Software Develpoment for Industrial Robots

3. Homogenous coordinates

- Define some new mapping:

$$V \longrightarrow V \times R$$

$$\begin{pmatrix} v_1 \\ \cdots \\ v_n \end{pmatrix} \longmapsto \begin{pmatrix} v_1 \\ \cdots \\ v_n \\ 1 \end{pmatrix}$$

- The new (n+1)-dimensional vector space is called homogenous coordinates

Rotations

- Benefit: Rotations <u>and</u> translations may be described as matrix operations

- Example: Rotation in 2D

$$A_\alpha := \begin{pmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

- Example: Translation in 2D

$$A_{\begin{pmatrix} x \\ y \end{pmatrix}} := \begin{pmatrix} 1 & 0 & x \\ 0 & 1 & y \\ 0 & 0 & 1 \end{pmatrix}$$

Rotation&Translation

- In general:

$$T(\alpha,\vec{p}):V\to V$$

$$v\mapsto\begin{pmatrix}R_\alpha & \vec{p}\\ 0 & 1\end{pmatrix}v=\begin{pmatrix}\cos\alpha & -\sin\alpha & p_1\\ \sin\alpha & \cos\alpha & p_2\\ 0 & 0 & 1\end{pmatrix}v$$

- Inverse of this mapping:

$$T^{-1}(\alpha,\vec{p}):V\to V$$

$$v\mapsto\begin{pmatrix}R_\alpha^{-1} & -R_\alpha^{-1}\vec{p}\\ 0 & 1\end{pmatrix}v=\begin{pmatrix}\cos\alpha & \sin\alpha & -p_1\cos\alpha-p_2\sin\alpha\\ -\sin\alpha & \cos\alpha & p_1\sin\alpha-p_2\cos\alpha\\ 0 & 0 & 1\end{pmatrix}v$$

Definition: Co-Sys Transformation

Let CS‘ be some coordinate system defined by some translation-rotation $T\left(\alpha,\begin{pmatrix}x\\y\end{pmatrix}\right)$ in CS. Then some point P with coordinate vector v‘ in CS‘, can also be described by the coordinate vector

$$v = T\left(\alpha,\begin{pmatrix}x\\y\end{pmatrix}\right)v'$$

in CS.

Transformation of coordinate systems

Corollar:
Consequently, a point P with coordinate vector v‵ in CS‵, can also be described by the coordinate vector

$$v' = T^{-1}\left(\alpha, \begin{pmatrix} x \\ y \end{pmatrix}\right) v$$

in CS‵.

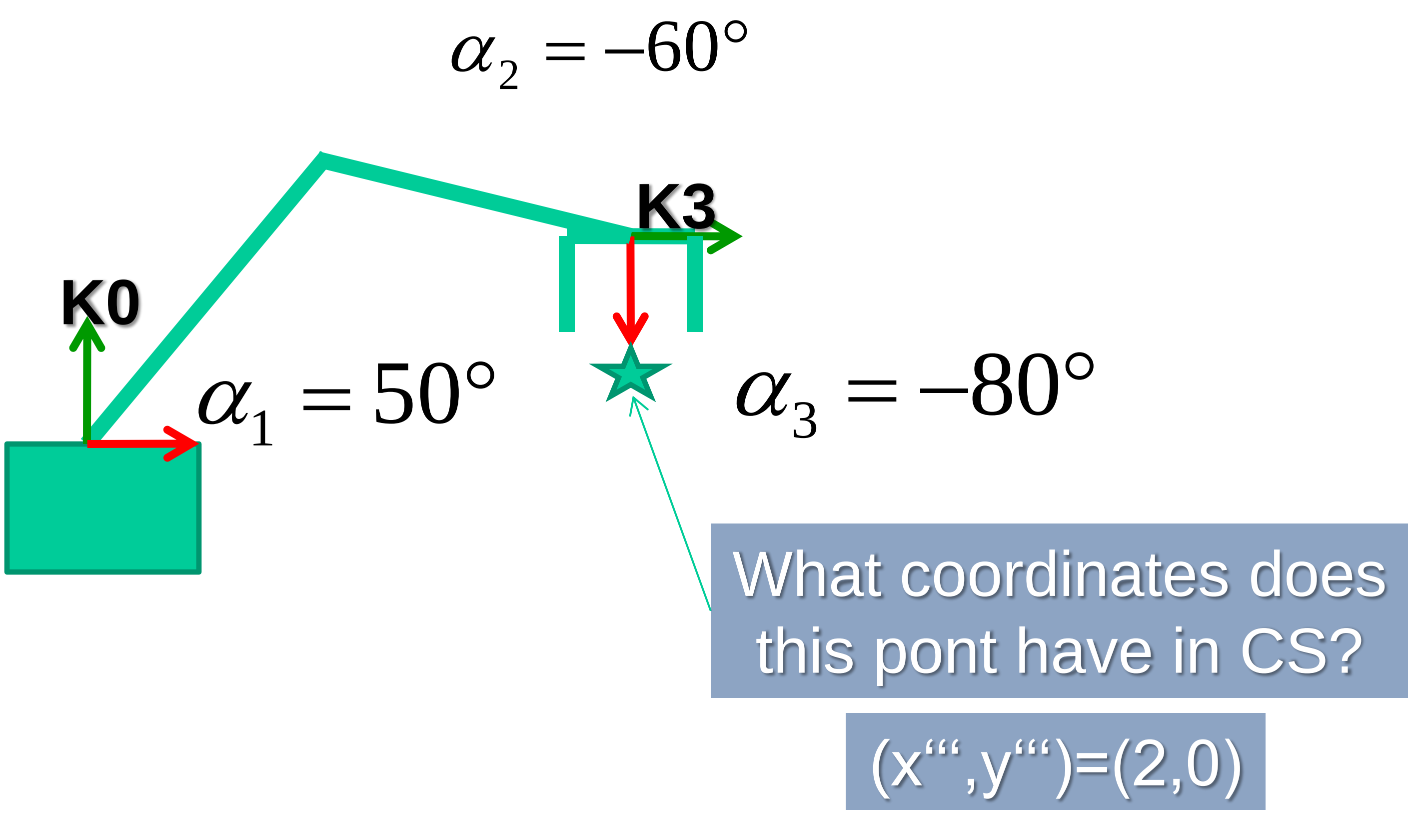
Consequence:
Switiching coordinate systems is very easy!

