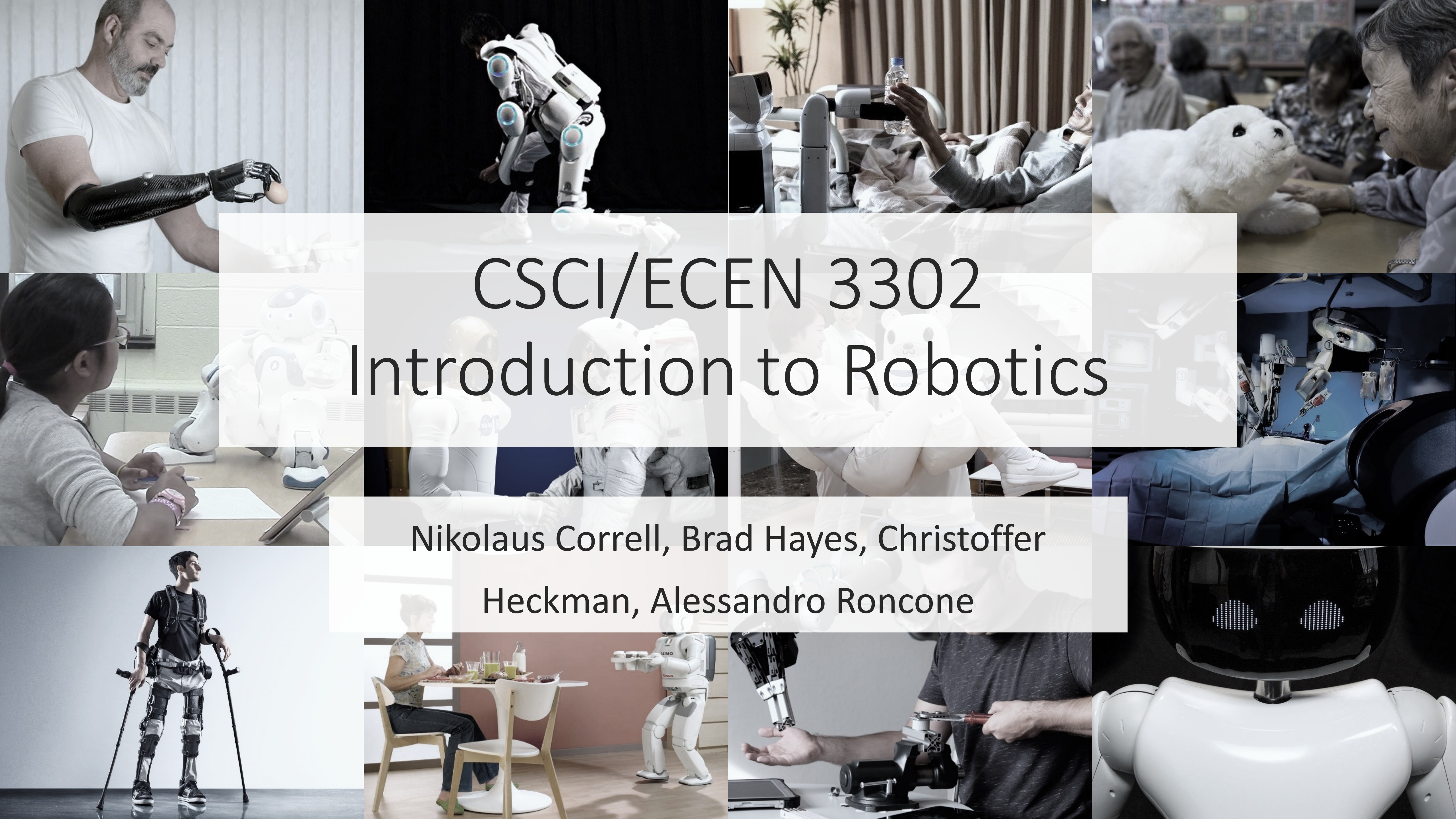
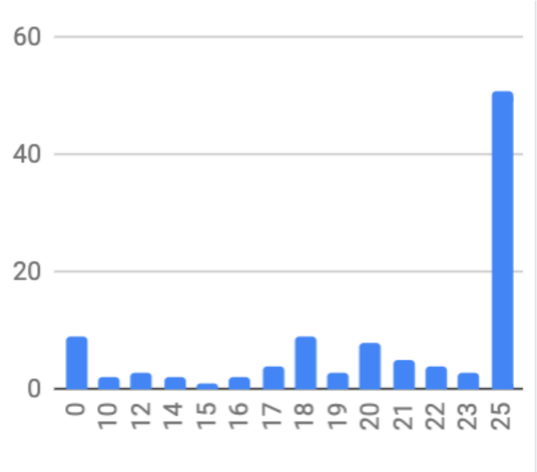


CSCI/ECEN 3302 Introduction to Robotics

Nikolaus Correll, Brad Hayes, Christoffer Heckman, Alessandro Roncone

Administrivia

- Groups are now finalized
  - Moving forward, all labs will be group assignments
  - If you don’t have a team, please reach out!
  - Coordinate with your groupmates, don’t miss lab without talking to them!
- Homework 0 graded, feel free to reach out on Piazza
- Lab 1 due tonight!
  - We currently have 26 submission out of 100 for Lab 1
- Homework 1 is out

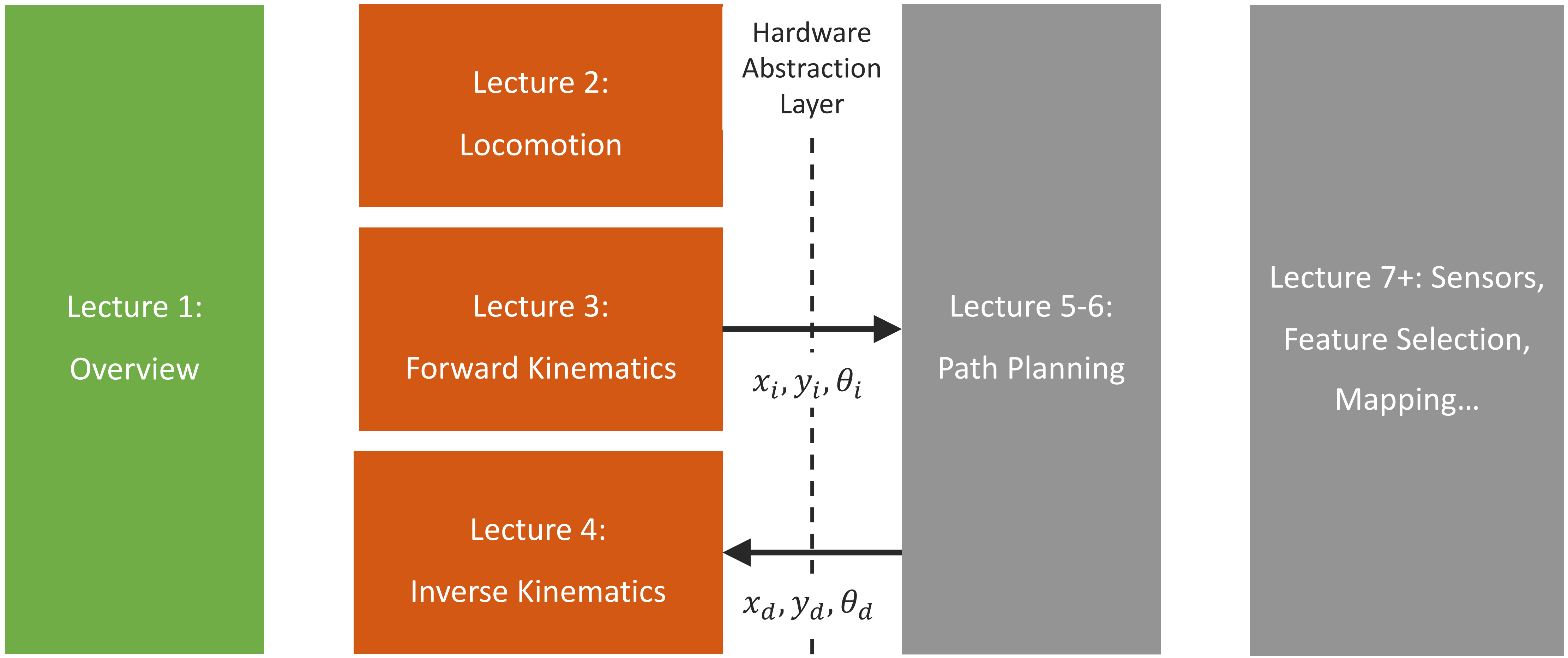


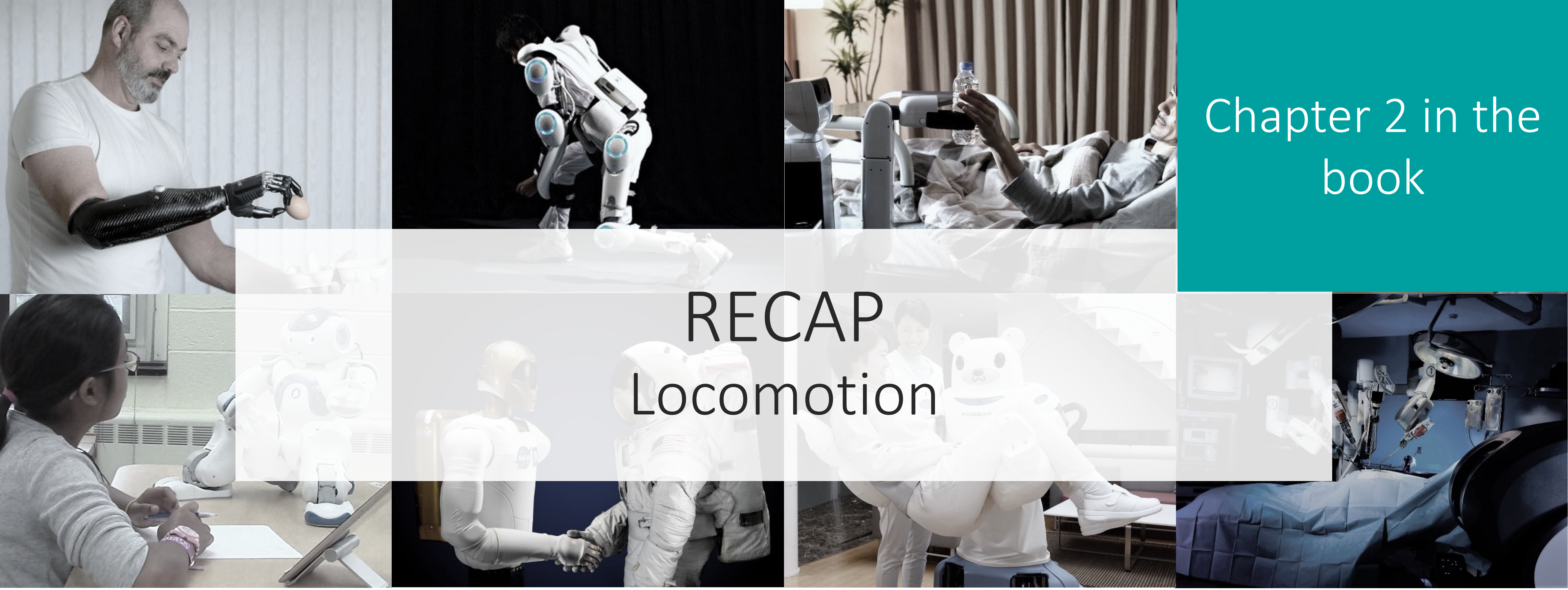
Median = 23

Mean = 19.8


Questions?

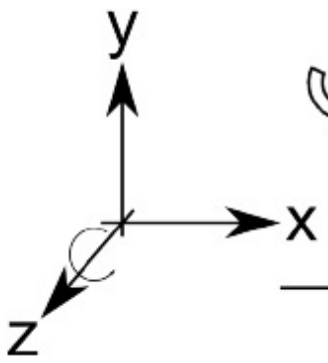
Roadmap

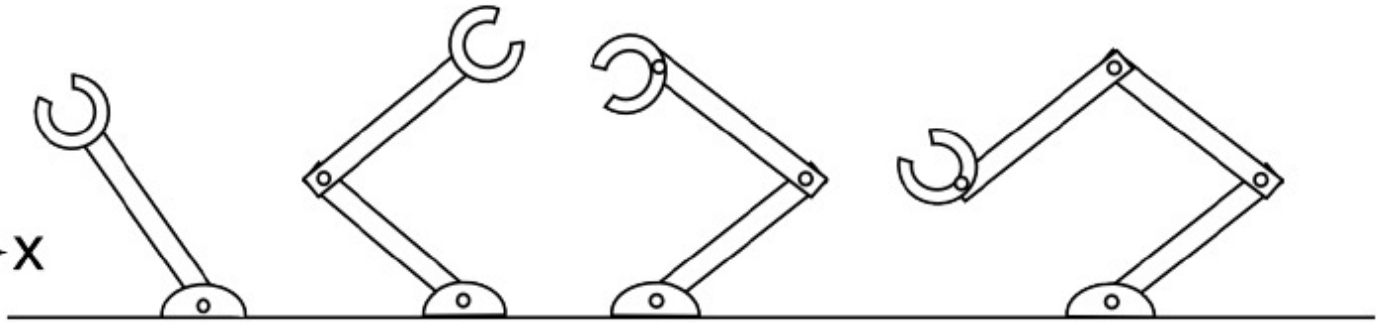

Lecture 1:
Overview
Lecture 2:
Locomotion
Lecture 3:
Forward Kinematics
Lecture 4:
Inverse Kinematics
Hardware
Abstraction
Layer
x_i, y_i, θ_i
x_d, y_d, θ_d
Lecture 5-6:
Path Planning
Lecture 7+: Sensors,
Feature Selection,
Mapping…

