

Weihnachtsprojekt: Simulation und Regelung eines Knickarmroboters

A. Paul

24. Januar 2025

In diesem Projekt wird ein 2-DOF-Knickarmroboter simuliert, bei dem sich jeweils ein Aktuator in jedem Gelenk befindet. Das System ist in Abbildung 1 dargestellt.

Hierbei kommen die in der Vorlesung MModellierung und Simulation in der Mechatronik "vorgestellten Methoden zur Anwendung. Zur Simulation wird insbesondere Matlab sowie dessen Erweiterung Simulink verwendet.

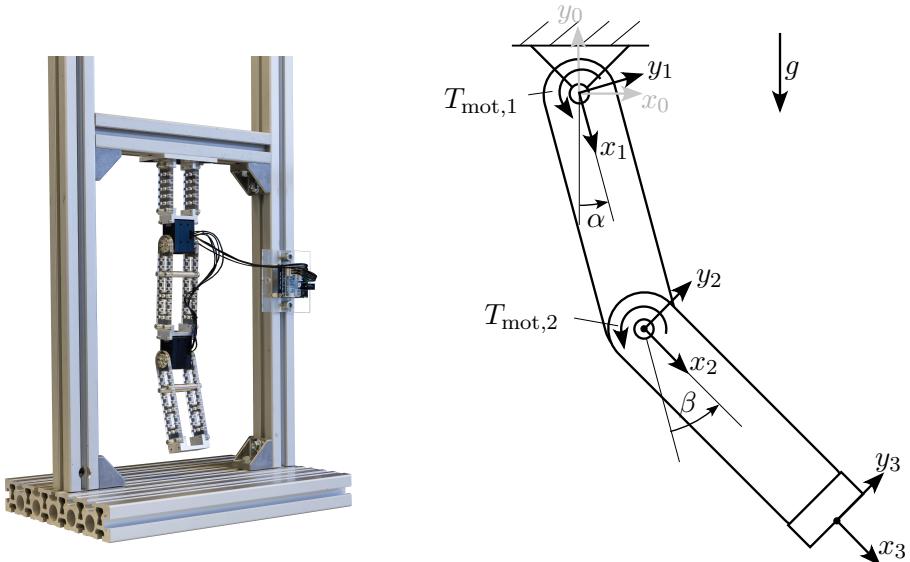


Abbildung 1: Foto sowie schematische Darstellung des zu untersuchenden Roboters [Fuchs23].

1 Berechnung der Kinematik des Roboters

In diesem Kapitel wird die Vorwärtskinematik des Knickarmroboters ermittelt. Dazu wird die modifizierte DH-Konvention verwendet. Die DH-Parameter werden zunächst bestimmt, die Ergebnisse sind in Tabelle 1 dargestellt.

Zur Berechnung der Vorwärtskinematik wird die Symbolic Toolbox in Matlab verwendet. Es werden Transformationsmatrizen definiert, wie in [Fehr24b]. Diese werden anschließend als Matlab-Funktionen exportiert, mit den Variablen α , β und den Längen des Roboters. Die hier berechneten Größen werden in den späteren Aufgaben benötigt. In den folgenden Skripten können diese Funktionen aufgerufen



werden. Dieses Vorgehen ermöglicht es, dass bei einem Fehler, z.B. in der Berechnung der Endeffektorposition, nicht in jedem Skript eine Korrektur durchgeführt werden muss. Stattdessen wird die fehlerhafte Funktion neu berechnet und die aktualisierte Version wird automatisch in den nachfolgenden Skripten verwendet. Diese Vorgehensweise wird in allen folgenden Skripten beibehalten.

Achse	a_{i-1}	α_{i-1}	d_i	θ_i	Art
1	0	0	0	α	rot.
2	l_1	0	0	β	rot.
3	l_2	0	0	0	trans.

Tabelle 1: Beispielhafte DH-Parameter.

Zur Aufgabe korrespondierendes Matlab Skript: Aufgabe1_Kinematik.m

2 Bestimmung der Bewegungsgleichungen des planaren 2-Arm-Roboters

In diesem Kapitel wird die Bewegungsgleichung des Knickarmroboters ermittelt. Dazu werden die Lagrange'schen Gleichungen zweiter Art verwendet. Das Vorgehen und alle verwendeten Gleichungen wurden aus [Fehr24c] entnommen.

