Lecture 8: Inverse Kinematics

Iñigo Iturrate

Assistant Professor

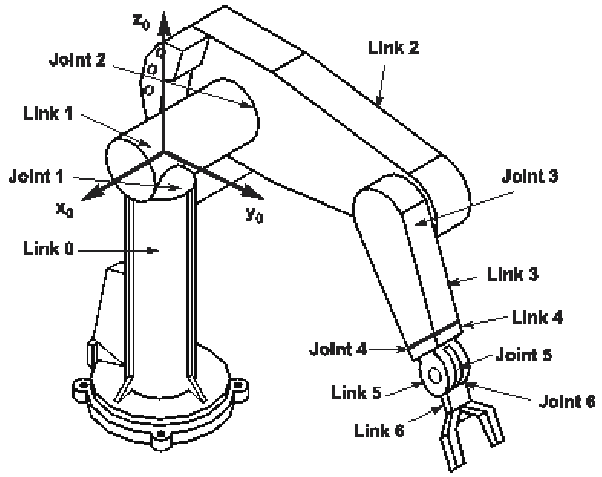
SDU Robotics,

The Maersk McKinney Moller Institute,

University of Southern Denmark

Ø27-604-3

inju@mmmi.sdu.dk



SDU ROBOTICS

What are we doing today?

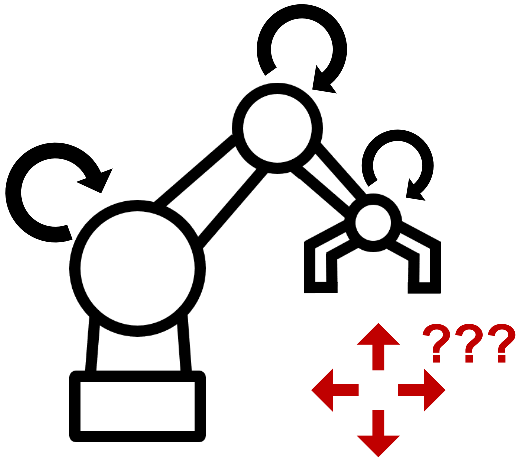
1. CAD Modelling in Autodesk Inventor – *Guest Lecture by Aljaz Kramberger*
2. CAD Assemblies in Autodesk Inventor – *Guest Lecture by Aljaz Kramberger*
3. Introduction to Robotics & Recap of Linear Algebra and Mathematical Notation
4. Translations & Rotation Matrices
5. Other Representations for Orientation
6. Transformation Matrices
7. DH Parameters & Forward Kinematics
8. **Inverse Kinematics** ***(Today)***
9. Kinematic Simulation
10. Velocity Kinematics & the Jacobian Matrix
11. More about the Jacobian & Trajectory Generation
12. Manipulability, More on the Robotic Systems Toolbox

Last time we saw…

We can **describe** a robot's **kinematic structure using DH parameters**.

Forward Kinematics describes how **motions of the joints translate into motion of the end-effector**:

- We can obtain the **Forward Kinematics from the DH parameters.**
- We can also derive the **forward kinematics from transformations** between frames.

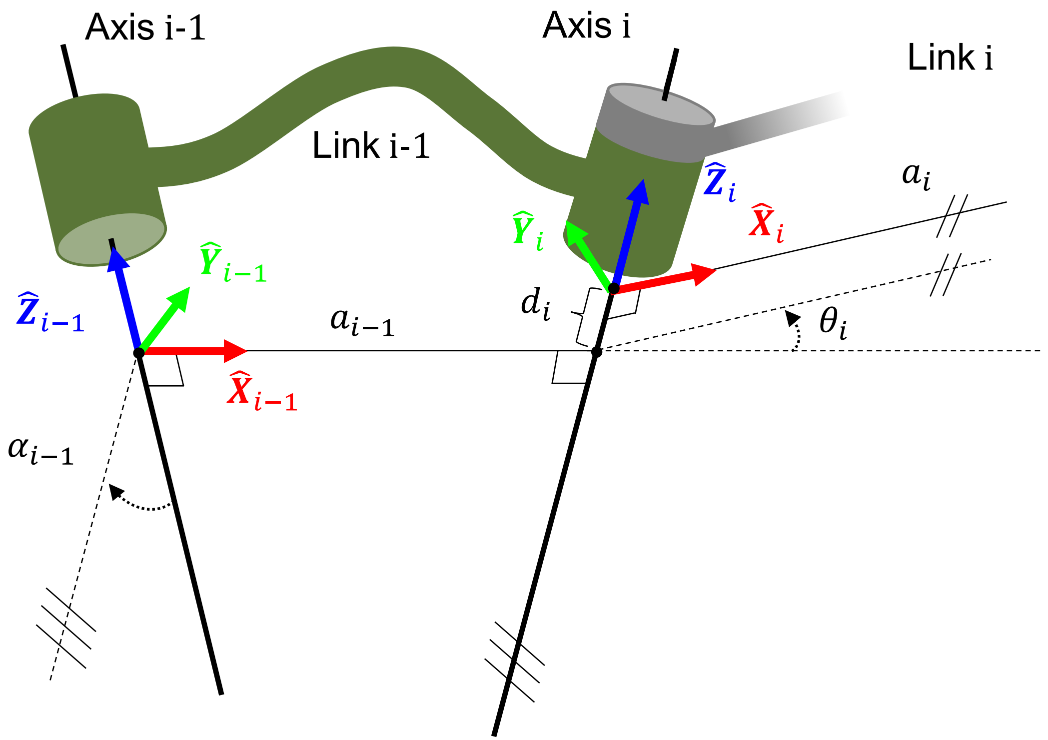


Recap: DH Parameters

Using **Modified** (Craig) **DH Parameters**:

We can describe a whole robot by **attaching frames to links**:

- **Z-axis** of frame $\{i\}$
  is coincident with joint axis i.
- **X-axis** of frame $\{i\}$
  points along a_i in the direction from joint i to joint $i+1$.



Then, we can **assign four parameters to each link**:

- $\boldsymbol{\alpha_i}$ – The angle from $\hat{Z}_i$ to $\hat{Z}_{i+1}$ measured about $\hat{X}_i$
- $\boldsymbol{a_i}$ – The distance from $\hat{Z}_i$ to $\hat{Z}_{i+1}$ measured along $\hat{X}_i$
- $\boldsymbol{d_i}$ – The distance from $\hat{X}_{i-1}$ to $\hat{X}_i$ measured along $\hat{Z}_i$
- $\boldsymbol{\theta_i}$ – The angle from $\hat{X}_{i-1}$ to $\hat{X}_i$ measured about $\hat{Z}_i$

Recap: FK from Modified DH Parameters

These variables are <u>always</u> fixed

This is fixed for a revolute joint

This is fixed for a prismatic joint

$$ {}^{i-1}_{i}\boldsymbol{T} = \boldsymbol{R}_X(\alpha_{i-1})\boldsymbol{D}_X(a_{i-1})\boldsymbol{D}_Z(d_i)\boldsymbol{R}_Z(\theta_i) $$

$$ = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c\alpha_{i-1} & -s\alpha_{i-1} & 0 \\ 0 & s\alpha_{i-1} & c\alpha_{i-1} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & a_{i-1} \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c\theta_i & -s\theta_i & 0 & 0 \\ s\theta_i & c\theta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ = \begin{bmatrix} c\theta_i & -s\theta_i & 0 & a_{i-1} \\ s\theta_i c\alpha_{i-1} & c\theta_i c\alpha_{i-1} & -s\alpha_{i-1} & -s\alpha_{i-1} d_i \\ s\theta_i s\alpha_{i-1} & c\theta_i s\alpha_{i-1} & c\alpha_{i-1} & c\alpha_{i-1} d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

SDU ROBOTICS

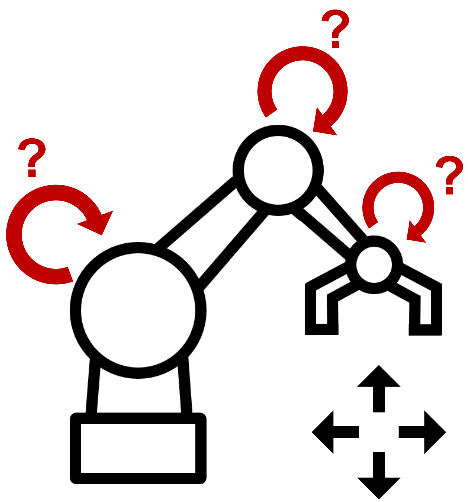
Today…

What if we want to do **the opposite**?

In most tasks, we will have a **defined task motion** that the robot needs to follow in Cartesian space, **not a defined joint space motion**.

The problem then becomes:

> “Given a **target position and orientation** of the robot **end-effector**, **what joint position** values will **reach that target**?”



Topics for Today

Part I: Joint and Cartesian Spaces

- Joint space & Cartesian space
- Mapping between them (Forward and Inverse Kinematics)

Part II: Inverse Kinematics

- Closed-form vs. Numerical solutions
- Examples of closed-form solutions for 2R Planar Manipulator & Universal Robots (6R Manipulator)

Part III: Practical Considerations

- Workspace in relation to IK
- DOF vs number of IK solutions
- Choice of IK solutions
- Repeatability and Accuracy

Part I: Joint and Cartesian Spaces & Mapping between Them

Joint & Cartesian Spaces

In robotics, we are constantly working with **two different spaces**:

- **Joint (or Operational) space**
  - The space where the robot actually operates.
  - Defines **how its joints** (and their associated **motors**) **actually move**.
  - It is the space **we have direct (low-level) control over**.

- **Cartesian (or Task) space**
  - "Our world", where we want the robot to actually act and perform a task.
  - Defines **how the end-effector/tool** of the robot **should move**.
  - We **cannot control it directly** (at a low level).

Joint (Configuration) Space

The **dimensionality** of the space **will depend on** the **number and kind of joints (i.e. DOF)** in the robot.

For an n-DOF robot:

- **Positions** in joint space are given by $\boldsymbol{q} \in \mathbb{R}^n$:

$$\boldsymbol{q} = \begin{bmatrix} q_1 \\ \vdots \\ q_n \end{bmatrix}$$

- **Velocities** are given by $\dot{\boldsymbol{q}} \in \mathbb{R}^n$:

$$\dot{\boldsymbol{q}} = \begin{bmatrix} \dot{q}_1 \\ \vdots \\ \dot{q}_n \end{bmatrix}$$

- **Accelerations** are given by $\ddot{\boldsymbol{q}} \in \mathbb{R}^n$:

$$\ddot{\boldsymbol{q}} = \begin{bmatrix} \ddot{q}_1 \\ \vdots \\ \ddot{q}_n \end{bmatrix}$$

Where each q_i is:
- An angle θ for revolute joints
- A displacement d for prismatic joints

Cartesian (Task) Space

The **dimensionality** of the space **will depend on** the kind of task. Usually, we will work in a 3D environment.

