Zusammenfassung - Robotik

Julian Shen

1. Juni 2023

1 Mathematische Grundlagen

Kinematik ist die reine geometrische Beschreibung von Bewegung eines Manipulators oder Roboters. Das essentielle Konzept ist die **Position**.

Statik behandelt Kräfte und Momente, die sich auf einen ruhenden Mechanismus auswirken. Das essentielle Konzept ist die **Steifigkeit**.

Dynamik analysiert die Kräfte und Momente, die durch Bewegung und Beschleunigung eines Mechanismus und einer zusätzlichen Last entstehen.

Freiheitsgrade (DoF) ist die Anzahl unabhängiger Parameter, die zur kompletten Spezifikation der Lage eines Objekts benötigt werden, z.B. Starrkörper hat in 2D 3 DoF und in 3D 6 DoF.

Starrkörperbewegungen werden durch zwei Eigenschaften charakterisiert:

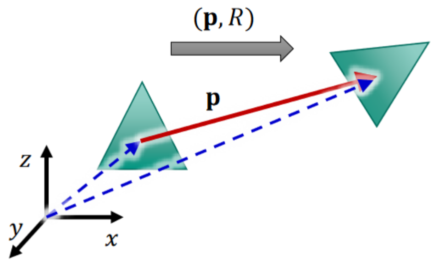
1. Distanz zweier beliebiger Punkte ist konstant
2. Orientierungen im Körper bleiben erhalten

SO(3) und SE(3):

- SO(3): **Spezielle Orthogonale Gruppe**, die **Rotationen** repräsentiert
- SE(3): **Spezielle Euklidische Gruppe**, die **Transformationen** repräsentiert
- Elemente aus SO(3) werden als reale 3×3 orthogonale Matrizen R (Zeilen- und Spaltenvektoren orthonormal) beschrieben und erfüllen

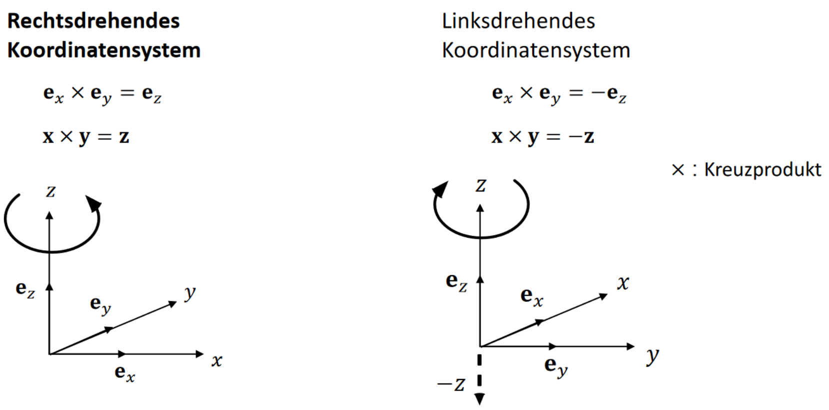
$$R^\top R = 1 \qquad \text{mit} \qquad \det(R) = 1$$

- Elemente aus SE(3) sind von der Form $(\mathbf{p}, R)$ mit $\mathbf{p} \in \mathbb{R}^3$ und $R \in \mathrm{SO}(3)$ und beschreiben Verknüpfungen von Rotationen und Translationen



Euklidischer Raum: Vektorraum $\mathbb{R}^3$ mit dem Skalarprodukt.

- Punkt **a** im euklidischen Raum wird durch Vielfache der Einheitsvektoren $\mathbf{e_x}, \mathbf{e_y}, \mathbf{e_z}$ beschrieben
- Wir benutzen **rechtsdrehende Koordinatensysteme**



Lineare Abbildungen (Transformationen), die den euklidischen Raum auf sich selbst abbilden, nennt man **Endomorphismen**:

$$\phi(\cdot) \colon \mathbb{R}^3 \to \mathbb{R}^3$$

- Endomorphismen können durch quadratische Matrizen repräsentiert werden:

$$\phi(\mathbf{a}) = A \cdot \mathbf{a}, \qquad A \in R^{3\times 3}$$

- A beschreibt einen Basiswechsel zwischen den originalen Basisvektoren $\mathbf{e_x}, \mathbf{e_y}, \mathbf{e_z}$ und den neuen Basisvektoren $\mathbf{e'_x}, \mathbf{e'_y}, \mathbf{e'_z}$:

$$A = (\mathbf{e'_x} \quad \mathbf{e'_y} \quad \mathbf{e'_z}) \cdot (\mathbf{e_x} \quad \mathbf{e_y} \quad \mathbf{e_z})^{-1}$$

Bijektive Endomorphismen nennt man **Isomorphismen**.