Zur Aufgabe korrespondierendes Matlab Skript: Aufgabe2_Dynamik.m

2.1 Teilaufgabe a): Wahl der generalisierten Koordinaten

Wie in Abbildung 1 erkennbar, besitzt das System zwei unabhängige Freiheitsgrade. Dementsprechend müssen zwei generalisierte Koordinaten gewählt werden, um die Position und Orientierung (POSE) jedes Körpers des Systems eindeutig zu beschreiben. Es bietet sich an, die generalisierten Koordinaten folgendermaßen zu wählen:

$$\mathbf{y} = \begin{bmatrix} \alpha \\ \beta \end{bmatrix}, \quad \dot{\mathbf{y}} = \begin{bmatrix} \dot{\alpha} \\ \dot{\beta} \end{bmatrix}, \quad \ddot{\mathbf{y}} = \begin{bmatrix} \ddot{\alpha} \\ \ddot{\beta} \end{bmatrix}.$$

In jedem der Winkel α und β ist ein Aktuator angebracht. Dies hat zur Folge, dass ein vollaktuierter Zustand entsteht, wodurch die Eingangsmatrix \mathbf{B} vollen Rang hat und somit invertierbar ist. Dadurch ist die inverse Dynamik als Regelungsstrategie leicht anwendbar. Die genaue Anwendung der inversen Dynamik wird jedoch in späteren Kapiteln ausführlicher beschrieben.

2.2 Teilaufgabe b): Bestimmung der Bewegungsgleichung

In dieser Teilaufgabe wird die Bewegungsgleichung nach Lagrange zweiter Art berechnet. Die Schwerpunkte (SP) der beiden Arme liegen bezüglich der körperfesten Koordinatensysteme jeweils auf halber Länge in negativer x -Richtung:

$$x_{i,SP} = -\frac{\ell_i}{2} \quad ; \quad y_{i,SP} = 0.$$

2.2.1 Bestimmung der Jacobimatrizen des Roboters

Es werden die Transformationsmatrizen aufgerufen, die in Aufgabe 1 berechnet werden. Dann werden weitere Transformationsmatrizen berechnet, die die Position und Orientierung (POSE) jedes Körpers bezüglich des stationären Inertialsystems beschreiben. Aus diesen Matrizen werden die Ortsvektoren der Schwerpunkte im Inertialsystem $\mathbf{r}_{0,SP1}^0, \mathbf{r}_{0,SP2}^0$ berechnet. Es ergeben sich folgende Vektoren:

$$\mathbf{r}_{0,SP1}^0 = \begin{bmatrix} l_1 \sin(\alpha) \\ \frac{l_1}{2} \\ \frac{l_1 \cos(\alpha)}{2} \\ 0 \end{bmatrix}; \quad \mathbf{r}_{0,SP2}^i = \begin{bmatrix} \frac{l_2(\sin(\alpha) \sin(\beta) - \cos(\alpha) \cos(\beta))}{2} + l_1 \sin(\alpha) \\ \frac{l_2(\cos(\alpha) \sin(\beta) + \cos(\beta) \sin(\alpha))}{2} + l_1 \cos(\alpha) \\ 0 \end{bmatrix}.$$

Es kann die Rotationsmatrix aus den Transformationsmatrizen entnommen werden, und darauf können die Rotationsvektoren der beiden Körper ermittelt werden. Da es sich in diesem Fall um ein leichtes System handelt, werden diese jedoch in dem Skript definiert und nicht berechnet. Die Rotationsvektoren ergeben sich zu:

$$\mathbf{s}_{0,SP1}^0 = \begin{bmatrix} 0 \\ 0 \\ \alpha + \alpha_0 \end{bmatrix}; \quad \mathbf{s}_{0,SP2}^0 = \begin{bmatrix} 0 \\ 0 \\ \alpha + \beta + \alpha_0 + \beta_0 \end{bmatrix}$$

