Robot Modelling II

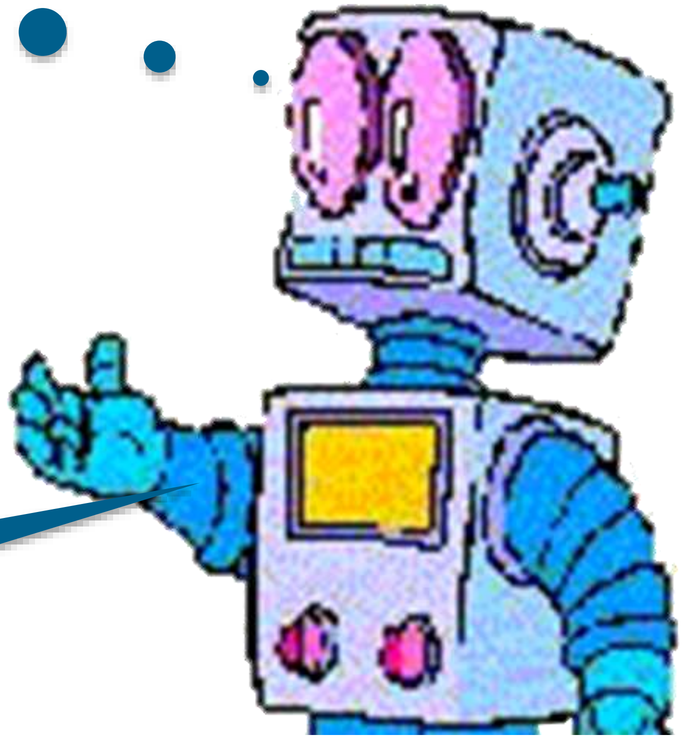
Prof. Dr. Karsten Berns
Robotics Research Lab
Department of Computer Science
University of Kaiserslautern, Germany

Content

- Degrees of freedom of a robotic system
- Geometric model
- Kinematic model
- Direct kinematic problem
- Inverse kinematic problem
- Dynamic model

Inverse Kinematic Problem

How should I move my hand there?



Find the joint angles

Inverse Kinematic Problem (IK)

- From D-H-parameters and the positon of the gripper one should calculate the joint angles $\rightarrow$ solve equation for $\vec{\theta}$

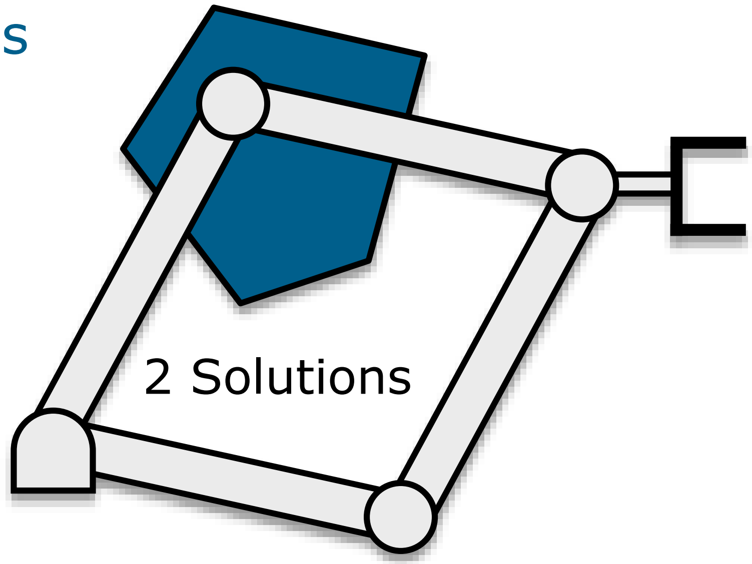
$$ {}^{\mathrm{BASIS}}_{\mathrm{TCP}}A = {}^{\mathrm{BASIS}}_{1}A(\theta_1) \cdot {}^{1}_{2}A(\theta_2) \cdots {}^{n-2}_{n-1}A(\theta_{n-1}) \cdot {}^{n-1}_{n}A(\theta_n) $$

- This yields 12 equations with n unknowns
- For Puma 260: 12 equations, 6 unknowns

IK-Problem

- Acceptable configurations: Not all mathematical solutions can be reached mechanically
  - Limitation of joint angles
  - Singular configuration
  - Endpoint does not belong to workspace

- Uniqueness: Multiple configurations (combinations of joint angles) result in the same position of the end effector
- How to choose a suitable solution?

