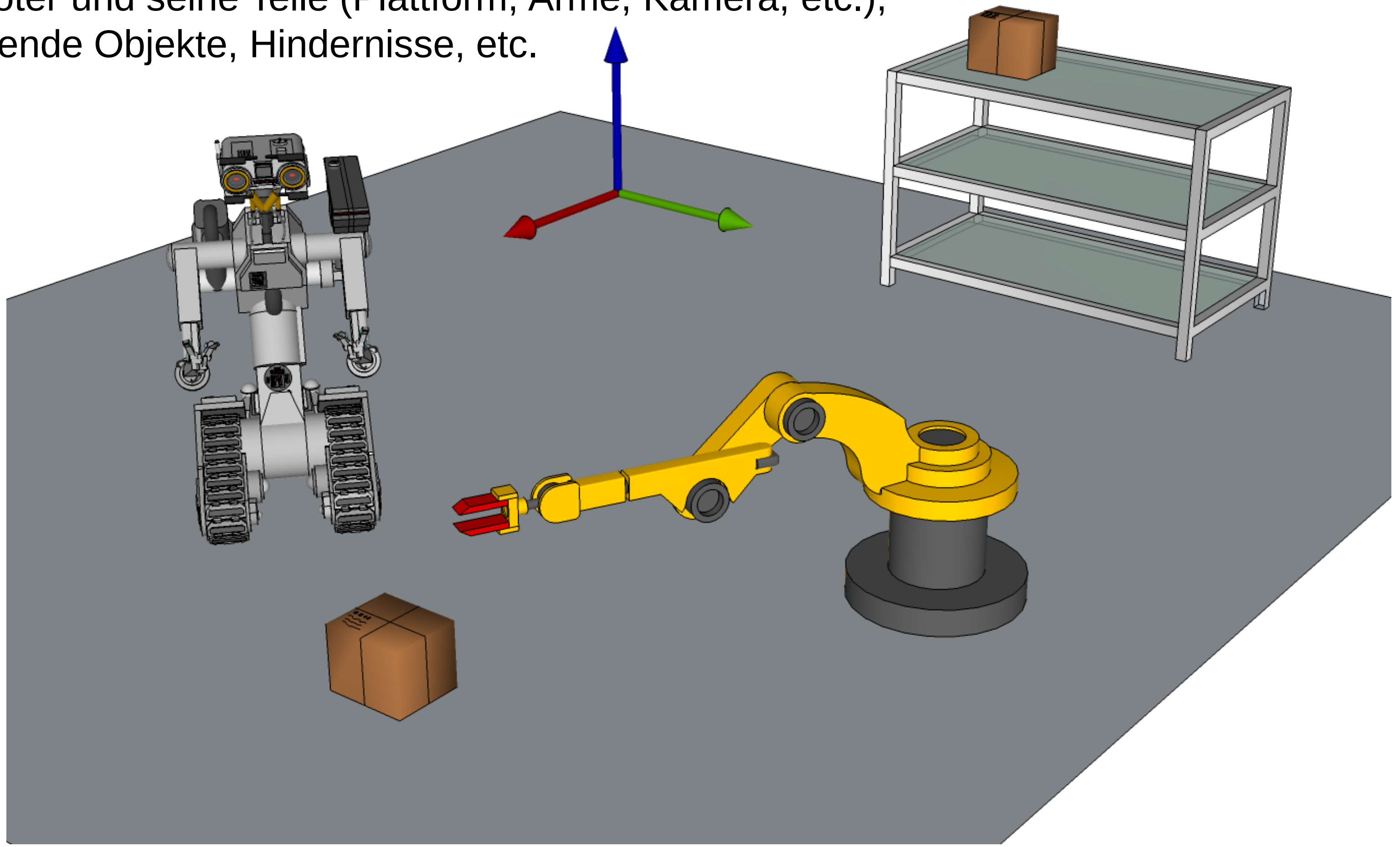
# Position und Orientierung

- Grundlagen
  - Koordinatensysteme, Punkte und Körper, Position und Orientierung
- Transformationen
  - Rotation, Translation, homogene Koordinaten, Transformationsmatrizen
- 3D-Rotationen und Eulerwinkel
- Bemerkungen
- Roboterkinematik am Beispiel Puma 560
- Weitere Darstellungsarten für Rotationen
  - Drehvektoren und Quaternionen
- Geodätische Koordinatensysteme

Unterlagen erstellt nach Vorlage der Folien von Prof. Dr. Oliver Bittel

# Position und Orientierung in Robotik-Anwendungen

- In Robotikanwendungen gibt es eine Vielzahl von Objekten, deren Position und Orientierung (Pose) in einem raumfesten Koordinatensystem beschrieben werden müssen
- Objekte: Roboter und seine Teile (Plattform, Arme, Kamera, etc.), zu manipulierende Objekte, Hindernisse, etc.

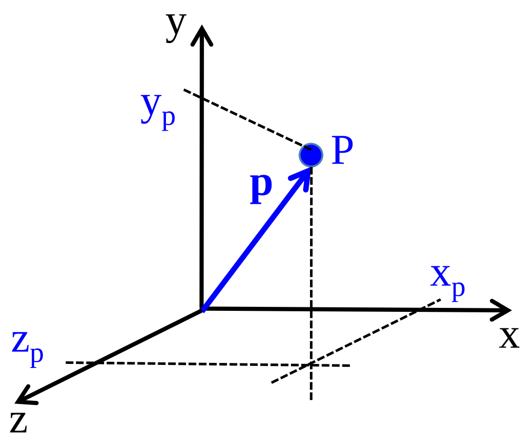


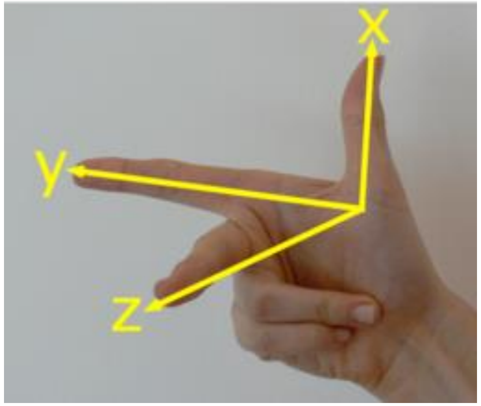
# Koordinatensysteme und Punkte

- Ein Punkt P wird dargestellt als Ortsvektor **p** in einem kartesischen Koordinatensystem (KS):

$$\mathbf{p} = \begin{pmatrix} x_p \\ y_p \\ z_p \end{pmatrix}$$

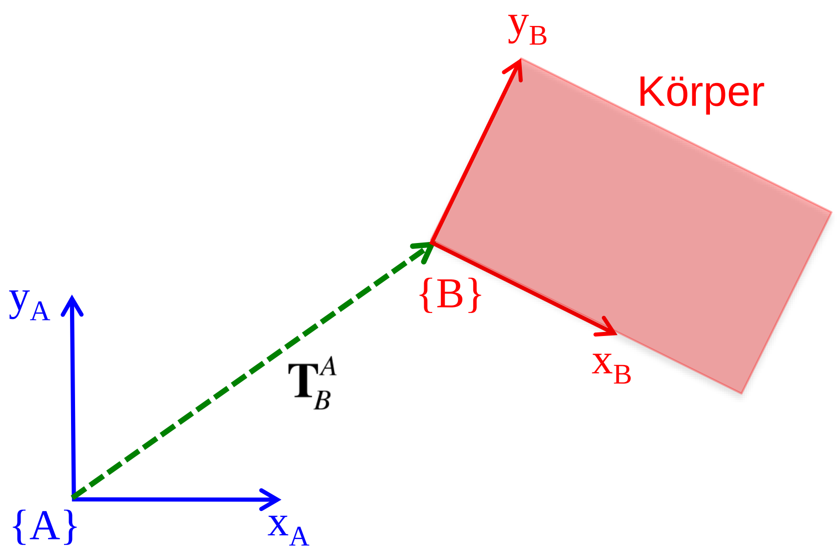
- Ein kartesisches KS besteht aus rechtwinkligen Achsen, die im 3D-Fall ein rechtshändiges KS bilden.



# Position und Orientierung von Körpern

- Körper werden mit einem körperfesten KS B versehen.
- Position und Orientierung (Pose) eines Körpers
  bzgl. eines Bezugs-KS A wird dann beschrieben
  durch eine Transformation, die das KS A in das KS B überführt.



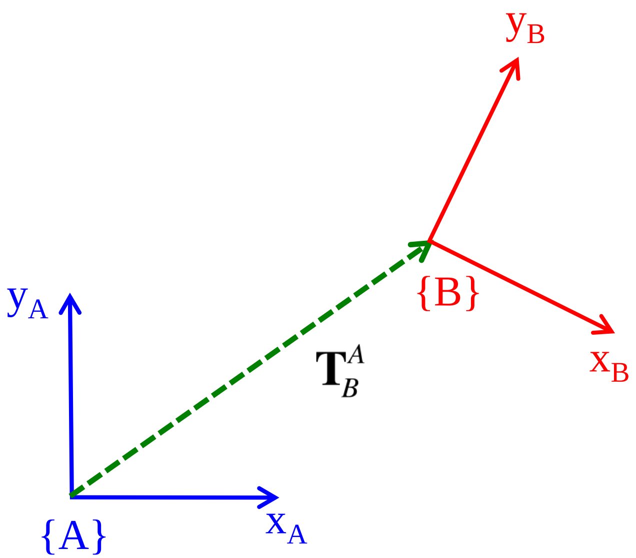

Die Transformation

$$\mathbf{T}_B^A$$

führt das KS A in KS B über.

# Transformationen

- Transformation $\mathbf{T}_B^A$ ist zusammengesetzt aus
  - Translation:
    wie ist Ursprung von B gegenüber Ursprung von A verschoben.
  - Rotation:
    wie ist B gegenüber A rotiert.




- Translation wird durch Vektor beschrieben.
- Verschiedene Darstellungen für Rotation:

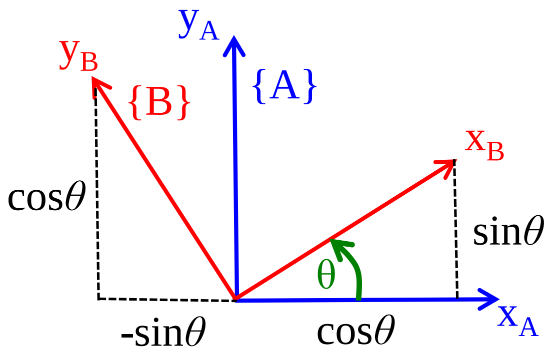
| 2D | 3D |
|---|---|
| 2D-Rotationsmatrix | 3D-Rotationsmatrix |
| Drehwinkel | 3 Eulerwinkel |
| | Drehachse und Drehwinkel |
| | Quaternionen |

- Rotationsmatrix und Translation lassen sich zu einer Transformationsmatrix zusammenfassen. Einfache Rechenregeln für Transformationsmatrizen.

# 2D-Rotationsmatrix

- Entsteht das 2D-KS B aus dem 2D-KS A durch eine Drehung um den Winkel θ (gegen den Uhrzeigersinn), so lässt sich die Transformation als eine 2D-Rotationsmatrix darstellen:

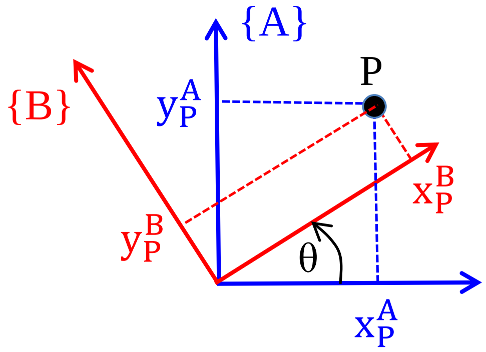
$$\mathbf{T}_B^A = \mathbf{R}(\theta) = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}$$




- Die Spaltenvektoren von $\mathbf{T}_B^A$ bilden das KS von B ausgedrückt bzgl. des KS A.

