Steuerung und Regelung eines Doppelpendels

Projektseminar von Tobias Gebhard und Frederik Tesar
Tag der Einreichung: 21. Oktober 2020

1. Gutachten: Prof. Dr.-Ing. Ulrich Konigorski
2. Gutachten: Dr.-Ing. Eric Lenz
Darmstadt

TECHNISCHE UNIVERSITÄT DARMSTADT

REGELUNGSTECHNIK UND MECHATRONIK rtm

Fachbereich Elektrotechnik und Informationstechnik

Steuerung und Regelung eines Doppelpendels

Projektseminar von Tobias Gebhard und Frederik Tesar

1. Gutachten: Prof. Dr.-Ing. Ulrich Konigorski
2. Gutachten: Dr.-Ing. Eric Lenz

Tag der Einreichung: 21. Oktober 2020

Darmstadt

Inhaltsverzeichnis

1 Einleitung

2 Modellierung

In diesem Kapitel wird die Modellierung des Gesamtsystems erläutert und auf dessen Implementierung in MATLAB und SIMULINK eingegangen.

2.1 Modell des Schlitten-Pendel-Systems

Die Modellierung des Schlitten-Pendel-Systems orientiert sich zunächst an den Modellen der vergangenen Arbeiten. Diese bezogen sich auf die Herleitung von [11]. Dabei gibt es die Variante *Kraftsystem*, das als Eingang die Kraft annimmt, welche am Schlitten wirkt, sowie das vereinfachte *Beschleunigungssystem*, das direkt die Beschleunigung des Schlittens als Eingang erhält.

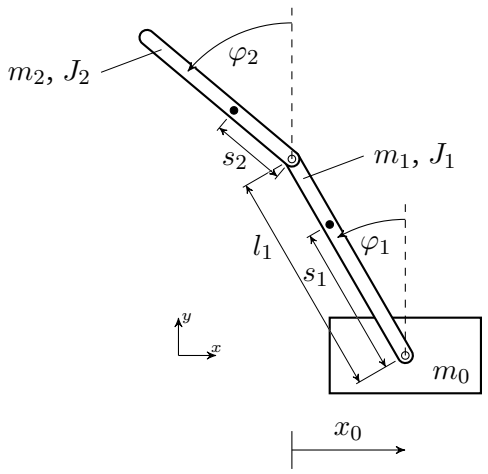


Abbildung 2.1: Doppelpendel

2.1.1 Koordinaten

Das System aus Schlitten und Doppelpendel hat 3 Freiheitsgrade, die mit den *Minimal-Koordinaten*

$$q_0 = x_0$$
$$q_1 = \varphi_1$$
$$q_2 = \varphi_2$$

beschrieben werden. Die Koordinaten sind nach Abbildung 2.1 definiert.

Die Schwerpunktskoordinaten der Pendelstäbe ergeben sich zu

$$
\begin{aligned}
x_1 &= x_0 - s_1 \sin\varphi_1 \\
y_1 &= s_1 \cos\varphi_1 \\
x_2 &= x_0 - l_1 \sin\varphi_1 - s_2 \sin\varphi_2 \\
y_2 &= l_1 \cos\varphi_1 + s_2 \cos\varphi_2 \ .
\end{aligned}
$$

2.1.2 Herleitung der Bewegungsgleichungen

Um auf die Bewegungsgleichungen des Systems zu gelangen, wird in [11] der *Lagrange*-Formalismus verwendet. Dazu werden zunächst die kinetische und potentielle Energie des Gesamtsystems bestimmt sowie die nicht-konservativen Kräfte/Momente.

Die kinetische Gesamtenergie ergibt sich zu

$$
T = \frac{1}{2} m_0 \dot{x}_0^2 + \frac{1}{2} m_1 (\dot{x}_1^2 + \dot{y}_1^2) + \frac{1}{2} m_2 (\dot{x}_2^2 + \dot{y}_2^2) + \frac{1}{2} J_1 \dot{\varphi}_1^2 + \frac{1}{2} J_2 \dot{\varphi}_2^2 \tag{2.1}
$$

und die potentielle Energie beträgt

$$
U = m_1 g y_1 + m_2 g y_2 \ . \tag{2.2}
$$

Die nicht-konservativen Kräfte/Momente setzen sich aus der Antriebskraft F am Schlitten (in x_0-Richtung) und den Reibungs-Kräften/Momenten zusammen und lauten

$$
Q_0^* = F + F_d + F_c \tag{2.3a}
$$

$$
Q_1^* = M_{d1} + M_{c1} - M_{d2} - M_{c2} \tag{2.3b}
$$

$$
Q_2^* = M_{d2} + M_{c2} \tag{2.3c}
$$

mit den viskosen Dämpfungen

$$
F_d = -d_0 \dot{x}_0 \tag{2.4a}
$$

$$
M_{d1} = -d_1 \dot{\varphi}_1 \tag{2.4b}
$$

$$
M_{d2} = -d_2 (\dot{\varphi}_2 - \dot{\varphi}_1) \ . \tag{2.4c}
$$

Mit dem Formalismus ergeben sich die 3 *gekoppelten* Bewegungsgleichungen für die Minimal-Koordinaten. Um nach den zweiten Ableitungen aufzulösen, muss allerdings noch das Gleichungssystem gelöst werden.

Damit ergeben sich die Bewegungsgleichungen als Funktion der Minimal-Koordinaten, der Systemparameter und des Eingangs.

$$
\begin{aligned}
\ddot{x}_0 &= f_{x_0}(\dots) \\
\ddot{\varphi}_1 &= f_{\varphi_1}(\dots) \\
\ddot{\varphi}_2 &= f_{\varphi_2}(\dots)
\end{aligned}
$$

Die Berechnung der Ableitungen und das Lösen des Gleichungssystems geschah bisher händisch, was im Allgemeinen fehleranfällig ist. In dieser Arbeit werden die Bewegungsgleichungen mithilfe der *symbolischen Toolbox* von MATLAB gelöst. Die für den *Lagrange*-Formalismus benötigten Ableitungen werden symbolisch berechnet und das Gleichungssystem mit dem symbolischen Solver gelöst und vereinfacht. Diese Vorgehensweise ist nicht nur weniger fehleranfällig, dadurch lässt sich das System auch sehr flexibel modifizieren.

2.1.3 Zustandsraummodell

Um das Schlitten-Pendel-System als Zustandsraummodell (mit ausschließlich ersten Ableitungen) darzustellen, wird folgender Zustandsvektor definiert:

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix} = \begin{bmatrix} x_0 \\ \dot{x}_0 \\ \varphi_1 \\ \dot{\varphi}_1 \\ \varphi_2 \\ \dot{\varphi}_2 \end{bmatrix} \tag{2.5}$$

Mit den Bewegungsgleichungen ergibt sich für das Kraftsystem (Eingangsgröße $u = F$) das folgende nichtlineare, eingangsaffine Zustandsraummodell:

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x},u) \tag{2.6}$$

$$= \boldsymbol{a}(\boldsymbol{x}) + \boldsymbol{b}(\boldsymbol{x}) \cdot F \tag{2.7}$$

$$\frac{\mathrm{d}}{\mathrm{d}t} \begin{bmatrix} x_0 \\ \dot{x}_0 \\ \varphi_1 \\ \dot{\varphi}_1 \\ \varphi_2 \\ \dot{\varphi}_2 \end{bmatrix} = \begin{bmatrix} f_1(\boldsymbol{x},u) \\ f_2(\boldsymbol{x},u) \\ f_3(\boldsymbol{x},u) \\ f_4(\boldsymbol{x},u) \\ f_5(\boldsymbol{x},u) \\ f_6(\boldsymbol{x},u) \end{bmatrix} = \begin{bmatrix} \dot{x}_0 \\ a_2(\boldsymbol{x}) \\ \dot{\varphi}_1 \\ a_4(\boldsymbol{x}) \\ \dot{\varphi}_2 \\ a_6(\boldsymbol{x}) \end{bmatrix} + \begin{bmatrix} 0 \\ b_2(\boldsymbol{x}) \\ 0 \\ b_4(\boldsymbol{x}) \\ 0 \\ b_6(\boldsymbol{x}) \end{bmatrix} \cdot F \tag{2.8}$$

Beim Beschleunigungssystem gilt für den Eingang $u = \ddot{x}_0 = a$, womit die Schlittenbeschleunigung direkt vorgegeben und die Dynamik des Schlittens umgangen wird. Damit wird die erste der drei gekoppelten Bewegungsgleichungen ersetzt. Das geänderte Gleichungssystem muss wieder gelöst werden und für das Zustandsraummodell folgt:

$$\frac{\mathrm{d}}{\mathrm{d}t} \begin{bmatrix} x_0 \\ \dot{x}_0 \\ \varphi_1 \\ \dot{\varphi}_1 \\ \varphi_2 \\ \dot{\varphi}_2 \end{bmatrix} = \begin{bmatrix} \dot{x}_0 \\ 0 \\ \dot{\varphi}_1 \\ a_4(\boldsymbol{x}) \\ \dot{\varphi}_2 \\ a_6(\boldsymbol{x}) \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ 0 \\ b_4(\boldsymbol{x}) \\ 0 \\ b_6(\boldsymbol{x}) \end{bmatrix} \cdot a \tag{2.9}$$

Tabelle 2.1 gibt Aufschluss über die Abhängigkeiten von Zuständen und Systemparametern. Grau hinterlegte Variablen sind nur im Kraftmodell vorhanden. Der Zustand x_0 hat keinen Einfluss auf die Systemdynamik.

Am Versuchsstand können die Schlittenposition und beide Pendelwinkel über Sensoren gemessen werden. Daher ergibt sich für den Ausgang beider Systeme

$$\boldsymbol{y} = \boldsymbol{h}(\boldsymbol{x}) = \mathbf{C}\boldsymbol{x} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \boldsymbol{x} = \begin{bmatrix} x_0 \\ \varphi_1 \\ \varphi_2 \end{bmatrix} . \tag{2.10}$$

Es handelt sich demnach um ein *SIMO-System* (single input multiple output).

Tabelle 2.1: Abhängigkeiten von Zuständen und Parametern

Zustände	$\dot{x}_0$, φ_1, $\dot{\varphi}_1$, φ_2, $\dot{\varphi}_2$
Trägheitsparameter	m_0, m_1, m_2, J_1, J_2
Geometrieparameter	l_1, s_1, s_2
Reibungsparameter	d_0, d_1, d_2, F_{c0}, M_{c10}, M_{c20}, $\dot{x}_{0,c76}$, $\dot{\varphi}_{1,c76}$, $\dot{\varphi}_{2,c76}$
Umgebungskonstanten	g

2.1.4 Coulomb-Reibung

Neben der einfach zu beschreibenden viskosen Dämpfung (2.4) gibt es in der Realität komplexere Arten von Reibung, die die Systembeschreibung erschweren. Eine Übersicht zu verschiedenen Reibungsmodellen ist in [13] gegeben.

Da die Coulomb-Reibung sowohl des Schlittens als auch der Pendelstäbe einen wesentlichen Einfluss zu haben scheint, darf diese nicht vernachlässigt werden. In den bisherigen Modellierungen wurde höchstens die Coulomb-Reibung des Schlittens berücksichtigt. Bei der Neukonstruktion ist jedoch in den Gelenken eine höhere Coulomb-Reibung vorhanden (siehe Abschnitt 2.3.3), weswegen diese ebenfalls modelliert wird. Die eigentlich vorhandene Haftreibung wird nicht modelliert.

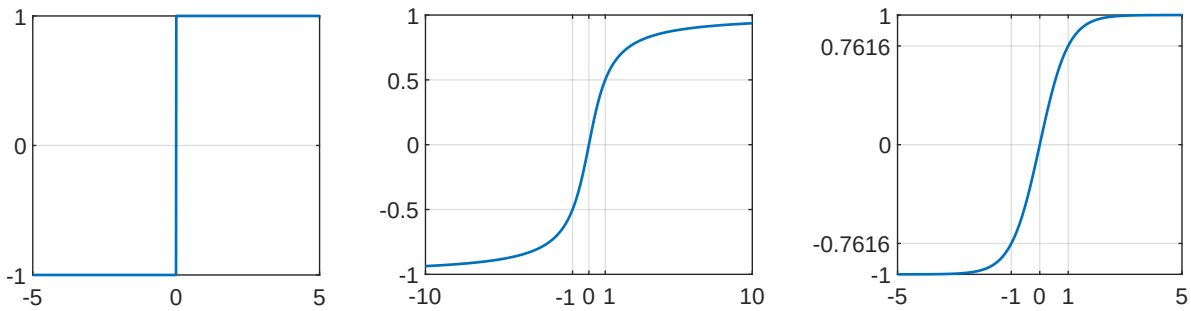
Die Formel der Gleitreibung lautet eigentlich

$$F_c = F_{c0} \cdot \operatorname{sign}(\dot{x}_0) \ ,$$

allerdings führt diese Implementierung aufgrund der signum-Funktion (siehe Abbildung 2.2a) zu Komplikationen in der Simulation. In der Nähe des Vorzeichenwechsels ist die Funktion unendlich steil, was bei Nulldurchgängen in der Simulation problematisch ist, weil die Schrittweite dadurch sehr klein werden muss. Zwar kann man in SIMULINK die Option „Zero Crossing Detection“ abschalten, es kommt dann allerdings meistens zu einem „Rattern“. [9]

In [14] und [8] wurde der Verlauf der signum-Funktion bei sehr niedrigen Geschwindigkeiten mit der arctan Funktion angenähert (siehe Abbildung 2.2b).

$$F_c = F_{c0} \cdot \frac{2}{\pi} \arctan\left(\frac{\dot{x}_0}{150}\right)$$



(a) $\operatorname{sign}(x)$ **(b)** $\frac{2}{\pi}\arctan(x)$ **(c)** $\tanh(x)$

Abbildung 2.2: Annäherung signum-Funktion – Vergleich

Die Funktion nähert sich jedoch nur sehr langsam dem Endwert an. In dieser Arbeit wird daher die $\tanh$ Funktion verwendet, welche deutlich früher den Endwert annimmt (siehe Abbildung 2.2c).

$$F_c = -F_{c0} \cdot \tanh\left(\frac{\dot{x}_0}{\dot{x}_{0,c76}}\right) \tag{2.11a}$$

$$M_{c1} = -M_{c10} \cdot \tanh\left(\frac{\dot{\varphi}_1}{\dot{\varphi}_{1,c76}}\right) \tag{2.11b}$$

$$M_{c2} = -M_{c20} \cdot \tanh\left(\frac{\dot{\varphi}_2 - \dot{\varphi}_1}{\dot{\varphi}_{2,c76}}\right) \tag{2.11c}$$

Dabei ist $\dot{x}_{0,c76}$ gerade die Geschwindigkeit, bei der $\tanh(1) = 0.7616$ von F_{c0} erreicht ist.

2.1.5 Ruhelagen

Die Ruhelagen bzw. Arbeitspunkte eines nichtlinearen Systems ergeben sich aus

$$\dot{\boldsymbol{x}}_R = \boldsymbol{f}(\boldsymbol{x}_R, u_R) = \boldsymbol{0}\ .$$

Für das Schlitten-Pendel-System gibt es aufgrund der Periodizität von Sinus und Kosinus theoretisch unendlich viele Ruhelagen. Es werden daher die folgenden 4 prinzipiell unterschiedlichen betrachtet:

Tabelle 2.2: Die 4 Ruhelagen

Arbeitspunkt	u	x_0	$\dot{x}_0$	φ_1	$\dot{\varphi}_1$	φ_2	$\dot{\varphi}_2$	Stabilität
1	0	x_{0R}	0	π	0	π	0	stabil
2	0	x_{0R}	0	π	0	0	0	instabil
3	0	x_{0R}	0	0	0	π	0	instabil
4	0	x_{0R}	0	0	0	0	0	instabil

Die Schlittenposition x_0 ist dabei beliebig, da der Schlitten an jeder Position eine Ruhelage annehmen kann. Die Reihenfolge der Arbeitspunkte entspricht den Pendelpositionen „von unten nach oben“.

2.1.6 Beschränkungen

Die Länge der Schlittenführung ist am realen Versuchsstand endlich, weswegen die Schlittenposition begrenzt ist. Es gilt für die Position des Schlittenmittelpunktes:

$$-0{,}8\,\mathrm{m} \leq x_0 \leq 0{,}8\,\mathrm{m}$$

2.2 Motor-Modell

Das Motormodell besteht aus den drei Baugruppen Spannungs-Strom-Wandler, Gleichstrommotor und Getriebe. Die Modellierung der Baugruppen basiert auf den in Franke [6] erstmalig aufgestellten Gleichungen, die auch in den nachfolgenden Arbeiten zum Versuchsstand zur Anwendung kamen. Das Modell des Gleichstrommotors wird im Rahmen dieser Arbeit nun zusätzlich um die Berücksichtigung der Gegeninduktion des Motors erweitert.

2.2.1 Spannungs-Strom-Wandler

Bei dem am Versuchsstand eingesetzten Spannungs-Strom-Wandler handelt es sich um einen Servoverstärker, der ursprünglich zur Drehzahlregelung von Gleichstrommotoren vorgesehen war. Entsprechend folgt der Verstärker dem Prinzip einer übergeordneten Drehzahlregelung mit unterlagerter Stromregelung. Für die Anwendung am Versuchsstand ist der übergeordnete Drehzahlregelkreis jedoch aufgetrennt und in einen Spannungs-Strom-Wandler umfunktioniert worden. Dieser dient zur Vorgabe eines konstanten Ankerstroms mittels Pulsweitenmodulation (PWM) der Zwischenkreisspannung des Wandlers proportional zur eingehenden Steuerspannung. Das stationäre Verhalten kann daher durch einen Proportionalitätsfaktor K_{UI} beschrieben werden.

$$I_a = K_{UI} \cdot U_{\text{Steuer}} \tag{2.12}$$

Gemäß Franke [6] lässt sich die Dynamik des Wandlers durch ein PT_1-Glied modellieren, sodass sich für den Wandler im Laplace-Bereich die Gleichung

$$I_a(s) = \frac{K_{UI}}{1 + T_{UI} \cdot s} \cdot U_{\text{Steuer}}(s) \tag{2.13}$$

ergibt.

2.2.2 Gleichstrommotor

Bei dem verwendeten Motor handelt es sich um eine fremderregte Gleichstrommaschine, wobei die magnetische Erregung durch einen Permanentmagneten erzeugt wird [6]. Das elektromagnetische Drehmoment des Gleichstrommotors ist näherungsweise proportional zum Ankerstrom [3]. Hierbei wird vorausgesetzt, dass keine magnetische Sättigung vorliegt.

$$M_e = K_I \cdot I_a \tag{2.14}$$

Neben dem elektromagnetischen Drehmoment sind außerdem parasitäre Reibmomente zu berücksichtigen. Die Modellierung der Reibung von Motor und Getriebe wird zusammen mit der Schlittenreibung in Abschnitt 2.1 behandelt. Ein rückwirkendes Moment durch die Federkraft des Riemens wird auf Grund der Annahme unendlicher Riemensteifigkeit vernachlässigt.

Weiterhin wird das Drehmoment durch die Gegeninduktion geschwächt. Dieser Effekt ist in den Motormodellen von Franke [6] und den Nachfolgearbeiten bisher nicht berücksichtigt worden. Da die

Erfahrung am realen Versuchsstand jedoch gezeigt hat, dass eine Modellierung der Gegeninduktion sinnvoll erscheint, wird diese im Rahmen dieser Arbeit in das Motormodell integriert.

Zum besseren Verständnis des Effekts wird zunächst das physikalische Prinzip der Gegeninduktion betrachtet. Fließt ein Strom durch den ruhenden Anker, der sich im Magnetfeld der Permanentmagneten befindet, so wirkt senkrecht zu den Richtungen des Stroms und des Magnetfelds die Lorenzkraft auf die in der Ankerwicklung befindlichen Ladungsträger (Drei-Finger-Regel bzw. Rechte-Hand-Regel in technischer Stromrichtung). Durch das entstehende Drehmoment beginnt der Anker zu rotieren. Auf Grund der Rotation bewegen sich die Ladungsträger nun zusätzlich zur eigentlichen Stromrichtung auch in Rotationsrichtung des Ankers. Auf diese Bewegungskomponente kann nun erneut das Prinzip der Lorentzkraft angewendet werden. Die resultierende zusätzliche Kraftkomponente, die in Abhängigkeit der Rotationsgeschwindigkeit auf die Ladungsträger wirkt, zeigt nun gegen die Stromrichtung (Lenz'sche Regel). Der resultierenden Ladungsbeschleunigung kann ein Spannungsabfall über der Ankerwicklung zugeordnet werden, die sogenannte induzierte Gegenspannung oder Gegen-EMK (Gegen-Elektromagnetische-Kraft). Der in Folge sinkende Ankerstrom reduziert das Drehmoment des Motors. [3]

Am Versuchsstand wird dieser Effekt bei geringen Winkelgeschwindigkeiten durch den Stromregler des Spannungs-Strom-Wandlers kompensiert. Ab einer bestimmten Winkelgeschwindigkeit bei konstantem Sollstrom wird die induzierte Gegenspannung größer als die maximale Zwischenkreisspannung des Spannungs-Strom-Wandlers, sodass nicht mehr ausgeregelt werden kann. Nun nimmt der Ankerstrom und damit das Drehmoment mit steigender Winkelgeschwindigkeit ab bis die Leerlaufdrehzahl erreicht ist.

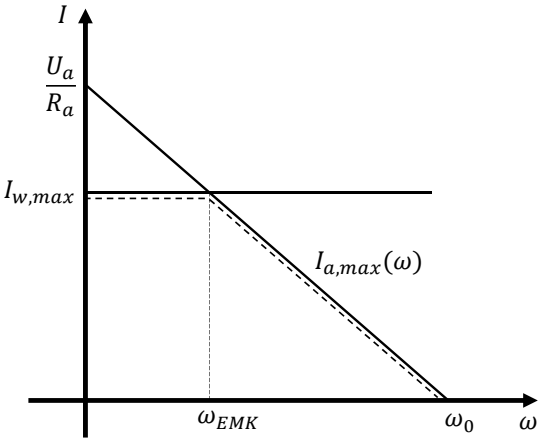


Abbildung 2.3: Stromkennlinie

Auf Grund der integrierten Strombegrenzung zum Schutz von Motor und Verstärker wird der vom Wandler bereitstellbare Strom zusätzlich begrenzt. Der dadurch parametrierte Maximalstrom des Wandlers wird durch den Stromregler zunächst konstant gehalten bis die Strombegrenzung der Gegen-EMK ab einer bestimmten Winkelgeschwindigkeit ω_{EMK} die des Wandler unterschreitet (siehe Abbildung 2.3). An dieser Stelle tritt ein „Knick" im Stromverlauf auf. Für den Stromregler wird dabei vereinfachend angenommen, über ausreichend hohe Dynamik und Stellenergie zu verfügen, um den Strom bis zur EMK-Grenze zu jedem Zeitpunkt konstant halten zu können.

