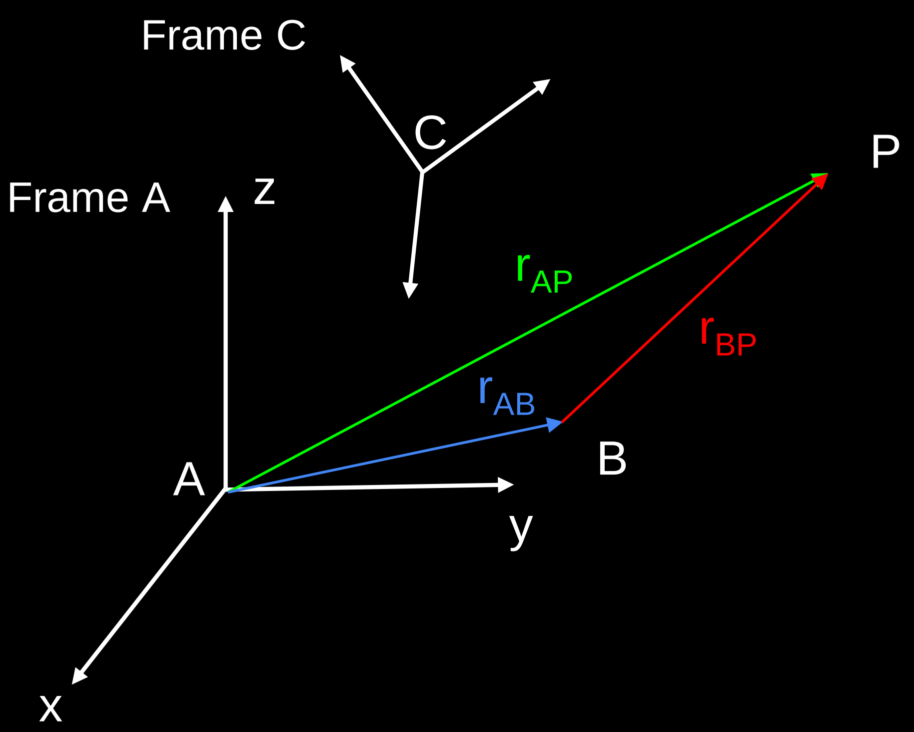
franka.de

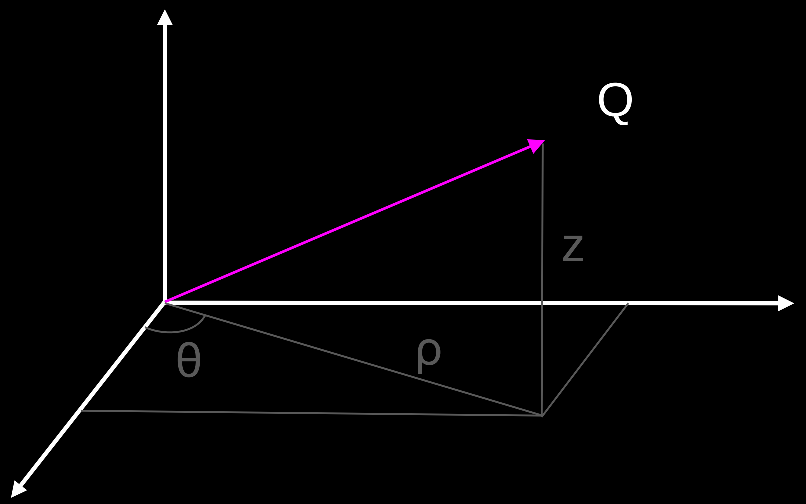


Videos from Orbit

ETH zürich SoftRobotics Laboratory

Points, Lines, and Coordinates





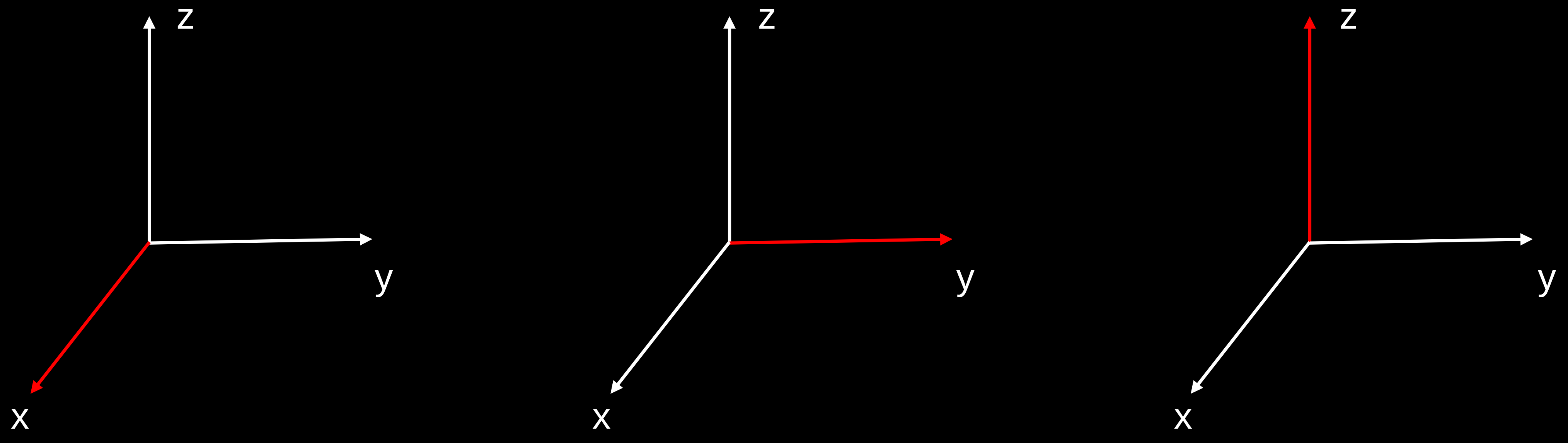
Point P in Cartesian Coordinates Frame A: $_{A}X_{P} = \begin{pmatrix} x \\ y \\ z \end{pmatrix}$

Point Q in Cylindrical Coordinate: $X_{Q} = \begin{pmatrix} \rho \\ \theta \\ z \end{pmatrix}$

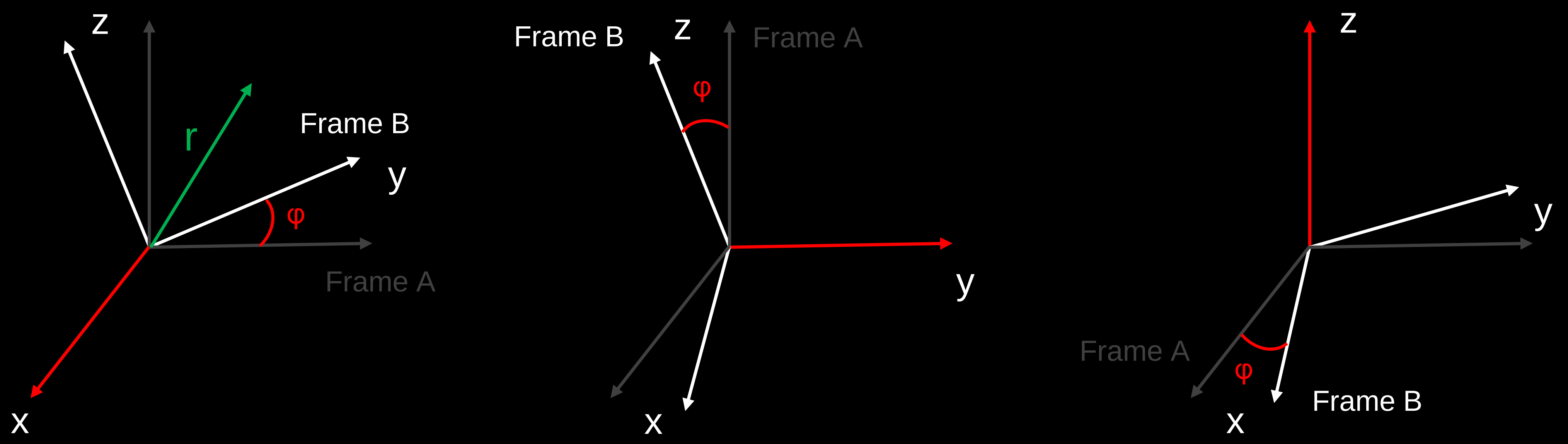
$$_{A}r_{AP} = {_{A}r_{AB}} + {_{A}r_{BP}}$$
$$_{A}r_{AP} \neq {_{A}r_{AB}} + {_{C}r_{BP}}$$

ETH zürich SoftRobotics Laboratory

Rotation



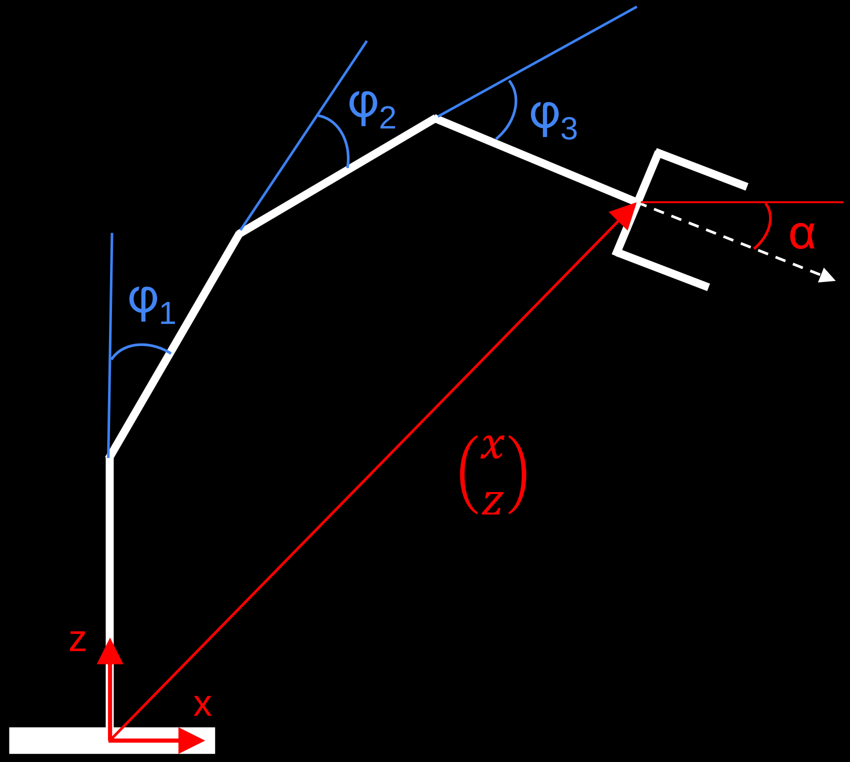
Rotation



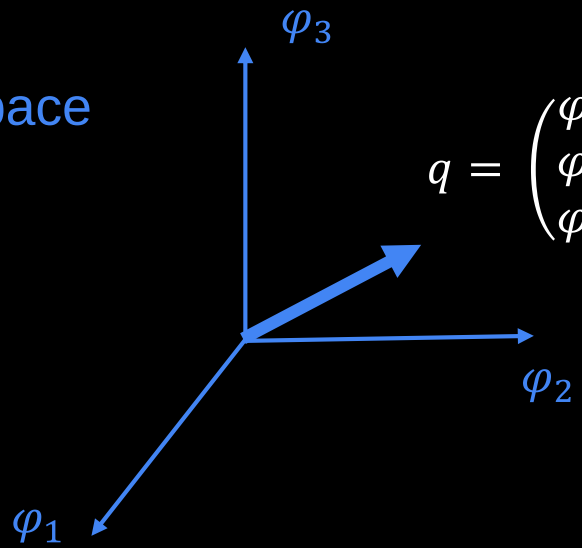
$$C_x(\varphi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\varphi & -sin\varphi \\ 0 & sin\varphi & cos\varphi \end{bmatrix} \quad C_y(\varphi) = \begin{bmatrix} cos\varphi & 0 & sin\varphi \\ 0 & 1 & 0 \\ -sin\varphi & 0 & cos\varphi \end{bmatrix} \quad C_z(\varphi) = \begin{bmatrix} cos\varphi & -sin\varphi & 0 \\ sin\varphi & cos\varphi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$${}_A r = C_{AB} \cdot {}_B r \rightarrow \begin{pmatrix} {}_A x \\ {}_A y \\ {}_A z \end{pmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\varphi & -sin\varphi \\ 0 & sin\varphi & cos\varphi \end{bmatrix} \begin{pmatrix} {}_B x \\ {}_B y \\ {}_B z \end{pmatrix} = \begin{pmatrix} {}_B x \\ {}_B y \cdot cos\varphi - {}_B z \cdot sin\varphi \\ {}_B y \cdot sin\varphi + {}_B z \cdot cos\varphi \end{pmatrix}$$