- Koordinatentransformation:

$$\mathbf{p}^{\mathrm{A}} = \mathbf{T}_B^A \mathbf{p}^{\mathrm{B}} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} \mathrm{x}_{\mathrm{P}}^{\mathrm{B}} \\ \mathrm{y}_{\mathrm{P}}^{\mathrm{B}} \end{pmatrix}$$



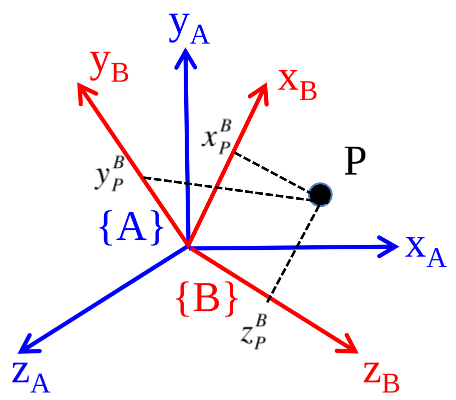

$\mathbf{p}^{\mathrm{A}}$: Darstellung des Punkts P im KS A

$\mathbf{p}^{\mathrm{B}}$: Darstellung des Punkts P im KS B

# 3D-Rotationsmatrizen

- Entsteht das 3D-KS B aus dem 3D-KS A durch eine beliebige Drehung, so lässt sich die Transformation als eine 3D-Rotationsmatrix darstellen:

$$\mathbf{T}_B^A = \mathbf{R} = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix}$$




- Die Spaltenvektoren von **R** bilden die Einheitsvektoren von B ausgedrückt bzgl. des KS A.

- Koordinatentransformation:

$$\mathbf{p}^A = \mathbf{T}_B^A \mathbf{p}^B = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix} \begin{pmatrix} x_P^B \\ y_P^B \\ z_P^B \end{pmatrix}$$

$\mathbf{p}^A$: Darstellung des Punkts P im KS A

$\mathbf{p}^B$: Darstellung des Punkts P im KS B

# Eigenschaften von Rotationsmatrizen

- Rotationsmatrizen R sind orthonormal; d.h. die Spaltenvektoren haben die Einheitsnorm und sind orthogonal zueinander:

$$\mathbf{R}^T\mathbf{R} = \mathbf{R}\mathbf{R}^T = \mathbf{I}$$

- Die Spaltenvektoren bilden ein rechtshändiges System:

$$\det(\mathbf{R}) = 1$$

- Die Menge der Rotationsmatrizen bilden eine (i.a. nicht-kommutative) Gruppe mit

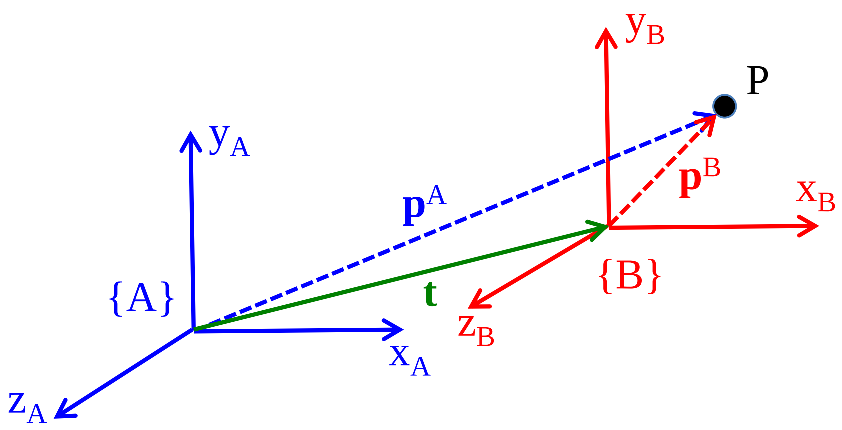
$$\mathbf{R}^{-1} = \mathbf{R}^T$$

- Wird auch SO(2) (2D) bzw. SO(3) (3D) genannt (special orthogonal group)

# Translation von Koordinatensysteme

- Entsteht das KS B aus dem KS A durch eine Translation (Verschiebung) um den Vektor **t**, dann gilt für die Koordinatentransformation:

$$\mathbf{p}^A = \mathbf{p}^B + \mathbf{t}$$



# Homogene 3D-Translationsmatrix

- Wir führen homogene Koordinaten ein:

$$\mathbf{p} = (x, y, z)^T \quad \longrightarrow \quad \tilde{\mathbf{p}} = (x, y, z, 1)^T$$

- Damit lässt sich die Translation als Matrix schreiben

$$\mathbf{p}^A = \mathbf{p}^B + \mathbf{t}$$

$$\begin{pmatrix} \mathbf{p}^A \\ 1 \end{pmatrix} = \begin{pmatrix} \mathbf{I}_{3*3} & \mathbf{t} \\ \mathbf{0}_{1*3} & 1 \end{pmatrix} \begin{pmatrix} \mathbf{p}^B \\ 1 \end{pmatrix}$$

$$\tilde{\mathbf{p}}^A = \begin{pmatrix} \mathbf{I}_{3*3} & \mathbf{t} \\ \mathbf{0}_{1*3} & 1 \end{pmatrix} \tilde{\mathbf{p}}^B$$

- Wir führen die Abk. $\mathbf{Tl(t)}$ für eine Translationsmatrix mit Translationsvektor $\mathbf{t}$ ein:

$$\mathbf{Tl}(\mathbf{t}) = \begin{pmatrix} \mathbf{I}_{3*3} & \mathbf{t} \\ \mathbf{0}_{1*3} & 1 \end{pmatrix}$$

# Homogene 3D-Rotationsmatrix

- Um auch Punkte mit homogenen Koordinaten rotieren zu können, erweitern wir Rotationsmatrizen $\mathbf{R}$ zu homogenen Rotationsmatrizen $\tilde{\mathbf{R}}$:

$$\tilde{\mathbf{R}} = \begin{pmatrix} \mathbf{R} & \mathbf{0}_{3*1} \\ \mathbf{0}_{1*3} & 1 \end{pmatrix}$$

- Koordinatentransformation mit homogenen Koordinaten:

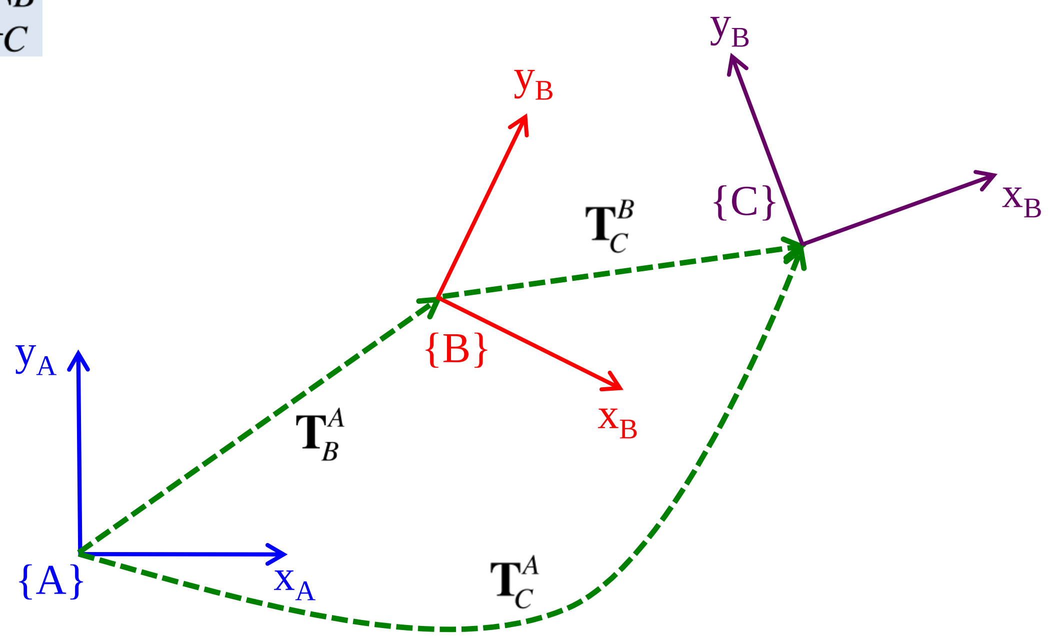
$$\begin{pmatrix} \mathbf{p}^A \\ 1 \end{pmatrix} = \begin{pmatrix} \mathbf{R} & \mathbf{0}_{3*1} \\ \mathbf{0}_{1*3} & 1 \end{pmatrix} \begin{pmatrix} \mathbf{p}^B \\ 1 \end{pmatrix} = \begin{pmatrix} \mathbf{R}\mathbf{p}^B \\ 1 \end{pmatrix}$$

$$\tilde{\mathbf{p}}^A = \tilde{\mathbf{R}} \tilde{\mathbf{p}}^B$$

# Komposition von Transformationen

- Eine Kette von Transformationen lässt sich mittels Matrixmultiplikation zu einer Transformation zusammenfassen

$$\mathbf{T}_C^A = \mathbf{T}_B^A \mathbf{T}_C^B$$



# Rotation und Translation in einer Transformationsmatrix (3D)

- Das KS B entsteht aus dem KS A durch eine Translation um den Vektor **t** und eine anschließende Rotation **R** des verschobenen KS.
- Wir führen zwei Einzelschritte ein:
  – KS C entsteht aus KS A durch Translation um **t**:

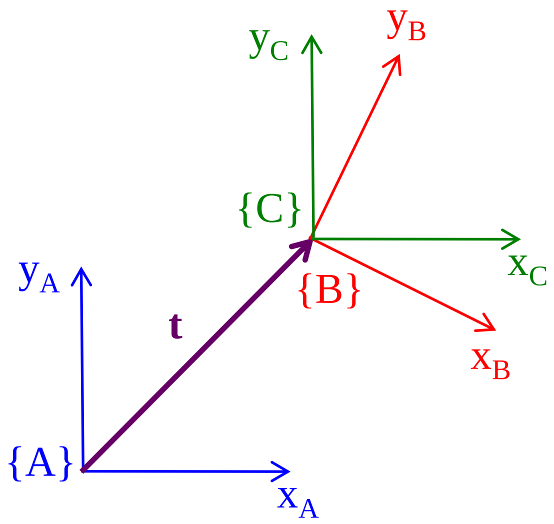
$$\mathbf{T}_C^A = \mathbf{Tl}(\mathbf{t})$$

  – KS B entsteht aus KS C durch Rotation **R**:

$$\mathbf{T}_B^C = \tilde{\mathbf{R}}$$

- Insgesamt

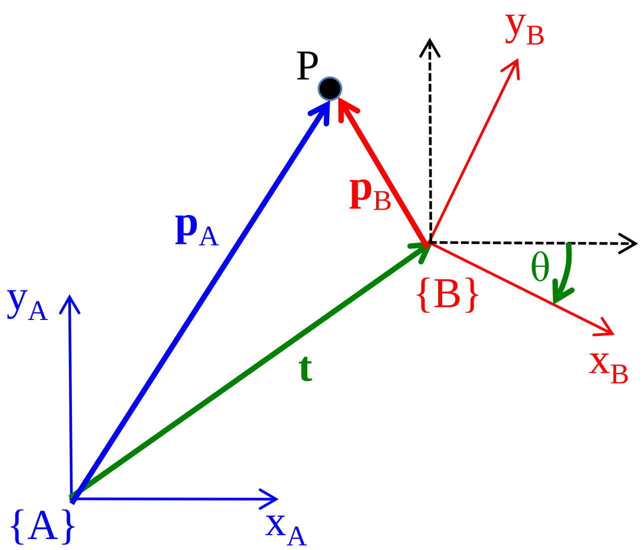
$$\begin{aligned}\mathbf{T}_B^A &= \mathbf{T}_C^A\,\mathbf{T}_B^C = \mathbf{Tl}(\mathbf{t})\,\tilde{\mathbf{R}} \\ &= \begin{pmatrix}\mathbf{I}_{3*3} & \mathbf{t} \\ \mathbf{0}_{1*3} & 1\end{pmatrix}\begin{pmatrix}\mathbf{R} & \mathbf{0}_{3*1} \\ \mathbf{0}_{1*3} & 1\end{pmatrix} \\ &= \begin{pmatrix}\mathbf{R} & \mathbf{t} \\ \mathbf{0}_{1*3} & 1\end{pmatrix}\end{aligned}$$




z-Achsen sind weggelassen.

# Rotation und Translation in einer Transformationsmatrix (2D)

- Das KS B geht aus dem KS A durch eine Translation um den Vektor **t** und eine Drehung um Winkel θ des verschobenen KS hervor.




- Homogene Transformationmatrix:

$$\mathbf{T}_B^A = \mathbf{Tl}(\mathbf{t})\,\widetilde{\mathbf{R}}(\theta)$$

$$= \begin{pmatrix} \mathbf{R}(\theta) & \mathbf{t} \\ \mathbf{0}_{1*2} & 1 \end{pmatrix}$$

# Homogene Transfomationsmatrizen

- Matrizen $\mathbf{T}$ der Bauart

$$\mathbf{T} = \begin{pmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}_{1*3} & 1 \end{pmatrix}$$

wobei $\mathbf{R}$ eine Rotationsmatrix und $\mathbf{t}$ ein Translationsvektor ist, werden auch homogene Transformationsmatrizen genannt.