Enough theory!

➔ Let’s use this stuff!

4. (2D) Direct kinematics

- Example: 2-arm robot (in some 2D disc world)



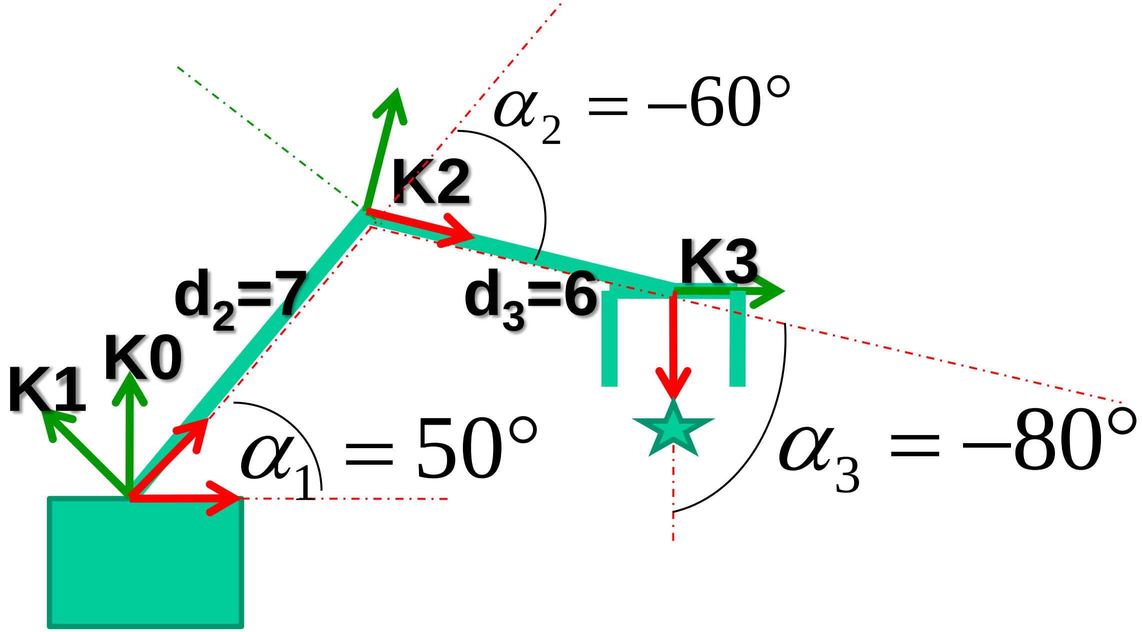
Direct kinematics

- Definition:
  The direct kinematics of a robot is an operation which takes joint angles as input and calculates the position of the robot‘s hand.