Joint Space and Task Space



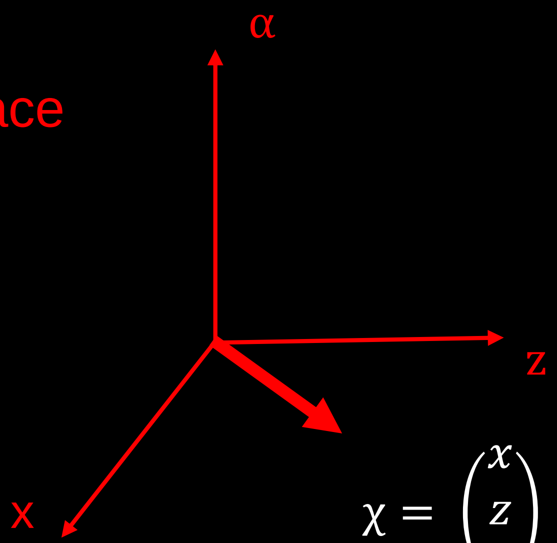
Joint Space



$$q = \begin{pmatrix} \varphi_1 \\ \varphi_2 \\ \varphi_3 \end{pmatrix}$$

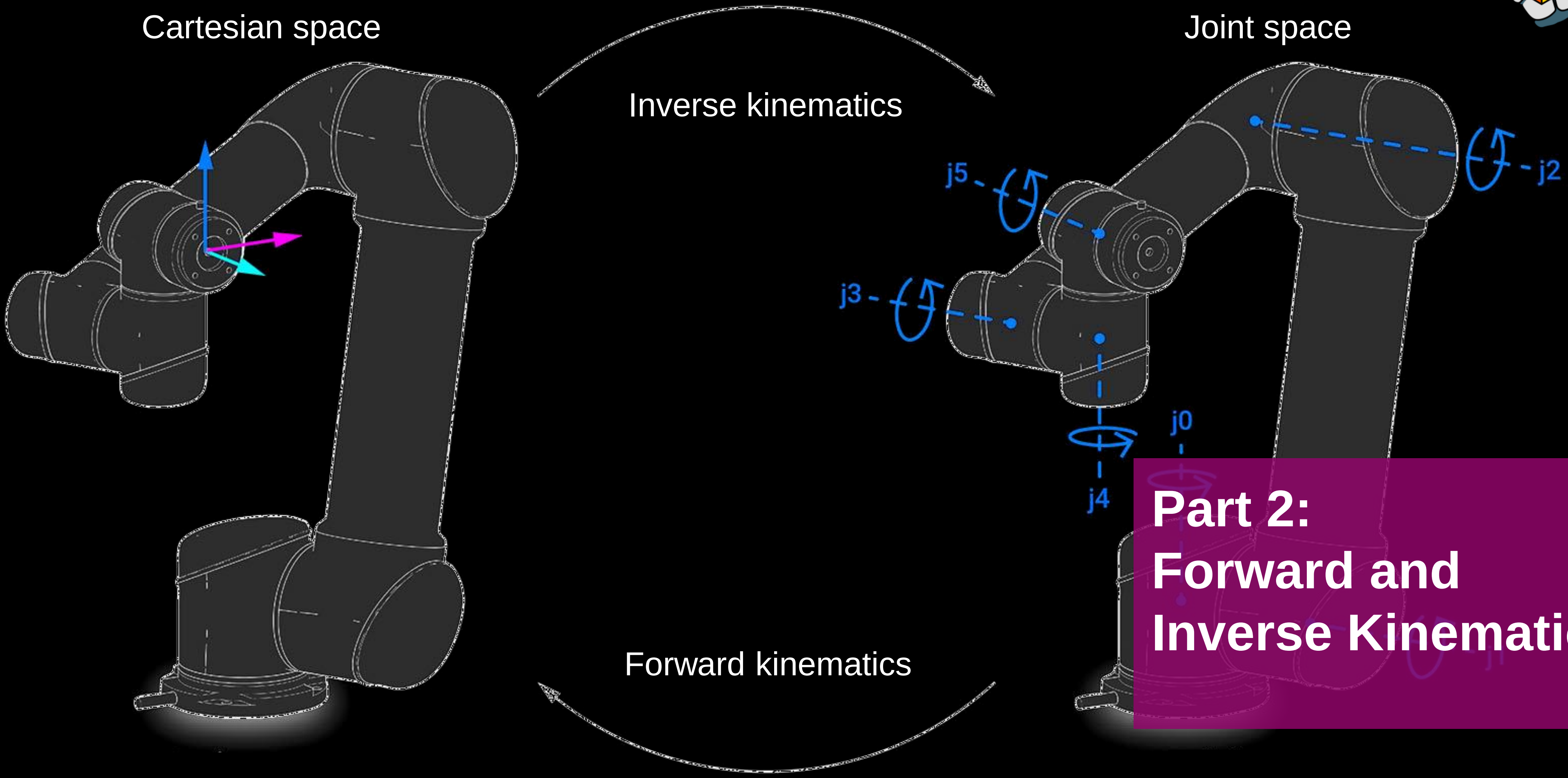
Joint Coordinates

Task Space



$$\chi = \begin{pmatrix} x \\ z \\ \alpha \end{pmatrix}$$

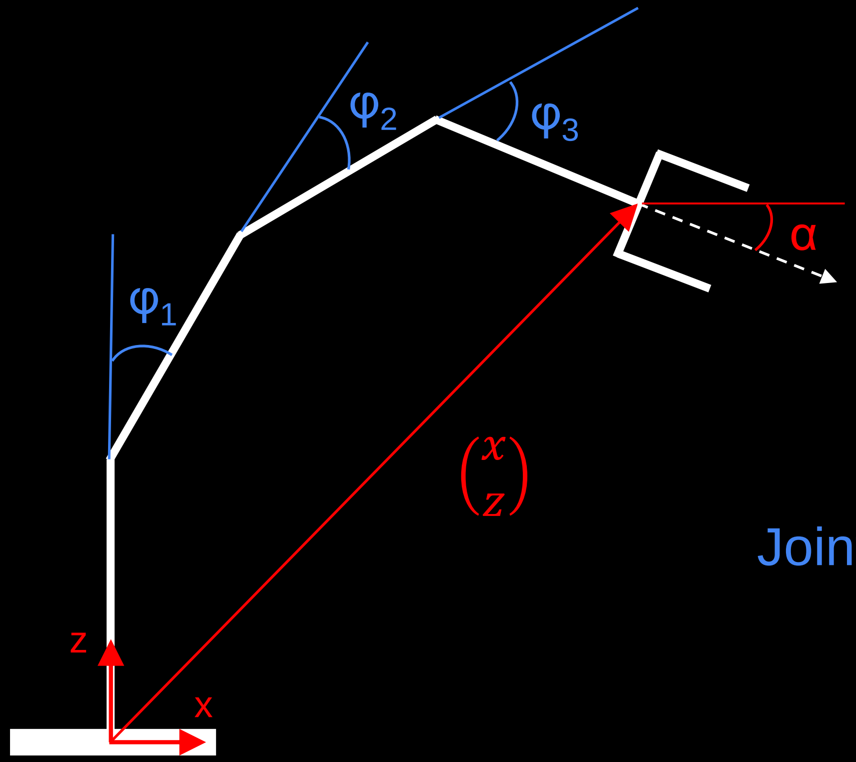
Task Coordinates



Part 2: Forward and Inverse Kinematics

compas_fab

Forward and Inverse Kinematics

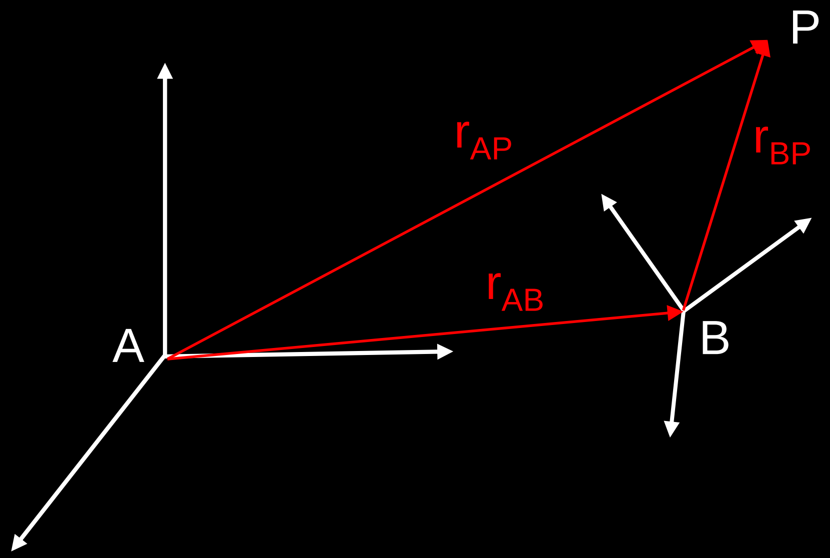


Joint Space → Forward Kinematics → Task Space

Task Space → Inverse Kinematics → Joint Space

From greek ***kinema*** = motion

Homogeneous Transformation Matrix



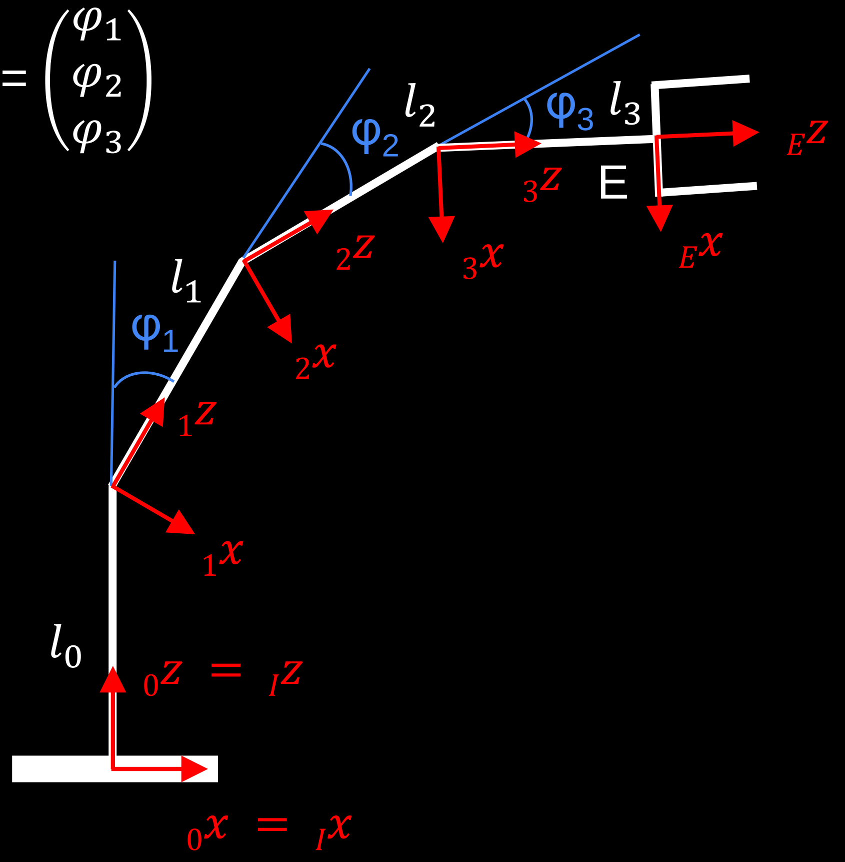
$$r_{AP} = r_{AB} + r_{BP}$$

$${}_A r_{AP} = {}_A r_{AB} + {}_A r_{BP} = {}_A r_{AB} + C_{AB} \cdot {}_B r_{BP}$$

$$\begin{pmatrix} {}_A r_{AP} \\ 1 \end{pmatrix} = \underbrace{\begin{bmatrix} \boxed{C_{AB}} & \boxed{{}_A r_{AB}} \\ 0_{1x3} & 1 \end{bmatrix}}_{T_{AB}} \begin{pmatrix} {}_B r_{BP} \\ 1 \end{pmatrix}$$

Homogeneous Transformation Matrix

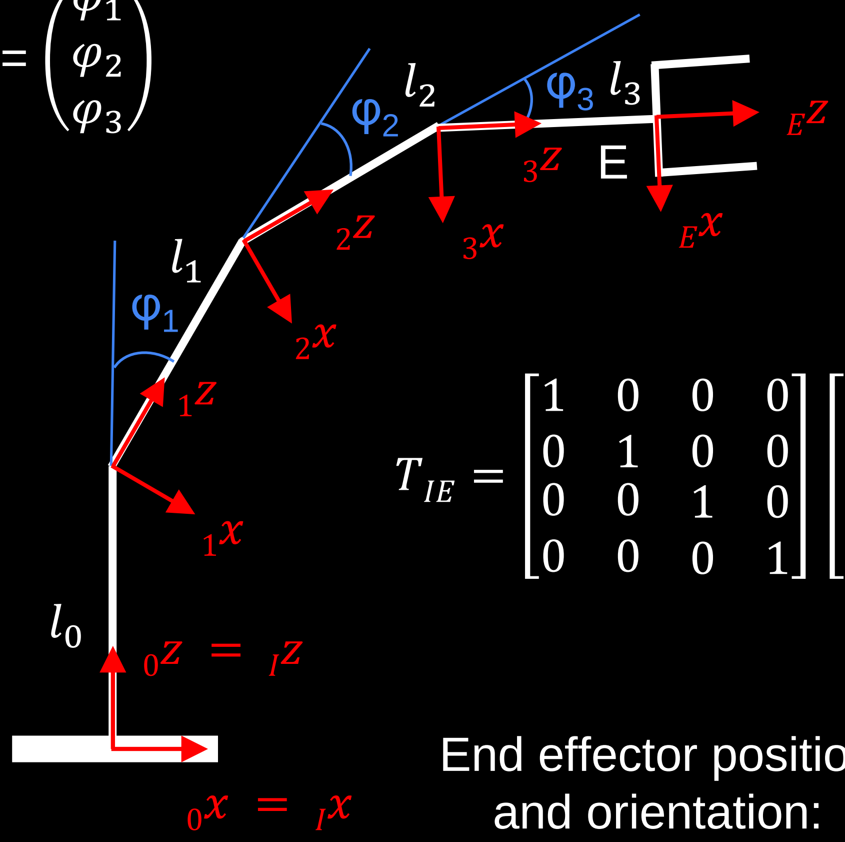
$q = \begin{pmatrix} \varphi_1 \\ \varphi_2 \\ \varphi_3 \end{pmatrix}$



$$T_{IE} = T_{I0} \cdot T_{01} \cdot T_{12} \cdot T_{23} \cdot T_{3E}$$

Homogeneous Transformation Matrix

$$q = \begin{pmatrix} \varphi_1 \\ \varphi_2 \\ \varphi_3 \end{pmatrix}$$



$$T_{IE} = TI_0 \cdot T_{01} \cdot T_{12} \cdot T_{23} \cdot T_{3E}$$

$$T_{IE} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c_1 & 0 & s_1 & 0 \\ 0 & 1 & 0 & 0 \\ -s_1 & 0 & c_1 & l_0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c_2 & 0 & s_2 & 0 \\ 0 & 1 & 0 & 0 \\ -s_2 & 0 & c_2 & l_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c_3 & 0 & s_3 & 0 \\ 0 & 1 & 0 & 0 \\ -s_3 & 0 & c_3 & l_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & l_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

End effector position and orientation:

$${}_IX_E(q) = \begin{pmatrix} l_1 \sin(\varphi_1) + l_2 \sin(\varphi_1 + \varphi_2) + l_3 \sin(\varphi_1 + \varphi_2 + \varphi_3) \\ l_0 + l_1 \cos(\varphi_1) + l_2 \cos(\varphi_1 + \varphi_2) + l_3 \cos(\varphi_1 + \varphi_2 + \varphi_3) \\ \varphi_1 + \varphi_2 + \varphi_3 \end{pmatrix}$$

Forward Differential Kinematics and Jacobian

$$\delta X_E \approx \frac{\delta X_E(q)}{\delta q}\delta q = J_{EA}(q)\delta q$$

with $J_{EA} = \frac{\delta X_E}{\delta q} = \begin{bmatrix} \frac{\delta X_1}{\delta q_1} & \cdots & \frac{\delta X_1}{\delta q_n} \\ \vdots & \ddots & \vdots \\ \frac{\delta X_m}{\delta q_1} & \cdots & \frac{\delta X_m}{\delta q_n} \end{bmatrix}$

$$\dot{X}_E = J_{EA}(q)\dot{q}$$

with $J_{EA}(q) \in \mathbb{R}^{m\times n}$

Inverse Kinematics

- Previously we showed that:

$$J(q)\dot{q} = \chi_e = \begin{bmatrix} \dot{p}_e \\ w_e \end{bmatrix}$$

- If we invert it we obtain:

$$\dot{q} = J^{+}\chi_e \text{ with } \mathrm{J}^{+} = J^{T}\left(JJ^{T}\right)^{-1}$$

- And in a differential form:

$$\Delta\chi_e = J^{+}\Delta q$$

Algorithm 1 Numerical Inverse Kinematics

$\mathbf{q} \leftarrow \mathbf{q}^0$ ▷ Start configuration
while $\|\chi_e^* - \chi_e(\mathbf{q})\| > tol$ **do** ▷ While the solution is not reached
    $\mathbf{J}_{eA} \leftarrow \mathbf{J}_{eA}(\mathbf{q}) = \frac{\partial \chi_e}{\partial \mathbf{q}}(\mathbf{q})$ ▷ Evaluate Jacobian for $\mathbf{q}$
    $\mathbf{J}_{eA}^{+} \leftarrow (\mathbf{J}_{eA})^{+}$ ▷ Calculate the pseudo inverse
    $\Delta\chi_e \leftarrow \chi_e^* - \chi_e(\mathbf{q})$ ▷ Find the end-effector configuration error vector
    $\mathbf{q} \leftarrow \mathbf{q} + \mathbf{J}_{eA}^{+}\Delta\chi_e$ ▷ Update the generalized coordinates
end while

A possible inverse kinematics algorithm

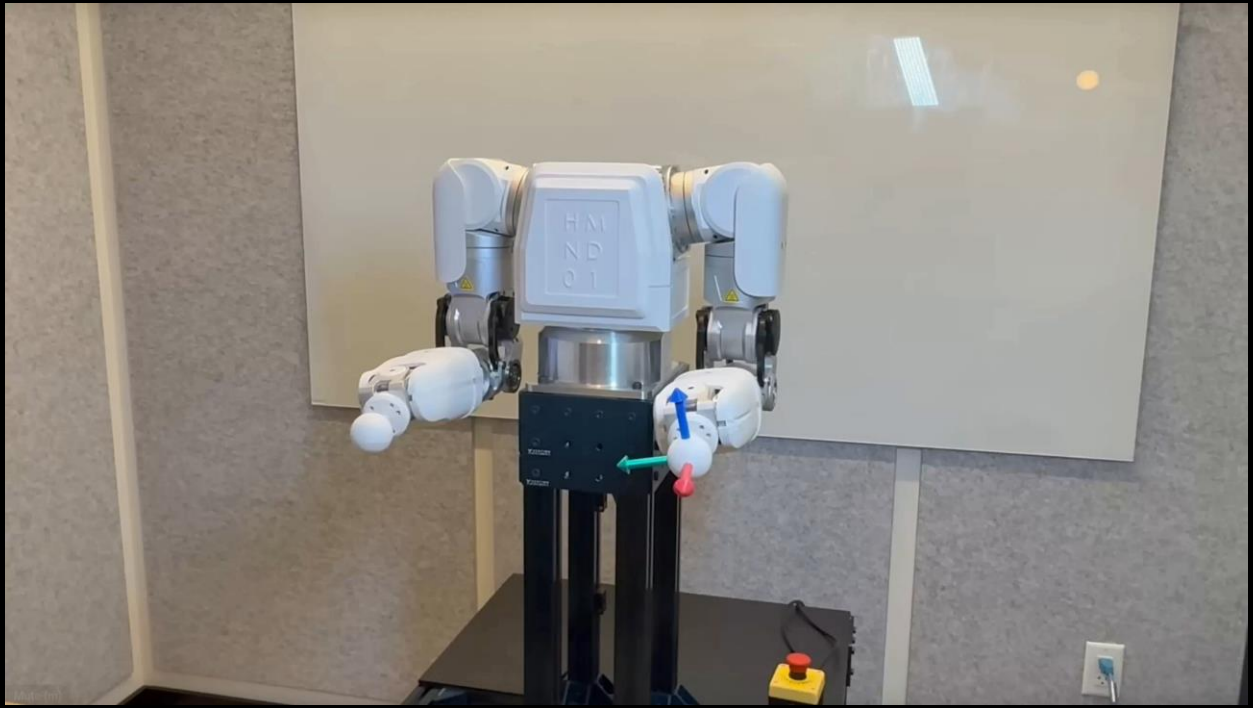
Robot Dynamics Class @ ETH Zurich

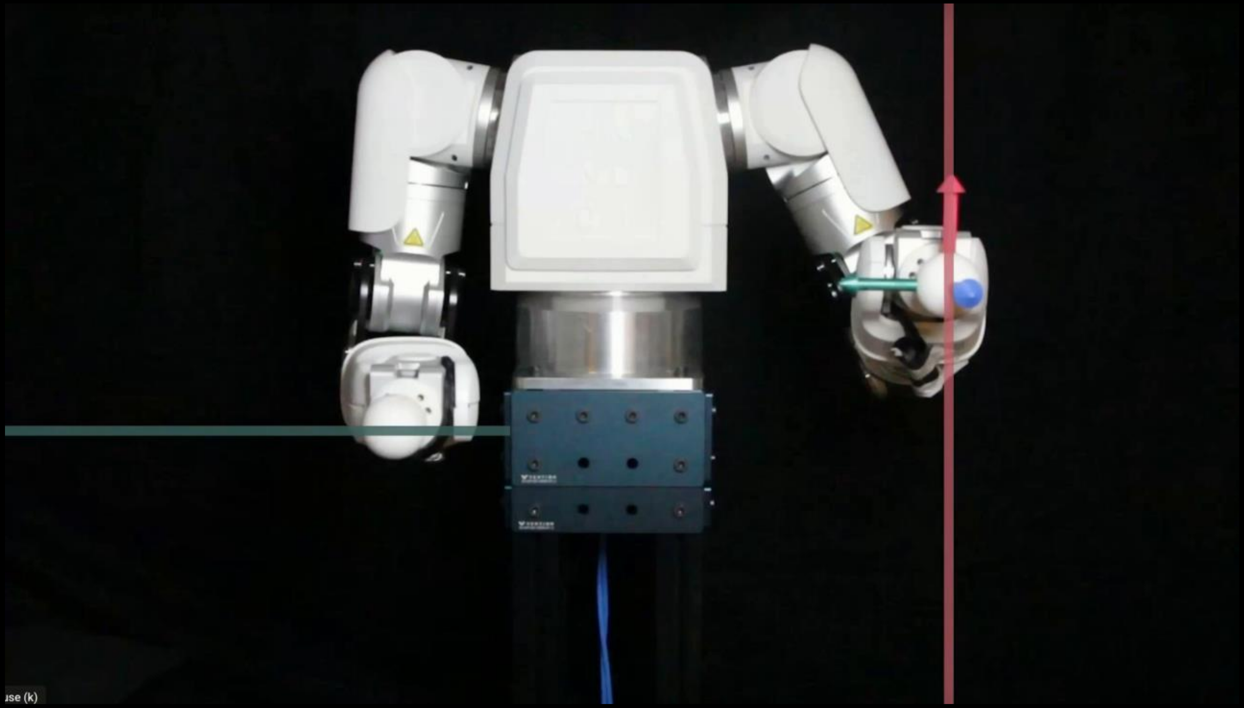
To overcome stability issues, the update can be scaled by a factor k