Es werden zunächst nur positive Winkelgeschwindigkeiten betrachtet. Der Abschnitt des Verlaufs in

Abbildung 2.3 mit konstantem maximalen Strom wird durch

$$I_a = I_{w,\max} = \text{const.}\ , \qquad 0 \leq \omega \leq \omega_{\text{EMK}} \tag{2.15}$$

beschrieben, wobei $I_{w,\max}$ der Begrenzungsstrom des Wandlers ist.

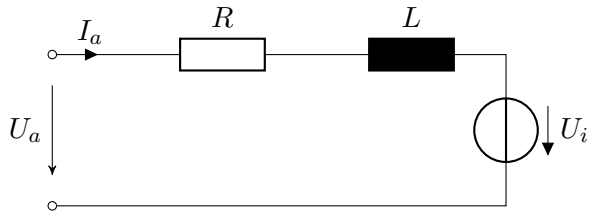


Abbildung 2.4: Ersatzschaltbild eines Gleichstrommotors

Der Verlauf des Bereichs mit zeitlich abnehmender maximaler Stromstärke wird aus dem Ersatzschaltbild des Gleichstrommotors in Abbildung 2.4 hergeleitet. Aus dem zweiten Kirchhoff'schen Gesetz (Maschenregel) ergibt sich

$$U_a = RI_a + L\frac{\mathrm{d}I_a(t)}{\mathrm{d}t} + U_i\ . \tag{2.16}$$

Für die Kennlinie des maximalen Stroms liegt der Tastgrad (engl.: *duty cylcle*) der pulsweitenmodulierten Ankerspannung bei 100%, sodass

$$U_{a,\max} = const.$$

gilt. Die elektrische Zeitkonstante des Motors, die mit den Angaben des Datenblatts [???] zu

$$\tau_{\text{el}} = \frac{L}{R} \approx 0{,}0023 \ll 1\ ,$$

berechnet werden kann, ist ausreichend klein, dass näherungsweise von stationärem Betrieb ausgegangen werden kann und für die selbstinduzierte Spannung

$$L\frac{\mathrm{d}I_a(t)}{\mathrm{d}t} \approx 0$$

gilt. Die induzierte Gegenspannung wird gemäß [3] durch Proportionalität zur Winkelgeschwindigkeit

$$U_i = K_I \cdot \omega$$

modelliert. Damit kann Gleichung (2.16) zur Geraden

$$I_{a,\max}(\omega) = \frac{U_a}{R} - \frac{K_I}{R}\omega\ , \qquad \omega > \omega_{\text{EMK}} \tag{2.17}$$

umgeformt werden. Die Beschreibung des Bereichs negativer Winkelgeschwindigkeiten kann aus den Gleichungen (2.15) und (2.17) durch eine Punktspiegelung des in Abbildung 2.3 dargestellten Verlaufs in den dritten Quadranten gewonnen werden.

Zusammenfassend ergibt sich für den Verlauf des maximal verfügbaren Stroms

$$I_{a,\max}(\omega) = \begin{cases} -\frac{U_a}{R} - \frac{K_I}{R}\omega, & \omega < -\omega_{\mathrm{EMK}} \\ I_{w,\max} \cdot \operatorname{sign}(\omega) & -\omega_{\mathrm{EMK}} \leq \omega \leq \omega_{\mathrm{EMK}} \\ \frac{U_a}{R} - \frac{K_I}{R}\omega, & \omega > \omega_{\mathrm{EMK}} \end{cases} .$$

2.2.3 Getriebe

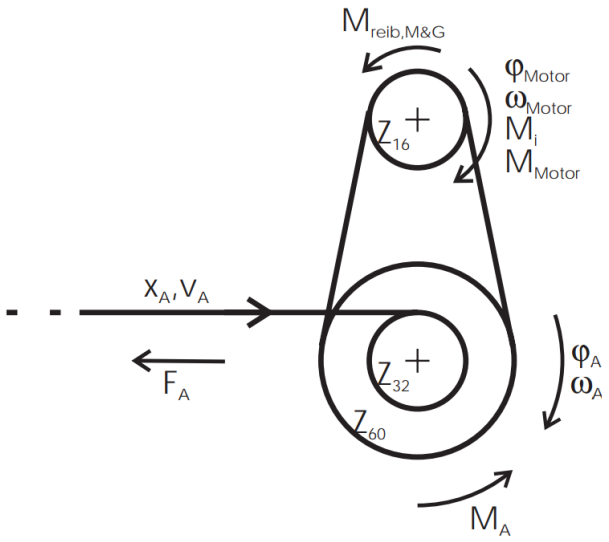


Abbildung 2.5: Getriebe [6]

Die Rotorbewegung des Motors wird über das in Abbildung 2.5 dargestellte Getriebe und das Antriebszahnrad auf den Zahnriemen weitergegeben. Auf Grund der hohen Steifigkeit des mit eingebetteten Stahlseilen unterstützten Riemens wird dieser wie in Apprich [2] als unendlich starr angenommen, sodass die Verbindung von Motor und Schlitten ohne Federkopplung durch eine Übersetzungskonstante modelliert werden kann.

$$\dot{x}_0 = K_G \cdot \omega$$

Die Übersetzungskonstante K_G zwischen der Winkelgeschwindigkeit des Motors und der Schlittengeschwindigkeit wird über das Zahnverhältnis und den Antriebsradius berechnet.

$$K_G = \frac{Z_{16}}{Z_{60}} \cdot r_{32} .$$

Der Antriebsradius r_{32} ist dabei der Abstand zwischen der neutralen Phase des Riemens und der Drehachse.

2.3 Modellparameter

Bevor die in dieser Arbeit verwendeten Parametersätze für Motor- und Schlitten-Pendel-Modell vorgestellt werden, soll im folgenden Abschnitt zunächst ein Überblick über die Entwicklung der in den vergangenen Arbeiten verwendeten Modellparameter gegeben werden.

2.3.1 Begründeter Stand der Modellparameter

Die für die Modellierung erforderlichen Systemparameter des Versuchsstands wurden erstmalig 1997 von Franke [6] durch Messungen identifiziert. Die Antriebseinheit aus Spannungs-Strom-Wandler, Motor, Getriebe und Riemen ist seitdem nicht verändert worden. Daher repräsentieren die von Franke [6] identifizierten Modellparameter in Bezug auf die Antriebseinheit weiterhin den aktuellen Stand (siehe Abschnitt 2.3.2)

Während zuvor noch ein Einfachpendel verwendet worden war, konstruierte Apprich [2] 2009 erstmalig ein Doppelpendel für den Versuchsstand. Von den Änderungen betroffen war neben den Pendeln auch die Schlittenmasse, da auch der obere Teil des Schlittens neu konstruiert wurde. Die Modellparameter für die neuen Pendelstäbe wurden, anders als bei Franke [6], nicht gemessen, sondern aus dem CAD-Modell abgeleitet. Dies betrifft die Massen, Trägheitsmomente, Längen und Schwerpunkte der beiden Stäbe. Die Masse des Schlittens wurde von Franke [6] übernommen. Durch Messungen wurden lediglich die viskose und trockene Reibung des Schlittens gegenüber den Schienen erneut identifiziert.

Im selben Jahr wurde von Kämmerer [7] die viskose Lagerreibung d_1 zwischen Stab 1 und dem Schlitten als fehlender Modellparameter durch Messungen ergänzt. Die viskose Lagerreibung d_2 zwischen Stab 1 und Stab 2 wurde rechnerisch bestimmt, da das Lager gegen Ende der Arbeit getauscht werden musste. Die viskose Dämpfung des Schlittens, die von Apprich [2] zuvor gemessen worden war, wurde durch einen deutlich höheren Schätzwert ersetzt. Außerdem wurde erstmalig die Masse von Schlitten und Antrieb zu einer schlittenseitig wirkenden Gesamtmasse zusammengefasst und ebenfalls als Schätzwert ausgewiesen.

Die Reibwerte d_1 und d_2 wurden 2011 durch Kisner [8] erneut bestimmt. Durch die *Prediction-Error Minimization Method* aus der *System Identification Toolbox* von MATLAB wurden die Parameter d_1 und d_2 so variiert, dass die quadratische Fehlersumme minimal wird. Als Fehler ist hierbei die Differenz zwischen den gemessenen und den vom Modell vorhergesagten zeitlichen Winkelverläufen $\varphi_1(t)$ und $\varphi_2(t)$ bei ruhendem Schlitten und frei gewählter Anfangsauslenkung zu verstehen. Für die Optimierung wurden zudem nicht näher beschriebene Anfangsschätzwerte für d_1 und d_2 gewählt. Es wird davon ausgegangen, dass es sich um Erfahrungswerte handelt, da sie einerseits nicht mit den zuletzt von Kämmerer [7] bestimmten Werten übereinstimmen, jedoch andererseits zu guten Ergebnissen am Versuchsstand führten. Statt der optimierten Werte wurden in den Nachfolgearbeiten die Anfangsschätzwerte weiterverwendet. Der Reibwert d_0 der viskosen Schlittenreibung, der nicht Gegenstand der Optimierung war, wurde mit einem deutlich höheren Wert als bei Kämmerer [7] angegeben. Da keine explizite Begründung vorliegt, wird von einem Erfahrungswert ausgegangen. Die zuvor von Kämmerer [7] geschätzte effektive Gesamtmasse von Schlitten und Antrieb wurde nach unten korrigiert, wobei die Hintergründe für diesen Schritt ebenfalls nicht bekannt sind. Der neue Wert ist jedoch plausibel und wird daher ebenfalls als erfahrungsbasierter Schätzwert verstanden. Der ist in den weiteren Arbeiten nicht mehr verändert worden, sodass er als aktueller Stand zu betrachten ist. Bei Kisner [8] wurde

erstmalig auch eine Begrenzung der Stellkraft von $F_{\max} = 400\,\mathrm{N}$ bezüglich des Schlittens angegeben, die als gegeben dokumentiert ist.

2011 wurde außerdem von Noupa [12] sowohl die viskose als auch die trockene Schlittenreibung gemessen, wobei besonders für die viskose Reibung eine hohe Richtungsabhängigkeit beobachtet wurde. Die gemessenen Werte wurden in den weiteren Arbeiten jedoch nicht weiter beachtet.

Auf Grund eines Austauschs des Lagers von Stab 2 wurde 2014 von Brehl [4] eine erneute Identifikation der viskosen Dämpfungskonstanten d_2 messungsbasiert durchgeführt. Dabei wurden auch Länge und Masse von Stab 2 gemessen, womit ebenfalls das Massenträgheitsmoment neu berechnet wurde. In den Nachfolgearbeiten wurden jedoch nur die Dämpfungskonstante weiterverwendet, während für Länge, Masse und Trägheitsmoment weiterhin die CAD-Werte von Apprich [2] verwendet wurden.

Chang [14] konstruierte im Sommersemester 2019 ein neues Doppelpendel, wobei Schlitten und Antrieb nicht verändert worden sind. Die neuen Modellparameter wurden wieder aus dem CAD-Modell abgeleitet. Die Angabe des Schwerpunkts s_2 von Stab 2 scheint in der Ausarbeitung jedoch zu fehlen. Die Dämpfungskonstanten d_1 und d_2 wurden von den Vorgängern übernommen, da bei der Konstruktion die gleichen Rillenkugellager gewählt wurden wie bei Apprich [2]. Für d_1 wurde der Schätzwert von Kisner [8] und für d_2 der Messwert von Brehl [4] übernommen, wobei in Changs Ausarbeitung versehentlich für beide Parameter Apprich [2] als Quelle zitiert wird. Die viskose und die trockene Reibung des Schlittens wurden durch Messung selbst bestimmt. Wie bei Noupa [12] wurde bei der viskosen Reibung eine auffällige Richtungsabhängigkeit festgestellt. Die Berücksichtigung der Richtungsabhängigkeit mit einer Vorsteuerung führte jedoch zu einer Verschlechterung des Systemverhaltens. Daher wurde schließlich die linksseitige Dämpfungskonstante für beide Seiten übernommen. Da das neu konstruierte Pendel noch nicht für die weiteren Bestandteile der Arbeit, wie die Auslegung der Regelung und deren Erprobung am Versuchsstand, zur Verfügung stand, wurde weiterhin auf die CAD-Werte von Apprich [2] zurückgegriffen. Der Betrag der maximalen Stellkraft $F_{\max}$ wurde zudem von dem von Kisner [8] zuletzt genannten Wert von $400\,\mathrm{N}$ auf $421\,\mathrm{N}$ erhöht. Der neue Wert lässt sich rechnerisch nachvollziehen, wie Abschnitt 2.3.2 entnommen werden kann.

Im Rahmen einer Verifikation der von Chang [14] angegebenen Modellparameter für das neue Doppelpendel, wurden im Wintersemester 2019/2020 durch Ribeiro [13] die Parameter aus dem CAD-Modell erneut abgeleitet. Da hierbei nicht genauer auf den Anlass der Neubestimmung eingegangen wurde, sich die neu bestimmten Parameter jedoch deutlich von den Vorherigen unterscheiden, wird angenommen, dass sich die von Chang [14] angegebenen Werte im Rahmen der Verifikation als fehlerbehaftet herausstellt hatten. Darüber hinaus wurde durch Messung die Reibung der Pendelgelenke identifiziert. Hierbei wurde neben der viskosen Reibung erstmalig auch die trockene Reibung in den Gelenken ermittelt. Die Parameter von Ribeiro [13] zu den Pendelstäben sind somit aktueller Stand. Übernommen wurde weiterhin die von Kisner [8] geschätzte Gesamtmasse mit Schlitten und Antrieb. Für die maximale Stellkraft wurde im Gegensatz zur Vorgängerarbeit Chang [14] statt $421\,\mathrm{N}$ wieder $400\,\mathrm{N}$ angenommen. Es wird vermutet, dass dadurch eine Stellkraftreserve für die Regelung vorgehalten werden sollte.

2.3.2 Parameter des Motor-Modells

Gemäß Abschnitt 2.3.1 werden die Modellparameter für das Motormodell Franke [6] entnommen.

Die Steuerspannung U_{Steuer} am Eingang des Spannungs-Strom-Wandlers darf maximal $\pm 10\,\mathrm{V}$ betragen. Ein Betrag größer als $10\,\mathrm{V}$ sollte vermieden werden, da sonst der Impulsstrom über den zulässigen Wert ansteigen und der Stromregler zerstört werden kann.

Die Verstärkung K_{UI} des Spannungs-Strom-Wandlers kann durch ein Potentiometer bis zu einem Wert von etwa $2\,\mathrm{A/V}$ eingestellt werden, sodass mit der maximalen Steuerspannung der maximal zulässige Impulsstrom von $20\,\mathrm{A}$ erreicht wird. Die zuletzt dokumentierte Einstellung liegt bei

$$K_{UI} = 1{,}87\,\frac{\mathrm{A}}{\mathrm{V}}\,.$$

Der Wert beinhaltet bereits eine Idealisierung, da Messungen von Franke [6] gezeigt haben, dass die reale Verstärkung am Versuchsstand eine leichte Richtungsabhängigkeit bezüglich des Vorzeichens aufweist.

Die Zeitkonstante T_{UI} wird als ausreichend klein angesehen, sodass die Dynamik des Wandlers vernachlässigt werden kann.

Alle weiteren verwendeten Parameter des Motormodells sind in Tabelle 2.3 aufgeführt.

Tabelle 2.3: Parameter – Motorsystem
Tabelle Dummy

Mit der statischen Verstärkung zwischen Eingang U_{Steuer} und Ausgang F

$$K_{\mathrm{MotorGain}} = K_{UI} \cdot K_I \cdot \frac{1}{K_G} \cdot \frac{1}{r_{32}} = 42{,}075\,\frac{\mathrm{N}}{\mathrm{V}} \tag{2.18}$$

lässt sich die maximale Stellkraft

$$F_{\max} = U_{\mathrm{Steuer,max}} \cdot K_{\mathrm{staticgain}} = 420{,}75\,\mathrm{N} \tag{2.19}$$

berechnen, wobei $K_{UI} = 1{,}87\,\mathrm{A/V}$ ist. Für $K_{UI,\max} = 2\,\mathrm{A/V}$ läge die maximale Stellkraft bei $F_{\max} = 450\,\mathrm{N}$.

2.3.3 Parameter des Schlittendoppelpendels

Ausgehend von Abschnitt 2.3.1 werden in dieser Arbeit drei Parametersätze angelegt (siehe Tabelle 2.4). Dies ermöglicht einen Vergleich der unterschiedlichen Systeme hinsichtlich des Systemverhaltens und der Stabilisierbarkeit.

Aufgrund der Neukonstruktion des Doppelpendels und dabei aufgetretener Schwierigkeiten in der Regelung ist ein Vergleich zum vorigen System von Interesse. Die Parameter der alten Konstruktion werden mit „Apprich09“ bezeichnet, obwohl hier auch neu bestimmte Werte von Kisner [8] und Brehl [4] (Reibung und Schlittenträgheit) enthalten sind. Der zweite Parametersatz „Chang19“ stellt die Daten der Ausarbeitung von [14] dar, in welcher die Neukonstruktion stattfand. Da diese erst in der nächsten Arbeit (Ribeiro [13]) in Betrieb genommen wurde und dort sowohl die CAD-Parameter erneut bestimmt als auch die Reibungsparameter identifiziert wurden, werden deren Werte als korrekte Parameter des neukonstruierten Doppelpendels betrachtet.

Es wird somit im Folgenden zwischen den „alten Parametern“ (`Apprich09`) und den „neuen Parametern“ (`Ribeiro20`) unterschieden. Der größte Unterschied zwischen beiden Systemen ist die Coulomb-Reibung in den Pendelgelenken, die zusätzlich zur viskosen Reibung erstmals bei Ribeiro [13] bestimmt wurde. Da bei der Neukonstruktion die Messsignalübertragung zur Vermeidung einer Kabelaufwickelung über

Tabelle 2.4: Parameter – Schlitten-Pendel-System

Bezeichnung	Symbol	Einheit	Apprich09	Chang19	Ribeiro20
Masse des Schlittens	m_0	kg	16,5	16,5	16,5
Masse des ersten Pendels	m_1	kg	0,615	0,329	0,8534
Masse des zweiten Pendels	m_2	kg	0,347	0,3075	0,3957
Trägheitsmoment des ersten Pendels	J_1	$\mathrm{kg\,m^2}$	0,00647	0,01457	0,01128
Trägheitsmoment des zweiten Pendels	J_2	$\mathrm{kg\,m^2}$	0,00407	0,00334	0,003343
Länge des ersten Pendels	l_1	m	0,2905	0,325	0,282
Länge des zweiten Pendels	l_2	m	0,3388	0,305	0,280
Schwerpunkt des ersten Pendels	s_1	m	0,0775	0,1425	0,09373
Schwerpunkt des zweiten Pendels	s_2	m	0,146	0,114254	0,114254
Viskose Dämpfung des Schlittens	d_0	$\frac{\mathrm{Ns}}{\mathrm{m}}$	17	17,6	17
Viskose Dämpfung des ersten Pendels	d_1	$\frac{\mathrm{Nms}}{\mathrm{rad}}$	0,0091	0,005	0,00768
Viskose Dämpfung des zweiten Pendels	d_2	$\frac{\mathrm{Nms}}{\mathrm{rad}}$	0,0006905	0,005	0,000285
Coulomb-Reibung des Schlittens	F_{c0}	N	16,232	17,5	13,43
Coulomb-Reibung des ersten Pendels	M_{c10}	Nm	0	0,0538	0,0538
Coulomb-Reibung des zweiten Pendels	M_{c20}	Nm	0	0,0000912	0,0000912
Erdbeschleunigung	g	$\frac{\mathrm{m}}{\mathrm{s^2}}$	9,81	9,81	9,81

einen Schleifring realisiert wurde, besteht die Vermutung, dass dieser für die erhöhte Coulomb-Reibung verantwortlich ist. Dies macht das System bereits um einen sehr kleinen Arbeitsbereich nichtlinear, was die Regelung erschwert.

Für die Skalierungsparameter $\dot{x}_{0,c76}$, $\dot{\varphi}_{1,c76}$ und $\dot{\varphi}_{2,c76}$ der Coulomb-Reibung (siehe Abschnitt 2.1.4) wird jeweils 0,01 angenommen.

2.4 Aufbau des Simulationsmodells

In dieser Arbeit wird das Doppelpendel-System rein simulativ betrachtet. Das Simulationsmodell wird im Vergleich zu Vorgängerarbeiten wesentlich systematischer und detaillierter aufgebaut. Dadurch sind umfangreiche und automatisierte Tests möglich, mit denen das System umfassend untersucht werden kann.

Besonders wird in dieser Arbeit Wert auf einen strukturierten, modularen Aufbau gelegt. Dies soll flexible Änderungen am System, wie z. B. Systemparameter oder Reglerparameter ermöglichen. Von der Berechnung der Bewegungsgleichungen über die Parametrisierung des Systems bis zur Reglerberechnung kann das Modell vollständig und konsistent initialisiert werden. Außerdem soll dadurch die Nutzbarkeit und Wiederverwendbarkeit in zukünftigen Arbeiten gewährleistet sein.

Das Simulationsmodell voriger Arbeiten wie [14] ist nur wenig strukturiert aufgebaut, enthält Redundanzen und viele fest-kodierte Parameter. Es wird daher lediglich zur Orientierung verwendet. Hart-kodierte Parameter werden in dieser Arbeit vermieden, sodass jeder Parameter bei der Initialisierung geändert werden kann (ohne das Simulink-Modell ändern zu müssen). Auch Gleichungen können variabel ausgelegt werden, was durch die symbolische Schreibweise von Matlab ermöglicht wird. Code-Dopplungen

werden vermieden. Die gesamte Initialisierung ist stark funktionalisiert. Dadurch wird sichergestellt, dass Änderungen nur an einer Stelle vorgenommen werden müssen und automatisch direkt an allen entsprechenden Stellen angepasst werden.

Die SIMULINK-Modelle und MATLAB-Funktionen zur Modellierung des Systems und Initialisierung des Simulationsmodells befinden sich im Ordner `Modell`.