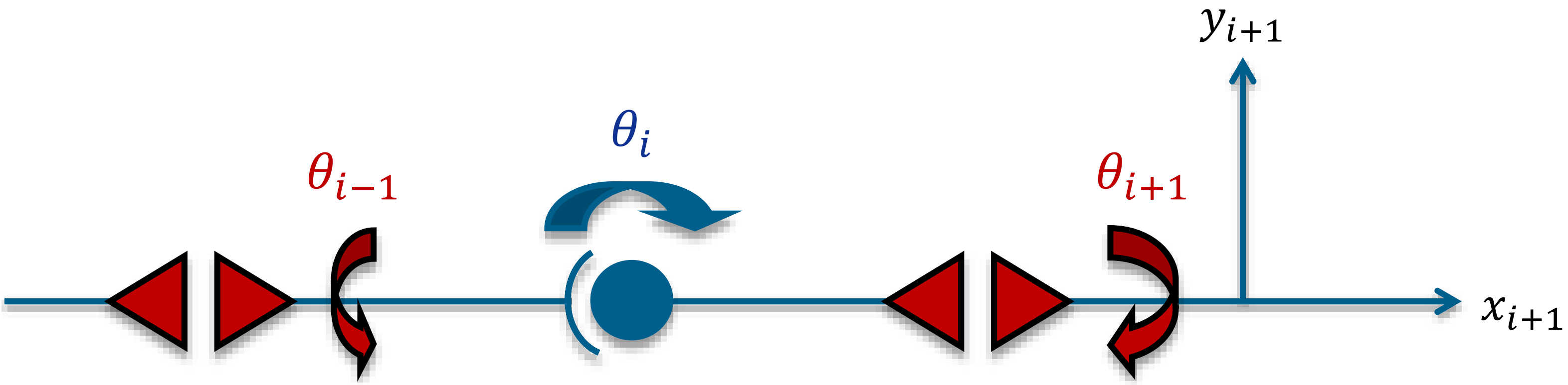


IK-Problem: Singular Configuration

Specifying constraints, e.g.:

$$f(\theta_{i-1}, \theta_{i+1}) = a\left({}^{t-1}\theta_{i-1} - {}^{t}\theta_{i-1}\right)^2 + b\left({}^{t-1}\theta_{i+1} - {}^{t}\theta_{i+1}\right)^2 = \min$$

with weights a and b



IK-Problem

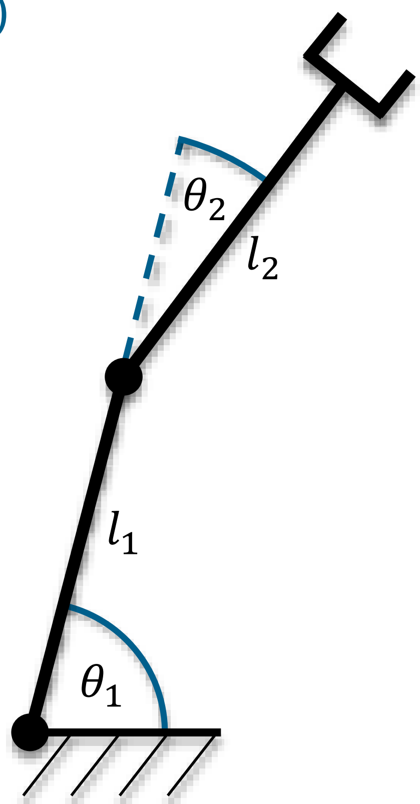
- No generally usable approach
- Velocity:
  Calculation of velocities must be fast
- Methods
  - Algebraic/Geometric methods
    (solution in closed form)
  - Numerical methods

Example: Planar 2-Link Robot Arm

$$x = l_1 \cos\theta_1 + l_2 \cos(\theta_1 + \theta_2)$$
$$y = l_1 \sin\theta_1 + l_2 \sin(\theta_1 + \theta_2)$$
$$\phi = \theta_1 + \theta_2$$

Abbreviations:

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



Algebraic Solution

- Forward kinematics:

$$ {}^{0}_{2}T = \begin{bmatrix} c_{12} & -s_{12} & 0 & l_1 c_1 + l_2 c_{12} \\ s_{12} & c_{12} & 0 & l_1 s_1 + l_2 s_{12} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

- Position and orientation of end effector by:

$$ {}^{BASIS}_{TCP}T = \begin{bmatrix} c\varphi & -s\varphi & 0 & x \\ s\varphi & c\varphi & 0 & y \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

Algebraic Solution

- Comparison of coefficients

$$
\begin{aligned}
c\varphi &= c_{12} \\
s\varphi &= s_{12} \\
x &= l_1 c_1 + l_2 c_{12} \\
y &= l_1 s_1 + l_2 s_{12}
\end{aligned}
$$

- Sum of squares for the last two equations

$$x^2 + y^2 = l_1^2 + l_2^2 + 2 l_1 l_2 c_2$$

- therefore $c_2 = \frac{x^2 + y^2 - l_1^2 - l_2^2}{2 l_1 l_2}$

Algebraic Solution

- When does a solution exist?
- Why two solutions for θ_2?