Anschließend können die Jacobimatrizen der Translation \mathbf{J}_T und der Rotation \mathbf{J}_R folgendermaßen berechnet werden:

$$\mathbf{J}_{T,SPi} = \frac{\partial \mathbf{r}_{0,SPi}^0}{\partial \mathbf{y}}; \quad \mathbf{J}_{R,SPi} = \frac{\partial \mathbf{s}_{0,SPi}^0}{\partial \mathbf{y}}$$

es ergibt sich:

$$\mathbf{J}_{T,SP1} = \begin{bmatrix} \frac{l_1 \cos(\alpha)}{2} & 0 \\ -\frac{l_1 \sin(\alpha)}{2} & 0 \\ 0 & 0 \end{bmatrix}; \quad \mathbf{J}_{R,SP1} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \end{bmatrix};$$

$$\mathbf{J}_{T,SP2} = \begin{bmatrix} \frac{l_2(\cos(\alpha) \sin(\beta) + \cos(\beta) \sin(\alpha))}{2} + l_1 \cos(\alpha) & \frac{l_2(\cos(\alpha) \sin(\beta) + \cos(\beta) \sin(\alpha))}{2} \\ -\frac{l_2(\sin(\alpha) \sin(\beta) - \cos(\alpha) \cos(\beta))}{2} - l_1 \sin(\alpha) & -\frac{l_2(\sin(\alpha) \sin(\beta) - \cos(\alpha) \cos(\beta))}{2} \\ 0 & 0 \end{bmatrix};$$

$$\mathbf{J}_{R,SP2} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 1 \end{bmatrix}.$$

2.2.2 Bestimmung der potentiellen Energie

Die potentielle Energie kann mit der zuvor geleisteten Arbeit leicht berechnet werden, durch das Skalarprodukt zwischen dem jeweiligen Schewrpunktsortsvektor und einem zuvor definierten Gravitationsvektor:

$$U_i = \mathbf{r}_{0,SPi}^{0 \top} \mathbf{g}^0,$$

wobei

$$\mathbf{g}^0 = \begin{bmatrix} 0 \\ g \\ 0 \end{bmatrix}.$$



Es ergibt sich:

$$U_1 = \frac{gl_1 m_1 \cos(\alpha)}{2}; U_2 = gm_2 \left(\frac{l_2 (\cos(\alpha) \sin(\beta) + \cos(\beta) \sin(\alpha))}{2} + l_1 \cos(\alpha) \right).$$

2.2.3 Bestimmung der nichtkonservativen Kräfte

Die nichtkonservativen Kräfte berücksichtigt nur das Reibmoment, das aus viskoser und statischer Reibung besteht. Die Reibung wird nach Stribeck modelliert. Um die Stetigkeit des Reibmoments zu erreichen, wird die Signum-Funktion durch den Arcustangens approximiert. Es ergibt sich folgende Formulierung:

$$\tau_{R,\alpha} = \frac{2}{\pi} \cdot \arctan(b_\alpha \dot{\alpha}) \cdot (F_{s1} + |\dot{\alpha}| \cdot \mu_{v1}),$$

wobei b_α ein wählbarer Parameter ist, mit dem die Steigung der Arcustangens Funktion eingestellt werden kann. Dasselbe Prinzip wird angewandt auf $\tau_{R,\beta}$.

2.2.4 Bestimmung der Bewegungsgleichung in symbolischer Matrixform

Bei der Berechnung der Bewegungsgleichung ergeben sich große Terme, weshalb nur noch das Vorgehen beschrieben wird. Es wird zunächst die Massenmatrix berechnet. Diese berechnet sich wie folgt:

$$\mathbf{M}_i = m_i \cdot \mathbf{J}_{T,SPi}^T \cdot \mathbf{J}_{R,SPi} + \mathbf{J}_{R,SPi}^T \cdot \mathbf{S}_{0,i} \cdot \mathbf{I}_{SP,i} \cdot \mathbf{S}_{0,i}^T \cdot \mathbf{J}_{R,SPi}$$

wobei $\mathbf{I}_{SP,i}$ das Massenträgheitsmoment des Körpers \mathbf{K}_i bezüglich seines Schwerpunktes beschreibt und $\mathbf{S}_{0,i}$ dessen Rotationsmatrix ist. Die gesamte Massenmatrix ergibt sich durch Summation der Einzelmatrizen:

$$\mathbf{M} = \sum_{i=1}^{n_B} \mathbf{M}_i$$

Anschließend kann die Matrix \mathbf{D} mithilfe der Christoffel Symbole berechnet werden

$$h_{ijk} := \frac{1}{2} \left(\frac{\partial M_{kj}}{\partial y_i} + \frac{\partial M_{ki}}{\partial y_j} - \frac{\partial M_{ij}}{\partial y_k} \right)$$