2.4.1 Aufbau in SIMULINK

Subsysteme

In SIMULINK lassen sich sogenannte *Subsysteme* erstellen, um ein Simulationsmodell übersichtlicher zu gestalten. Diese Subsysteme (im Folgenden auch Module genannt) lassen sich zudem als Datei externalisieren (*Referenced Subsystem*), wodurch sie einerseits seperat bearbeitet werden können und andererseits an mehreren Stellen modular wiederverwendet werden können. Somit wird lediglich in der obersten Schicht (die Testebene) ein SIMULINK-*Model* verwendet.

In dieser Arbeit werden immer Module erstellt, sofern es sinnvoll erscheint. Dadurch wird das Gesamtsystem übersichtlich gehalten und kann sehr flexibel modifiziert werden. Die einzelnen Module (wie z. B. Motor, Gesamtmodell, Zustandsregler) stellen zudem wiederverwendbare Teile dar und können so an verschiedenen Stellen im Projekt eingebaut werden, aber möglicherweise auch am SIMULINK-Modell des Versuchsstandes zum Einsatz kommen. So könnte relativ einfach derselbe Regler an dem echten und dem simulierten System eingesetzt und verglichen werden.

Maske

Bei Subsystemen kann eine sogenannte *Maske* eingerichtet werden, wodurch Parameter übergeben werden können. In deren Abhängigkeit können in einem Subsystem auch initialisierende Berechnungen durchgeführt werden.

Grundsätzlich können in SIMULINK alle Variablen aus dem globalen MATLAB-Workspace verwendet werden, was allerdings fehleranfällig ist. Geschickter ist es, dem Simulationsmodell eine einzige Variable vom Typ `struct` zu übergeben, in der alle Parameter vorhanden sind. So werden jedem Modul über die jeweilige Maske nur genau die Parameter übergeben, von denen es abhängt. Dadurch bleibt der Gesamtaufbau modularer und strukturierter.

Scopes und ToWorkspace

Um die Signalverläufe direkt zu untersuchen, werden an allen wichtigen Stellen in den Simulationsmodellen *Scopes* installiert. So lässt sich das Systemverhalten oder Probleme in der Regelung schnell und einfach analysieren. Oft werden Scopes mit mehreren Eingängen verwendet, um Signale besser vergleichen zu können.

Für alle Variablen, die später in MATLAB für weitere Auswertungen zur Verfügung stehen sollen, werden *ToWorkspace*-Blöcke verwendet. Diese geben die Daten an die Ausgabevariable `out`.

Für die Screenshots werden diese Blöcke meist aus Übersichtsgründen entfernt.

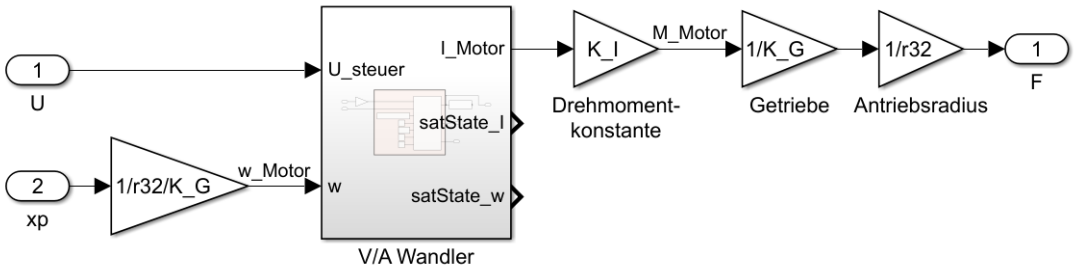


Abbildung 2.6: Motormodell in SIMULINK

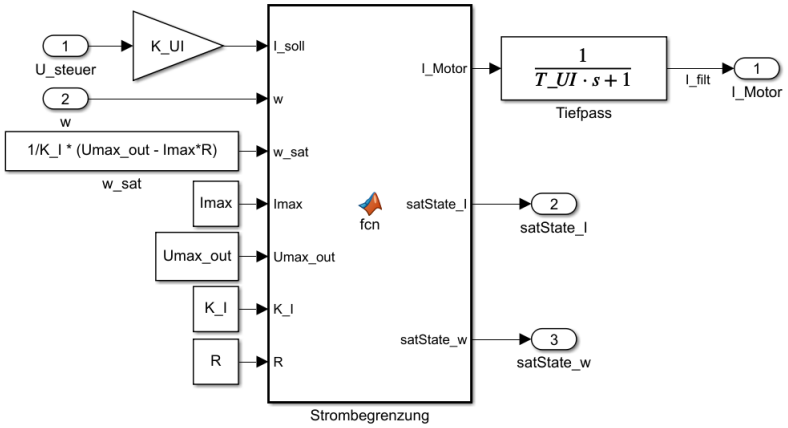


Abbildung 2.7: V/A Wandler in SIMULINK

2.4.2 SIMULINK Module und Modelle

Motor.slx

Das Motormodell wird Abschnitt 2.2 entsprechend als Modul implementiert (siehe Abbildung 2.6). Eingangsgröße ist die Steuerspannung sowie die Schlittengeschwindigkeit, da sich daraus die Induktionsspannung ergibt. Ausgangsgröße ist die Kraft, die am Schlitten wirkt. Der V/A Wandler wird als Subsystem implementiert und gibt den Motorstrom aus (siehe Abbildung 2.7). Die Berechnung der Strombegrenzung (siehe Abschnitt 2.2.2) wird als *matlabFunction* realisiert. Dabei werden die Signale `satState_w` (falls die Sättigungsdrehzahl erreicht ist) und `satState_I` (falls der Sollstrom nicht erreicht werden kann) gesetzt und können später analysiert werden. Die Tiefpass-Charakteristik der Induktivität wird mit dem PT_1-Glied (2.13) dargestellt.

SchlittenPendel.slx

Das Zustandsraummodell des Schlitten-Pendel-Systems wird mit einer *S-Function* dargestellt. Die zugehörige MATLAB-Datei `SchlittenPendelFunc.m` wird automatisch bei der Initialisierung erstellt (siehe Abschnitt 2.4.4). Lediglich die Anfangswerte werden über die Simulationsparameter übergeben.

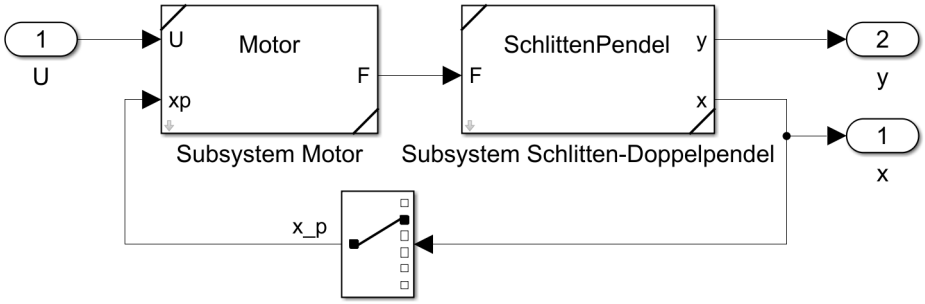


Abbildung 2.8: Gesamtmodell in SIMULINK

Gesamtmodell.slx

Dieses Modul stellt das gesamte Modell dar und fasst die beiden vorigen Module zusammen (siehe Abbildung 2.8). Die Eingangsspannung wird an den Motor gegeben und dessen Ausgang F ist die Schnittstelle zum Schlitten-Doppelpendel. Die Schlittengeschwindigkeit wird zum Motor zurückgeführt, da durch diese die Motordrehzahl bestimmt wird.

Gesamtmodell_Test.slx

Dies ist ein SIMULINK-Modell und bindet das Gesamtmodell-Modul ein. Auf dieses können testweise verschiedene Eingangsverläufe gegeben werden, um das Systemverhalten auf Plausibilität zu überprüfen, beispielsweise:

- Keine Spannung, aber Anregung durch Anfangsauslenkung
- Konstante Spannung
- Sinusförmige Spannung

Da die durchgeführten Tests am Modell plausible Ergebnisse zeigen, wird im Folgenden von der Korrektheit des Simulationsmodells ausgegangen.

2.4.3 Parameter

Größere zusammenhängende Parameterdaten werden als Datei bzw. Funktion ausgelagert, um eine einfache Austauschbarkeit zu erreichen. Die drei Parametersätze des Schlitten-Pendels (siehe Tabelle 2.4) sowie die Motorparameter befinden sich im Ordner `Modellparameter`.

2.4.4 Initialisierung

`Init.m`

2.4.5 Auswertung

Nach einer Simulation ist es nützlich, die wesentlichen Ergebnisse direkt ablesen zu können (z. B. ob der Schlitten außerhalb der Begrenzung war oder ob der Motor in Sättigung war). Außerdem wird eine Plot-Funktion implementiert, die die wichtigsten Verläufe darstellt. Um das Verhalten des Doppelpendels besser interpretieren zu können, wird zudem eine Animationsfunktion erstellt, die direkt das Pendelverhalten visualisiert. Die entsprechenden MATLAB-Funktionen sind im Ordner `Auswertung` vorhanden.

plot

animate

2.4.6 Weitere MATLAB-Funktionen

`SchlittenPendelNLZSR.m`

`SchlittenPendelSymA.m`

`SchlittenPendelSymF.m`

`SchlittenPendelRuhelagen.m`

`SchlittenPendelFunc.m`

3 Arbeitspunkt-Regelung

Nachdem im letzten Kapitel die Modellierung des Gesamtsystems erläutert wurde, stellt sich nun die Frage, wie sich das System regeln lässt. Dabei geht es einerseits um die Regelung an den 4 Arbeitspunkten (siehe Abschnitt 2.1.5), d. h. das Stabilisieren und Halten dieser, sowie um Trajektorien, die das Doppelpendel von einem Arbeitspunkt in einen anderen überführen (siehe folgendes Kapitel 4).

Alle Arbeitspunkt-Regelungen vergangener Arbeiten basieren auf einer Linearisierung und der anschließenden Auslegung eines *Riccati*-Reglers. Die Güteparameter wurden dabei jedes Mal anders und meist heuristisch bestimmt. Da am Versuchsstand nicht alle Zustände gemessen werden können, wurden meistens Beobachter mit Polplatzierung ausgelegt. Kämmerer [7] verglich diesen mit einem Zustandsvariablenfilter, mit welchem allerdings keine zufriedenstellende Ergebnisse gelangen. Apprich [2] und Kämmerer [7] setzten Störgrößenbeobachter ein, die in den folgenden Arbeiten nicht mehr verwendet wurden. Alle Arbeiten setzten Vorsteuerungsmethoden ein, da die Coulomb-Reibung am Schlitten bei der Regelung stets problematisch war. Chang [14] erstellte ein Simulationsmodell zur Arbeitspunkt-Regelung, das als Orientierung dient.

Beim Reglerentwurf wird zunächst dem Vorgehen vergangener Arbeiten gefolgt. Einzelne Aspekte in der Steuerung/Regelung werden anschließend verbessert. Die Reglerparameter der vorigen Arbeiten werden überprüft und optimiert. Außerdem wird der Einfluss des Beobachter anlysiert.

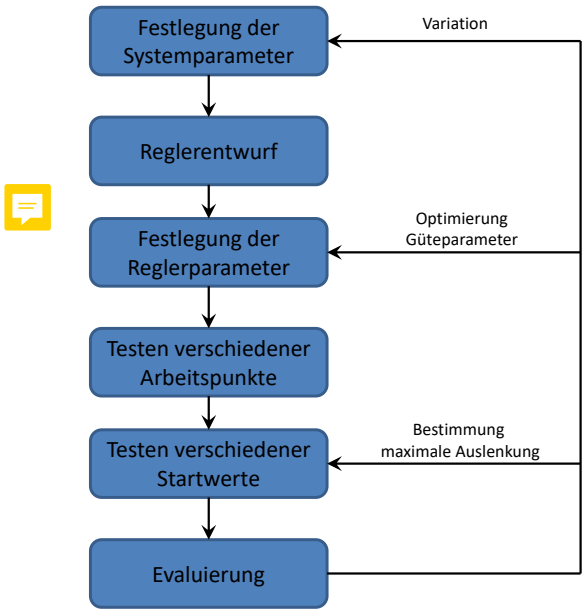


Abbildung 3.1: Vorgehen bei der AP-Regelung

Abbildung 3.1 stellt das prinzipielle Vorgehen bei der Untersuchung und Optimierung der Arbeitspunkt-Regelung dar. Es werden zunächst Auswertungsmethoden implementiert, die einzelne Simulationstests evaluieren. Mit den Anfangswerttests wird das Ermitteln der maximalen Winkelabweichung, ab welcher das System nicht mehr stabilisiert werden kann, automatisiert. Dadurch können im nächsten Schritt die Güteparameter systematisch optimiert werden. Abschließend wird die Regelbarkeit in Abhängigkeit der Systemparameter untersucht.

Regler können nach verschiedenen Gesichtspunkten wie Dynamik, Überschwingen, Energieverbrauch optimiert werden. In dieser Arbeit wird bei der Untersuchung und Optimierung des Doppelpendels stets die Stabilität priorisiert. Die Optimierung des Einschwingvorgangs steht nicht im Vordergrund. Daher werden vor allem die instabilen Arbeitspunkte 2, 3 und 4 untersucht. Außerdem werden hauptsächlichen die maximalen Winkelabweichungen betrachtet, da diese wesentlich für die Stabiltät sind. Horizontale Fahrten in x-Richtung sind zwar möglich, eventuell müssen dafür aber andere Reglerparameter eingesetzt werden.

3.1 System-Linearisierung und Analyse

Das nicht-lineare System wird zunächst linearisiert, dessen Eigenwerte analysiert, sowie auf Steuer- und Beobachtbarkeit überprüft.

3.1.1 Linearisierung

Da das Zustandsraummodell des Schlitten-Pendel-Systems (2.7) nicht-linear ist, muss es um jeden Arbeitspunkt linearisiert werden, bevor lineare Regelungsmethoden angewandt werden können.

$$\mathbf{A} = \left.\frac{\partial \boldsymbol{f}(\boldsymbol{x},u_R)}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}=\boldsymbol{x}_R} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & \frac{\partial a_2(\boldsymbol{x})}{\partial \dot{x}_0} & \frac{\partial a_2(\boldsymbol{x})}{\partial \varphi_1} & \frac{\partial a_2(\boldsymbol{x})}{\partial \dot{\varphi}_1} & \frac{\partial a_2(\boldsymbol{x})}{\partial \varphi_2} & \frac{\partial a_2(\boldsymbol{x})}{\partial \dot{\varphi}_2} \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & \frac{\partial a_4(\boldsymbol{x})}{\partial \dot{x}_0} & \frac{\partial a_4(\boldsymbol{x})}{\partial \varphi_1} & \frac{\partial a_4(\boldsymbol{x})}{\partial \dot{\varphi}_1} & \frac{\partial a_4(\boldsymbol{x})}{\partial \varphi_2} & \frac{\partial a_4(\boldsymbol{x})}{\partial \dot{\varphi}_2} \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & \frac{\partial a_6(\boldsymbol{x})}{\partial \dot{x}_0} & \frac{\partial a_6(\boldsymbol{x})}{\partial \varphi_1} & \frac{\partial a_6(\boldsymbol{x})}{\partial \dot{\varphi}_1} & \frac{\partial a_6(\boldsymbol{x})}{\partial \varphi_2} & \frac{\partial a_6(\boldsymbol{x})}{\partial \dot{\varphi}_2} \end{bmatrix}_{\boldsymbol{x}=\boldsymbol{x}_R} \tag{3.1}$$

$$\mathbf{B} = \left.\frac{\partial \boldsymbol{f}(\boldsymbol{x}_R,u)}{\partial u}\right|_{u=u_R} = \boldsymbol{b}(\boldsymbol{x}_R) = \begin{bmatrix} 0 \\ b_2(\boldsymbol{x}_R) \\ 0 \\ b_4(\boldsymbol{x}_R) \\ 0 \\ b_6(\boldsymbol{x}_R) \end{bmatrix} \tag{3.2}$$

Somit ergeben sich für jeden Arbeitspunkt unterschiedliche Systemmatrizen $\mathbf{A}$ und Eingangsmatrizen $\boldsymbol{B}$. Beim Beschleunigungssystem fällt zusätzlich die zweite Zeile und die zweite Spalte (bis auf die 1) weg, da dort $a_2 = 0$ ist und keine Abhängigkeit von $\dot{x}_0$ besteht (siehe Tabelle 2.1), außerdem ist $b_2 = 1$. Die Ausgangsmatrix $\mathbf{C}$ ist immer gleich, da die Ausgangsgleichung (2.10) linear ist.

3.1.2 Eigenwerte

Anhand der Eigenwerte können Aussagen über die Dynamik eines Systems getroffen werden. Das Beschleunigungssystem hat aufgrund der zweifachen Integration des Eingangs $\ddot{x}_0$ stets zwei Eigenwerte in 0. Mit den Apprich-Parametern ergeben sich folgende Eigenwerte:

Tabelle 3.1: Eigenwerte des Beschleunigungssystems (Parameter: Apprich)

AP1	AP2	AP3	AP4
0	0	0	0
0	0	0	0
$-0.2972 + 11.4177\text{j}$	7.8028	6.8596	11.1308
$-0.2972 - 11.4177\text{j}$	-7.9367	-7.2314	-11.7251
$-0.0418 + 4.8515\text{j}$	$-0.1858 + 7.0392\text{j}$	$-0.0669 + 7.8676\text{j}$	4.8089
$-0.0418 - 4.8515\text{j}$	$-0.1858 - 7.0392\text{j}$	$-0.0669 - 7.8676\text{j}$	-4.8926

Im Arbeitspunkt 1 ergeben sich zwei konjugiert-komplexe Polpaare, die sich physikalisch mit der Schwingung der beiden Pendel begründen lassen. Sie befinden sich aufgrund der Dämpfung in der linken s-Halbebene, es ist der einzige stabile Arbeitspunkt. Arbeitspunkt 2 und 3 besitzen jeweils ein konjugiert-komplexes Polpaar, da dort das erste bzw. zweite Pendel nach unten zeigt und damit „stabil“ ist. Der Realteil ist bei Arbeitspunkt 3 deutlich kleiner, was auf die geringere Dämpfung des zweiten Pendelgelenks zurückzuführen ist (siehe Tabelle 2.4). Aufgrund der Instabilität des anderen Pendels ergibt sich jeweils ein positiv reeller Eigenwert. In Arbeitspunkt 4 stehen beide Pendel oben, weswegen es dort zwei Eigenwerte in der rechten s-Halbebene gibt.

Tabelle 3.2: Eigenwerte des Beschleunigungssystems (Parameter: Ribeiro)

AP1	AP2	AP3	AP4
0	0	0	0
0	0	0	0
-174.3727	-172.9211	-173.2473	-175.4362
-0.3493	-0.3494	0.3471	0.3471
$-0.6385 + 7.1578\text{j}$	6.7715	$-0.6443 + 7.2040\text{j}$	6.7469
$-0.6385 - 7.1578\text{j}$	-7.6900	$-0.6443 - 7.2040\text{j}$	-7.6570

Mit den neuen Parametern (Ribeiro) ergibt sich ein anderes Bild (siehe Tabelle 3.2). Bei Arbeitspunkt 1 und 2 verschwindet ein Polpaar, welches dem ersten Pendel zugeordnet wurde. Ein sehr schneller Eigenwert kommt hinzu, außerdem ein langsamer reeller, der bei Arbeitspunkt 3 und 4 positiv ist. Die Ursache für diese Änderung liegt vermutlich in der Modellierung der Coulomb-Reibung (siehe Abschnitt 2.1.4). Bei der Linearisierung um die Ruhelage wird die Annäherungsfunktion der signum-Funktion um 0 linearisiert, wo sie sehr steil ist. Dies resultiert in einer sehr hohen Dämpfung. Daher ist es sinnvoll, die Coulomb-Reibung der Pendelstäbe für den AP-Reglerentwurf zu vernachlässigen. Wird die Coulomb-Reibung zu 0 gesetzt, ergeben sich prinzipiell ähnliche Werte wie in Tabelle 3.1.

3.1.3 Steuerbarkeit und Beobachtbarkeit

Voraussetzung für eine Regelung ist die Steuerbarkeit des Systems. Diese kann für lineare Systeme oder linearisierte Systeme an einem Arbeitspunkt mit dem Steuerbarkeitskriterium nach KALMAN überprüft werden [1]. Außerdem sollte ein System beobachtbar sein, falls nicht alle Zustände bekannt sind und daher über einen Beobachter ermittelt werden.

Das Schlitten-Pendel-System (Beschleunigungssystem) ist an allen 4 Arbeitspunkten vollständig steuer- und beobachtbar.

3.2 Aufbau der Regelung

Die Aufgabe eines Reglers bzw. einer (Vor-)Steuerung ist es nun, anhand der Messgrößen $\boldsymbol{y} = (x_0\, \varphi_1\, \varphi_2)^{\mathrm{T}}$ (2.10) eine geeignete Steuerspannung U_{Steuer} an den Motor vorzugeben. Dabei teilt sich die Steuerung/Regelung in mehrere Schritte auf.

Die eigentliche Regelung bildet ein Zustandsregler, welcher entweder am Kraftsystem oder am Beschleunigungssystem ausgelegt ist. Dessen Stellgröße kann im ersten Fall (F_{soll}) direkt an die Motorvorsteuerung gegeben werden. Beim Beschleunigungssystem muss die Ausgangsstellgröße a_{soll} in einem weiteren Block zu einer Sollkraft bestimmt werden. Diese Aufteilung in einen Arbeitspunktregler am einfacheren Beschleunigungssystem und eine unterlagerte Geschwindigkeitsregelung, um der störenden Reibung am Schlitten zu begegnen, hat sich in vergangenen Arbeiten als sinnvoll herausgestellt.

3.2.1 Zustandsregler

Die Arbeitspunkt-Regelung wird mit einem Zustandsregler realisiert ($u = -\mathbf{K}\hat{\boldsymbol{x}}$). Dieser regelt ein System grundsätzlich in den Ursprung, es muss also noch eine Zustandstransformation durchgeführt werden. Für jeden der Arbeitspunkte (2.1.5) wird der Zustand, der für den Zustandsregler die Endlage darstellt, vorher von den Messgrößen abgezogen. Nach Berechnung der Stellgröße und Schätzung des Zustands wird der Arbeitspunkt wieder dazu addiert.

Die Verstärkung $\mathbf{K}$ wird durch Lösen des LQ-Problems bestimmt [1]. Parameter für den Entwurf sind somit die positiv-definite Matrix $\mathbf{Q} = \mathrm{diag}\begin{pmatrix} q_1 & q_2 & q_3 & q_4 & q_5 & q_6 \end{pmatrix}$ zur Bewertung der Zustände und R für die Stellgröße. Da es sich jeweils um ein *linear-zeitinvariantes System* (LZI) handelt, ist die Matrix $\mathbf{P}$ aus der *Riccati*-Gleichung, und damit auch $\mathbf{K}$, konstant.