-> slower convergence

$$q \leftarrow q + kJ_{eA}^{+}\Delta\chi_e \text{ with } k \in (0,1)$$

Inverse Kinematics





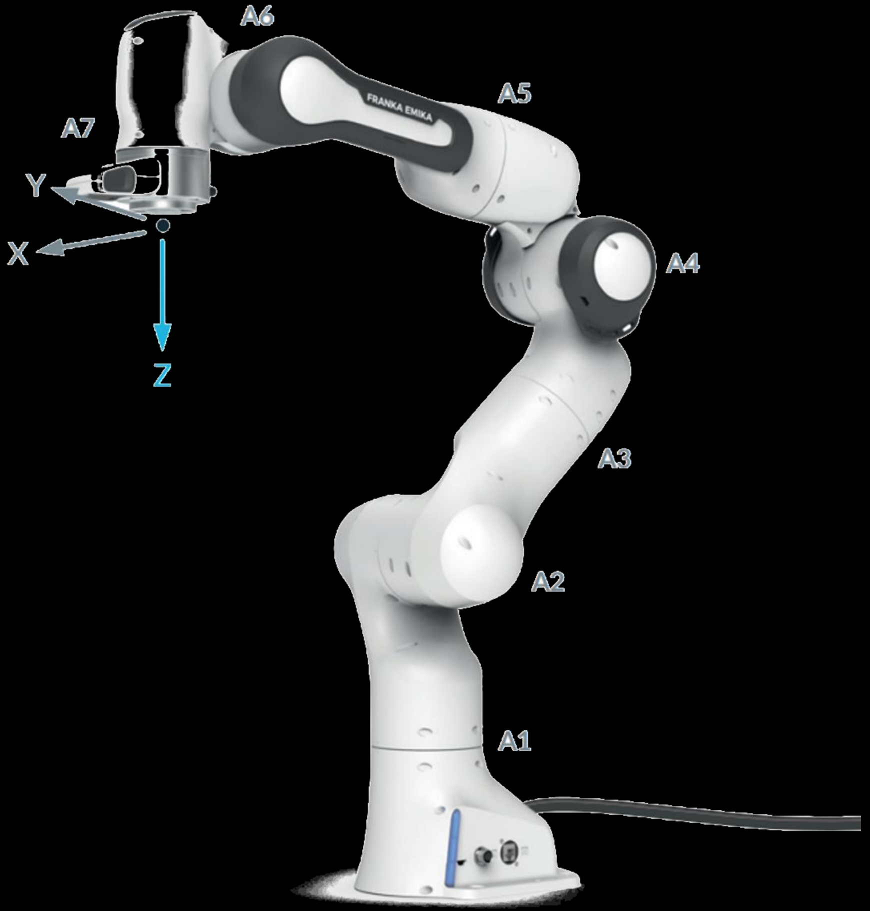
https://thehumanoid.ai/

ETH zürich

Part 3:
Kinematics and Dynamics for Hand Joints

SoftRobotics Laboratory

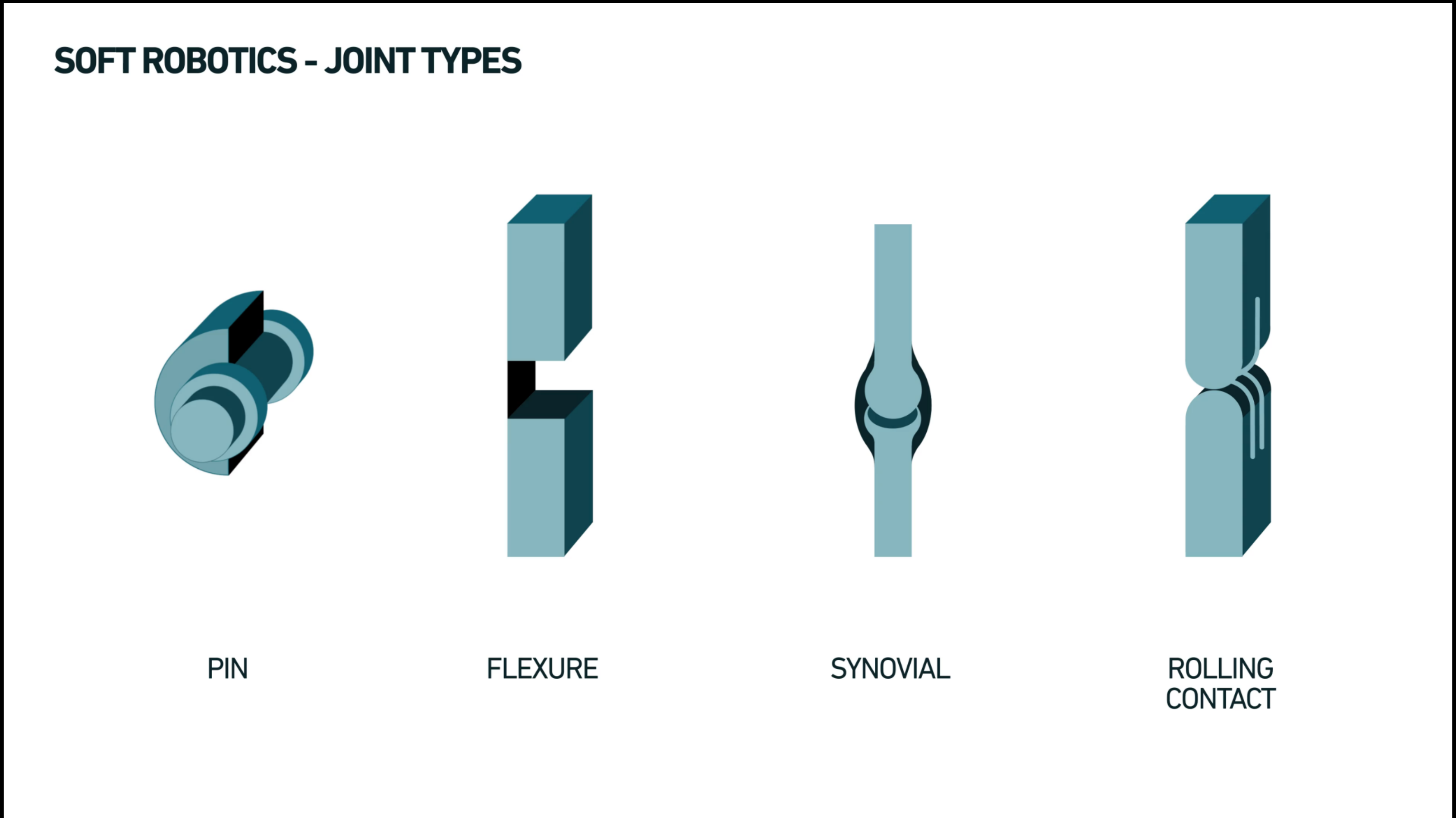
Difference Between Conventional Robots and Robotic Hands