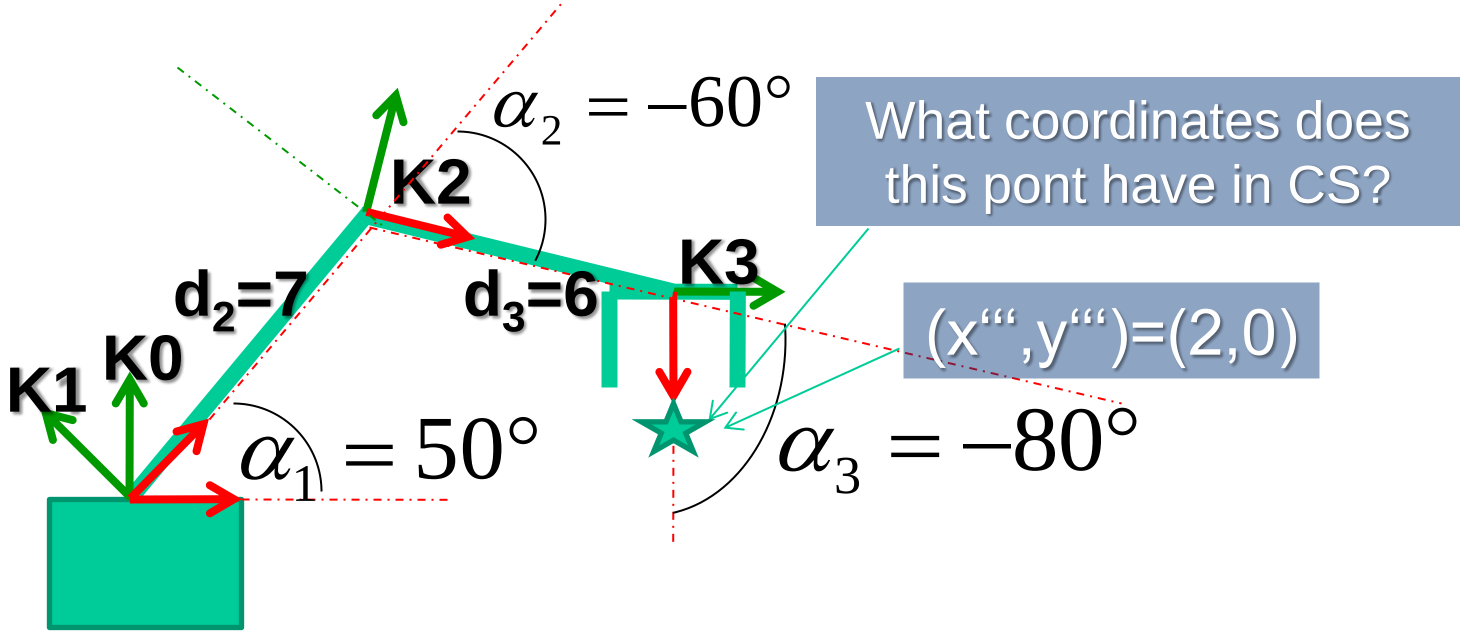
- Core idea:
  Direct kinematics is simply a set of well-defined coordinate system transformations

Chains of coordinate systems

- Convention: Put one coordinate system into each joint, such that the X axis is pointing towards the next joint.



- Now it‘s simple to express any point in K0!



$$v = T\left(50°, \begin{pmatrix}0\\0\end{pmatrix}\right) T\left(-60°, \begin{pmatrix}7\\0\end{pmatrix}\right) T\left(-80°, \begin{pmatrix}6\\0\end{pmatrix}\right) \begin{pmatrix}2\\0\end{pmatrix}$$

Summary

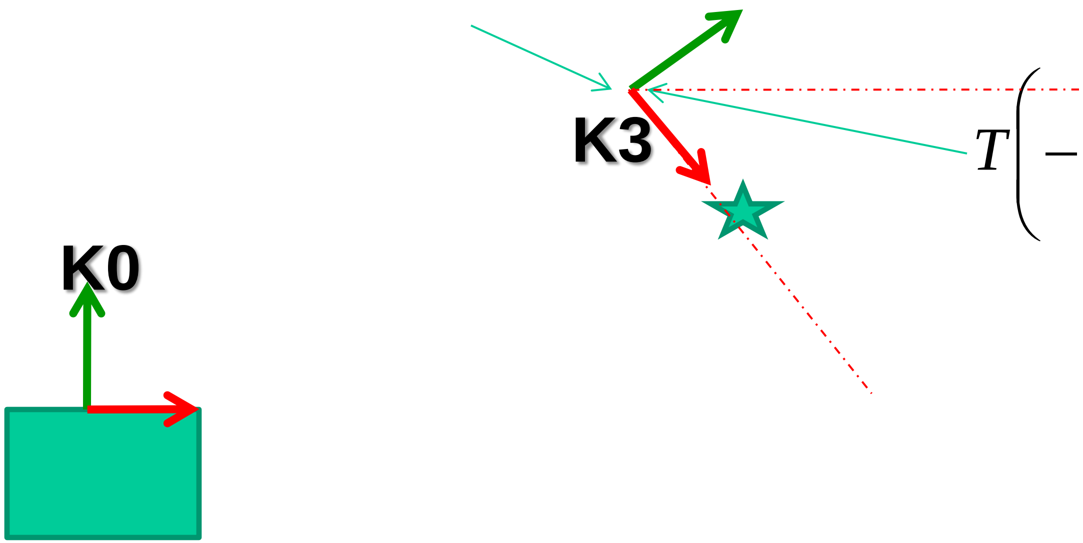
- The direct kinematics of a robot can be understood as a simple chain of coordinate system transformations.