Für jeden Arbeitspunkt muss ein eigener Regler entworfen werden, da jedem Arbeitspunkt ein anderes lineares System zugrunde liegt (siehe Abschnitt 3.1.1). Der AP-Regler ist somit nur in der Umgebung der Ruhelage gültig und kann zu große Abweichungen aufgrund der Nichtlinearität möglicherweise nicht ausregeln. Außerdem sind für jeden Arbeitspunkt andere Güteparameter $\mathbf{Q}$ und R optimal. Die systematische simulative Optimierung erfolgt in Abschnitt 3.5.

Das Lösen der *Riccati*-Gleichung und die Berechnung von $\mathbf{K}$ für alle Arbeitspunkte erfolgt automatisiert in MATLAB.

3.2.2 Zustandsermittlung

Ein Zustandsregler benötigt zur Regelung zu jedem Zeitpunkt die Information aller Zustände $\boldsymbol{x}$ (2.5) des Systems. Beim Schlitten-Pendel-System werden die drei Zustände x_0, φ_1 und φ_2 gemessen. Deren Ableitungen, die Zustände $\dot{x}_0$, $\dot{\varphi}_1$ und $\dot{\varphi}_2$ müssen noch ermittelt werden. Dabei wird von drei Möglichkeiten ausgegangen, die in der Simulation miteinander verglichen werden können:

1. Zustandsmessung
2. Beobachter
3. Differenzieren

Am Versuchsstand erfolgt die Zustandsermittlung durch einen Beobachter.

Zustandsmessung

Bei der Zustandsmessung wird von einer idealen Messung aller Zustände ausgegangen, d. h. $\hat{\boldsymbol{x}} = \boldsymbol{x}$. Diese Variante wird in der Simulation meist zuerst eingesetzt, um die Regelung des Systems besser beurteilen zu können und spezielle Probleme durch die Zustandsermittlung im Nachhinein analysieren zu können. Am realen Versuchsstand kann diese Methode nicht eingesetzt werden.

Beobachter

Der Schätzzustand $\hat{\boldsymbol{x}}$ wird durch einen *Luenberger-Beobachter* ermittelt. Dieser „simuliert“ ein lineares Schätzsystem ($\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$) und regelt Unterschiede in den Ausgangsgrößen $\boldsymbol{y}$ und $\hat{\boldsymbol{y}}$ mit der Beobachter-Matrix $\mathbf{L}$ aus [1]. Wie beim Zustandsregler muss auch hier die Auslegung für jeden Arbeitspunkt separat erfolgen. Damit gilt allerdings auch, dass der Beobachter nur in der Nähe dieser Ruhelage gültig ist, bei zu großen Abweichungen kann er aufgrund der Linearisierung die Zustände nicht mehr richtig schätzen.

$\mathbf{L}$ wird in dieser Arbeit mittels Polplatzierung ausgelegt. Einflussparameter sind somit die Pole des Beobachters $\boldsymbol{p}_b$. Diese sollten üblicherweise weiter links liegen als die Pole des geregelten Systems $\boldsymbol{p}_r$. Sie können aber nicht beliebig weit links liegen, da sonst Messrauschen verstärkt wird.

In [4] werden sie beispielsweise hintereinander auf der negativen reellen Achse verteilt:

$$\boldsymbol{p}_b = \begin{bmatrix} -40 & -41 & -42 & -43 & -44 & -45 \end{bmatrix}$$

Im Allgemeinen ist es aus oben genannten Gründen sinnvoll, die Beobachter-Pole in Abhängigkeit der Pole des geschlossenen Regelkreises zu definieren. Bei [14] werden sie nach

$$\boldsymbol{p}_b = \boldsymbol{p}_r - 25$$

bestimmt, wodurch allerdings komplexe Beobachter-Pole möglich sind. In dieser Arbeit werden sie daher meist folgendermaßen berechnet:

$$\boldsymbol{p}_b = \Re\left\{\boldsymbol{p}_r \cdot 5\right\}$$

Neben dem Messvektor $\boldsymbol{y}$ ist auch die Stellgröße u Eingang des Beobachter. Dafür könnte man direkt die geforderte Stellgröße des Zustandsregler u_{reg} verwenden. Allerdings wird letztere bei bestimmten

Regelabweichungen häufig deutlich über der maximalen Stellgröße liegen, sodass die Ausgangsstellgröße spätestens beim Motor begrenzt wird. Wenn der Beobachter dies nicht mitbekommt und daher sein System mit der unbegrenzten Regler-Stellgröße simuliert, gibt es zwangsläufig Abweichungen zwischen Beobachter-System und realem System. Der Zustand wird nicht mehr richtig geschätzt, folglich ergibt sich ein schlechteres Regelverhalten, bis hin zur Instabilität.

Aus diesen Gründen ist es sinnvoll, eine Vorsteuerung zu implementieren (siehe Abschnitt 3.2.4), die die Stellgröße schon vor der Ausgabe an den Motor begrenzt und die tatsächliche Stellgröße zurück an den Beobachter gibt, sodass dieser das reale System besser nachbilden kann. Somit weicht die Stellgröße nur noch bei Stellgrößeneinbrüchen anderer Art (siehe Abschnitt 2.2.2) ab.

Differenzieren

Die übrigen Zustände werden durch Differentiation der Messgrößen und evtl. Tiefpass-Filterung gebildet:

$$\hat{\boldsymbol{x}} = \begin{bmatrix} 1 & 0 & 0 \\ \frac{\mathrm{d}}{\mathrm{d}t} & 0 & 0 \\ 0 & 1 & 0 \\ 0 & \frac{\mathrm{d}}{\mathrm{d}t} & 0 \\ 0 & 0 & 1 \\ 0 & 0 & \frac{\mathrm{d}}{\mathrm{d}t} \end{bmatrix} \boldsymbol{y} = \begin{bmatrix} x_{0m} \\ \frac{\mathrm{d}}{\mathrm{d}t} x_{0m} \\ \varphi_{1m} \\ \frac{\mathrm{d}}{\mathrm{d}t} \varphi_{1m} \\ \varphi_{2m} \\ \frac{\mathrm{d}}{\mathrm{d}t} \varphi_{2m} \end{bmatrix} \tag{3.3}$$

In der Simulation kam es hierbei zu Problemen, da für eine möglichst glatte und fehlerfreie Differentiation eine kleine Schrittweite notwendig ist, außerdem hat sich die Simulation in manchen Fällen aufgehängt. Daher wird diese Variante in dieser Arbeit nicht weiter verfolgt.

Trotzdem sei angemerkt, dass diese Möglichkeit am realen Versuchsstand in Kombination mit einer sinnvollen Filterung der Messgrößen und den Ableitungen möglicherweise Vorteile gegenüber einem Beobachter darstellt. Es ist kein Einschwingen der Startzustände nötig, es gibt keine Probleme bei Stellgrößenstörungen und die Differentiation muss nicht für jeden Arbeitspunkt einzeln ausgelegt werden.

3.2.3 Vorsteuerung F/a und a/v-Regler

Ist der AP-Regler nach dem Beschleunigungssystem ausgelegt, gibt dieser als Stellgröße eine Sollbeschleunigung a_{soll} für den Schlitten vor. Aus dieser muss die geforderte Motorkraft bestimmt werden. Sie kann zum Großteil mit einer Vorsteuerung bestimmt werden, die Differenz wird mit einem Geschwindigkeitsvergleich ausgeregelt.

a/v-Regler

Die Sollgeschwindigkeit v_{soll} wird durch Integration von a_{soll} berechnet. Dadurch kann ein Soll/Ist-Vergleich mit der realen bzw. geschätzten Schlittengeschwindigkeit $\dot{x}_0$ durchgeführt werden. Die Regelung wird durch einen P-Regler umgesetzt:

$$F_{av} = K_{pv}(v_{\mathrm{soll}} - \hat{\dot{x}}_0)$$

Brehl [4] ging für K_{pv} beispielsweise von 500 aus, während im SIMULINK-Modell von [14] 150 verwendet wird. Der Parameter scheint am realen Modell einen starken Einfluss auf die Stabilität und ein mögliches „Ratterverhalten“ zu haben. In dieser Arbeit wird ein Wert von 150 angenommen und der Parameter nicht weiter untersucht, da es in der Simulation meist nur geringe Abweichungen von der Sollgeschwindigkeit gibt.

Als problematischer stellte sich hingegen ein möglicher *Windup-Effekt* heraus: Ist die Sollbeschleunigung größer als wegen der Stellgrößenbegrenzung tatsächlich erreicht werden kann, wird v_{soll} immer weiter aufintegriert und damit auch F_{av}, obwohl die Motorkraft bereits maximal ist. Wenn das System wieder aus der Stellgrößenbegrenzung ist und die Vorsteuerungskraft eigentlich das Vorzeichen wechselt, ist der I-Anteil noch vorhanden und muss erst abgebaut werden. In dieser Zeit übersteigt F_{av} daher F_{vorst} und die Stellgröße geht somit praktisch in die falsche Richtung, was die Regelung sehr schnell instabil werden lassen kann. Aus diesem Grund wird a_{soll} vor der Integration mit einem Sättigungsglied auf

$$a_{\text{soll,max}} = \frac{F_{\text{max}}}{m}$$

begrenzt.

Vorsteuerung (M,C,D)

Der a/v-Regler alleine könnte die Schlittenkraft zwar regeln, allerdings kann der Hauptteil der Kraft mit einer Vorsteuerung bestimmt werden, wodurch die Dynamik besser ist. Die Vorsteuerung besteht im Wesentlichen aus der Trägheitskraft, da bei reiner Betrachtung des Schlittens $F_{\text{soll}} = m_0 a_{\text{soll}}$ gilt. Wenn davon ausgegangen wird, dass die Reibungsparameter hinreichend genau bestimmt wurden, kann auch die Reibung des Schlittens kompensiert werden. Während [4] nur die Trägheitskraft vorsteuerte, wird in [14] die Vorsteuerung folgendermaßen berechnet:

$$F_{\text{vorst}} = m a_{\text{soll}} + F_{c0} \operatorname{sign}(v_{\text{soll}}) + d_0 \dot{x}_0$$

Vorsteuerung mit Systemgleichung

Auch die obige Gleichung bestimmt die benötigte Kraft nicht exakt, da es je nach Winkellage der Pendel rückwirkende Kräfte auf den Schlitten gibt. Man kann daher einen Schritt weiter gehen und die benötigte Kraft zur Beschleunigung des Schlittens direkt aus der entsprechenden Bewegungsgleichung herleiten. Auch wenn am realen Versuchsstand die Beschleunigung aufgrund von Ungenauigkeiten nicht exakt mit a_{soll} übereinstimmt, wird dem a/v-Regler dadurch weitere „Arbeit abgenommen“.

Die Beziehung zwischen der Schlittenbeschleunigung und der Kraft am Schlitten wird durch die zweite Zustandsgleichung (2.8) beschrieben:

$$a = \ddot{x}_0 = f_2(\boldsymbol{x}, F) \tag{3.4}$$

Diese Gleichung kann leicht nach der Kraft umgestellt werden:

$$F_{\text{soll}} = \frac{a_{\text{soll}} - a_2(\boldsymbol{x})}{b_2(\boldsymbol{x})} \tag{3.5}$$

Damit diese Berechnung für beliebige Systemparameter und somit unterschiedliche Funktionen a_2 und b_2 erfolgen kann, wird die Bestimmung der Vorsteuerkraft in dieser Arbeit symbolisch in MATLAB automatisiert.

Ermittlung reale Beschleunigung

In Abschnitt 3.2.2 wurde erläutert, dass es sinnvoll ist, die reale Stellgröße an den Beobachter zu geben. Im folgenden Vorsteuerungsblock (Abschnitt 3.2.4) wird die vor dem Motor begrenzte Kraft F_{real} zurückgegeben, da die Stellgrößenbegrenzung $F_{\max}$ bekannt ist. Ist der Beobachter allerdings am Beschleunigungssystem ausgelegt, geht er bei der Stellgröße von der Beschleunigung aus. Somit muss aus F_{real} zunächst a_{real} bestimmt werden.

Dies kann ähnlich wie bei der Kraftvorsteuerung durch die Trägheit geschehen, was allerdings ungenau ist und der Beobachter demzufolge von einer falschen Eingangsgröße ausgeht. Daher wird die Ermittlung der tatsächlichen Beschleunigung wie im vorigen Abschnitt über die Systemgleichung vorgenommen. Es gilt (3.4).

3.2.4 Motor Vorsteuerung

Bisher wurde für die Regelung die Ermittlung der Kraft, die am Schlitten wirken soll, betrachtet. Die Schnittstelle zum Motor besteht durch Vorgabe der Steuerspannung. Die ausgegebene Spannung stellt daher letztendlich die eigentliche Stellgröße dar und muss in Abhängigkeit der Sollkraft bestimmt werden.

Auch wenn das Motorsystem für die Simulation genauer modelliert wird (siehe Abschnitt 2.2), handelt es sich (abgesehen von einer sehr schnellen Tiefpass-Dynamik) im Wesentlichen um ein reines Verstärkungsglied, das mit der Konstante $K_{\text{MotorGain}}$ (2.18) beschrieben wird. Die Vorsteuerung rechnet daher mit der Konstante die Sollkraft in die Sollspannung um.

$$U_{\text{soll}} = \frac{F_{\text{soll}}}{K_{\text{MotorGain}}}$$

In der Motor Vorsteuerung wird auch die Stellgrößenbegrenzung berücksichtigt. Wie schon in Abschnitt 3.2.2 erläutert wurde, wird die Motorspannung vor der Ausgabe begrenzt. Die tatsächliche Stellgröße kann dadurch an den Beobachter gegeben werden.

3.3 Simulationsmodell in SIMULINK

Die SIMULINK-Modelle und MATLAB-Funktionen zur Arbeitspunkt-Regelung befinden sich im Ordner `Regelung`.

Die Arbeitspunkt-Regelung wird wie in Abschnitt 3.2 beschrieben als SIMULINK-Modell implementiert (siehe Abbildung 3.2). Das Gesamtmodell-Modul (siehe Abschnitt 2.4.2) wird hier eingebunden und stellt die Regelstrecke dar. Davor befinden sich drei Blöcke der Regelung/Vorsteuerung. Alternativ kann auch zum Arbeitspunkt-Regler am Kraftsystem umgeschaltet werden.

APRegler.slx

Ist eine Art „Wrapper“ für den eigentlich Zustandsregler, um die Arbeitspunkt-Transformation durchzuführen. Erwartet als Parameter die Daten zum Arbeitspunkt.

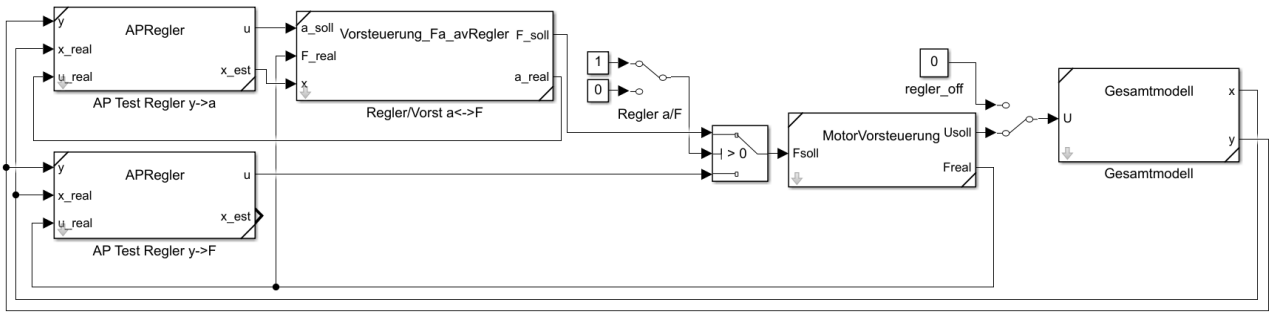


Abbildung 3.2: Aufbau der Arbeitspunkt-Regelung

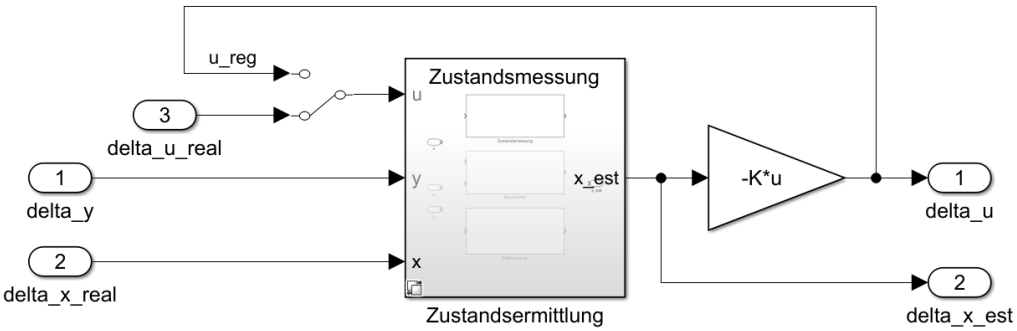


Abbildung 3.3: Zustandsregler in SIMULINK

Zustandsregler.slx

Implementiert einen gewöhnlichen Zustandsregler, bei dem alle Arten der Zustandsermittlung (siehe Abschnitt 3.2.2) möglich sind (siehe Abbildung 3.3). Die Auswahl wird durch ein *Variant Subsystem* realisiert, das durch die Variable `simparams.Zustandsermittlung` angesteuert wird (siehe Abbildung 3.4). Die Stellgröße für den Beobachter kann zwischen intern und extern umgeschaltet werden. Benötigt als Parameter die Verstärkung `K` sowie eventuelle Daten für den Beobachter.

Beobachter.slx

Implementiert einen Standard-Beobachter. Eingänge sind `u` und `y`, Ausgang sind die geschätzten Zustände `x_est`. Die Matrizen des linearen Systems, sowie `L` und die Beobachterstartwerte werden über die Maske übergeben.

MotorVorsteuerung.slx

Enthält die in Abschnitt 3.2.4 beschriebene Vorsteuerung zum Motor.

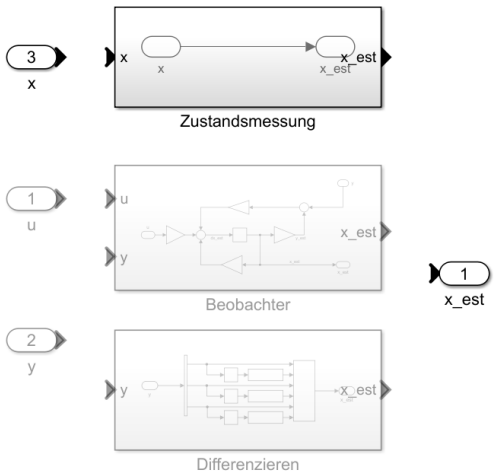


Abbildung 3.4: Zustandsermittlung in SIMULINK

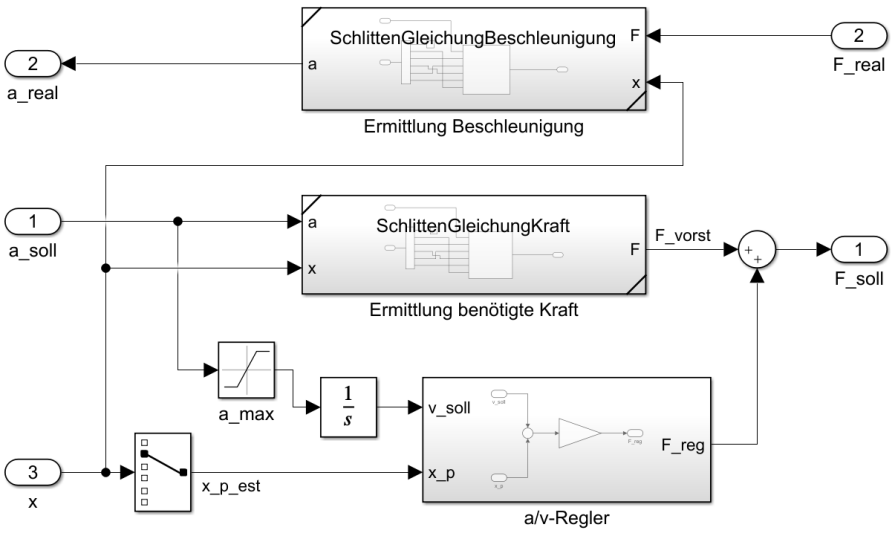


Abbildung 3.5: Aufbau der Vorsteuerung

Vorsteuerung_Fa_avRegler.slx

SchlittenVorsteuerung.slx

Enthält die „alte“ Variante der Vorsteuerung (siehe Abschnitt 3.2.3). Dabei werden die Schlittenparameter (m_0, F_{c0} und d_0) verwendet.

SchlittenGleichungKraft.slx und SchlittenGleichungBeschleunigung.slx

3.3.1 Initialisierung

3.3.2 Weitere MATLAB-Funktionen

SimAP

3.4 Anfangswert-Tests

Ziel dieser Arbeit ist die „Regelbarkeit“ des Systems zu untersuchen. Dafür müssen zunächst Kriterien definiert werden, inwiefern das System unter festgelegten System- und Entwurfsparametern regelbar ist.

Dazu wird wieder Abbildung 3.1 herangezogen. Ist ein System fest ausgelegt, können die verschiedenen Arbeitspunkte mit bestimmten Startwerten getestet werden. Dabei gibt es jedoch unzählige Kombinationsmöglichkeiten, die nicht alle getestet werden können. Es muss daher eine Auswahl an Startwerten getroffen werden (beispielsweise wird immer nur ein Zustand gesetzt, alle anderen sind am Arbeitspunkt). Selbst dann ist es mit manuellem Testen und Ausprobieren aufwändig, ein umfassendes Bild vom Regelverhalten zu erlangen. Die Optimierung der „äußeren Schleifen“ wird somit langwierig und auf diese Weise nicht zielführend.

Daher wird die „untere Schleife“ automatisiert. Wird ein Arbeitspunkt-Test durchgeführt, können die Simulationsdaten ausgewertet werden und gewisse Gütemaße und Performance-Indikatoren bestimmt werden, um die Regelgüte zu beschreiben. Im Vordergrund steht aber meist die Stabilität und damit die Frage, bei welchen Werten diese nicht mehr vorhanden ist.