Calculation of θ_1

$$\cos(\theta_1+\theta_2) = c_{12} = c_1 c_2 - s_1 s_2$$
$$\sin(\theta_1+\theta_2) = s_{12} = c_1 s_2 + c_2 s_1$$

$$\begin{aligned} x &= k_1 c_1 - k_2 s_1 \\ y &= k_1 s_1 + k_2 c_1 \\ k_1 &= l_1 + l_2 c_2 \\ k_2 &= l_2 s_2 \end{aligned}$$

$$T_{0,2} = \begin{bmatrix} c_{12} & -s_{12} & 0 & l_1 c_1 + l_2 c_{12} \\ s_{12} & c_{12} & 0 & l_1 s_1 + l_2 s_{12} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Let $r = \sqrt{k_1^2 + k_2^2}$ and $\gamma = \mathrm{ATAN2}(k_2, k_1)$

Algebraic Solution

$$k_1 = r \cdot \cos\gamma$$
$$k_2 = r \cdot \sin\gamma$$
$$x/_r = \cos\gamma\cos\theta_1 - \sin\gamma\sin\theta_1$$
$$y/_r = \cos\gamma\sin\theta_1 + \sin\gamma\cos\theta_1$$

or

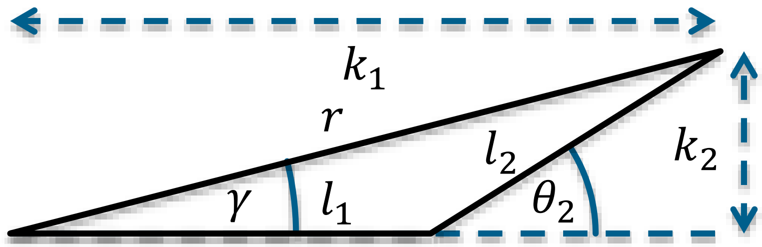
$$x/_r = \cos(\gamma + \theta_1)$$
$$y/_r = \sin(\gamma + \theta_1)$$

$$\gamma + \theta_1 = \mathrm{ATAN2}\left(\frac{y}{r}, \frac{x}{r}\right) = \mathrm{ATAN2}(y, x)$$
$$\rightarrow \theta_1 = \mathrm{ATAN2}(y, x) - \mathrm{ATAN2}(k_2, k_1)$$

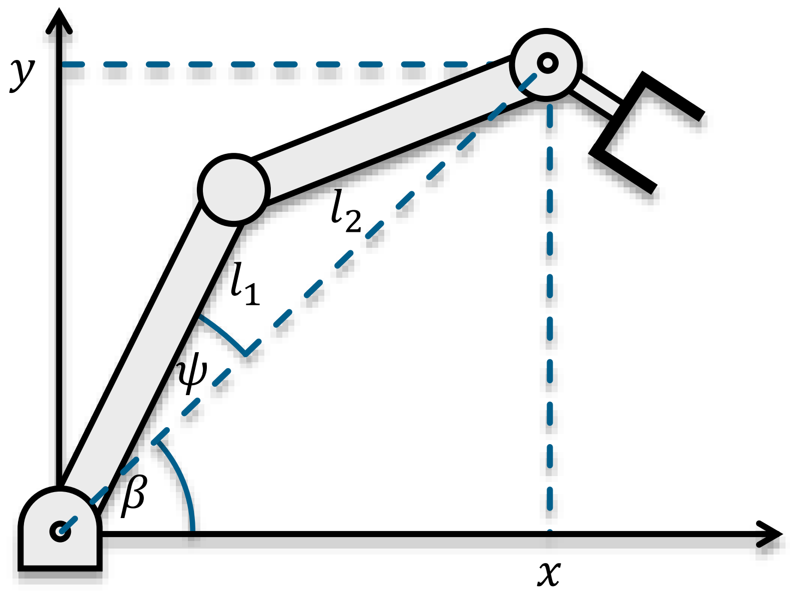
$$k_1 = l_1 + l_2 c_2$$
$$k_2 = l_2 s_2$$
$$r = \sqrt{k_1^2 + k_2^2}$$
$$\gamma = \mathrm{ATAN2}(k_2, k_1)$$
$$x = k_1 c_1 - k_2 s_1$$
$$y = k_1 s_1 + k_2 c_1$$

Constraint for angles:

$$\phi = \theta_1 + \theta_2$$



Geometric Solution

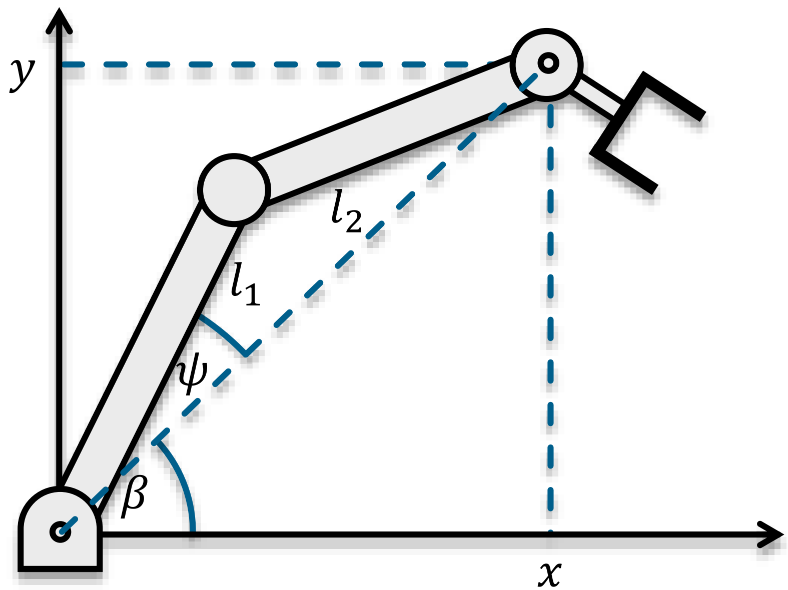


Law of cosine:

$$x^2 + y^2 = l_1^2 + l_2^2 - 2l_1 l_2 \cos(180 - \theta_2)$$

$$\rightarrow \cos\theta_2 = \frac{x^2 + y^2 - l_1^2 - l_2^2}{2l_1 l_2}$$

Geometric Solution



$$l_2^2 = x^2 + y^2 + l_1^2 - 2l_1\sqrt{x^2 + y^2}\cos\psi$$

$$\cos\psi = \frac{x^2 + y^2 + l_1^2 - l_2^2}{2l_1\sqrt{x^2 + y^2}}$$

$$\theta_1 = \beta \pm \psi \text{ with } 0 \leq \psi \leq 180$$

The IK-Problem

- Calculate T_{TCP} by multiplying with homogeneous transformation matrices

$$ {}^{\text{BASIS}}_{\text{TCP}}T = {}^{\text{BASIS}}_{1}A(\theta_1) \cdot {}^{1}_{2}A(\theta_2) \cdots {}^{n-2}_{n-1}A(\theta_{n-1}) \cdot {}^{n-1}_{n}A(\theta_n) \qquad (1) $$

- T_{TCP} is a homogeneous 4×4 matrix describing the desired position and orientation of the end effector

$$ {}^{\text{BASIS}}_{\text{TCP}}T = \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (2) $$

Algorithms to Solve IK-Problems

- Given: Transformation matrix (where e.g. $n = 6$)
- Wanted: Joint angles θ_1 to θ_n

1. Invert ${}^0_1A(\theta_i)$ and multiply (1) by ${}^0_1A^{-1}$ from both sides
2. Find in the newly formed equation system one equation which has only one unknown and solve it
3. Find equations in the equation system, which can be solved by substituting the angle we found in the last step
4. If there is no suitable equation, invert the matrix ${}^{i}_{i+1}A(\theta_{i+1})$
5. Repeat steps $1 - 4$ until all joint angles are found

Closed Form Solutions

By inverting transformation matrices and multiplying from the left or the right one creates new matrix equations which might give a closed form solution for some angles.

$$ {}^{BASIS}_{1}A^{-1} \cdot {}^{Basis}_{TCP}T = {}^{1}_{2}A \cdot {}^{2}_{3}A \cdot {}^{3}_{4}A \cdot {}^{4}_{5}A \cdot {}^{5}_{6}A $$

$$ {}^{1}_{2}A^{-1} \cdot {}^{BASIS}_{1}A^{-1} \cdot {}^{Basis}_{TCP}T = {}^{2}_{3}A \cdot {}^{3}_{4}A \cdot {}^{4}_{5}A \cdot {}^{5}_{6}A $$

$$ {}^{2}_{3}A^{-1} \cdot {}^{1}_{2}A^{-1} \cdot {}^{BASIS}_{1}A^{-1} \cdot {}^{Basis}_{TCP}T = {}^{3}_{4}A \cdot {}^{4}_{5}A \cdot {}^{5}_{6}A $$

$$ {}^{3}_{4}A^{-1} \cdot {}^{2}_{3}A^{-1} \cdot {}^{1}_{2}A^{-1} \cdot {}^{BASIS}_{1}A^{-1} \cdot {}^{Basis}_{TCP}T = {}^{4}_{5}A \cdot {}^{5}_{6}A $$

$$ {}^{4}_{5}A^{-1} \cdot {}^{3}_{4}A^{-1} \cdot {}^{2}_{3}A^{-1} \cdot {}^{1}_{2}A^{-1} \cdot {}^{BASIS}_{1}A^{-1} \cdot {}^{Basis}_{TCP}T = {}^{5}_{6}A $$

$$ {}^{Basis}_{TCP}T \cdot {}^{5}_{6}A^{-1} = {}^{BASIS}_{1}A \cdot {}^{1}_{2}A \cdot {}^{2}_{3}A \cdot {}^{3}_{4}A \cdot {}^{4}_{5}A $$

$$ {}^{Basis}_{TCP}T \cdot {}^{5}_{6}A^{-1} \cdot {}^{4}_{5}A^{-1} = {}^{BASIS}_{1}A \cdot {}^{1}_{2}A \cdot {}^{2}_{3}A \cdot {}^{3}_{4}A $$

$$ {}^{Basis}_{TCP}T \cdot {}^{5}_{6}A^{-1} \cdot {}^{4}_{5}A^{-1} \cdot {}^{3}_{4}A^{-1} = {}^{BASIS}_{1}A \cdot {}^{1}_{2}A \cdot {}^{2}_{3}A $$