- This works for
  - 2D and 3D analogously
  - Rotational as well as translational joints

- Question:
  Which types of robot kinematics may not be solved with this approach?

Inverse kinematics

Which angles place the robot's hand to the desired point?



$$T\left(-55°,\begin{pmatrix}9\\5\end{pmatrix}\right)$$

5. (2D) Inverse kinematics

- Definition:
  The inverse kinematics of a robot is an operation which takes the position of the robot‵s hand as input and calculates joint angles.

- Possible startegies for solution:
  - Analytical: take the direct kinematics formula and isolate all angles
  - Numerical: use some apporixmation algorithm for computing the solution
  - **Geometric:** use geometric understanding of the robot‵s layout

Computer Systems in Engineering

Geometric solution

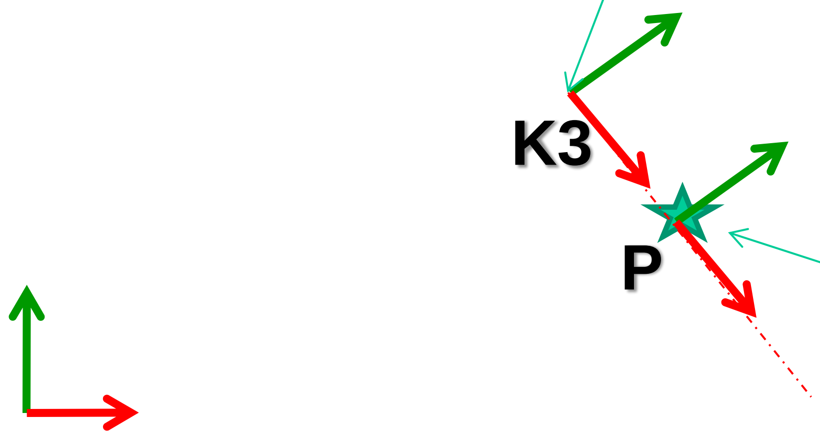
Core idea: Use the first two joints for bringing the robot`s wrist to a desired position, then use the third axis to find the correct orientation!