- Sie bilden eine nicht-kommutative Gruppe die auch SE(2) (2D) bzw. SE(3) (3D) genannt wird. (SE = special Euclidian group)

- Inverse der Transformationsmatrix (hier 3D):

$$\mathbf{T}^{-1} = \begin{pmatrix} \mathbf{R}^T & -\mathbf{R}^t\mathbf{t} \\ \mathbf{0}_{1*3} & 1 \end{pmatrix}$$

$$\mathbf{T}\mathbf{T}^{-1} = \begin{pmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}_{1*3} & 1 \end{pmatrix} \begin{pmatrix} \mathbf{R}^T & -\mathbf{R}^T\mathbf{t} \\ \mathbf{0}_{1*3} & 1 \end{pmatrix}$$

$$= \begin{pmatrix} \mathbf{R}\mathbf{R}^T & -\mathbf{R}\mathbf{R}^T\mathbf{t} + \mathbf{t} \\ \mathbf{0}_{1*3} & 1 \end{pmatrix} = \begin{pmatrix} \mathbf{I}_{3*3} & \mathbf{0}_{3*1} \\ \mathbf{0}_{1*3} & 1 \end{pmatrix} = \mathbf{I}_{4*4}$$

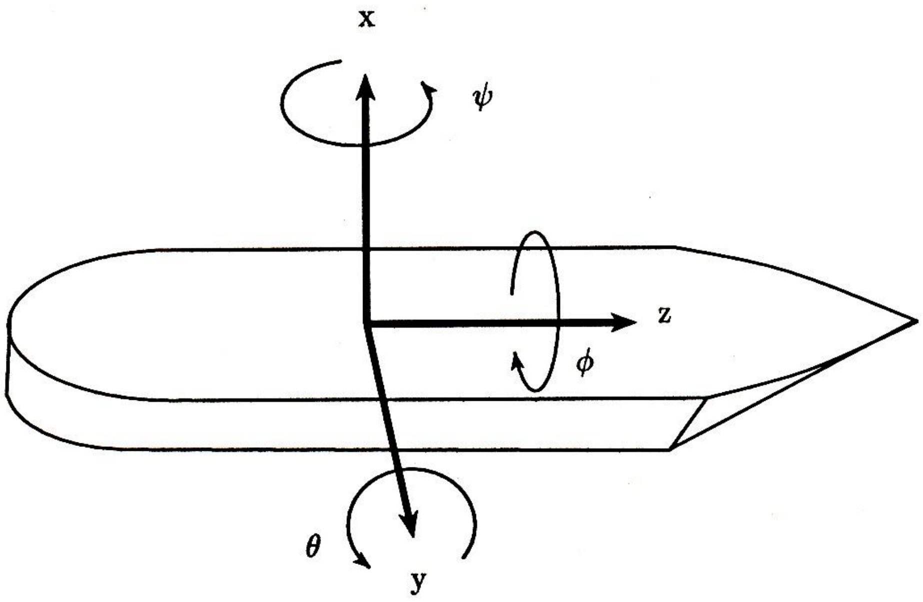
# Position und Orientierung

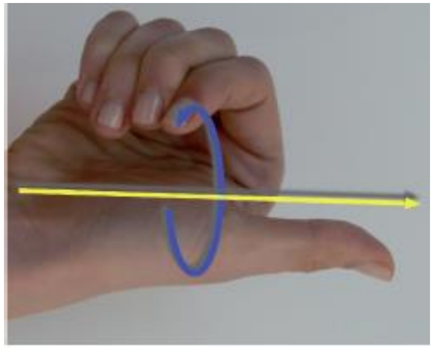
- Grundlagen
  Koordinatensysteme, Punkte und Körper,
  Position und Orientierung
- Transformationen
  Rotation, Translation, homogene Koordinaten,
  Transformationsmatrizen
- 3D-Rotationen und Eulerwinkel
- Bemerkungen
- Roboterkinematik am Beispiel Puma 560
- Weitere Darstellungsarten für Rotationen
  Drehvektoren und Quaternionen
- Geodätische Koordinatensysteme

# 3D-Rotation

- Euler-Theorem:

  Eine beliebige Drehung (in 3D) lässt sich mit drei Drehwinkeln (Eulerwinkel) um 3 ausgewählte Achsen beschreiben. Dabei müssen zwei aufeinanderfolgende Achsen unterschiedlich sein.
- Die Drehung um eine Achse wird auch elementare Rotation genannt.
- Die Drehung ist immer in Richtung der Drehachse im mathematisch positiven Sinn (Rechte-Hand-Regel)



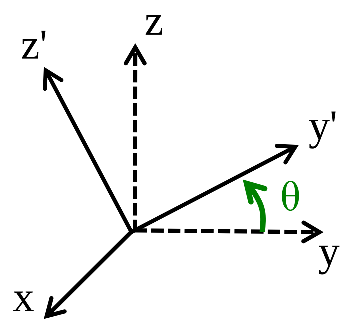



# Elementare Rotationsmatrizen

- Die elementaren Rotationsmatrizen ergeben sich wie bei den 2D-Rotationsmatrizen.

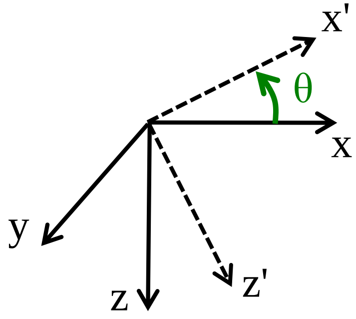
- Drehung um die x-Achse

$$\mathbf{R}(x,\theta) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{pmatrix}$$



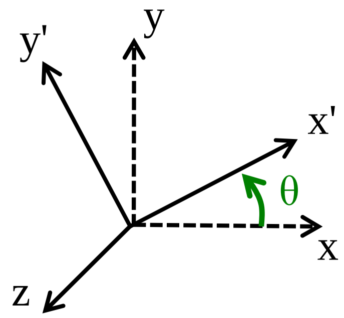

- Drehung um die y-Achse

$$\mathbf{R}(y,\theta) = \begin{pmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{pmatrix}$$




- Drehung um die z-Achse

$$\mathbf{R}(z,\theta) = \begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix}$$



# 12 Euler-Drehsysteme

- Notation:

  | | |
  |---|---|
  | Start-KS: | xyz |
  | KS nach 1. Rotation: | x'y'z' |
  | KS nach 2. Rotation: | x''y''z'' |

- 6 Drehsysteme, wobei um 3 unterschiedliche Achsen gedreht wird.

  Beispiel zy'x'':
  Drehe um z-Achse,
  drehe dann um gedrehte y-Achse y'
  und drehe dann um gedrehte x-Achse x''.

- Weitere 6 Drehsysteme, wobei 1. und 3. Drehachse identisch sind.

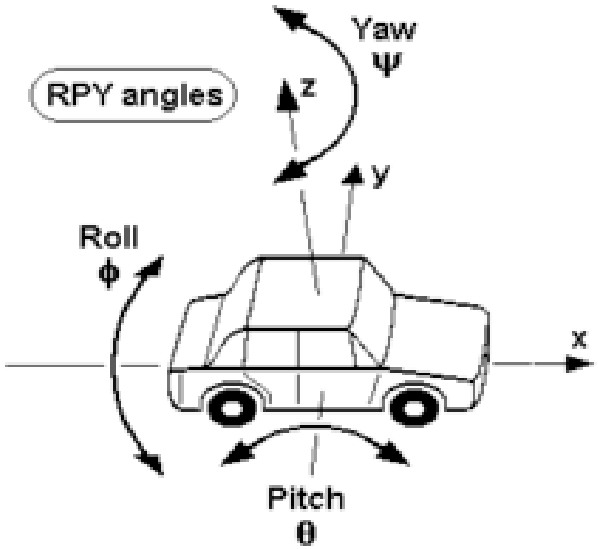
  Beispiel zx'z''

- Drehsysteme sind gleichwertig.
  Verwendung hängt vom Einsatzgebiet ab.

# Beispiel: zy'x''-Drehsystem yaw-pitch-roll (1)

- drehe um z-Achse um Winkel ψ (gieren, schwenken, engl. yaw)
  drehe um y-Achse um Winkel θ (neigen, nicken, engl. pitch)
  drehe um x-Achse um Winkel Φ (rollen, schlingern, engl. roll)

$$\underbrace{\mathbf{R}(zy'x'',\psi,\theta,\phi)}_{\mathbf{T}_B^A} = \underbrace{\mathbf{R}(z,\psi)}_{\mathbf{T}_{C'}^A}{}^*\underbrace{\mathbf{R}(y,\theta)}_{\mathbf{T}_{C''}^{C'}}{}^*\underbrace{\mathbf{R}(x,\phi)}_{\mathbf{T}_B^{C''}}$$



# Beispiel: zy'x''-Drehsystem yaw-pitch-roll (2)

- ausmultipliziert ergibt sich:

$$\mathbf{R}(zy'x'', \psi, \theta, \phi)$$

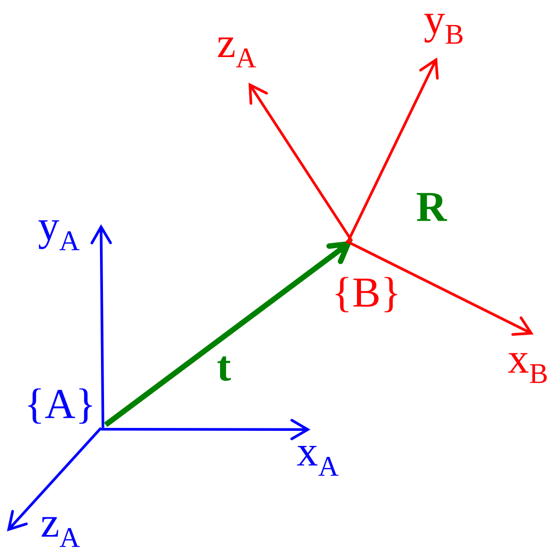
$$= \mathbf{R}(z, \psi)^{*}\mathbf{R}(y, \theta)^{*}\mathbf{R}(x, \phi)$$

$$= \begin{pmatrix} C\psi C\theta & C\psi S\theta S\varphi \; - \; S\psi C\varphi & C\psi S\theta C\varphi \; + \; S\psi S\varphi \\ S\psi C\theta & S\psi S\theta S\varphi \; + C\psi C\varphi & S\psi S\theta C\varphi \; - C\psi S\varphi \\ -S\theta & C\theta S\varphi & C\theta C\varphi \end{pmatrix}$$

mit C = cos und S = sin

# 3D-Transformation

- Das KS B geht aus KS A hervor durch eine Translation mit dem Vektor **t** und durch eine Drehung des verschobenen KS um 3 Euler-Winkel ψ, θ, Φ in einem gegebenem Drehsystem.