$$ {}^{Basis}_{TCP}T \cdot {}^{5}_{6}A^{-1} \cdot {}^{4}_{5}A^{-1} \cdot {}^{3}_{4}A^{-1} \cdot {}^{2}_{3}A^{-1} = {}^{BASIS}_{1}A \cdot {}^{1}_{2}A $$

$$ {}^{Basis}_{TCP}T \cdot {}^{5}_{6}A^{-1} \cdot {}^{4}_{5}A^{-1} \cdot {}^{3}_{4}A^{-1} \cdot {}^{2}_{3}A^{-1} \cdot {}^{1}_{2}A^{-1} = {}^{BASIS}_{1}A $$

Jacobi-Matrix

Let $f: \mathbb{R}^n \rightarrow \mathbb{R}$ be the total derivative of y

$\vec{y} = f(\vec{x})$ for $\vec{x}, \vec{y} \in \mathbb{R}^n$

$$
\begin{gathered}
\overrightarrow{y_1} = f_1(x_1, x_2, \dots, x_n) \\
\overrightarrow{y_1} = f_1(x_1, x_2, \dots, x_n) \\
\vdots \\
\overrightarrow{y_n} = f_n(x_1, x_2, \dots, x_n)
\end{gathered}
$$

$$
\begin{gathered}
dy_1 = \frac{df_1}{dx_1} dx_1 + \frac{df_1}{dx_2} dx_2 + \cdots + \frac{df_1}{dx_n} dx_n \\
dy_2 = \frac{df_2}{dx_1} dx_1 + \frac{df_2}{dx_2} dx_2 + \cdots + \frac{df_2}{dx_n} dx_n \\
\vdots \\
dy_n = \frac{df_n}{dx_1} dx_1 + \frac{df_n}{dx_2} dx_2 + \cdots + \frac{df_n}{dx_n} dx_n
\end{gathered}
$$

Jacobi-Matrix in Vector Notation

- Vector notation

$$\begin{pmatrix} dy_1 \\ dy_2 \\ \vdots \\ dy_n \end{pmatrix} = \begin{pmatrix} \frac{df_1}{dx_1} & \frac{df_1}{dx_2} & \cdots & \frac{df_1}{dx_n} \\ \frac{df_2}{dx_1} & \frac{df_2}{dx_2} & \cdots & \frac{df_2}{dx_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{df_n}{dx_1} & \frac{df_n}{dx_2} & \cdots & \frac{df_n}{dx_n} \end{pmatrix} \begin{pmatrix} dx_1 \\ dx_2 \\ \vdots \\ dx_3 \end{pmatrix}$$

- Alternatively $d\vec{y} = df(\vec{x}) = \frac{df(\vec{x})}{d\vec{x}} d\vec{x}$ with a Jacobi-matrix $J(\vec{x}) = \frac{df(\vec{x})}{d\vec{x}}$

Numeric Methods

$$\vec{x}(t) = f\left(\vec{\theta}(t)\right) \Rightarrow \frac{d\vec{x}(t)}{dt} = \dot{\vec{x}}(t) = J\left(\vec{\theta}\right)\dot{\vec{\theta}}(t)$$

$$J\left(\vec{\theta}\right) \in R^{n\times m} \quad J_{ij} = \frac{df_i}{d\theta_j} \quad 1 \leq i \leq m \quad 1 \leq j \leq n$$

- Cartesian degrees of freedom m
- Number of joints n
- Translation and angle velocities of the TCP
  (e.g. differential temporal change of the Euler-angles)
  $$\dot{\vec{x}}(t) = \left(\dot{p}_x, \dot{p}_y, \dot{p}_z, \dot{\alpha}, \dot{\beta}, \dot{\gamma}\right)^T$$
- Difference quotient instead of differential quotient
  $$\Delta\vec{\theta} = J\left(\vec{\theta}\right)^{-1}\Delta\vec{x}$$

Numeric Methods

- Calculate change in description vector $\Delta\vec{x}$
- Calculate needed changes in joint angles $\Delta\vec{\theta}$ via the inverse Jacobi-matrix
- Approximate solution, since when $\Delta\vec{x}$ changes a constant Jacobi-Matrix is assumed for the corresponding $\Delta\vec{\theta}$ changes
- After calculating $\Delta\vec{\theta}$ the Jacobi-matrix is updated
- Iteratively decreasing error