$$T\left(-55°,\begin{pmatrix}9\\5\end{pmatrix}\right)$$

Inverse kinematics

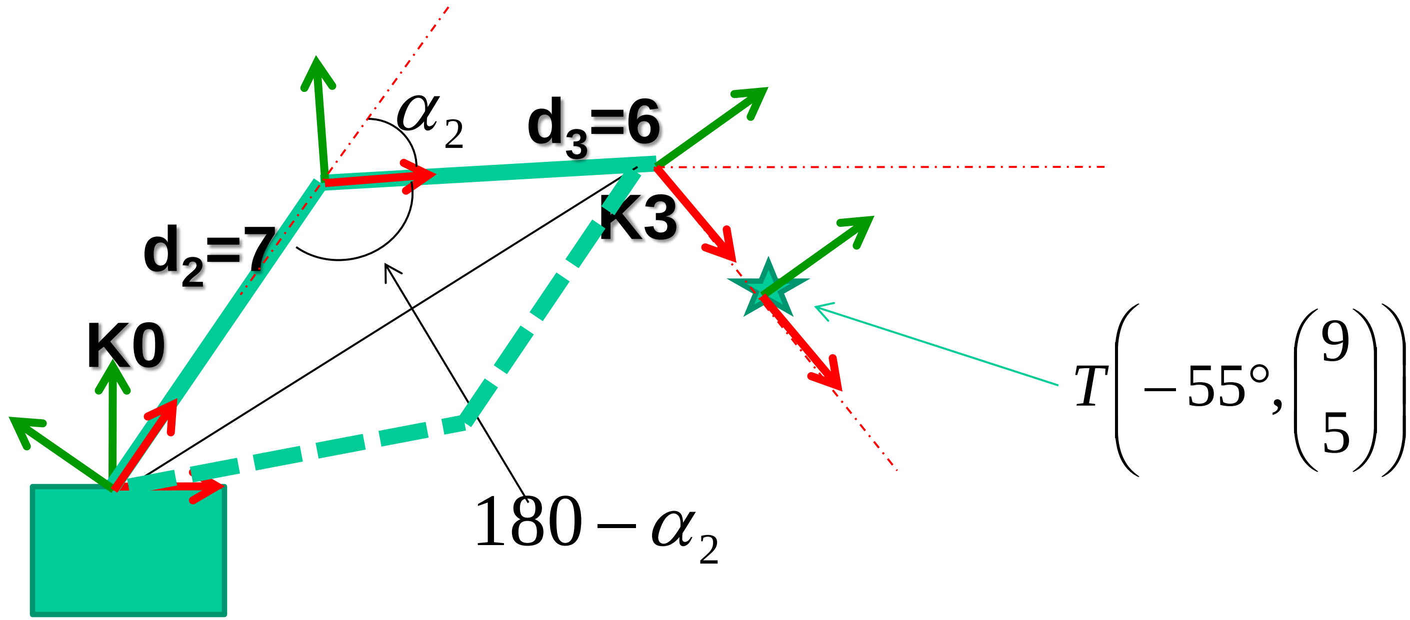
0. Compute wrist position

$$T\left(-55°, T\left(-55°, \begin{pmatrix}9\\5\end{pmatrix}\right)\begin{pmatrix}-2\\0\end{pmatrix}\right)$$



$$T\left(-55°, \begin{pmatrix}9\\5\end{pmatrix}\right)$$

1. Compute angle α_2

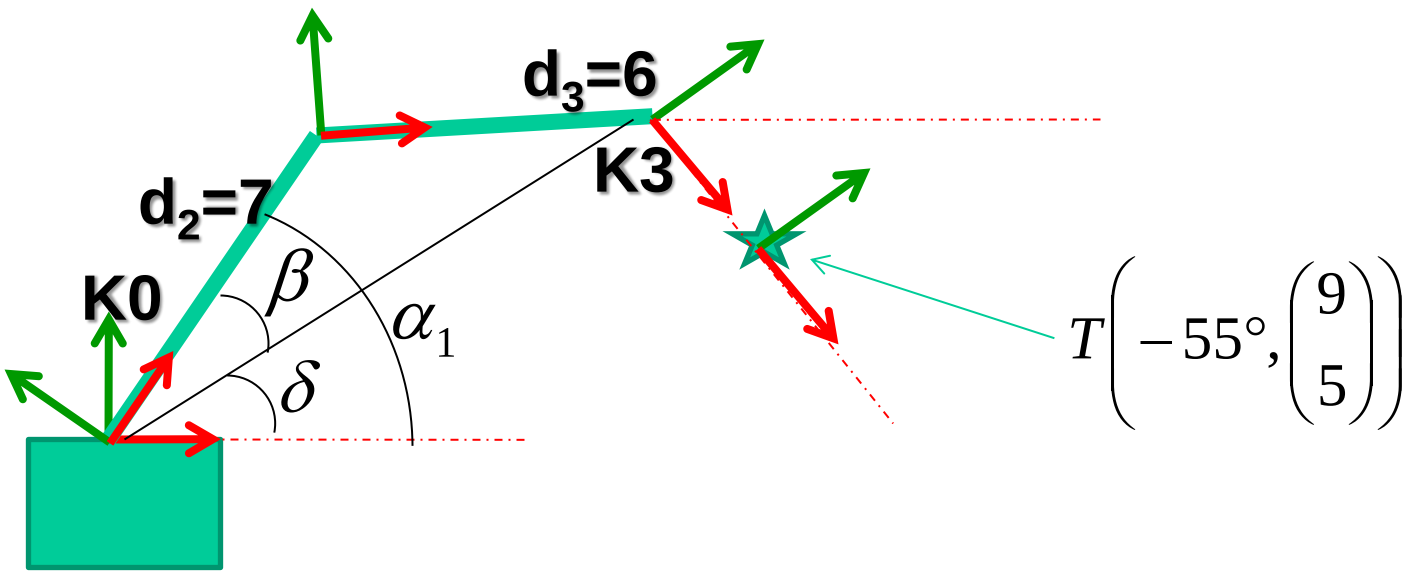


$$\left.\begin{aligned}\cos\rho &= -\cos(\pi-\rho)\\ c^2 &= a^2+b^2-2ab\cos\gamma\end{aligned}\right\}$$