- Homogene Transformationmatrix für zy'x''-Drehsystem:

$$\mathbf{T}_B^A = \mathbf{T}(zy'x'', \psi, \theta, \phi, \mathbf{t})$$

$$= \begin{pmatrix} \mathbf{R}(zy'x'', \psi, \theta, \phi) & \mathbf{t} \\ \mathbf{0}_{1*3} & 1 \end{pmatrix}$$

# Position und Orientierung

- Grundlagen
  - Koordinatensysteme, Punkte und Körper,
  - Position und Orientierung
- Transformationen
  - Rotation, Translation, homogene Koordinaten,
  - Transformationsmatrizen
- 3D-Rotationen und Eulerwinkel
- Bemerkungen
- Roboterkinematik am Beispiel Puma 560
- Weitere Darstellungsarten für Rotationen
  - Drehvektoren und Quaternionen
- Geodätische Koordinatensysteme

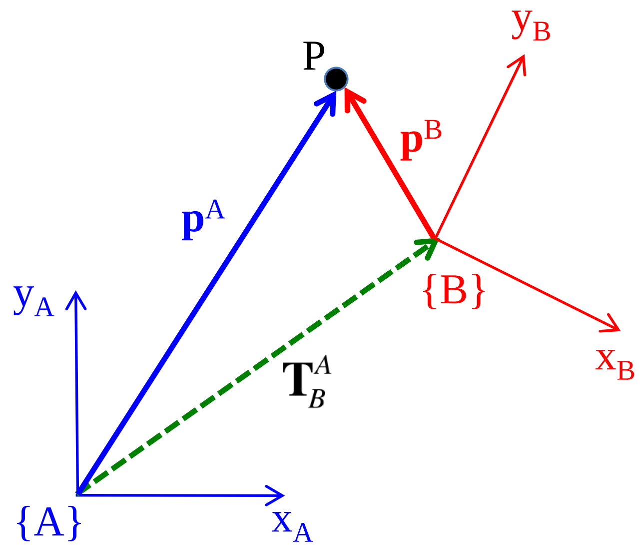
# KS-Bezüge beachten bei Koordinatentransformation

- Die Darstellung eines Punktes P in einem KS B
  lässt sich durch Multiplikation mit einer Transformationsmatrix $\mathbf{T}_B^A$
  in die Darstellung von P in KS A überführen
  (Koordinatentransformation):

$$\mathbf{p}^A = \mathbf{T}_B^A \mathbf{p}^B$$

$\mathbf{p}^{\mathrm{A}}$: Darstellung des Punktes P im KS A

$\mathbf{p}^{\mathrm{B}}$: Darstellung des Punktes P im KS B



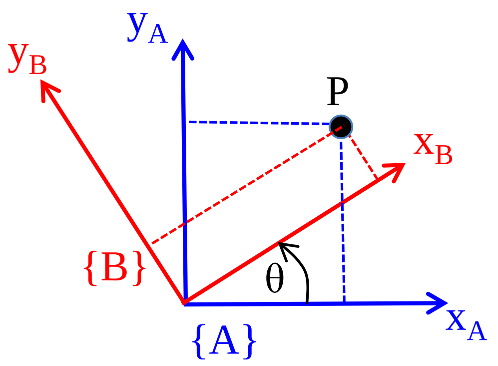

- Merkhilfe Indexlöschung: tiefgestellter Index von **T** und darauffolgender hochgestellter Index lassen sich (in Gedanken) streichen:

$$\mathbf{p}^A = \mathbf{T}_{\cancel{B}}^A \mathbf{p}^{\cancel{B}}$$

# Koordinatentransformation vs. Rotation

- Eine Rotationsmatrix kann sowohl zum Wechsel des KS als auch zur Rotation eines Punktes eingesetzt werden. Also Vorsicht!
- KS B sei gegenüber KS A um Winkel θ gedreht:

$$\mathbf{T}_B^A = \mathbf{R}(\theta) = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}$$

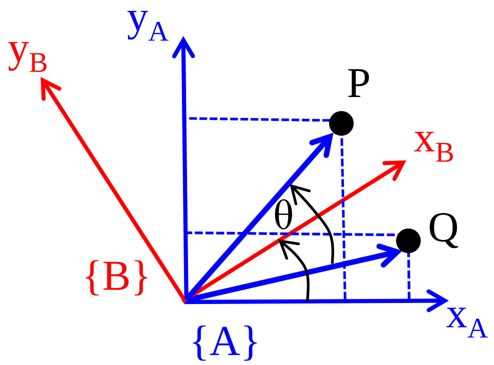



- Wechsel des KS von B nach A für einen Punkt P:

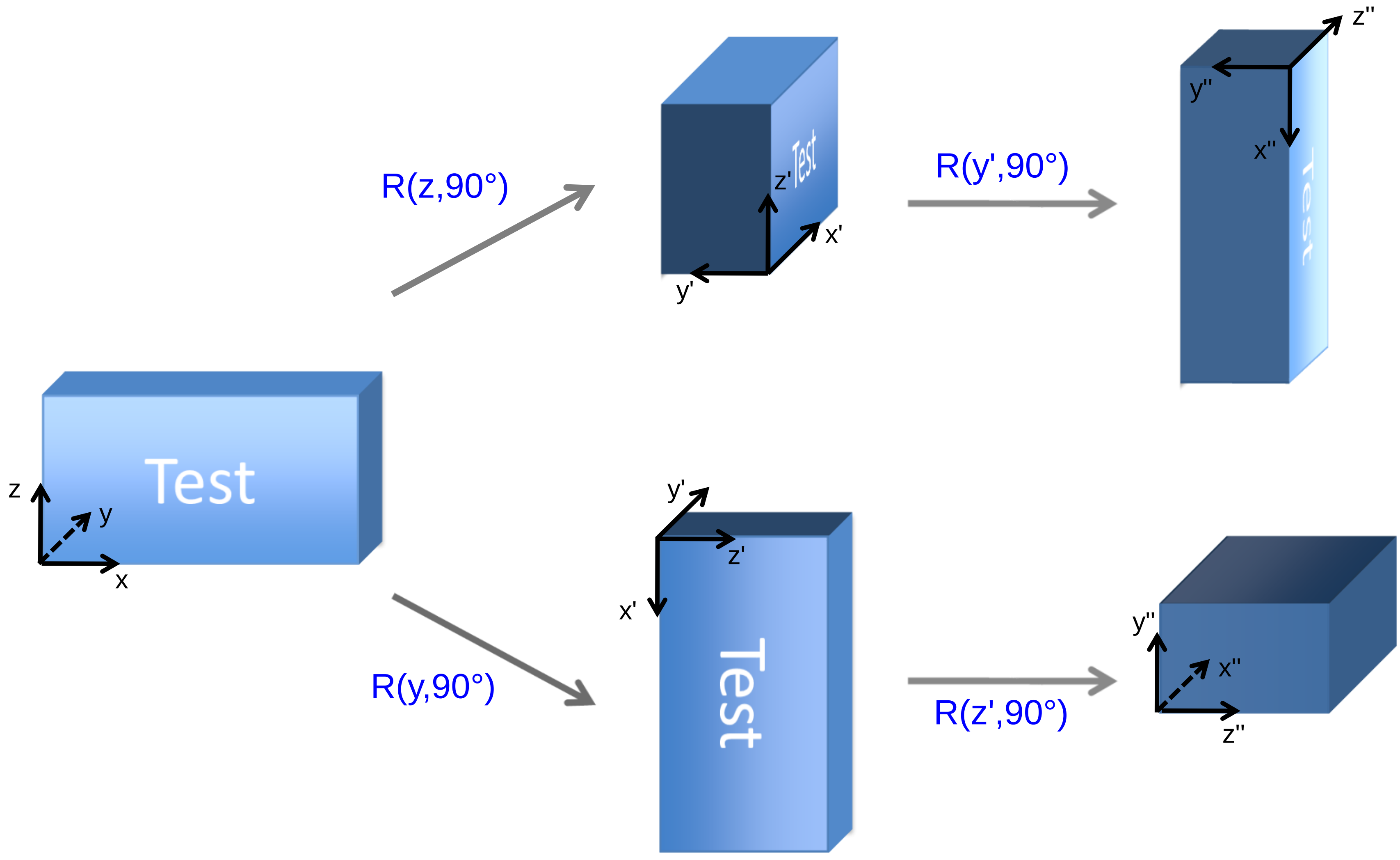
$$\mathbf{p}^A = \mathbf{T}_B^A \mathbf{p}^B$$

- Fasst man dagegen $\mathbf{p}^B$ als Vektor im KS A auf, dann erhält man ein Punkt Q mit $\mathbf{q}^A = \mathbf{p}^B$. Dann entsteht P durch Rotation des Punktes Q um den Winkel θ im KS A.

$$\begin{aligned} \mathbf{p}^A &= \mathbf{T}_B^A \mathbf{q}^A \\ &= \mathbf{T}_B^A \mathbf{p}^B \end{aligned}$$

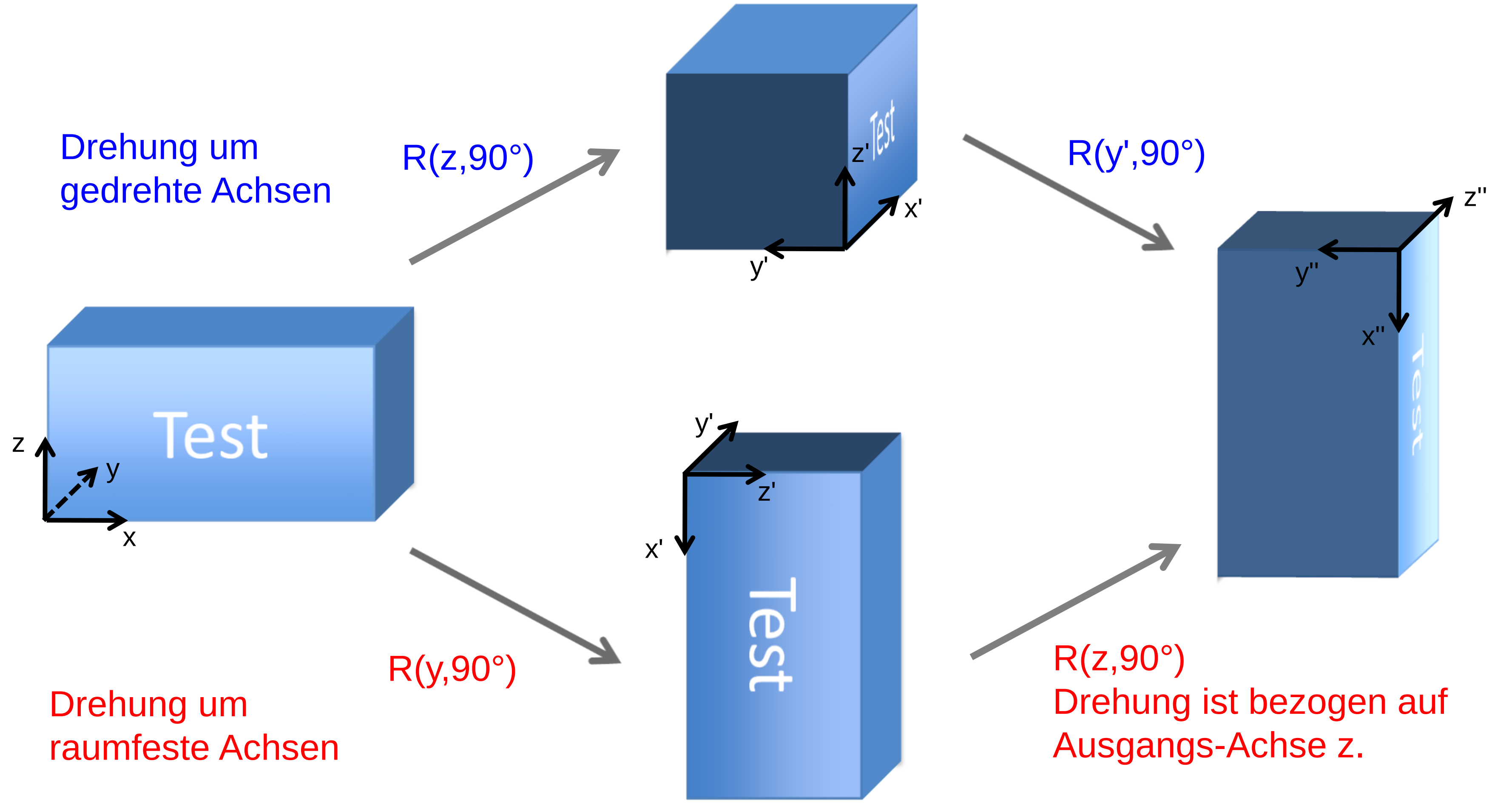


# Rotationsmatrizen sind i.a. nicht kommutativ



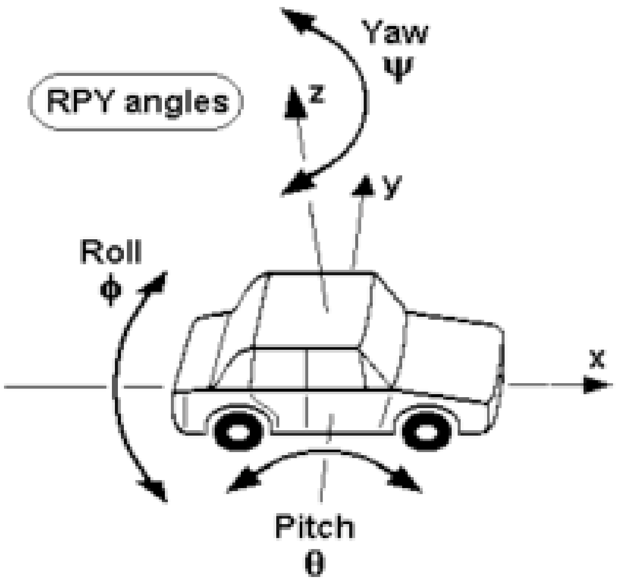
# Drehung um raumfeste Achsen

- Drehung in einem Euler-Drehystem (Drehungen um gedrehte Achsen) entspricht Drehung um raumfeste Achsen (Drehungen sind bezogen auf Ausgangs-KS) in umgekehrter Reihenfolge.