3.4.1 Auswertung eines Arbeitspunkt-Tests

Wenn `SimAP` aufgerufen wird, werden neben den Simulationsdaten auch die Ergebnisse von `Auswertung/AP/APAus` zurückgegeben. Dort werden u. a. folgende Gütemaße, die an das übliche quadratische *Riccati*-Gütemaß

angelehnt sind, berechnet:

$$J_x = \int_0^{t_\mathrm{end}} \Delta\boldsymbol{x}^\mathrm{T}(t)\,\mathbf{Q}\,\Delta\boldsymbol{x}(t)\,\mathrm{d}t = \int_0^{t_\mathrm{end}} \Delta x_\mathrm{norm}(t)\,\mathrm{d}t \tag{3.6a}$$

$$J_{x,\mathrm{est}} = \int_0^{t_\mathrm{end}} \Delta\boldsymbol{x}_\mathrm{est}^\mathrm{T}(t)\,\mathbf{Q}\,\Delta\boldsymbol{x}_\mathrm{est}(t)\,\mathrm{d}t \tag{3.6b}$$

$$J_u = \int_0^{t_\mathrm{end}} R\cdot\Delta u(t)^2\,\mathrm{d}t \tag{3.6c}$$

$$J_F = \int_0^{t_\mathrm{end}} R\cdot F(t)^2\,\mathrm{d}t \tag{3.6d}$$

mit

$$\mathbf{Q} = \mathrm{diag}\,(25, 1, 50, 1, 50, 1)$$
$$R = 1$$

In MATLAB werden sie als Summe berechnet (der Einfachheit halber konstante Schrittweite vorausgesetzt).

Für die Beurteilung der Stabilität, also ob der Zustand am Ende der Simulationszeit noch in der Nähe des Arbeitspunktes ist, wird $\Delta x_\mathrm{norm}(t_\mathrm{end})$ verwendet. Ist dieser skalare Wert unter einer festgelegten Schranke, wird von einer Stabilisierung des Arbeitspunktes ausgegangen.

Außerdem wird für jeden Zustand die Maximalabweichung vom Arbeitspunkt bestimmt, da diese ein wichtiges Gütekriterium ist, insbesondere die Schlittenposition, da diese Beschränkungen unterliegt (siehe Abschnitt 2.1.6).

Des Weiteren kann anhand von $\Delta x_\mathrm{norm}(t)$ eine Einschwingzeit bestimmt werden. Diese ist dort, wo der Norm-Verlauf eine gewisse Schranke nicht mehr überschreitet.

3.4.2 x0-Test

Die Funktion `Regelung/x0_Tests/x0_Test.m` stellt die Basis für einen Anfangswerttest dar. Für den angegebenen Arbeitspunkt ermittelt sie mit einzelnen Simulationen und steigenden Anfangsauslenkungen vom Arbeitspunkt schrittweise die maximalen Startwerte. Dabei wird mit obigen Methoden ermittelt, ob die Stabilisierung erfolgreich war. Falls nicht, wird abgebrochen und der Maximalwert sowie einige der Gütemaße für allen Einzelergebnisse zurückgegeben.

Die Schrittweite `y_st = [ x_st phi1_st phi2_st ]` gibt die Schrittweite an, mit der die Tests ausgeführt werden. Eine kleinere Schrittweite löst zwar die Ergebnisse besser auf, führt aber zu einer höheren Rechenzeit.

Die Ergebnisse eines Anfangswerttests an AP4 sind beispielhaft in Abbildung 3.6 dargestellt. Auf der Abszisse wird der Startwert der Auslenkung (hier φ_2) aufgetragen. In Abhängigkeit von dieser werden die maximalen Abweichungen der drei Ausgänge vom Arbeitspunkt dargestellt. In diesem Fall kann das System bis zu einem Startwert von $\varphi_{2,0} = 19\,^\circ$ stabilisiert werden. Die Startauslenkung wird auch als schwarze Linie eingezeichnet. Die Maximalabweichung ist stets oberhalb oder auf dieser.

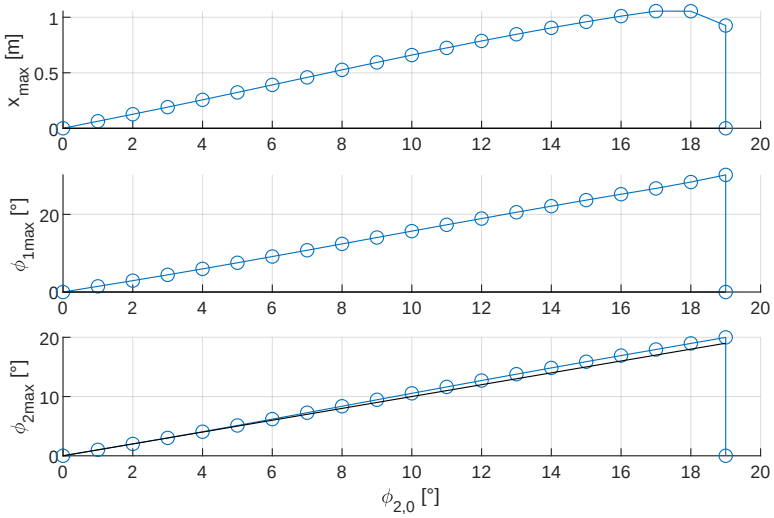


Abbildung 3.6: Anfangswerttest $\mathrm{AP}4$, Auslenkung φ_2

3.4.3 Kritische Anfangswert-Tests

Wenn immer nur ein Zustand ausgelenkt wird, ergeben sich pro Arbeitspunkt 3 Tests, also insgesamt 9 Tests (unter Vernachlässigung des stabilen $\mathrm{AP}1$). Es hat sich dabei gezeigt, dass Abweichungen von x_0 kein Problem bezüglich der Stabilität darstellen. Außerdem sind bei Arbeitspunkt 2 und 3 Auslenkungen des jeweils „stabilen“ Pendels unkritisch.

Aus diesem Grund sind für die Untersuchung der Stabilität nur die folgenden vier (kritischen) Anfangswert-Tests relevant:

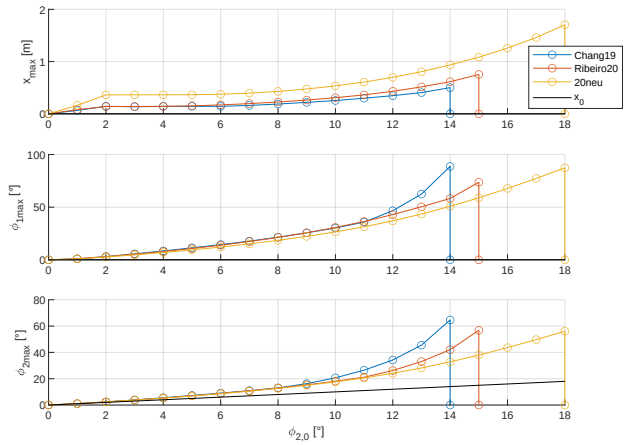
- Arbeitspunkt 2, Auslenkung φ_2
- Arbeitspunkt 3, Auslenkung φ_1
- Arbeitspunkt 4, Auslenkung φ_1
- Arbeitspunkt 4, Auslenkung φ_2

In `x0_Tests/x0_Test_APs.m` werden diese vier Tests durchgeführt.

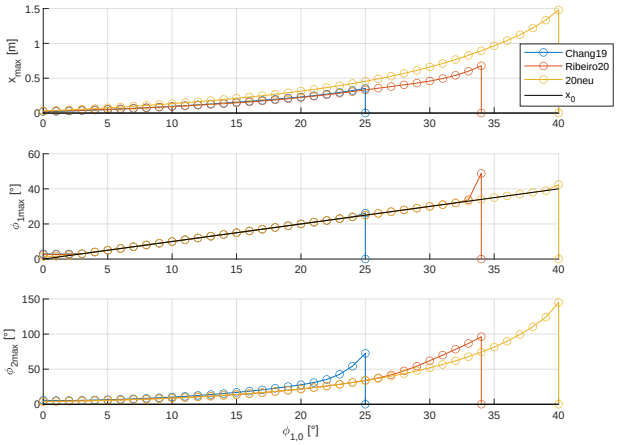
3.5 QR Parameter Tests

3.6 System Parameter Tests

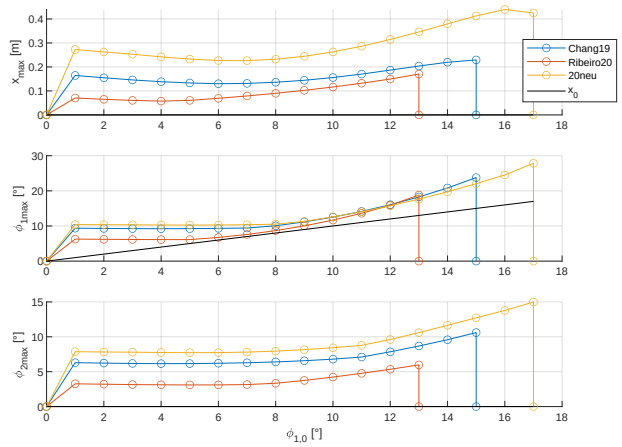
System Ribeiro



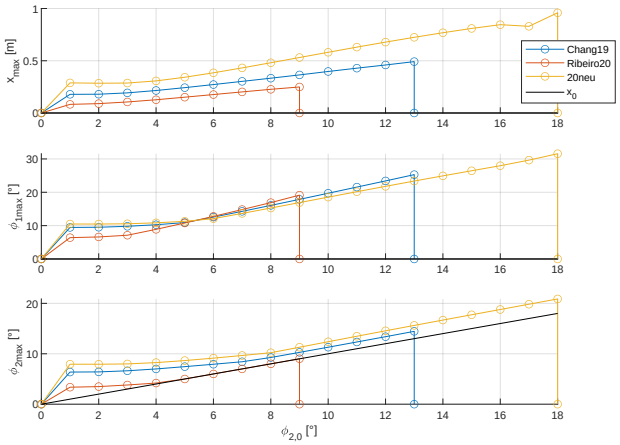
(a) AP2 (Auslenkung φ_2)



(b) AP3 (Auslenkung φ_1)

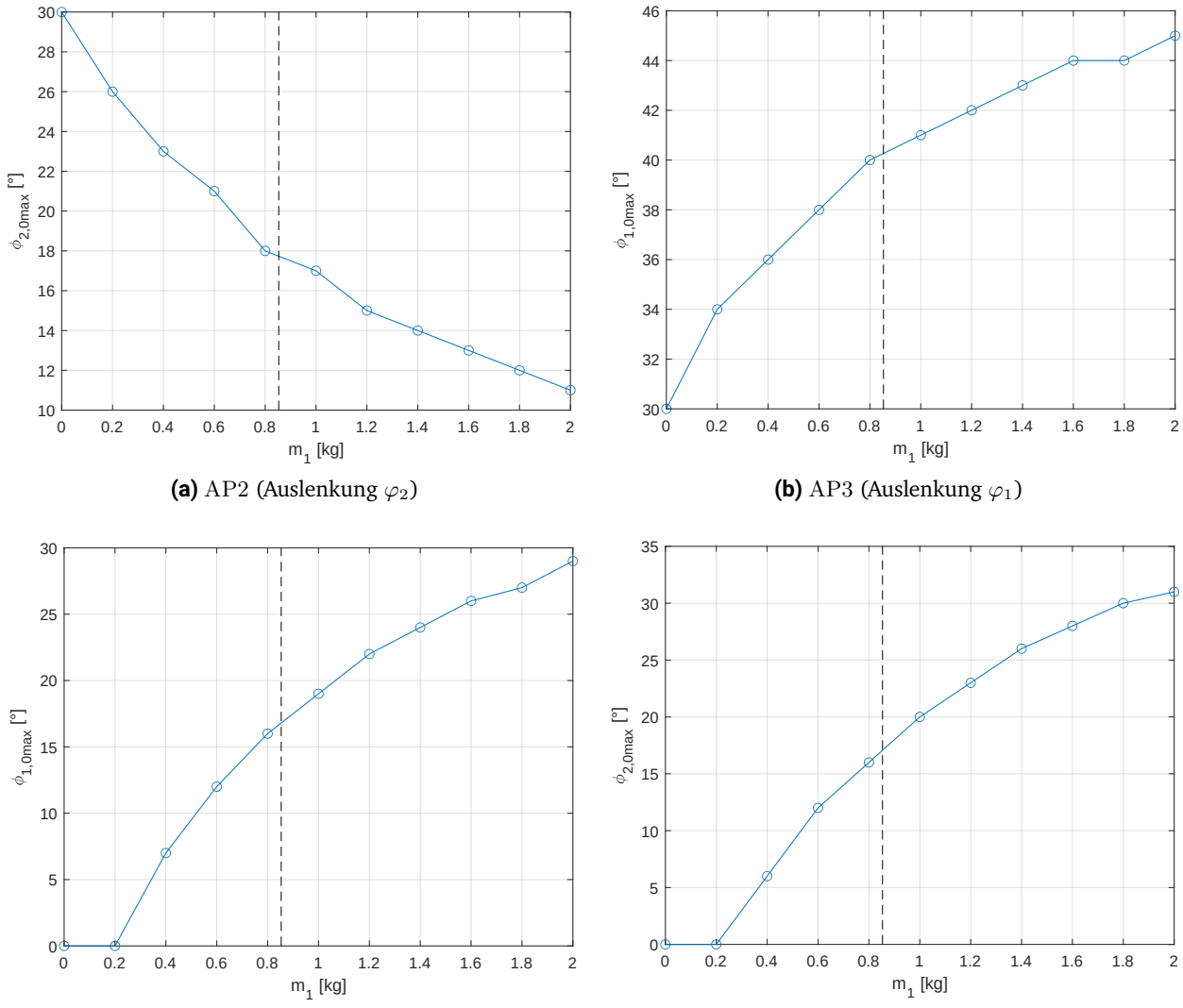


(c) AP4 (Auslenkung φ_1)



(d) AP4 (Auslenkung φ_2)

Abbildung 3.7: Maximalabweichungen – Vergleich QR-Parameter (System Ribeiro)



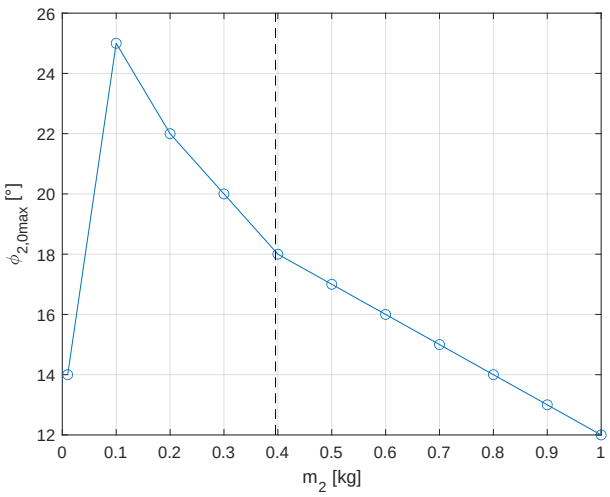
(a) AP2 (Auslenkung φ_2)

(b) AP3 (Auslenkung φ_1)

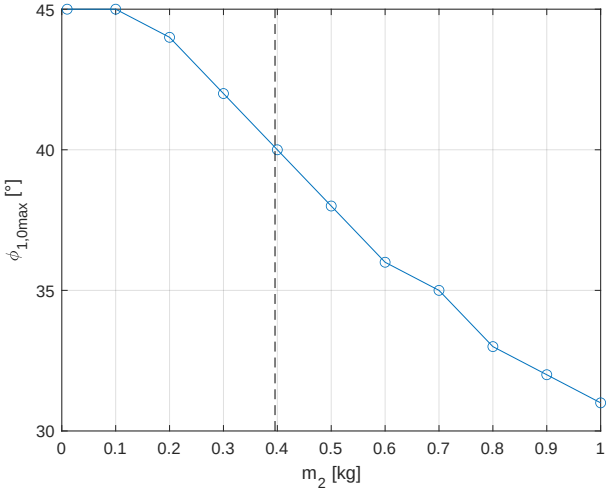
(c) AP4 (Auslenkung φ_1)

(d) AP4 (Auslenkung φ_2)

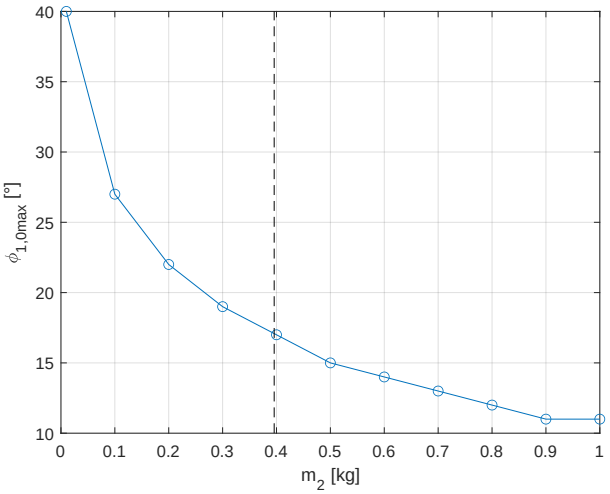
Abbildung 3.8: Maximale Startwerte – Variation m_1



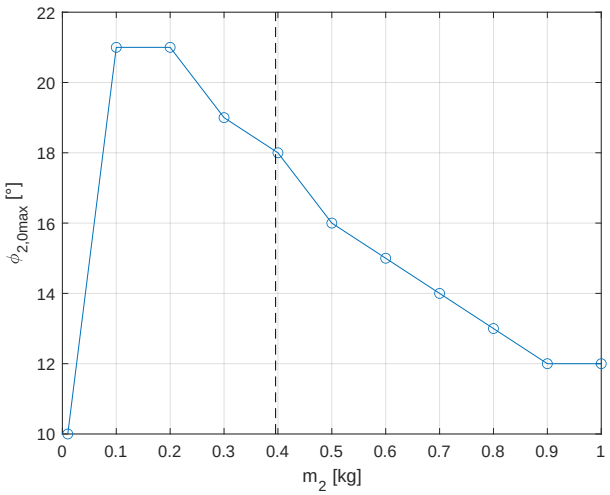
(a) AP2 (Auslenkung φ_2)



(b) AP3 (Auslenkung φ_1)

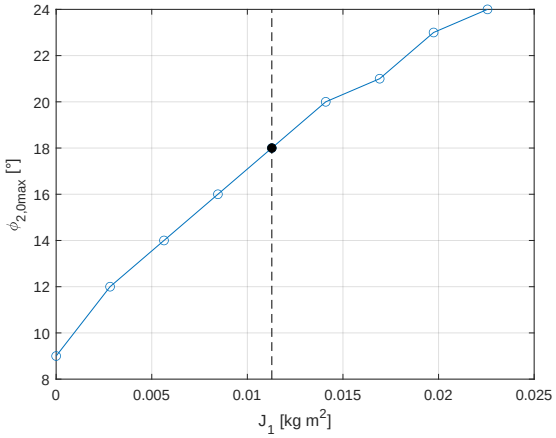


(c) AP4 (Auslenkung φ_1)

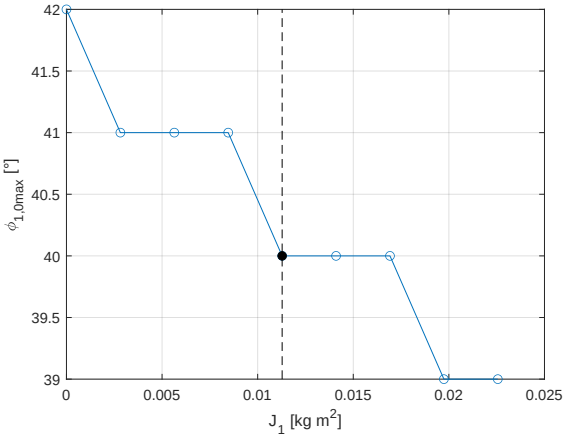


(d) AP4 (Auslenkung φ_2)

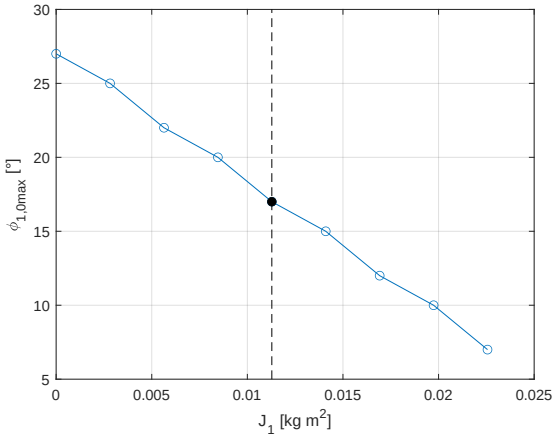
Abbildung 3.9: Maximale Startwerte – Variation m_2



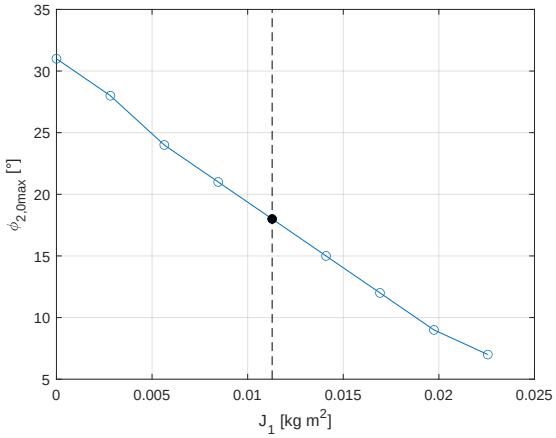
(a) AP2 (Auslenkung φ_2)



(b) AP3 (Auslenkung φ_1)

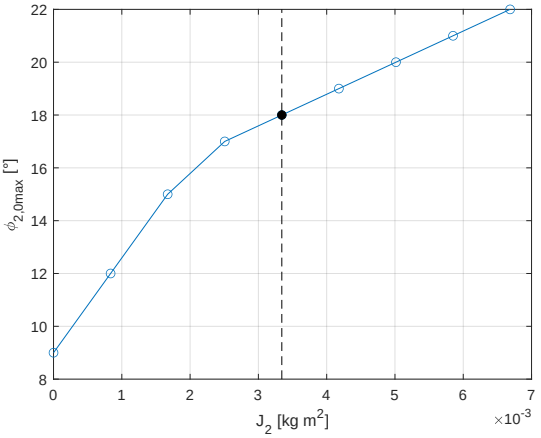


(c) AP4 (Auslenkung φ_1)

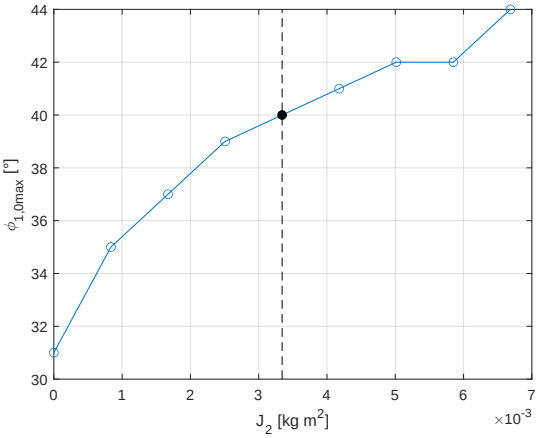


(d) AP4 (Auslenkung φ_2)

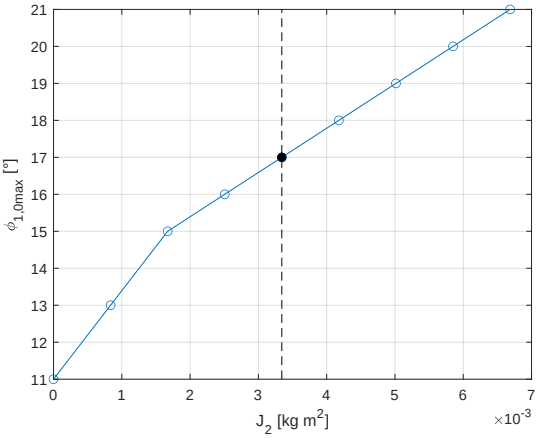
Abbildung 3.10: Maximale Startwerte – Variation J_1



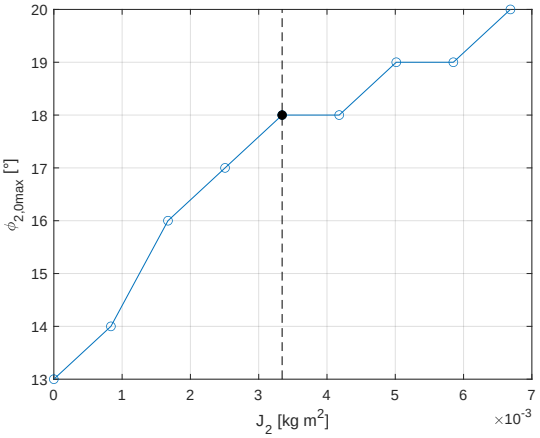
(a) AP2 (Auslenkung φ_2)



(b) AP3 (Auslenkung φ_1)



(c) AP4 (Auslenkung φ_1)



(d) AP4 (Auslenkung φ_2)

Abbildung 3.11: Maximale Startwerte – Variation J_2

4 Trajektorien

4.1 Einleitung

4.2 Implementierung der Trajektorienberechnung

Die Implementierung der Trajektorienberechnung basiert auf den Erkenntnissen der Vorgängerarbeit Fauvé [5] und erfolgt in MATLAB mit Hilfe des *CasADi*-Frameworks. Es wird einerseits auf eine Erhöhung der Modularisierung im Vergleich zur Vorgängerarbeit geachtet, andererseits werden weitere Funktionalitäten ergänzt, um die Trajektorienberechnung für die Anwendungen im Rahmen dieser Arbeit zu optimieren.