$$D_{kj} = \sum_{i=1}^{n_B} h_{ijk}(y) \dot{y}_i$$

Nun kann der Gravitationskraftvektor in den verallgemeinerten Koordinaten berechnet werden durch:

$$\mathbf{g} = \frac{\partial U}{\partial \mathbf{y}}.$$

Die restlichen Teile der Bewegungsgleichung sind bereits in den Raum der verallgemeinerten Koordinaten definiert und lauten:

$$\tau_R = \begin{bmatrix} \tau_{R,\alpha} \\ \tau_{R,\beta} \end{bmatrix}; \tau_M = \begin{bmatrix} \tau_{M,\alpha} \\ \tau_{M,\beta} \end{bmatrix};$$

Es ergibt sich eine Bewegungsgleichung der Form:

$$\mathbf{M}(\mathbf{y}) \cdot \ddot{\mathbf{y}} + \mathbf{D}(\mathbf{y}, \dot{\mathbf{y}}) \cdot \dot{\mathbf{y}} + \mathbf{g}(\mathbf{y}) = \tau_R + \tau_M \quad (1)$$



3 Planung der Trajektorie des Endeffektors

In dieser Aufgabe wird die Trajektorie berechnet, welcher der Endeffektor folgen soll. Der Startpunkt und der Endpunkt sind in der Aufgabenstellung definiert:

$$\mathbf{p}_{EF}^0 = \begin{bmatrix} 0 \\ -0.228 \text{ m} \\ 0 \end{bmatrix}; \mathbf{p}_{EF}^d = \begin{bmatrix} 4\sqrt{6} - 4\sqrt{2} - 10 \\ -4\sqrt{6} - 4\sqrt{2} - 10\sqrt{3} \\ 0 \end{bmatrix} \cdot \frac{1}{125} \text{ m}$$

Die Trajektorie wird mit dem in der Vorlesung 18 präsentierten Interpolations-Skript generiert. Dazu muss nur ein Anfangspunkt, ein Endpunkt angegeben und die Anfahrtszeit t_a angegeben werden. Es wird $t_a = 2 \text{ s}$ gesetzt. Es werden die Koeffizienten eines Polynoms generiert, in diesem Fall eines der Ordnung 3, welches die beiden Punkte interpoliert. Daraus kann dann die Sollposition $\mathbf{r}_{des}^{EF}(t)$ bestimmt werden. Durch einfache Differentiation ergeben sich $\dot{\mathbf{r}}_{des}^{EF}(t)$ und $\ddot{\mathbf{r}}_{des}^{EF}(t)$.

Die Problematik liegt bei der Inversen Kinematik, darin, aus der gewünschten Position des Endeffektors die zugehörigen Gelenkwinkel zu berechnen. Dazu wird aus einer Funktion die Position des Endeffektors $r_{EF}(y)$ in Abhängigkeit der verallgemeinerten Koordinaten geladen. Es wird $\mathbf{r}_{des}^{EF}(t)$ alle $\Delta t = 0,05 \text{ s}$ ausgewertet. Die resultierenden Datenpunkte werden dann gleich $r_{EF}(y)$ gesetzt und mit der `solve()`-Funktion von Matlab gelöst. Die daraus resultierenden Datenpunkte werden dann mit der `polyfit()`-Funktion interpoliert, in diesem Fall mit einem Polynom 5. Ordnung. Es muss auf Mehrdeutigkeiten aufgepasst werden, in diesem Fall gibt es keine Unstetigkeiten.

Im Nachhinein wäre es besser gewesen, die p_{EF}^i in die Inverse Kinematik zu geben und anschließend das Skript aus Vorlesung 18 mit diesen Werten auszuführen. Diese Vorgehensweise hätte zu einer saubereren Lösung geführt.