# Beispiel: xyz-Drehsystem yaw-pitch-roll

- xyz-Drehsystem mit raumfesten Achsen:
  - drehe um x-Achse um $\Phi$,
  - dann um y-Achse um $\theta$ und
  - dann um z-Achse um $\psi$
- xyz-Drehsystem entspricht zy'x''-Drehsystem.




$$\mathbf{R}(xyz,\phi,\theta,\psi)$$
$$= \mathbf{R}(z,\psi)^{*}\mathbf{R}(y,\theta)^{*}\mathbf{R}(x,\phi)$$
$$= \mathbf{R}(zy'x'',\psi,\theta,\phi)$$

# Position und Orientierung

- Grundlagen
  - Koordinatensysteme, Punkte und Körper, Position und Orientierung
- Transformationen
  - Rotation, Translation, homogene Koordinaten, Transformationsmatrizen
- 3D-Rotationen und Eulerwinkel
- Bemerkungen
- Roboterkinematik am Beispiel Puma 560
- Weitere Darstellungsarten für Rotationen
  - Drehvektoren und Quaternionen
- Geodätische Koordinatensysteme

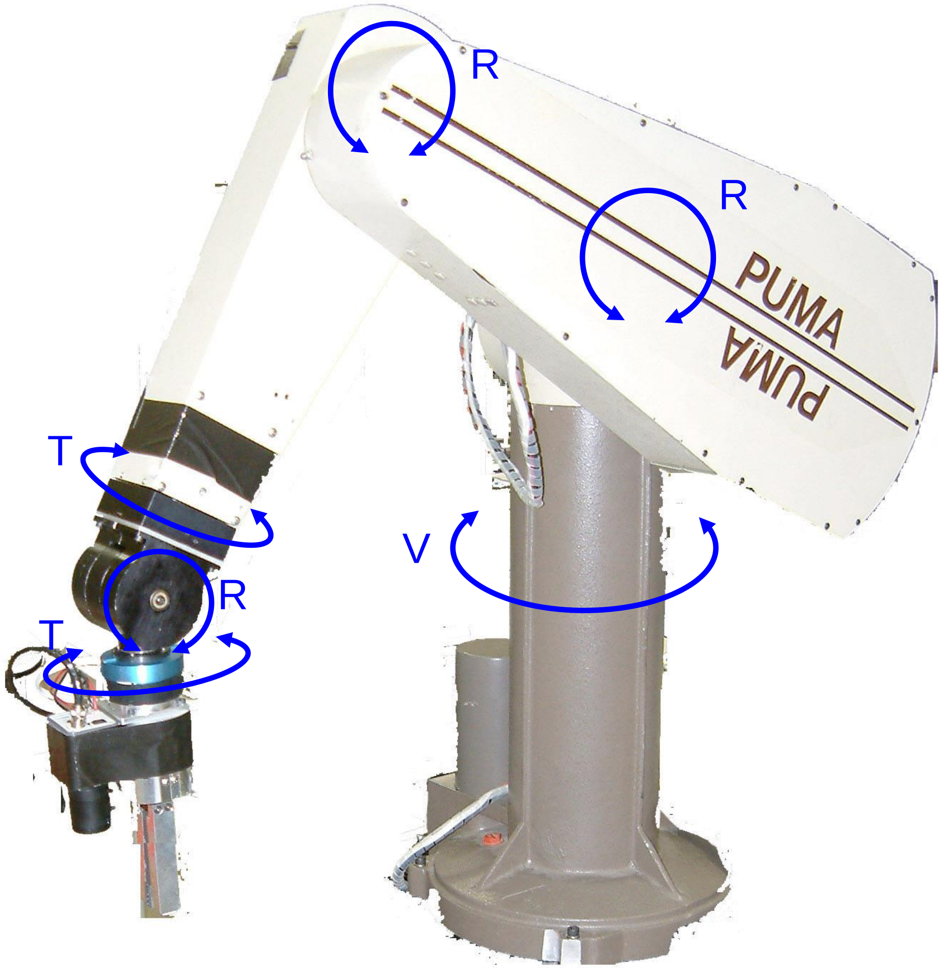
# Puma 560

- PUMA = programmable universal machine for assembly
- hat eine typische Struktur für ein 6 DOF-Gelenkarm-Roboter
- 6 Drehgelenke:

$$\underbrace{\text{VRR}}_{\text{Arm}} – \underbrace{\text{TRT}}_{\text{Handgelenk}}$$

V = Revolvergelenk
R = Rotationsgegelenk
T = Torsionsgelenk

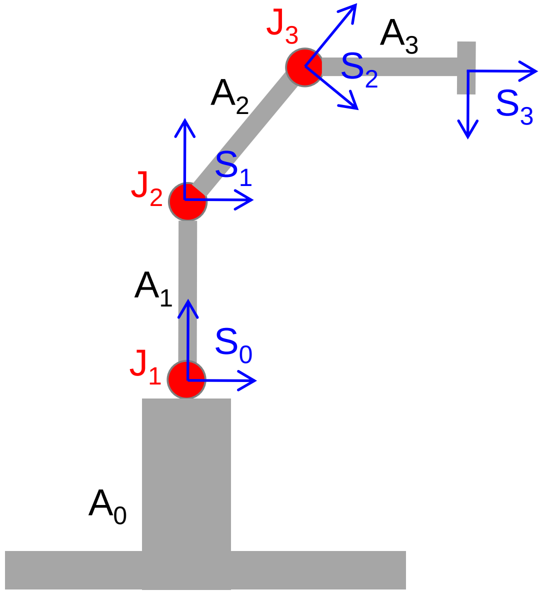
- Um ein Objekt zu greifen, wird mit dem Arm die Position und mit dem Handgelenk die Orientierung des Objekts eingestellt (6 DOF)




gruju.blogspot.com; 2016

# Denavit-Hartenberg (DH) Konvention (1)

- KS'e werden meist nach der DH-Konvention festgelegt.
- Gelenke (joints) und Arme (links) sind nummeriert: $J_1$, $J_2$, ..., $J_n$ und $A_0$, $A_1$, ..., $A_n$.
- Gelenk $J_i$ verbindet Arm $A_{i-1}$ und $A_i$.
- Das KS $S_i$ ist dem Arm $A_i$ fest zugeordnet und liegt (für $i \leq n-1$) im Drehgelenk $J_{i+1}$
- KS $S_0$ ist das ortsfeste Ausgangs-KS.
- Das letzte KS $S_n$ ist meist auch gleichzeitig das Endeffektor-KS mit den Achsen nsa:
  - a = approach vector
  - s = sliding vector
  - n = normal vector

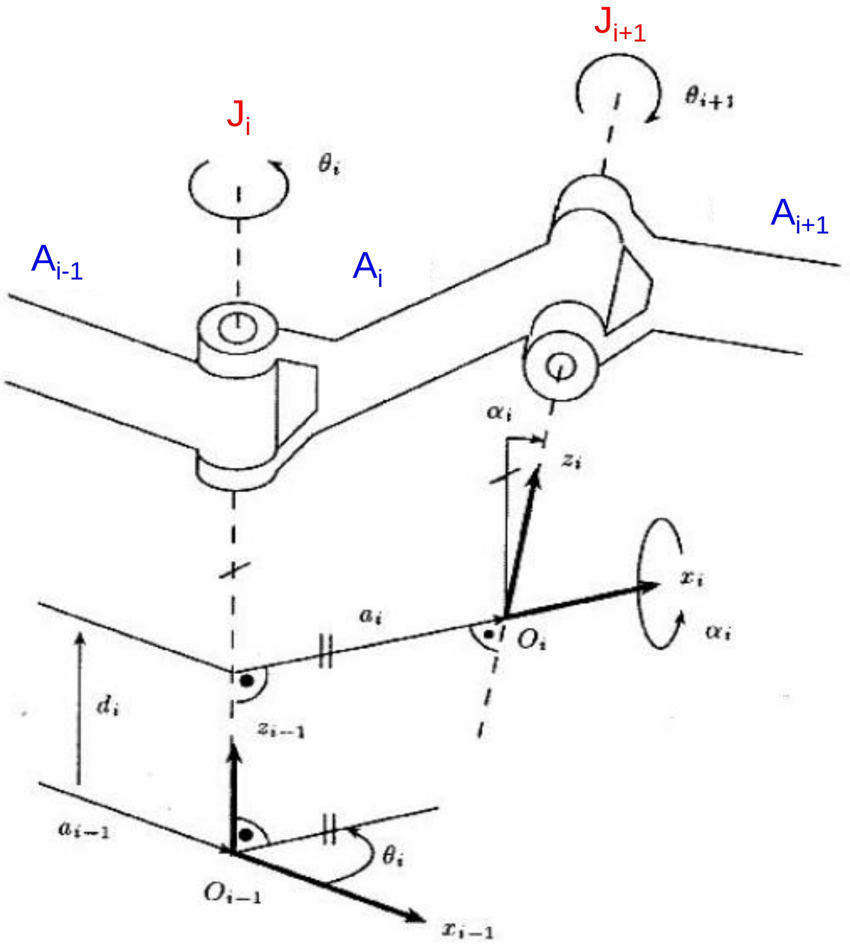



Roboterarm mit 3 Gelenken

# Denavit-Hartenberg-Konvention (2)

Das KS $S_i$ ist am Arm $A_i$ fixiert, wobei:

- Die $z_i$-Achse wird entlang der Bewegungsachse des Gelenks $J_{i+1}$ gelegt.
- Die $x_i$-Achse ist normal zur $z_{i-1}$-Achse und zeigt von ihr weg.
- Die $y_i$-Achse wird so festgelegt, dass KS $S_i$ ein rechtshändiges KS ist.



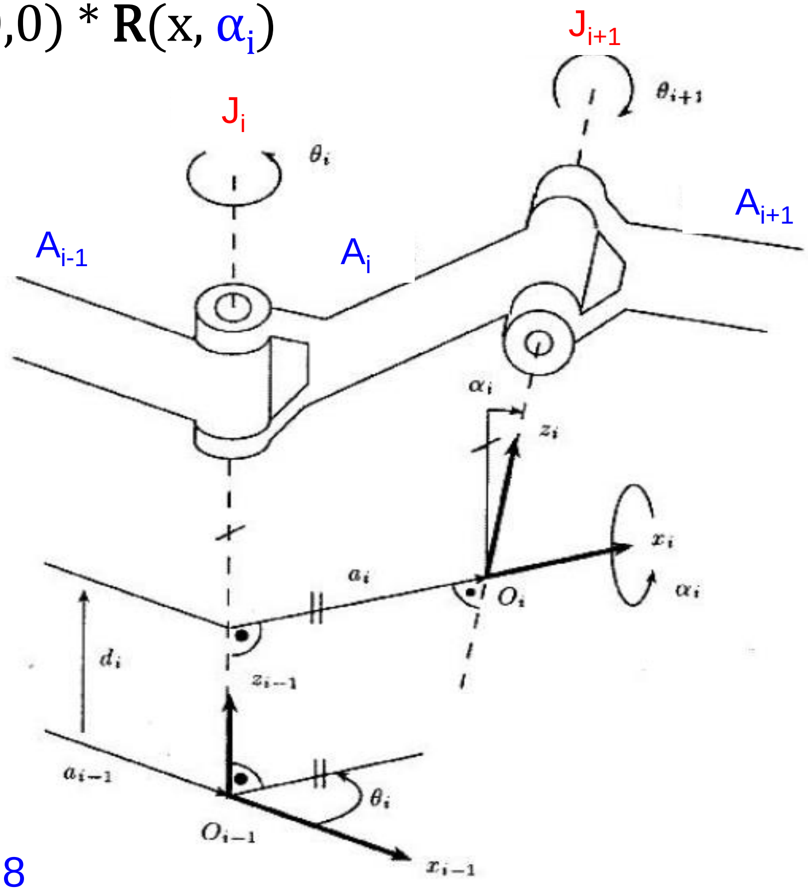

[aus Siegert, Bocionek, *Robotik: Programmierung intelligenter Roboter*, Springer 1996]

# Denavit-Hartenberg-Konvention (3)

- Die Lage des KS $S_i$ gegenüber $S_{i-1}$ wird beschrieben durch 4 DH-Parameter $\theta_i$, $\alpha_i$, $a_i$ und $d_i$ und folgender Transformation:

$$\mathbf{T}_i^{i-1} = \mathbf{Tl}(0,0,d_i) * \mathbf{R}(z, \theta_i) * \mathbf{Tl}(a_i,0,0) * \mathbf{R}(x, \alpha_i)$$