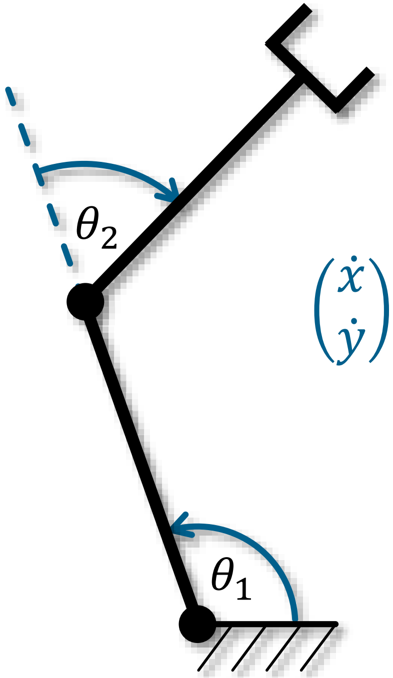
Numeric Methods

- $n = m$: Non redundant manipulator
  - Jacobi Matrix is square and can be inverted
- $n > m$: Underdetermined system
  - Redundant manipulator
  - Invers Jacobi-matrix does not exist
  - Generalized invers of Jacobi-matrix „pseudoinverse“
- $n < m$: Overdetermined system
  - Often no solution or only a subspace
  - Invers Jacobi-matrix does not exist
  - Generalized invers of Jacobi-matrix „pseudoinverse“

Numeric Methods

- Approaches can be used for all robot types with arbitrary degrees of freedom
- Further problems
  - Susceptible for singularities
  - Long runtimes
  - An arbitrary solution will be found

Example: Planar 2-Link Robot Arm



$$x = l_1 \cos\theta_1 + l_2 \cos(\theta_1 + \theta_2)$$
$$y = l_1 \sin\theta_1 + l_2 \sin(\theta_1 + \theta_2)$$

$$\begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} = J(\vec{\theta})\dot{\vec{\theta}}$$
$$= J(\vec{\theta}) \cdot \begin{pmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{pmatrix}$$
$$= \begin{bmatrix} -l_1 \sin\theta_1 - l_2 \sin(\theta_1 + \theta_2) & -l_2 \sin(\theta_1 + \theta_2) \\ l_1 \cos\theta_1 + l_2 \cos(\theta_1 + \theta_2) & l_2 \cos(\theta_1 + \theta_2) \end{bmatrix} \cdot \begin{pmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{pmatrix}$$

Example: Planar 2-Link Robot Arm

- The Jacobi-matrix needs to be inverted

$$\begin{pmatrix}\Delta\theta_1\\ \Delta\theta_2\end{pmatrix} = \underbrace{\frac{1}{l_1 l_2 \sin\theta_2} = \begin{bmatrix} l_2 c_{12} & l_2 s_{12} \\ -l_1 c_{12} - l_1 c_1 & -l_1 s_{12} - l_1 s_1 \end{bmatrix}}_{J(\vec{\theta})^{-1}} \begin{pmatrix}\Delta x\\ \Delta y\end{pmatrix}$$

- If $\theta_2 = 0, \pm 180$ then $J(\theta)$ is singular!
- All singular configurations are at the boundaries of the working space
- Abbreviations
  - $c_{12} = \cos(\theta_1 + \theta_2)$
  - $s_{12} = \sin(\theta_1 + \theta_2)$
  - $c_i = \cos\theta_i$
  - $s_i = \sin\theta_i$

Numeric Methods: Optimization

Overdetermined system $(n < m)$

- Not enough degrees of freedom to reach pose
- Approximate solutions yield $\left\| J\Delta\vec{\theta} - \Delta\vec{x} \right\|^2 \rightarrow min$
- Via pseudo-inverse $J^+ \coloneqq (J^T J)^{-1} J^T$ one gets $\Delta\vec{\theta} = J^+ \Delta\vec{x}$
- With difference vector $\Delta\vec{x} = \overrightarrow{x_{target}} - \overrightarrow{x_{current}} = \left(\Delta p_x, \Delta p_y, \Delta p_z, \Delta\alpha, \Delta\beta, \Delta\gamma\right)^T$

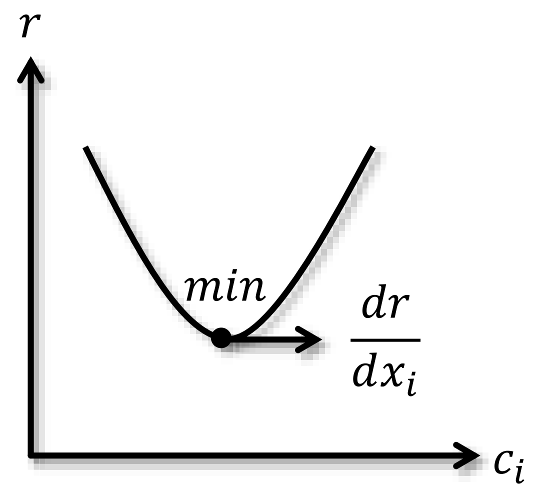
Optimization

Minimization of mean squared error (Gauß)

- Overdetermined system $J \cdot \Delta\vec{x} = \Delta\vec{\theta}$ where $i = 1, \dots, m > n$

$$\sum_{k=1}^{m} j_{ik} \Delta x_k = \theta_i$$

- Minimize $r = \sum_{i=1}^{m} (\sum_{k=1}^{n} j_{ik} \Delta x_k - \Delta\theta_i)^2 \rightarrow min$
- (Necessary) Condition is $\nabla r = 0$, i.e. $\frac{\mathrm{d}r}{\mathrm{dx_i}} \overset{!}{=} 0$ for $i = 1, \dots, n$