franka.de

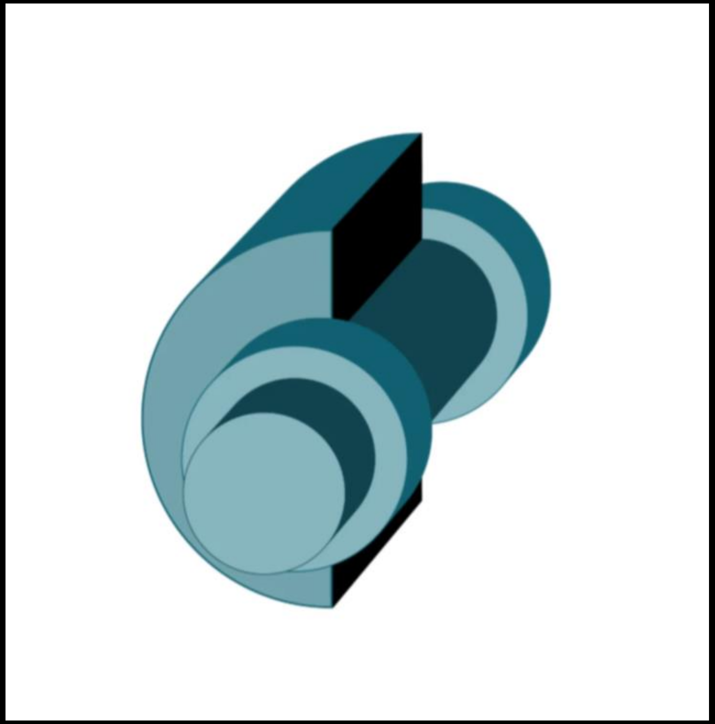
abb.com

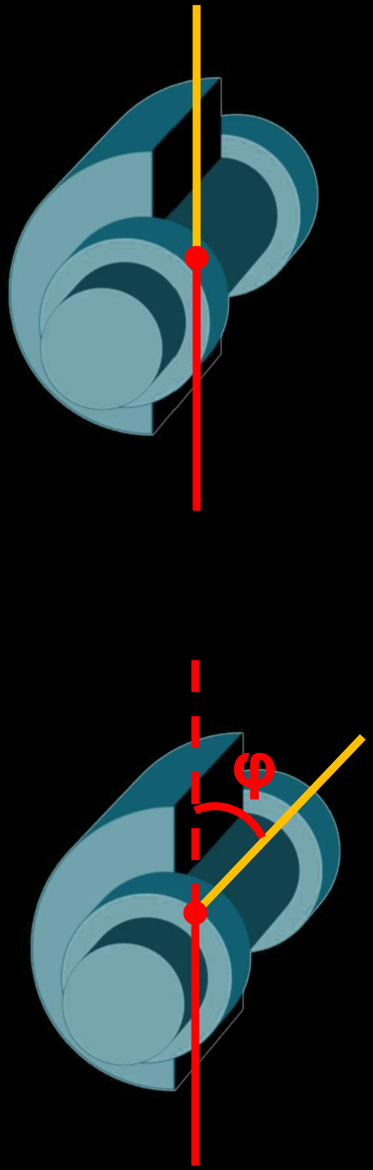
Recap: Different Types of Joints



Pin Joint

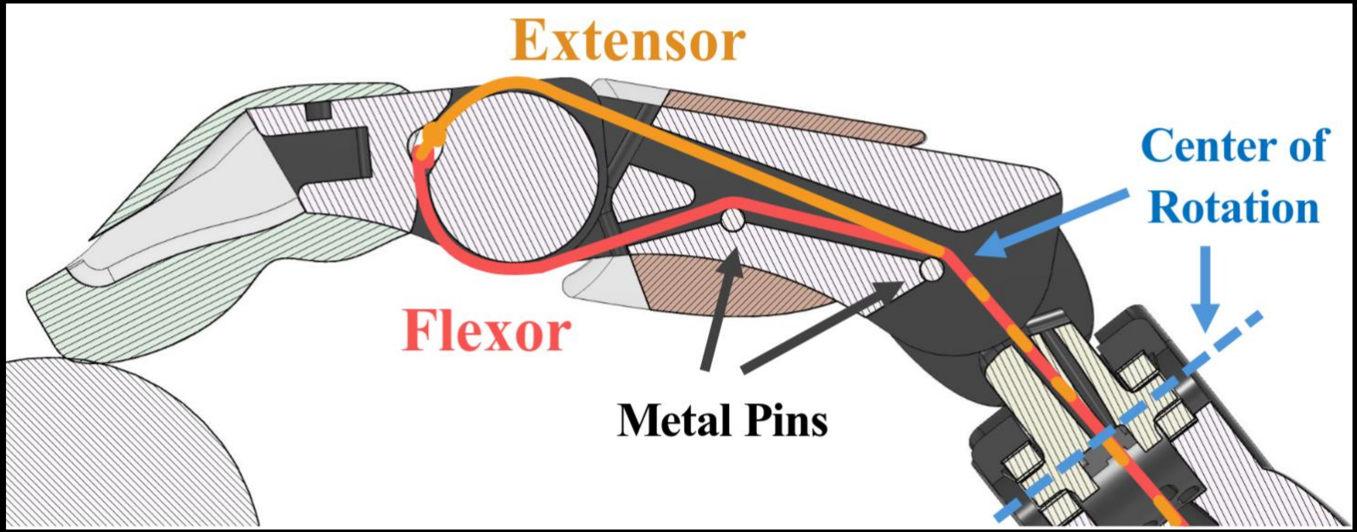
What the ORCA Hand uses



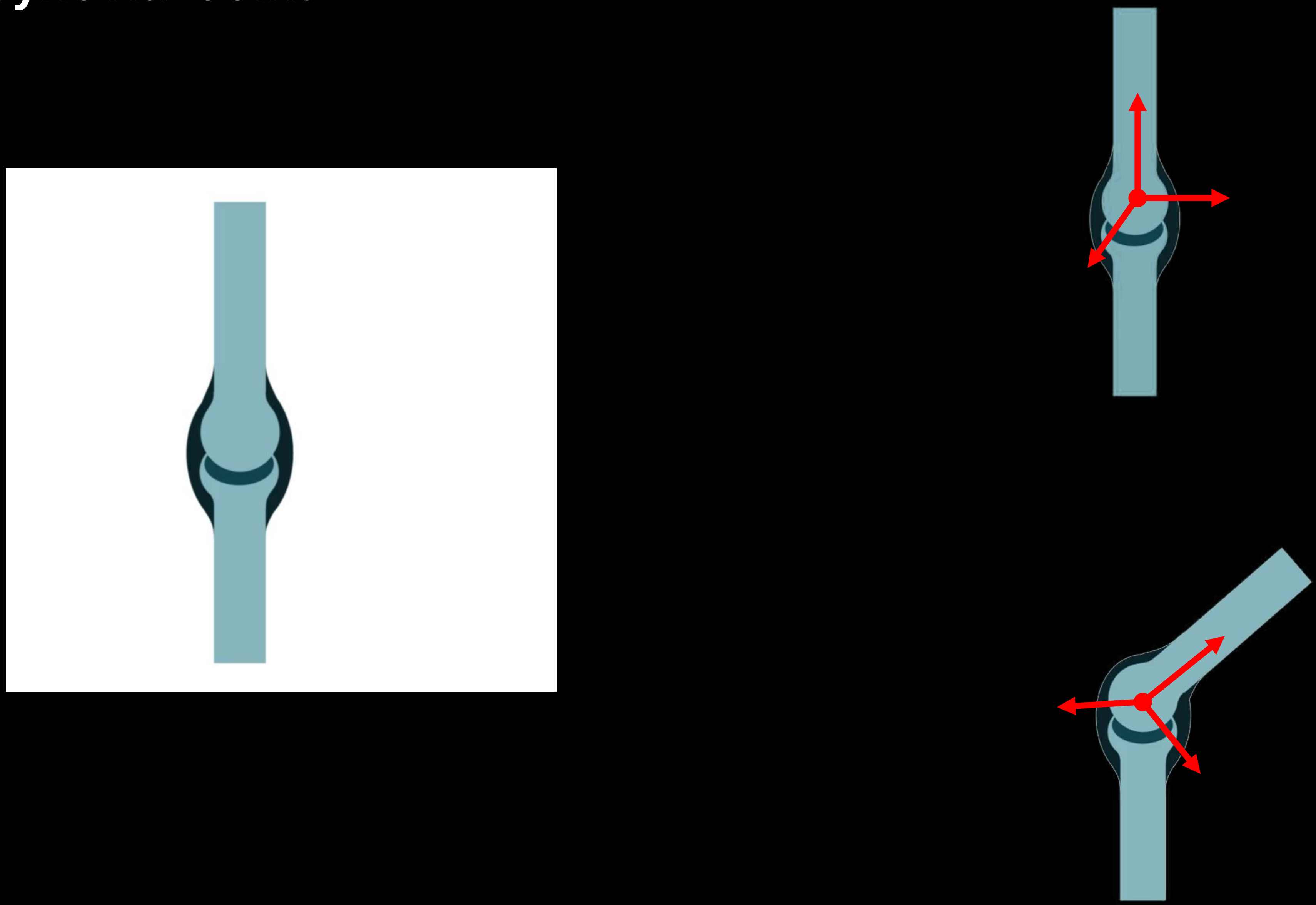


Pin Joint

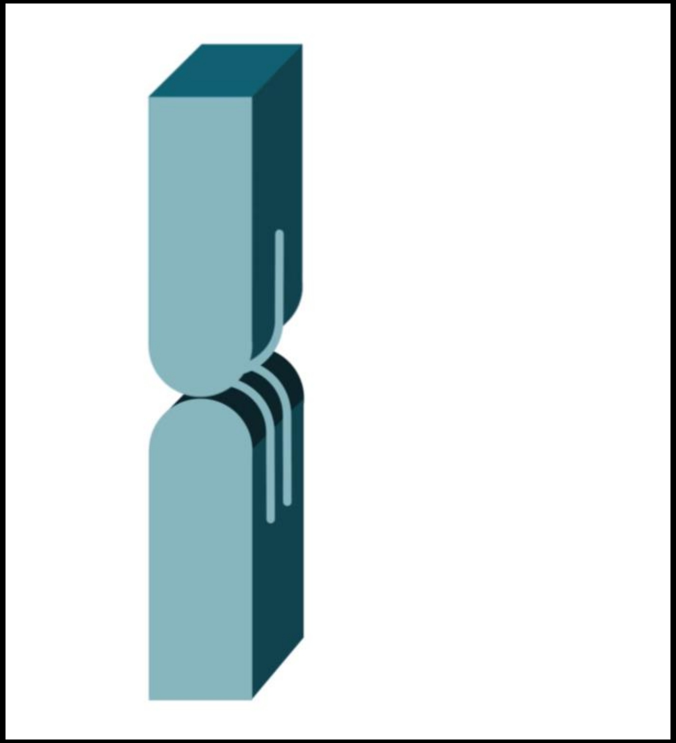
What the ORCA Hand uses

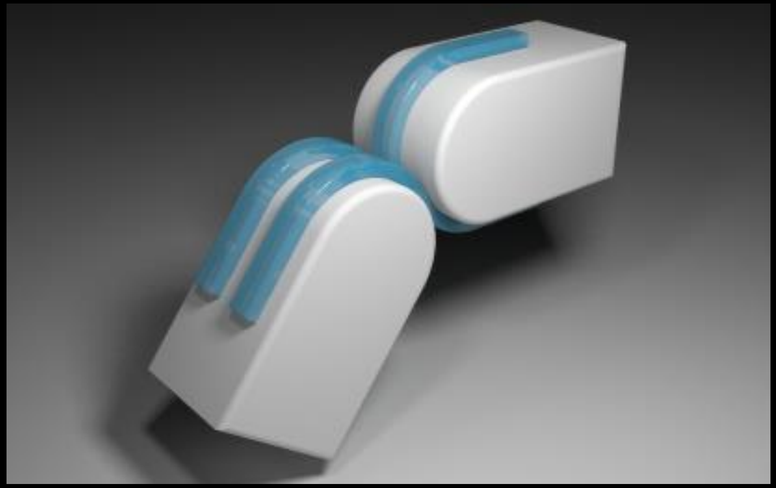


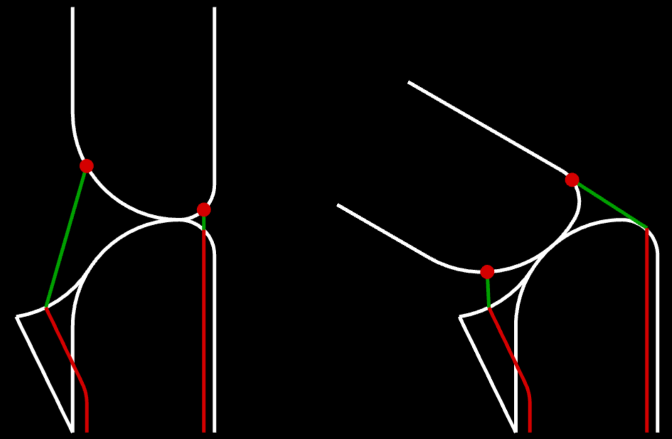
Synovial Joint



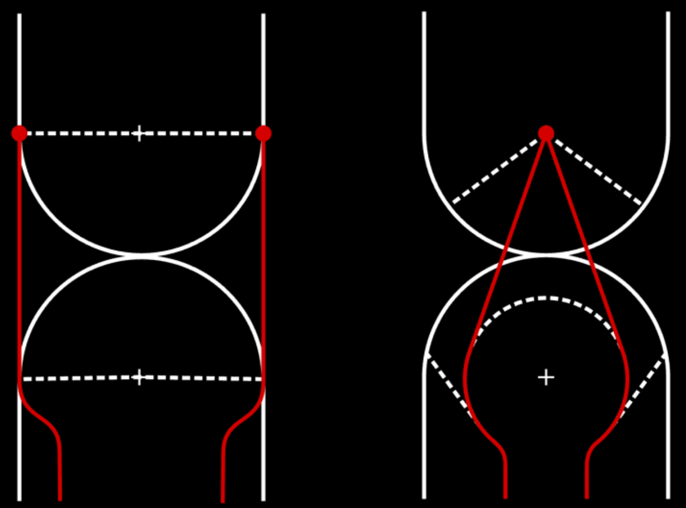
Rolling Contact Joint – Joints Used on Faive Hand

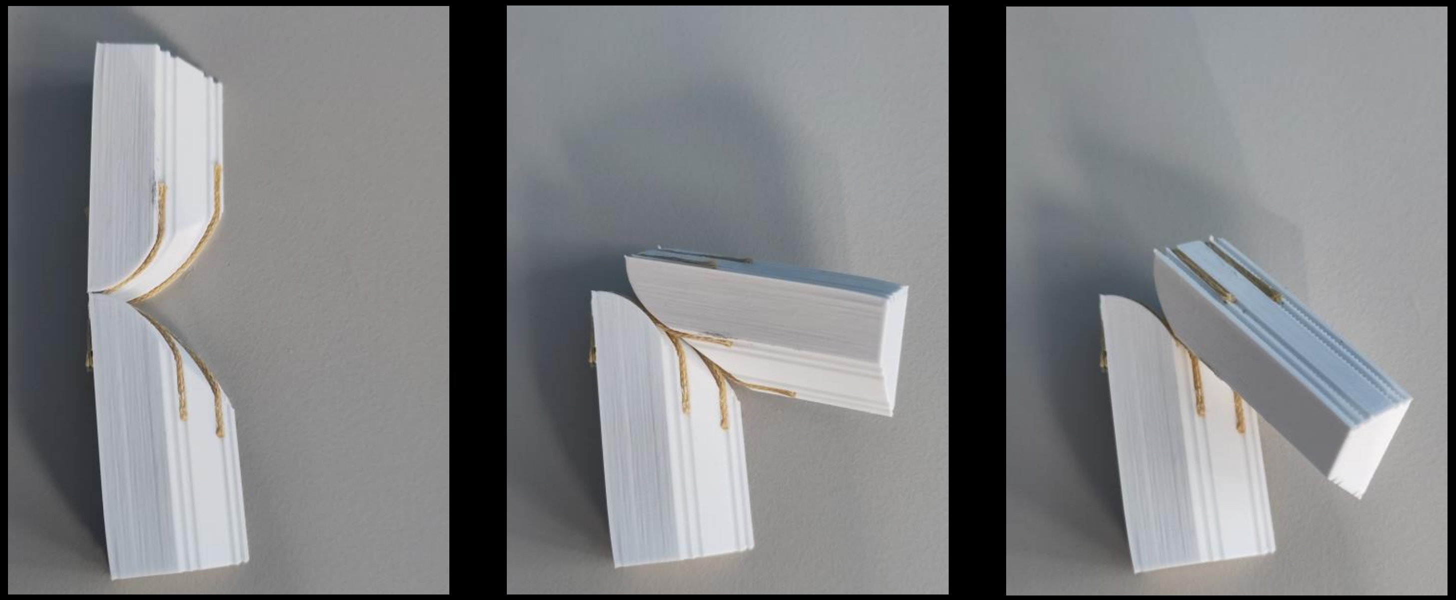




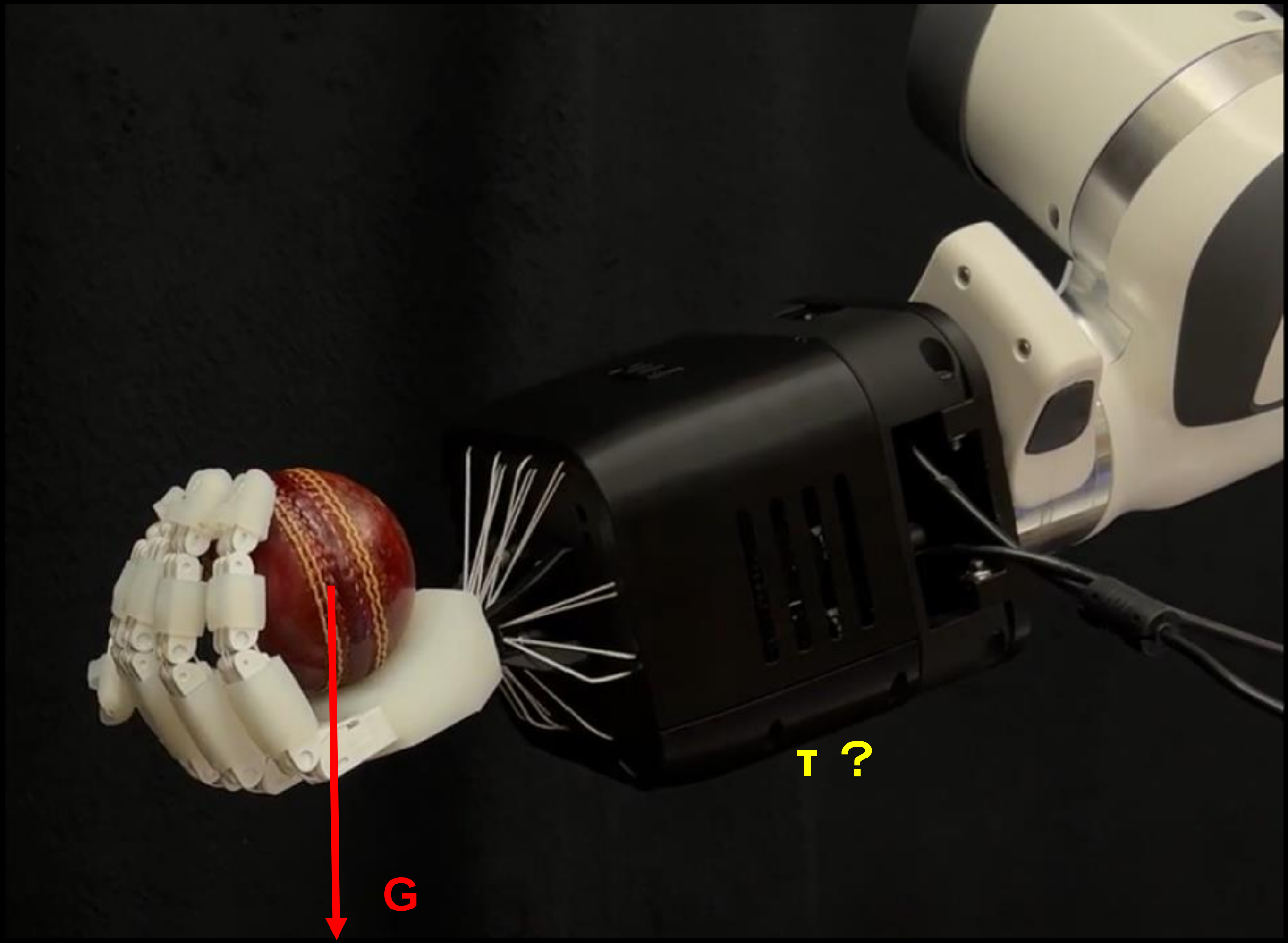


Kinematics for Rolling Contact Joint





Dynamics



Toshimitsu et al. (2023) https://srl-ethz.github.io/get-ball-rolling/

Dynamics

Tendon Lengths

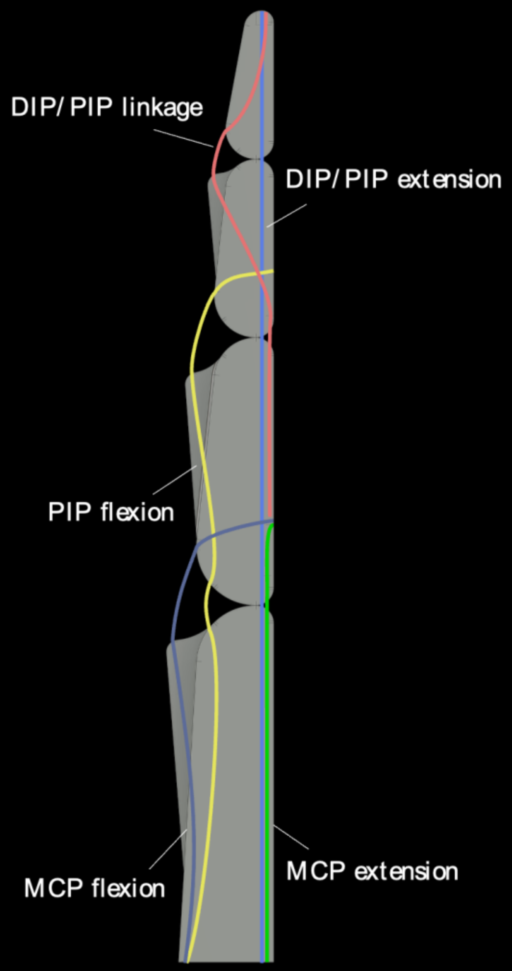
$$p = g(l) = g\big(f(q)\big) = F(q)$$

Motor Positions

Joint Angles

Dynamics – Jacobian

$$J_m = \begin{bmatrix} \frac{\partial p_1}{\partial q_1} & \frac{\partial p_1}{\partial q_2} \\ \frac{\partial p_2}{\partial q_1} & \frac{\partial p_2}{\partial q_2} \end{bmatrix}$$



ETH zürich SoftRobotics Laboratory

Dynamics

Velocity of the finger joints

$$\dot{p} = J_m \cdot \dot{q}$$

Velocity of the motors

$$\tau^T \cdot \dot{q} = T^T \cdot \dot{p}$$

Conservation of Power

$$\tau^T \cdot \dot{q} = T^T \cdot J_m \cdot \dot{q}$$

$$\tau = {J_m}^T \cdot T$$

Dynamics

Previous slide: $\tau = {J_m}^T \cdot T$

$$\dot{X}_{fingertip} = J_{fingertip} \cdot \dot{q}$$

$$\tau^T \cdot \dot{q} = {F_{fingertip}}^T \cdot \dot{X}_{fingertip}$$

$$\tau^T \cdot \dot{q} = {F_{fingertip}}^T \cdot J_{fingertip} \cdot \dot{q}$$

$$\tau = {J_{fingertip}}^T \cdot F_{fingertip}$$

Dynamics

$$\tau = {J_m}^T \cdot T$$

$$\tau = {J_{fingertip}}^T \cdot F_{fingertip}$$

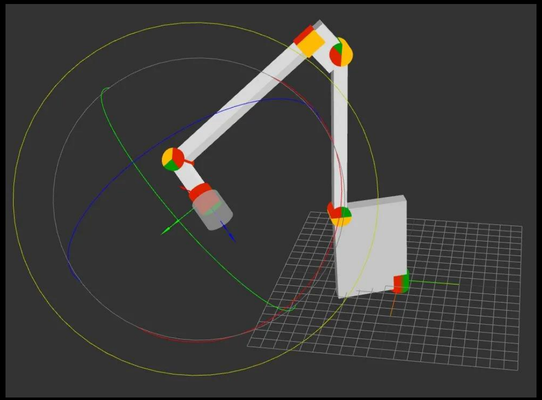
$$T = \left({J_m}^T\right)^{-1} \cdot {J_{fingertip}}^T \cdot F_{fingertip}$$

Summary of Kinematics and Dynamics

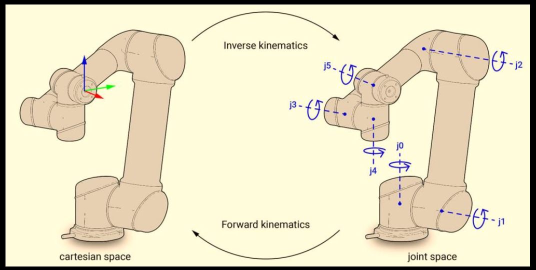
- Intro to Robot Kinematics and Dynamics
  - Representing points and lines in different coordinates and frames
  - Rotational matrix
  - Joint space and task space

- Forward and Inverse Kinematics
  - Homogeneous transformation matrix
  - Forward differential kinematics and Jacobian
  - Inverse kinematics

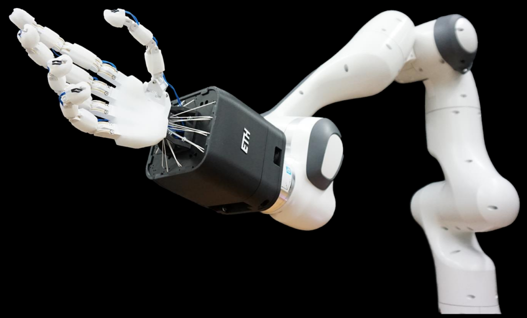
- Kinematics and Dynamics for hand joints
  - Hand Joints
  - Kinematics for rolling joints
  - Dynamics for rolling joints



Marginally Clever Robots



compas_fab



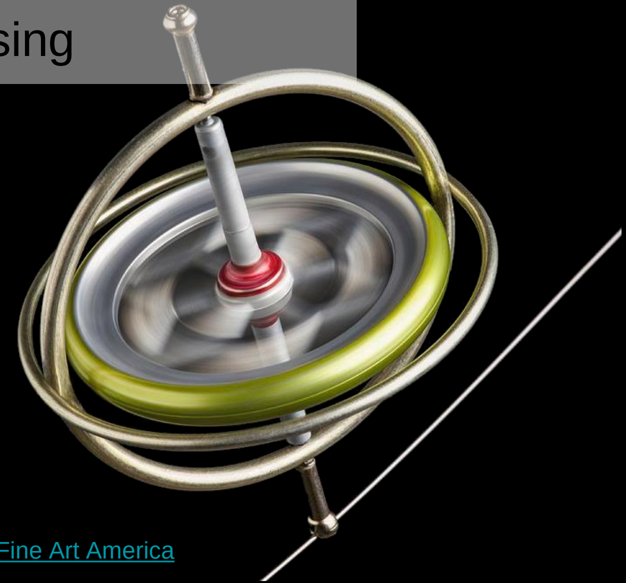
Faive Robotics

ETH zürich

SoftRobotics Laboratory

Implementing Control Strategies for Manipulation!