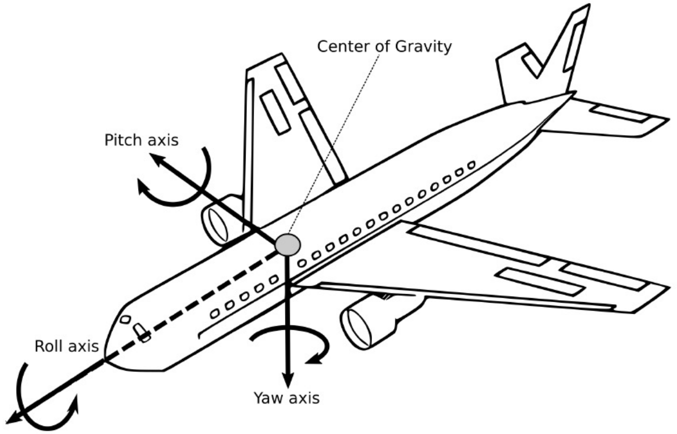


Chapter 2 in the book

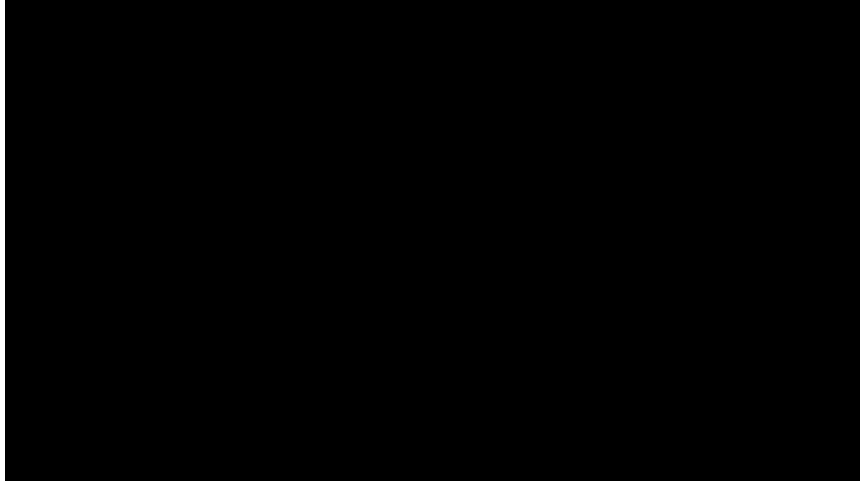
RECAP
Locomotion



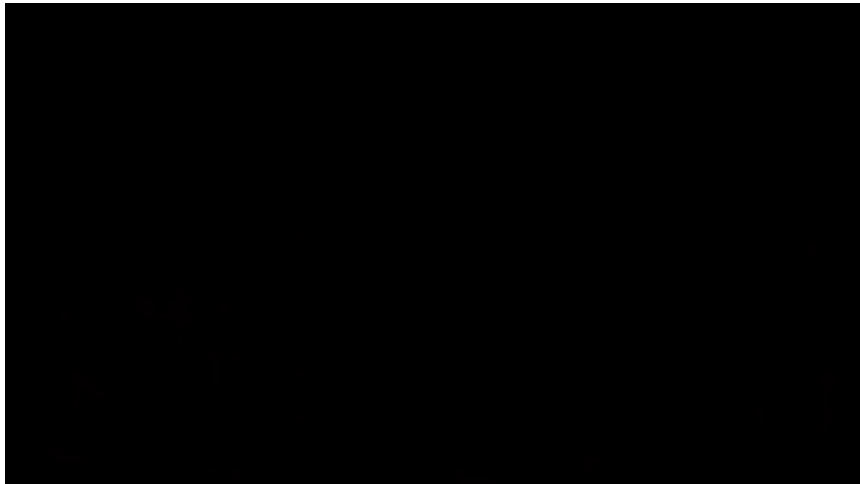




Actuation



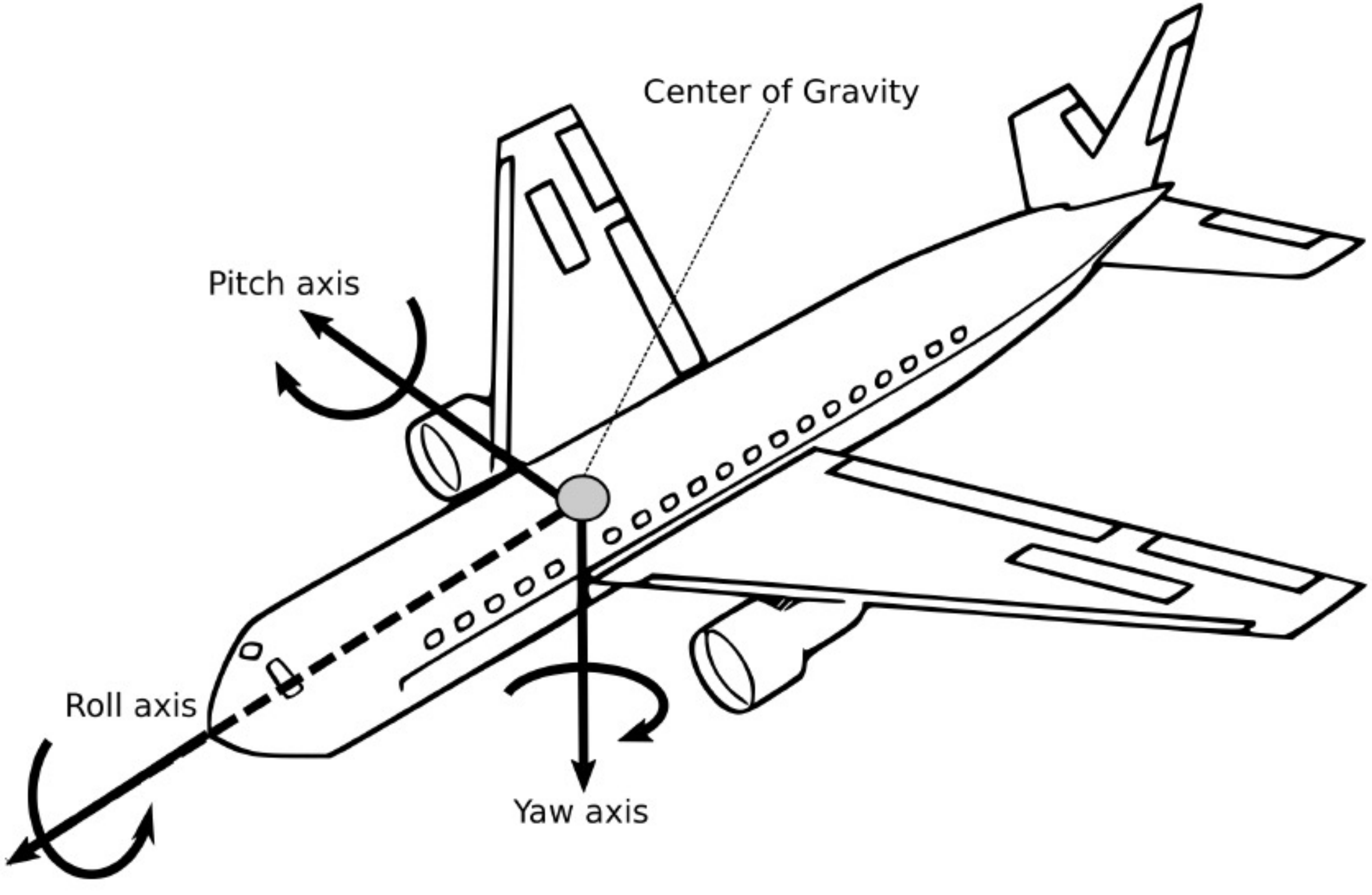
- Locomotion – Moving oneself
  - Enacting a force on / utilizing a force within the environment
  - Rolling, walking, running, jumping, sliding, flying, etc.



- Manipulation – Moving others
  - Enacting forces on objects

Degrees of Freedom [DoFs]

- 6-dimensional:
  - X, Y, Z
  - Yaw, Pitch, and Roll
- 3D position + 3D orientation = 6D pose
- Fully characterizes the state of the plane



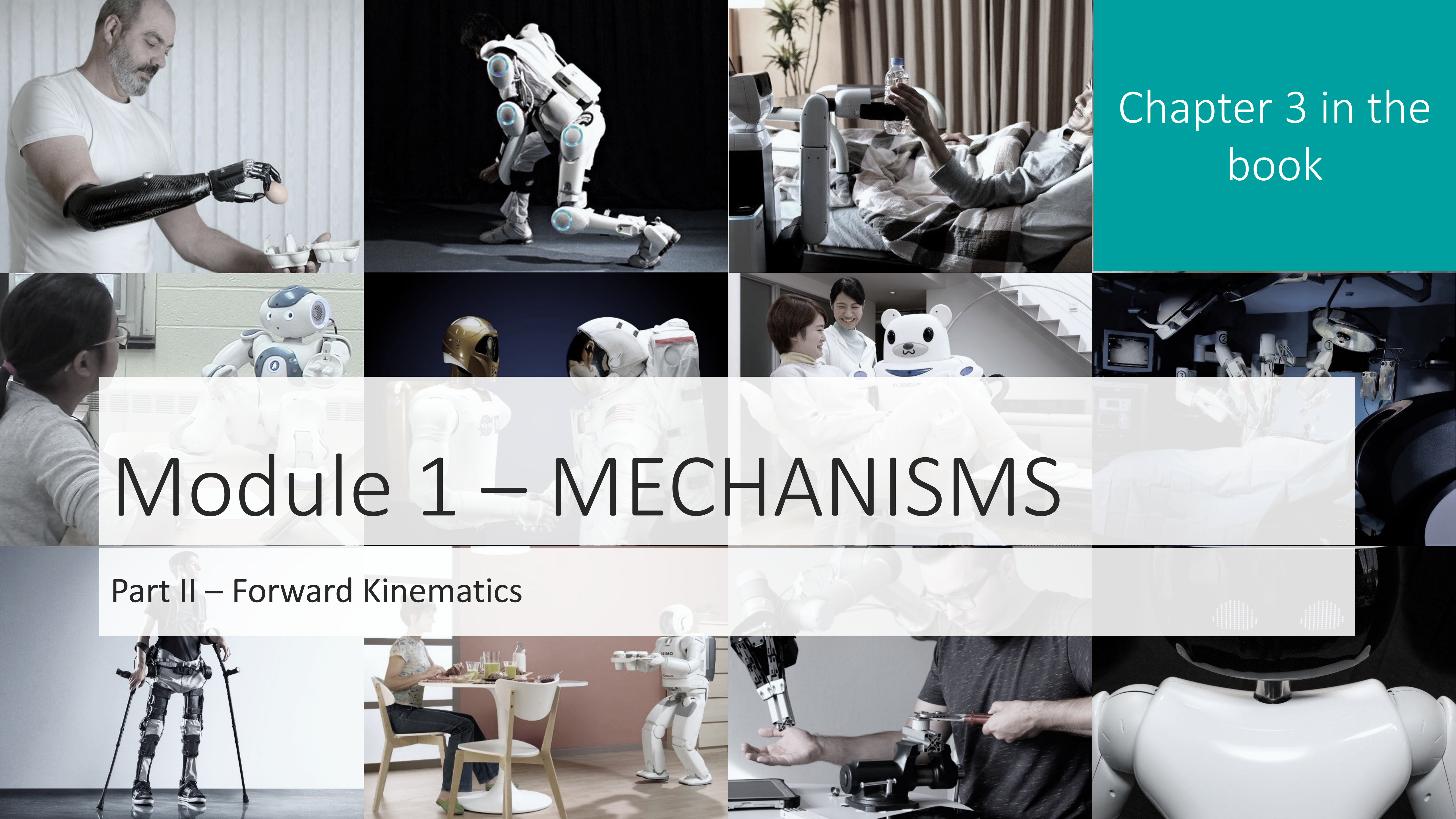
Degrees of Freedom in Joint and Cartesian Space

- DoF = Number of controllable dimensions of actuation
- Most robot arms have 6-DoF at their end effector
  - DoF in cartesian space
  - They are not directly controllable

- Robots can be controlled through motors in the joints
  - DoFs in joint space
  - REDUNDANCY: Robot DoF > Environment DoF
  - Why would you want this?

cartesian space == operational space == task space == environment

How many DoFs does a human arm have?