For a 3D environment:

- **Poses** in Cartesian space are usually described by $\mathbf{x} \in \mathbb{R}^6$:

$$\boldsymbol{x} = \begin{bmatrix} \boldsymbol{p} \\ \boldsymbol{\phi} \end{bmatrix}$$

Where:

- $\boldsymbol{p} \in \mathbb{R}^3$ is a position vector $[p_x, p_y, p_z]^T$
- $\boldsymbol{\phi} \in \mathbb{R}^3$ or $\boldsymbol{\phi} \in SO(3)$ is an orientation vector $[\phi_x, \phi_y, \phi_z]^T$ *(which can be in many representations)*

- **Velocities** are usually given by $\dot{\boldsymbol{x}} \in \mathbb{R}^6$:

$$\dot{\boldsymbol{x}} = \boldsymbol{v} = \begin{bmatrix} \dot{\boldsymbol{p}} \\ \dot{\boldsymbol{\phi}} \end{bmatrix} \text{ or } \dot{\boldsymbol{x}} = \begin{bmatrix} \dot{\boldsymbol{p}} \\ \boldsymbol{\omega} \end{bmatrix}$$

Where:

- $\dot{\boldsymbol{\phi}} \in \mathbb{R}^3$ is the time-derivative of the orientation vector
- $\boldsymbol{\omega} \in \mathbb{R}^3$ is an angular velocity

- **Accelerations** are usually given by $\ddot{\boldsymbol{x}} \in \mathbb{R}^6$:

$$\ddot{\boldsymbol{x}} = \boldsymbol{a} = \begin{bmatrix} \ddot{\boldsymbol{p}} \\ \ddot{\boldsymbol{\phi}} \end{bmatrix} \text{ or } \ddot{\boldsymbol{x}} = \begin{bmatrix} \ddot{\boldsymbol{p}} \\ \dot{\boldsymbol{\omega}} \end{bmatrix}$$

Different Cases with Mappings

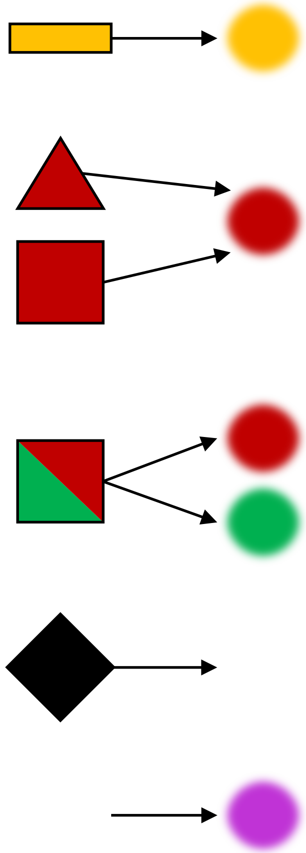
One-to-one:

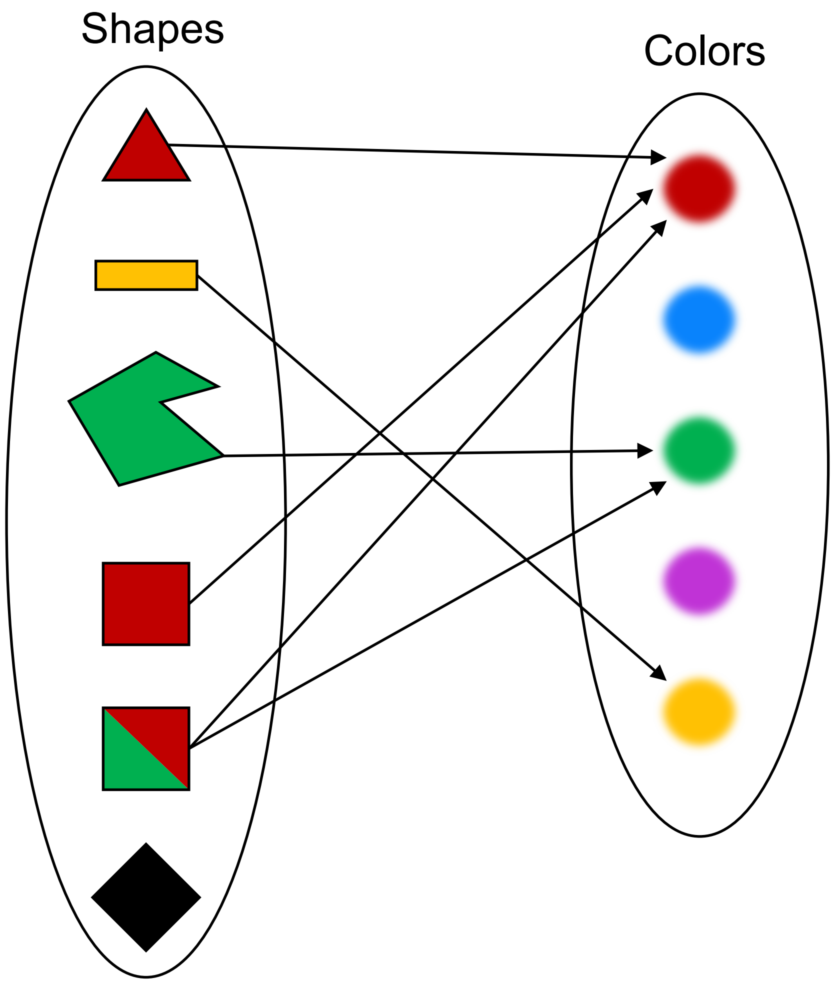
Many-to-one:

One-to-many:

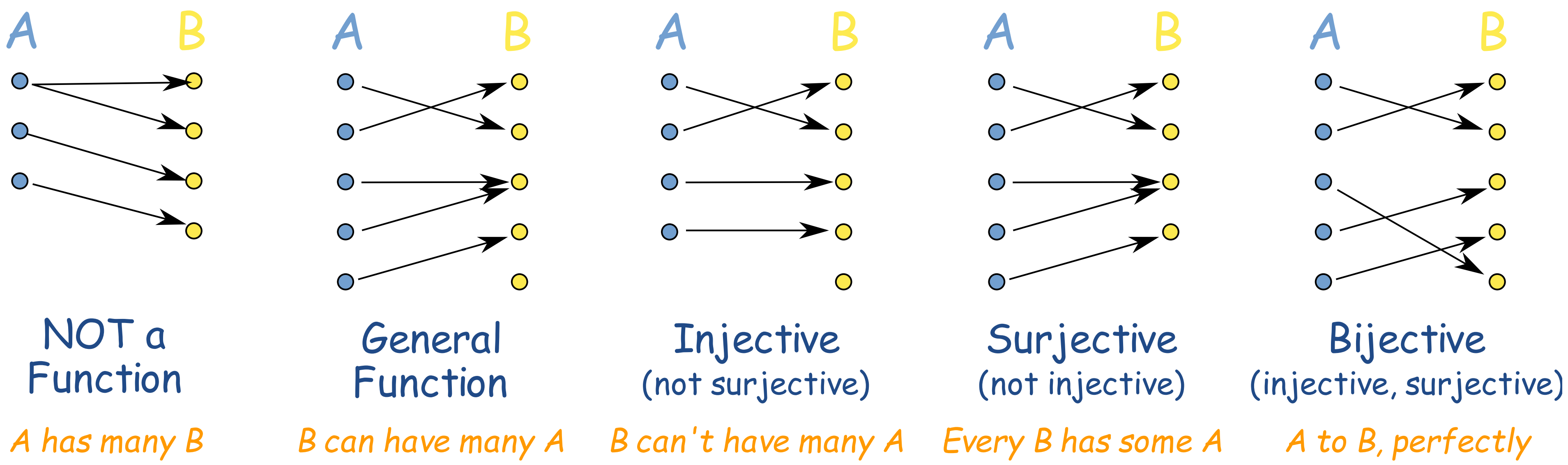
One-to-none:

None-to-one:





Different Cases with Mappings (More Formally)



Source: https://www.mathsisfun.com/sets/injective-surjective-bijective.html

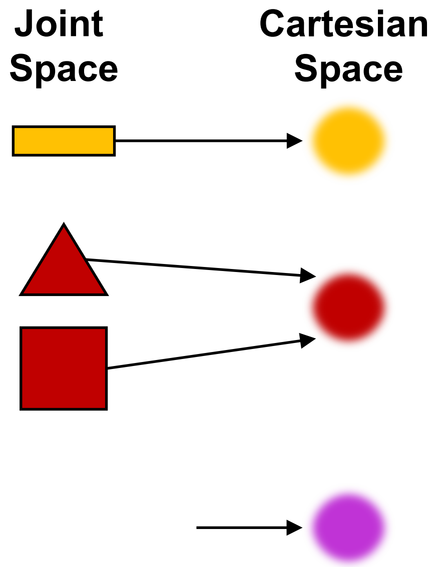
Forward Kinematics (Joint → Cartesian)

Forward Kinematics is the name given to the **mapping from joint space to Cartesian space**.

In other words, **if we know the position of the joints, what is the position of the end-effector?**

This is the easy problem.

Why? Because the mappings will be:



- One-to-one
- Many-to-one
- None-to-one

Inverse Kinematics (Cartesian → Joint)

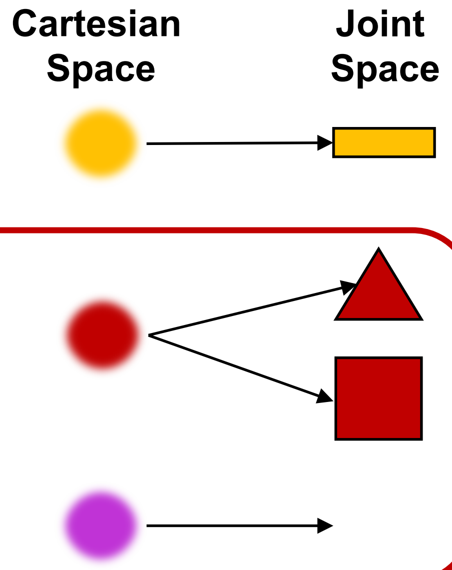
Inverse Kinematics is the name given to the **mapping from Cartesian space to joint space**.

In other words, **if we know the position of the end-effector, how should we place the joints?**

This is <u>the HARD problem</u>.

Why? Because the mappings will be:

- One-to-one
- One-to-many
- One-to-none



And this can be nasty!

SDU ROBOTICS

Part II: Inverse Kinematics

Inverse Kinematics as a Mathematical Problem

The forward kinematics of a robot are specified by a transformation matrix: ${}^{Base}_{Tool}\boldsymbol{T}(\boldsymbol{\theta}) = \begin{bmatrix} r_{11} & r_{12} & r_{13} & p_x \\ r_{21} & r_{22} & r_{23} & p_y \\ r_{31} & r_{32} & r_{33} & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$

Let’s take a Universal Robots 6R manipulator as an example.

We can write a **system of 12 equations with 6 unknowns**[1]:

$$\begin{bmatrix} r_{11} \\ r_{12} \\ \vdots \\ p_z \end{bmatrix} = \begin{bmatrix} \cos(\theta_6)\big(\sin(\theta_1)\sin(\theta_5) + \cos(\theta_2+\theta_3+\theta_4)\cos(\theta_1)\cos(\theta_5)\big) - \sin(\theta_2+\theta_3+\theta_4)\cos(\theta_1)\sin(\theta_6) \\ -\sin(\theta_6) * (\sin(\theta_1) * \sin(\theta_5) + \cos(\theta_2+\theta_3+\theta_4)\cos(\theta_1) * \cos(\theta_5)) - \sin(\theta_2+\theta_3+\theta_4)\cos(\theta_1)\cos(\theta_6) \\ \vdots \\ d_1 + d_5\big(\sin(\theta_2+\theta_3)\sin(\theta_4) - \cos(\theta_2+\theta_3)\cos(\theta_4)\big) + a_3\sin(\theta_2+\theta_3) + a_2\sin(\theta_2) - d_6\sin(\theta_5)(\cos(\theta_2+\theta_3)\sin(\theta_4) + \sin(\theta_2+\theta_3)\cos(\theta_4)) \end{bmatrix}$$

Solving this for $[\boldsymbol{\theta_1}, \boldsymbol{\theta_2}, \cdots, \boldsymbol{\theta_6}]$ will give us the Inverse Kinematics.