- Eigenschaften:
  1. Winkel bleiben erhalten
  2. Längen bleiben erhalten
  3. Händigkeit beleibt erhalten
- Eine spezielle Art von Isomorphismen ist die **Rotationsgruppe** SO(3)

Rotationsgruppe SO(3):

- SO(3) ist nicht kommutativ: $A \cdot B \cdot \mathbf{x} \neq B \cdot A \cdot \mathbf{x}$ mit $\mathbf{x} \in \mathbb{R}^3$ und $A, B \in \mathrm{SO}(3)$
- Für alle $R \in \mathrm{SO}(3)$ ist $R^{-1} = R^\top$, die Inverse kann also leicht berechnet werden

Rotationen in 2D:

- Rotation in der xy-Ebene um $(0,0)$ ist eine **lineare Transformation**
- **Rotationsmatrix**: $R_\alpha(\mathbf{x}) = \begin{pmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{pmatrix} \cdot \mathbf{x}$ mit $RR^\top = R^\top R = I$ und $\det(R) = 1$
- Rotation um einen Punkt $\mathbf{c} \neq (0,0)$ ist keine lineare Transformation. Verschiebe dafür die Ebene um $-\mathbf{c}$, rotiere und verschiebe wieder um $+\mathbf{c}$ zurück:

$$R_{\mathbf{c},\alpha} = R_\alpha(\mathbf{x} - \mathbf{c}) + \mathbf{c} = R_\alpha(\mathbf{x}) + (-R_\alpha(\mathbf{c}) + \mathbf{c})$$

- $R_{\mathbf{c},\alpha}$ ist eine nichtlineare Transformation und heißt **affine Transformation**. Sie unterscheidet sich von R_α nur durch das Addieren einer Konstante

Rotationen in 3D:

- Eine 2D Rotation in der xy-Ebene ist eine 3D Rotation um die z-Achse:

$$R_{\mathbf{z},\alpha} = \begin{pmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

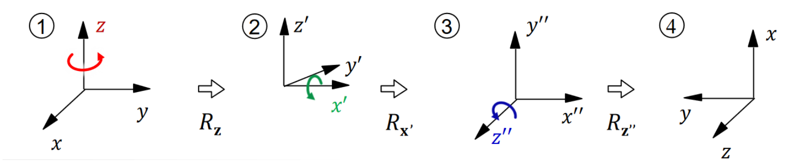
- Rotationen können verkettet werden: $\phi_{\mathbf{z},\gamma}\left(\phi_{\mathbf{y},\beta}\left(\phi_{\mathbf{x},\alpha}(\mathbf{a})\right)\right), \qquad \mathbf{a} \in \mathbb{R}^3$

Probleme mit Rotationsmatrizen:

- **Redundanz**: Neun Werte für eine Rotationsmatrix
- Probleme im Bereich des maschinellen Lernens

Eulerwinkel:

- Es ist möglich jede Rotation durch drei Rotationen um jeweils eine Rotationsachse darzustellen
- **Euler-Konvention**: $\mathbf{z}\ \mathbf{x}'\ \mathbf{z}''$ (lokale Drehung, Drehung verändert Achsen) oder $\mathbf{x}\ \mathbf{y}\ \mathbf{z}$ (globale Drehung, Drehung um feste Achsen)
- Winkel α, β, γ sind **Eulerwinkel** und beschreiben den Grad der Drehungen



