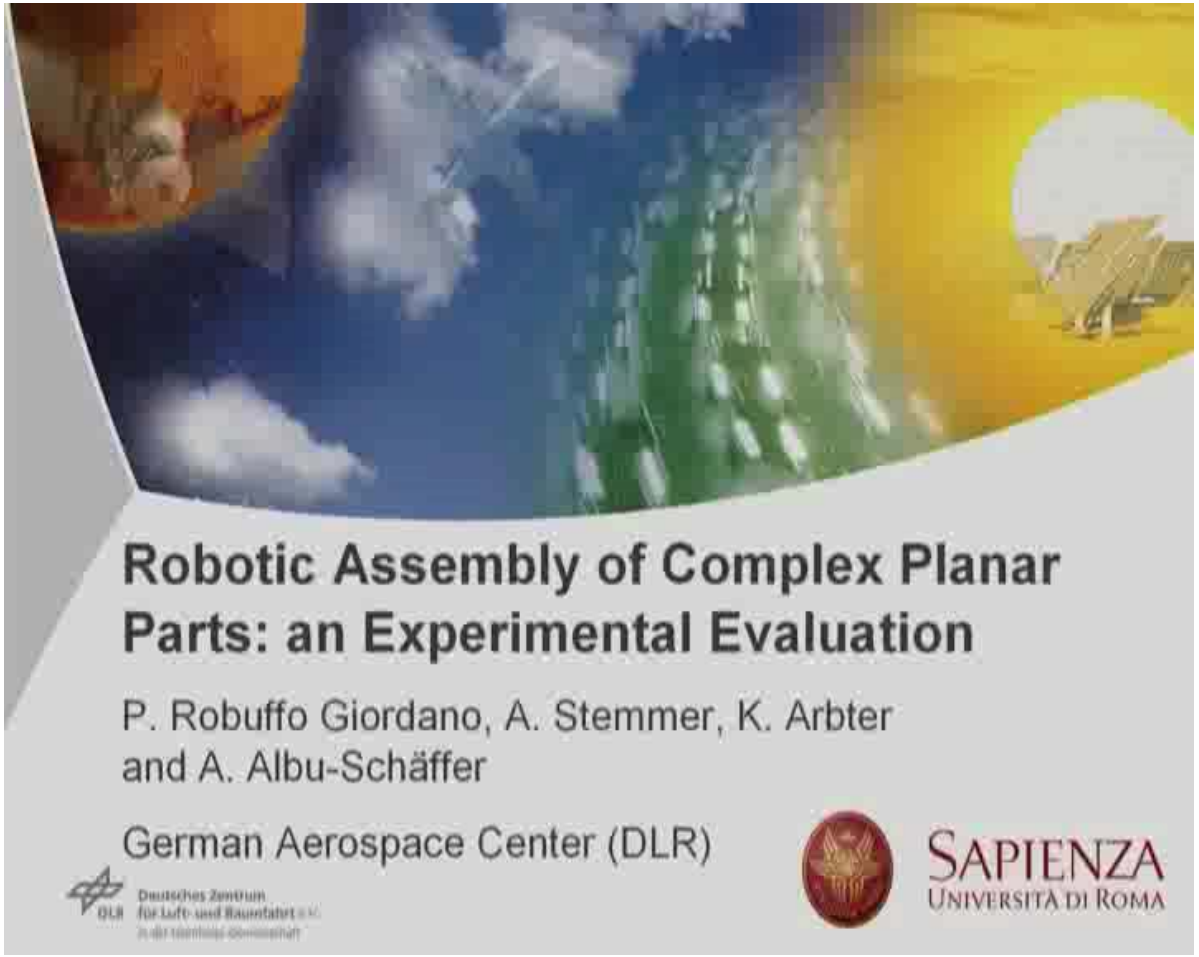
Aufgabenorientierte Regelung – Entkopplung im Task-Raum

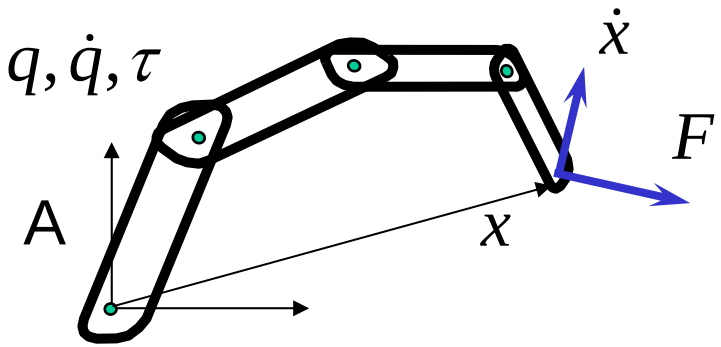
Motivation


Robotic Assembly of Complex Planar Parts: an Experimental Evaluation
P. Robuffo Giordano, A. Stemmer, K. Arbter and A. Albu-Schäffer
German Aerospace Center (DLR)
Deutsches Zentrum für Luft- und Raumfahrt
SAPIENZA
UNIVERSITÀ DI ROMA