Zur Aufgabe korrespondierendes Matlab Skript: Aufgabe3_Trajektorienplanung.m und InverseKinematik02.m

4 Erstellen des Simulationsmodell in Simulink

Zunächst muss das System zweiter Ordnung in den Zustandsraum gebracht werden:

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{\mathbf{y}} \\ \mathbf{M}(\mathbf{y})^{-1}(\tau_R + \tau_M - \mathbf{D}(\mathbf{y}, \dot{\mathbf{y}}) \cdot \dot{\mathbf{y}} - \mathbf{g}(\mathbf{y})) \end{bmatrix}.$$

Anschließend kann, wie in Abbildung 2, in Simulink modelliert werden. Im Block **Roboter-Modell** sind die in Aufgabe 2 berechneten Matrizen hinterlegt, um die Bewegungsgleichung zu bilden. Dieser Block erzeugt den Output $\dot{\mathbf{x}}$, welcher anschließend integriert wird. Es ergibt sich \mathbf{x} , welcher anschließend wieder zurückgeführt wird, da die Systemmatrizen davon abhängen. Die blau hinterlegten Flächen müssen nicht betrachtet werden, da sie nur zur Parameterübergabe oder um Blöcke für das Postprocessing handelt.

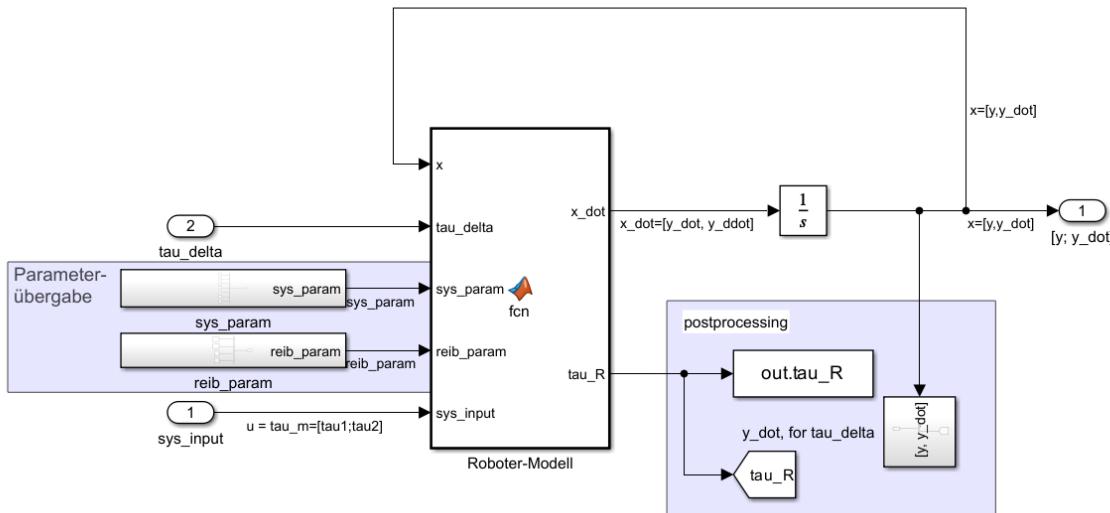


Abbildung 2: Modell des Knickarmroboters.

Zur Aufgabe korrespondierendes Matlab Skript: Sim_Knickarmroboter_AndreasPaul.slx

5 Implementieren der Regelung

Zunächst wird das System aufgebaut, nach dem in [Fehr24a] dargestellten Regelsystem. Es ergibt sich der in Abbildung 1 gezeigte Aufbau.