- **Vorteile**: Kompakter und aussagekräftiger als Rotationsmatrizen

- **Nachteile**:
  - Nicht eindeutig: In der Euler-Konvention $\mathbf{x}\ \mathbf{y}'\ \mathbf{z}''$ beschreiben die Eulerwinkel $(45°, -90°, 45°)$ und und $(30°, -90°, 60°)$ die gleiche Rotation
  - Nicht kontinuierlich: Kleine Änderung in der Orientierung können zu großen Änderungen der Eulerwinkel führen
  - **Gimbal Lock**: Bei bestimmten Winkeln werden zwei Achsen voneinander abhängig $\Rightarrow$ Ein Freiheitsgrad geht verloren

Bewertung der Darstellung von Orientierung mit 3×3-Matrizen:

- **Vorteil**: Vektor und Rotationsmatrix sind anschaulich $\Rightarrow$ übliche Form der Eingabe von Posen
- **Nachteil**: Darstellung als $(\mathbf{p}, R)$ mit $\mathbf{p} \in \mathbb{R}^3$ und $R \in \mathrm{SO}(3)$ führt dazu, dass Translation und Rotation getrennt durchgeführt werden müssen

$\rightarrow$ **Ziel**: Geschlossene Darstellung von Rotation und Translation in einer Matrix

Affine Transformationen:

- Der **affine Raum** ist eine Erweiterung zum euklidischen Raum
- Beinhaltet Vektoren, die in **erweiterten, homogenen Koordinaten** ausgedrückt werden: $a = (a_x \quad a_y \quad a_z \quad h)^\top, h \in \{0, 1\}$, wobei a für $h = 0$ einen Ortsvektor und für $h = 1$ einen Richtungsvektor beschriebt
- Für Rotationsmatrix R und Translation $\mathbf{t}$ gilt nun:

$$\mathbf{b} = A \cdot \mathbf{x} + \mathbf{t} \Leftrightarrow \begin{pmatrix} \mathbf{b} \\ 1 \end{pmatrix} = \begin{pmatrix} A & \mathbf{0} \\ \mathbf{0}^\top & 1 \end{pmatrix} \begin{pmatrix} \mathbf{x} \\ 1 \end{pmatrix} + \begin{pmatrix} \mathbf{t} \\ 0 \end{pmatrix} = \begin{pmatrix} A & \mathbf{t} \\ \mathbf{0}^\top & 1 \end{pmatrix} \begin{pmatrix} \mathbf{x} \\ 1 \end{pmatrix}$$

  womit also nun Translation und Rotation als eine allgemeine homogene 4×4-Matrix beschrieben werden kann:

$$T = \begin{pmatrix} R & \mathbf{t} \\ \mathbf{0}^\top & 1 \end{pmatrix}, \qquad T \in \mathrm{SE}(3) \text{ mit } \mathbf{t} \in \mathbb{R}^3 \text{ und } R \in \mathrm{SO}(3)$$

- Eine **Translationsmatrix**, die eine Verschiebung um $\mathbf{t} = (t_x \quad t_y \quad t_z)^\top$ beschreibt, ist demnach:

$$T_{\text{trans}} = \begin{pmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

- Es gilt $T^{-1} = \begin{pmatrix} R^\top & -R^\top \cdot \mathbf{t} \\ \mathbf{0}^\top & 1 \end{pmatrix}$. Diese bildet $\mathbf{b} = R \cdot \mathbf{x} + \mathbf{t}$ wieder auf $\mathbf{x}$ ab

- **Interpretationen von homogenen** 4×4**-Matrizen**:
  - **Lagebeschreibung**: ${}^{A}P_{B}$ beschreibt die Lage des Koordinatensystems B relativ zum Koordinatensystem A
  - **Transformationsabbildung** zwischen Koordinatensystemen:

    $$ {}^{A}T_{B} \colon {}^{B}P \to {}^{A}P, \qquad {}^{A}P = {}^{A}T_{B} \cdot {}^{B}P $$

    transformiert Koordinatensystem B in Koordinatensystem A
  - **Transformationsoperator** innerhalb eines Koordinatensystems:

    $$ T \colon {}^{A}P_{1} \to {}^{A}P_{2}, \qquad {}^{A}P_{2} = T \cdot {}^{A}P_{1} $$

    transformiert einen Punkt P_1 in einen Punkt P_2 innerhalb des Koordinatensystems A

  Beispiele 1/58-59

- Lagebeschreibungen können als Matrixprodukt verkettet werden, z.B.