Only 3 of the 9 equations for the **rotation are independent → We end up with 6 equations/6 unknowns.**

SDU ROBOTICS

[1]This comes from the FK, which you have calculated using DH Parameters.

Closed-form vs. Numerical Solutions

Note that the equations **IK** are **non-linear and transcendental**!

There are **two main approaches** to solving the **inverse kinematics** problem:

- **Closed-form**: An analytical solution based on the forward kinematics transform equations.
  - \+ Fast to compute
  - \+ Exact
  - \- Does not exist for all robots
  - \- Can be complex to calculate
  - \- Needs to be calculated for each specific robot

- **Numerical**: A numerical solution based on an approximation and iterative attempts.
  - \+ Possible for all robots
  - \+ The same method can be used generally for any robot
  - \- Slow to compute
  - \- Inexact

We will not go into this today.

IK of 2R Planar Manipulator

Objective: Obtain (q_1, q_2)

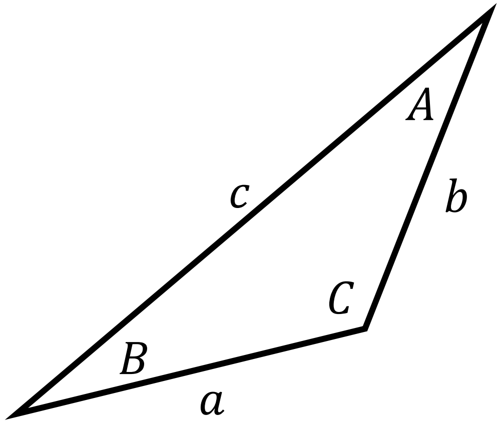
Given:

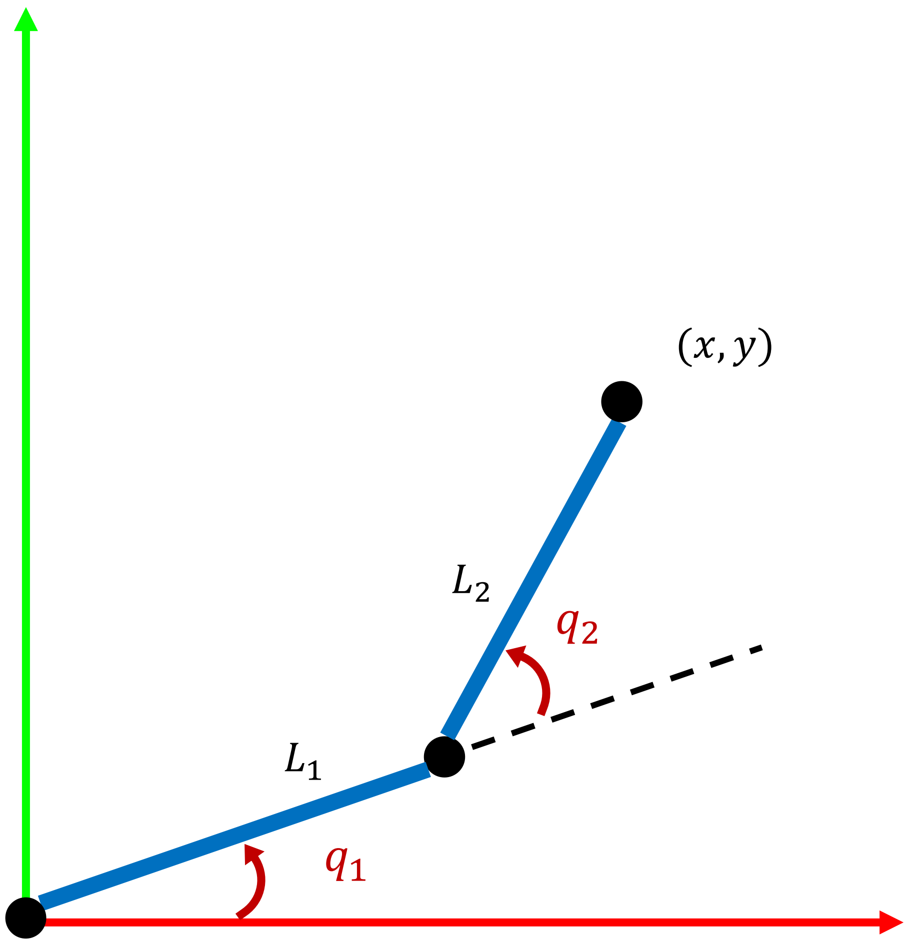
- End-effector position: (x, y)
- Length of links: L_1 and L_2

Exercise: 10-15 minutes

Hint: Use the rule of cosines:

$$a^2 = b^2 + c^2 - 2bc\cos(A)$$
$$b^2 = a^2 + c^2 - 2ac\cos(B)$$
$$c^2 = a^2 + b^2 - 2ab\cos(C)$$

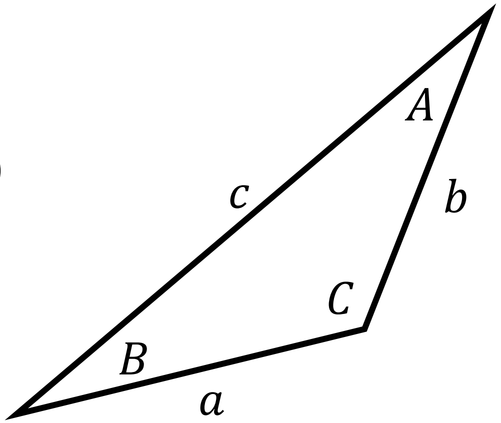




IK of 2R Planar Manipulator: Solution for q_2

$$c^2 = a^2 + b^2 - 2ab\cos(C)$$

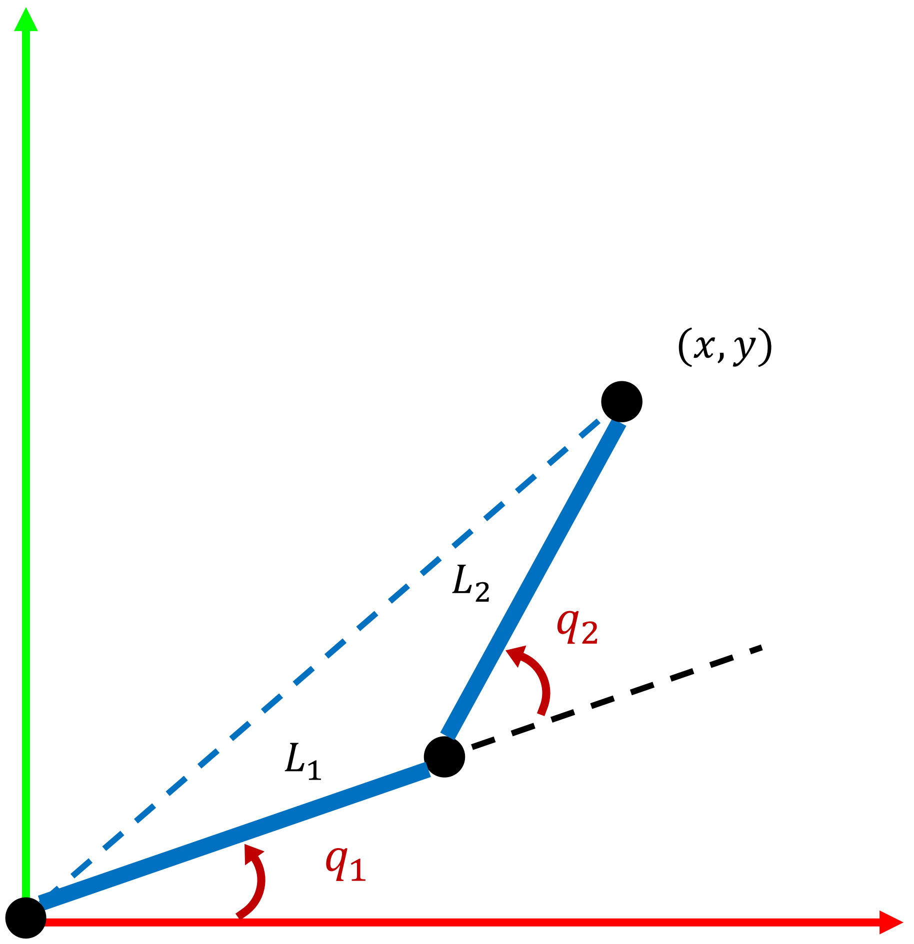
$$\cos(C) = \frac{a^2 + b^2 - c^2}{2ab}$$



$$\cos(\pi - q_2) = \frac{{L_1}^2 + {L_2}^2 - \left(\sqrt{x^2 + y^2}\right)^2}{2L_1L_2}$$

$$\cos(q_2) = -\frac{{L_1}^2 + {L_2}^2 - x^2 - y^2}{2L_1L_2}$$

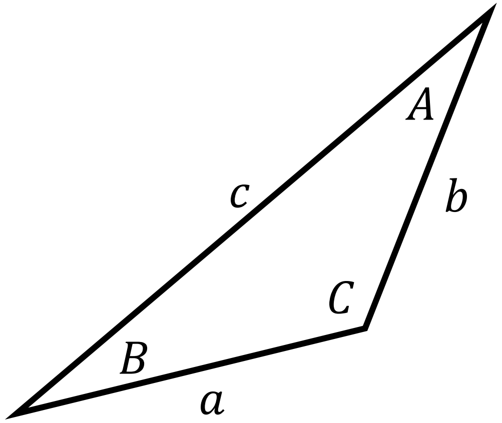
$$\boxed{q_2 = \pm\,\mathrm{acos}\left(\frac{x^2 + y^2 - {L_1}^2 - {L_2}^2}{2L_1L_2}\right)}$$