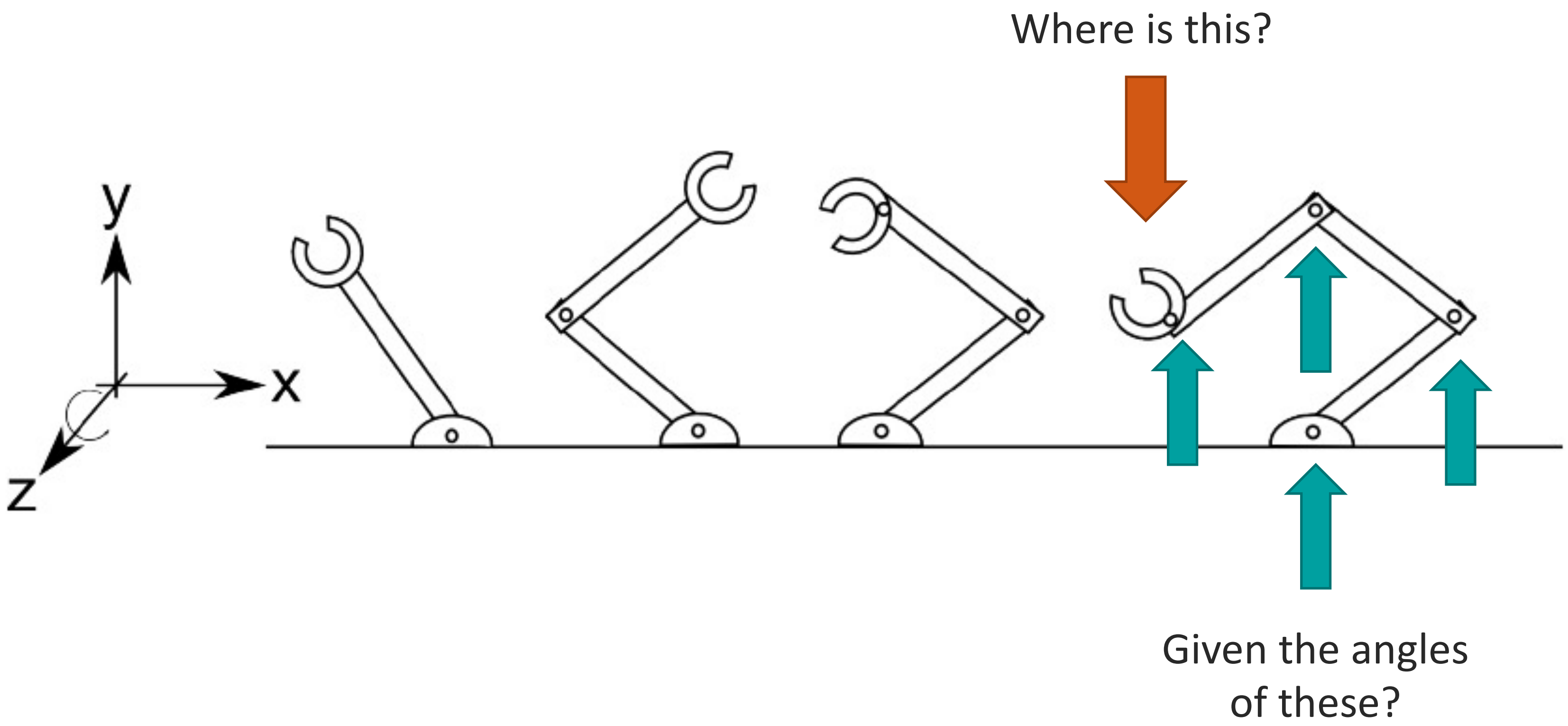


Chapter 3 in the book

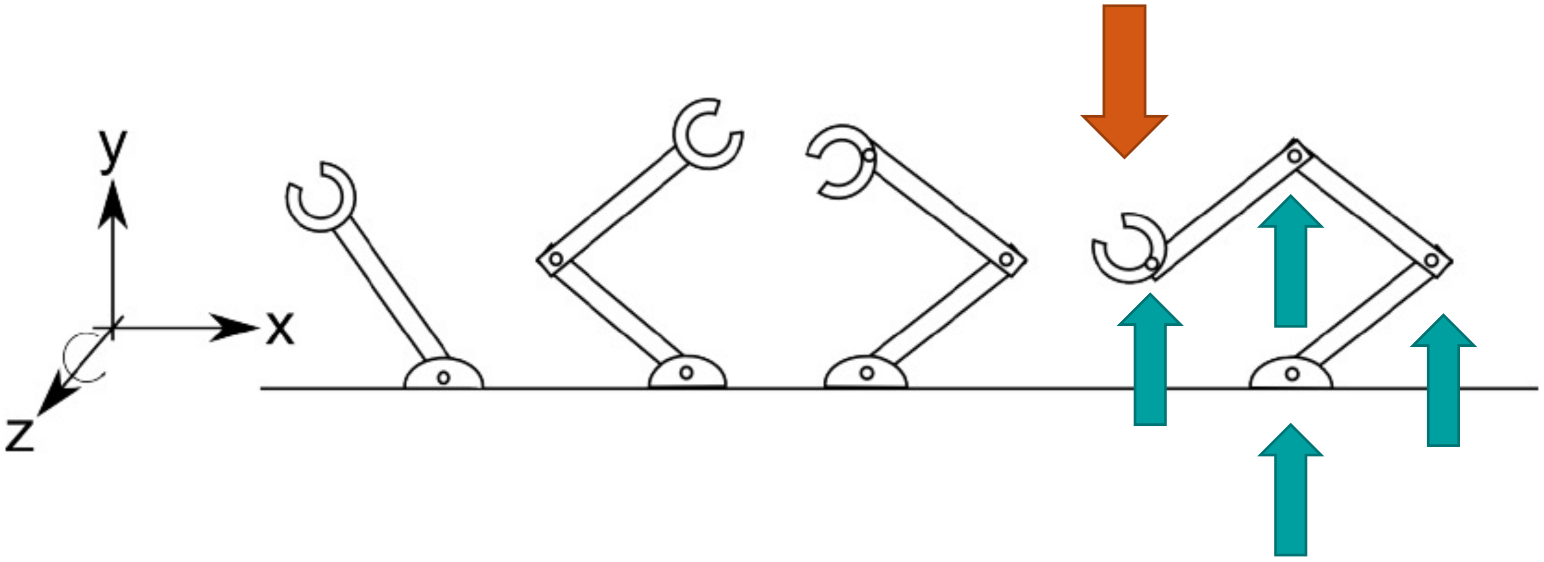
Module 1 – MECHANISMS

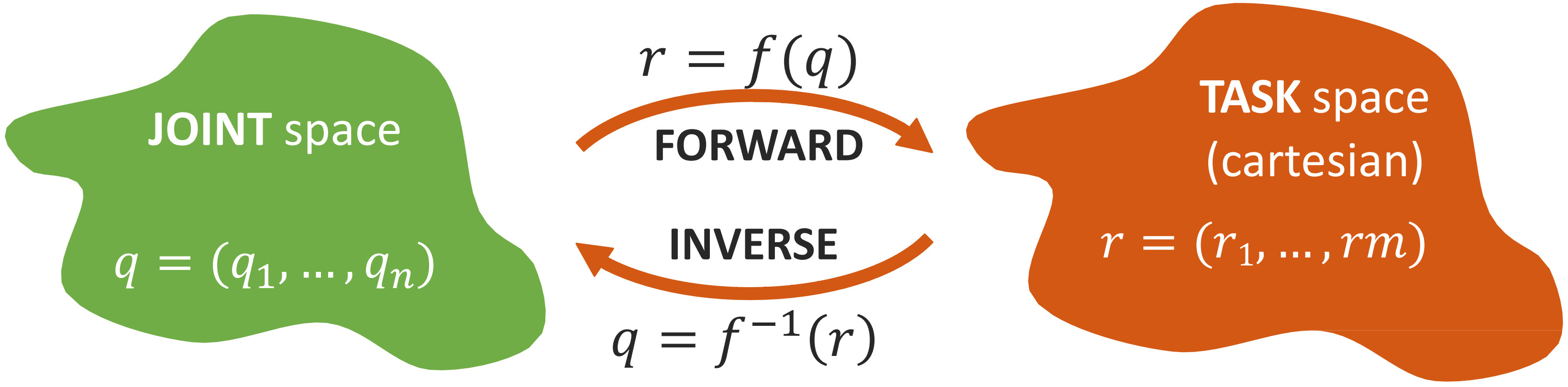
Part II – Forward Kinematics

Forward (Direct) Kinematics



Forward (Direct) Kinematics





Forward Kinematics

Given a robot configuration, what is the end-effector’s pose?

Inverse Kinematics

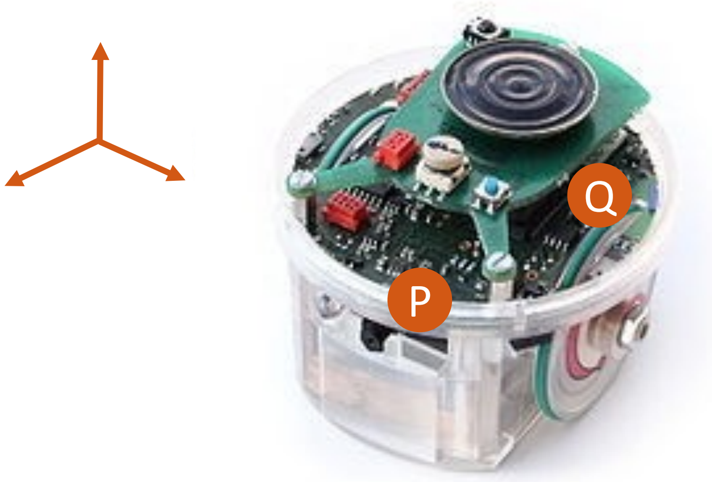
How to move an arm in space in order to reach for a specific pose [or position, or orientation]?

Forward Kinematics – Definitions

- Coordinate Systems
- Nested Coordinate Systems
- Introduction to Rotations: Deriving 2D Rotation Equations

Chapter 3.1

Rigid Body Transformations



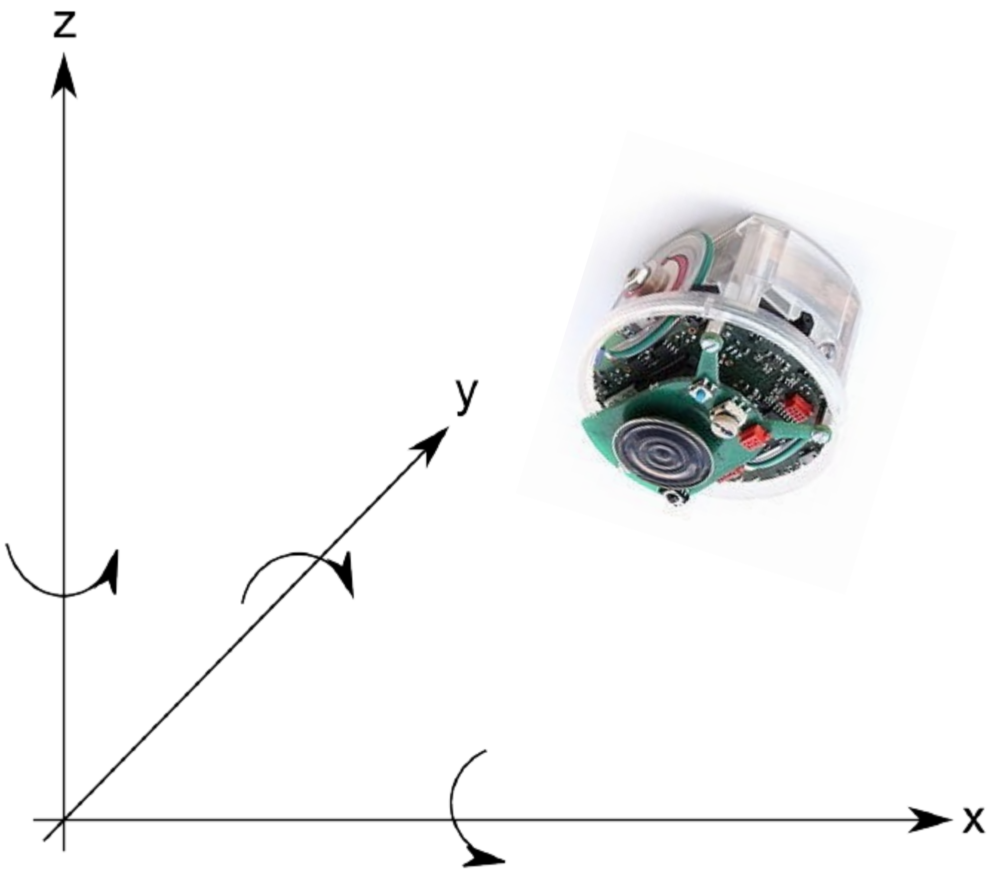
- Rigid Body:
  - $O \in \mathbb{R}^3$
  - $\forall p, q \in O$:
  - $\|p(t) - q(t)\| = \|p(0) - q(0)\|$

- Rigid Body Transformation:

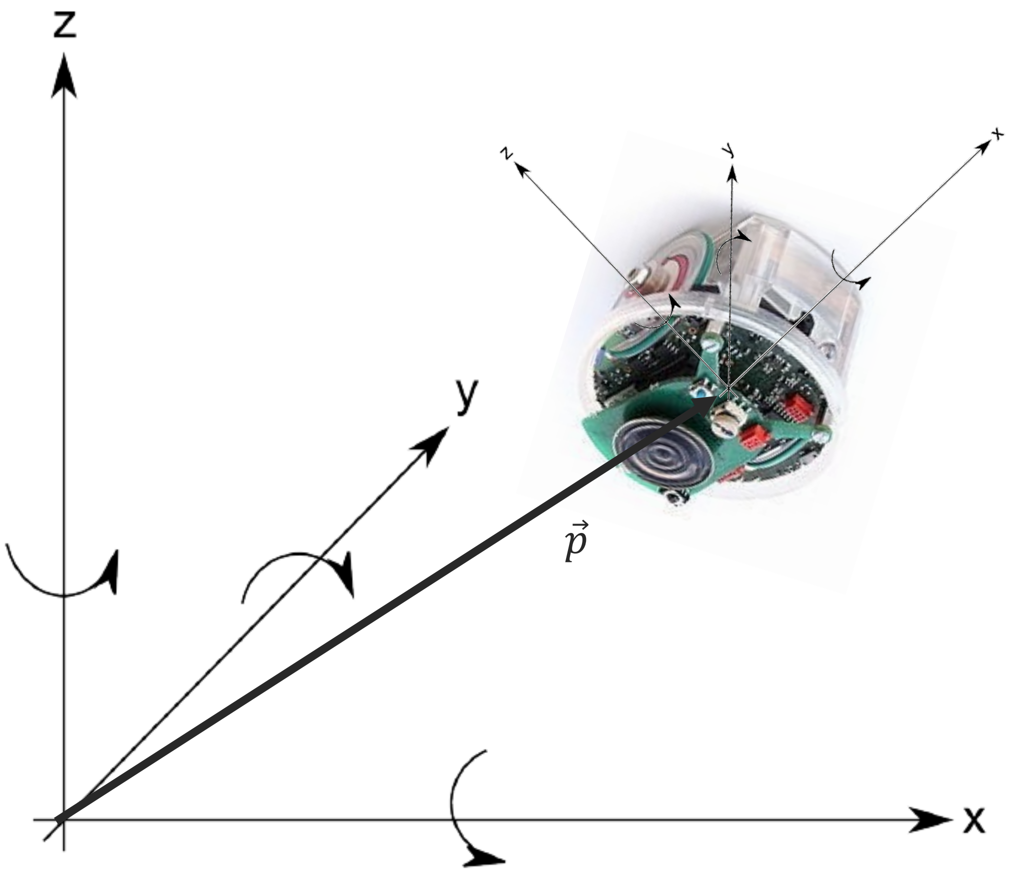
  A mapping $\mathbf{g}: \mathbb{R}^3 \rightarrow \mathbb{R}^3$ is a rigid body transformation if:

  - Length is preserved: $\|g(p) - g(q)\| = \|p - q\| \; \forall p, q \in \mathbb{R}^3$
  - Cross product is preserved: $g_*(v \times w) = g_*(v) \times g_*(w) \forall v, w \in \mathbb{R}^3$

What do we need to describe pose?



What do we need to describe pose?



Notation

- $\vec{\mathrm{p}} = c_1\vec{b}_1 + c_2\vec{b}_2 + c_3\vec{b}_3$

  - Basis vectors $\vec{b}$ identify the coordinate system axes
  - Basis vectors need to be *orthonormal*
  - Coefficients c express how far to travel along each axis

- $\vec{\mathrm{p}} = \underbrace{\begin{bmatrix} \vec{b}_1 & \vec{b}_2 & \vec{b}_3 \end{bmatrix}}_{\mathrm{R}} \begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix}$

Shorthand: $\vec{\mathrm{p}} = \mathrm{R}\vec{c}$

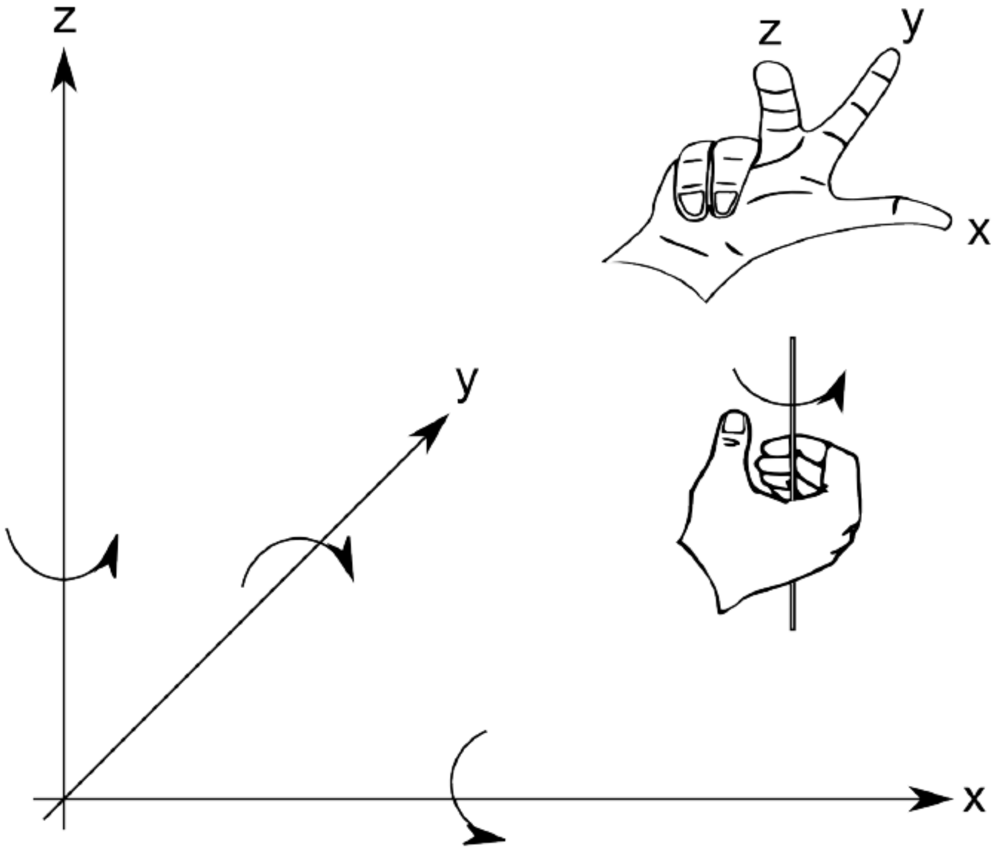
Example

- $\vec{p} = c_1 \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} + c_2 \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} + c_3 \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$
  - Basis vectors $\vec{b}$ identify the coordinate system axes
  - Basis vectors need to be *orthonormal*
  - Coefficients c express how far to travel along each axis