$$ {}^{\mathrm{BKS}}T_{O_3} = {}^{\mathrm{BKS}}T_{O_1} \cdot {}^{O_1}T_{O_2} \cdot {}^{O_2}T_{O_3} $$

Quaternionen:

- Repräsentation ohne Nachteile von Rotationsmatrizen und Eulerwinkeln
- Menge der **Quaternionen** $\mathbb{H}$ ist definiert durch:

  $$ \mathbb{C} + \mathbb{C}j \qquad \text{mit } j^2 = -1 \text{ und } i \cdot j = -j \cdot i $$

  wobei i die imaginäre Einheit ist
- Ein Element $\mathbf{q} \in \mathbb{H}$ hat die Form:

  $$ \mathbf{q} = (a, \mathbf{u})^\top = a + u_1 i + u_2 j + u_3 k \qquad \text{mit } a \in \mathbb{R}, \mathbf{u} \in \mathbb{R}^3 \text{ und } k = i \cdot j $$

- a heißt **Realteil** und $\mathbf{u} = (u_1, u_2, u_3)^\top$ heißt **Imaginärteil**
- **Rechenregeln**:

  $$ \boxed{\begin{array}{ccccccccc} i^2 & = & j^2 & = & k^2 & = & i \cdot j \cdot k & = & -1 \\ i \cdot j & = & -j \cdot i & = & k & & \text{(nicht kommutativ!)} & & \\ k \cdot i & = & -i \cdot k & = & j & & & & \end{array}} $$

- **Rechenoperationen**: Seien $\mathbf{q} = (a, \mathbf{u})^\top, \mathbf{r} = (b, \mathbf{v})^\top$ zwei Quaternionen
  - **Addition**: $\mathbf{q} + \mathbf{r} = (a + b, \mathbf{u} + \mathbf{b})^\top$

- **Skalarprodukt**: $\langle \mathbf{q} \mid \mathbf{r} \rangle = a \cdot b + \langle \mathbf{v} \mid \mathbf{u} \rangle = a \cdot b + v_1 \cdot u_1 + v_2 \cdot u_2 + v_3 \cdot u_3$
  - **Multiplikation**: $\mathbf{q} \cdot \mathbf{r} = (a + u_1 i + u_2 j + u_3 k) \cdot (b + v_1 i + v_2 j + v_3 k)$
  - **Konjugierte** Quaternion: $\mathbf{q}^* = (a, -\mathbf{u})^\top$
  - **Norm**: $|\mathbf{q}| = \sqrt{\mathbf{q} \cdot \mathbf{q}^*} = \sqrt{\mathbf{q}^* \cdot \mathbf{q}} = \sqrt{a^2 + u_1^2 + u_2^2 + u_3^2}$
  - **Inverse**: $\mathbf{q}^{-1} = \dfrac{\mathbf{q}^*}{|\mathbf{q}|^2}$
- **Einheitsquaternion** $\mathbb{S}^3 = \{\mathbf{q} \in \mathbb{H} \mid \|q\|^2 = 1\}$
- Beschreibung eines Vektors $\mathbf{p} \in \mathbb{R}^3$ als Quaternion $\mathbf{q}$: $\mathbf{q} = (0, \mathbf{p})^\top$
- Beschreibung eines Skalars $s \in \mathbb{R}^3$ als Quaternion $\mathbf{q}$: $\mathbf{q} = (s, \mathbf{0})^\top$
- Sei eine Rotation beschrieben durch eine Drehachse $\mathbf{a}$ mit $\mathbf{a} = 1$ und einen Drehwinkel θ, dann existiert hierfür eine Repräsentation als Quaternion: $\mathbf{q} = \left(\cos \frac{\theta}{2}, \mathbf{a} \sin \frac{\theta}{2}\right)$
- Ein Punkt $\mathbf{v}$ wird mit einer Quaternion $\mathbf{q}$ rotiert durch: $\mathbf{v'} = \mathbf{q}\mathbf{v}\mathbf{q}^{\mathbf{-1}} = \mathbf{q}\mathbf{v}\mathbf{q}^*$, wobei die letzte Gleichheit gilt, weil $\mathbf{q}$ ein Einheitsquaternion ist
- Verkettung von Rotationen $f \circ h$: $f(h(\mathbf{v})) = \mathbf{q}(\mathbf{r}\mathbf{v}\mathbf{r}^*)\mathbf{q}^*$