1. Sensing

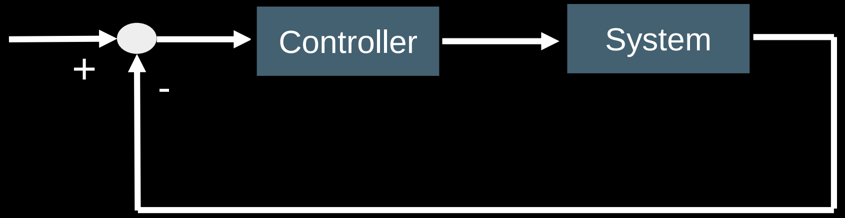


Fine Art America

Next Week

2. Control

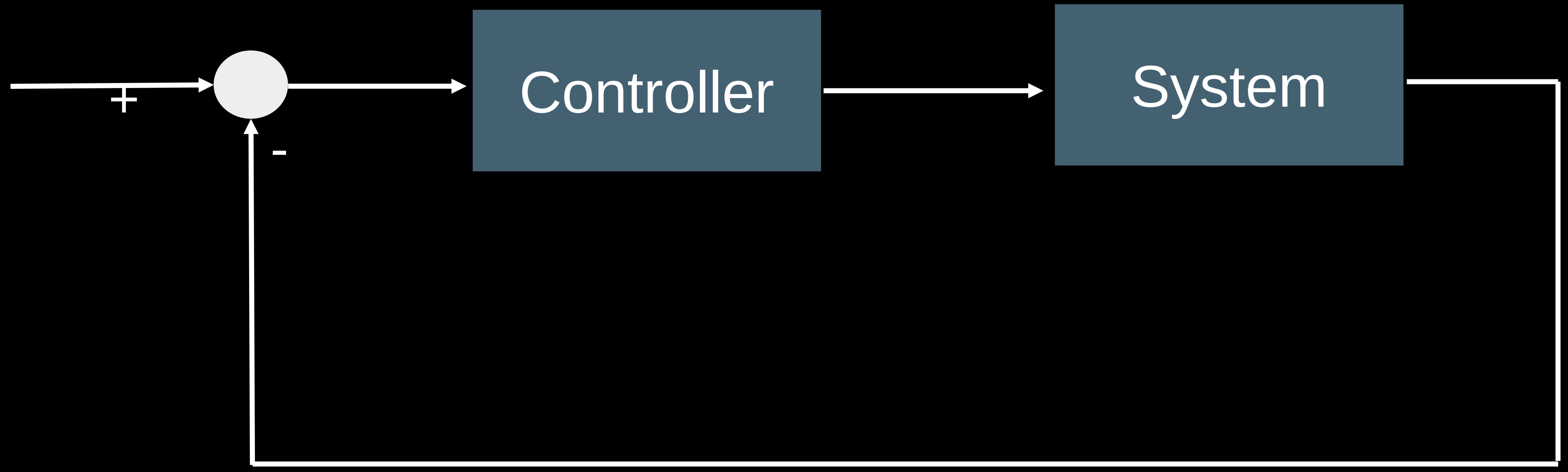
Wikimedia



ETH zürich

SoftRobotics Laboratory

ETH zürich

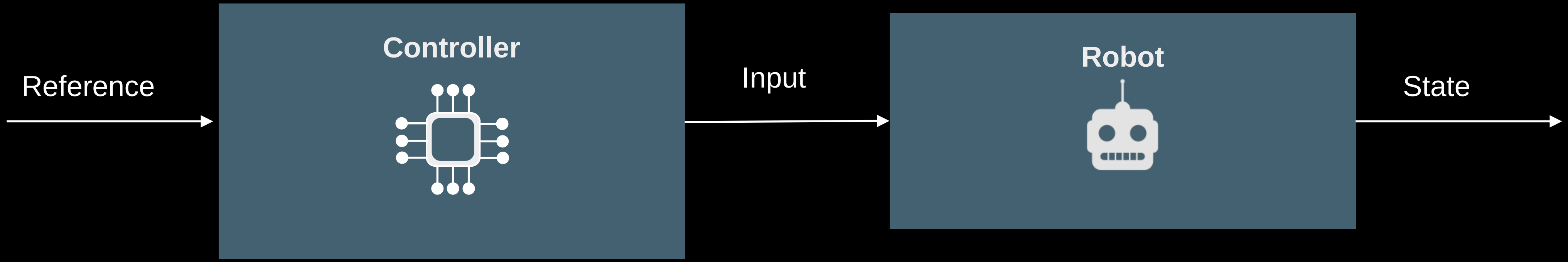


Part 4a: Feedback Control

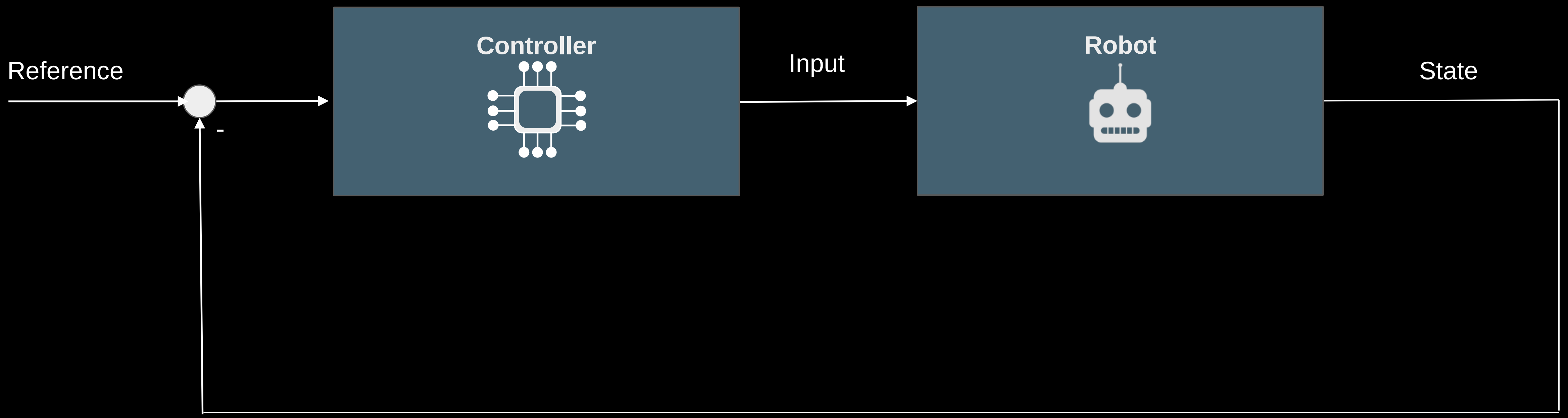
SoftRobotics Laboratory

Wikimedia

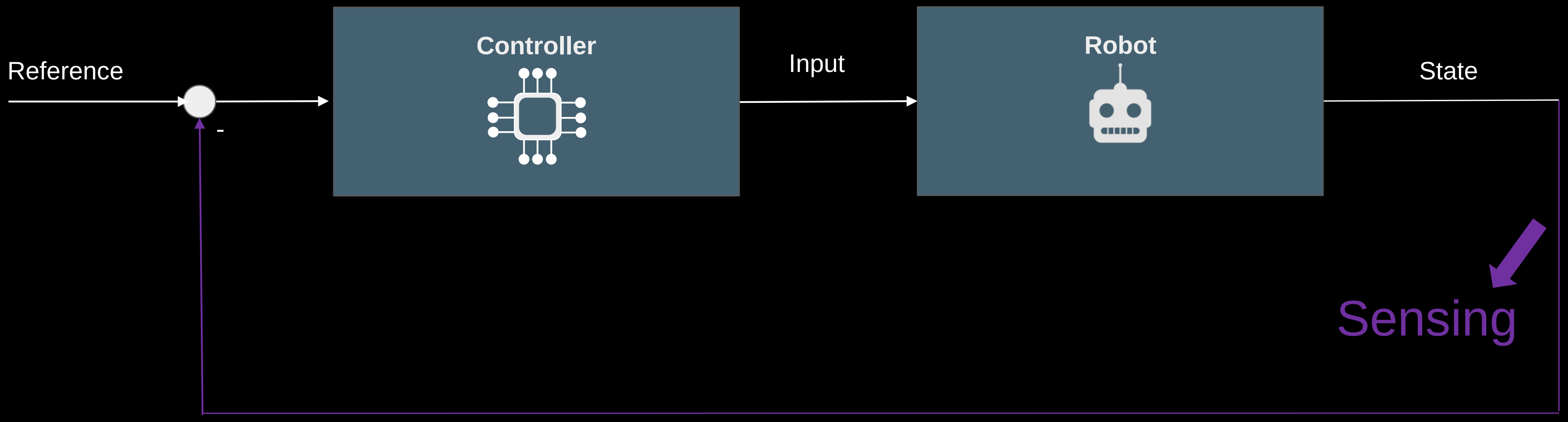
Simplest controller possible: Open loop



Closed Loop Controller



Closed Loop Controller

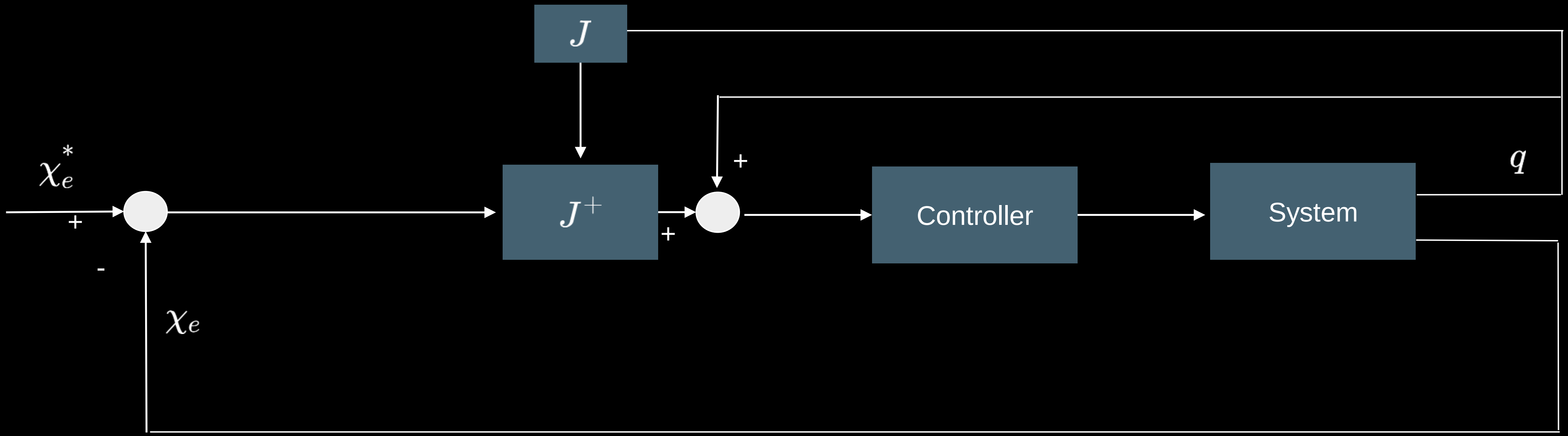


Remember - Inverse Kinematics

Numerical Inverse Kinematics (iterative approach):

$$q \leftarrow q + kJ_{eA}^{+}\Delta\chi_e \text{ with } k \in (0,1)$$

Inverse Kinematics Control



Trajectory Control

We can use a closed loop controller, but we need to add a component for the desired velocities

We define $$\Delta r_e^t = r_e^*(t) - r_e(q^t)$$

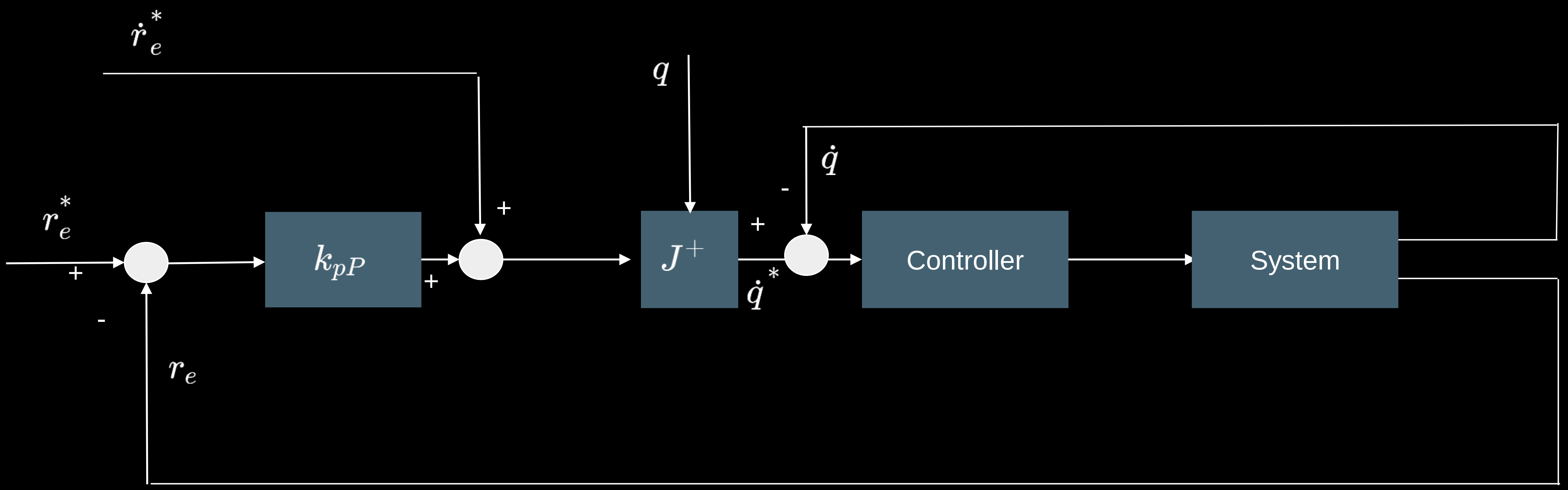
And the desired joint velocity $$\dot{q}^* = J_{e0_P}^+(q^t) \cdot (\dot{r}_e^*(t) + k_{pP} \Delta r_e^t)$$

If we have a desired rotation rate we write $$\dot{q}^* = J_{e0_R}^+(q^t) \cdot (\omega_e^*(t) + k_{pR} \Delta \phi)$$

Where ϕ are the angles used to represent the orientation of the end effector.

ETH zürich SoftRobotics Laboratory

Trajectory Control



ETH zürich SoftRobotics Laboratory

Dynamic control

The dynamic model is

$$M(q)\ddot{q} + b(q,\dot{q}) + g(q) = \tau + J_c(q)^T F_c$$

With:

$M(q)$: Generalized mass matrix

$q, \dot{q}, \ddot{q}$: Generalized position, velocity and acceleration vector

$b(q,\dot{q})$: Coriolis and centrifugal terms

$g(q)$: Gravitational terms

τ : External generalized forces

F_c : External Cartesian forces

$J_c(q)$: Geometric Jacobian corresponding to the external forces

Dynamic control

The dynamic model is

$$M(q)\ddot{q} + b(q,\dot{q}) + g(q) = \tau + J_c(q)^T F_c$$

If we know the desired generalized accelerations, velocities and poses we can write

$$\ddot{q}^* = k_p(q^* - q) + k_d(\dot{q}^* - \dot{q})$$

Thus the joint torques will be

$$\tau^* = M(q)\ddot{q}^* + b(q,\dot{q}) + g(q)$$

Task-space control

Remember that $J(q)\dot{q} = \chi_e = \begin{bmatrix} \dot{p}_e \\ w_e \end{bmatrix}$

If you derive that with respect to time: $\dot{\chi}_e = J(q)\ddot{q} + \dot{J}(q)\dot{q}$

And if we solve the dynamics equation for the joint acceleration and substitute in the equation above we get:

$$\dot{\chi}_e = JM^{-1}(\tau - b - g) + \dot{J}\,\dot{q}$$

Finally, remembering that $\tau = J_e^T F_e$

We can write $$\Lambda_e \dot{\chi}_e + \mu + p = F_e$$

$$\Lambda_e = (J_e M^{-1} J_e^T)^{-1}$$
$$\mu = \Lambda_e J_e M^{-1} b - \Lambda_e \dot{J}_e \dot{q}$$
$$p = \Lambda_e J_e M^{-1} g$$

Task-space control

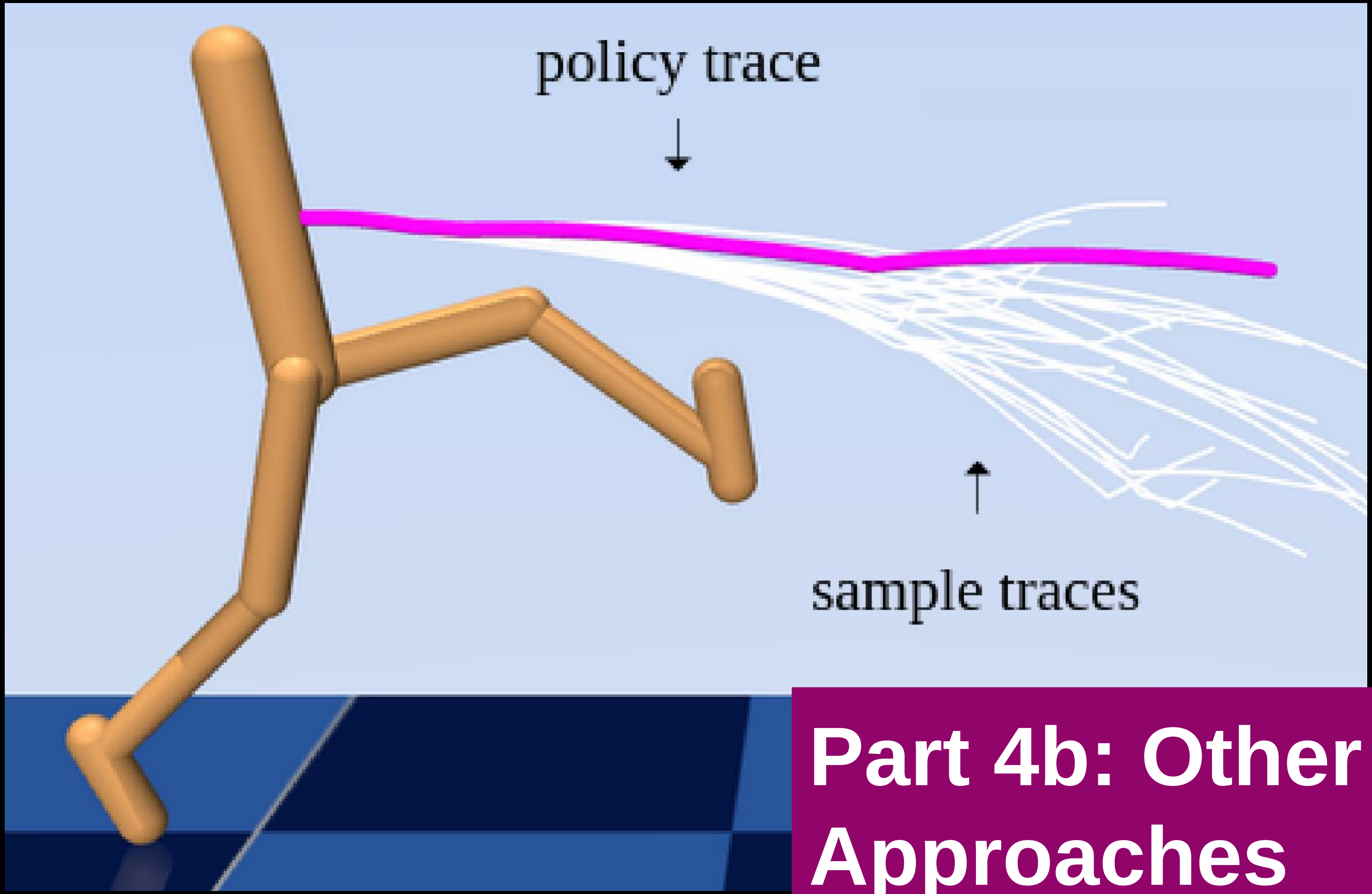
Defining the dynamics uniquely depending on the state of the end effector allows us to design a control loop

$$\boxed{\dot{\chi}_e^*} = \begin{pmatrix} r_e^* - r_e \\ \Delta\phi_e \end{pmatrix} + k_d(\chi_e^* - \chi_e)$$

Trajectory/Task space control → PID Control

- **Idea:** Compare desired vs actual output → **compute error** → apply correction (Proportional, Integral, Derivative).
- **Strengths:** Simple, cheap, widely used, doesn't require a full model.
- **Weaknesses:** Limited with nonlinear or high-dimensional systems; needs tuning; not predictive.