Robotergleichungen in aufgabenbezogenen Koordinaten

Robotermodell im Gelenkraum:

$$M(q)\ddot{q} + c(q,\dot{q}) + g(q) = \tau$$



q-Koordinaten von Q ($\mathbb{R}^6$ oder 6-Torus)
$\dot{q}$- Vektor
$c(q,\dot{q}), g(q), \tau$ - Kovektoren
$M(q)$ - kovarianter Tensor zweiter Ordnung

Vorwärtskinematik: lokale Abbildung zwischen Mannigfaltigkeiten (***Q*** und ***SE3***)

$$f : Q \to SE3, \quad x = f(q)$$

f ist eine lokale 1:1 Abbildung zwischen Mannigfaltigkeiten

- für nicht-redundante Manipulatoren (dim(Q)=6) und
- in nicht-singulären Konfigurationen

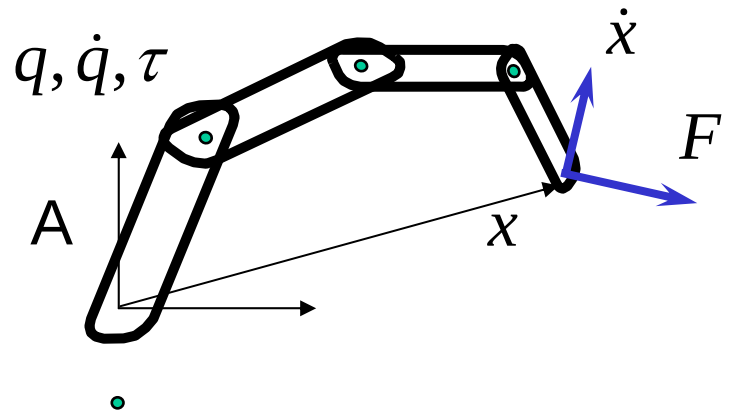
Dann wissen wir wie alle Größen transformieren!

Robotergleichungen in aufgabenbezogenen Koordinaten

Robotermodell in ***SE3*** Koordinaten:

$$\boxed{M_K(q)\ddot{x} + c_K(q,\dot{q}) + g_K(q) = F}$$

$$x = f(q); \qquad q = f^{-1}(x)$$
$$\dot{x} = J(q)\dot{q}; \qquad \dot{q} = J^{-1}(q)\dot{x}$$



Und die Beschleunigung?

$$\ddot{x} = J(q)\ddot{q} + \dot{J}(q)\dot{q}$$

Zusatzinfo:(ein Vektor auf der $2n$ Mannigfaltigkeit mit Koordinaten $(q,\dot{q})$: *Tangentenbündel*) -Tafelerklärung

$$F = J^{-T}(q)\tau; \qquad \tau = J^{T}(q)F$$ (das gleiche mit $g_K(q)$)

$$c_K(q,\dot{q}) = J^{-T}(q)c(q,\dot{q}) \underline{\; - M_K(q)\dot{J}(q)\dot{q}}$$

$$M_K(x) = J^{-T}(q)M(q)J^{-1}(q); \quad M(q) = J^{T}(q)M_K(x)J(q)$$

alternative Formulierung:

$$M_x(x)\ddot{x} + c_x(x,\dot{x}) + g_x(x) = F$$

eingeschränkt auf Konfigurationen in denen $x = f(q)$ eine 1:1 Abbildung ist

Zusatzfolie Jakobimatrix Manipulator

wenn $T_{i,j} = \begin{bmatrix} R_{i,j} & p_{i,j} \\ 0 & 1 \end{bmatrix}$

für serielle Manipulatoren mit rotatorischen Gelenken ist das basic Jacobian:

$$J_j(q) = \begin{bmatrix} R_{j,1} z_1 \times p_{1,n+1}, & \cdots & , R_{j,n} z_n \times p_{n,n+1} \\ R_{j,1} z_1, & \cdots & , R_{j,n} z_n \end{bmatrix}$$

z_i : Achse Gelenk i

$z_i, p_{i,n+1}$ ausgedrückt im Koordinatensystem der Achse i

für serielle Manipulatoren mit translatorischen Gelenken ist das basic Jacobian:

$$J_j(q) = \begin{bmatrix} R_{j,1} z_1, & \cdots, R_{j,n} z_n \\ 0, & \cdots, 0 \end{bmatrix}$$

(Skizze an der Tafel)

zur Herleitung siehe z.B. [Spong, Khalil, Murray]

SE3 Koordinaten und Jakobimatrizen

(Was bedeutet hier genau F? Wie berechne ich J?)