IK of 2R Planar Manipulator: Solution for q_1

$$q_1 = \varphi - B$$

$$\varphi = \mathrm{atan2}(y, x)$$

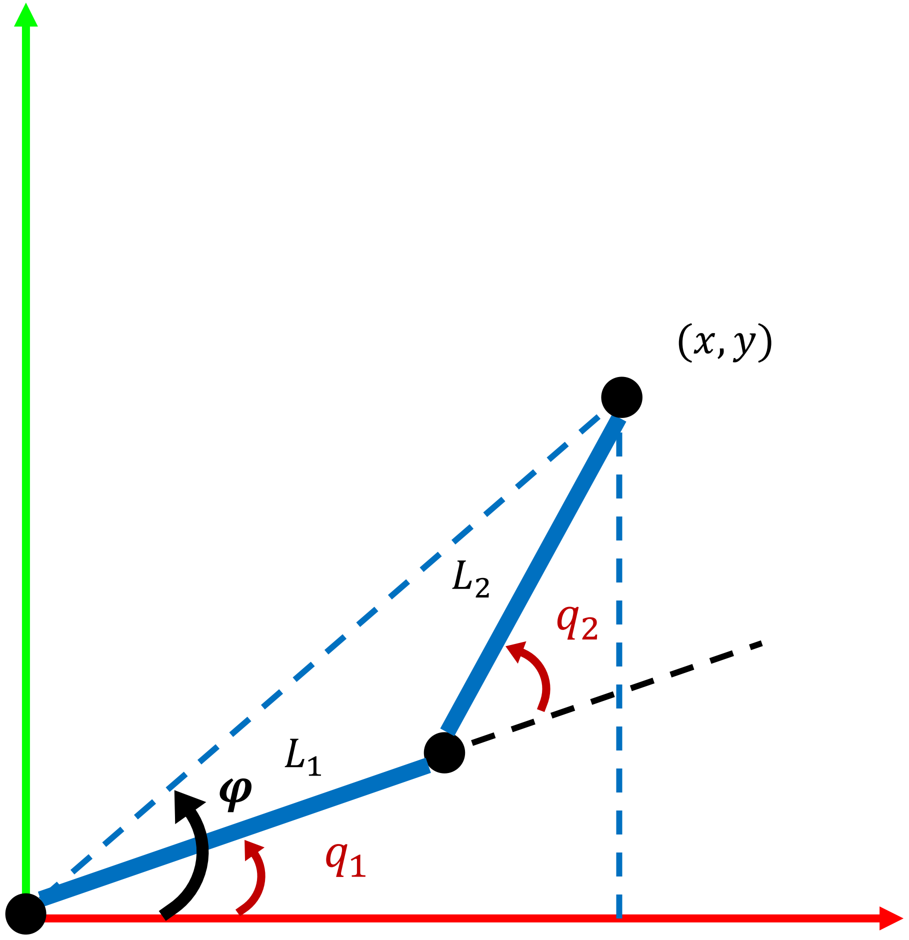


Using law of cosines:

$$b^2 = a^2 + c^2 - 2ac\cos(B)$$

$$B = \pm\,\mathrm{acos}\left(\frac{{L_1}^2 + x^2 + y^2 - {L_2}^2}{2L_1\sqrt{x^2 + y^2}}\right)$$

$$\boxed{q_1 = \mathrm{atan2}(y, x) \mp \mathrm{acos}\left(\frac{{L_1}^2 + x^2 + y^2 - {L_2}^2}{2L_1\sqrt{x^2 + y^2}}\right)}$$



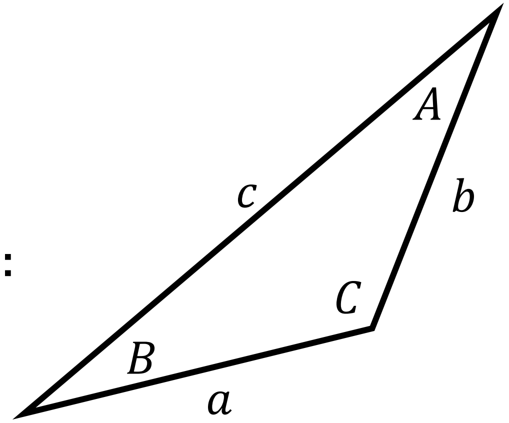
IK of 2R Planar Manipulator: Alternate Solution for q_1

$$q_1 = \varphi - B$$

$$\varphi = \text{atan2}(y, x)$$

Using law of sines (alternate solution):

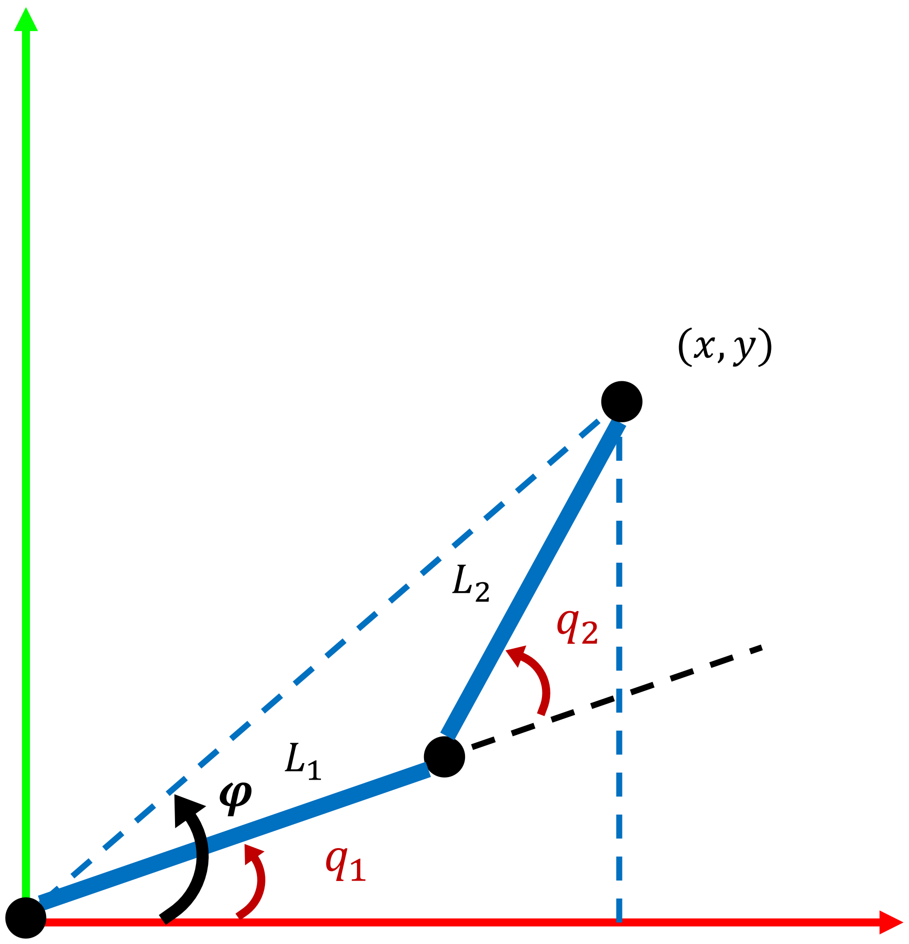
$$\frac{a}{\sin A} = \frac{b}{\sin B} = \frac{c}{\sin C}$$



$$B = \text{asin}\left(\frac{b \sin C}{c}\right)$$

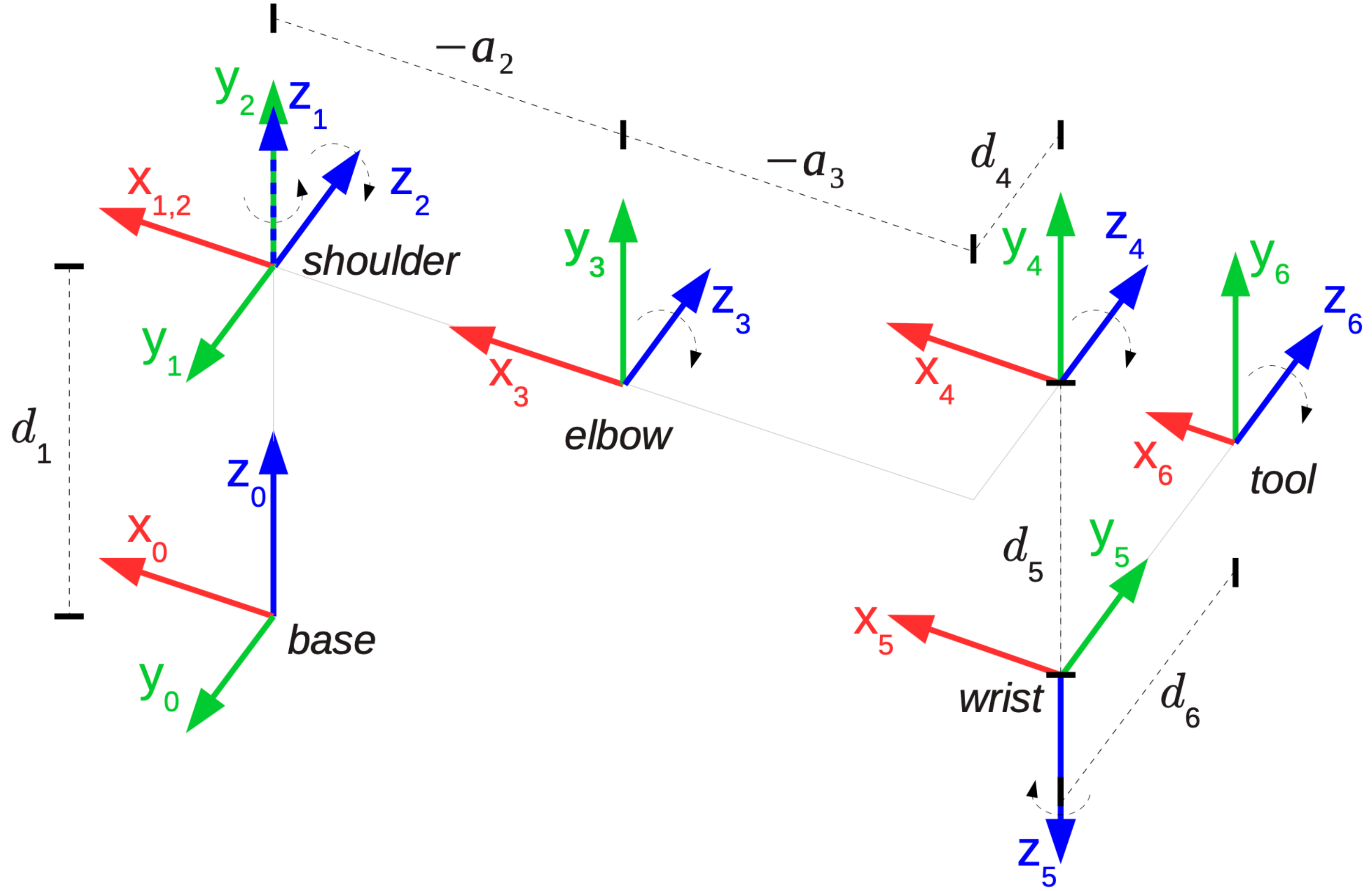
$$B = \text{asin}\left(\frac{L_2 \sin(\pi - q_2)}{c}\right) = \text{asin}\left(\frac{L_2 \sin(q_2)}{c}\right)$$

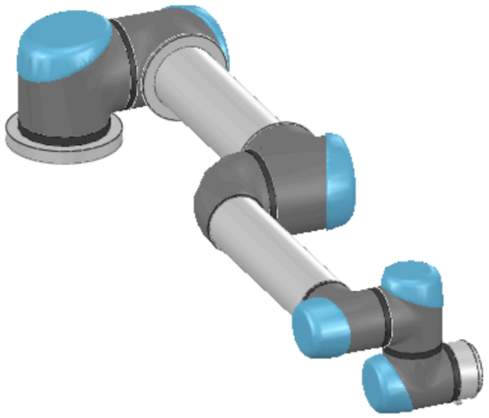
$$\boxed{q_1 = \text{atan2}(y, x) \mp \text{asin}\left(\frac{L_2 \sin(q_2)}{c}\right)}$$



Example: IK of Universal Robots 6R Manipulator

We start with the **frame assignment** according to the **modified DH parameter** convention.