$$\frac{21}{84}=\cos 180-\alpha_2 \Rightarrow \alpha_2=\pm 73°$$

Inverse kinematics

2. Compute angle α_1

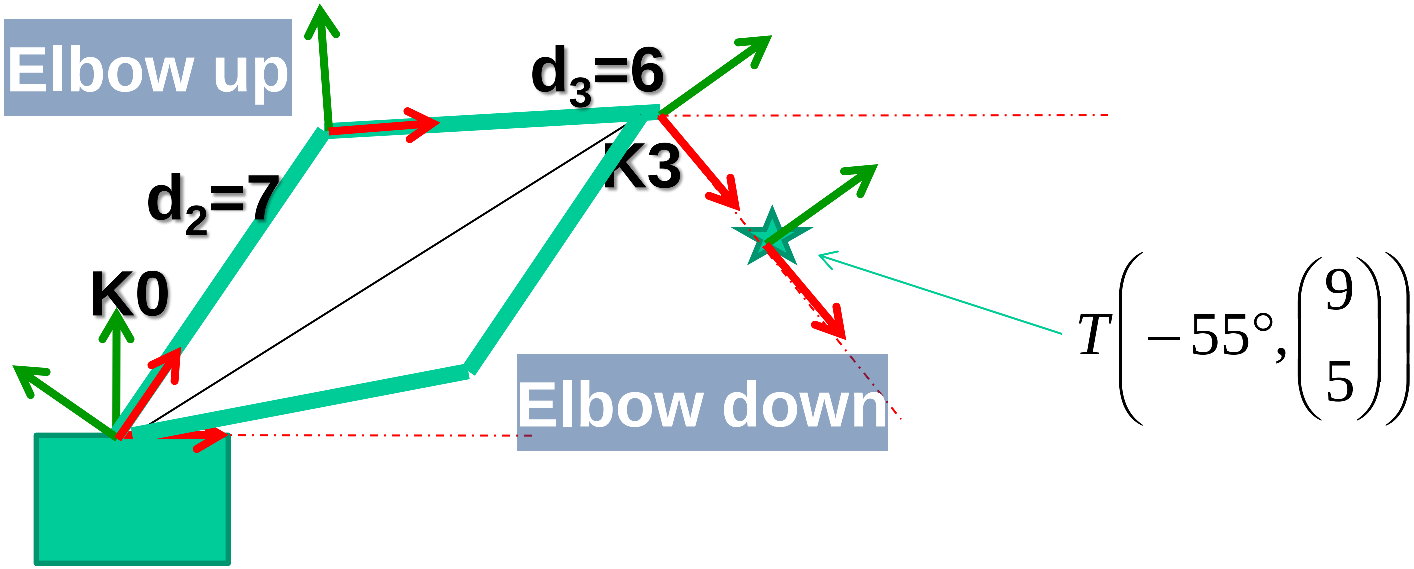


$$\delta = \arctan 2(9;5)$$

$$d_3{}^2 = d_2{}^2 + (x^2 + y^2) - 2d_2\sqrt{x^2+y^2}\cos\beta$$

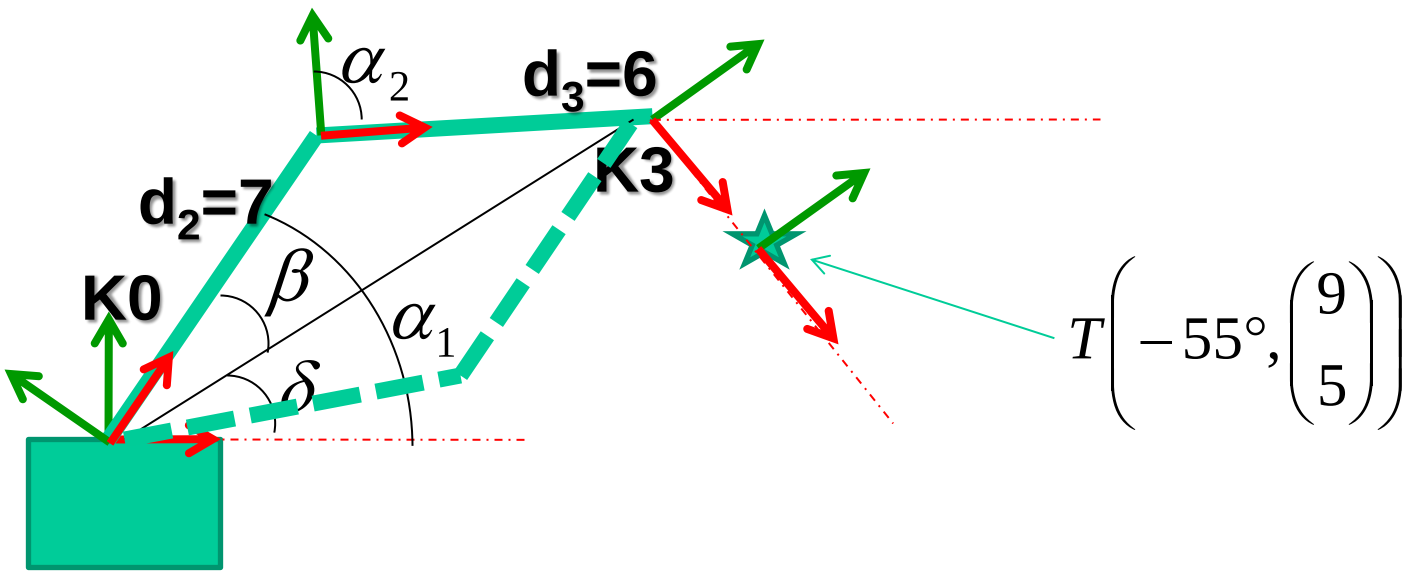
$$\alpha_1 = \delta + \beta = 29° \pm 34°$$

2 possible solutions



Orientation of the hand

3. Compute axis α_3



$$\alpha_3 = -55 - \alpha_2 - \alpha_1$$

$$\alpha_{3,elbow\_up} = -45° \quad \alpha_{3,elbow\_down} = -123°$$

Summary

- Orthonormal (right-handed) coordinate systems differ only in a translation and one rotation.