Robotermodell in ***SE3*** Koordinaten x:

$$\boxed{M_K(q)\ddot{x} + c_K(q,\dot{q}) + g_K(q) = F}$$

Ist hier F ein Wrench $F = (f, m)^T$?

$$x = f(q); \qquad q = f^{-1}(x)$$

$$\boldsymbol{x} = \begin{bmatrix} p_x \\ p_y \\ p_z \\ \alpha \\ \beta \\ \gamma \end{bmatrix} \quad \dot{\boldsymbol{x}} = \begin{bmatrix} \dot{p}_x \\ \dot{p}_y \\ \dot{p}_z \\ \dot{\alpha} \\ \dot{\beta} \\ \dot{\gamma} \end{bmatrix}$$

F ist eine verallgemeinerte Kraft dual zu $\dot{x}$ s.d. $F^T \dot{x} = P$

Jakobimatrizen allgemein:

$$\dot{x} = \frac{\mathrm{d}f(q)}{\mathrm{d}t} = \frac{\partial f(q)}{\partial q}\dot{q} = J(q)\dot{q}$$

Der duale Vector zu m ist ω.

$$\dot{x}_j = J_j(q)\dot{q}, \quad \tau = J_j^T(q)F_j \quad \text{mit} \quad \dot{x}_j = \begin{bmatrix} v_j \\ \omega_j \end{bmatrix}, \quad F_j = \begin{bmatrix} f_j \\ m_j \end{bmatrix}$$

(basic Jacobian, geometric Jacobian)

zur Herleitung on J_j siehe z.B. [Spong, Khalil, Murray]

ausgedrückt im Koordinatensystem j

z.B. ausgedrückt im TCP Koordinatensystem - **body** Jacobian

Zusatzfolie Jakobimatrizen für Orientierung

(wie komme ich von ω zur notwendigen Vektordarstellung von SO3?)

Die Winkelgeschwindigkeit ω ist zwar nicht zu lokalen Koordinaten von SO3 integrierbar, es besteht aber folgende Beziehung zur Ableitung von R:

$$\hat{\omega}^{j}_{i,j} = R^{T}_{i,j}\dot{R}_{i,j}$$

(Herleitung z.B. Murray, S.52)

Matrix Repräsentierung eines Vektors

$$\hat{\omega} = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix} \text{ s.d. } \quad \omega \times r = \hat{\omega} r$$

Vektorprodukt als Matrixoperation

Da lokale Koordinaten x_R von SO3 auch in Abhängigkeit von R ausgedrückt sind

$$x_R = f_R(R_v) \Rightarrow \dot{x}_R = J_{f_R}\dot{R}_v$$

R_v enthält alle Elemente von R als Vektor angeordnet

kann man die Transformation zwischen den beiden Vektortypen herleiten:

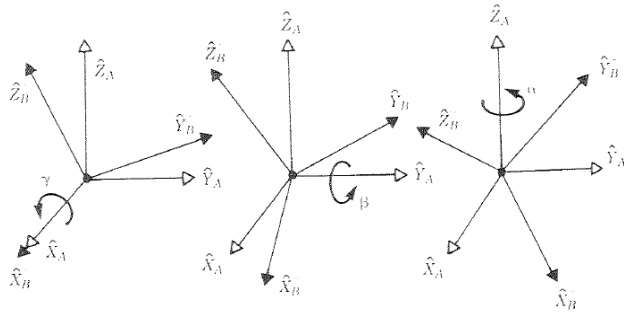
$$\dot{x}_R = J_R\omega$$

Somit gilt für die Jacobimatrix

$$\dot{x} = \begin{bmatrix} v \\ \dot{x}_R \end{bmatrix} = \underbrace{\begin{bmatrix} I_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & J_R \end{bmatrix}}_{J_{TR}} \begin{bmatrix} v \\ \omega \end{bmatrix} = \underbrace{J_{TR}J_b}_{J}\dot{q} = J\dot{q}$$

Zusatzfolie Jakobimatrizen Orientierung

Beispiel RollPitchYaw-Winkel:



$$\gamma = \text{atan2}(r_{32}, r_{33})$$

$$\beta = \text{atan2}\left(-r_{31}, \sqrt{r_{11}^2 + r_{21}^2}\right)$$

$$\alpha = \text{atan2}(r_{21}, r_{11})$$

(Singularität bei β=+/-90°)

$$\begin{bmatrix} \dot{\gamma} \\ \dot{\beta} \\ \dot{\alpha} \end{bmatrix} = J_R \omega = \begin{bmatrix} \frac{\cos\alpha}{\cos\beta} & \frac{\sin\alpha}{\cos\beta} & 0 \\ -\sin\alpha & \cos\alpha & 0 \\ \tan\beta\cos\alpha & \tan\beta\sin\alpha & 1 \end{bmatrix} \omega$$