Based on ”Kinematics of a UR5” by Rasmus Skovgaard Andersen, Aalborg University

Example UR IK: Finding θ_1 (I)

First, notice that we can **find frame 5 from frame 6**:

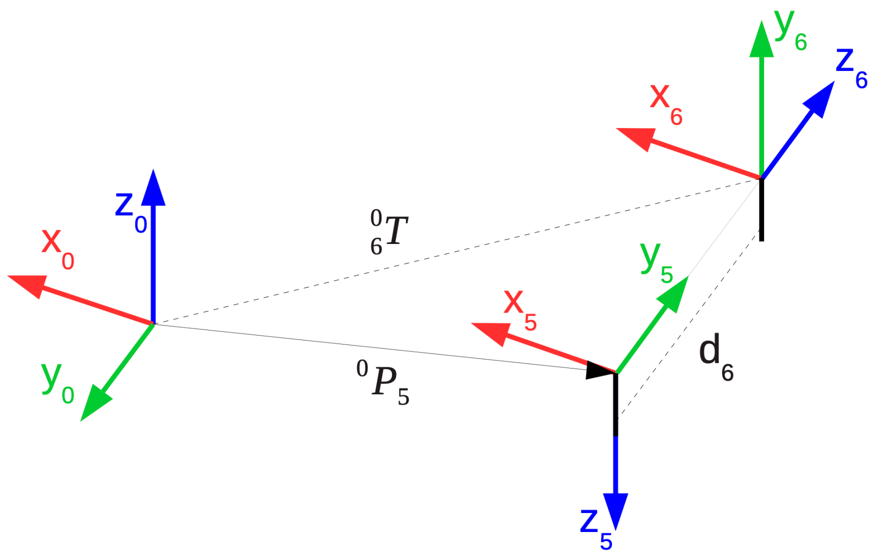


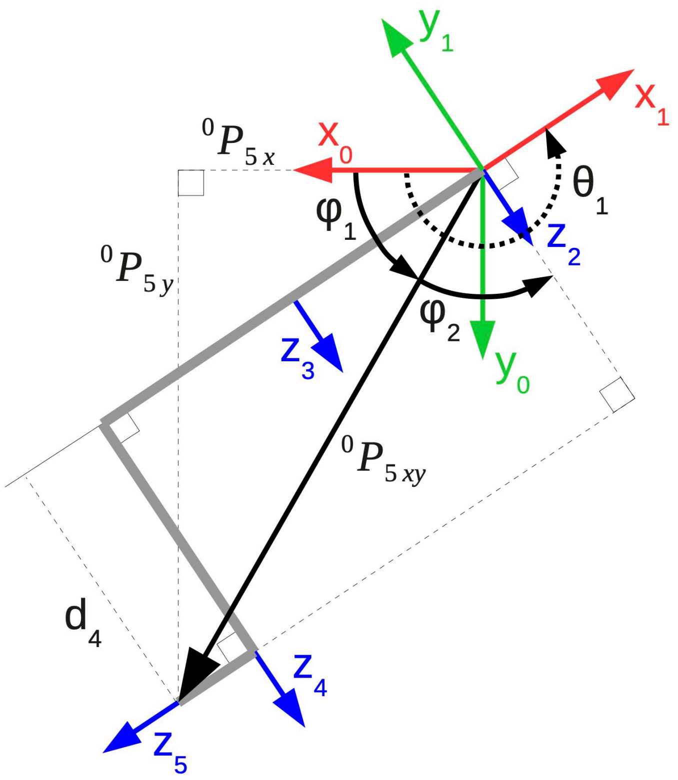
Figure 2: *Finding the origin of frame 5.*

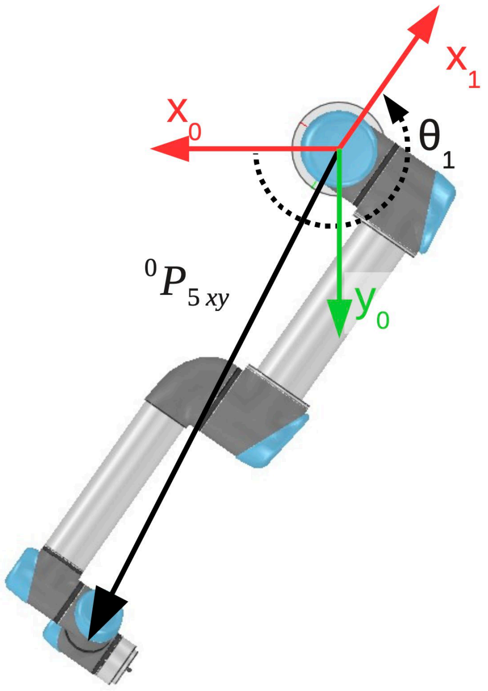
$$^0P_5 = {}^0P_6 - d_6 \cdot {}^0\hat{Z}_6 \Leftrightarrow$$

$$^0P_5 = {}^0_6T \begin{bmatrix} 0 \\ 0 \\ -d_6 \\ 1 \end{bmatrix}$$

We will use this to find θ_1.

Example UR IK: Finding θ_1 (II)





We break down θ_1 into two angles:

$$\theta_1 = \phi_1 + \left(\phi_2 + \frac{\pi}{2}\right)$$

We determine ϕ_1 by looking at the triangle formed by ${}^0P_{5x}$, ${}^0P_{5y}$ and ${}^0P_{5xy}$:

$$\phi_1 = \operatorname{atan2}({}^0P_{5y}, {}^0P_{5x})$$

We determine ϕ_2 by looking at the triangle formed by d_4, ${}^0P_{5x}$ and ${}^0P_{5xy}$:

$$\cos(\phi_2) = \frac{d_4}{|{}^0P_{5xy}|} \Rightarrow \phi_2 = \pm\operatorname{acos}\left(\frac{d_4}{|{}^0P_{5xy}|}\right) \Leftrightarrow$$

$$\phi_2 = \pm\operatorname{acos}\left(\frac{d_4}{\sqrt{{}^0P_{5x}{}^2 + {}^0P_{5y}{}^2}}\right)$$

$$\theta_1 = \phi_1 + \phi_2 + \frac{\pi}{2} \Leftrightarrow$$

Shoulder left/right

$$\theta_1 = \operatorname{atan2}\left({}^0P_{5y}, {}^0P_{5x}\right) \pm \operatorname{acos}\left(\frac{d_4}{\sqrt{{}^0P_{5x}{}^2 + {}^0P_{5y}{}^2}}\right) + \frac{\pi}{2}$$

Example UR IK: Finding θ_5

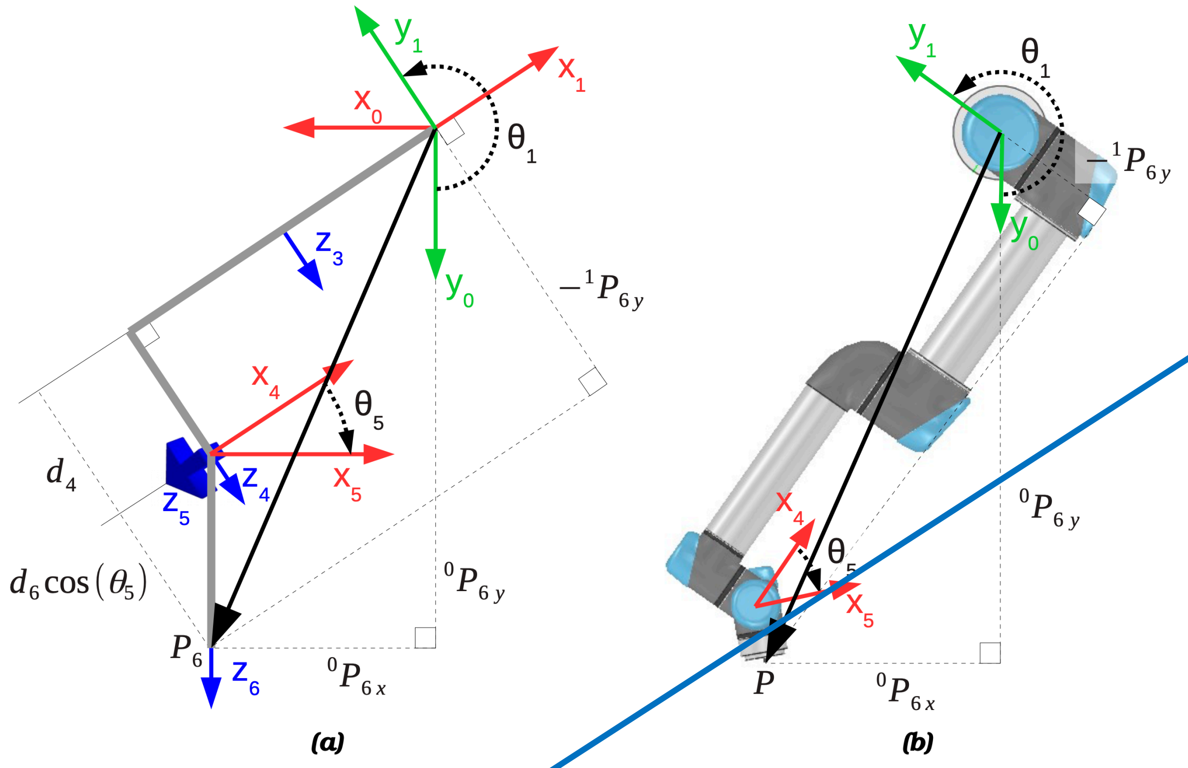


Figure 4: *Robot (including frame 6) seen from above.*

Notice that ${}^1P_{6y}$ only depends on θ_5.

Notice also that we can write:

$$\boxed{-{}^1P_{6y} = d_4 + d_6 \cos\theta_5}$$

But we can also look at is as a rotation around the z-axis of frame 0:

$$ {}^0P_6 = {}^0_1R \cdot {}^1P_6 \Leftrightarrow $$

$$ {}^1P_6 = {}^0_1R^\top \cdot {}^0P_6 \Leftrightarrow $$

$$\begin{bmatrix} {}^1P_{6x} \\ {}^1P_{6y} \\ {}^1P_{6z} \end{bmatrix} = \begin{bmatrix} \cos(\theta_1) & \sin(\theta_1) & 0 \\ -\sin(\theta_1) & \cos(\theta_1) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} {}^0P_{6x} \\ {}^0P_{6y} \\ {}^0P_{6z} \end{bmatrix} \Rightarrow$$

$$\boxed{{}^1P_{6y} = {}^0P_{6x} \cdot (-\sin\theta_1) + {}^0P_{6y} \cdot \cos\theta_1}$$

$$-d_4 - d_6 \cos\theta_5 = {}^0P_{6x}(-\sin\theta_1) + {}^0P_{6y}\cos\theta_1 \Leftrightarrow$$