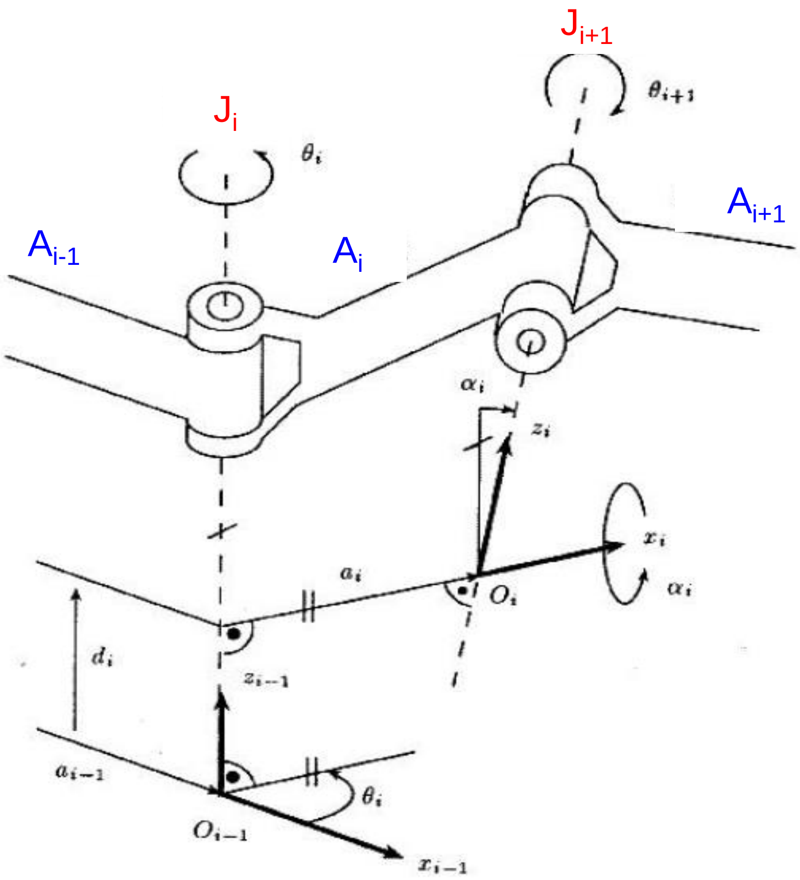
  - verschiebe entlang der z-Achse um $d_i$
  - drehe um z-Achse um $\theta_i$
  - verschiebe entlang der neuen x-Achse um $a_i$
  - Drehung um die neue x-Achse um $\alpha_i$



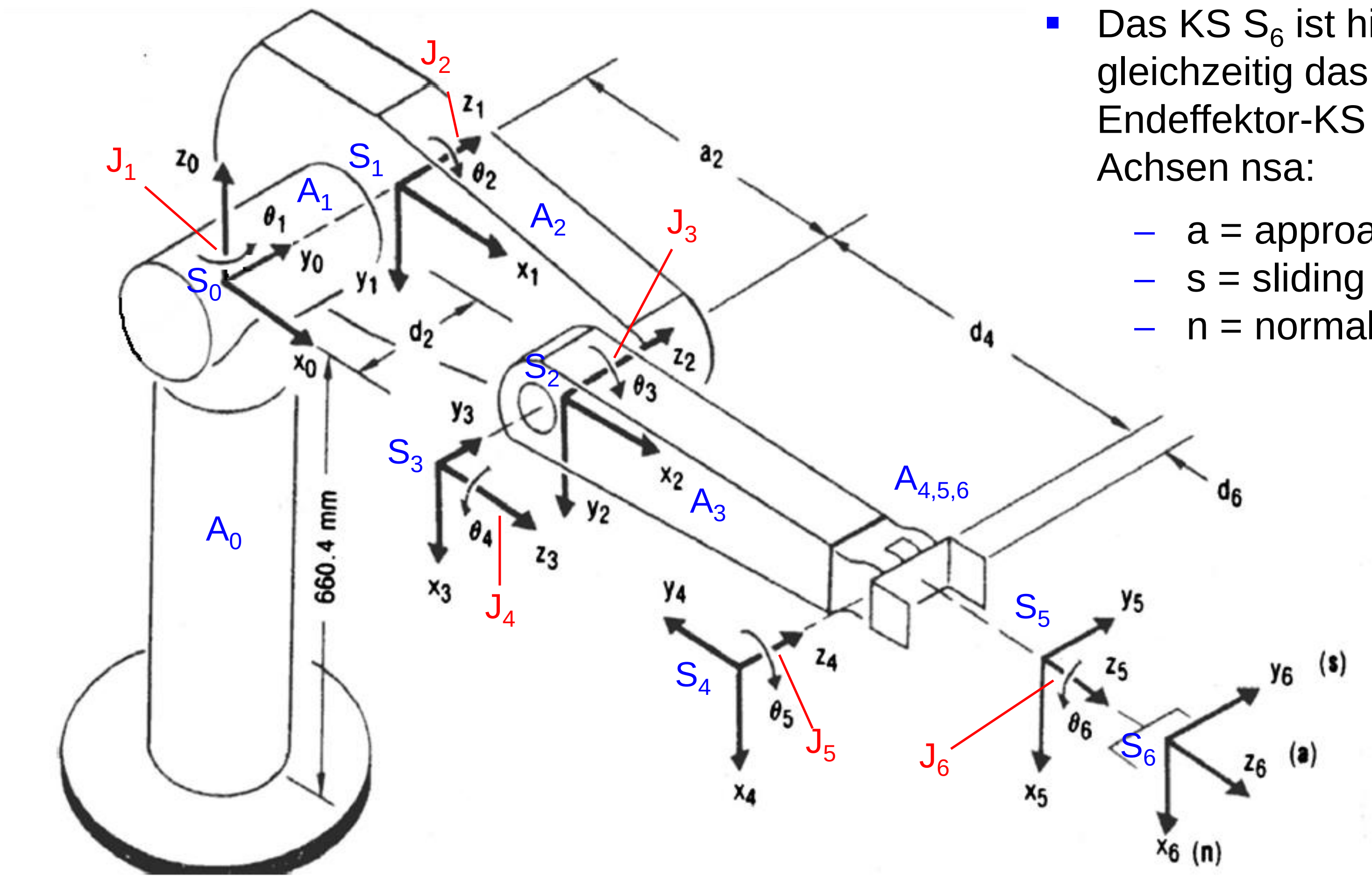

Animation der DH-Parameter:
https://www.youtube.com/watch?v=rA9tm0gTln8

# Denavit-Hartenberg-Konvention (4)

- $\theta_i$ ist der am Gelenk $J_i$ eingestellte Drehwinkel und ist variabel.
- $\alpha_i$ ist der konstante Neigungswinkel zwischen den beiden Drehachsen.
- Die DH-Konvention legt das KS $S_i$ für $1 \leq i \leq n-1$ eindeutig fest, sofern die Achsen $z_i$ und $z_{i-1}$ windschief zueinander sind.
- Die Lage der x,y-Achsen von $S_0$ können frei gewählt werden; beachte dass die $z_0$-Achse entlang der Bewegungsachse des Gelenks $J_1$ gelegt werden muss.
- Das KS $S_n$ kann frei gewählt werden, sofern die $x_n$-Achse senkrecht zur $z_{n-1}$-Achse liegt.



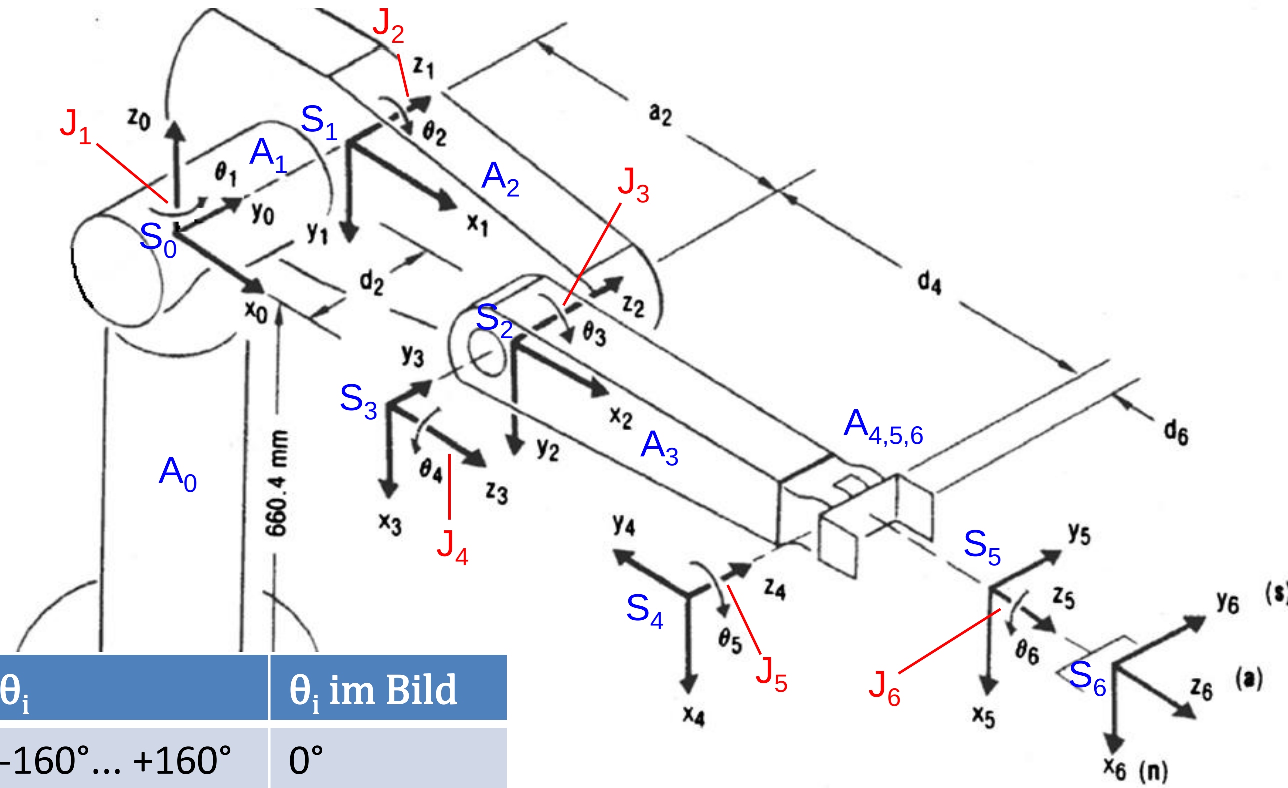
# Koordinatensysteme nach DH-Konvention für Puma 560




[leicht verändert aus Fahmy und Gahny, Neuro-fuzzy inverse model control structure of robotic manipulators utilized for physiotherapy applications, 2013.]

- Das KS $S_6$ ist hier auch gleichzeitig das Endeffektor-KS mit den Achsen nsa:
  - a = approach vector
  - s = sliding vector
  - n = normal vector

# DH-Parameter für Puma 560

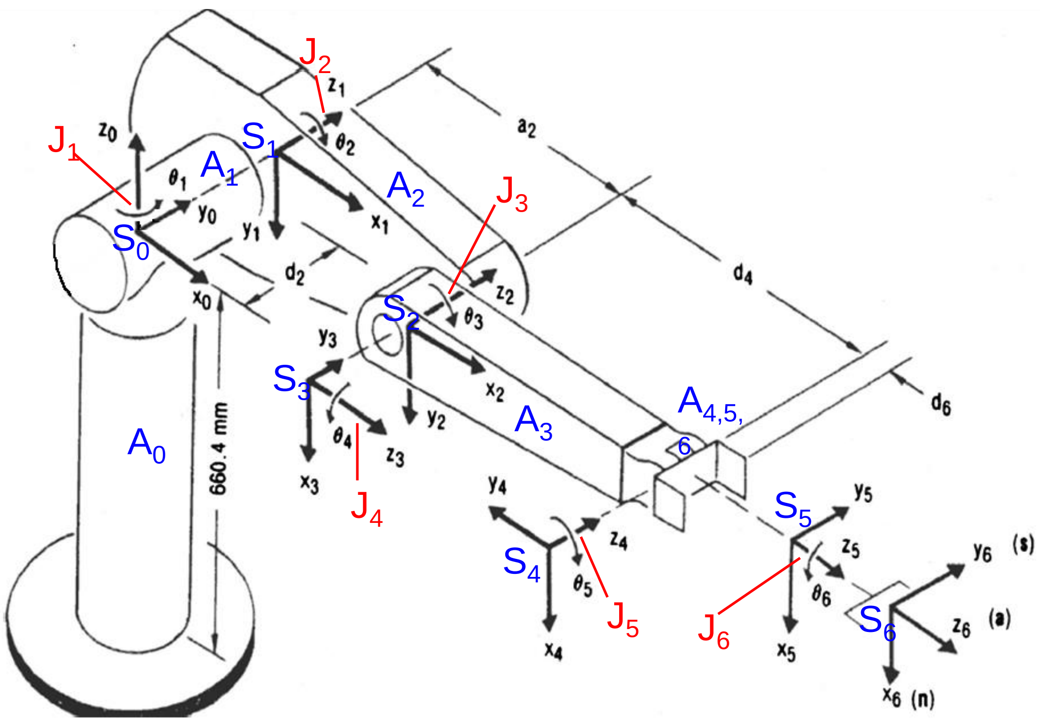



| i | $\alpha_i$ | $a_i$ [m] | $d_i$ [m] | $\theta_i$ | $\theta_i$ im Bild |
|---|---|---|---|---|---|
| 1 | -90° | 0 | 0 | -160°... +160° | 0° |
| 2 | 0° | 0.43 | 0.15 | -225°... +45° | 0° |
| 3 | 90° | 0 | 0 | -45°... +225° | 90° |
| 4 | -90° | 0 | 0.43 | -110°... +170° | 0° |
| 5 | 90° | 0 | 0 | -100°... +100° | 0° |
| 6 | 0° | 0 | 0.06 | -266°... +266° | 0° |

$$\mathbf{T}_i^{i-1} = \mathbf{Tl}(0,0,\mathrm{d_i}) * \mathbf{R}(\mathrm{z}, \theta_\mathrm{i}) * \mathbf{Tl}(\mathrm{a_i},0,0) * \mathbf{R}(\mathrm{x}, \alpha_\mathrm{i})$$