Beispiel Quaternionen (Euler Parameter):

$$\lambda = [\lambda_0, \lambda_1, \lambda_2, \lambda_3] = \left[\cos\frac{\theta}{2}, \vec{k}\sin\frac{\theta}{2}\right]$$

$$\lambda_0 = \frac{1}{2}\sqrt{r_{11} + r_{22} + r_{33} + 1}$$

$$\lambda_1 = \frac{1}{2}\text{sgn}(r_{32} - r_{23})\sqrt{r_{11} - r_{22} - r_{33} + 1}$$

$$\lambda_2 = \frac{1}{2}\text{sgn}(r_{13} - r_{31})\sqrt{-r_{11} + r_{22} - r_{33} + 1}$$

$$\lambda_3 = \frac{1}{2}\text{sgn}(r_{21} - r_{12})\sqrt{-r_{11} - r_{22} + r_{33} + 1}$$

$$J_R = \frac{1}{2}\begin{bmatrix} -\lambda_1 & -\lambda_2 & -\lambda_3 \\ \lambda_0 & \lambda_3 & -\lambda_2 \\ -\lambda_3 & \lambda_0 & \lambda_1 \\ \lambda_2 & -\lambda_1 & \lambda_0 \end{bmatrix}$$

Entkoppelte Positionsregelung in Task-Koordinaten

(Feedback Linearisierung oder I/O Linearisierung oder Computed Torque Controller)

Robotermodell in $\boldsymbol{SE3}$ Koordinaten x:

$$\boxed{M_K(q)\ddot{x} + c_K(q,\dot{q}) + g_K(q) = F}$$

Regler

$$\boxed{F = M_K(q)[\ddot{x}_d + K(x_d - x) + D(\dot{x}_d - \dot{x})] + c_K(q,\dot{q}) + g_K(q)}$$

$$\tau = J^T(q)F$$

Kommandierung eines Gelenkdrehmoments erfordert:
- Direktantriebe oder
- Gelenkdrehmomentregelung

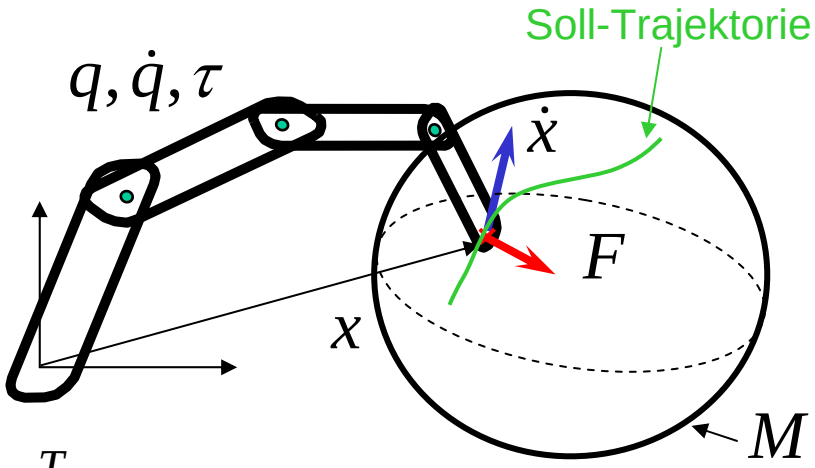
Es folgt eine lineare, entkoppelte Fehlerdynamik:

$$\boxed{\ddot{e} + D\dot{e} + Ke = 0} \qquad e = x_d - x$$

Drehmomentschnittstelle

Hybride Kraft-/Positionsregelung in Task-Koordinaten

Der Roboter soll entlang einer Trajektorie auf eine Mannigfaltigkeit M (Objektoberfläche) fahren und eine vorgegebene Kraft auf M ausüben!



wenn x_M : r-dim. lokale Koordinaten für M, wollen wir x_M bzw. $\dot{x}_M$ regeln, sowie m-r Wrenches F_R, die reziprok zu $\dot{x}_M$ sind, d.h. $F_R{}^T \dot{x}_M = 0$

Wir wählen somit als Geschwindigkeitsvektor $\dot{x} = (\dot{x}_M, \dot{x}_R)$ ← dual zu F_R

Robotermodell in ***SE3*** Koordinaten x:

$$\boxed{M_K(q)\ddot{x} + c_K(q,\dot{q}) + g_K(q) = F - F_{Rext}}$$

← Kontaktkraft

Der Regler F hat dann zwei Komponenten:

F_M - für Positionsregelung

F_R - für Kraftregelung

$$F_C = \begin{bmatrix} F_M \\ F_R \end{bmatrix}$$

Hybride Kraft-/Positionsregelung in Task-Koordinaten

$$M_K(q)\ddot{x} + \cancel{c_K(q,\dot{q}) + g_K(q)} = F - \cancel{F_{R\,ext}}$$

zusätzliche Rückkopplung der externen Kraft

Regler:

$$\boxed{F = M_K(q)F_C + \cancel{c_K(q,\dot{q}) + g_K(q)} + \cancel{(I - M_K(q))F_{R\,ext}}}$$

$$\boxed{\tau = J^T(q)F}$$

Durch einsetzen, resultiert die Fehlerdynamik:

$$\ddot{x} = F_C - F_{R\,ext}$$

oder aufgetrennt in Koordinaten für Positions- und Kraftregelung:

$$\ddot{x}_M = F_M$$

$$\ddot{x}_R = F_R - F_{R\,ext}$$

Hybride Kraft-/Positionsregelung in Task-Koordinaten

$$\ddot{x}_M = F_M$$

$$\ddot{x}_R = F_R - F_{R\,ext}$$

1) Entkopplungsregler für Position (wie gehabt)

$$\boxed{F_M = \ddot{x}_{Md} + D(\dot{x}_{Md} - \dot{x}_M) + K(x_{Md} - x_M)}$$

Es folgt eine lineare, entkoppelte Fehlerdynamik:

$$\boxed{\ddot{e}_M + D\dot{e}_M + Ke_M = 0} \qquad e_M = x_{Md} - x_M$$

2) Kraftregler

$$\boxed{F_R = F_{Rd} + K_F(\dot{F}_{R\,d} - \dot{F}_{R\,ext})}$$

Es folgt die Fehlerdynamik:

$$\boxed{\ddot{x}_R = K_F\dot{e}_F + e_F} \qquad e_F = F_{R\,d} - F_{R\,ext}$$

$\ddot{x}_R = 0$ mit idealem starren Kontakt

Hybride Kraft-/Positionsregelung in Task-Koordinaten

$$\ddot{x}_R = F_R - F_{R\,ext}$$

2) Kraftregler – nicht ideal starrer Kontakt: $F_{R\,ext} = K_R x_R$

Roboter + Umgebungssteifigkeit

$$\boxed{F_R = K_R^{-1}\left(\ddot{F}_{R\,d} + K_F\,(\dot{F}_{R\,d} - \dot{F}_{R\,ext})\right) + F_{Rd}}$$

Es folgt die Fehlerdynamik:

$$\boxed{\ddot{e}_F + K_F \dot{e}_F + K_R e_F = 0} \qquad e_F = F_{R\,d} - F_{R\,ext}$$

Vorteil des Reglers: optimale, lineare, völlig entkoppelte Dynamik des geregelten Systems

Nachteil: Das Verhalten des Systems wird tiefgreifend verändert, d.h.

- -beschränkte Robustheit (gegenüber Parameterungenauigkeiten ($M, g, c \ldots, K_R$)
- -empfindlich gegenüber unmodellierte Dynamik (z.B. Reibung)
- -große Stellgrößeneinwirkung notwendig - Stellgrößensättigung

-Beispiel: 2dof Manipulator auf Kreis an der Tafel

Redundante Roboter

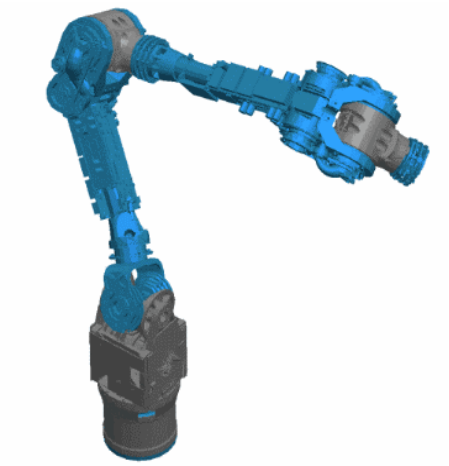
Aufgaben-Raum (Task Space): m dimensional $m \leq 6$

Gelenk-Raum Q: n dimensional

redundanter Manipulator: $n > m$

n-m: Redundanzgrad

n=7, m=6



n=10, m=6

Nullraumbewegung für Justin

Redundante Roboter: Dynamik

Für $m<n$ sind die Koordinaten x nicht ausreichend, um den Roboter komplett zu beschreiben. Allerdings kann das dynamische Verhalten des Endeffektors dadurch komplett beschrieben werden.

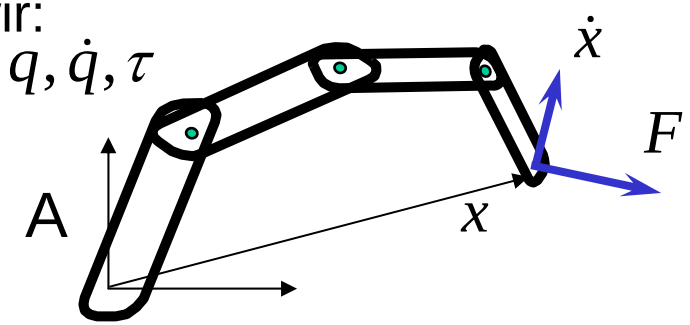
Für nichtredundante Roboter hatten wir:

$$\boxed{M_K(q)\ddot{x} + c_K(q,\dot{q}) + g_K(q) = F}$$

$$\dot{x} = J(q)\dot{q}; \qquad \boxed{\dot{q} = J^{-1}(q)\dot{x}}$$

?? $$\boxed{F = J^{-T}(q)\tau;} \qquad \tau = J^T(q)F$$

$$\boxed{M_K(x) = J^{-T}(q)M(q)J^{-1}(q);}$$



Für redundante Roboter ist J_{mxn} nicht quadratisch, also nicht invertierbar!