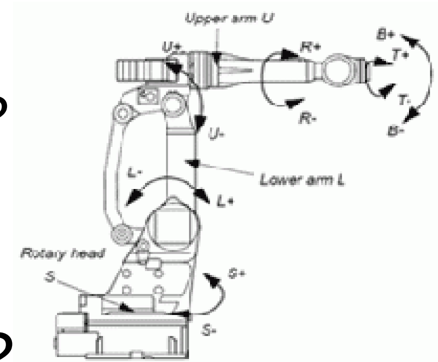
- For kinematics computation, it is very useful to place one coordinate system into each joint.

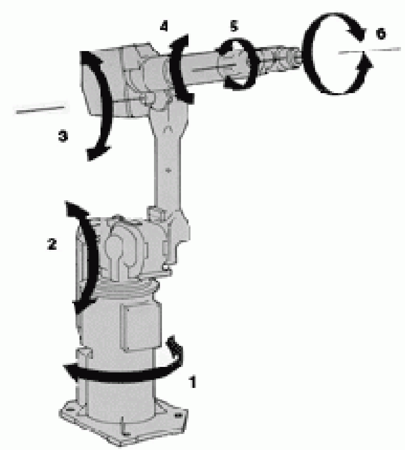
- A simple geometric solution to the inverse kinematics problem is to compute first position of the robot‘s wrist and then it‘s orientation.

4. Basic concept of industrial robots

How are industrial robots programmed?

- Programming languages?
- Atomic operations?
- Specific challenges?
- Process of programming?





Your ideas...

- Programming languages?
- Atomic operations?
- Specific challenges/programming layers?
- Programming process?

Some first ideas

- Programming languages?
  - Feedback, Events, Responses
  - Schleifen, Verzweigung, Rekursion, OO
  - Regel-basiert / deklarativ / Unterspezifikation
- Atomic operations?
  - Sollwerte für Gelenke setzen + Zeitinformation
  - Zielpunkt in Koordinaten angeben
  - LIN (kartesisch), SPLINE
  - Atomic commands sollen “thread safe”
- Specific challenges/programming layers?
  - Low level: HW-Gelenke, Top level: Geometrie
  - Aufteilung: Hersteller-spezifisch vs. anwendungs-spezifisch
- Programming process?
  - Testen ist schwieriger
  - Auswahl HW, Auswahl Progsprache in Abhänigkeit von Aufgabe
  - Modularisierung der SW / SW-Architektur
  - Mischen von Offline-Programmierung mit Teachen

Movements

- Often people only describe movements by pathes (e.g. move on a straight line). A path (ger. **Bahn**) is a subset of 3D (resp. 6D) space.

- However, in most applications not pathes but trajectories (ger. **Trajektorie**) are intended. A trajectory is a mapping of time to a path.

Atomic commands

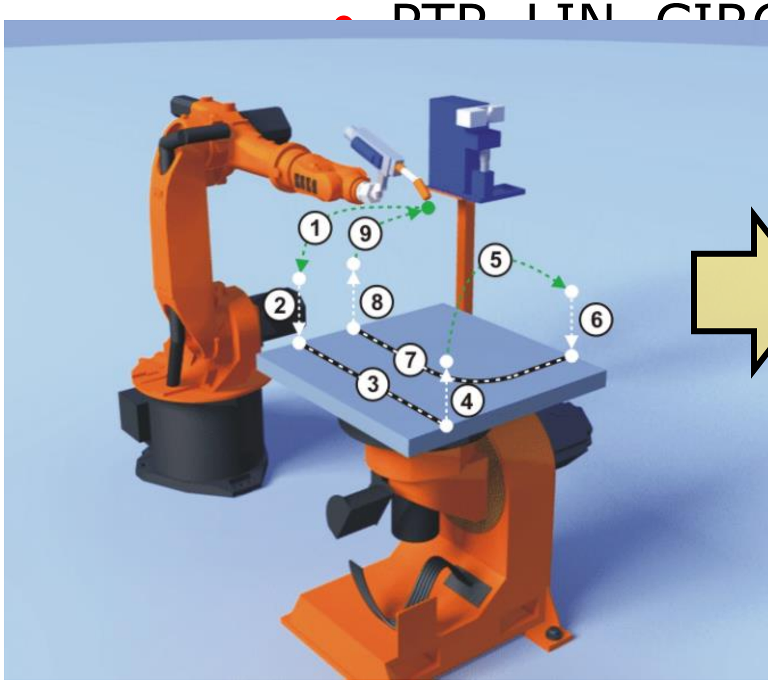
- Different pathes:
  - Linear movement
  - Point-to-point movement
  - Circular movement