- $\vec{p} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix}$

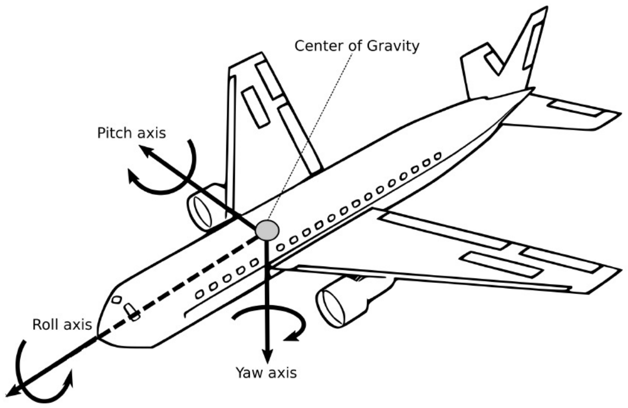
Shorthand: $\vec{p} = R\vec{c} = \vec{c}$

Coordinate Systems and Right Hand Rule



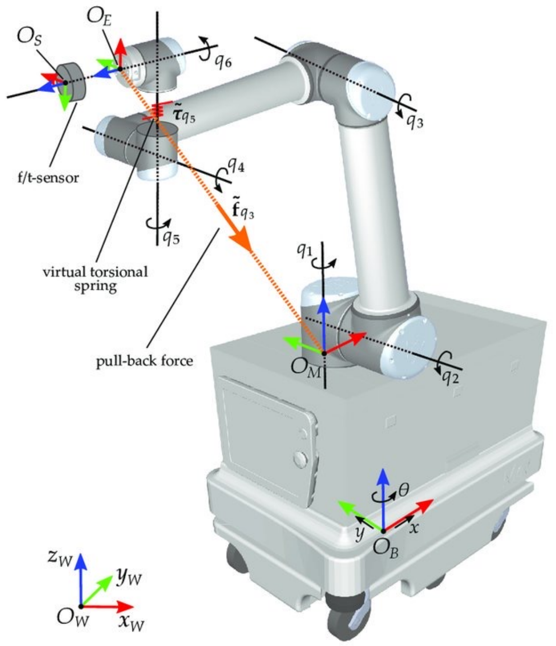
- Thumb along x-axis
- Index along y-axis
- Middle along z-axis
- It’s the order that counts!

Nested Coordinate Systems

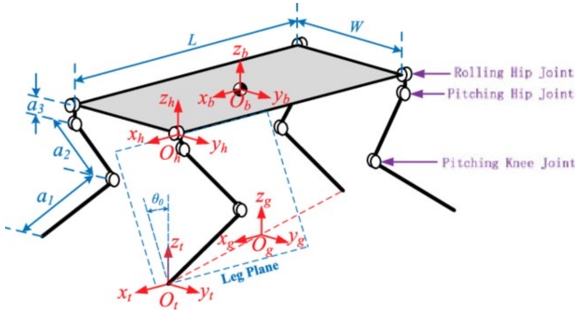


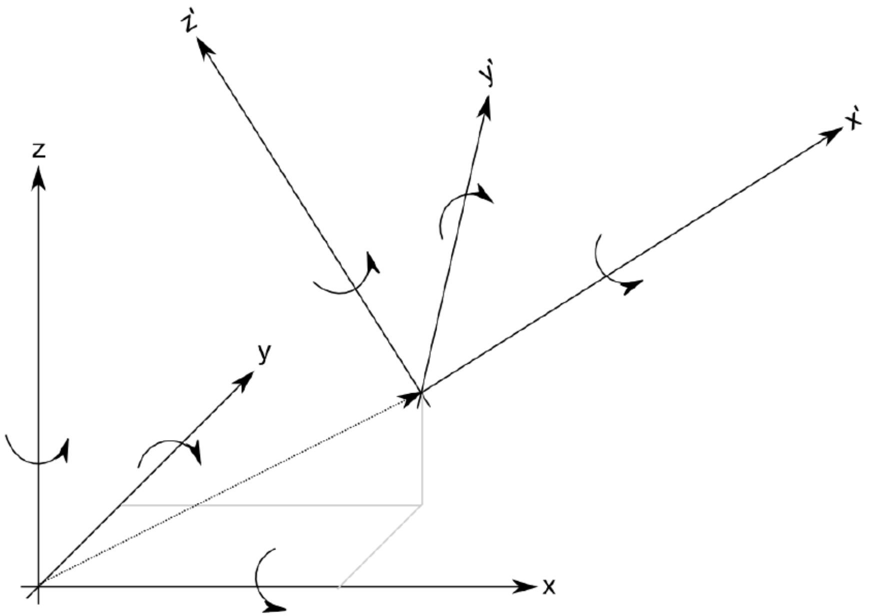
- Recall: an airplane can be modeled as a single rigid body
  - A position and coordinate system fully characterizes its kinematic state
- How can a robot manipulator be modeled?
  - Multiple rigid bodies connected together by joints
  - Each DoF / point of actuation presents a new Coordinate System w.r.t. the previous linkage

Nested Coordinate Systems



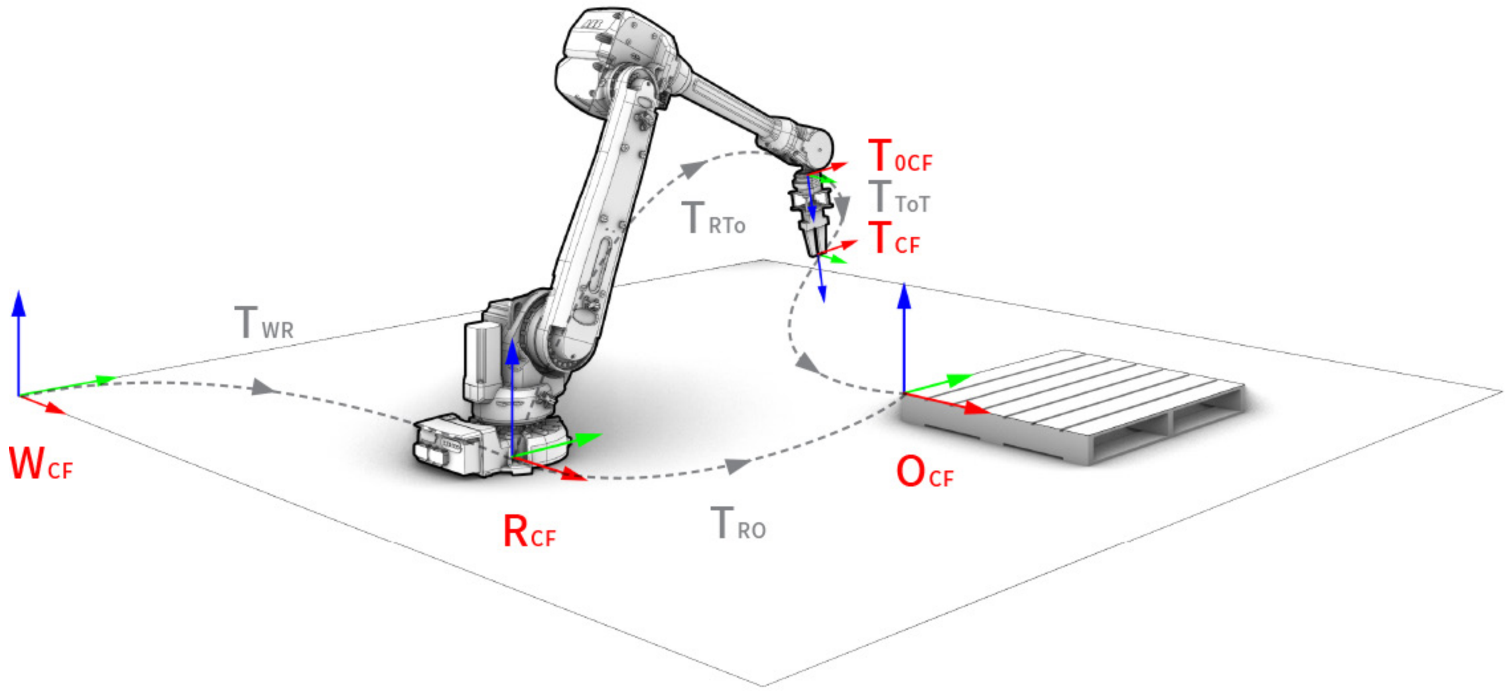
- Each DoF / point of actuation presents a new Coordinate System w.r.t. the previous linkage
- Applies to both manipulators and mobile platforms!



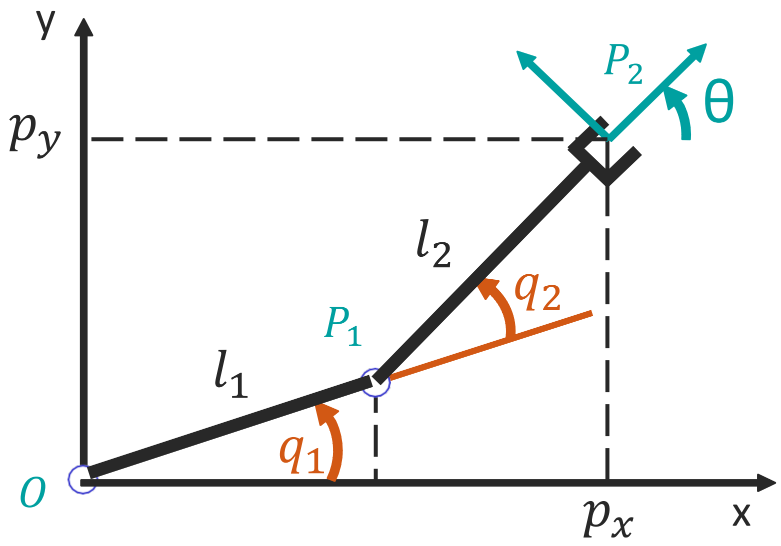


Nested Coordinate Systems

- This idea can also be used to describe the pose of objects in the environment



Example: direct kinematics of 2R arm



$q = \begin{bmatrix} q_1 \\ q_2 \end{bmatrix} \Leftarrow$ Joint-space DOFs

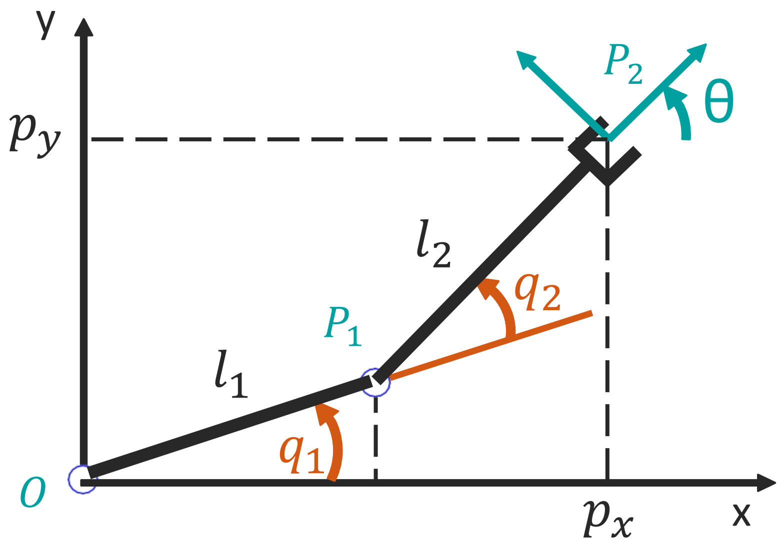
$r = \begin{bmatrix} p_{2,x} \\ p_{2,y} \\ \theta \end{bmatrix} \Leftarrow$ Operational-space DOFs

$r=f(q)$

$$p_{2,x} = \cdots$$
$$p_{2,y} = \cdots$$
$$\theta = \cdots$$

Please take a look at the example in the book too!!! Section 3.1.1

Example: direct kinematics of 2R arm



$q = \begin{bmatrix} q_1 \\ q_2 \end{bmatrix} \Leftarrow$ Joint-space DOFs

$r = \begin{bmatrix} p_{2,x} \\ p_{2,y} \\ \theta \end{bmatrix} \Leftarrow$ Operational-space DOFs

$r = f(q)$

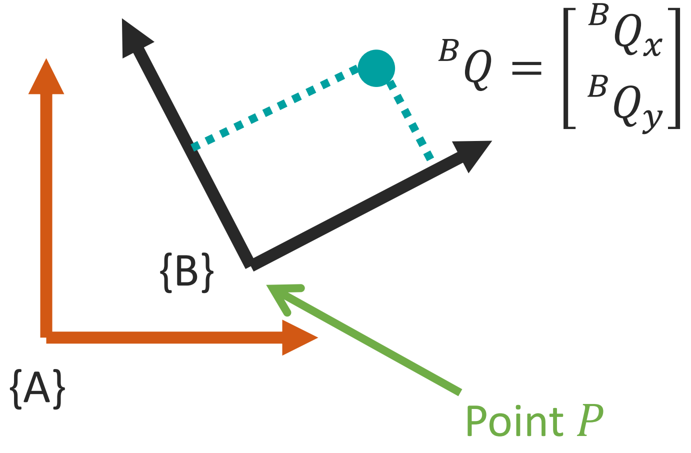
$$\begin{aligned} p_{2,x} &= l_1 \cos(q_1) + l_2 \cos(q_1 + q_2) \\ p_{2,y} &= l_1 \sin(q_1) + l_2 \sin(q_1 + q_2) \\ \theta &= q_1 + q_2 \end{aligned}$$

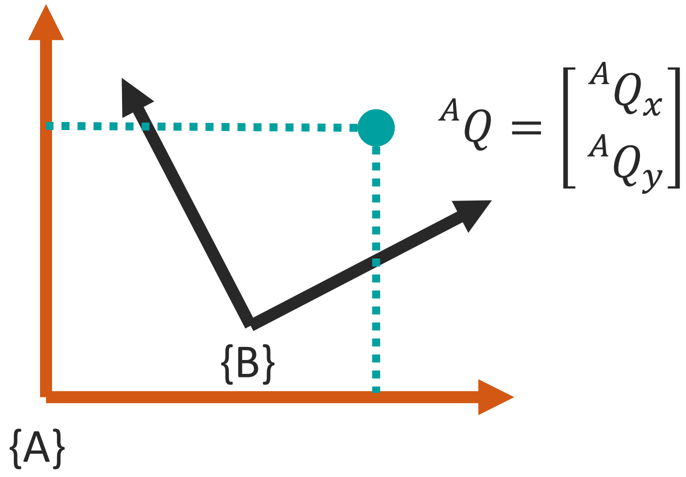
For more general cases, we need a “method”!