# Vorwärts- und Rückwärtskinematik

# Vorwärtskinematik Puma 560




- Die Pose von KS $S_6$ bzgl. $S_0$ lässt sich beschreiben durch eine Verkettung von 6 Transformationen mit den Gelenkwinkeln $\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6$:

$$\mathbf{T}_6^0 = \mathbf{T}_1^0\,\mathbf{T}_2^1\mathbf{T}_3^2\mathbf{T}_4^3\mathbf{T}_5^4\mathbf{T}_6^5 \quad \text{mit}$$

$$\mathbf{T}_i^{i-1} = \mathbf{Tl}(0,0,\mathrm{d_i}) * \mathbf{R}(\mathrm{z}, \theta_\mathrm{i}) * \mathbf{Tl}(\mathrm{a_i},0,0) * \mathbf{R}(\mathrm{x}, \alpha_\mathrm{i})$$

# Rückwärtskinematik Puma 560 (1)

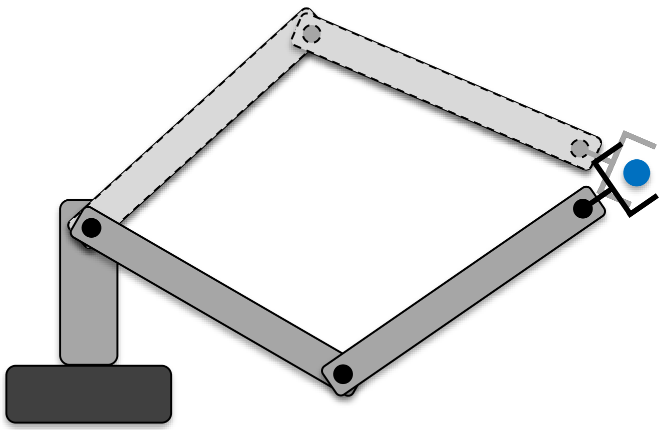
- Die Pose $\mathbf{T}$ des Endeffektor-KS $S_6$ bzgl. des ortsfesten Ausgangs-KS $S_0$ lässt sich beschreiben durch eine Verkettung von 6 Transformationen mit den Gelenkwinkeln $\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6$:

$$
\begin{aligned}
\mathbf{T} &= \mathbf{T}_6^0 \\
&= \mathbf{T}_1^0 \, \mathbf{T}_2^1 \mathbf{T}_3^2 \mathbf{T}_4^3 \mathbf{T}_5^4 \mathbf{T}_6^5 \\
&= f(\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6)
\end{aligned}
$$

- Bei der Rückwärtskinematik ist die Pose $\mathbf{T}$ gegeben und die Gelenkwinkel $\theta_1$, $\theta_2$, $\theta_3$, $\theta_4$, $\theta_5$, $\theta_6$ müssen bestimmt werden.
- Gesucht ist also die Inverse von f.

# Rückwärtskinematik Puma 560 (2)

- Probleme:
  - f ist i.a. nicht-linear.
  - Inverse von f ist eventuell nicht explizit darstellbar.
  - Lösung ist nicht eindeutig.
  - Es existiert keine Lösung.



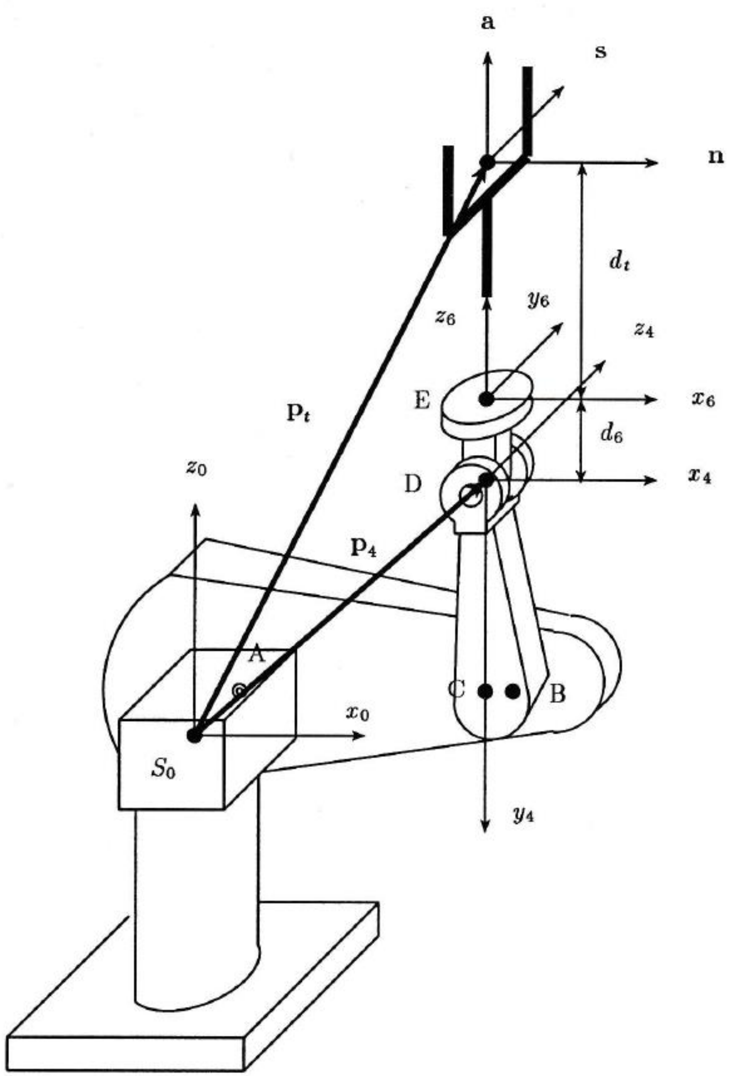
- Ansätze:
  - Explizite Lösung durch algebraische Umformungen.
  - Explizite Lösung durch geometrische Überlegungen.
  - Numerische Lösung.
    Z.B. Bestimmung der Nullstellen von

$$f(\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6) - \mathbf{T} = 0$$

    mit einem Newton-Verfahren.

# Rückwärtskinematik Puma 560 (3)

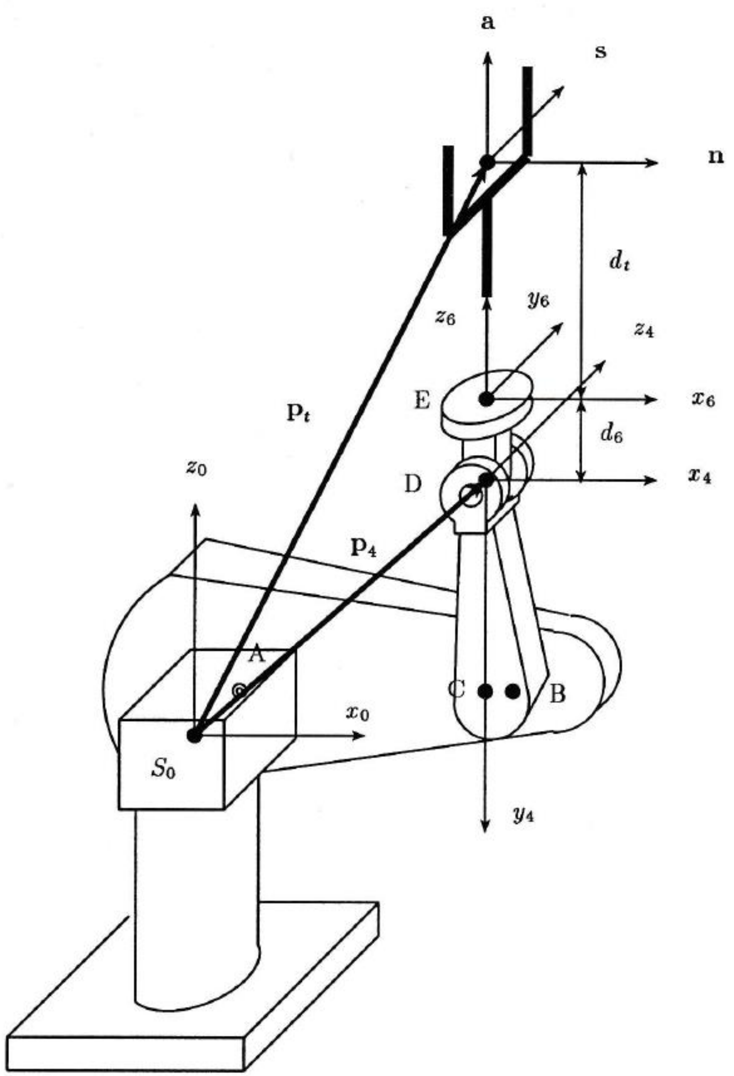
- Die Gelenkachsen des Puma 560 liegen entweder rechtwinklig oder parallel zueinander.
- Die drei Handgelenksachsen schneiden sich in einem Punkt. Daher ändern die Handgelenksachsen nur die Orientierung des Endeffektors (keine Translation). Wird auch Kugelgelenk (spherical wrist) genannt.
- Dadurch wird die Rückwärtsrechnung vereinfacht und eine Lösung durch geometrische Überlegung möglich.




[aus Siegert, Bocionek, *Robotik: Programmierung intelligenter Roboter*, Springer 1996]

# Rückwärtskinematik Puma 560 (4)

- Vorgehensweise (Skizze):
  1. Gegeben ist die Wunsch-Pose $\mathbf{T}$ des Endeffektor-KS durch den Punkt $\mathbf{p_t}$ des KS-Ursprungs und der KS-Achsen **n**, **s**, **a**.
  2. Daraus lässt sich der Urspung $\mathbf{p_4}$ von KS $S_4$ einfach bestimmen.
  3. Aus dem Ursprung $\mathbf{p_4}$ von KS $S_4$ lassen sich die ersten drei Gelenkwinkel $\theta_1$, $\theta_2$, $\theta_3$ durch Rückwärtsrechnung vergleichsweise einfach bestimmen (3-Arm-Roboterproblem).
  4. Bestimme dann die letzten drei Gelenkwinkel $\theta_4$, $\theta_5$, $\theta_6$ durch algebraische Rückwärtsrechnung so, dass das Endeffektor-KS die gewünschte Orientierung **n**, **s**, **a** hat.