Die Berechnung der Trajektorien wird durch fünf Matlab-Funktionen modularisiert:

- `searchTrajectories`
- `calculateTrajectory`
- `getODE`
- `getInitDev`
- `determineAPinit`

Vom Anwender aufgerufen wird lediglich `searchTrajectories`. Die weiteren vier Funktionen werden auch für andere Anwendungen dieser Arbeit eingesetzt, wodurch das Kopieren von Code vermieden wird. Änderungen, die mehrere Funktionen und Skripte betreffen, können daher effizient an einer Stelle im Programmcode durchgeführt werden. Die Argumente der Funktionen werden so gewählt, dass die in dieser Arbeit zu untersuchenden Parameter leicht durch den Funktionsaufruf in einer Schleife variiert werden können. Weitere Parameter die im Rahmen dieser Arbeit innerhalb der Funktionen konstant festgelegt sind, können zu einem späteren Zeitpunkt jedoch leicht zu den Funktionsargumenten ergänzt werden.

Im Folgenden werden die beiden wichtigsten Funktionen `searchTrajectories` und `calculateTrajectory` genauer beschrieben, wobei auch auf die Verwendung der anderen Funktionen eingegangen wird.

4.2.1 searchTrajectories

Durch die Funktion `searchTrajectories` wird vom Anwender eine Trajektoriensuche in Auftrag gegeben. Als Beispiel für die Anwendung kann der folgende Programmcode eines MATLAB-Skriptes betrachtet werden:

```
%% Demo-Aufruf von searchTrajectories
mode = 2;               % Suchprogramm
N = 500;                % Prädiktionshorizont
T = 0.005;              % Schrittweite
sol = 'RK4';            % Integrator für Kontinuitätsbedingung
params= SchlittenPendelParams_Apprich09(); % Parametersatz
u_max = 410;            % Maximale Stellkraft
coulMc = false;         % Gelenkreibung nicht berücksichtigen
coulFc = true;          % Schlittenreibung berücksichtigen

% Starte Suche nach Trajektorien
searchTrajectories(mode, N, T, sol, params, u_max, coulMc, coulFc)
```

Zunächst wird ein Suchprogramm ausgewählt. Es kann zwischen drei Programmen ausgewählt werden:

1. Suche nach der im Rahmen dieser Arbeit definierten Vergleichstrajektorie
2. Suche nach der Aufschwungtrajektorie mit der Variationsstrategie
3. Suche nach allen 12 Trajektorien mit der Variationsstrategie

Dem Ansatz der Programmauswahl liegen die Erkenntnisse von Fauvé [5] bezüglich einer Variationsstrategie zu Grunde. Da auf Grund der nichtlinearen MPC nur lokal optimiert wird, kann je nach Anfangswert ein anderes Optimum erreicht werden. Eine Variation des Anfangswertes ermöglicht somit eine effektive Suche nach Trajektorien für einen bestimmten Arbeitspunktwechsel. Im Code-Beispiel ist Suchprogramm 2 ausgewählt, der gesuchte Arbeitspunktwechsel ist allgemein der Aufschwung von AP1 nach AP4. Hierbei können innerhalb einer Periode vier mögliche Winkelanfangsauslenkungen gegenüber AP4 variiert werden, die als Startwert AP1 definieren:

$$\begin{bmatrix}0\\0\\\pi\\0\\\pi\\0\end{bmatrix}, \quad \begin{bmatrix}0\\0\\-\pi\\0\\\pi\\0\end{bmatrix}, \quad \begin{bmatrix}0\\0\\\pi\\0\\-\pi\\0\end{bmatrix}, \quad \begin{bmatrix}0\\0\\-\pi\\0\\-\pi\\0\end{bmatrix}.$$

Eine weitere Variation durch Ausnutzung der 2π-Periodizität bringt gemäß Fauvé [5] hingegen keine Vorteile, sondern verlängert die Berechnungszeit bei gleichzeitig sinkender Wahrscheinlichkeit, dass der Algorithmus zu einer gültigen Lösung konvergiert. Die möglichen Anfangsauslenkungen werden über die Funktion `getInitDev` aufgerufen. Neben den Anfangsauslenkungen der Pendel hat auch die Schlittenposition Einfluss auf die Lösungsfindung. Daher wird diese mit den folgenden auf Fauvé [5] basierenden Auslenkungen symmetrisch zur Bahnmitte variiert (siehe Tabelle 4.1).

Tabelle 4.1: Variation der Schlittenposition

$x_{0,\text{init}}$	$x_{0,\text{end}}$
$-0{,}5$	$0{,}5$
$-0{,}3$	$0{,}3$
$-0{,}1$	$0{,}1$
0	0

Als letzte Komponente der Variationsstrategie wird noch die Positionsbeschränkung variiert. Obwohl der Versuchsstand eine Positionsbeschränkung von $\pm 0{,}8\,\text{m}$ vorgibt, ist es sinnvoll, auch diese zu variieren, da der Algorithmus sich sensitiv gegenüber den Nebenbedingungen verhält. Durch Verschärfung oder Lockerung der Positionsbeschränkung verändert sich die Optimierungslandschaft und damit die Chance, eine Trajektorie zu finden. Bei Lockerungen der Beschränkung muss später immer geprüft werden, ob bei den gefundenen Trajektorien die Positionsbeschränkungen eingehalten werden. Es werden folgende von Fauve [5] entnommene Variationen implementiert.

$$-\underline{g}_{x_0} = \overline{g}_{x_0} \in \{0{,}6;\ 0{,}8;\ 1;\ 1{,}2;\ 1{,}4\}$$

Insgesamt werden bei Suchprogramm 2 somit $4 \cdot 4 \cdot 5 = 80$ Trajektorienberechnungen durchlaufen. Bei Suchprogramm 3 werden die beschriebenen Variationen auf alle zwölf Trajektorien ausgeweitet, indem der Endwert durch alle vier Arbeitspunkte iteriert und die Variation der Winkelanfangsauslenkungen um die vier zusätzlich entstehenden Kombinationen durch die Arbeitspunkte 2 und 3 erweitert wird. Insgesamt werden $8 \cdot 4 \cdot 5 \cdot 4 = 640$ Trajektorienberechnungen durchgeführt. Im Rahmen dieser Arbeit wird eine Vergleichstrajektorie für die in Kapitel 4.6 behandelten Parameteruntersuchungen verwendet. Diese wird mit Suchprogramm 1 berechnet.

Nachdem das Suchprogramm für die Trajektorienberechnung ausgewählt ist, werden Prädiktionshorizont, Schrittweite und Integrationsverfahren für die NMPC konfiguriert. Als Integratoren stehen das Eulerverfahren „Euler“ und das Runge-Kutta-Verfahren „RK4“ zur Verfügung. Außerdem wird einer der in Kapitel 2.3.3 besprochenen Parametersätze übergeben sowie eine maximale Stellkraft definiert. Diese stellt eine weitere Variationsmöglichkeit dar, die von Fauve [5] nicht untersucht worden ist. Daher wird sie zur manuellen Variation in den Argumenten der `searchTrajectories` zur Verfügung gestellt. Als Letztes wird noch bestimmt, ob für die Trajektorienberechnung die Coulombreibung im Schlitten bzw. in den Gelenken berücksichtigt werden soll.

Die berechneten Trajektorien werden automatisch gespeichert. Es wird eine Namenskonvention zur Identifizierbarkeit der gespeicherten Ergebnisse eingeführt, die an folgendem Trajektoriennamen beispielhaft gezeigt wird: `Traj14_dev0_-3.14_-3.14_x0max0.8_Fmax410` .

Am Anfang steht die verallgemeinerte Trajektorienbezeichnung mit AP_{init} und AP_{end}, wobei AP_{init} erst noch aus den Anfangsauslenkungen abgeleitet werden muss. Dies geschieht mit Hilfe der Funktion `determineAPinit`. Es folgen der Variationswert der Schlittenposition, die Winkelanfangsauslenkung von φ_1 und φ_2, die Positionsbeschränkung und die maximale Stellkraft. Wenn nicht anders angegeben, werden die Ergebnisse im Ordner `searchResults` in einer definierten Ordnerstruktur gespeichert, deren Benennung sich nach der Konfiguration der NMPC richtet. Im obigen Beispiel werden die Ergebnisse in folgendem Unterordner gespeichert, wobei `Fc` anzeigt, dass die Coulomb-Reibung des Schlittens in der Berechnung berücksichtigt wurde: `Results_app09_Fc_T0.005N500_RK4` .

Optional kann der Funktion `searchTrajectories` auch ein Pfad für den Speicherordner übergeben werden. Außerdem können zusätzliche Erweiterungen für den Trajektoriennamen übergeben werden, um später bei der Variation von Modellparametern die Variationswerte im Dateinamen zu berücksichtigen. Als weitere Option kann noch bestimmt werden, ob alle berechneten Trajektorien oder nur die mit gültiger Lösung gespeichert werden sollen.

4.2.2 calculateTrajectory

Die Funktion `calculateTrajectory` wird innerhalb von `searchTrajectories` zur eigentlichen Berechnung einer bestimmten Trajektorie aufgerufen. Sie implementiert die NMPC mit Hilfe des *CasADi*-Frameworks auf Grundlage der Erkenntnisse der Vorgängerarbeit. Für detaillierte Hintergrundinformationen wird daher auf die Ausarbeitung von Fauvé [5] verwiesen. Als Argument erwartet die Funktion eine Struktur mit verschiedenen Konfigurationsparametern, die im Quelltext ausführlich kommentiert sind.

Zunächst muss ein Optimalsteuerungsproblem (OCP) mit Zielfunktion und Nebenbedingungen definiert werden. Als Zielfunktion wird

$$J = \sum_{i=1}^{N} \left((x_i - x_{\text{end}})^T \mathbf{Q} (x_i - x_{\text{end}}) + (u_i - u_{\text{end}})^T R (u_i - u_{\text{end}}) \right) + \sum_{i=2}^{N} (u_i - u_{i-1})^T S (u_i - u_{i-1})$$

implementiert. Sie besteht aus dem bekannten *Lagrange'schen* Güteintegral der LQ-Regelung und einem Bestrafungsterm für Änderungen in der Stellgröße, durch den hochfrequente Stellgrößenverläufe reduziert werden. Die Wahl der Gewichtungsmatrizen $\mathbf{Q}$, R und S basiert auf den Erkenntnissen von Fauvé [5]. Tests bestätigen diese Wahl, wobei S etwas nach unten korrigiert wird. Die Matrizen werden anschließend nicht mehr variiert. Sie sind der Funktion `calculateTrajectory` als Konfigurationsparameter zwar zu übergeben, werden in `searchTrajectories` jedoch für die weiteren Anwendungen im Rahmen der Arbeit hartkodiert.

$$\mathbf{Q} = \begin{bmatrix} 500 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0{,}01 & 0 & 0 & 0 & 0 \\ 0 & 0 & 100 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0{,}1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 100 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0{,}1 \end{bmatrix}, \quad R = 5 \cdot 10^{-7}, \quad S = 1{,}5 \cdot 10^{-8}$$

Neben der Zielfunktion sind darüber hinaus folgende Nebenbedingungen zu implementieren:

- Kontinuitätsbedingungen
- Anfangsbedingung
- Endwertbedingung
- Positionsbegrenzung des Schlittens
- Stellkraftbegrenzung

Dabei ist zu beachten, dass die Endwertbedingung, anders als die restlichen Nebenbedingungen, nicht als harte Grenze formuliert, sondern implizit durch die Zielfunktion angenähert wird.

Um eine möglichst hohe Vergleichbarkeit mit der Vorgängerarbeit Fauvé [5] zu erzielen, wird für die Systemgleichungen das Kraftmodell (2.8) gewählt. Die nichtlinearen Gleichungen werden mit Hilfe der Funktion `getODE` in *CasADi*-Symbolik zur Verfügung gestellt.

Die Stellstrombegrenzung in Abhängigkeit der Winkelgeschwindigkeit des Motors wird in Abschnitt 2.2.2 hergeleitet. Diese muss für die Implementierung noch in eine Stellkraftbegrenzung in Abhängigkeit der Schlittengeschwindigkeit umgerechnet werden. Mit

$$\omega = \frac{1}{r_{32}\ K_G}\ \dot{x}_0$$

und

$$I = \frac{r_{32}\ K_G}{K_I}\ \cdot F$$

kann Gleichung (2.17) in die Form

$$F_{\max} = \frac{K_I}{r_{32}\ K_G} \cdot \frac{U_{a,\max}}{R_a} - (\frac{K_I}{r_{32}\ K_G})^2 \cdot \frac{\dot{x}_0}{R_a}$$

gebracht werden. Gleiches gilt für (2.15). Die Fallunterscheidung nach Vorzeichen der Bewegungsrichtung wird wie in Abschnitt 2.2.2 berücksichtigt.

In *CasADi* sind zwei Arten von Beschränkungen zu unterscheiden: Pfadbeschränkungen in Form von Gleichungen und Ungleichungen (z.B. Systemgleichungen) sowie Ober- und Untergrenzen für die Wertebereiche der Optimierungsvariablen (z. B. $u_{\min} < u < u_{\max}$). Die konstante Stellbegrenzung durch den Spannungs-Strom-Wandler wird durch eine Begrenzung des Wertebereichs der Eingangsgröße F realisiert. Die variable Stellbegrenzung durch die Gegeninduktion wird hingegen als Pfadbeschränkung in Form von Ungleichungen implementiert:

$$0 \leq \frac{K_I}{r_{32}\ K_G} \cdot \frac{U_{a,\max}}{R_a} - (\frac{K_I}{r_{32}\ K_G})^2 \cdot \frac{\dot{x}_0}{R_a} - F \quad .$$

Das zu lösende OCP wird durch Diskretisierung in ein endlich-dimensionales nichtlineares Programm (engl.: *Nonlinear Programming*, NLP) überführt, was auch als direkter Ansatz bzw. „first discretize, then optimize“ bezeichnet wird. Dies geschieht durch Einteilung des OCP in N Intervalle der Schrittweite T, wobei N als Zeit- oder Prädiktionshorizont bezeichnet wird. Als numerische Lösungsmethode wird das in Fauvé [5] empfohlene *Direct-Multiple-Shooting*-Verfahren implementiert. Neben dem Eingangsgrößenverlauf zählt bei diesem Verfahren auch der Zustandsgrößenverlauf zu den Optimierungsvariablen. Anders als beim *Single Shooting* werden anstelle des gesamten Prädiktionshorizonts für jedes Intervall die Systemgleichungen gelöst. Dadurch wird das Randwertproblem zunächst in N Anfangswertprobleme zerlegt, die durch die Systemgleichungen und einen jeweils zu optimierenden Startwert definiert werden. Um sicherzustellen, dass die abschnittsweise gelösten Systemgleichungen einen stetigen Verlauf ergeben, werden N Kontinuitätsbedingungen implementiert. Der Lösungsverlauf eines Intervalls soll zum nächsten Zeitschritt dem Startwert des nächsten Intervalls entsprechen. Vorteil des Verfahrens gegen über dem *Single Shooting* ist, dass die Schrittfehler des Integrationsverfahrens sich lediglich über ein Intervall fortpflanzen anstelle des gesamten Prädiktionshorizonts.

Zur Lösung der Systemgleichungen werden das Euler-Verfahren und das Runge-Kutta-Verfahren 4. Ordnung implementiert. Als Lösungsverfahren für das NLP wird nach dem Beispiel von Fauvé [5] das von *CasADi* bereitgestellte IPOPT-Verfahren (Innere-Punkte-Verfahren, engl.: *Interior Point Optimization*) eingesetzt. Die maximale Anzahl an Optimierungsiterationen wird großzügig mit 20000 Iterationen begrenzt, da bei zugeschalteter Reibung in den Systemgleichungen die durchschnittliche Iterationszahl deutlich ansteigen kann.

Vor Beginn der Optimierung müssen alle Optimierungsvariablen mit Startwerten initialisiert werden. Auf Grund des Multiple-Shooting-Ansatzes sind Startwerte über den gesamten Prädiktionshorizont sowohl für die Eingangsgröße als auch für den Zustandsvektor zu schätzen. Für den diskretisierten Eingang u werden die Startwerte mit dem Nullvektor $u_0 = [0\ 0\ \dots\ 0]^T m \in \mathbb{R}^{1\times N}$ initialisiert. Für die Zustände $\boldsymbol{x}$ wird ein S-förmiger Verlauf von x_{init} nach x_{end} geschätzt, da hiermit bei Fauvé [5] gute Ergebnisse erzielt wurden. Der Verlauf wird durch Interpolation mit Hilfe der von MATLAB bereitgestellten Funktion `chip` über 4 Stützstellen gewonnen. Die zugrunde liegende Erwartung ist, dass das System erst langsam anfährt, dann stark beschleunigt und sich am Ende in der Nähe des Zielwerts nur noch langsam ändert.

Da die Funktion `calculateTrajectory` eigens für die Trajektorienberechnung vorgesehen ist, wird, anders als bei Fauve [5], auf eine Iterationsschleife für die Anwendung als Regler verzichtet.

4.3 Implementierung der Trajektorienfolgeregelung

Zur Überprüfung der Stabilisierbarkeit der berechneten Trajektorien wird eine Trajektorienfolgeregelung als zeitvarianter Zustandsregler nach [10] realisiert.

Dazu wird das nichtlineare System zunächst wie in Abschnitt 3.1.1 linearisiert. Die Linearisierung um die Trajektorie führt im Gegensatz zum zeitinvarianten Arbeitspunkt auf ein *linear-zeitvariantes System* (LZV):

$$\Delta\dot{\boldsymbol{x}} = \mathbf{A}(t)\Delta\boldsymbol{x}(t) + \mathbf{B}(t)\Delta F(t)$$

mit

$$\begin{aligned}\boldsymbol{x}(t) &= \boldsymbol{x}_{\text{Traj}}(t) + \Delta\boldsymbol{x}(t)\\ F(t) &= F_{\text{Traj}}(t) + \Delta F(t)\ .\end{aligned}$$

Hierfür lässt sich der linear-zeit*in*variante LQ-Reglerentwurf aus Abschnitt 3.2.1 übertragen, indem das zeitvariante Gütemaß

$$J(t) = \int_0^\infty \Delta\boldsymbol{x}^{\mathrm{T}}(t)\mathbf{Q}\Delta\boldsymbol{x}(t) + R\cdot(\Delta F(t))^2\ \mathrm{d}t$$

durch Lösen der zeitvarianten *Riccati*-Differentialgleichung

$$\dot{\mathbf{P}}(t) = \mathbf{P}(t)\mathbf{B}(t)R^{-1}\mathbf{B}^{\mathrm{T}}(t)\mathbf{P}(t) - \mathbf{P}(t)\mathbf{A}(t) - \mathbf{A}^{\mathrm{T}}(t)\mathbf{P}(t) - \mathbf{Q}$$

minimiert wird. Die Gleichungen werden als Endwertproblem rückwärts in der Zeit gelöst, wobei das System der Endlage $t_{\text{end}} = (N+1)\cdot T$ als Randwert vorgegeben wird.