Lösung: pseudoinverse Matrix: $$\boxed{J_{m\times n}J^{\#}_{n\times m} = I_{m\times m}}$$ $(J^{\#T}_{m\times n}J^{T}_{n\times m} = I_{m\times m})$

man bemerke, dass $J^{\#}_{n\times m}J_{m\times n} \neq I_{n\times n}$ $(J^{T}_{n\times m}J^{\#T}_{m\times n} \neq I_{n\times n})$

mit $$\boxed{J^{\#} = A\ J^T(JAJ^T)^{-1}}$$

A-p.d. matrix

$A=I$ Sonderfall:
Moore-Penrose
Pseudoinverse
$J^{\#} = J^T_{n\times m}(JJ^T)^{-1}$

Redundante Roboter: Dynamik

dann ist:

nicht –singulärer Fall!

$$\dot{q} = J^{\#}(q)\dot{x} \qquad F = J^{\#T}(q)\tau$$

$$M_R(x) = J^{\#T}(q)M(q)J^{\#}(q)$$

und wieder $\boxed{M_R(q)\ddot{x} + c_R(q,\dot{q}) + g_R(q) = F}$

Aber: Wie wählt man die Matrix A richtig?

(Unterschiedliche A führen zu unterschiedlichen Ergebnissen für M_R !?!)

Alles, was man mit einer Pseudoinversen erreichen kann, kann man auch ohne lösen!

Ansatz: um M_R zu berechnen, berechnet man zuerst die Inverse M_R^{-1}

Mobilitätsmatrix

M_R^{-1} ist ein Tensor vom Typ (0,2) –zweimal kontravariant

transformiert also wie folgt:

weil $p^T M_R^{-1} p = E_{kin}$

$p = M_R \dot{x}$ Impuls, kovariant

$$M_R^{-1} = JM^{-1}J^T$$

folglich $\boxed{M_R = (JM^{-1}J^T)^{-1}}$

Redundante Roboter: Dynamik

$$J^{\#} = A\ J^T \underline{(JAJ^T)^{-1}} \qquad\qquad M_R = \underline{(JM^{-1}J^T)^{-1}}$$

Es lässt sich direkt überprüfen, dass für $A = M^{-1}(q)$ $\Updownarrow$

$$M_R(x) = J^{\#T}(q)M(q)J^{\#}(q)$$

Also, es gilt für die Pseudoinverse:

$$\boxed{J^{\#} = M^{-1}\ J^T (JM^{-1}J^T)^{-1}}$$

Übung: Man überprüfe, dass mit dieser Pseudoinversen, die beiden Ausdrücke für M_R äquivalent sind

kleine Formelsammlung

$(AB)^T = B^T A^T$ $\quad$ $A, B -$ Matrizen

$(AB)^{-1} = B^{-1} A^{-1}$ $\quad$ $A, B -$ invertierbare Matrizen

Eigenschaften der Massen- und Steifigkeitsmatrizen $\quad A \in \{M, K\}$

A ist positiv definit, d.h.

$$v^T A v > 0, \quad \forall v, \|v\| \neq 0$$

$$\angle(v, Av) < 90° \quad \forall v$$

Alle Eigenwerte von A sind positiv

A ist symmetrisch, d.h.