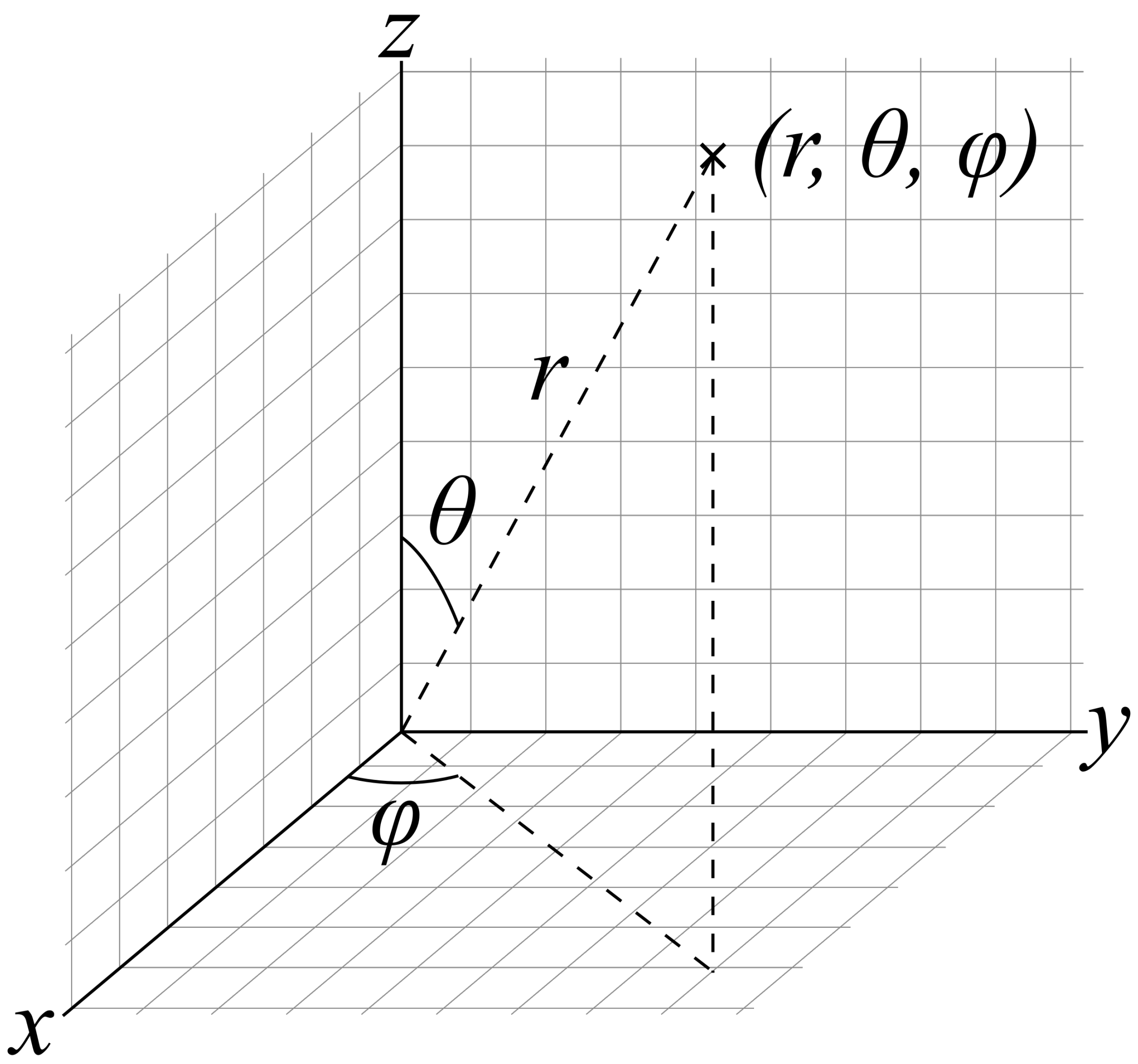
Wrist up/down

$$\cos\theta_5 = \frac{{}^0P_{6x}\sin\theta_1 - {}^0P_{6y}\cos\theta_1 - d_4}{d_6} \Leftrightarrow \quad \theta_5 = \pm\,\mathrm{acos}\left(\frac{{}^0P_{6x}\sin\theta_1 - {}^0P_{6y}\cos\theta_1 - d_4}{d_6}\right)$$

SDU ROBOTICS

Based on "Kinematics of a UR5" by Rasmus Skovgaard Andersen, Aalborg University

Refresher: Spherical Coordinates

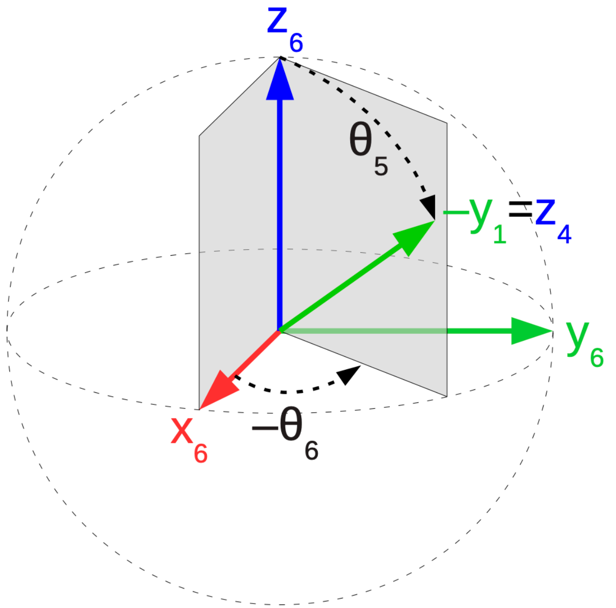


Conversion to Cartesian coordinates:

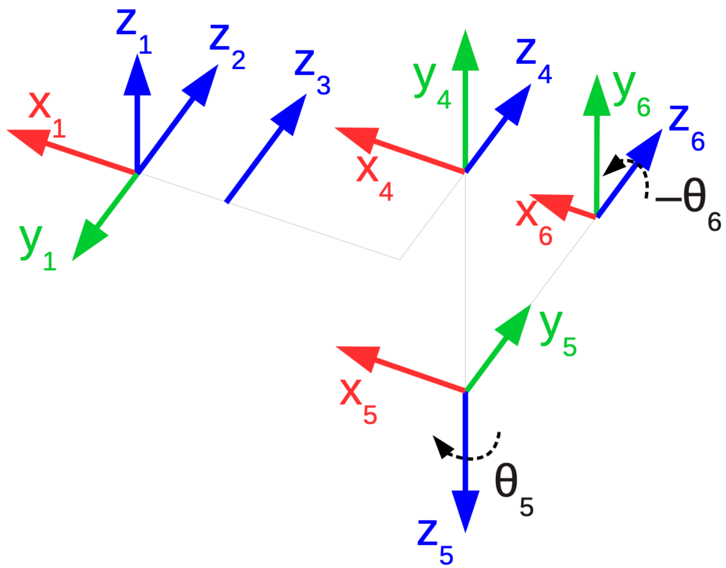
$$x = r\cos\varphi\,\sin\theta,$$
$$y = r\sin\varphi\,\sin\theta,$$
$$z = r\cos\theta.$$

SDU ROBOTICS

Example UR IK: Finding θ_6 (I)



***(a)** Polar angles for* $-{}^6\hat{Y}_1$



***(b)** Reference view of the relevant frames*

Notice that ${}^6\hat{Y}_1$ is always parallel to ${}^6\hat{Z}_{2,3,4}$

This implies it only depends on θ_5 and θ_6.

We can visualize ${}^6\hat{Y}_1$ in spherical coordinates and then re-write this in Cartesian coordinates:

$$
{}^6\hat{Y}_1 = \begin{bmatrix} -\sin\theta_5\cos\theta_6 \\ \sin\theta_5\sin\theta_6 \\ -\cos\theta_5 \end{bmatrix}
$$

We can express ${}^6\hat{Y}_1$ in terms of a rotation of frame 0 around the z-axis (like we did for θ_5):

$$
{}^6\hat{Y}_1 = {}^6\hat{X}_0 \cdot (-\sin\theta_1) + {}^6\hat{Y}_0 \cdot \cos\theta_1 \Leftrightarrow
$$

$$
{}^6\hat{Y}_1 = \begin{bmatrix} -{}^6\hat{X}_{0x} \cdot \sin\theta_1 + {}^6\hat{Y}_{0x} \cdot \cos\theta_1 \\ -{}^6\hat{X}_{0y} \cdot \sin\theta_1 + {}^6\hat{Y}_{0y} \cdot \cos\theta_1 \\ -{}^6\hat{X}_{0z} \cdot \sin\theta_1 + {}^6\hat{Y}_{0z} \cdot \cos\theta_1 \end{bmatrix}
$$

Example UR IK: Finding θ_6 (II)

Combining the first two entries of the two expressions from the previous slide:

$$\left.\begin{aligned} -\sin\theta_5\cos\theta_6 &= -{}^6\hat{X}_{0x}\cdot\sin\theta_1 + {}^6\hat{Y}_{0x}\cdot\cos\theta_1 \\ \sin\theta_5\sin\theta_6 &= -{}^6\hat{X}_{0y}\cdot\sin\theta_1 + {}^6\hat{Y}_{0y}\cdot\cos\theta_1 \end{aligned}\right\} \Leftrightarrow \left\{\begin{aligned} \cos\theta_6 &= \frac{{}^6\hat{X}_{0x}\cdot\sin\theta_1 - {}^6\hat{Y}_{0x}\cdot\cos\theta_1}{\sin\theta_5} \\ \sin\theta_6 &= \frac{-{}^6\hat{X}_{0y}\cdot\sin\theta_1 + {}^6\hat{Y}_{0y}\cdot\cos\theta_1}{\sin\theta_5} \end{aligned}\right\} \Rightarrow$$

$$\boxed{\theta_6 = \operatorname{atan2}\left(\frac{-{}^6\hat{X}_{0y}\cdot\sin\theta_1 + {}^6\hat{Y}_{0y}\cdot\cos\theta_1}{\sin\theta_5}, \frac{{}^6\hat{X}_{0x}\cdot\sin\theta_1 - {}^6\hat{Y}_{0x}\cdot\cos\theta_1}{\sin\theta_5}\right)}$$

If the denominator is 0 or both numerators are 0, then $\sin\theta_5 = 0$, and the solution is undefined.

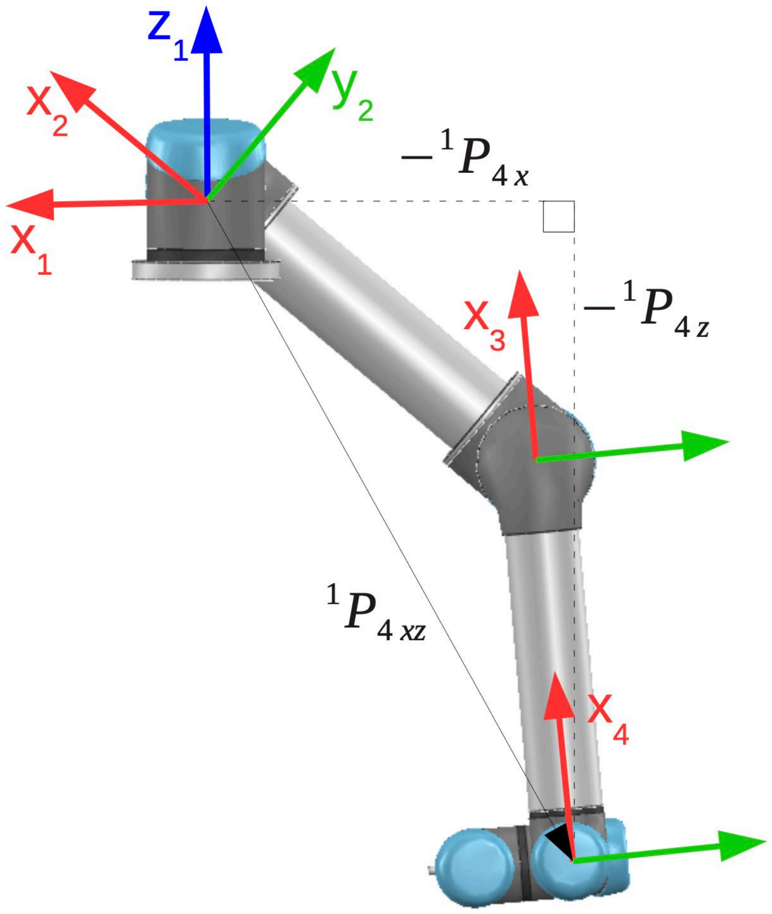
Question: What does it mean physically on the robot?