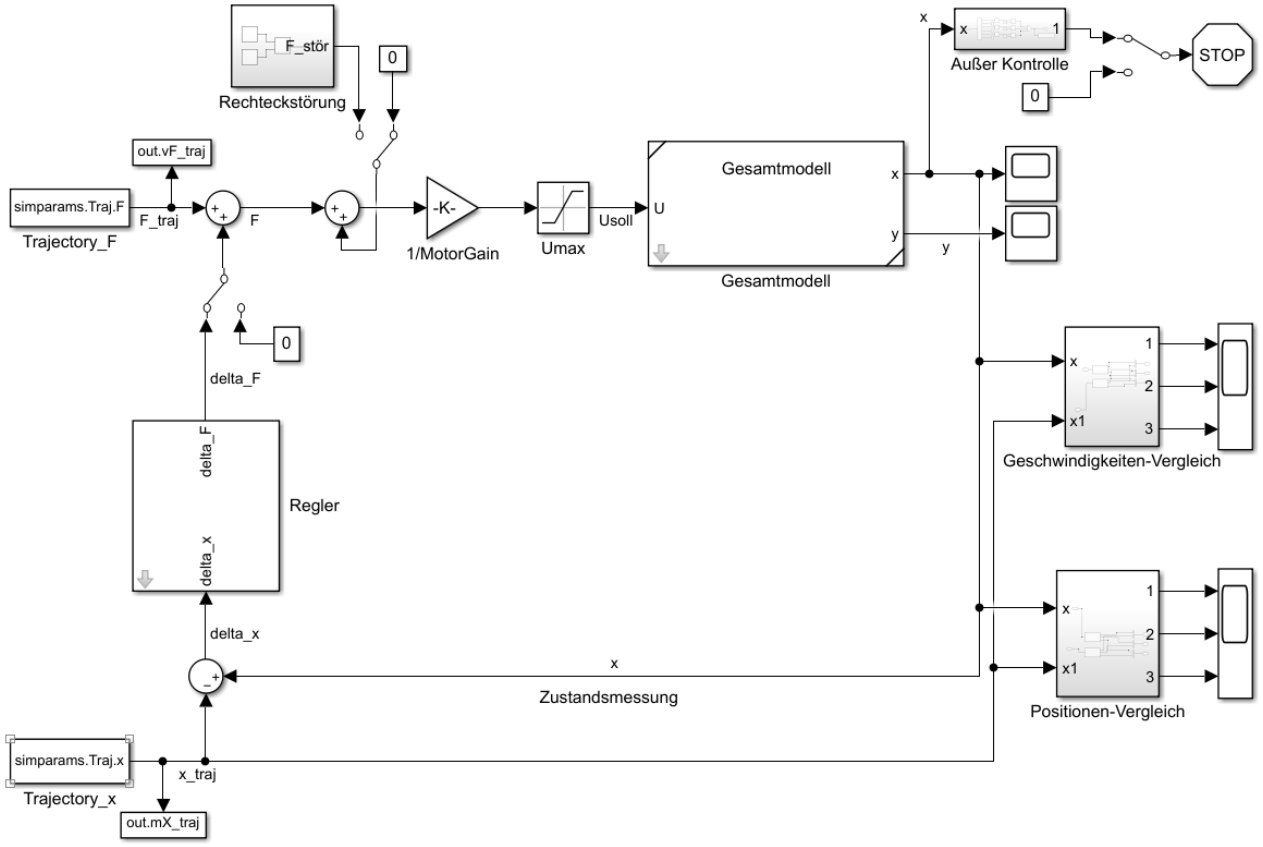


Abbildung 4.1: Trajektorienfolgeregelung in Simulink

Damit lässt sich die Reglerverstärkung

$$\mathbf{K}(t) = R^{-1}\mathbf{B}^{\mathrm{T}}(t)\mathbf{P}(t)$$

für das lineare, zeitvariante Regelgesetz

$$\Delta F(t) = -\mathbf{K}(t)\Delta\boldsymbol{x}(t)$$

berechnen.

Für die Implementierung der Linearisierung und die Berechnung der Reglerverstärkung werden die vom Fachgebiet RTM zur Verfügung gestellten „common“-Funktionen `linSys` und `getTrajFBController_LQR` verwendet. Um die Regelung zu initialisieren wird das MATLAB-Skript `InitTrajReg` erstellt. Die Initialisierung beinhaltet das Laden eines Modellparametersatzes und der zu simulierenden Trajektorie, die Berechnung des trajektorienspezifischen Zustandsreglers sowie die Bereitstellung der Reglerdaten für die Simulation.

Das Simulationsmodell ist in Abbildung 4.1 dargestellt.

Die Trajektorien werden der Simulation als `timeseries` mithilfe von `FromWorkspace` Blöcken zugeführt. Für die anschließende Auswertung in MATLAB hingegen können benötigte Signale über `ToWorkspace` an MATLAB gereicht werden. Um instabiles Verhalten zu erkennen und vorzeitig zu beenden, wird

das Subsystem `AußerKontrolle` implementiert. Optional kann die Regelung zu Vergleichszwecken abgeschaltet werden. Ebenso kann kurzzeitig eine Rechteckstörung aufgeschaltet werden. Die genannten Optionen lassen sich über `ManualSwitches` ein- und ausschalten, wobei deren Zustand aus MATLAB heraus über die Funktion `set_param` angesteuert werden kann.

4.4 Weitere Implementierungen in MATLAB

...

4.5 Stabilisierbarkeit in der Simulation

Für die mittels NMPC berechneten Trajektorien soll gezeigt werden, dass eine Stabilisierung in der Simulation mit Hilfe eines Trajektorienfolgereglers möglich ist. Im Rahmen der Vorgängerarbeit Fauvé [5] war es nicht gelungen, die berechneten Trajektorien am störungsfreien System durch einen zeitvarianten Regler zu stabilisieren. Um das System zu stabilisieren, waren die Gewichtungsmatrizen $\mathbf{Q}$ und R für die Berechnung des Reglers variiert worden.

Aus diesem Grund wird ein anderer Ansatz gewählt, um eine Stabilisierung der mittels NMPC berechneten Trajektorien zu erreichen. Aus dem Skript [9] zur Vorlesung *Modellbildung und Simulation* ist bekannt, dass die Stabilität der Simulation maßgeblich von der Schrittweite und dem Integrationsverfahren abhängt. Bezüglich der Trajektorien können diese an zwei Stellen variiert werden: Einerseits innerhalb der NMPC zur Berechnung der Trajektorie und andererseits in der Simulation in SIMULINK. Lässt sich eine Trajektorie mit verschiedenen Integrationsverfahren stabil simulieren, kann von numerischer Stabilität ausgegangen werden. Es werden daher verschiedene Schrittweiten und Integrationsverfahren für die Berechnung einer Vergleichstrajektorie angewendet und anschließend in der Simulation auf numerische Stabilität und Stabilisierbarkeit in der Regelschleife überprüft. Die Vergleichstrajektorie wird in Abschnitt 4.5.2 definiert.

4.5.1 Vorgehen

Im Gegensatz zu den linearen Systemen können für nichtlineare Systeme nicht ohne Weiteres Stabilitätsgebiete für die verschiedenen Simulationsverfahren angegeben werden. Daher sind prinzipiell nur heuristische Ansätze anwendbar, um eine geeignete Schrittweite zu finden. Allgemein ist bei einer kleineren Schrittweite ein geringerer Schrittfehler und somit eine höhere Güte der Trajektorie zu erwarten. Andererseits steigen Rundungsfehler und Rechenzeit an. Bezüglich der Trajektorienberechnung wirkt sich besonders die Berechnungsdauer dominant aus. Bei einer kleineren Schrittweite muss darauf geachtet werden, dass der Prädiktionshorizont groß genug gewählt wird. Anderenfalls wird keine gültige Lösung mehr gefunden. Bei Fauvé [5] wurde für die Trajektorienberechnung eine Schrittweite von $T = 0{,}01$ (in Sekunden) bei einem Prädiktionshorizont von $N = 350$ verwendet. Versuche zur Findung einer geeigneten Variation zu den von Fauvé [5] verwendeten Parametern ergeben, dass $T = 0.005$ und $N = 500$ einen geeigneten Kompromiss aus möglichst kleiner Schrittweite und akzeptabler Rechenzeit liefern.

Als Integrationsverfahren zur Trajektorienberechnung wurde von Fauvé [5] das Euler-Verfahren eingesetzt. Als Alternative wird dem Euler-Verfahren nun das Runge-Kutta-Verfahren 4. Ordnung (RK4) gegenübergestellt.

Als weiterer Einflussfaktor auf die Güte der Trajektorien und somit auch auf ihre Stabilisierbarkeit werden die Systemparameter vermutet. Ihr Einfluss auf die Trajektorienberechnung wird in Abschnitt 4.6 näher untersucht. Hierzu soll die Variation einzelner Systemparameter von einem Anfangs-Parametersatz ausgehen, für den sich die definierte Vergleichstrajektorie finden und stabilisieren lässt. Daher wird neben Schrittweite und Integrationsverfahren auch zwischen den in Abschnitt 2.3.3 definierten Parametersätzen *Apprich* und *Ribeiro* unterschieden.

Die Versuche zur Stabilisierbarkeit werden störungsfrei und unter Vernachlässigung der Reibung durchgeführt. Auf das Verhalten unter Berücksichtigung der Reibwerte wird in Abschnitt 4.6 eingegangen.

Um eine Vergleichbarkeit zur Vorgängerarbeit herzustellen, werden die Versuche sowohl mit als auch ohne Berücksichtigung der Gegeninduktion des Motormodells durchgeführt.

Die Skripte `TFRSim_SchlittenPendel_run` und `TFRSim_Gesamtmodell_run` werden zur Durchführung der Simulationen implementiert. Beide Funktionen führen automatisiert drei Simulationen mit den Solvern `ode1` (Euler), `ode4` (Runge-Kutta-4) und `ode45` (Dormand-Prince) für eine mit `InitTrajReg` geladene Trajektorie durch und erstellen Plots und Animationen der Simulationsergebnisse. Die Trajektorien werden für die Dauer des Prädiktionshorizonts und mit der gleichen Schrittweite, wie bei ihrer Berechnung verwendet wurde, simuliert. Ausnahme stellt `ode45` aufgrund der variablen Schrittweite da.

Die Funktion `TFRSim_SchlittenPendel_run` ruft das Simulinkmodell `TFR_SchlittenPendel_test` auf, das die Trajektorie direkt am Schlittenpendel-System simuliert, während `TFRSim_Gesamtmodell_run` das Simulinkmodell `TFR_Gesamtmodell_test` aufruft, das die Trajektorie am Gesamtsystem einschließlich des modellierten Gegeninduktionseffekts simuliert. Zum Vergleich werden die Trajektorien jeweils auch ohne Regler simuliert. Da die Simulationen störungsfrei sind, wird zunächst erwartet, dass auch eine Steuerung bereits gute Ergebnisse liefert.

Für die Berechnung des Reglers wurden im Voraus verschiedene QR-Matrizen als Ausgangskonfiguration getestet einschließlich und sich schließlich für

$$\mathbf{Q} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \quad R = 0{,}1$$

entschieden, die der Arbeit von Chang [14] entnommen werden. In Abhängigkeit der Simulationsergebnisse werden diese durch sinnvolle Schätzung noch variiert, um ein möglichst gutes Ergebnis für jede Trajektorie zu erhalten.

4.5.2 Vergleichstrajektorie

Für die Versuche zur Stabilisierbarkeit und zum Einfluss der Systemparameter auf die Trajektorienberechnung (Abschnitt 4.6) wird eine Vergleichstrajektorie definiert.

Allgemein wird hierfür die klassische Aufschwungtrajektrie von AP1 nach AP4 gewählt. Es sind jedoch die in Abschnitt 4.2.2 vorgestellten Variationen zu beachten. Daher wird im Speziellen die Trajektorie `Traj14_dev0_-3.14_-3.14_x0max0.8` als Vergleichstrajektorie definiert. Anfangs- und Endzustand sind somit definiert als

$$\boldsymbol{x}_{\text{init}} = \begin{bmatrix} 0 \\ 0 \\ -\pi \\ 0 \\ -\pi \\ 0 \end{bmatrix}, \qquad \boldsymbol{x}_{\text{end}} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix},$$

wobei die Positionsbeschränkung $-0{,}8 \leq x_0 \leq 0{,}8$ als Nebenbedingung fest vorgegeben wird.

Eine maximale Stellkraft wird in Abhängigkeit der Versuche zur Vergleichstrajektorie ergänzt.

4.5.3 Ohne Gegeninduktion

Das System wird zunächst ohne die Gegeninduktion des Motormodells betrachtet, um im ersten Schritt an den Stand von Fauvé [5] anzuknüpfen. Entsprechend wird auch die Stellkraftbegrenzung auf die in Fauvé [5] verwendete Maximalkraft $F_{\max} = 400\,\text{N}$ eingestellt.

Die Ergebnisse für $T = 0.01$ und $N = 350$ sind in Tabelle 4.4 zusammengefasst.

Tabelle 4.2: $T = 0.01,\ N = 350,\ F_{\max} = 400$

Simulation		Apprich		Ribeiro	
Solver	**TFR**	**Euler**	**RK4**	**Euler**	**RK4**
ode1	ohne	leicht instabil	instabil	instabil	instabil
	mit	**stabil**	**stabil**	instabil	leicht instabil
ode4	ohne	instabil	instabil	instabil	instabil
	mit	instabil	instabil	instabil	**stabil**
ode45	ohne	instabil	instabil	instabil	instabil
	mit	instabil	instabil	instabil	leicht instabil

Es wird zwischen

stabil	Aufschwung gelingt, AP4 kann bis zum Ende der Simulation gehalten werden
leicht instabil	Aufschwung gelingt weitgehend, AP4 wird nicht gehalten oder Position ist instabil
instabil	Aufschwung gelingt nicht

unterschieden. Zum besseren Verständnis soll die Zuordnung der dargestellten Ergebnisse an Hand einiger repräsentativer Beispiele erläutert werden.

Zunächst wird der Eintrag links oben in Tabelle 4.4 betrachtet. Hierbei wird die Vergleichstrajektorie mit dem Apprich-Parametersatz und dem Euler-Verfahren berechnet und anschließend auch wieder mit

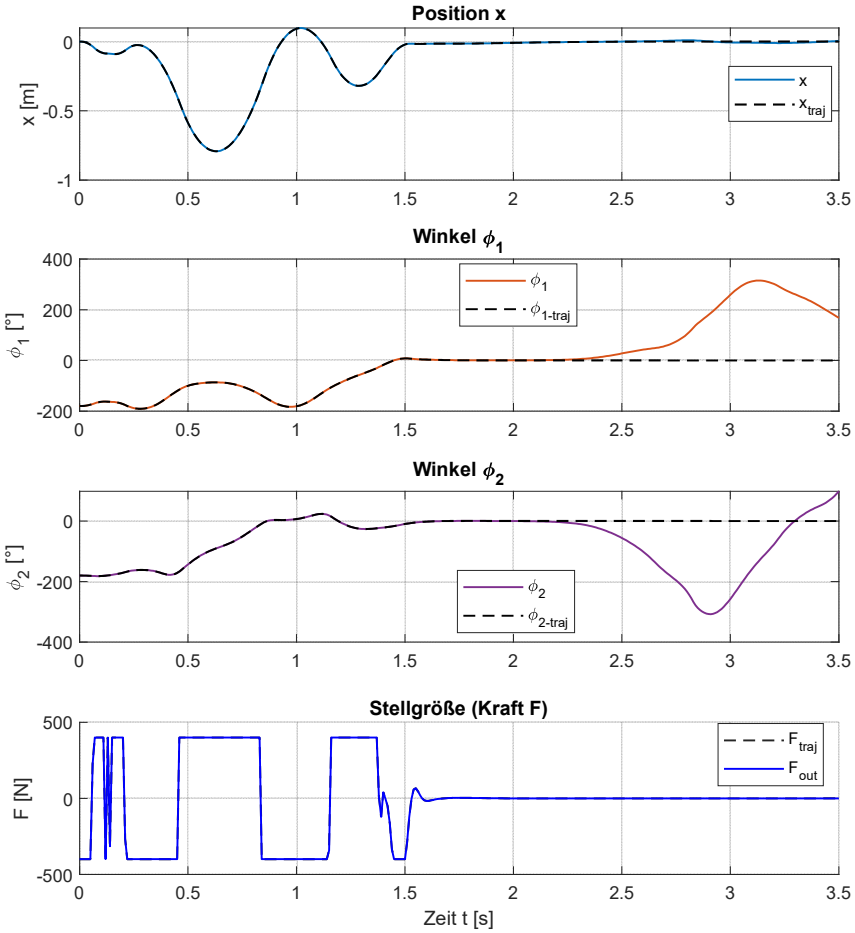


Abbildung 4.2: $T = 0{,}01$, $N = 350$, Apprich-Parameter, Euler-MPC, ode1-Sim, ohne Regelung

dem Euler-Verfahren am Schlittendoppelpendel simuliert. Die in Abbildung 4.2 dargestellten Ergebnisse zeigen die Verläufe der Ausgänge und der Stellkraft für das zunächst ungeregelte System. Es ist zu erkennen, dass der Aufschwung ohne Regelung gelingt, AP4 jedoch nicht gehalten wird. Das Ergebnis wird daher als *leicht instabil* beurteilt. In der anschließenden Simulation am geregelten System kann das Pendel schließlich bis zum Ende der Simulationszeit stabilisiert werden.

Mit den Verfahren `ode4` und `ode45` am ungeregelten System zeigen die Verläufe demgegenüber hohe Abweichungen von der Trajektorie. Erwartungsgemäß lassen sie sich anschließend am geregelten System nicht stabilisieren. Auch eine Variation der Simulationsschrittweite führt nicht zur Stabilisierung. Die Verläufe werden daher als *instabil* beurteilt.

Die berechnete Trajektorie lässt sich zwar stabilisieren, jedoch nur wenn die bei der Trajektorienberechnung gemachten Schrittfehler durch das gleiche Verfahren in der Simulation näherungsweise reproduziert werden. Dass die Steuerung alleine nicht ausreicht, obwohl mit gleicher numerischer Fehlerfortpflanzung simuliert wird, lässt sich einerseits damit begründen, dass die Nebenbedingungen des Optimierungsverfahrens mit bestmöglicher, jedoch endlicher Genauigkeit eingehalten werden. Die beobachteten Genauigkeiten für die Nebenbedingungen liegen zwischen 10^{-13} und 10^{-34} für (lokal) optimale Lösungen. Aufgrund numerischer Ungenauigkeiten erreicht das Pendel zudem auch bei ei-

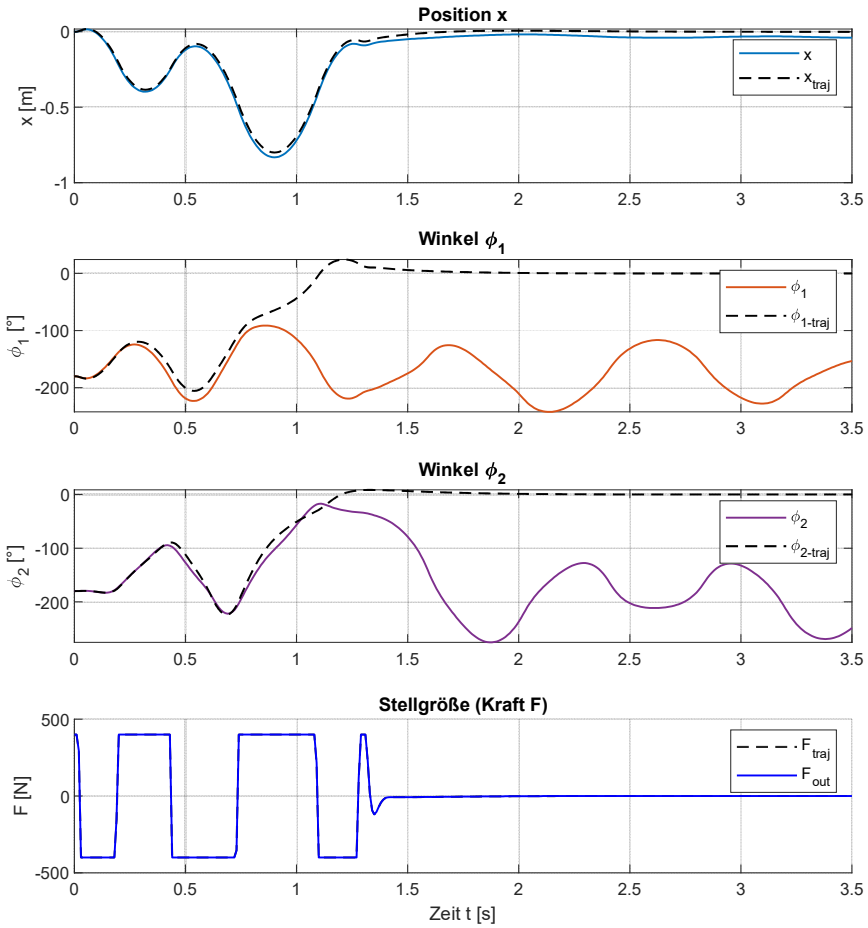


Abbildung 4.3: $T = 0{,}01$, $N = 350$, Ribeiro-Parameter, RK4-MPC, ode4-Sim, ohne Regelung

ner mittels Optimalsteuerungsentwurf berechneten Trajektorie den Arbeitspunkt AP4 nicht exakt und schwingt zurück [10]. Andererseits ist insbesondere zu beachten, dass der zu erreichende Endwert durch die Optimierung lediglich angenähert wird. Die im Rahmen der Arbeit beobachteten absoluten Fehler einzelner Zustandsgrößen im letzten Prädiktionsschritt lagen bei optimalen Lösungen im Bereich von 10^{-5} und $2 \cdot 10^{1}$.

Als weiteres Beispiel wird die für den Ribeiro-Parametersatz und das RK4-Verfahren berechnete Trajektorie in der Simulation mit `ode4` betrachtet. Die Verläufe sind für das ungeregelte System in Abbildung 4.3 zu sehen.

Es ist deutlich ersichtlich, dass der Aufschwung nicht geschafft wird. Das erste Pendel fällt bereits gleich zu Anfang des Aufschwungs zurück, sodass in der Folge beide Pendel unkontrolliert zu schwingen beginnen. Die Verläufe werden daher als *instabil* bewertet. In Abbildung 4.4 sind demgegenüber die Verläufe mit Trajektorienfolgeregelung zu sehen.