Please take a look at the example in the book too!!! Section 3.1.1

Nested Coordinate Systems

How can we express point Q, with position known in coordinate system {B} centered at P, in coordinate system {A} with origin (0,0)?

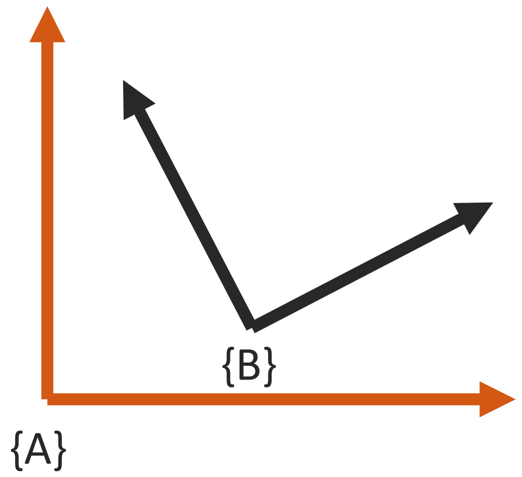




Nested Coordinate Systems

How can we express point Q, with position known in coordinate system {B} centered at P, in coordinate system {A} with origin (0,0)?

- How can we convert between {A} and {B}?

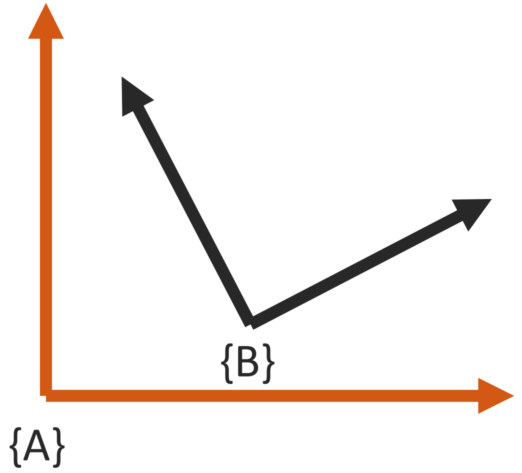


We need to find the **translation** and **rotation** that would move the axes of {A} onto {B}

Nested Coordinate Systems

How can we convert between {A} and {B}?

We need to find the translation and rotation that would move the axes of {A} onto {B}



Translation:

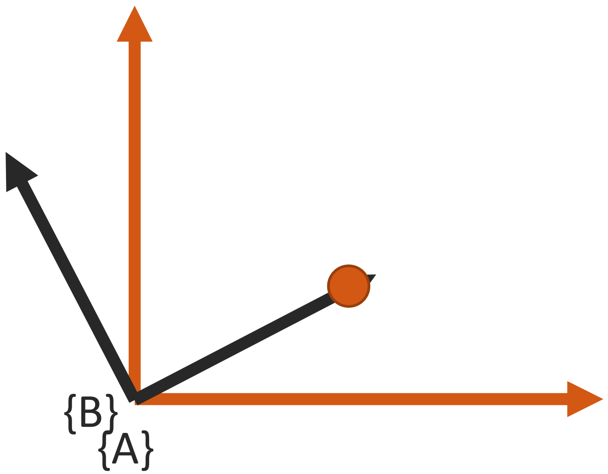
Subtract {A} from {B} to get the difference between origins.

$$T_x = \hat{B}_x - \hat{A}_x$$
$$T_y = \hat{B}_y - \hat{A}_y$$
$$T_z = \hat{B}_z - \hat{A}_z$$

Nested Coordinate Systems

How can we convert between {A} and {B}?

We need to find the translation and rotation that would move the axes of {A} onto {B}



Rotation:

For each axis of {B}, we need to find out how it maps onto {A}

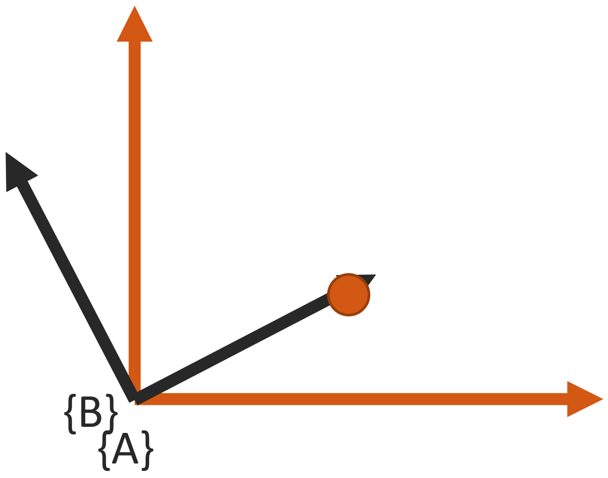
Starting with one axis:

How to map ${}^{B}\begin{bmatrix}1\\0\end{bmatrix}$ onto A?

$\{B\}_x$ contributes to both $\{A\}_x$ and $\{A\}_y$

Nested Coordinate Systems: Projection

- Problem: How to map ${}^{B}\begin{bmatrix}1\\0\end{bmatrix}$ onto A? $\{B\}_x$ contributes to both $\{A\}_x$ and $\{A\}_y$
- Solution: Need to find the amount of these contributions!



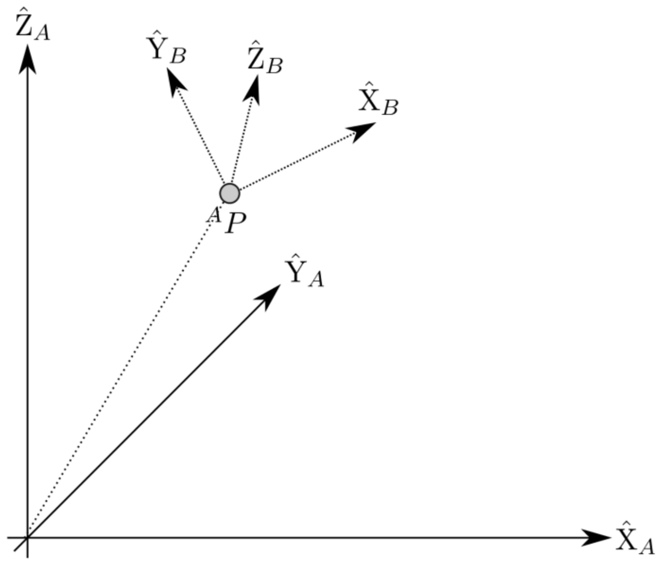
Recall:

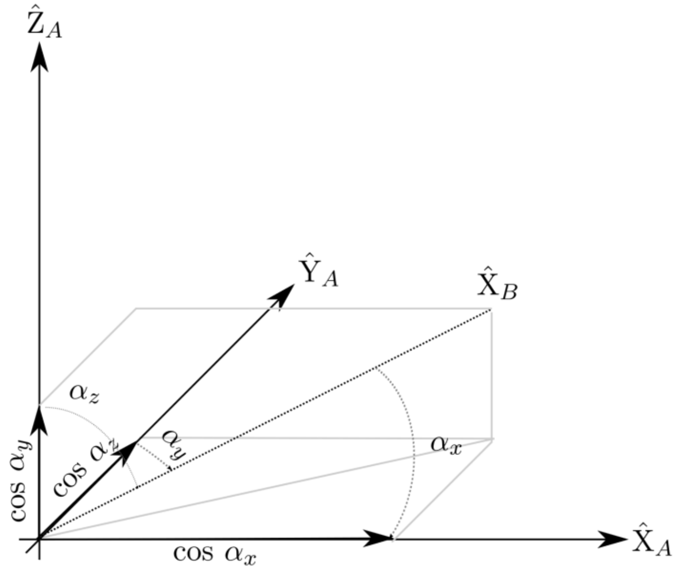
- Axes are unit-length vectors
- Dot product formula
  - $A \cdot B = \sum_i A_i B_i = |A| * |B| * \cos(\alpha)$

$\{B\}_x \cdot \{A\}_x$ = Amount of $\{B\}_x$ that contributes to $\{A\}_x$
$\{B\}_x \cdot \{A\}_y$ = Amount of $\{B\}_x$ that contributes to $\{A\}_y$

$$ {}^{A}_{B}R = \begin{bmatrix} \{B\}_x \cdot \{A\}_x & \{B\}_y \cdot \{A\}_x \\ \{B\}_x \cdot \{A\}_y & \{B\}_y \cdot \{A\}_y \end{bmatrix} $$

Expressing $\{B\}$ in $\{A\}$





$${}^{A}\hat{X}_B = (\hat{X}_B \cdot \hat{X}_A, \hat{X}_B \cdot \hat{Y}_A, \hat{X}_B \cdot \hat{Z}_A)^T$$

$${}^{A}\hat{Y}_B = (\hat{Y}_B \cdot \hat{X}_A, \hat{Y}_B \cdot \hat{Y}_A, \hat{Y}_B \cdot \hat{Z}_A)^T$$

$${}^{A}\hat{Z}_B = (\hat{Z}_B \cdot \hat{X}_A, \hat{Z}_B \cdot \hat{Y}_A, \hat{Z}_B \cdot \hat{Z}_A)^T$$

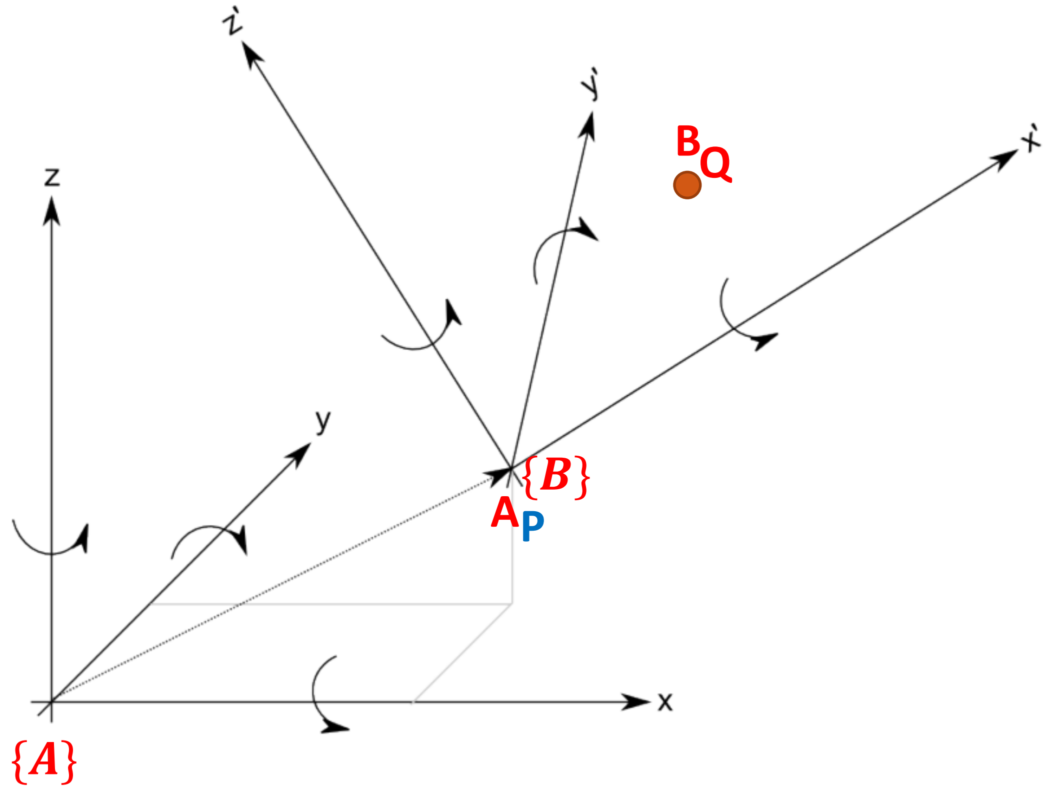
$${}^{A}_{B}R = [\,{}^{A}X_{B,}\ {}^{A}Y_{B,}\ {}^{A}Z_B] = \begin{bmatrix} \widehat{X_B} \cdot \widehat{X_A} & \widehat{Y_B} \cdot \widehat{X_A} & \widehat{Z_B} \cdot \widehat{X_A} \\ \widehat{X_B} \cdot \widehat{Y_A} & \widehat{Y_B} \cdot \widehat{Y_A} & \widehat{Z_B} \cdot \widehat{Y_A} \\ \widehat{X_B} \cdot \widehat{Z_A} & \widehat{Y_B} \cdot \widehat{Z_A} & \widehat{Z_B} \cdot \widehat{Z_A} \end{bmatrix}$$

Nested Coordinate Systems

How can we express point Q, with position known in coordinate system {B} centered at P, in coordinate system {A} centered at the origin (0,0)?

^{B}Q = Known

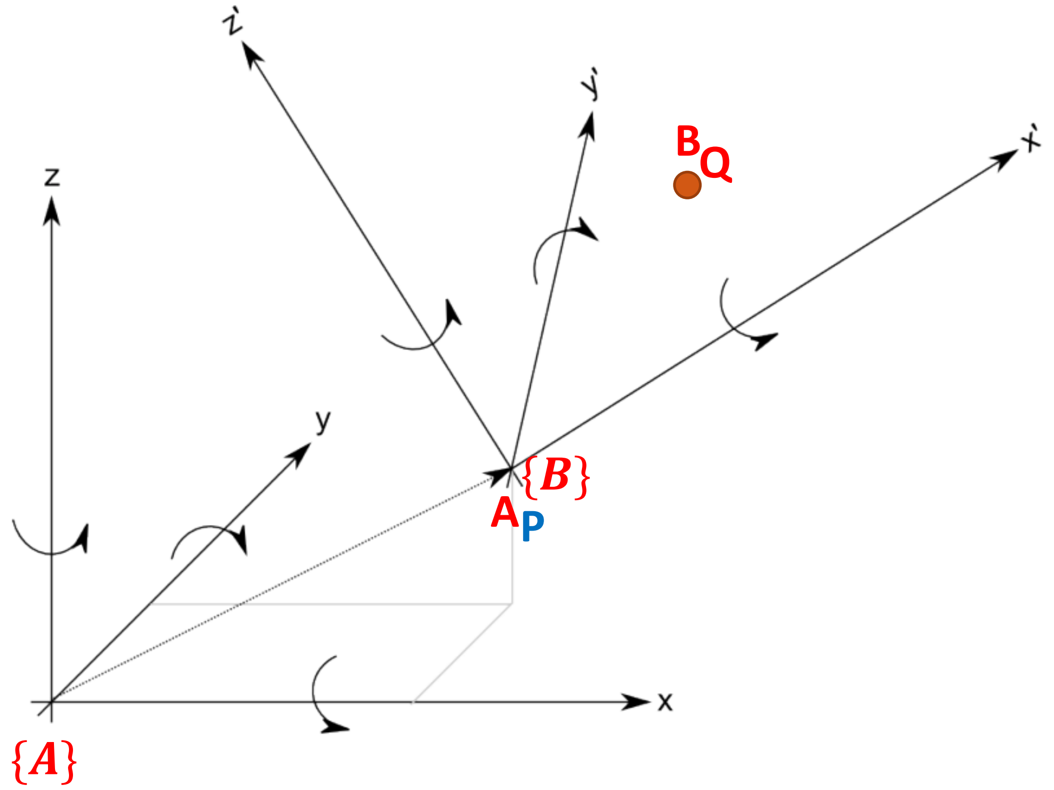
^{A}P = Known



Nested Coordinate Systems

How can we express point Q, with position known in coordinate system {B} centered at P, in coordinate system {A} centered at the origin (0,0)?