ETH zürich

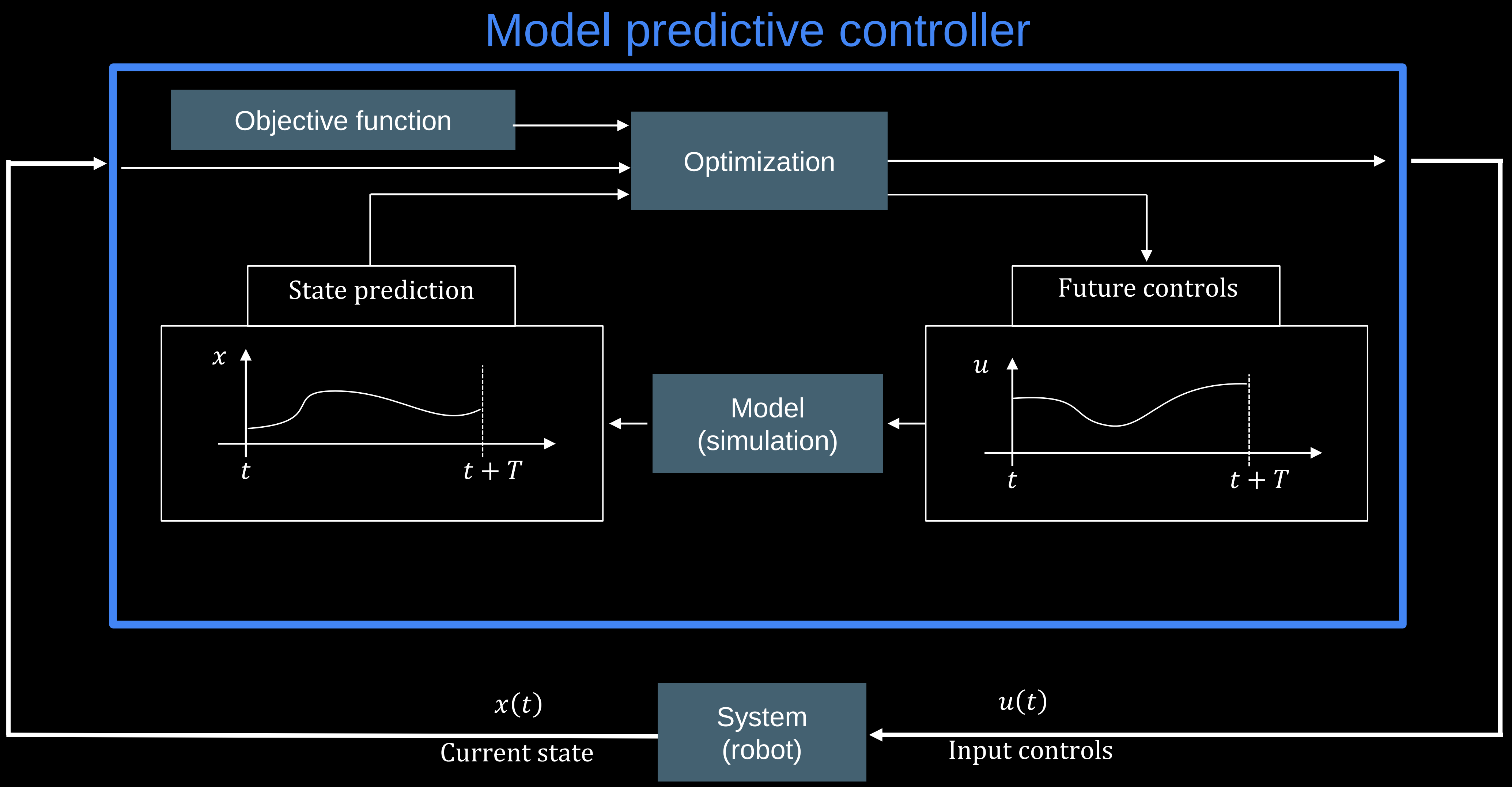


Mujoco PC

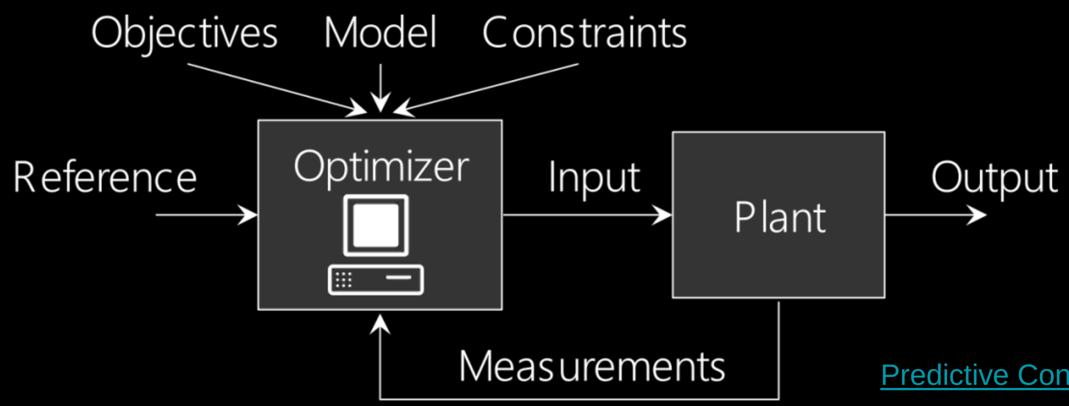
Part 4b: Other Control Approaches

SoftRobotics Laboratory

Model predictive control



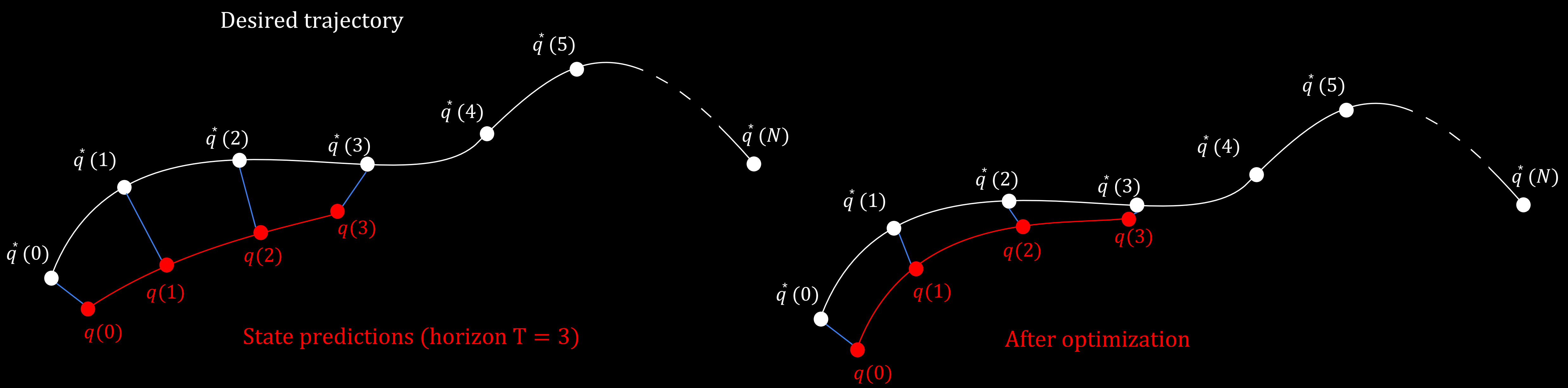
Model Predictive Control (MPC)



[Predictive Control Methods]

- **Idea:** Dynamics model to *predict future states* over a horizon → **optimize** control inputs to minimize a cost.
- **Strengths:** Handles constraints, anticipates the future.
- **Weaknesses:** Computationally expensive, requires an accurate model (sim-to-real gap).

Trajectory following with MPC



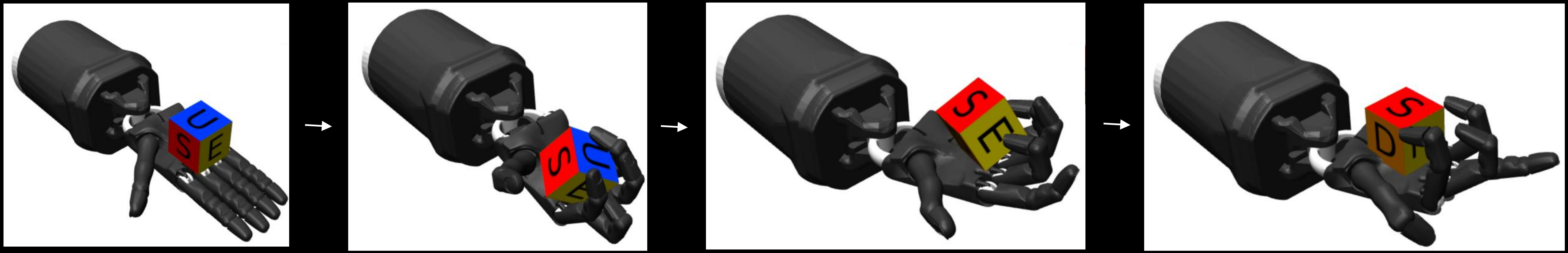
Objective $J = \sum_{t=0}^{T} c(t)$

where each step cost $c(t) = ||q^*(t) - q(t)||_2$

Cube reorientation with MPC

Goal orientation:

Howell et al. 2022, “Predictive Sampling: Real-time Behaviour Synthesis with MuJoCo”



System state $x(t)$ includes robot state $q(t)$, but also the object state.

Objective $J = \sum_{t=0}^{T} c(t)$

where $c(t) = ||cube\ orientation\ (t) - goal\ orientation||_2 + ||cube\ position\ (t) - palm\ center||_2$

Reinforcement Learning (RL) – Overview

Idea: Learn a policy (state → action mapping) by maximizing long-term reward through interaction.

- **Key characteristics:**
  - No explicit model required (model-free RL).
  - Works with nonlinear, high-dimensional dynamics.
- **Strengths:**
  - Captures "intelligent" behaviors.
  - Generalizes beyond what is explicitly modeled.
- **Weaknesses:**
  - Data hungry → needs simulation or many real-world trials (hard and expensive).
  - Transfer from sim to real can be hard (sim-to-real gap).

RL will be covered in **two weeks**.

Feedback Control vs MPC

Feedback control

MPC

Feedback Control vs MPC

Feedback control

- Computationally cheap.

MPC

- Expensive.

Feedback Control vs MPC

Feedback control

- Computationally cheap.
- Reacts to immediate residual.

MPC

- Expensive.
- Longer horizon. But still myopic after horizon T.

Feedback Control vs MPC

Feedback control

- Computationally cheap.
- Reacts to immediate residual.
- Doesn’t require a model.

MPC

- Expensive.
- Longer horizon. But still myopic after horizon T.
- Requires a computational model.
  - Sim2Real gap.

Feedback Control vs MPC

Feedback control

- Computationally cheap.
- Reacts to immediate residual.
- Doesn't require a model.
- Limited to regulation/tracking.

MPC

- Expensive.
- Longer horizon. But still myopic after horizon T.
- Requires a computational model.
  - Sim2Real gap.
- Can encode higher-level tasks.

MPC vs Reinforcement Learning

MPC

Reinforcement Learning

MPC vs Reinforcement Learning

MPC

- No offline training.

Reinforcement Learning

- Offline training needed.

MPC vs Reinforcement Learning

MPC

- No offline training.
- Requires a model.

Reinforcement Learning

- Offline training needed.
- Does not require a model.

MPC vs Reinforcement Learning

MPC

- No offline training.
- Requires a model.
- Limited to our state representations.

Reinforcement Learning

- Offline training needed.
- Does not require a model.
- Can discover latent representations, and "intelligent" behavior.

MPC vs Reinforcement Learning

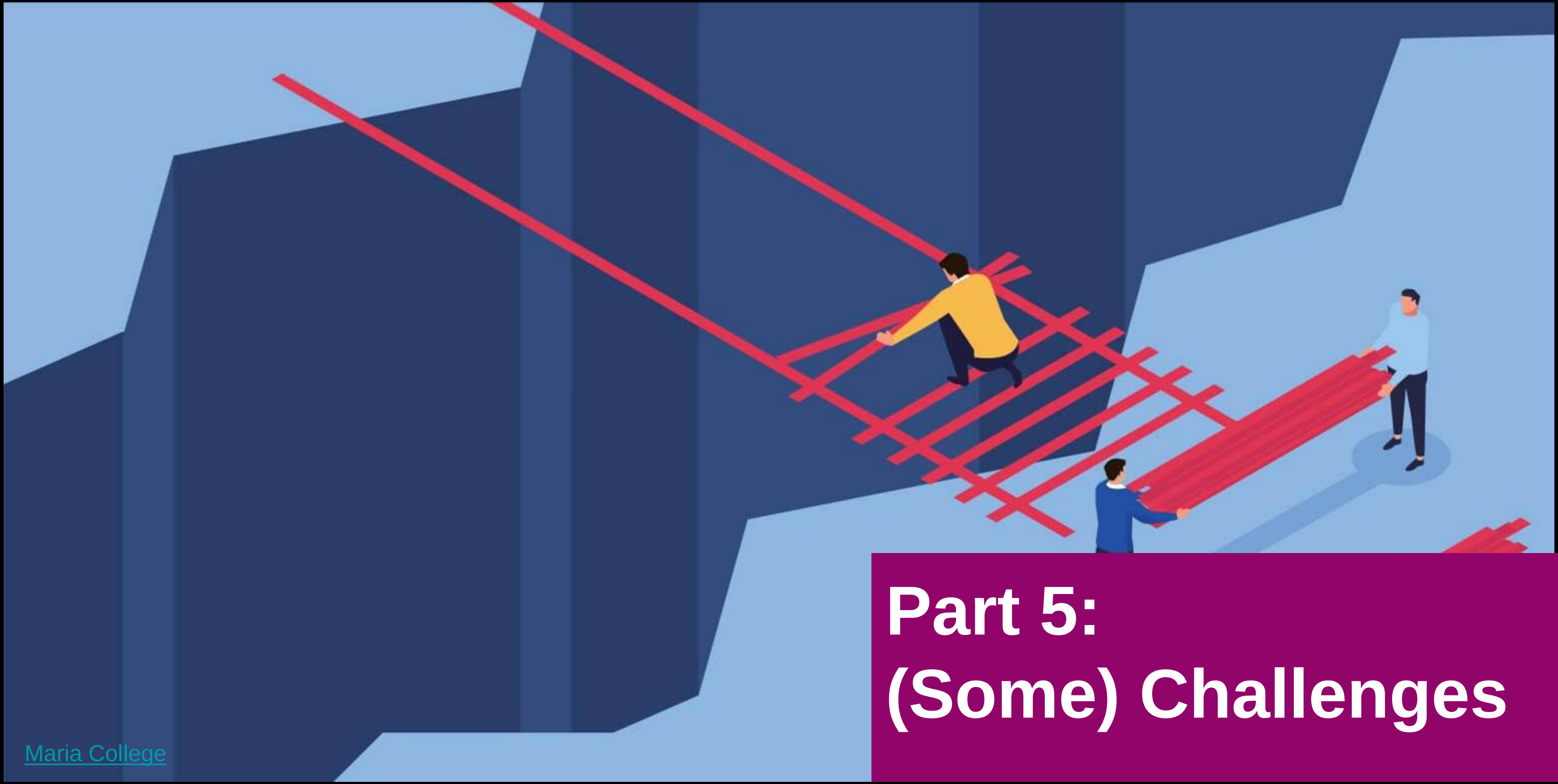
MPC

- No offline training.
- Requires a model.
- Limited to our state representations.
- Slower during execution.

Reinforcement Learning

- Offline training needed.
- Does not require a model.
- Can discover latent representations, and "intelligent" behavior.
- Learns a policy, a direct mapping from state to action.