Beispiel 1/72

Bewertung von Quaternionen:

- **Vorteile**: Kompakt, Anschaulich, Kein Gimbal Lock, Verkettung möglich, Stetige Repräsentation
- **Nachteil**: Nur Beschreibung von Rotation, keine Translation

SLERP Interpolation:

- SLERP Interpolation von $\mathbf{q_1}$ nach $\mathbf{q_2}$ mit Parameter $t \in [0, 1]$:

$$\mathrm{SLERP}(\mathbf{q_1}, \mathbf{q_2}, t) = \frac{\sin((1-t) \cdot \theta)}{\sin \theta} \cdot \mathbf{q_1} + \frac{\sin(t \cdot \theta)}{\sin \theta} \cdot \mathbf{q_2} \qquad \text{mit } \langle \mathbf{q_1} \mid \mathbf{q_2} \rangle = \cos \theta$$

- Ergebnis: Rotation mit konstanter Winkelgeschwindigkeit
- **Problem**: Orientierungen in SO(3) werden durch Einheitsquaternionen doppelt abgedeckt, weil $\mathbf{q}$ und $-\mathbf{q}$ der gleichen Rotation entsprechen

  $\Rightarrow$ SLERP berechnet deshalb nicht immer die kürzeste Rotation. Es muss geprüft werden, ob die Rotation von $\mathbf{q_1}$ zu $\mathbf{q_2}$ oder $-\mathbf{q_1}$ zu $\mathbf{q_2}$ kürzer ist

Duale Quaternionen:

- Erlauben es auch Translationen zu berücksichtigen
- **Duale Zahlen** sind Zahlen der Form:

$$d = p + \varepsilon \cdot s, \quad \text{wobei } \varepsilon^2 = 0$$

  mit **Primärteil** p, **Sekundärteil** s

- **Rechenoperationen**: Seien $d_1 = p_1 + \varepsilon \cdot s_1$ und $d_2 = p_2 + \varepsilon \cdot s_2$ duale Zahlen
  - **Addition**: $d_1 + d_2 = p_1 + p_2 + \varepsilon \cdot (s_1 + s_2)$
  - **Multiplikation**: $d_1 \cdot d_2 = p_1 \cdot p_2 + \varepsilon \cdot (p_1 \cdot s_2 + p_2 \cdot s_1)$
- **Duale Quaternionen**:

$$DQ = (d_1, d_2, d_3, d_4), \qquad d_i = dp_i + \varepsilon \cdot ds_i$$

  - Primärteil dp_i enthält den Winkelwert $\theta/2$
  - Sekundärteil ds_i enthält die Translationsgröße $d/2$
- **Multiplikationstabelle für duale Einheitsquaternionen**:

$\cdot$	1	i	j	k	ε	εi	εj	εk
1	1	i	j	k	ε	εi	εj	εk
i	i	-1	k	$-j$	εi	$-\varepsilon$	εk	$-\varepsilon j$
j	j	$-k$	-1	i	εj	$-\varepsilon k$	$-\varepsilon$	εi
k	k	j	$-i$	-1	εk	εj	$-\varepsilon i$	$-\varepsilon$
ε	ε	εi	εj	εk	0	0	0	0
εi	εi	$-\varepsilon$	εk	$-\varepsilon j$	0	0	0	0
εj	εj	$-\varepsilon k$	$-\varepsilon$	εi	0	0	0	0
εk	εk	εj	$-\varepsilon i$	$-\varepsilon$	0	0	0	0