  - Modern: (cubic) spline movement

- Different trajectories:
  - Velocity profiles
  - Acceleration profiles
  - Total movement time

Example: KUKA

- Very simple programming languages
  - No recursion
  - No object
  - Only global variables
  - PTP, LIN, CIRC




```
1  INI
2  DEF ...
3  PTP HOME VEL= 100 % DEFAULT
...
4  PTP P1 VEL= 100 % PDAT 1 Tool[1] Base[0]
5  PTP P2 VEL=100 % PDAT2 Tool[1] Base[0]
...
6  ...1()
7  PTP P7 Vel= 100 % PDAT
8  PTP P8 Vel= 100 % PDAT
9  ...2()
...
10 PTP P15 Vel= 100 % PDAT1
11 PTP P16 Vel= 60 % PDAT
12 PTP HOME VEL= 100 % DEFAULT
...
13 GLOBAL DEF ..._EHS(ConsiderCRErr:IN,Motion:IN,
   Service:IN;MsgType:IN,SeamState:OUT,ErrStrategy2Counthax:IN,
   TargetState:IN,CallFrom[]:OUT)
...
14 CASE #PartServiceRoutine01
...
15 CASE #PartServiceRoutine02
...
16 END
```

Different layers of control

- Electrical control:

  "How do I need to control voltage, such that the arm reaches/holds a position without tremor?"

- Realtime control:

  "How can I guarantee, that intended values are always available in time?"

- Robot control/path planing:

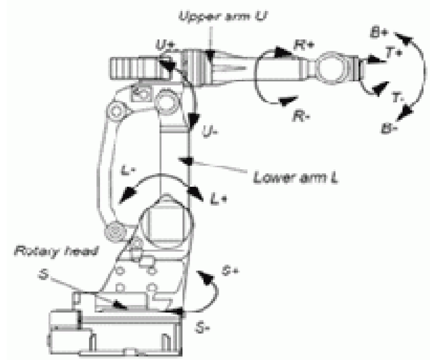
  "How do I calculate (sequences of) intended values for all axis of a robot?"

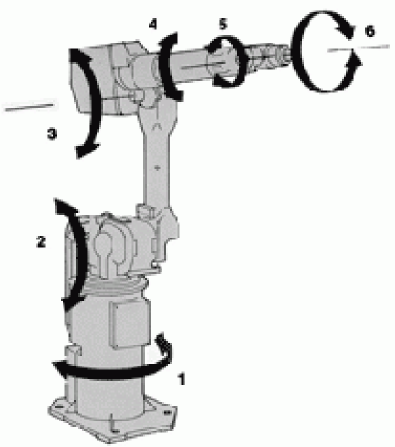
- High-level functionalities:

  "How can I specify and calculate a collision-free trajektory efficiently?"

5. Industrial 6 axis robots

“Standard-” industrial robots





Computer Systems in Engineering

“Standard-” industrial robots

- Design:
  - 6 rotational joints
  - Anthropomorphic hand (i.e. the last three axis intersect in one point)

- Our goal:
  Design and implementation of a control for such a robot.

Requirements

- Control should – in theory – be capable of control any standard robot

- Necessary commands:
  - Movement in axis space
  - Movement in Cartesian space: LIN, PTP
  - Velocity profiles
  - Synchronuous and asynchronuous movements

Next steps

- Identification of necessary subtasks
- Solving these subtask in separate groups
  - We will give support to each group during the next two weeks
  - Result: Presentation of necessary concepts, such that everybody else understands it (and can implement it)

- Important:
  - Early specification is important for
    - More easy system integration
    - Better understanding of you problems

What subtasks do we need to solve?

Subtasks

1. Positioning of the wrist
   Grotzke
2. Orientation of the wrist
   Stellmacher,
3. Standardized coordinate systems / layouts of robots
   Gonschorek, Bahl, Sulkowski
4. Computation of trajectories / velocity profiles
   Belov, Baier, Hagemann
5. Selection of solution /singularities
   Schubert,
6. Software architecture
7. Robot prgramming framework (openRAVE)
   Meuschke, Pauksch, Hettig

Kleine Zusatzkomplikation

- Eine Anforderung wurde vergessen!
- Wer macht die Vorwärtskinematik?
- Entscheiden Sie selbst, welche Gruppe am Sinnvollsten dieses Zusatzproblem mit bearbeitet!