ETH zürich



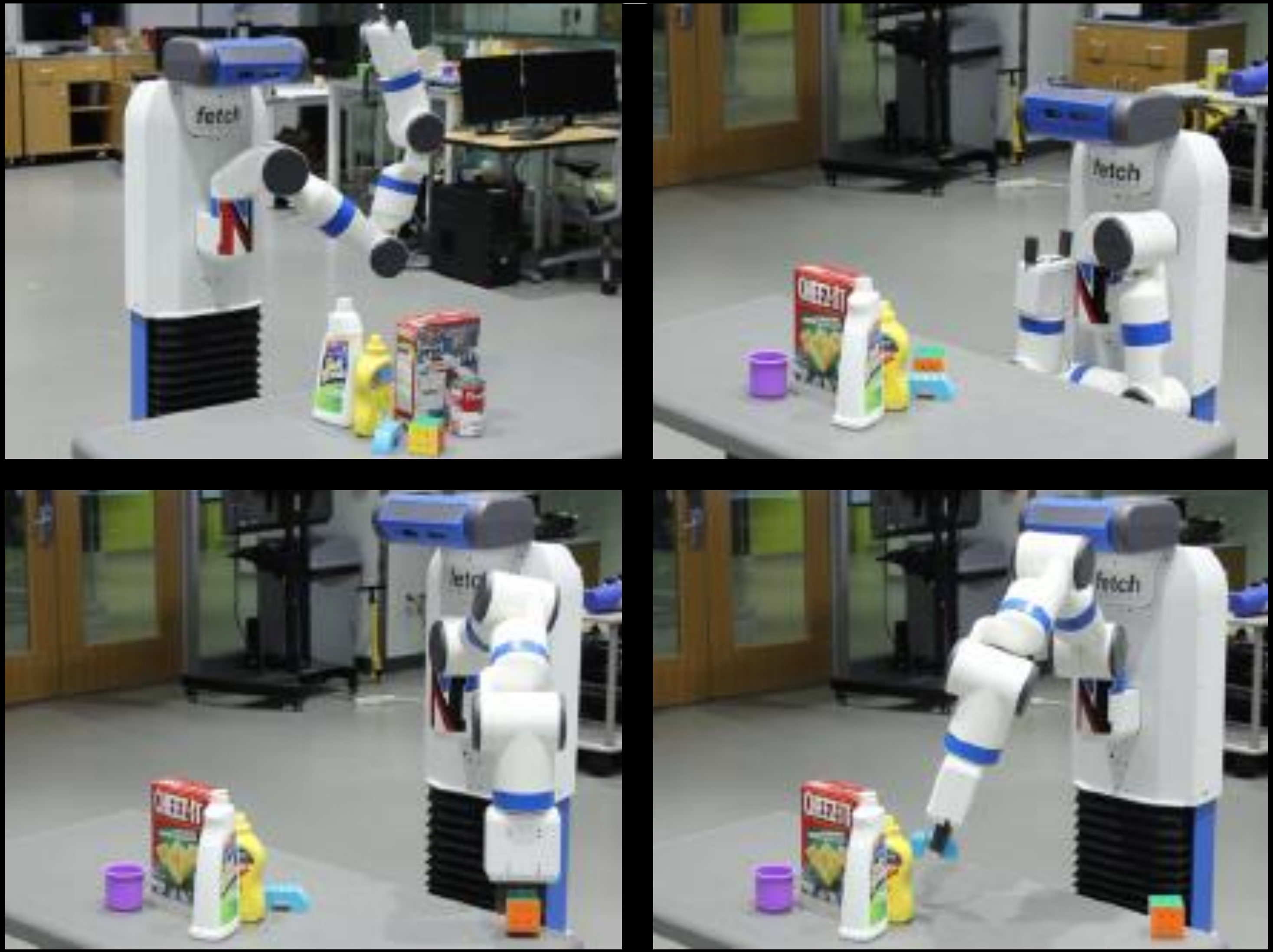
Part 5:
(Some) Challenges

SoftRobotics Laboratory

What should you expect?

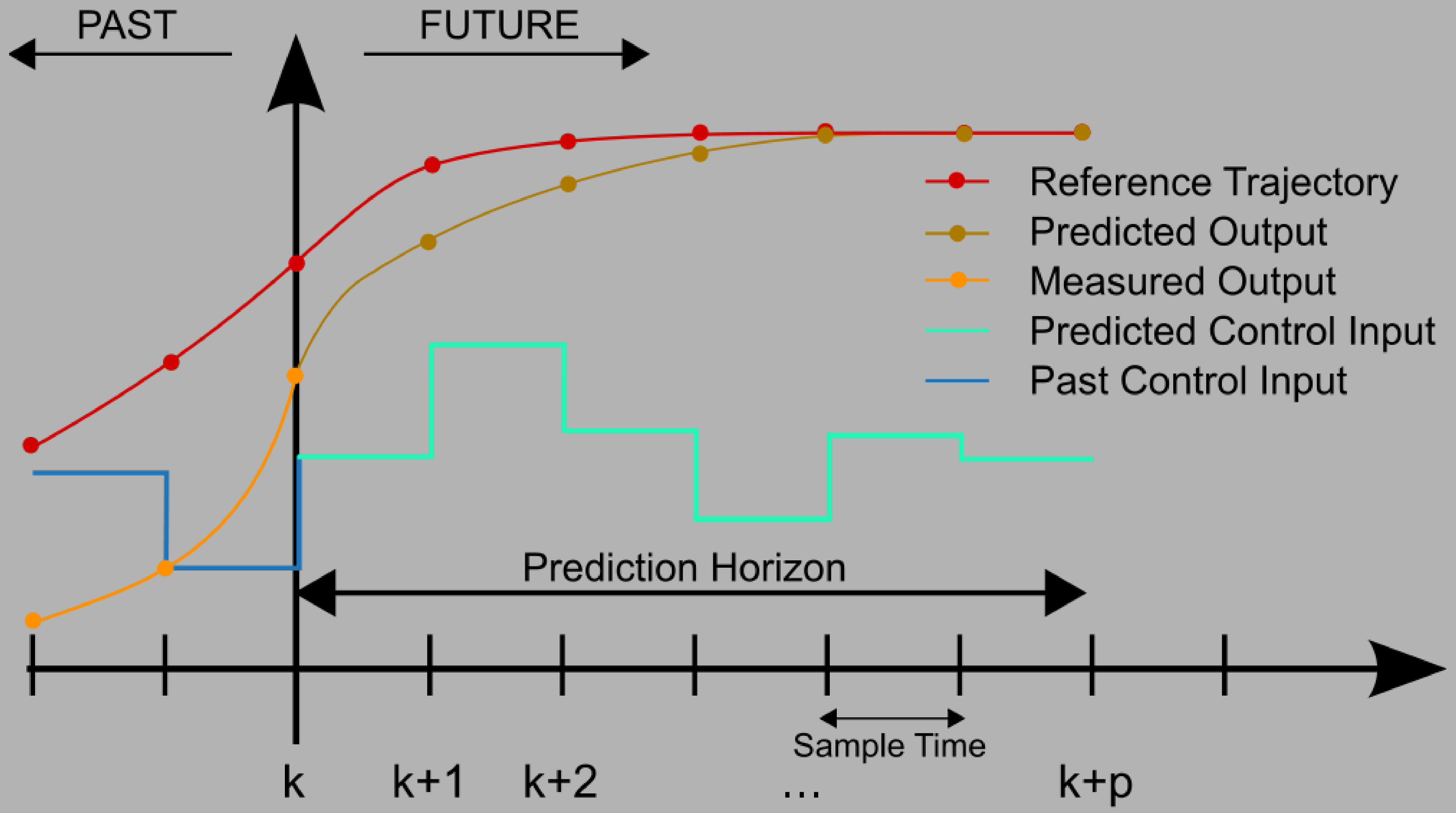
- Uncertainty and Partial Observability
- Long Horizon
- Under/Over actuation
- Sim-to-real gap
- Tendon strain + skin non-linearity
- Encoder's sensibility

Uncertainty and Partial Observability



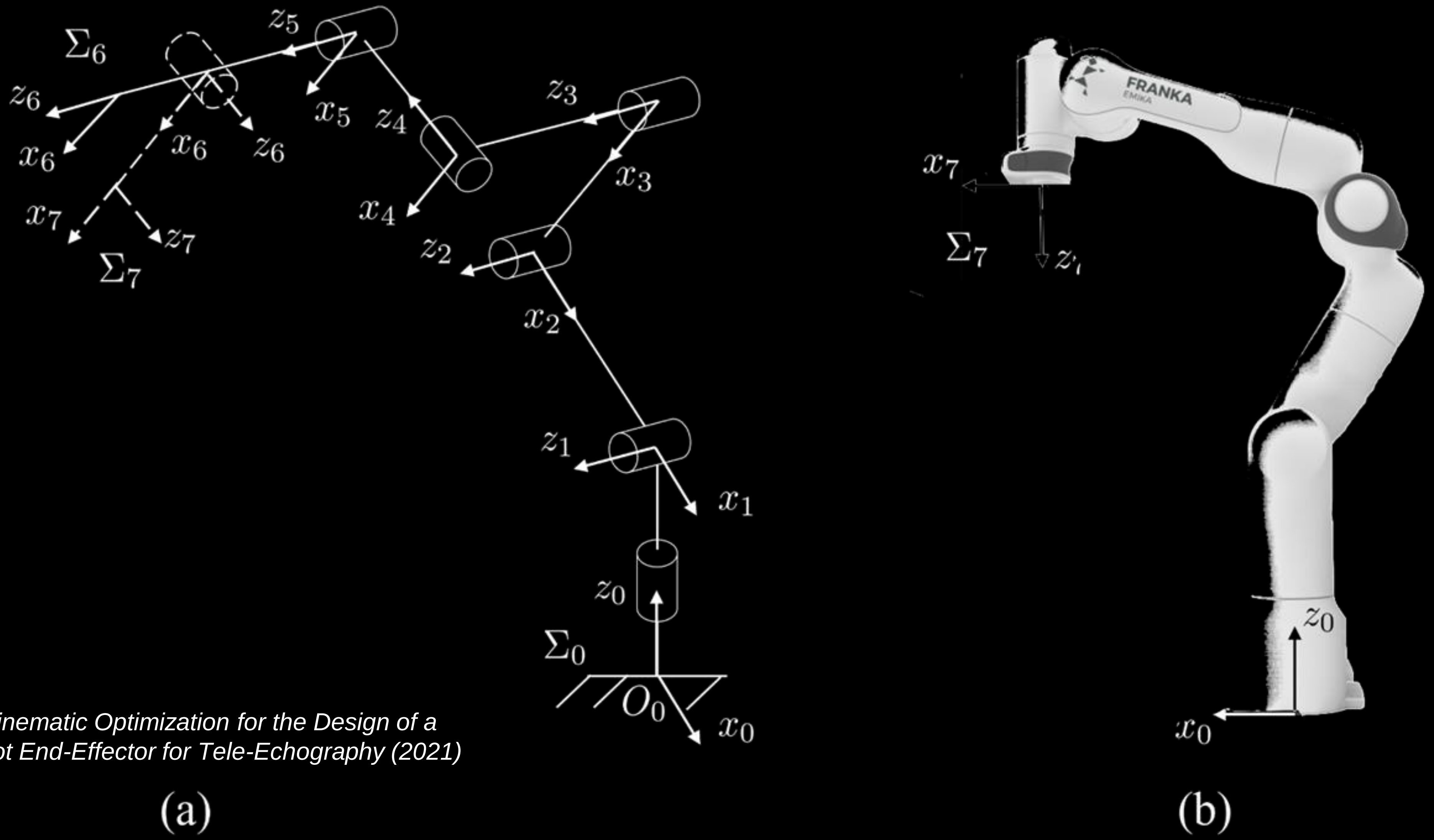
[Yuchen Xiao, Sammie Katt, Andreas ten Pas, Shengjian Chen, Christopher Amato.](#)
[**Online Planning for Target Object Search in Clutter under Partial Observability**.](#)
[IEEE International Conference on Robotics and Automation (ICRA), Montreal, Canada, May 2019.](#)

Long Horizon



[MPC and constrained systems , TU Eindhoven]

Underactuation and Overactuation

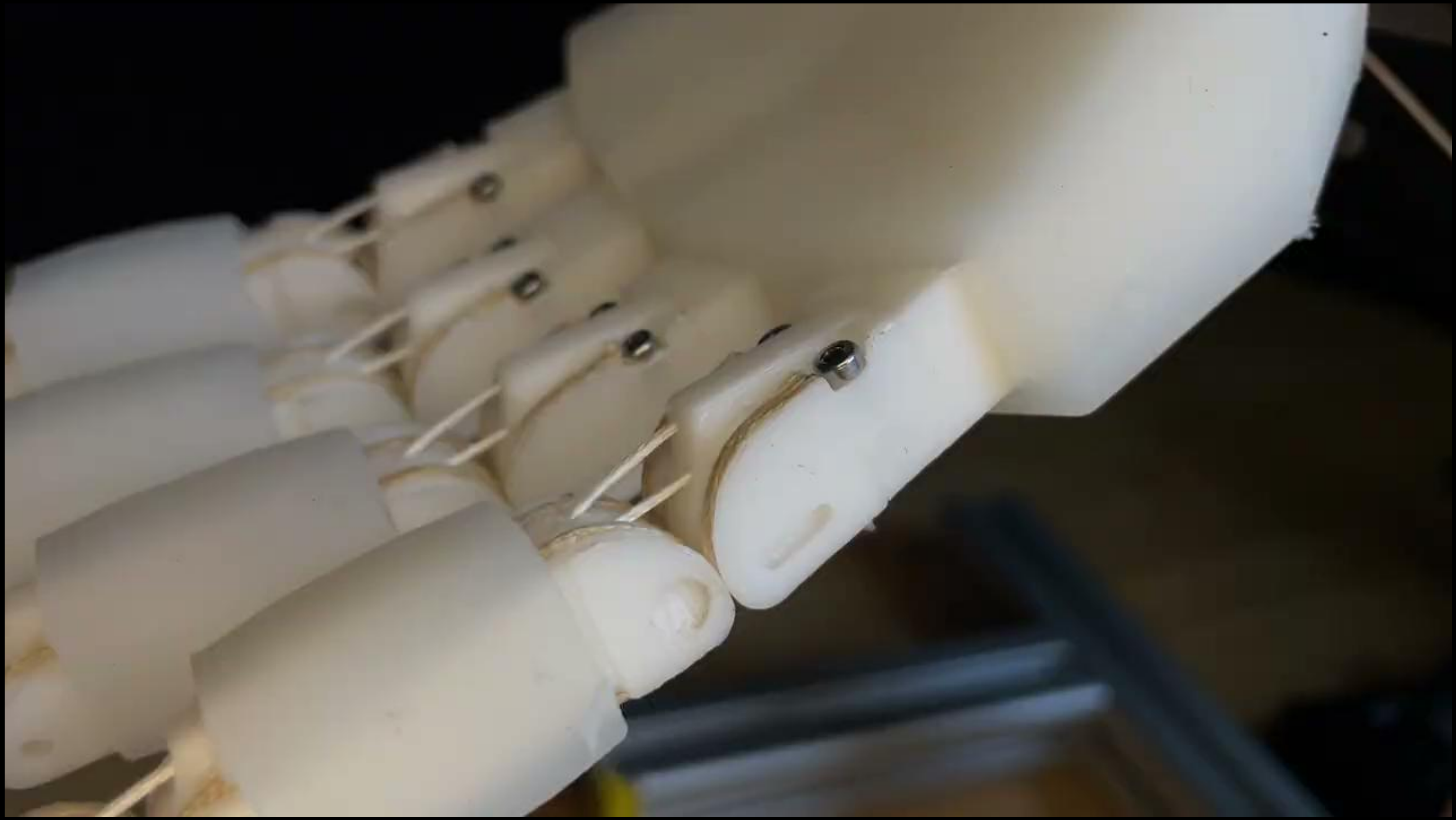


Filippeschi et al. *Kinematic Optimization for the Design of a Collaborative Robot End-Effector for Tele-Echography (2021)*