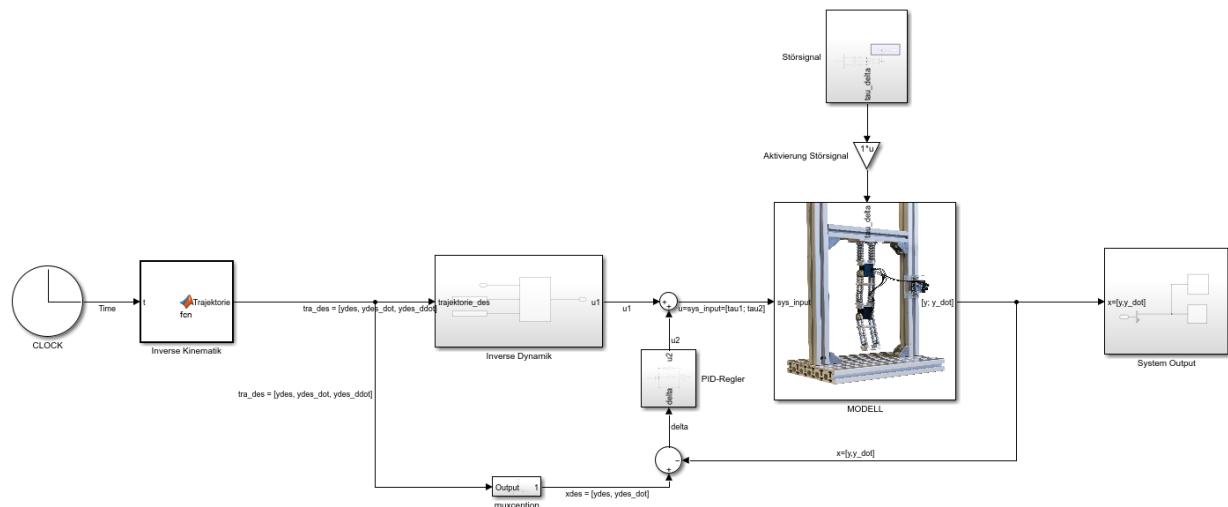


Abbildung 3: Simulink-Modell des geschlossenen Kreises des Knickarmroboters.

In den jeweiligen Blöcken werden die in den zuvor berechneten Systemmatrizen als Funktionen aufgerufen. Die Inverse Kinematik übergibt direkt die gewünschte Trajektorie im Raum der verallgemeinerten Koordinaten.

Das System wird über eine 2-DOF-Regelstruktur geregelt. Der Regler besteht aus einem Feedforward-Term, der eine Inverse Kinematik beinhaltet. Diese wird mit dem Modellieren der Aufgabe 2 durch Umstellen der Gleichung (1) berechnet.

$$\tau_M = \mathbf{M}(\mathbf{y}) \cdot \ddot{\mathbf{y}} + \mathbf{D}(\mathbf{y}, \dot{\mathbf{y}}) \cdot \dot{\mathbf{y}} + \mathbf{g}(\mathbf{y}) - \tau_R$$

Hier ist anzumerken, dass für die Inverse Dynamik und das Plant dasselbe Simulationsmodell verwendet wird. Dies hat zur Folge, dass für eine gewisse Wunschtrajektorie immer der perfekte Input gegeben wird. Somit ist das Implementieren eines weiteren Reglers unnötig. Hier kommt jedoch die Störung des Reibmoments dazu. Diese wird in der Inversen Dynamik nicht betrachtet und muss daher anders ausgeregelt werden. Dafür wird ein einfacher PID-Regler in Feedback-Schaltung angewendet. Die Faktoren des PID-Reglers werden zu Einheitsmatrizen entsprechender Ordnung gewählt und können anschließend verbessert werden.

Zur Aufgabe korresponierendes Matlab Skript: Sim_Knickarmroboter_AndreasPaul.slx

6 Modellierung des Störsignals

Die Störgröße soll auf den Reibterm wirken. Dieser ist nach Stribeck abhängig von der Rotationsgeschwindigkeit. Demnach wird hier die Störgröße als weißes Rauschen modelliert, welches mit einem zur Gelenkgeschwindigkeit linear proportionalen Faktor multipliziert wird. Der Aufbau in Simulink ist Abbildung 4 zu entnehmen.

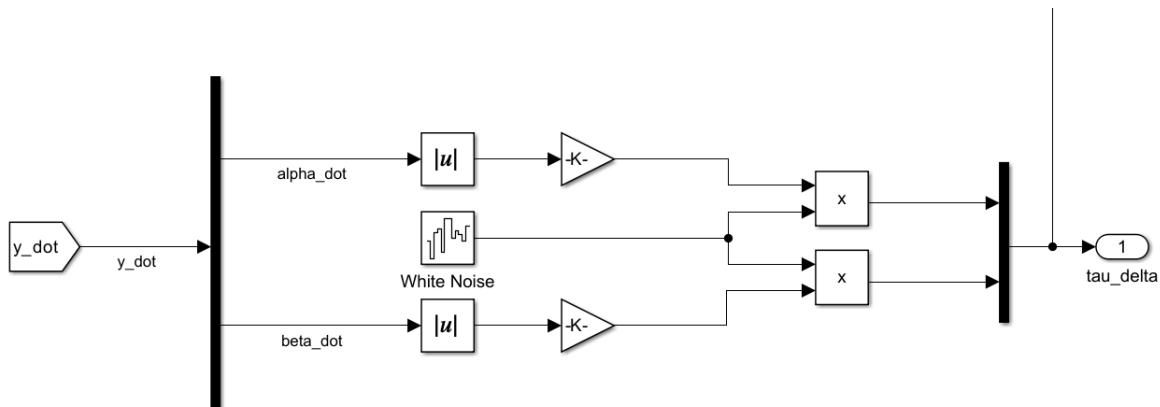


Abbildung 4: Simulink-Modell des geschlossenen Kreises des Knickarmroboters.