Hint: Look at the figure on the previous slide.

Answer: Axes 2, 3, 4 and 6 are aligned, making rotation around 6 redundant.


WOOHOO!
YOU HAVE REACHED A CHECKPOINT
imgflip.com

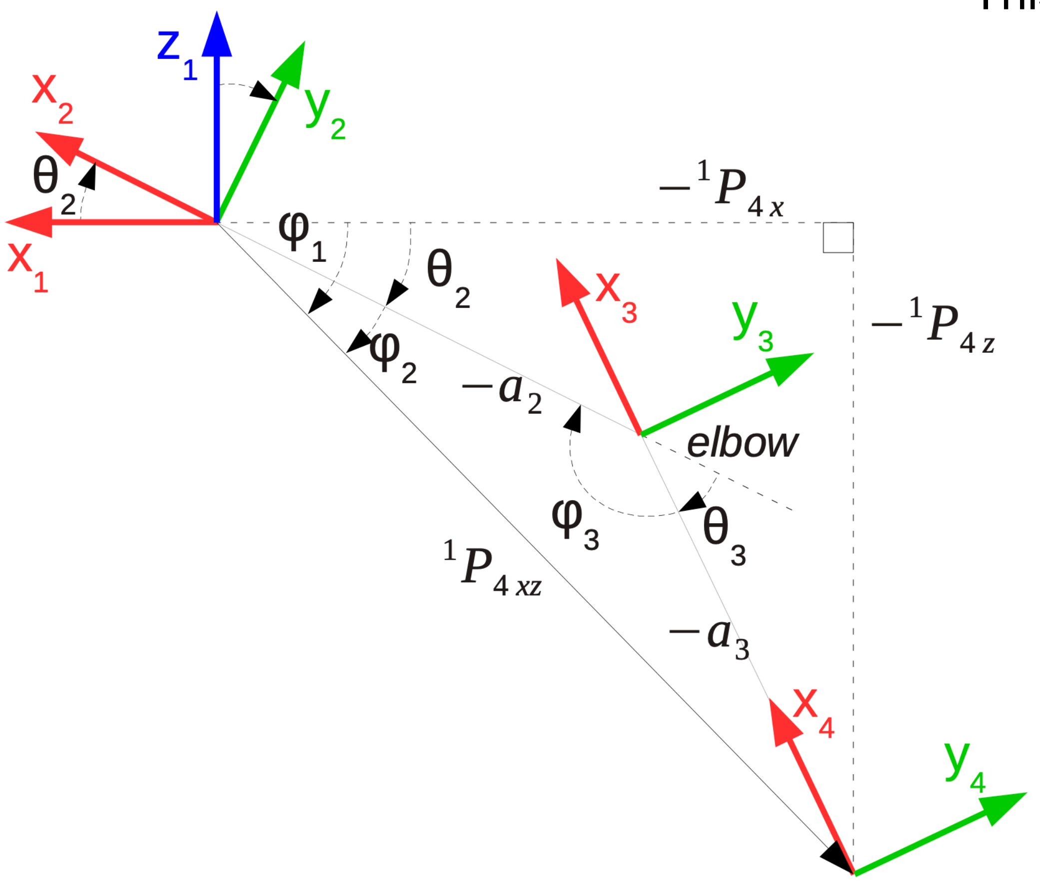
Example UR IK: Finding θ_2, θ_3 and θ_4 (I)



Notice that the three joints that are left consitute a 3R planar manipulator.

…and the position of a 3R planar manipulator can be solved for like a 2R, since the last joint only affects the orientation.

Example UR IK: Finding θ_3 and θ_2



This follows the **same logic as the 2R planar manipulator**:

$$\cos\theta_3 = -\frac{{a_2}^2 + {a_3}^2 - |{}^1P_{4xz}|^2}{2a_2a_3} \Leftrightarrow$$

$$\boxed{\theta_3 = \pm\text{acos}\left(\frac{|{}^1P_{4xz}|^2 - {a_2}^2 - {a_3}^2}{2a_2a_3}\right)}$$

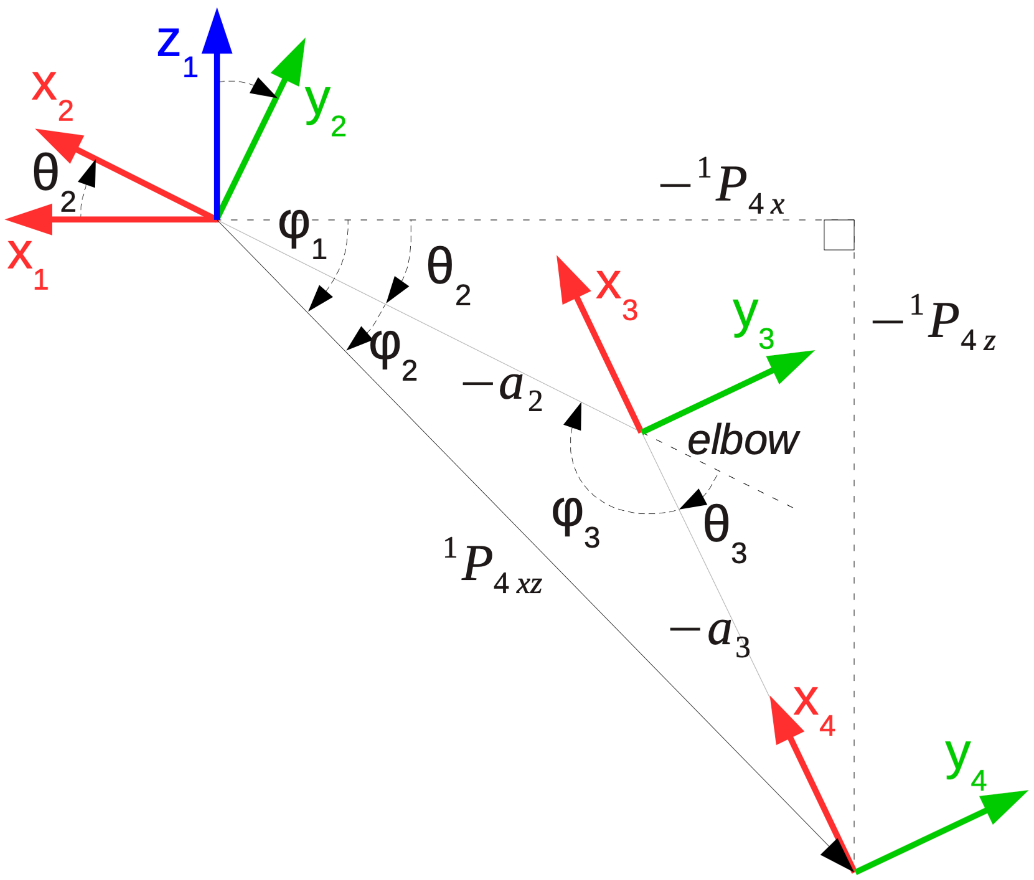
$$\phi_1 = \text{atan2}(-{}^1P_{4z}, -{}^1P_{4x})$$

$$\frac{\sin\phi_2}{-a_3} = \frac{\sin\phi_3}{|{}^1P_{4xz}|} \Leftrightarrow \phi_2 = \text{asin}\left(\frac{-a_3\sin\phi_3}{|{}^1P_{4xz}|}\right)$$

$$\boxed{\theta_2 = \phi_1 - \phi_2 = \text{atan2}(-{}^1P_{4z}, -{}^1P_{4x}) - \text{asin}\left(\frac{-a_3\sin\theta_3}{|{}^1P_{4xz}|}\right)}$$

Based on "Kinematics of a UR5" by Rasmus Skovgaard Andersen, Aalborg University

Example UR IK: Finding θ_4



From the definition of θ as a DH parameter:

"$\boldsymbol{\theta_i}$ – The angle from $\widehat{\boldsymbol{X}}_{i-1}$ to $\widehat{\boldsymbol{X}}_i$ measured about $\widehat{\boldsymbol{Z}}_i$"

From the definition of a rotation matrix:

$$ {}^A_B\boldsymbol{R} = [{}^A\widehat{\boldsymbol{X}}_B \quad {}^A\widehat{\boldsymbol{Y}}_B \quad {}^A\widehat{\boldsymbol{Z}}_B] $$

We then can then just look at the first column of the rotation of ${}^3_4\boldsymbol{T}$ and write:

$$ \boxed{\theta_4 = \text{atan2}({}^3\hat{X}_{4y}, {}^3\hat{X}_{4x})} $$

Based on "Kinematics of a UR5" by Rasmus Skovgaard Andersen, Aalborg University

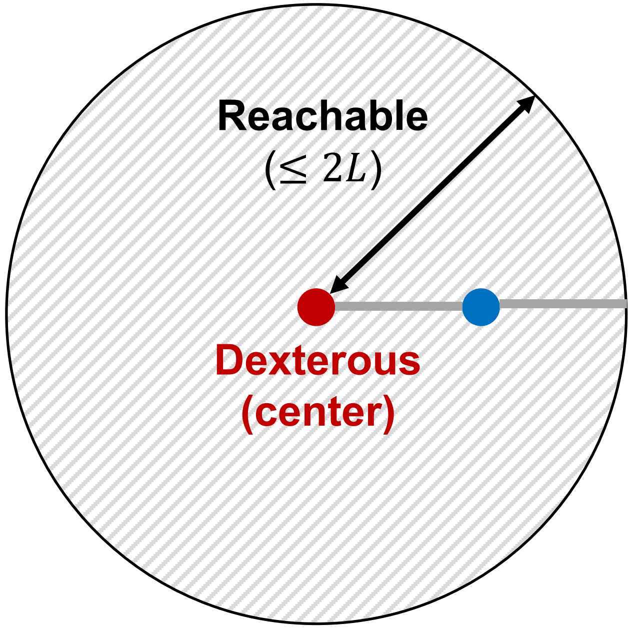
Part III: Practical Considerations

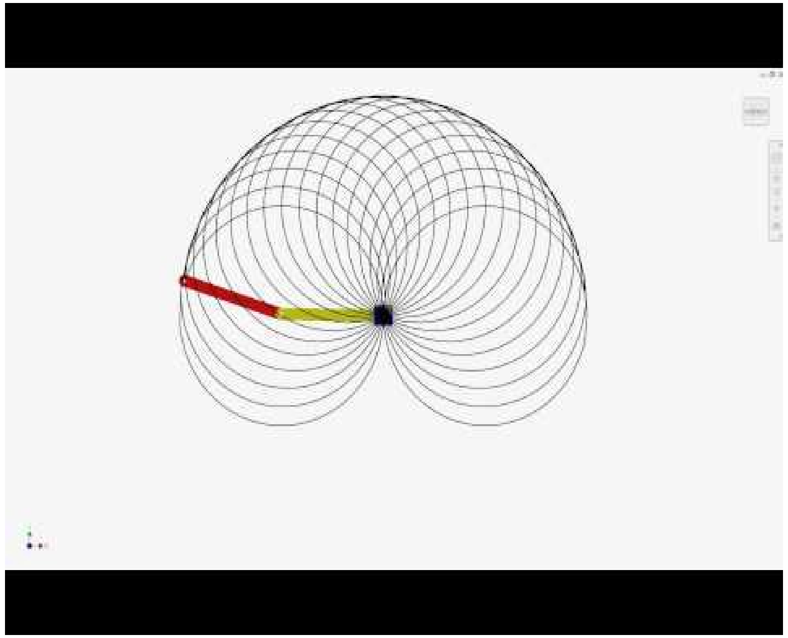
Workspace in relation to Inverse Kinematics

The **workspace** is the **volume of space that the end-effector can reach**. We can distinguish between:

- **Reachable workspace**: can be reached with **at least one orientation**.
- **Dexterous workspace**: can be reached with **any orientation**.