- ${}^{A}_{B}R = [\,{}^{A}X_{B,}\ {}^{A}Y_{B,}\ {}^{A}Z_{B}]$
- ${}^{A}X_{B} = [X_B \cdot X_A, X_B \cdot Y_A, X_B \cdot Z_A]^T$
- ${}^{A}Q = {}^{A}_{B}R * {}^{B}Q + {}^{A}P$



Homogenous Transform

- Instead of ${}^{A}Q = {}^{A}_{B}R * {}^{B}Q + {}^{A}P$, we can express the transformation as a single matrix multiplication:

$$\begin{bmatrix} {}^{A}Q \\ 1 \end{bmatrix} = \begin{bmatrix} & {}^{A}_{B}R & & {}^{A}P \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} {}^{B}Q \\ 1 \end{bmatrix}$$

- Inverse Transform:

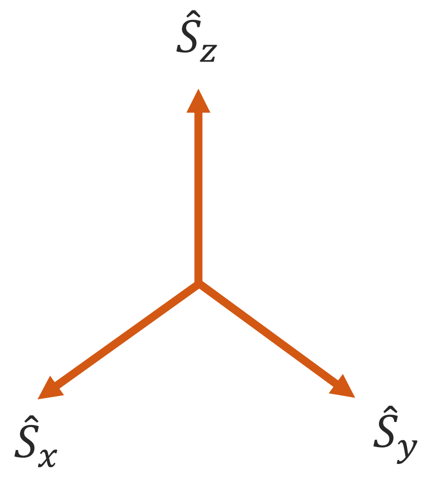
$$T^{-1} = \begin{bmatrix} R^T & -R^T\,{}^{A}P \\ 0 & 1 \end{bmatrix} \in SE(3)$$

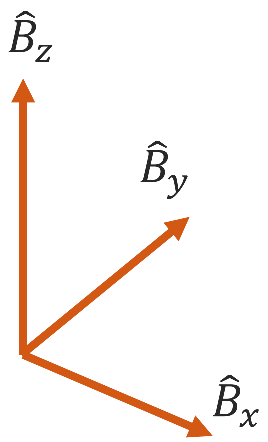
Example

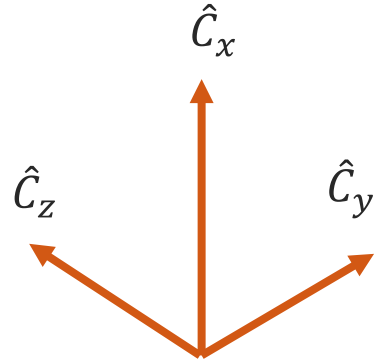
$$ {}^{A}\hat{X}_B = (\hat{X}_B \cdot \hat{X}_A, \hat{X}_B \cdot \hat{Y}_A, \hat{X}_B \cdot \hat{Z}_A)^T $$

$$ {}^{A}\hat{Y}_B = (\hat{Y}_B \cdot \hat{X}_A, \hat{Y}_B \cdot \hat{Y}_A, \hat{Y}_B \cdot \hat{Z}_A)^T $$

$$ {}^{A}\hat{Z}_B = (\hat{Z}_B \cdot \hat{X}_A, \hat{Z}_B \cdot \hat{Y}_A, \hat{Z}_B \cdot \hat{Z}_A)^T $$







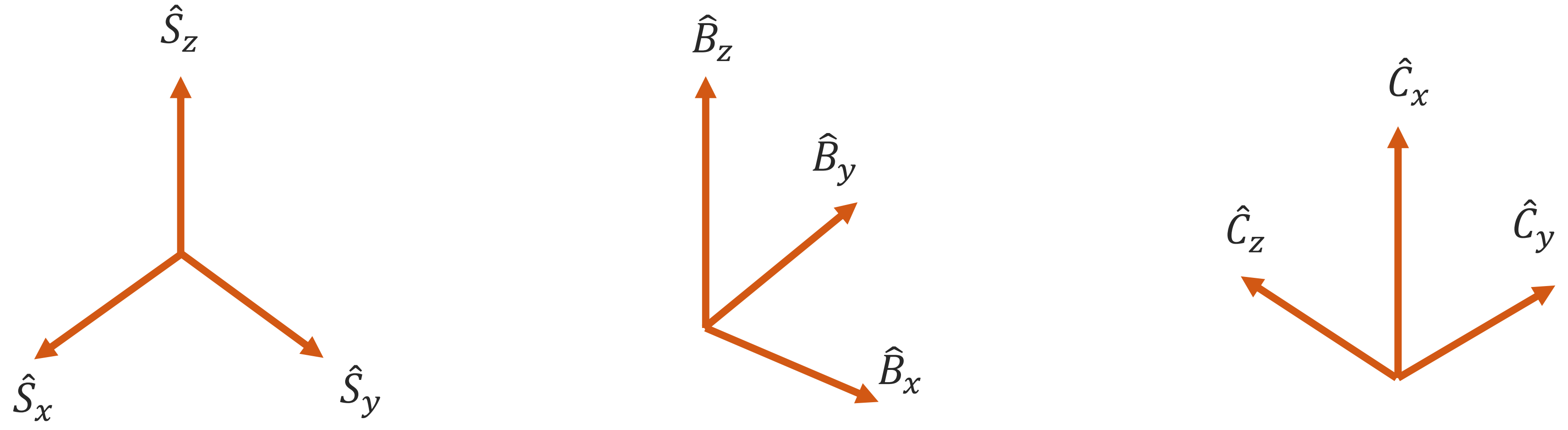
$$ {}^{\hat{S}}_{\hat{C}}R = \begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & -1 \\ 1 & 0 & 0 \end{bmatrix} $$

X-axis of $\hat{S}$ is the negative-Y-axis of $\hat{C}$

Y-axis of $\hat{S}$ is the negative-Z-axis of $\hat{C}$

Z-axis of $\hat{S}$ is the X-axis of $\hat{C}$

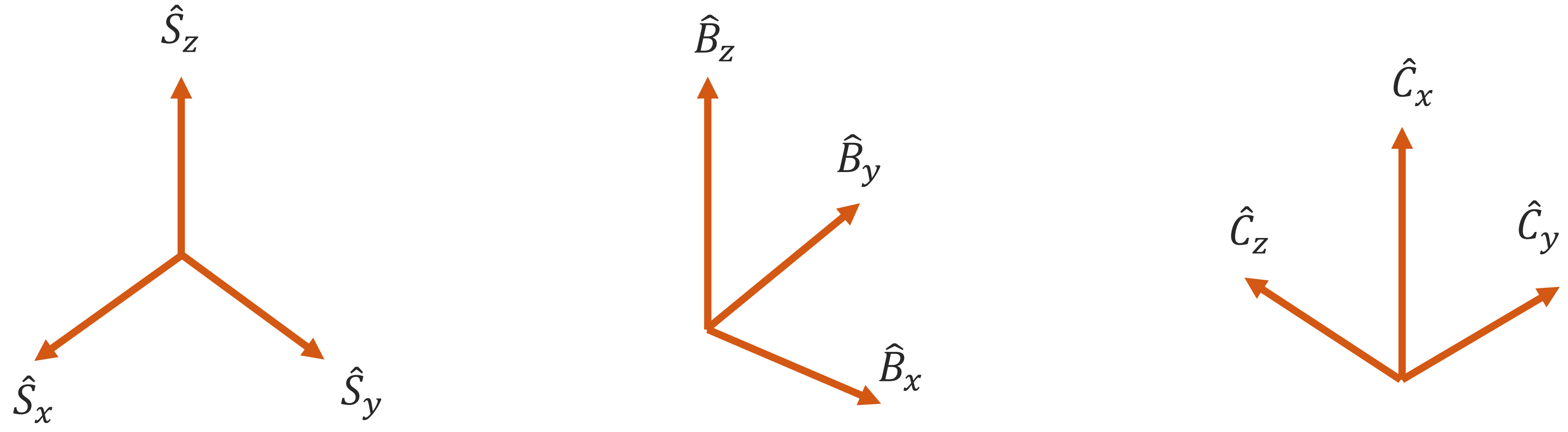
Example:
Expressing a Point in New Coordinate Frame



$$
{}^{\hat{S}}_{\hat{C}}R = \begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & -1 \\ 1 & 0 & 0 \end{bmatrix} \qquad {}^{\hat{C}}p = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}
$$

$$
{}^{\hat{S}}p = {}^{\hat{S}}_{\hat{C}}R\ {}^{\hat{C}}p = \begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & -1 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}
$$

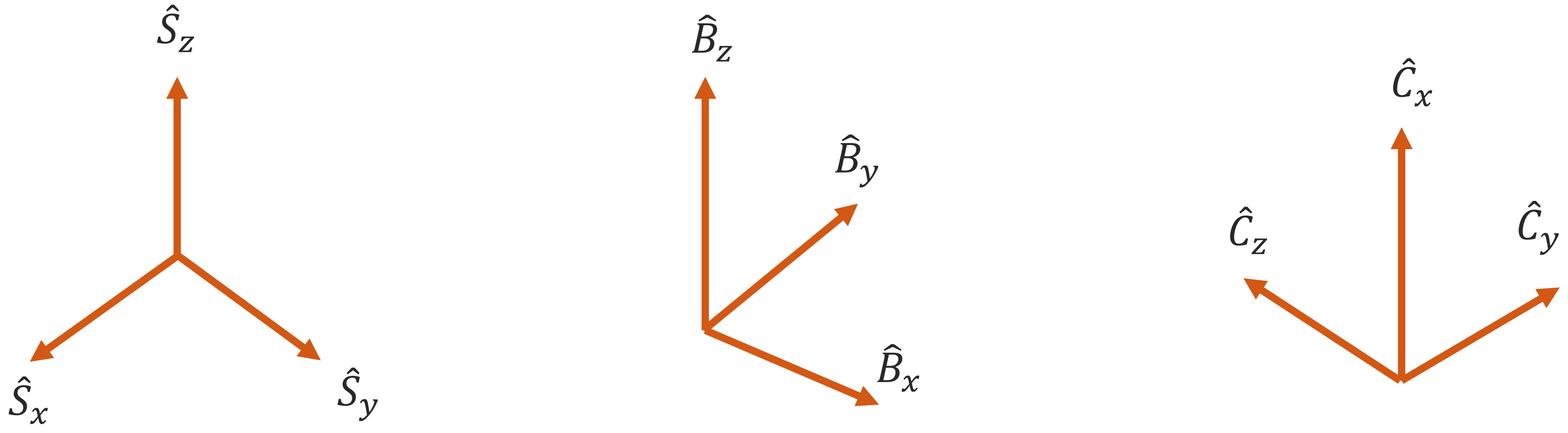
Example: Inverting a Rotation



$$ {}^{\hat{S}}_{\hat{C}}R = \begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & -1 \\ 1 & 0 & 0 \end{bmatrix} \qquad {}^{\hat{C}}_{\hat{S}}R = \begin{bmatrix} 0 & 0 & 1 \\ -1 & 0 & 0 \\ 0 & -1 & 0 \end{bmatrix} $$

Transpose!

Example: Composing Rotations



$$ {}^{\hat{S}}_{\hat{B}}R = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad {}^{\hat{B}}_{\hat{C}}R = \begin{bmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} $$

$$ {}^{\hat{S}}_{\hat{C}}R = {}^{\hat{S}}_{\hat{B}}R\,{}^{\hat{B}}_{\hat{C}}R = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & -1 \\ 1 & 0 & 0 \end{bmatrix} $$

Forward Kinematics of a differential wheel robot (i.e. the e-puck)

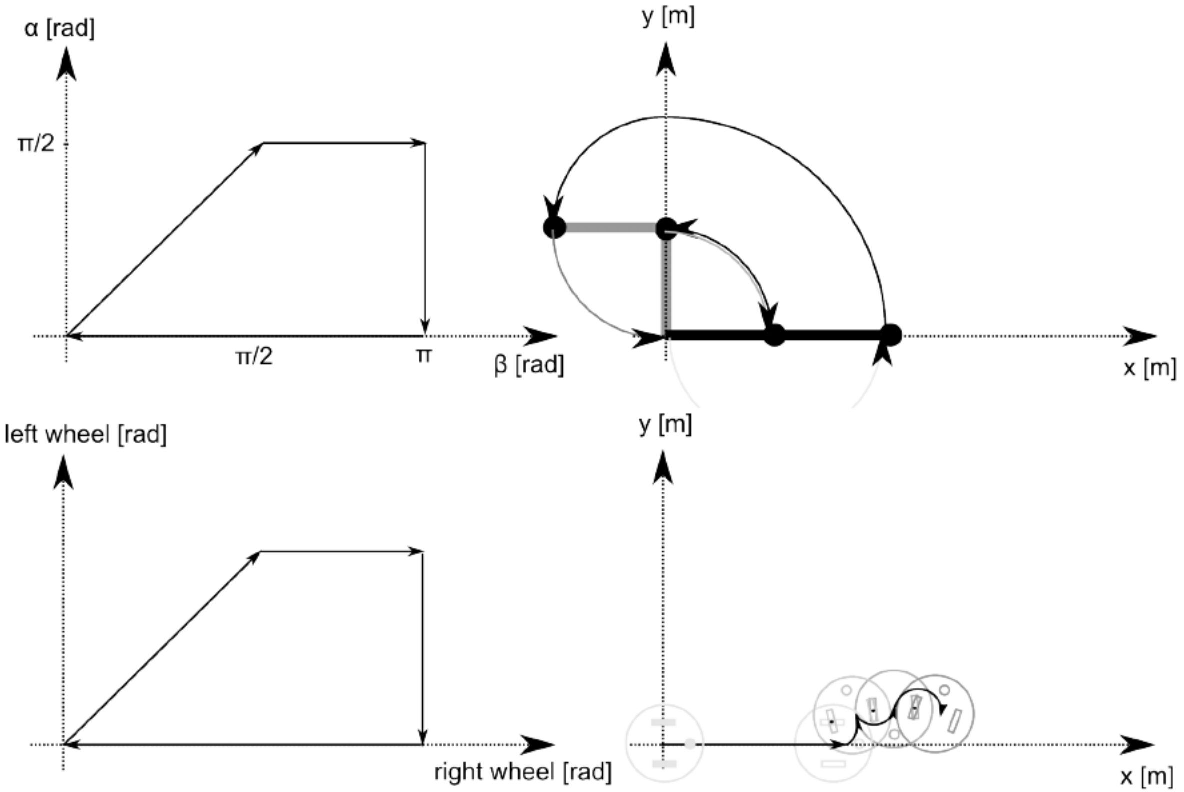
- Manipulator:
  - Forward kinematics is uniquely defined by its joint angles
  - Measure of joint angles through encoders is absolute
- Differential Wheels robot:
  - Measure of joint angles through encoders is relative
  - Encoders' values need to be integrated over time

Please take a look at Section 3.3.2 in the book

Forward Kinematics of a differential wheel robot (i.e. the e-puck)

Manipulator
[HOLONOMIC]

Differential Wheels
[NON-HOLONOMIC]



Configuration [joint] Space

Work [task] space

Please take a look at Section 3.3.2 in the book

Holonomic or Non-Holonomic on the 2D plane?