Optimization

- Minimization of mean squared error

$$\frac{dr}{dx_i} = \sum_{l=1}^{m} \frac{d}{dx_i}\left(\sum_{k=1}^{n} j_{lk}\Delta x_k - \Delta\theta_l\right)^2 \quad \text{(Chain rule)}$$

$$= \sum_{l=1}^{m} 2\left(\sum_{k=1}^{n} j_{lk}\Delta x_k - \Delta\theta_l\right)\frac{d}{dx_i}\sum_{k=1}^{n} j_{lk}\Delta x_k$$

$$= 2\sum_{l=1}^{m}\left(\sum_{k=1}^{n} j_{lk}\Delta x_k - \Delta\theta_l\right) j_{li}$$

$$= 2\sum_{l=1}^{m}\sum_{k=1}^{n} j_{li} j_{lk}\Delta x_k - 2\sum_{l=1}^{m} j_{li}\Delta\theta_l \stackrel{!}{=} 0$$

- System can now be written as $J^T \cdot J \cdot \Delta\vec{x} = J^T \cdot \Delta\vec{\theta}$ or $\vec{x} = (J^T \cdot J)^{-1} \cdot J^T \cdot \vec{\theta}$

Numeric Methods: Optimization

Underdetermined system $(n > m)$

- Too many degrees of freedom
- Additional conditions,
  e.g. „most natural" joint angles $\Delta\theta_i'$
- Constrained optimization $J\Delta\vec{\theta} = \Delta\vec{x}$:

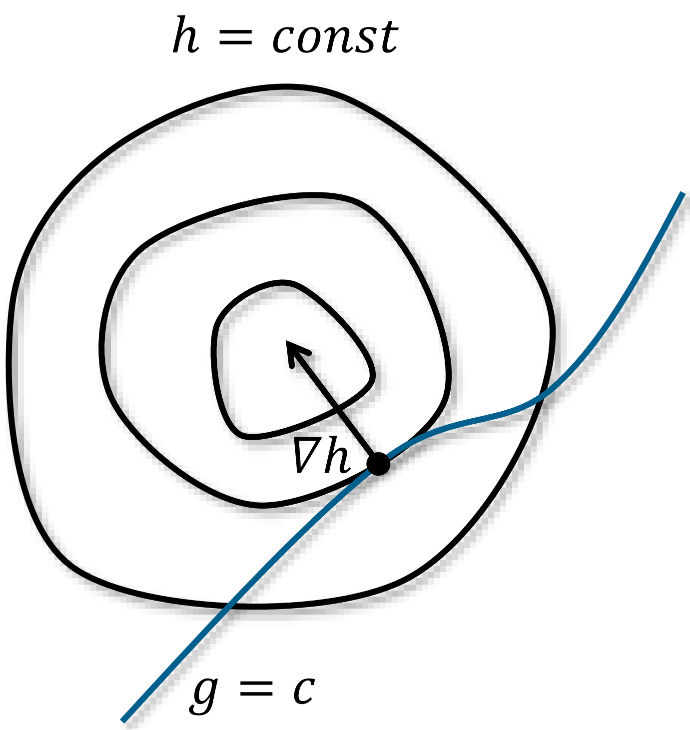
$$h := \sum_{i=1}^{n} w_i \left(\Delta\theta_i - \Delta\theta'_i\right)^2 \to \min$$

- Solution via Lagrange multiplier

$$r := h + \lambda^T\left(J\Delta\vec{\theta} - \Delta\vec{x}\right) \to min$$

Optimization: Lagrange Multiplier

- Idea: Find extrema of a function $h(x_1, \dots, x_n)$ constrained to $g(x_1, \dots, x_n) = c$
- Solution: $g \wedge \nabla h = \lambda h \nabla g$



Optimization: Lagrange Multiplier

Given:

- m „strict" constraints (end effector, comparison $g = c$) of the form $J \cdot \Delta\vec{x} = \Delta\vec{\theta}$

$$m \; \boxed{\quad J \quad} \; \underset{n}{\Big|} \; \Delta x = \underset{m}{\big|} \; \Delta\theta$$

- l „soft" constraints („natural" joints, $h \rightarrow min$) $A \cdot \Delta\vec{x} = \vec{q}$

$$l \; \overset{n}{\boxed{\quad A \quad}} \; \underset{n}{\Big|} \; \Delta x = \underset{l}{\big|} \; q$$

- With unknowns $x_i, (i = 1, \ldots, n), n > m,$

Optimization: Lagrange Multiplier

Approach:

- Minimize

$$r = \sum_{k=1}^{l}\left(\sum_{r=1}^{n} a_{kr}\Delta x_r - q_k\right)^2 + \sum_{k=1}^{m}\lambda_k\left(\sum_{r=1}^{n} j_{kr}\Delta x_r - \Delta\theta_k\right) \rightarrow \min$$