- Rotation um eine Achse $\mathbf{a}$ mit dem Winkel θ: $\mathbf{q_r} = \left(\cos\frac{\theta}{2}, \mathbf{a}\sin\frac{\theta}{2}\right) + \varepsilon \cdot (0,0,0,0)$
- Translation mit dem Vektor $\mathbf{t} = (t_x, t_y, t_z)$: $\mathbf{q_t} = (1,0,0,0) + \varepsilon \cdot (0, \frac{t_x}{2}, \frac{t_y}{2}, \frac{t_z}{2})$
- Kombination zu einer Transformation T: $\mathbf{q_T} = \mathbf{q_t}\mathbf{q_r}$
- Eine Transformation $\mathbf{q_T}$ wird auf einen Punkt $\mathbf{p}$ als duale Quaternion wie folgt angewendet: $\mathbf{p}' = \mathbf{q_T}\mathbf{p}\mathbf{q_T^*}$, mit $\mathbf{q_T^*} = (\mathbf{q_t}\mathbf{q_r})^* = \mathbf{q_r^*}\mathbf{q_t^*}$
- Konjugieren von $\mathbf{q} = \mathbf{p} + \varepsilon \cdot \mathbf{s}$: $\mathbf{q}^* = \mathbf{p}^* - \varepsilon \cdot \mathbf{s}^*$

Beispiel 1/83-85

Bewertung von dualen Quaternionen:

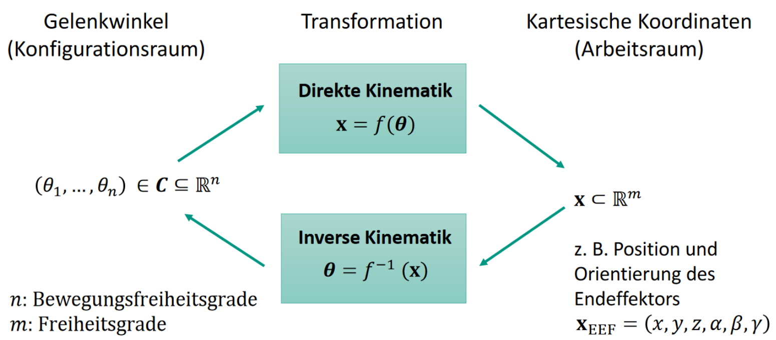
- **Vorteile**: Erlauben Lagebeschreibung und Transformationen, Geringere Redundanz (nur 8 statt 12 Werte bei homogener Matrix)
- **Nachteile**: Lage durch Angabe einer Dualquaternion zu beschreiben ist relativ schwierig, Komplexe Verarbeitungsvorschriften

2 Kinematik

Kinematisches Modell: Beschreibt Zusammenhänge zwischen **Gelenkwinkelraum** (Konfigurationsraum) und **Posenraum des Endeffektors** (Arbeitsraum)

Direkte und Inverse Kinematik:

- **Direkte Kinematik**:
  - Eingabe: Gelenkwinkelstellungen des Roboters
  - Ausgabe: Pose des Endeffektors
  - z.B. Wo befindet sich meine Hand?
- **Indirekte Kinematik**:
  - Eingabe: Zielpose des Endeffektors
  - Ausgabe: Gelenkwinkelstellungen
  - z.B. Wie bewege ich meine Hand zum Ziel?



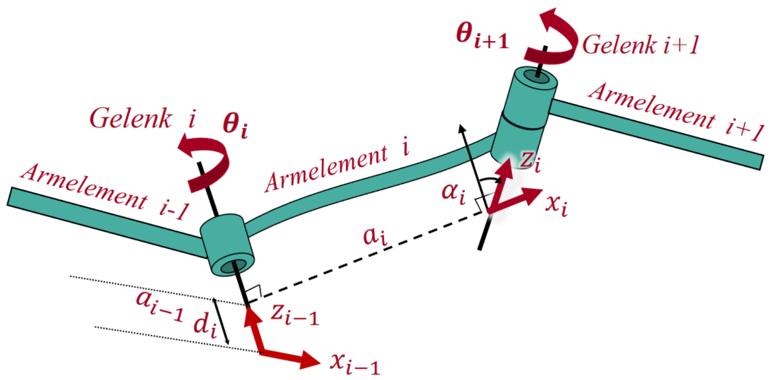
Kinematische Kette wird von mehreren Körpern gebildet, die durch Gelenke kinematisch verbunden sind (z.B. Roboterarm). Unterscheidung zwischen:

- **Offene** kinematische Kette: Nur ein Ende ist fest, anderes Ende frei bewegbar
- **Geschlossene** kinematische Kette: Beide Enden der Kette sind fest

Für jedes Glied müssen **6 Parameter** für die Transformation zwischen Gelenken bestimmt werden (3 Rotationsparameter, 3 Translationsparameter)