- Steering wheel is rotated 90 degrees then acceleration is applied for 1 second

vs.

- Acceleration is applied for 1 second then steering wheel rotated 90 degrees!

- Different ending configuration = Non-holonomic!

Forward Kinematics – Odometry

- How to model wheel motion
- Wheel motions to position updates
- Position updates to Forward Kinematics in Inertia frame

Chapter 3.3.2

Odometry

Derived from Greek words for “measure route”

Utilization of sensors to estimate changes in position over time

Useful for position/pose estimation!

Odometry aka: Where is the e-puck?

- How do we figure out where the e-puck is in the world?

What is our state vector?

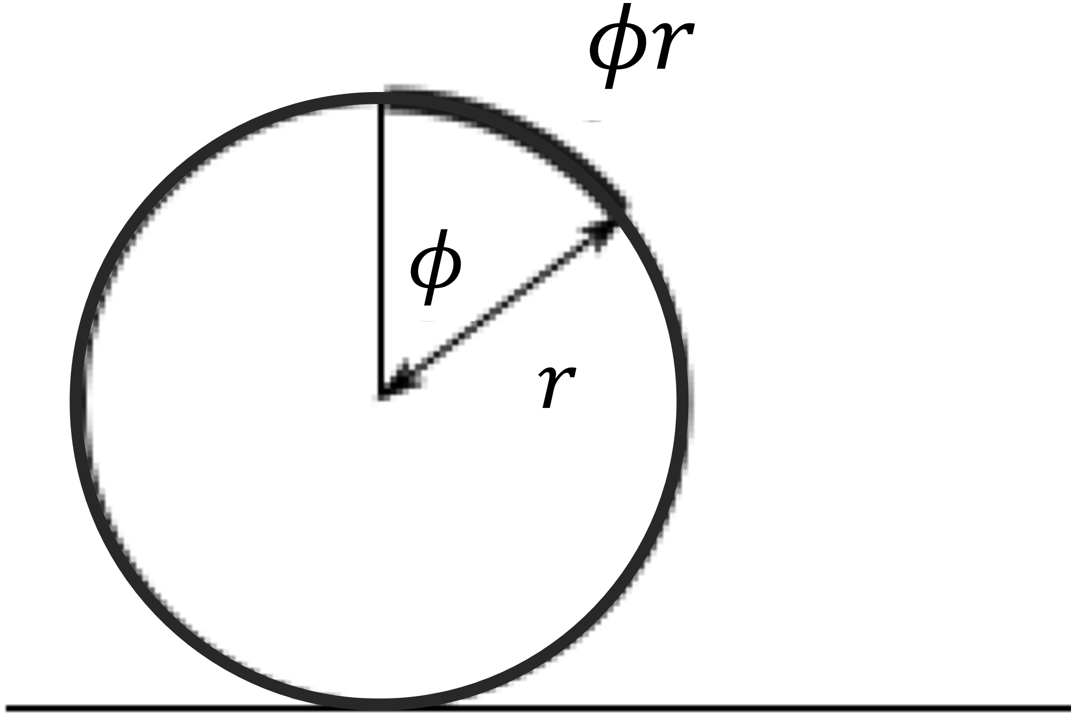
- What are we able to control?
  - [JOINT SPACE DoFs]
- What are we able to measure?
- What variables do we need to measure the robot’s motion in space?
  - [OPERATIONAL SPACE DoFs]

Odometry aka: Where is the e-puck?

- How do we figure out where the e-puck is in the world?

- Measuring the e-puck’s displacement

1. Wheel motion

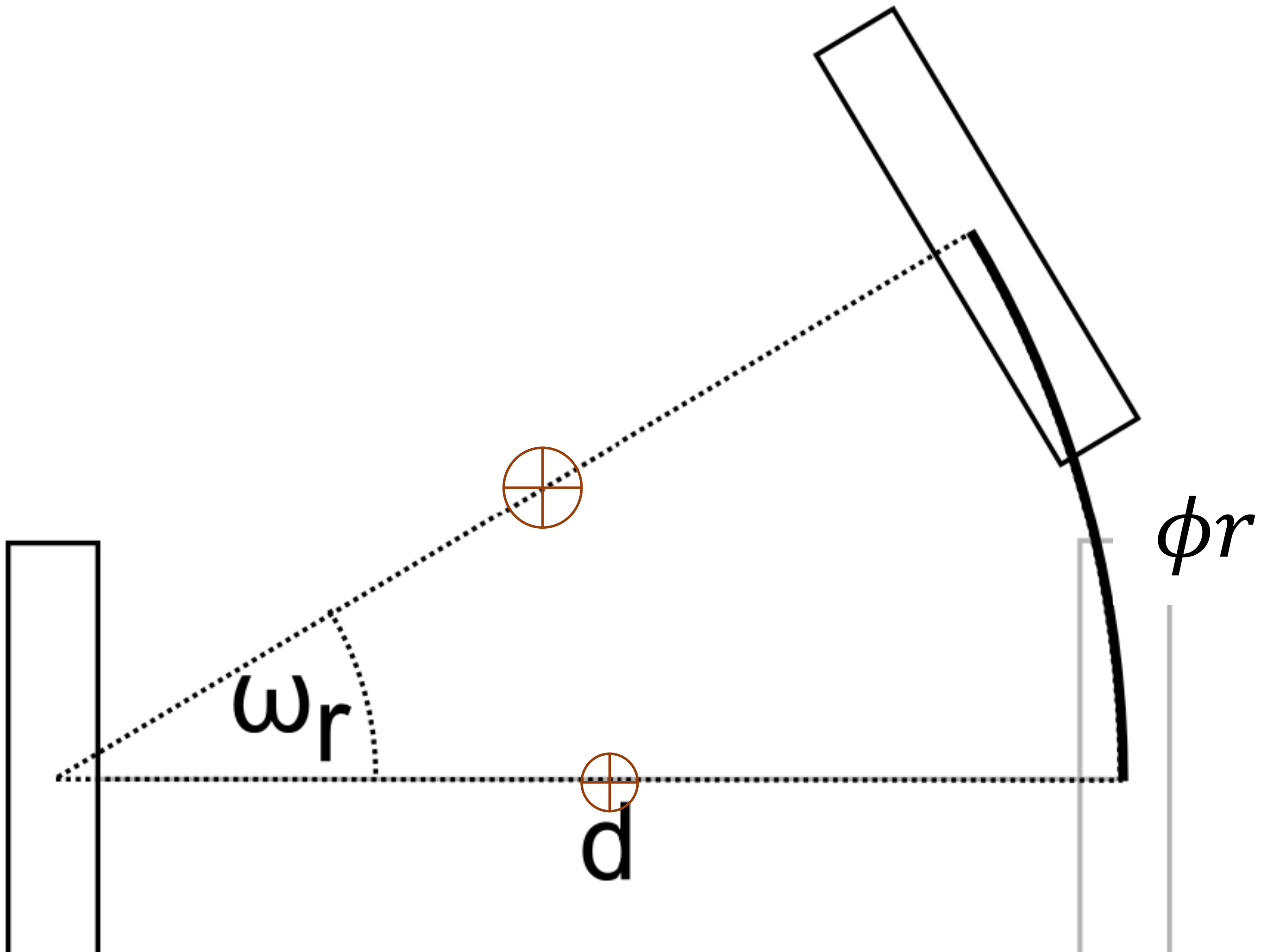


Distance traveled is angle of rotation times wheel radius:

$$x = r\phi$$

$$\dot{x} = r\dot{\phi}$$

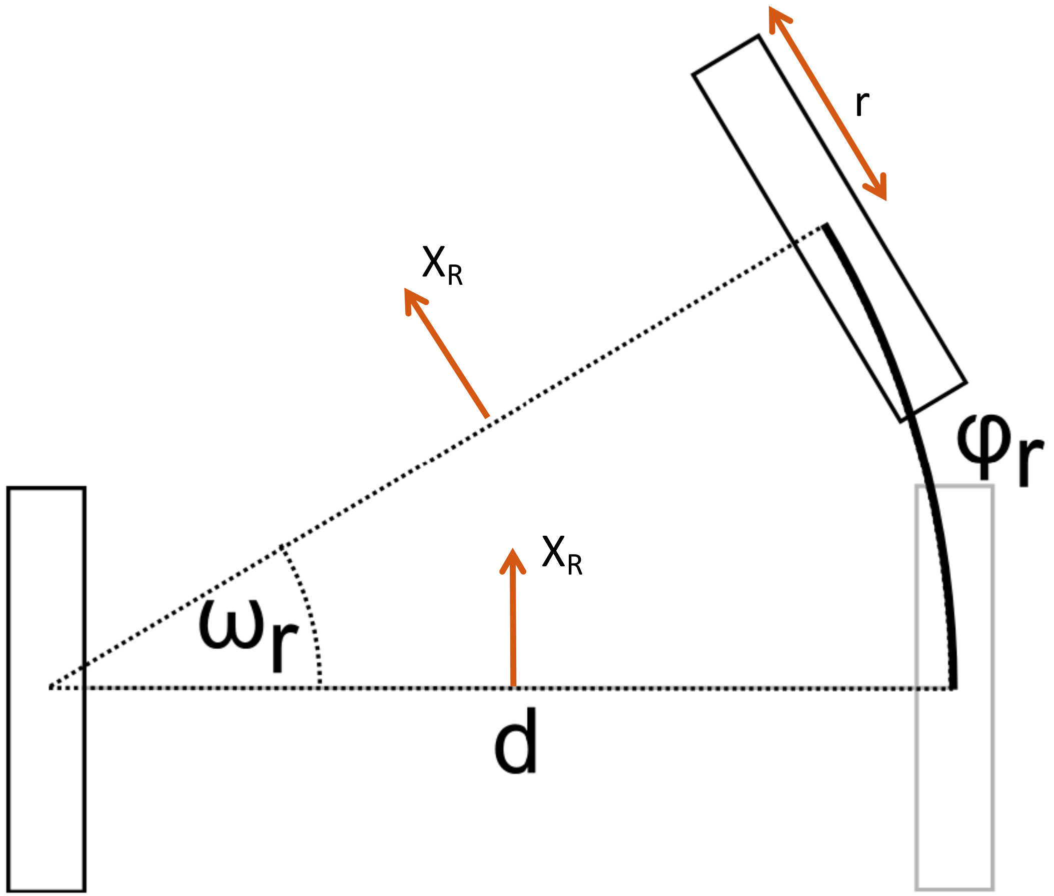
2. Wheel motion → Position Updates



What about the case where only one wheel is moving?

Its center of mass will move by $\frac{1}{2}\phi r$!

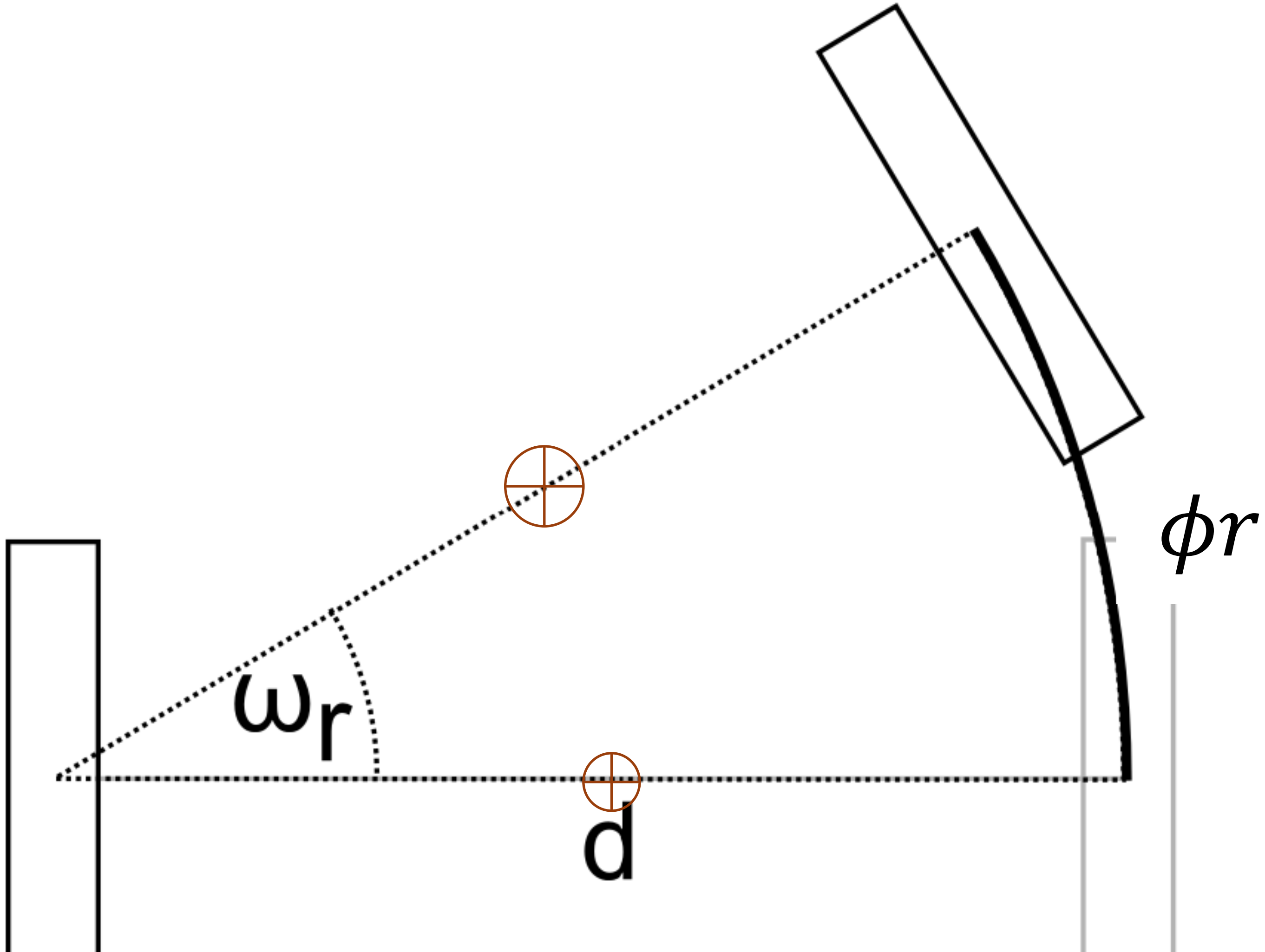
2. Wheel motion → Position Updates



$$\dot{x}_r = \frac{1}{2}\dot{\phi}r$$

$$\omega_r d = \phi_r r \;\rightarrow \dot{\omega}_r = \frac{\dot{\phi}_r r}{d}$$