$$A^T = A$$

Da A symmetrisch, p.d. ist, so gilt das Gleiche für

$J^T A J$ $\quad$ wenn J – nicht singulär

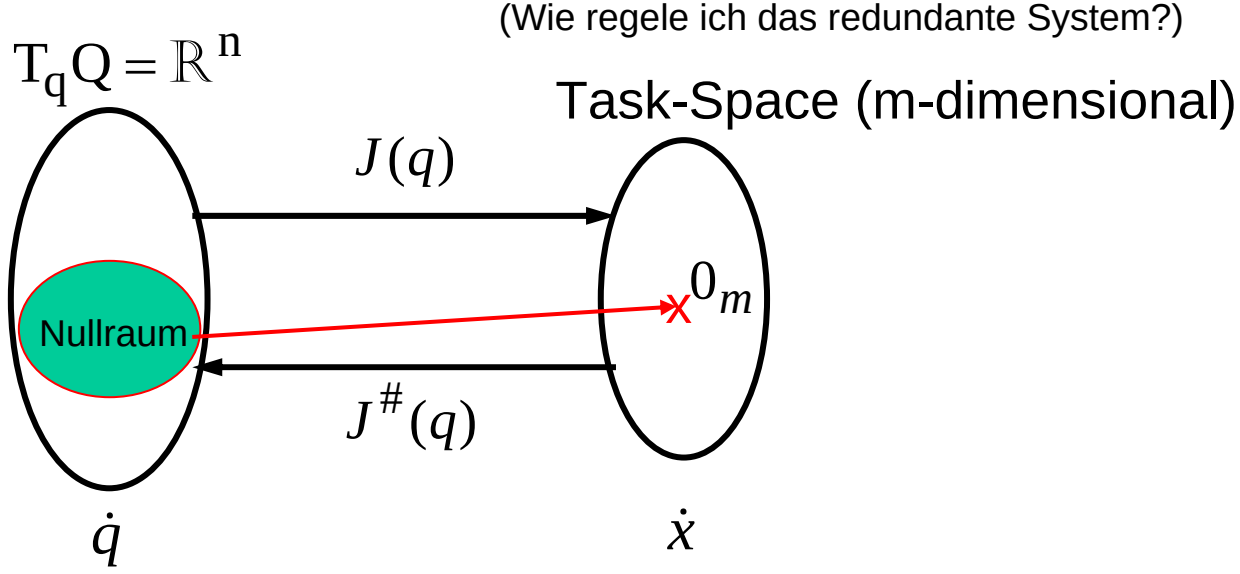
Orthogonale Zerlegung:

$A = U \Lambda U^T$ $\quad$ Λ - diagonal $\quad \lambda_i$ - Eigenwerte von A

U - orthogonale Matrix

Nullraumbewegung

(Wie regele ich das redundante System?)



Nullraumprojektion:

$$\dot{q}_N = (I - J^{\#}(q)J(q))\dot{q}_0$$

Überprüfung
$J(q)\dot{q}_N = 0$

Inverskinematik:

$$\boxed{\dot{q} = J^{\#}(q)\dot{x} + (I - J^{\#}(q)J(q))\dot{q}_0}$$

$$\delta q = J^{\#}(q)\delta x + (I - J^{\#}(q)J(q))\delta q_0$$

Interpretation der Pseudoinversen

(was bedeuten die unterschiedlichen Skalierungen in der Pseudoinversen?)

Inverskinematik: $\dot{q} = J^{\#}(q)\dot{x}$

$$\boxed{J^{\#} = A\ J^T (JAJ^T)^{-1}}$$

Aus „Grundlagen intelligenter Roboter“ wissen Sie, dass für $A=I$, d.h.

$$J^{\#} = J^T (JJ^T)^{-1}$$

die Inverskinematik $\|\dot{q}\|$ minimiert, unter der Zwangsbedingung

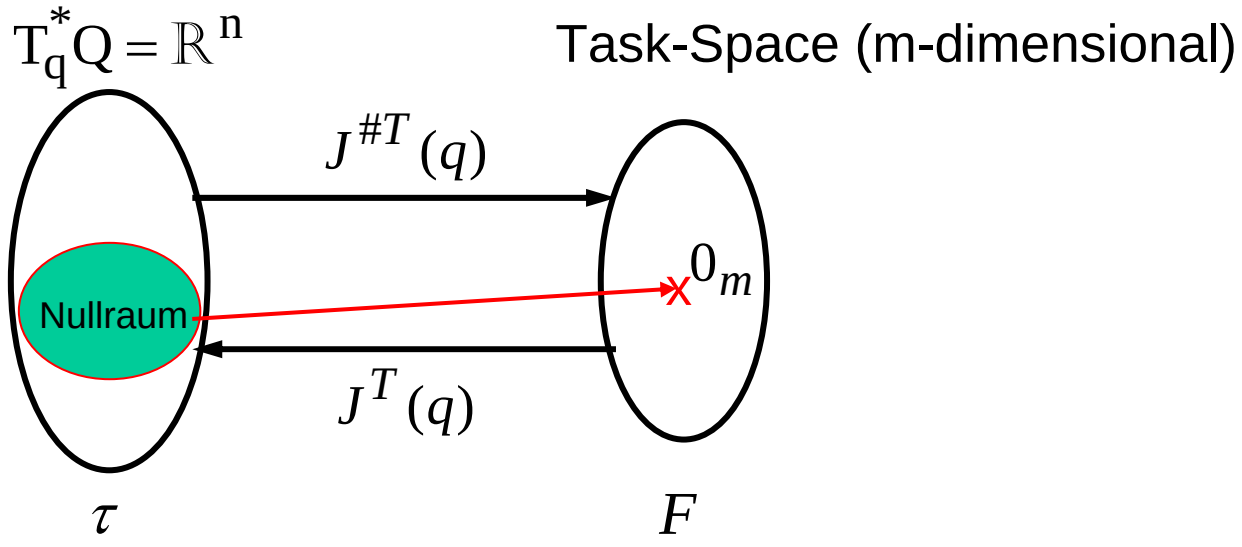
$$\dot{x} - J(q)\dot{q} = 0$$

Im Allgemeinen, wird für eine Gewichtungsmatrix A die Norm $\|\dot{q}\|_{A^{-1}}$ minimiert