Example: For a **2R planar manipulator with equal-length links**, $L_1 = L_2 = L$:





SDU ROBOTICS

DOF and Number of Solutions

The number of **DOF** of the robot will greatly **affect the number of IK solutions**.

For a 3D Cartesian space with 3-DOF position + 3-DOF orientation and an n-DOF robot:

- $n < 6$: Will often run into the problem of not having a solution

- $n = 6$: Will (in theory) be the minimum to cover the entire space:
  - Some points will have a single IK solution
  - Some points will have multiple solutions
  - (In practice, some points will present singularities) → **We will see this in upcoming lectures**

- $n > 7$: Will be redundant (always multiple solutions).

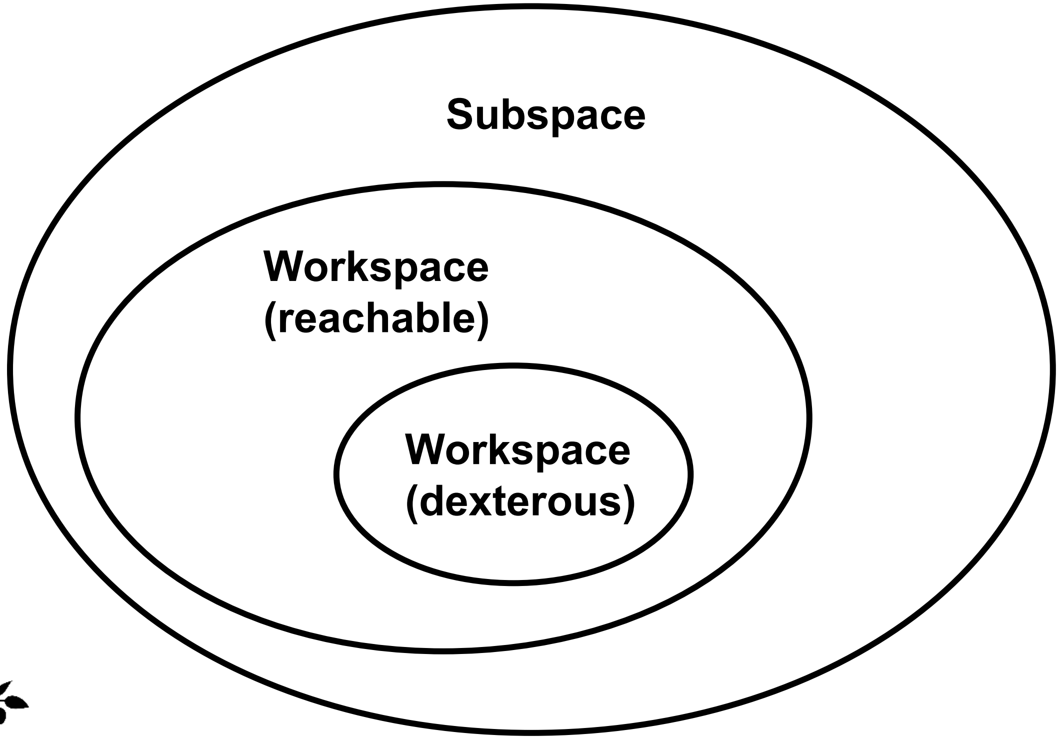
SDU ROBOTICS

Notion of Subspace for $n < 6$ DOF

A 6-DOF robot operates in general in 3D space, but can only move to points in its reachable workspace.

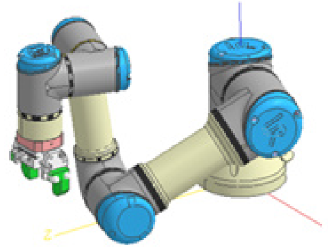
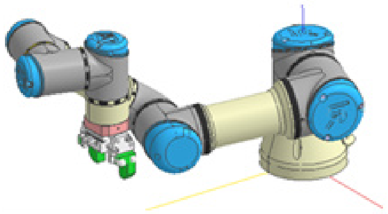
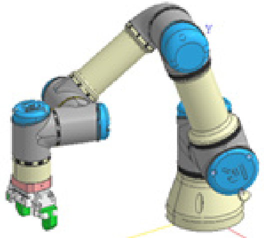
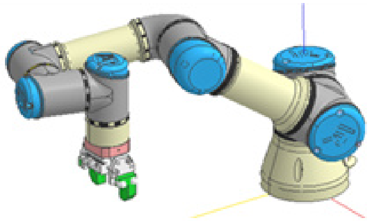
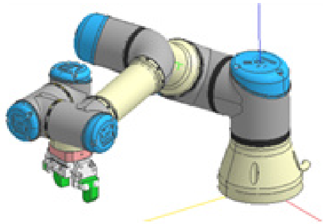
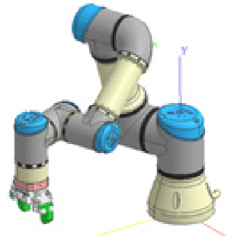
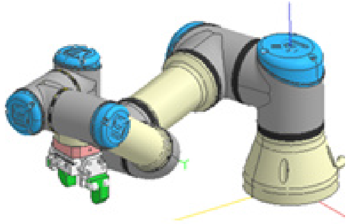
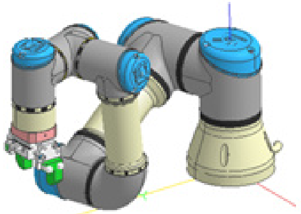
For $n < 6$ DOF, we can define a portion of space that it operates in, called a **subspace**.

→ Analyzing the subspace can help us determine whether the robot fits our needs.



Question: What is the subspace of A 2R planar robot?

Example – Multiple IK Solutions: UR (6-DOF)

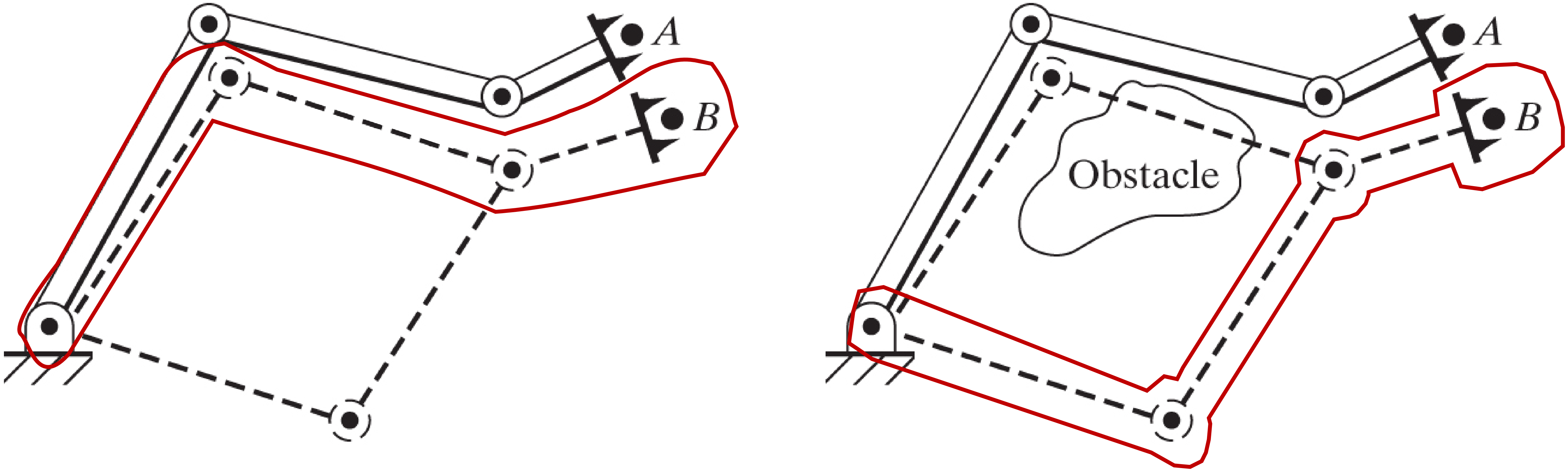
Pattern	Shoulder	Elbow	Wrist	Figure1 (Tool Down)
1	Left Side	Down	Tool Down: Outer	
2	Left Side	Down	Tool Down: Inner	
3	Left Side	UP	Tool Down: Outer	
4	Left Side	UP	Tool Down: Inner	
5	Right Side	UP	Tool Down: Inner	
6	Right Side	UP	Tool Down: Outer	
7	Right Side	Down	Tool Down: Inner	
8	Right Side	Down	Tool Down: Outer	

Multiple IK Solutions: 7-DOF

SDU ROBOTICS

Choice of IK Solutions

Different tasks will dictate a **different choice of IK solution**:



Repeatability & Accuracy (in relation to FK/IK)

Question: Considering what you know about FK and IK, **which mode of control** is (potentially) preferable if **reaching a point precisely** is required? **Why**? *(Discuss with your neighbors – 5 minutes).*

Joint-space control:

- Only **depends on** the **joint controllers**/the ability of the **joint encoders** to measure joint position.

→ **<u>Repeatability</u>**: measures how precisely a manipulator can **return to a known set of joint angles**.

Cartesian-space control:

- Depends on the **same as joint-space control** +
- Depends on **how precisely the DH parameters are known**. This is **subject to manufacturing tolerances**.

→ **<u>Accuracy</u>**: measures of how precisly a manipulator can **reach a point computed through IK**.

- **Calibration** of the parameters can be used to **improve accuracy**.

Repeatability $\geq$ Accuracy

Recap: What have we discussed today

- A robot can be controlled in **joint space** or **Cartesian space**. The choice of either will imply solving a problem of:
  - **Forward Kinematics**: given joint angles, determine the position of the end-effector.
  - **Inverse Kinematics**: given a position of the end-effector, determine (all sets of) the corresponding joint angles.
- **Inverse Kinematics** is a complex non-linear problem, which can be solved:
  - **Analytically** (**closed-form**): Only possible for some robots, hard to derive.
  - **Numerically (iterative)**: A numerical approximation to the solution.
- The **workspace** of a robot is closely related to the existence and number of IK solutions:
  - **Reachable**: the robot can reach the point with at least one orientation
  - **Dexterous**: the robot can reach the point with any orientation.
- The measures of **reapeatability** and **accuracy** are closely related to the FK and IK problems.

Take home message:

Inverse Kinematics is a much more complex problem than Forward Kinematics, but is **required to control a robot in task space**.

The **structure** and **number of DOF** of the robot will have a huge impact on the **number** and **type of IK solutions** available.

Thank you for today.

Iñigo Iturrate

Ø27-604-3

inju@mmmi.sdu.dk