Denavit-Hartenberg (DH) Konvention:

- Durch die geschickte Wahl der Koordinatensysteme lassen sich die Parameter zur Beschreibung eines Armelements auf 4 reduzieren
- **Regeln für Koordinatensysteme**:
  - z_{i-1}-Achse liegt entlang der Bewegungsachse des i-ten Gelenks
  - x_i-Achse verläuft entlang der gemeinsamen Normalen (Kreuzprodukt von z_{i-1} und z_i) von z_{i-1} und z_i
  - y_i-Achse vervollständigt das Koordinatensystem entsprechend der Rechte-Hand-Regel
- **Parameter des Armelements (DH-Paramerter)**:
  - **Armelementlänge** a_i beschreibt den Abstand von z_{i-1} zu z_i
  - **Armelementverdrehung** α_i beschreibt den Winkel von z_{i-1} zu z_i um x_i
  - **Gelenkabstand** d_i ist der Abstand zwischen der x_{i-1}-Achse und x_i-Achse entlang der z_{i-1}-Achse
  - **Gelenkwinkel** θ_i ist der Winkel von x_{i-1} zu x_i um z_{i-1}



- DH-Parameter beschreiben wie aufeinanderfolgende Gelenke ineinander transformiert werden
- **DH-Transformationsmatrizen**: Beschreibung mit homogenen Matrizen

  1. Rotation θ_i: $R_{z_{i-1}}(\theta_i) = \begin{pmatrix} \cos\theta_i & -\sin\theta_i & 0 & 0 \\ \sin\theta_i & \cos\theta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$

2. Translation d_i: $T_{z_{i-1}}(d_i) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{pmatrix}$

3. Translation a_i: $T_{x_i}(a_i) = \begin{pmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$

4. Rotation α_i: $R_{x_i}(\alpha_i) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha_i & -\sin\alpha_i & 0 \\ 0 & \sin\alpha_i & \cos\alpha_i & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$

- Zusammenführen zu einer Matrix:

$$\begin{aligned} A_{i-1,i} &= R_{z_{i-1}}(\theta_i) \cdot T_{z_{i-1}}(d_i) \cdot T_{x_i}(a_i) \cdot R_{x_i}(\alpha_i) \\ &= \begin{pmatrix} \cos\theta_i & -\sin\theta_i \cdot \cos\alpha_i & \sin\theta_i \cdot \sin\alpha_i & a_i \cdot \cos\theta_i \\ \sin\theta_i & \cos\theta_i \cdot \cos\alpha_i & -\cos\theta_i \cdot \sin\alpha_i & a_i \cdot \sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{pmatrix} \end{aligned}$$

- Inverse DH-Transformation:

$$A_{i-1,i}^{-1} = A_{i,i-1} = \begin{pmatrix} \cos\theta_i & \sin\theta_i & 0 & -a_i \\ -\cos\alpha_i \cdot \sin\theta_i & \cos\theta_i \cdot \cos\alpha_i & \sin\alpha_i & -d_i \cdot \sin\alpha_i \\ \sin\theta_i \cdot \sin\alpha_i & -\sin\alpha_i \cdot \cos\theta_i & \cos\alpha_i & -d_i \cdot \cos\alpha_i \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

- Durch Multiplikation der DH-Matrizen lässt sich die Lage der einzelnen Koordinatensysteme bezüglich des Bezugskoordinatensystems bestimmen

Direktes kinematisches Problem: Stellung des Endeffektors (EFF) in Bezug auf das BKS ist gegeben durch:

$$S_{\text{Basis, EEF}} = A_{0,1}(\theta_1) \cdot A_{1,2}(\theta_2) \cdot \dots \cdot A_{n-2,n-1}(\theta_{n-1}) \cdot A_{n-1,n}(\theta_n)$$

Lösung des Problems ergibt sich aus Einsetzen der Gelenkwinkel in obige Gleichung.