- r depends on $x_i, (i = 1, \ldots, n)$ and $\Delta_k, (k = 1, \ldots, m)$ on

$$\frac{dr}{dx_i} = 2\sum_{k=1}^{l}\left(\sum_{r=1}^{n} a_{kr}\Delta x_r - q_k\right) a_{ki} + \sum_{k=1}^{m}\lambda_k j_{ki} \overset{!}{=} 0$$

(Soft constraints, $\nabla h = \lambda \cdot \nabla g$)

$$\frac{dr}{d\lambda_i} = \sum_{r=1}^{n} j_{ir}\Delta x_r - \Delta\theta_i \overset{!}{=} 0$$

(Strict constraints, $g = c$)

Optimization: Lagrange Multiplier

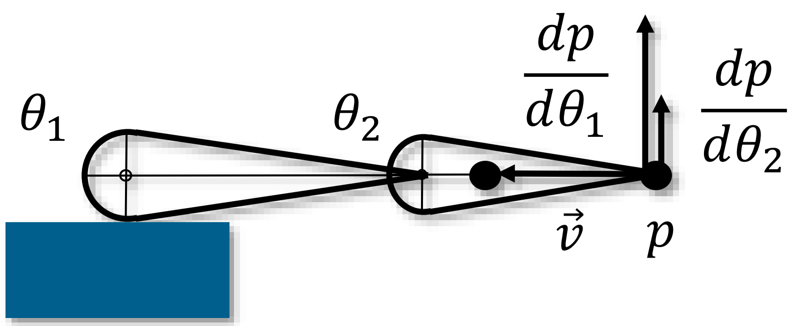
- Matrix form: $\begin{pmatrix} 2A^T A & J^T \\ J & 0 \end{pmatrix} \begin{pmatrix} \Delta\vec{x} \\ \vec{\lambda} \end{pmatrix} = \begin{pmatrix} 2A^T\vec{q} \\ \Delta\vec{\theta} \end{pmatrix}$
- Notes
  - Solution only consists of x_i
  - Lagrange multipliers λ_k are just used in order to find the solution

Optimization: Algorithm

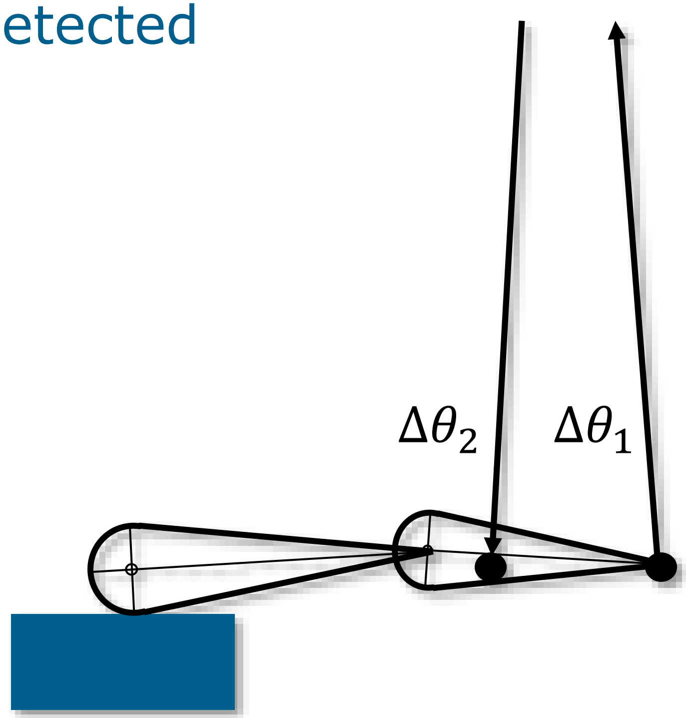
(1) Initial values for Θ and translation $\vec{v}$
(2) Determine $J(\Theta)$
(3) Eliminate redundant parameters as necessary (singularities)
(4) Solve for $\Delta\Theta$ (optimization if necessary with constraints)
(5) Update Θ and $\vec{v}$
(6) If solution not yet reached go back to (2)

Optimization: Problems

- Singular situations need to be detected
- Restrict $\Delta\Theta$



Blocked manipulator



Surrounding of singularity

Direct and Invers Kinematic

- Direct kinematic: $f\colon R^n \rightarrow R^m, \vec{x} = f\left(\vec{\theta}\right)$
- Invers kinematic: $f^{-1}\colon R^m \rightarrow R^n, \vec{\theta} = f^{-1}(\vec{x})$
- There exists …
  - … an unique solution
  - … a finite set of solutions
  - … an infinite set of solutions
  - … no solution

IK-Problems

	General Approaches	Special Approaches
Procedure	Iteration, general solving approach for equation systems	Graphic approaches based on trigonometric relations
Pros	General	Fast
Cons	Computationally expensive, slow	Only for special robot setups

Next Lecture

Robot modelling

- Velocity of rigid bodies
- Calculation of static forces/torques