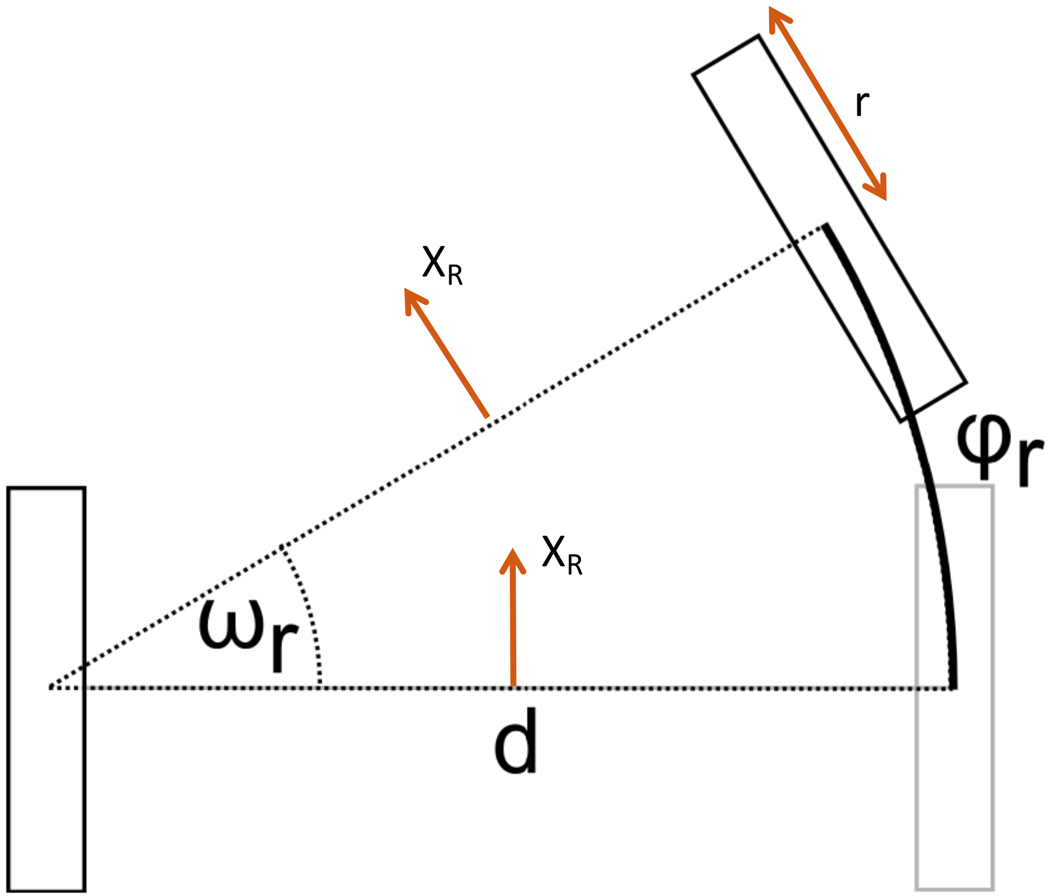
2. Wheel motions → Position Updates



What about the case where both wheels are moving at different speeds $\dot{\phi}_l$ and $\dot{\phi}_r$?

$$\dot{x}_r = \frac{r\dot{\phi}_l}{2} + \frac{r\dot{\phi}_r}{2}$$

2. Wheel motions → Position Updates

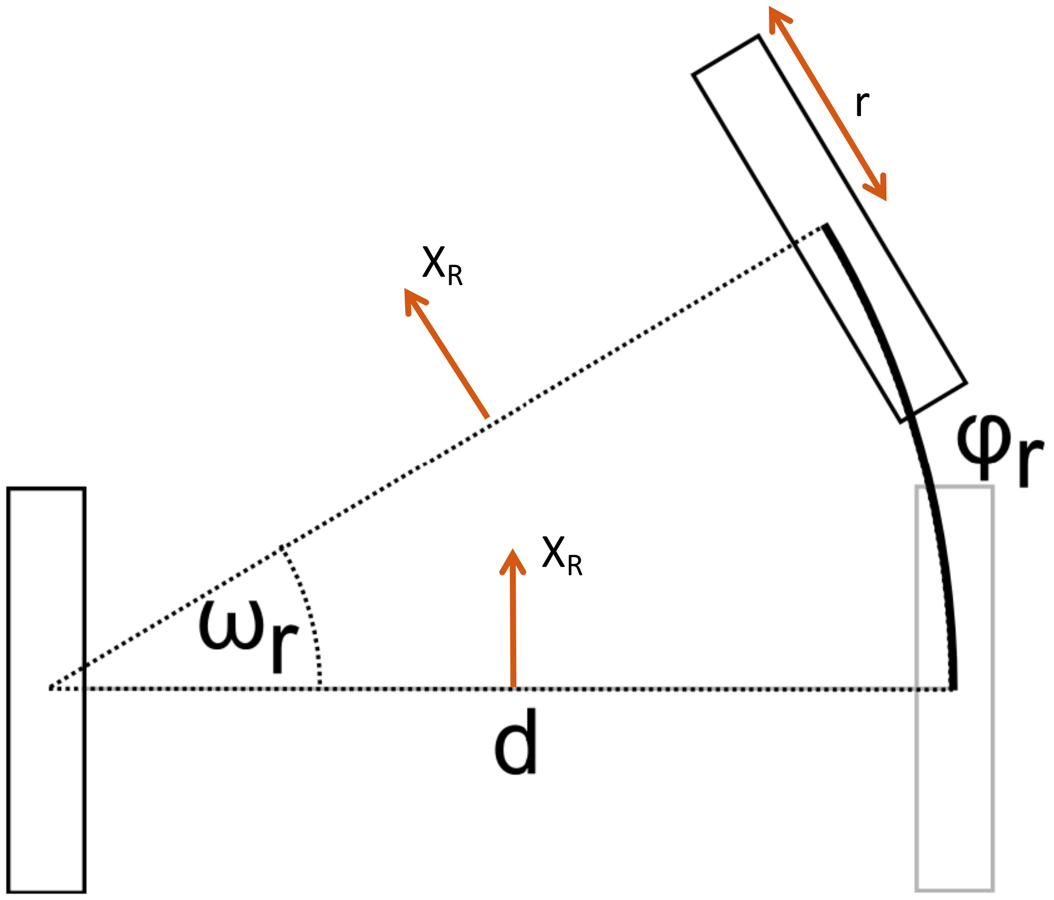


$$\dot{\omega}_r = \frac{\dot{\phi}_r r}{d}$$

$$\dot{\omega}_l = \frac{\dot{\phi}_l r}{d}$$

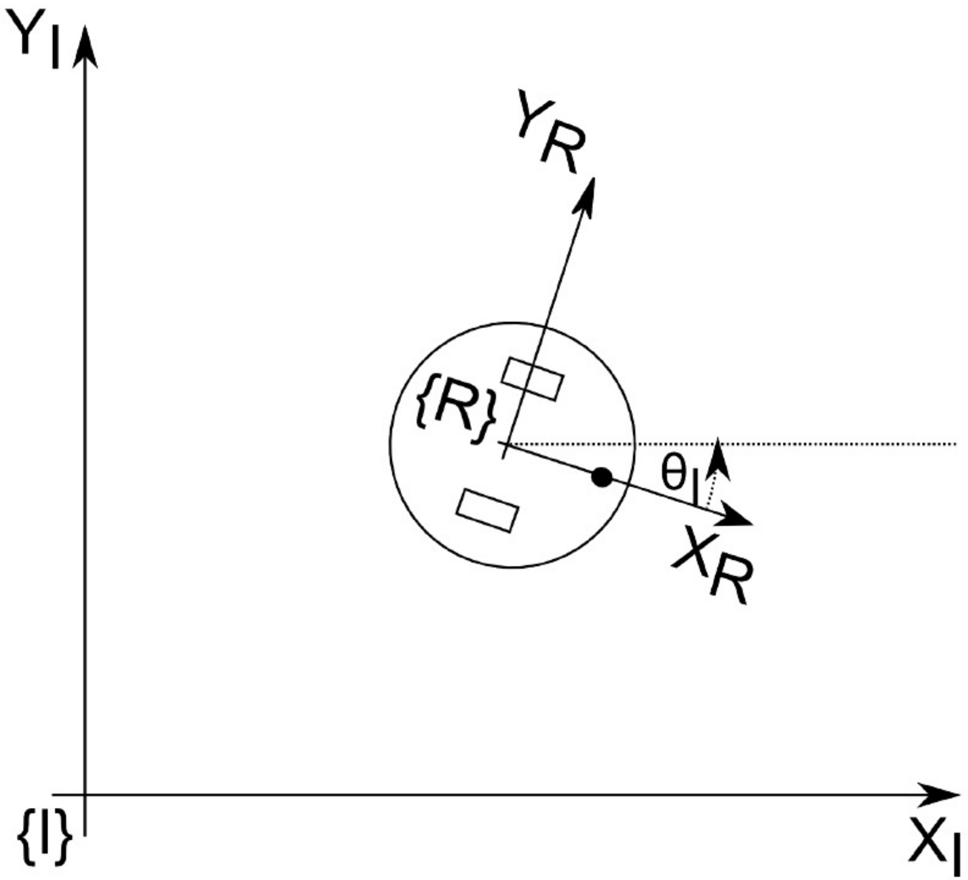
$$\dot{\theta} = \frac{\dot{\phi}_r r}{d} - \frac{\dot{\phi}_l r}{d}$$

2. Forward Kinematics of mobile robot



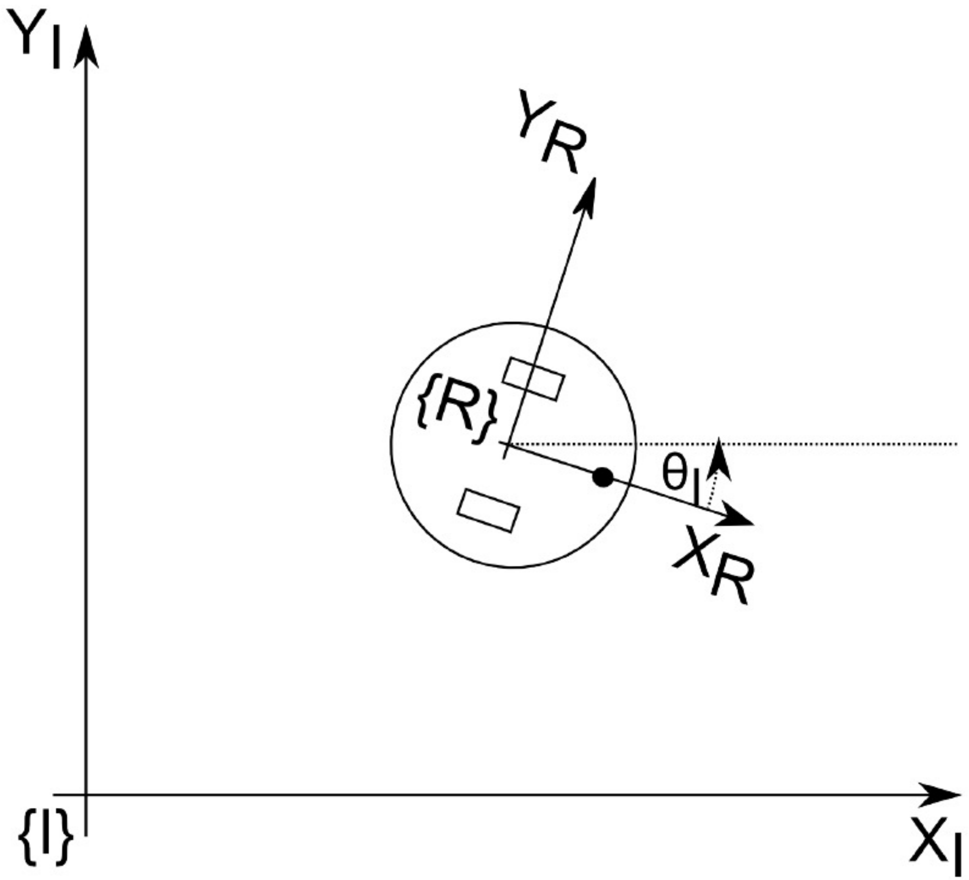
$$\begin{bmatrix} \dot{x}_R \\ \dot{y}_R \\ \dot{\theta}_R \end{bmatrix} = \begin{bmatrix} \frac{r\dot{\phi}_l}{2} + \frac{r\dot{\phi}_r}{2} \\ 0 \\ \frac{\dot{\phi}_r r}{d} - \frac{\dot{\phi}_l r}{d} \end{bmatrix}$$

3. Forward Kinematics + Odometry



$$\begin{bmatrix} \dot{x}_R \\ \dot{y}_R \\ \dot{\theta}_R \end{bmatrix} = \begin{bmatrix} \frac{r\dot{\phi}_l}{2} + \frac{r\dot{\phi}_r}{2} \\ 0 \\ \frac{\dot{\phi}_r r}{d} - \frac{\dot{\phi}_l r}{d} \end{bmatrix}$$

3. Forward Kinematics + Odometry



$$\dot{x}_{I,x} = cos(\theta)\dot{x}_R$$

$$\dot{x}_{I,y} = -sin(\theta)\dot{y}_R$$

$$\dot{x}_I = cos(\theta)\dot{x}_R - sin(\theta)\dot{y}_R$$

$$\dot{y}_I = sin(\theta)\dot{x}_R + cos(\theta)\dot{y}_R$$

$$\dot{\theta}_I = \dot{\theta}_R$$

$$\begin{pmatrix} \dot{x}_I \\ \dot{y}_I \\ \dot{\theta} \end{pmatrix} = \begin{pmatrix} cos(\theta) & -sin(\theta) & 0 \\ sin(\theta) & cos(\theta) & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \frac{r\dot{\phi}_l}{2} + \frac{r\dot{\phi}_r}{2} \\ 0 \\ \frac{\dot{\phi}_r r}{d} - \frac{\dot{\phi}_l r}{d} \end{pmatrix}$$

4. Speeds → How can we compute positions?

$$\begin{pmatrix} x_I(T) \\ y_I(T) \\ \theta(T) \end{pmatrix} =$$

Don’t forget…

- Take the quiz…
- Check out Homework 0
- Submit lab 1
- See you tomorrow