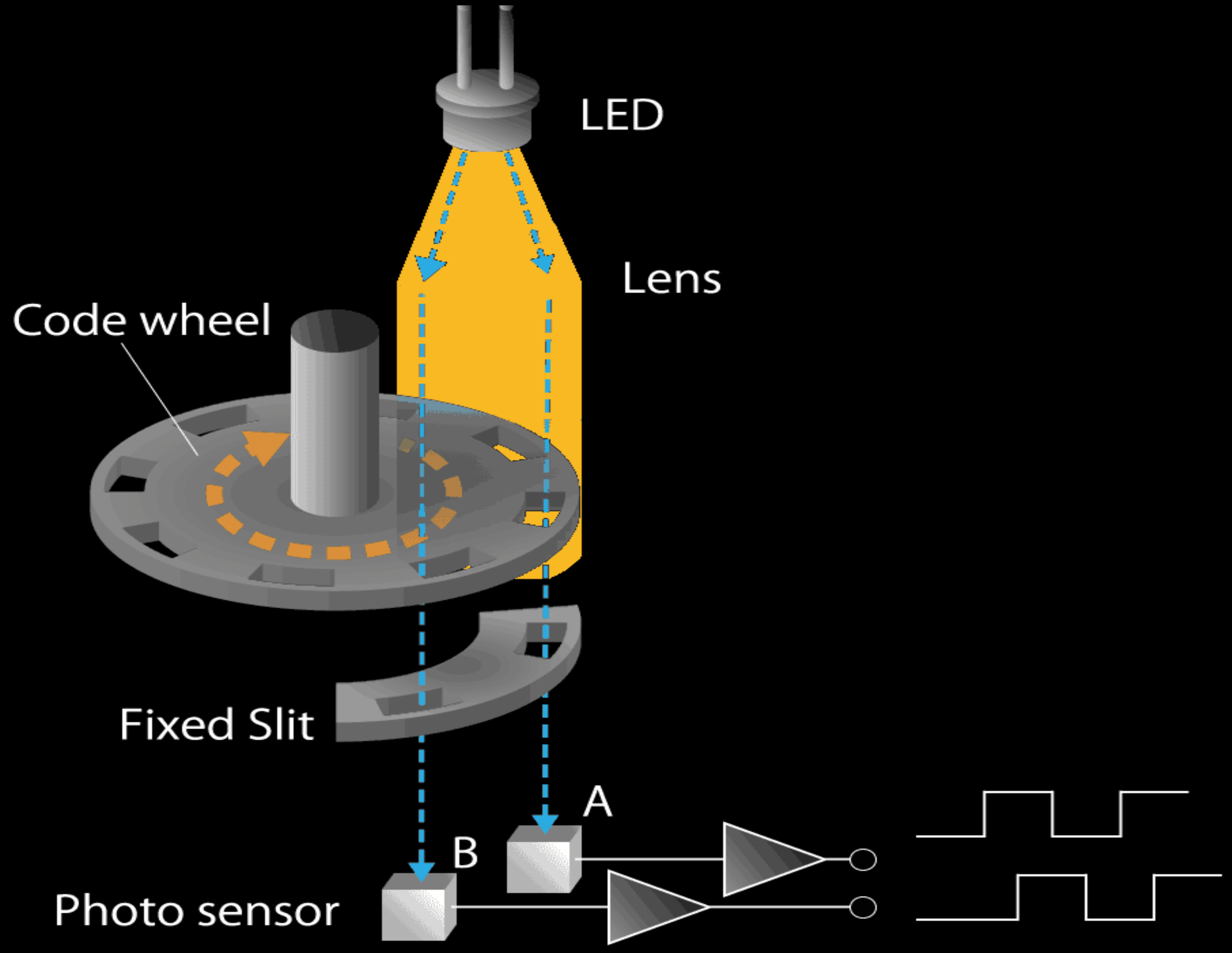
Sim-to-real gap

Everyday Robots

Tendon strain + skin non-linearity



Encoder's sensibility



Asahi Kasei Microdevices

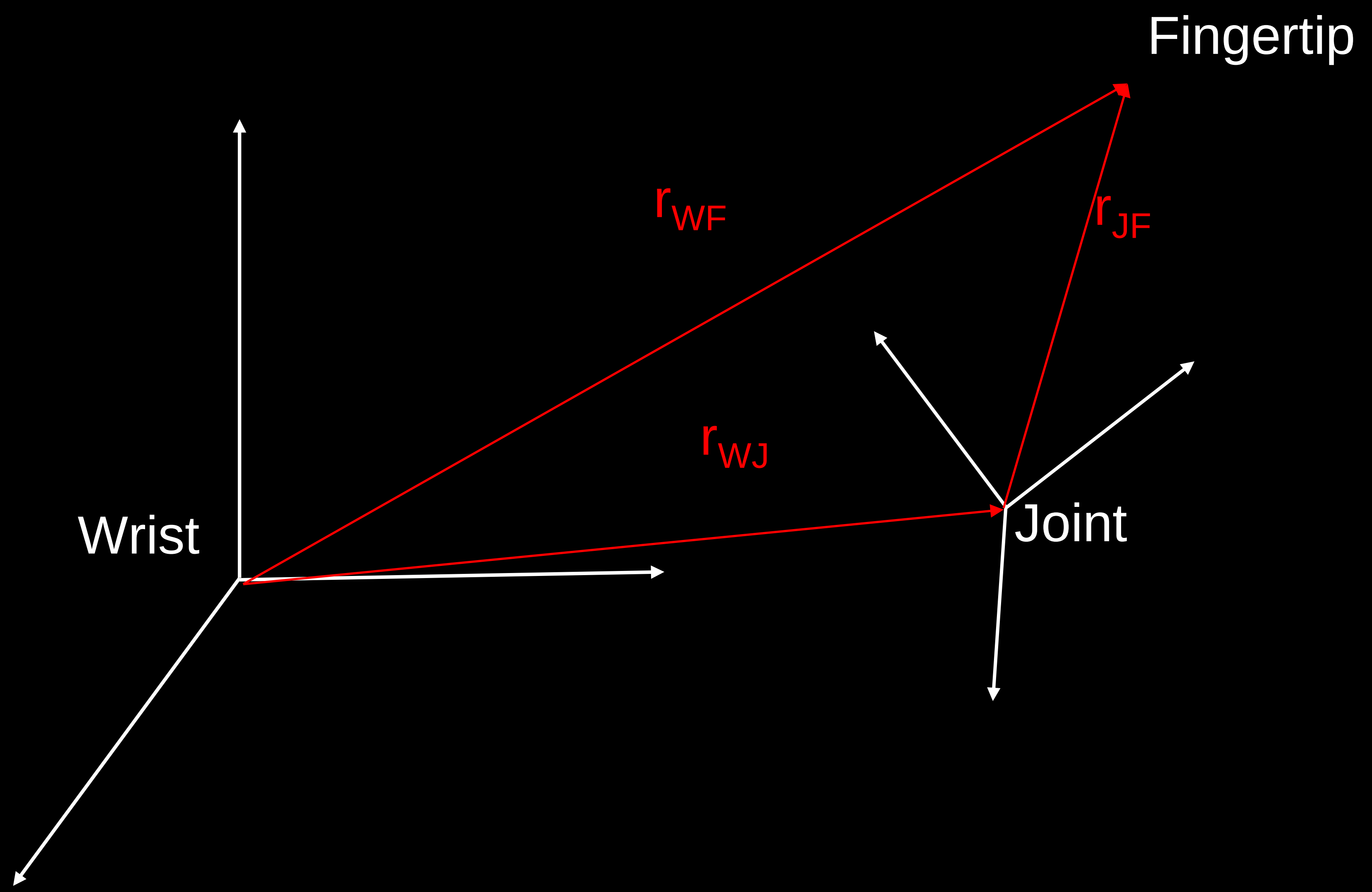
Backup Slides

Sensing the pose: two methods

- Direct methods: Direct reference to the world reference frame
  - The sensors obtain the absolute value of the state we are measuring

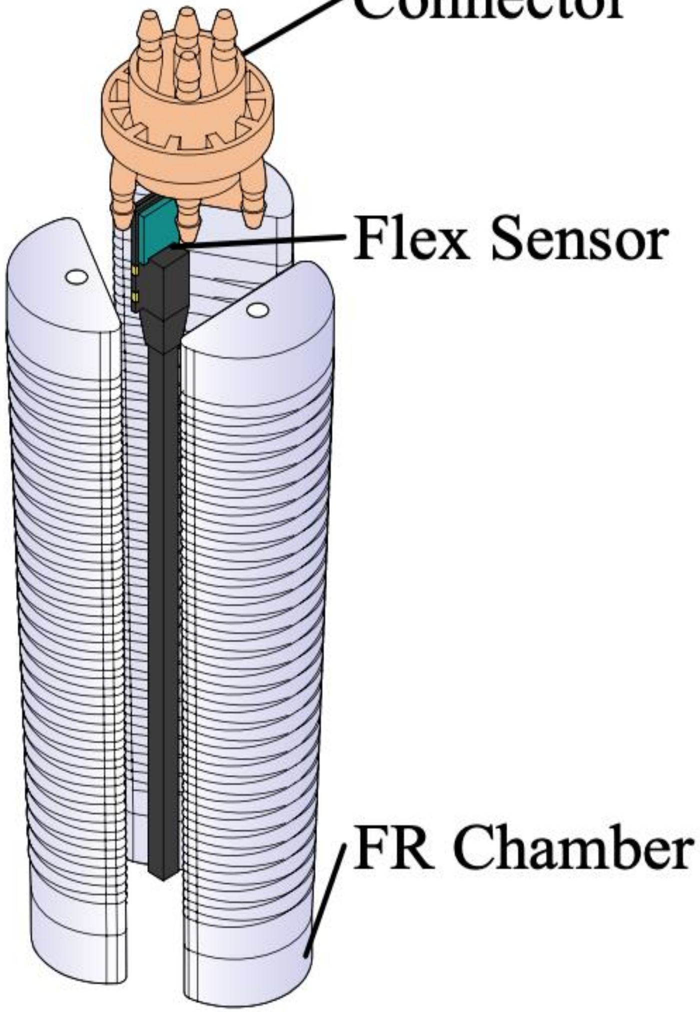
- Indirect methods: Obtain a measurement with reference to a second frame
  - The sensors will estimate a relative measurement that can be transformed into an absolute measurement

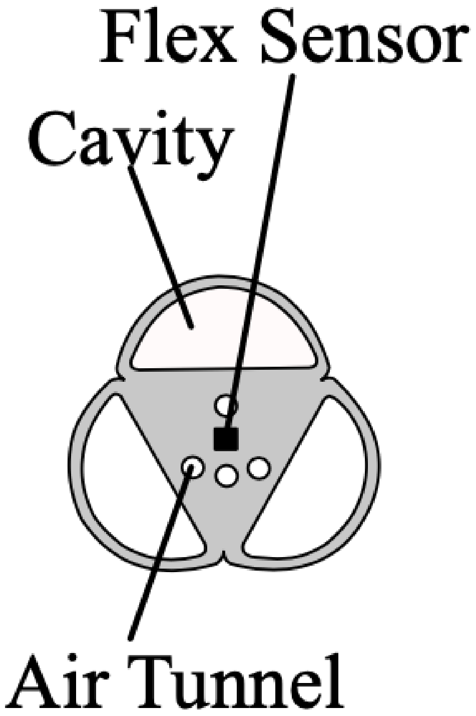
Second solution

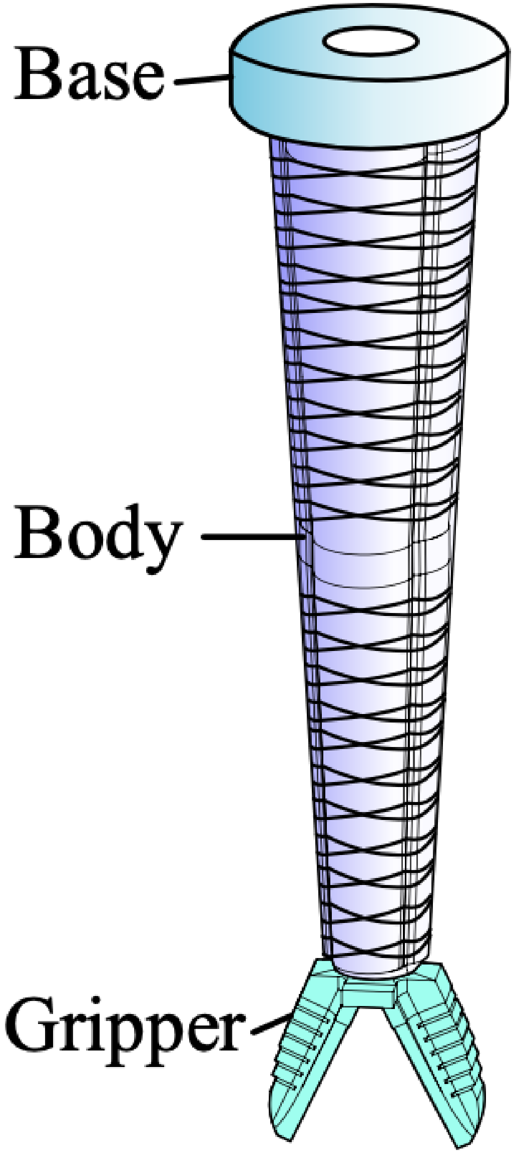


Indirect methods

e.g., Built-in Flex Sensor

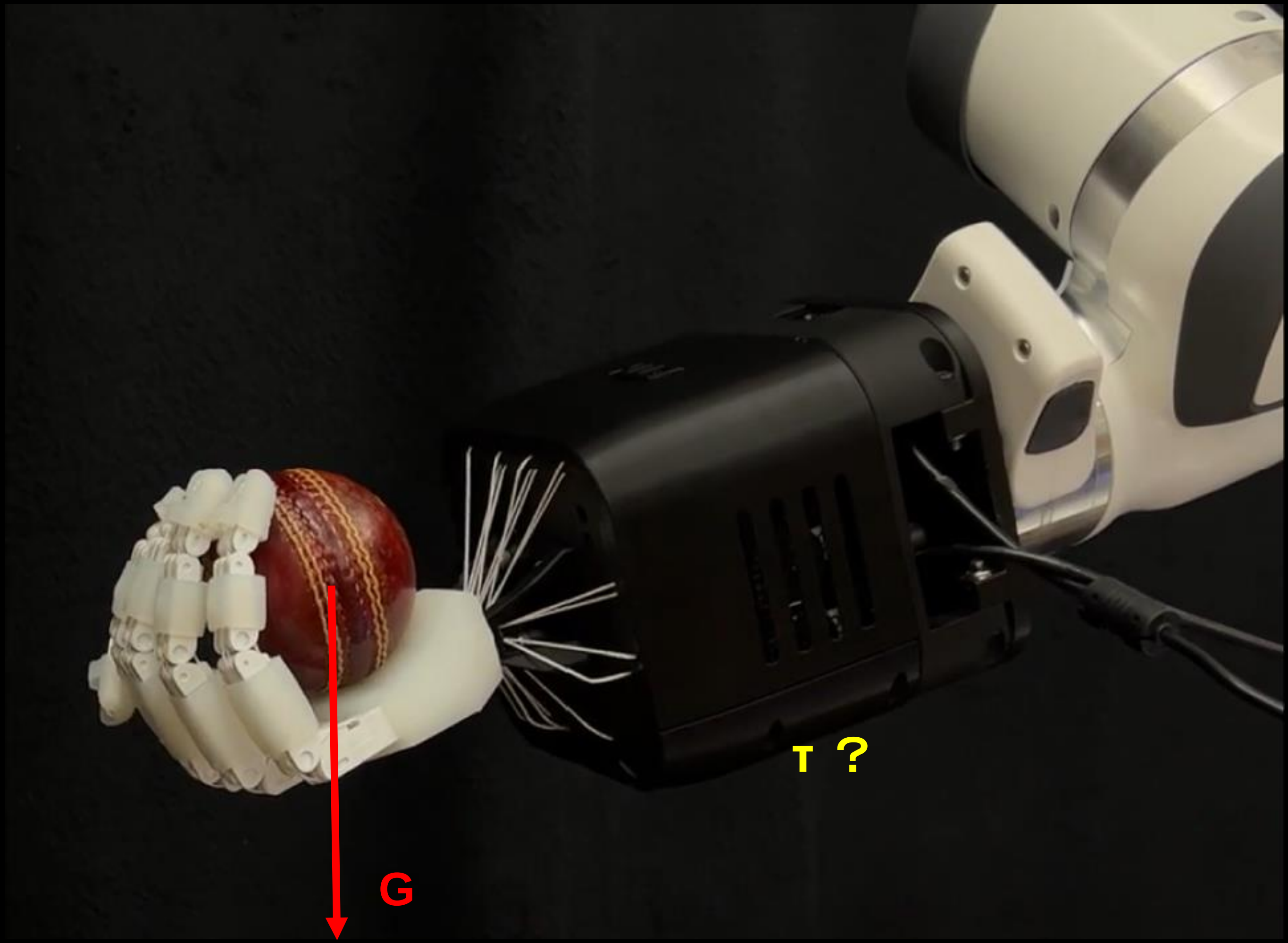






Toshimitsu, Y., Wong, K. W., Buchner, T., & Katzschmann, R. (2021, September). Sopra: Fabrication & dynamical modeling of a scalable soft continuum robotic arm with integrated proprioceptive sensing. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (pp. 653-660). IEEE.

Force control for tendon actuation



Toshimitsu et al. (2023) https://srl-ethz.github.io/get-ball-rolling/

Force control for tendon actuation

Tendon Lengths

$$p = g(l) = g\big(f(q)\big) = F(q)$$

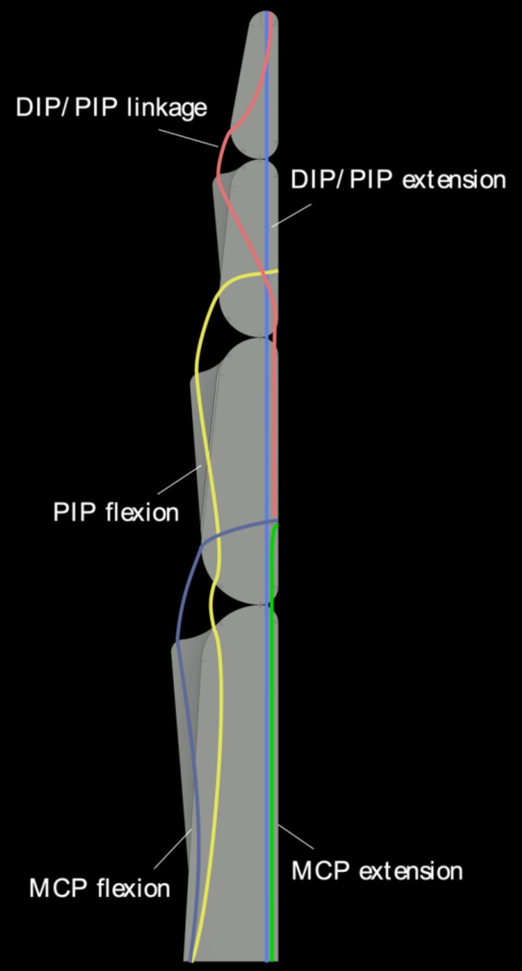
Motor Positions

Joint Angles

103

Force control for tendon actuation

$$J_m = \begin{bmatrix} \frac{\partial p_1}{\partial q_1} & \frac{\partial p_1}{\partial q_2} \\ \frac{\partial p_2}{\partial q_1} & \frac{\partial p_2}{\partial q_2} \end{bmatrix}$$



104

Force control for tendon actuation

Velocity of the finger joints

$$\dot{p} = J_m \cdot \dot{q}$$

Velocity of the motors

$$\tau^T \cdot \dot{q} = T^T \cdot \dot{p}$$

Conservation of Power

$$\tau^T \cdot \dot{q} = T^T \cdot J_m \cdot \dot{q} \quad \rightarrow \quad \tau = {J_m}^T \cdot T$$

105

Force control for tendon actuation

Previous slide: $\tau = {J_m}^T \cdot T$

$$\dot{X}_{fingertip} = J_{fingertip} \cdot \dot{q}$$

$$\tau^T \cdot \dot{q} = {F_{fingertip}}^T \cdot \dot{X}_{fingertip}$$

$$\tau^T \cdot \dot{q} = {F_{fingertip}}^T \cdot J_{fingertip} \cdot \dot{q}$$

$$\tau = {J_{fingertip}}^T \cdot F_{fingertip}$$

106

ETH zürich SoftRobotics Laboratory

Force control for tendon actuation

$$\tau = {J_m}^T \cdot T$$

$$\tau = {J_{fingertip}}^T \cdot F_{fingertip}$$

$$T = \left({J_m}^T\right)^{-1} \cdot {J_{fingertip}}^T \cdot F_{fingertip}$$

107

Outro no slide

ETH zürich SoftRobotics Laboratory

Useful links

https://link.springer.com/book/10.1007/978-3-319-54413-7
https://smartlabai.medium.com/a-brief-overview-of-imitation-learning-8a8a75c44a9c
https://underactuated.csail.mit.edu/index.html
https://www.kalmanfilter.net/default.aspx