D.h., für $A = \frac{1}{2} M^{-1}(q)$ wird die kinetische Energie minimiert:

$$\|\dot{q}\|_M^2 = \frac{1}{2}\dot{q}^T M(q)\dot{q}$$

Nullraummoment



Nullraumprojektion:

$$\tau_N = (I - J^T(q) J^{\#T}(q))\tau_0$$

Überprüfung
$J^{\#T}(q)\tau_N = 0$

Kommandiertes Gelenkmoment:

$$\boxed{\tau = J^T(q)F + (I - J^T(q) J^{\#T}(q))\tau_0}$$

kartesische Komponente
(primäre Aufgabe)

Nullraum – Komponente
(sekundäre Aufgabe)

Zusatzfolie: Nullraummoment

$$\boxed{\tau = J^T(q)F + (I - J^T(q)J^{\#T}(q))\tau_0}$$

Bemerkung: durch Einsetzen in

$$\boxed{J^{\#} = M^{-1}\ J^T(JM^{-1}J^T)^{-1}}$$

$$M(q)\ddot{q} + c(q,\dot{q}) + g(q) = \tau$$

und Multiplikation mit $J(q)M(q)^{-1}$ folgt:

$$\underline{\ddot{x}} + JM^{-1}(c(q,\dot{q}) + g(q)) - \dot{J}\dot{q} = JM^{-1}J^{\#T}F +$$

$$+ \underbrace{JM^{-1}(I - J^T J^{\#T})\tau_0}_{\Downarrow\ 0!}$$

(Herleitung an der Tafel)

Diese Pseudoinverse ist die einzige, die auch keine Beschleunigungen (nicht nur keine Kräfte) am TCP erzeugt = „dynamisch konsistent“

Zusatzfolie: Rücktransformation der Kartesischen Massenmatrix

Wenn man im redundanten Fall die kartesische Massenmatrix in Gelenkkoordinaten zurück transformiert,

$$M_J(q) = J^T(q) M_R(q) J(q)$$

erhält man nicht die komplette Massenmatrix im Gelenkraum, sondern nur den Anteil, der an am TCP sichtbar ist! Die Nullraumkomponente wird herausgefiltert, analog zur Transformation des Gelenkmomentes in TCP Koordinaten und zurück.

$$\boxed{M_J(q) \neq M(q)}$$

Stabilität im Nullraum

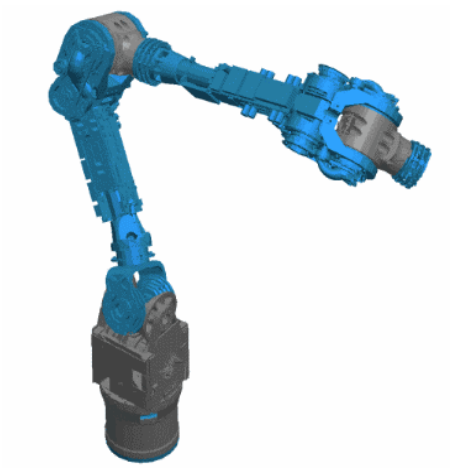
(Kann ein Roboter sich in kartesischen Koordinaten einschwingen und im Nullraum instabil sein?)

$$\boxed{M_R(q)\ddot{x} + c_R(q,\dot{q}) + g_R(q) = F}$$

Mit dem Computed Torque folgt eine lineare, entkoppelte Fehlerdynamik:

$$\boxed{\ddot{e} + D\dot{e} + Ke = 0} \qquad e = x_d - x$$

n=7, m=6



Die Nullraum Komponente sollte:

- mindestens ein Dämpfungsterm enthalten

$$\tau_0 = -D\dot{q}$$

- ein Term zur Vermeidung von Endanschlägen

$$\tau_0 = K_N(q_0 - q)$$

- evtl. Terme zur Kollisionsvermeidung, bzw. zur Optimierung weiterer Gütekriterien

Behandlung von Singularitäten

$$\dot{x}_m = J_{m\times n}(q)\dot{q}_n$$

$r > 0$

der Roboter ist singulär, wenn $Rang\{J_{m\times n}(q)\} = m - r < m$

für nicht redundante Roboter, heißt dies: $d(q) = \det(J_{m\times m}(q)) = 0$

für redundante Roboter, heißt dies: $d(q) = \det(J(q)J^T(q)) = 0$

mxm Matrix

$$d(q) = d_1(q)\cdot d_2(q)\cdot \ldots \cdot d_i(q)$$

i unterschiedliche Typen von Singularitäten

In der Nähe von Singularitäten werden (m-r) Koordinaten geregelt und der Manipulator als redundanter Manipulator betrachtet.

Eine Regelung in singulärer Richtung (lokale Koordinate d_i) erfolgt im Nullraum der primären Aufgabe