Beispiele 2/38-48

Oft interessiert man sich für verwandte Beziehungen wir z.B. Gelenkwinkelgeschwindigkeiten $\rightarrow$ Endeffektor-Geschwindigkeit. Dafür muss man die Vorwärtskinematik ableiten $\rightarrow$ **Jacobi-Matrix**

Jacobi-Matrix: Für eine differenzierbare Funktion $f\colon \mathbb{R}^n \to \mathbb{R}^m$, $f(\mathbf{x}) = \begin{pmatrix} f_1(\mathbf{x}) \\ \vdots \\ f_m(\mathbf{x}) \end{pmatrix}$ und $\mathbf{x} \in \mathbb{R}^n$ ist die Jacobi-Matrix für ein $\mathbf{a} \in \mathbb{R}^n$ wie folgt:

$$J_f(\mathbf{a}) = \begin{pmatrix} \frac{\partial f_1}{\partial x_1}(\mathbf{a}) & \cdots & \frac{\partial f_1}{\partial x_n}(\mathbf{a}) \\ \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1}(\mathbf{a}) & \cdots & \frac{\partial f_m}{\partial x_n}(\mathbf{a}) \end{pmatrix} \in \mathbb{R}^{m \times n}$$

Problem: Vorwärtskinematik ist matrixwertig $\to$ Jacobi-Matrix nicht definiert

Lösung: Vektorwertige Repräsentation wählen, z.B. mit Eulerwinkel

Geschwindigkeitsraum und Kraftraum:

- Annahme: Kinematische Kette bewege sich entlang einer Trajektorie $\theta\colon \mathbb{R} \to \mathbb{R}^n$, wobei θ die Gelenkwinkelstellungen zu einem Zeitpunkt t beschreibt
- Pose des End-Effektors $\mathbf{x}(t) \in \mathbb{R}^6$ zum Zeitpunkt t: $\mathbf{x}(t) = \mathbf{f}(\boldsymbol{\theta}(t))$, wobei f die Funktion für die Vorwärtskinematik ist
- **Beziehung zwischen Endeffektor- und Gelenkwinkelgeschwindigkeiten**:

$$\dot{\mathbf{x}}(t) = J_{\mathbf{f}}(\boldsymbol{\theta}(t)) \cdot \dot{\boldsymbol{\theta}}(t)$$

- **Beziehung zwischen Kräfte und Momente am End-Effektor und Drehmomenten in den Gelenken**:

$$\boldsymbol{\tau}(t) = J_{\mathbf{f}}^{\top}(\boldsymbol{\theta}(t)) \cdot \mathbf{F}(t)$$

  wobei $\mathbf{F}\colon \mathbb{R} \to \mathbb{R}^6$ der Kraft-Momenten-Vektor am End-Effektor und $\boldsymbol{\tau}\colon \mathbb{R} \to \mathbb{R}^n$ die Drehmomente in Gelenken ist
- Im Geschwindigkeits- und Kraftraum lässt sich die Frage nach der Inversen Kinematik durch die Inverse der Jacobi-Matrix lösen, z.B. Welche Gelenkwinkelgeschwindigkeiten sind notwendig, um eine End-Effektor-Geschwindigkeit zu realisieren? $\to$ Löse nach $\dot{\boldsymbol{\theta}}(t)$ auf

Berechnung der Jacobi-Matrix:

- Jede Spalte der Jacobi-Matrix gehört zum Gelenk θ_i der kinematischen Kette
- Wenn j-tes Gelenk ein Translationsgelenk ist, das eine Translation in Richtung des Einheitsvektors $\mathbf{z_j} \in \mathbb{R}^3$ durchführt, gilt für die j-te Spalte der Jacobi-Matrix:

$$J_j(\boldsymbol{\theta}) = \begin{bmatrix} \mathbf{z_j} \\ \mathbf{0} \end{bmatrix} \in \mathbb{R}^6$$

- Wenn j-tes Gelenk ein Rotationsgelenk ist, das eine Rotation um die Rotationsachse $\mathbf{z_j} \in \mathbb{R}^3$ an der Position $\mathbf{p_j} \in \mathbb{R}^3$ durchführt, gilt für j-te Spalte der Jacobi-Matrix:

$$J_j(\boldsymbol{\theta}) = \begin{bmatrix} \mathbf{z_j} \times (\mathbf{f}(\boldsymbol{\theta}) - \mathbf{p_j}) \\ \mathbf{z_j} \end{bmatrix} \in \mathbb{R}^6$$

Beispiel 2/68-72