[aus Siegert, Bocionek, *Robotik: Programmierung intelligenter Roboter*, Springer 1996]

# Position und Orientierung

- Grundlagen
  Koordinatensysteme, Punkte und Körper,
  Position und Orientierung
- Transformationen
  Rotation, Translation, homogene Koordinaten,
  Transformationsmatrizen
- 3D-Rotationen und Eulerwinkel
- Bemerkungen
- Roboterkinematik am Beispiel Puma 560
- Weitere Darstellungsarten für Rotationen
  Drehvektoren und Quaternionen
- Geodätische Koordinatensysteme

# Drehung um eine Drehachse

- Eine beliebige Drehung (in 3D) lässt sich auch als Drehung um einen Winkel $\theta$ um genau eine Achse – die Drehachse – beschreiben.
- Die Drehachse lässt sich durch einen Vektor $\mathbf{v}$ beschreiben, der in Richtung der Drehachse zeigt.
- Rotationsmatrizen $\mathbf{R}$ haben immer den reellen Eigenwert $\lambda_1 = 1$ und die beiden komplexen Eigenwerte $\lambda_{2,3} = \cos\theta \pm \mathrm{i}^* \sin\theta$.
- Der zum Eigenwert $\lambda_1 = 1$ gehörende Eigenvektor $\mathbf{v}_1$ beschreibt dann die Drehachse:

$$\mathbf{R}\mathbf{v}_1 = \mathbf{v}_1$$

- Der Drehwinkel $\theta$ ergibt sich aus den komplexen Eigenwerten

$$\lambda_{2,3} = \cos\theta \pm \mathrm{i}^* \sin\theta.$$

# Quaternionen (1)

- Quaternionen sind eine Verallgemeinerung von komplexen Zahlen und haben die Form:

$$\begin{aligned}\mathbf{q} &= s + i * v_1 + j * v_2 + k * v_3 \\ &= s + (v_1, v_2, v_3) \\ &= s + \mathbf{v}\end{aligned}$$

  Dabei sind s, $v_1$, $v_2$, $v_3$ reelle Zahlen und für i, j, k gilt:

$$i^2 = j^2 = k^2 = ijk = -1$$

- $\mathbf{q} = s + \mathbf{v} = s + (v_1, v_2, v_3)$ ist ein Einheitsquaternion, falls:

$$s^2 + v_1^2 + v_2^2 + v_3^2 = 1$$

# Quaternionen (2)

- Ein Einheitsquaternion $\mathrm{q} = \mathrm{s} + \mathbf{v}$ kann als eine Rotation mit Winkel $\theta$ um die Drehachse $\mathbf{n}$ verstanden werden, wobei folgender Zusammenhang besteht:

$$s = \cos\frac{\theta}{2} \quad \text{und} \quad \mathbf{v} = \mathbf{n} * \sin\frac{\theta}{2} \quad \text{mit } \|\mathbf{n}\| = 1$$

- Quaternionen-Operationen sind im Vergleich zu Operationen mit Rotationsmatrizen rechentechnisch einfacher und numerisch stabiler.

| Operation | Rotationsmatrix $\mathbf{R}$ | Einheitsquaternion $\mathrm{q} = \mathrm{s} + \mathbf{v}$ |
|---|---|---|
| Inverse | $\mathbf{R}^{-1} = \mathbf{R}^{\mathrm{T}}$ | $\mathrm{q}^{-1} = \mathrm{s} - \mathbf{v}$ |
| KS-Wechsel | $\mathbf{x}' = \mathbf{R}\mathbf{x}$ | $0 + \mathbf{x}' = \mathrm{q}\ (0+\mathbf{x})\ \mathrm{q}^{-1}$ |
| Komposition | $\mathbf{R}_1\mathbf{R}_2$ | $\mathrm{q}_1\mathrm{q}_2$ |

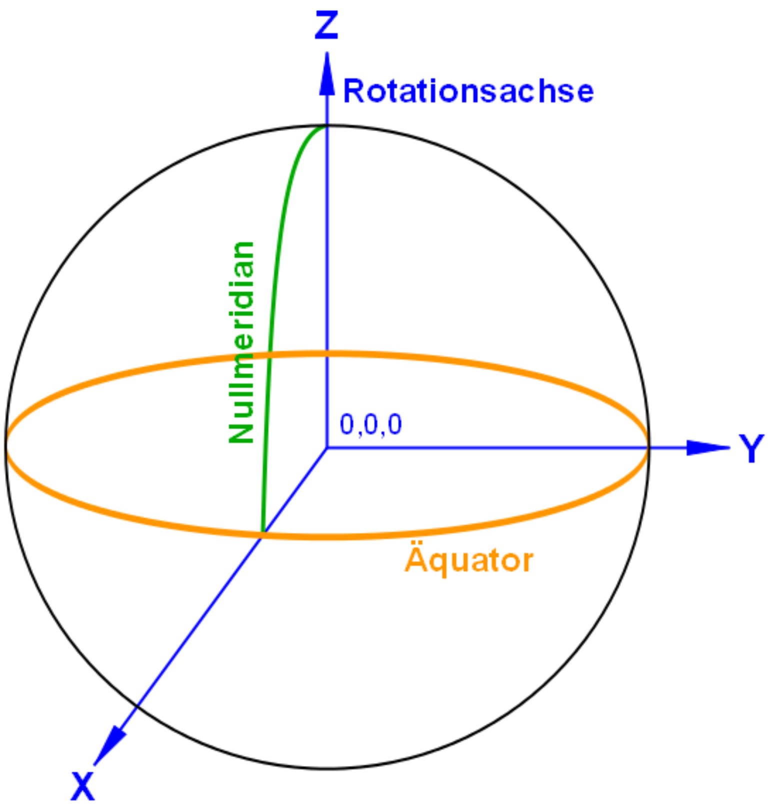
Introduction to Quaternions and 3D Rotation: https://eater.net/quaternions

# Geodätische Koordinatensysteme

- Für die Navigation im Outdoorbereich sind verschiedene geodätische Koordinatensysteme wichtig.

# ECEF

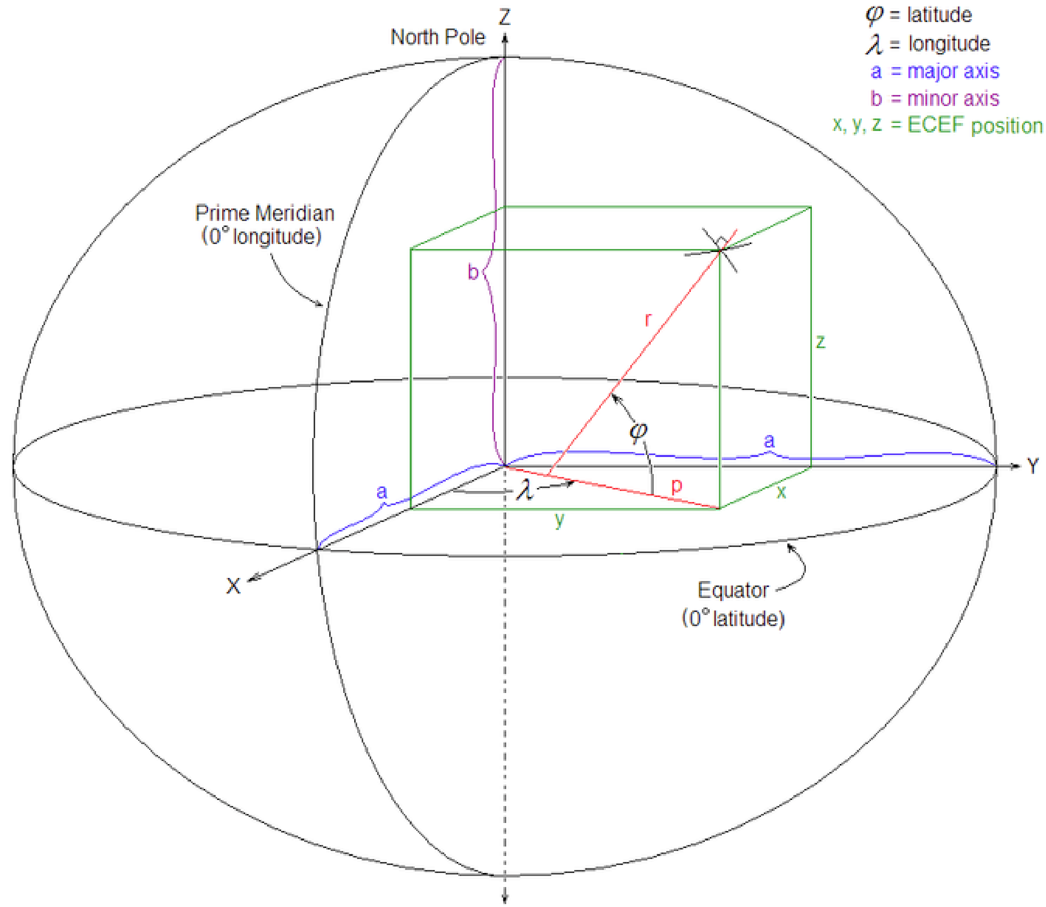
- ECEF (Earth-Centered Earth-Fixed) ist ein kartesisches KS, das im Mittelpunkt der Erde fixiert ist (also mit der Erde rotiert)
- Die z-Achse zeigt in Richtung Norden.
  Die x-Achse schneidet die Erde im Äquator beim Längengrad 0.
- Positionsbestimmung im GPS-System wird im ECEF-KS vorgenommen:
  aus den bekannten ECEF-Positionen von wenigstens 4 Satelliten und Entfernungsmessungen zu den Satelliten wird die unbekannte ECEF-Position eines GPS-Empfängers berechnet.




https://de.wikipedia.org/wiki/Geozentrisches_Koordinatensystem

# WGS 84

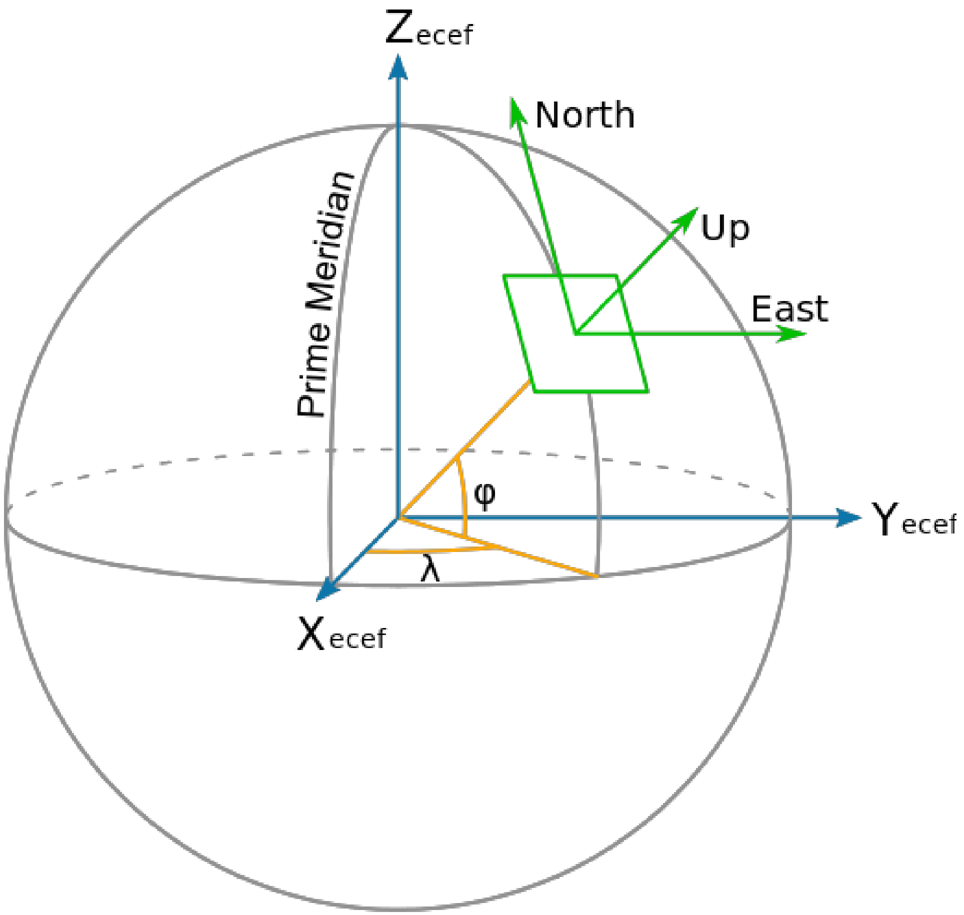
- WGS84 (World geodetic System) ist ein Standardmodell der Erde.
- Erde wird als Ellipsoid angenommen.
- Die Erdoberfläche ist in Längen- und Breitengrade eingeteilt.
- Eine Position wird beschrieben durch Angabe des Längengrads λ, des Breitengrads φ und der Höhe h (Abstand zur Ellipsoid-Oberfläche)
- Es gibt Formeln um aus ECEF-Koordinaten (x,y,z) die WGS84-Koordinaten (λ, φ, h) und umgekehrt zu berechnen.




http://en.wikipedia.org/wiki/ECEF

# ENU

- ENU = East-North-Up
- Navigation in einem kleineren lokalen Bereich wird bequemerweise in einem kartesischen KS durchgeführt.
- Dabei ist der Ursprung an einer bestimmten Position fixiert.
- Die z-Achse (Up) steht senkrecht zur Oberfläche des Ellipsoids.
- Die x-Achse zeigt in Richtung Ost (East) und die y-Achse in Richtung Norden (North).
- ENU ist also gegenüber ECEF verschoben und rotiert und lässt daher mit einer Transformationsmatrix beschreiben.




http://en.wikipedia.org/wiki/North_east_down