Zur Aufgabe korresponierendes Matlab Skript: Sim_Knickarmroboter_AndreasPaul.slx

7 Simulation und Visualisierung

In diesem Kapitel werden kurz die Simulationsergebnisse zusammengefasst. Abbildung 5 und Abbildung 6 zeigen jeweils die Wunschtrajektorie neben der tatsächlich gefahrenen. Es ist eine gute Übereinstimmung zu erkennen. Bei Betrachtung Abbildung 7 ist erkennbar, dass die Störgröße mehrere Größen Ordnungen über dem modellierten Reibmoment liegt. Der PID-Regler, ohne Tuning schafft es diesen jedoch erfolgreich auszuregeln.

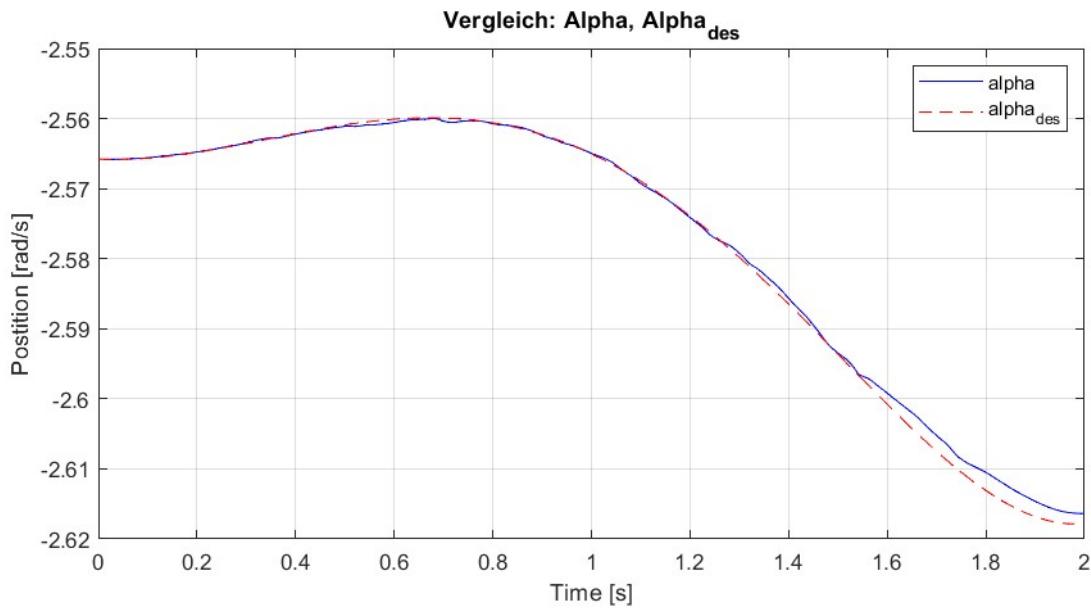


Abbildung 5: Vergleich: α und α_{des} .

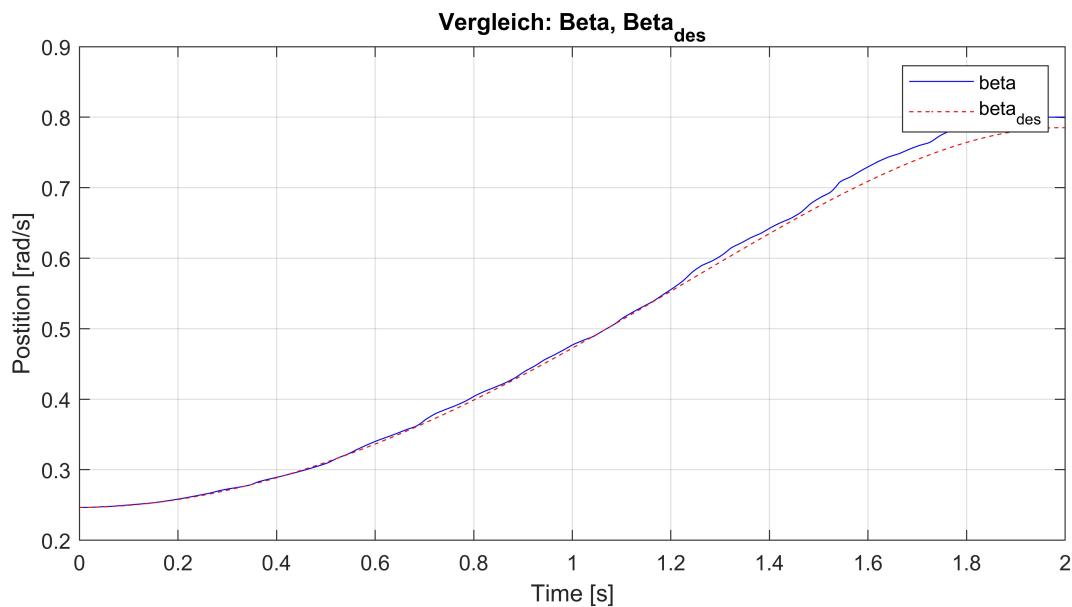


Abbildung 6: Vergleich: β und β_{des} .

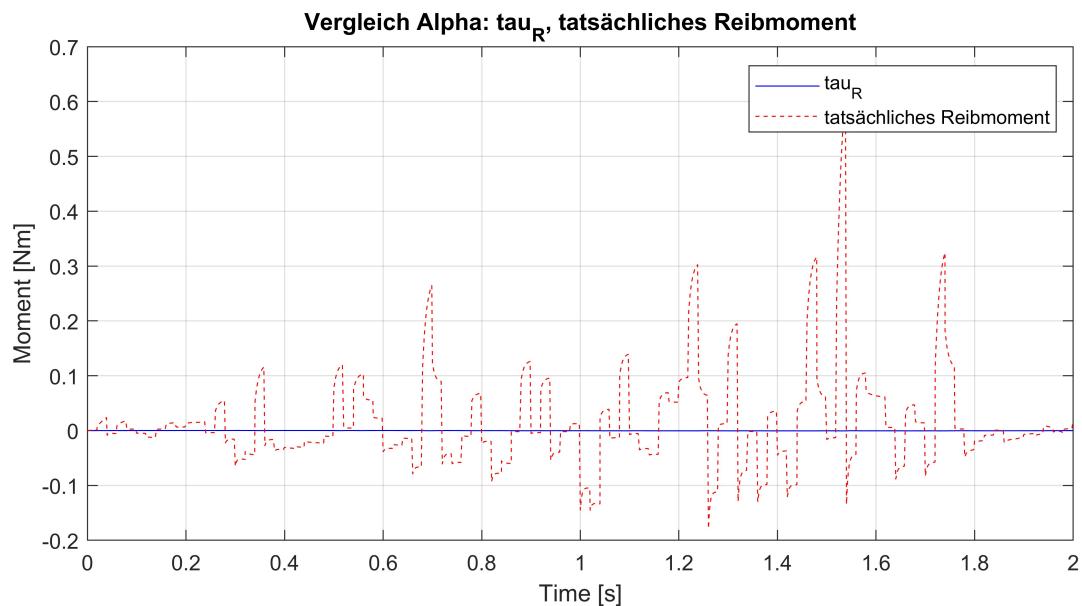


Abbildung 7: Vergleich: τ_R und $\tau_{R,Stör}$.

Zur Aufgabe korrespondierendes Matlab Skript: Postprocessing.m



Literatur

- [Fehr24a] Fehr, P.D.I.J.: Modellierung und Simulation in der Mechatronik, Vorlesung 18, 2024. Vorlesungsdruck, Institut für Technische und Numerische Mechanik, Universität Stuttgart.
- [Fehr24b] Fehr, P.D.I.J.: Modellierung und Simulation in der Mechatronik, Vorlesung 6, 2024. Vorlesungsdruck, Institut für Technische und Numerische Mechanik, Universität Stuttgart.
- [Fehr24c] Fehr, P.D.I.J.: Modellierung und Simulation in der Mechatronik, Vorlesung 9, 2024. Vorlesungsdruck, Institut für Technische und Numerische Mechanik, Universität Stuttgart.
- [Fuchs23] Fuchs, M.: Data-Driven Modeling of a Robotic Manipulator, 2023. Bachelorarbeit BSC-154, Institut für Technische und Numerische Mechanik, Universität Stuttgart.