Das Doppelpendel kann demnach stabilisiert werden, jedoch ist in der Schlittenposition ein „Weglaufen“ zu beobachten. Dieses sorgt dafür, dass die vorgegebene Positionsbegrenzung weit überschritten wird. Dies legt nahe, dass durch eine höhere Bestrafung der Position mit Hilfe der QR-Parameter des Reglers eine Verringerung der maximalen Auslenkung zu erwarten ist. Mit einer Erhöhung von $Q_{11} = 1$ auf

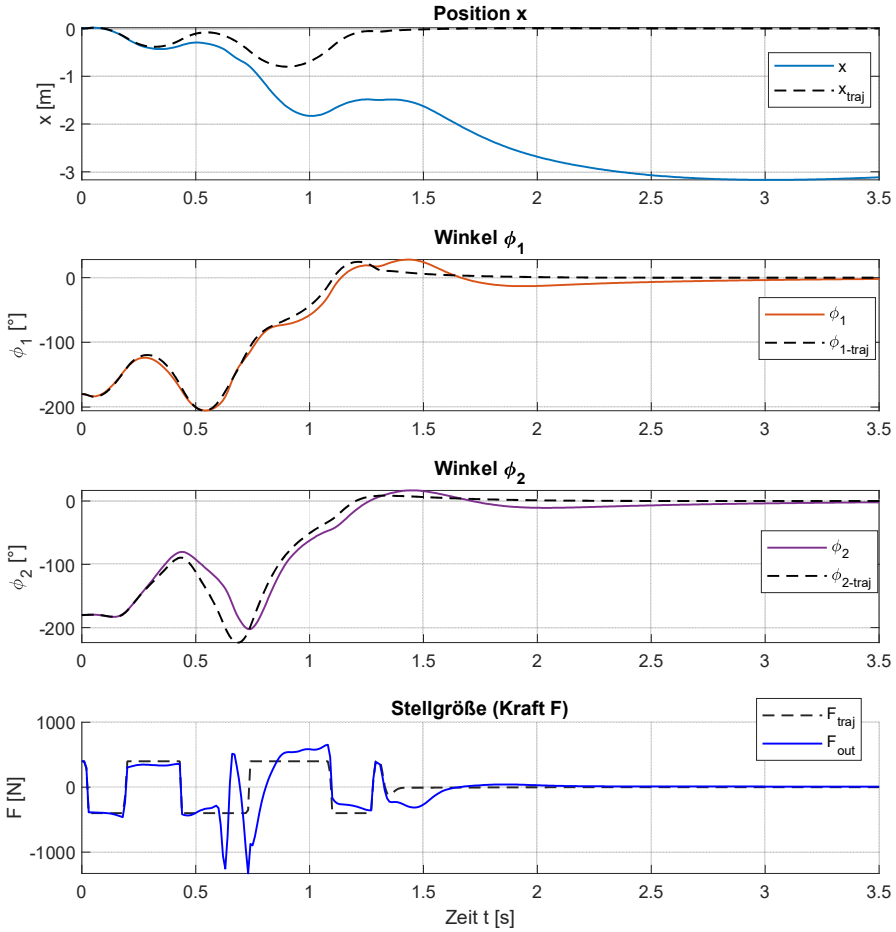


Abbildung 4.4: $T = 0{,}01$, $N = 350$, Ribeiro-Parameter, RK4-MPC, ode4-Sim, mit Regelung

wahlweise $Q_{11} = 1{,}05$ oder $Q_{11} = 4{,}5$ lässt sich die maximale Positionsauslenkung von etwa $-3{,}2\,\mathrm{m}$ auf $-2{,}5\,\mathrm{m}$ senken. Damit liegt die erreichte Auslenkung weiterhin weit außerhalb der zulässigen Begrenzung von $\pm 0{,}8\,\mathrm{m}$. Die durchgeführten Variationsversuche zeigen außerdem, dass die Stabilisierung der betrachteten Trajektorie empfindlich auf Veränderung der QR-Parameter reagiert. Bereits bei $Q_{11} = 1{,}07$ bzw. $Q_{11} = 4{,}6$ wird das System instabil. Da allgemein jedoch eine Stabilisierung erreicht wird, kann die Trajektorie für die Simulation `ode4` am geregelten System als *stabil* bewertet werden.

Nach dem beschriebenen Vorgehen werden auch die weiteren Simulationsversuche ausgewertet. Die Ergebnisse in Tabelle 4.4 zeigen, dass sich keine der betrachteten Trajektorien unter den beschriebenen Voraussetzungen numerisch stabil betreiben lässt.

Die Ergebnisse für $T = 0.005$ und $N = 500$ sind in Tabelle 4.4 dargestellt.

Tabelle 4.3: $T = 0.005,\ N = 500,\ F_{\max} = 400$

Simulation		**Apprich**		**Ribeiro**	
Solver	**TFR**	**Euler**	**RK4**	**Euler**	**RK4**
ode1	ohne	leicht instabil	leicht instabil	instabil	Trajektorie
	mit	**stabil**	**stabil**	instabil	
ode4	ohne	instabil	leicht instabil	instabil	nicht
	mit	instabil	**stabil**	instabil	
ode45	ohne	instabil	leicht instabil	instabil	gefunden
	mit	instabil	**stabil**	instabil	

Hierbei fällt auf, dass die Trajektorie, die mit Hilfe des Apprich-Parametersatzes und dem RK4-Verfahren berechnet wird, mit Hilfe des Reglers in allen drei Simulationen stabilisiert werden kann. Auch die Verläufe am ungeregelten System liefern bereits gute Ergebnisse. Die Stabilisierung durch den Regler erfolgt zudem innerhalb der vorgegebenen Positionsbegrenzung bei gleichzeitig geringer Abweichung von der Trajektorie. Auch bei Veränderung der QR-Parameter erweist sich die Stabilisierbarkeit der Trajektorie als unempfindlich, indem keine Destabilisierung im Rahmen der durchgeführten Variationen auftritt.

4.5.4 Mit Gegeninduktion

Wird die Gegeninduktion des Motors berücksichtigt und am Gesamtmodell simuliert, fällt zunächst auf, dass mit $F_{\max} = 400\,\mathrm{N}$ für den konstanten Bereich der Strombegrenzung keine der vier untersuchten Trajektorien gefunden wird. Durch eine Variation der Maximalkraftbegrenzung ergibt sich $F_{\max} = 410\,\mathrm{N}$ neue Grenze. Damit ist weiterhin eine Stellgrößenreserve von $11\,\mathrm{N}$ gegenüber der zuletzt am Versuchsstand eingestellten Maximalkraft von $F_{\max} = 421\,\mathrm{N}$ gewährleistet.

Für $T = 0.01$ und $N = 350$ kann dennoch keine der Trajektorien gefunden werden. Die Ergebnisse für $T = 0.005$ und $N = 500$ sind in Tabelle 4.4 zusammengetragen. Die Trajektorie auf Basis der Apprich-Parameter und des RK4-Verfahrens bestätigt die Ergebnisse aus Abschnitt 4.5.3. Sie ist ebenfalls numerisch stabil.

Durch Variation der QR-Parameter können die Simulationsergebnisse zudem noch weiter verbessert werden. Optimale Ergebnisse werden für folgende QR-Matrizen erzielt:

Tabelle 4.4: $T = 0.005,\ N = 500,\ F_{\max} = 410$

Simulation		Apprich		Ribeiro	
Solver	**TFR**	**Euler**	**RK4**	**Euler**	**RK4**
ode1	ohne mit	Trajektorie	instabil **stabil**	leicht instabil **stabil**	Trajektorie
ode4	ohne mit	nicht	instabil **stabil**	instabil instabil	nicht
ode45	ohne mit	gefunden	instabil **stabil**	instabil instabil	gefunden

Tabelle 4.5: Neue QR-Matrizen für Apprich-RK4-Trajektorie

ode1	$\mathbf{Q} = \mathrm{diag}\,(1, 1, 100, 1, 100, 1)$	$R = 0{,}01$
ode4, ode45	$\mathbf{Q} = \mathrm{diag}\,(1000, 0.01, 100, 0.1, 100, 0.1)$	$R = 0{,}001$

Die Ausgangsverläufe für die Simulation mit `ode45` und den neuen QR-Parametern sind in Abbildung 4.5 dargestellt. Im Vergleich zu den Plots in Abschnitt 4.5.3 wird im Stellgrößenverlauf zusätzlich zwischen F_{in} und F_{out} unterschieden. F_{in} bezeichnet den durch Regler und Trajektorie festgelegten Stellwert, der auf das Gesamtsystem gegeben wird. Auf Grund der Strombegrenzungskennlinie (vgl. Abbildung 2.3) kann am Ausgang des Motors jedoch ein abweichender Kraftverlauf entstehen, der mit F_{out} dargestellt wird. Da die Stromkennlinie in den Nebenbedingungen der Trajektorienberechnung berücksichtigt wird, entstehen Abweichungen zwischen F_{in} und F_{out} in erster Linie auf Grund des Reglers.

Fazit: Es lässt sich zeigen, dass die mit dem Verfahren der NMPC berechneten Trajektorien in der Simulation stabilisierbar sind. Zudem bestätigt sich die Vermutung, dass die Schrittweite und die Wahl des Integrationsverfahrens zur Berechnung der Trajektorien einen Einfluss auf die numerische Stabilität haben. Eine geringere Schrittweite und ein Integrationsverfahren mit geringerem Schrittfehler haben in den betrachteten Versuchsdurchführungen zu einer höheren Stabilität geführt. Darüber hinaus fällt auf, dass die Apprich-Parameter gegenüber den Ribeiro-Parametern ein günstigeres Verhalten aufweisen. Der Einfluss der Modellparameter wird im folgenden Kapitel daher näher untersucht.

4.6 Untersuchung des Einflusses der Modellparameter

Im Folgenden soll der Einfluss der Modellparameter auf die Berechnung der Trajektorien untersucht werden.

4.6.1 Vorgehen

Gegenstand der Untersuchung sind die Masse, das Massenträgheitsmoment und der Schwerpunkt jedes Pendelstabs. Die gewählten Wertebereiche für die Parametervariationen orientieren sich an den Werten der zum Versuchsstand aufgestellten Parametersätze aus Abschnitt 2.3.3. Hierbei wird die konstruktive

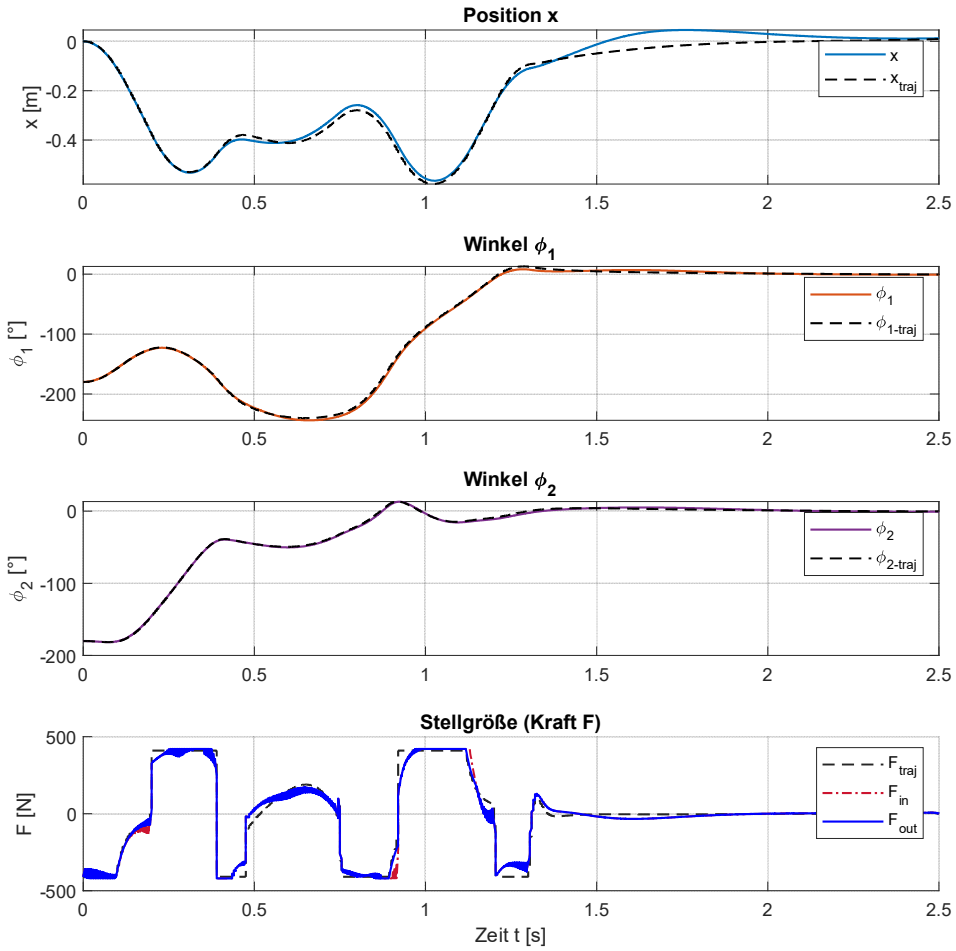


Abbildung 4.5: $T = 0{,}005$, $N = 500$, Apprich-Parameter, RK4-MPC, ode45-Sim, mit Regelung

Realisierbarkeit von Parameterwerten für den Versuchstand bewusst vernachlässigt, da in erster Linie der Einfluss der Modellparameter auf die Berechnungsergebnisse untersucht werden soll.

Die Schrittweite wird möglichst gering gewählt, um die Aussagekraft der Ergebnisse durch eine hohe Auflösung sicherzustellen. Auf Grund der hohen Berechnungszeiten für die Trajektorien, sind beliebig kleine Schrittweiten jedoch nicht möglich, sodass ein sinnvoller Kompromiss zwischen Auflösung und Berechnungszeit gefunden werden muss. Das geplante Variationsvorhaben ist in Tabelle 4.6 aufgeführt.

Tabelle 4.6: Parametervariationen

Parameter	Startwert	Endwert	Schrittweite	Anzahl Trajektorien
m_1 [kg]	0	2	0,01	201
m_2 [kg]	0	2	0,01	201
J_1 [kgm^2]	0	0,02	0,0001	201
J_2 [kgm^2]	0	0,02	0,0001	201
s_1 [m]	0	0,29	0,001	291
s_2 [m]	0	0,338	0,001	339
				1434

Zur Untersuchung des Einflusses einzelner Modellparameter werden diese gegenüber einem definierten Ausgangsparametersatz variiert, wobei alle weiteren Parameter konstant gehalten werden. Zur Durchführung der Versuche wird die Funktion `examParameters` implementiert. Ihr kann eine Versuchsreihe bestehend aus der Bezeichnung des zu untersuchenden Parameters und einem Vektor mit den Variationswerten des Parameters übergeben werden. Über den Aufruf der in Abschnitt 4.2.1 beschriebenen Funktion `searchTrajectories` werden damit die benötigten Trajektorienberechnungen ausgeführt. Die berechneten Trajektorien werden anschließend im Ordner `ParameterExams` gespeichert. Um die Ergebnisse später sinnvoll identifizieren zu können, wird der in Abschnitt 4.2.1 definierten Namenskonvention eine Endung bestehend aus der Bezeichnung des untersuchten Parameters und dessen Wert angefügt.

Zur Bewertung der betrachteten Parametervariationen wird zunächst unterschieden, ob für die jeweilige Variation eine Trajektorie gefunden werden kann oder nicht. Darüber hinaus wird das Gütemaß J_{dev} eingeführt, um die Güte der gefundenen Lösungen zu quantifizieren. Der Ansatz für das Gütemaß basiert auf der Definition von Trajektorien durch ihre Randwerte. Da diese durch die Optimierung lediglich angenähert werden (vgl. Abschnitt 4.5), ist der Fehler durch die Näherung der Anfangs- und der Endwertbedingung ein Kriterium für die Güte der Trajektorien.

Das Gütemaß wird daher mit

$$J_{\text{dev}} = \|\Delta \boldsymbol{x}_{\mathbf{init}}\| + \|\Delta \boldsymbol{x}_{\mathbf{end}}\|$$

definiert. $\Delta \boldsymbol{x}_{\mathbf{init}}$ und $\Delta \boldsymbol{x}_{\mathbf{end}}$ sind hierbei als die euklidischen Normen der mit den Fehlerdifferenzen gefüllten Zustandsvektoren in Anfangs- und Endlage der Trajektorie zu verstehen.

$$\|\Delta \boldsymbol{x}_{\mathbf{init}}\| = \sqrt{\sum_{i=1}^{6} \left(x_{\text{Traj,i}}(t_{\text{init}}) - x_{\text{init,i}}\right)^2}$$

mit $t_{\text{init}} = 0$ und

$$\|\Delta \boldsymbol{x}_{\mathbf{end}}\| = \sqrt{\sum_{i=1}^{6} \left(x_{\text{Traj,i}}(t_{\text{end}}) - x_{\text{end,i}}\right)^2}$$

mit $t_{\text{end}} = (N+1) \cdot T$

Wie in Abschnitt 4.5 erläutert, verhält sich der Endwertfehler dominant, sodass im Allgemeinen

$$J_{\text{dev}} \approx \|\Delta \boldsymbol{x}_{\mathbf{end}}\|$$

angenommen werden kann.

Basierend auf den Erkenntnissen des vorherigen Abschnitts wird folgende Ausgangskonfiguration für die Berechnung der Trajektorien gewählt:

- $T = 0{,}005$
- $N = 500$
- RK4-Integration
- Apprich-Parameter

Als Trajektorie wird die in Abschnitt 4.5.2 definierte Vergleichstrajektorie verwendet. Die Gewichtungsmatrizen für die Zielfunktion sind Abschnitt 4.2.2 zu entnehmen.

4.6.2 Ergebnisse

Die Auswertung erfolgt mit Hilfe der Funktion `plot_Jdev_params`. Mit dieser werden die Gütemaße berechnet und über dem untersuchten Parameter in einem Koordinatensystem aufgetragen. Hierbei werden auch die Werte des Apprich- und des Ribeiro-Parametersatzes nach Abschnitt 2.3.3 als Vergleichswerte durch die Funktion hinzugefügt. Berechnete Trajektorien ohne lokale Konvergenz zu einem Optimum werden als ungültig gewertet und durch ein rotes.

Die Ergebnisse für die Parametervariationen + jeweils Ribeiro und Apprich Variation, falls nicht inklusive

s1 in Berechnung...

Diskussion in Bearbeitung...

Tabelle 4.7: Statistik gültiger Trajektorien

Parameter	Unültig	Gültig	Gesamt	Gültigkeitsanteil
m_1 [kg]	177	26	203	13%
m_2 [kg]	189	14	203	7%
J_1 [kgm^2]	175	28	203	14%
J_2 [kgm^2]	79	124	203	61%
s_1 [m]				
s_2 [m]	190	150	340	44%

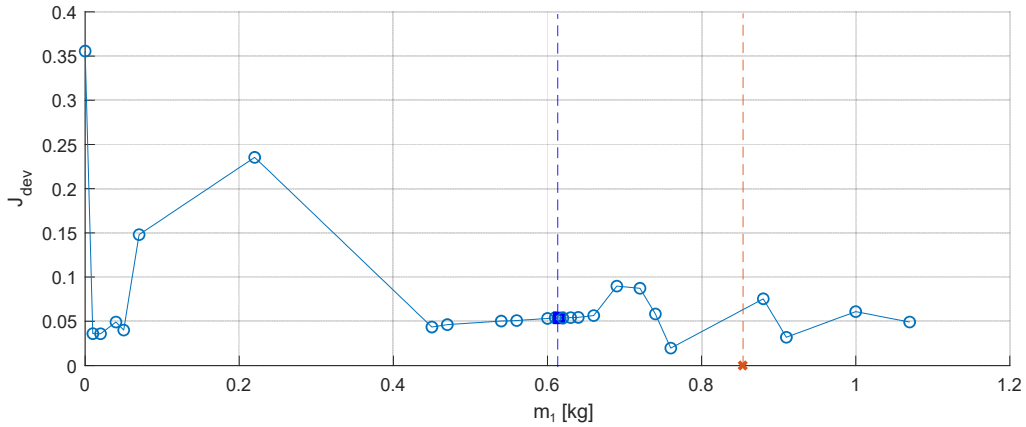


Abbildung 4.6: Variation m_1

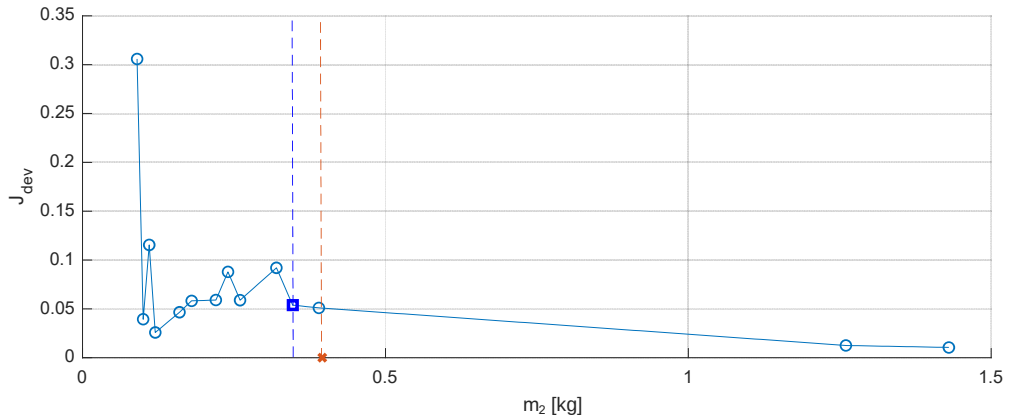


Abbildung 4.7: Variation m_2

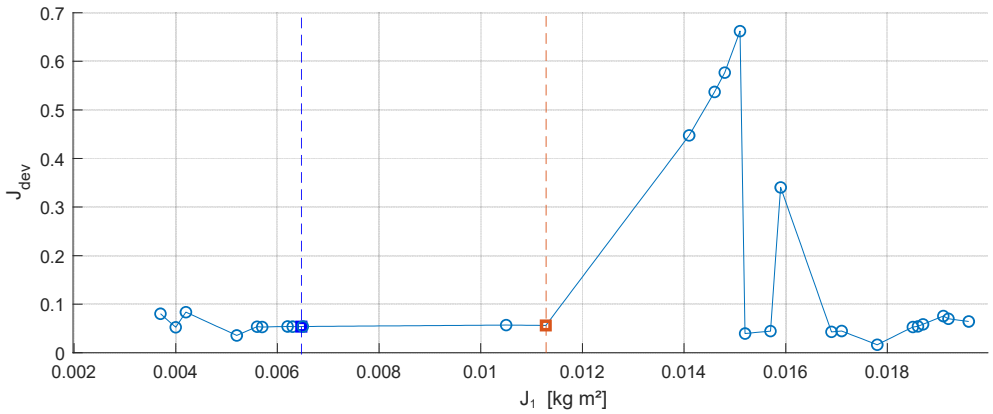


Abbildung 4.8: Variation J_1

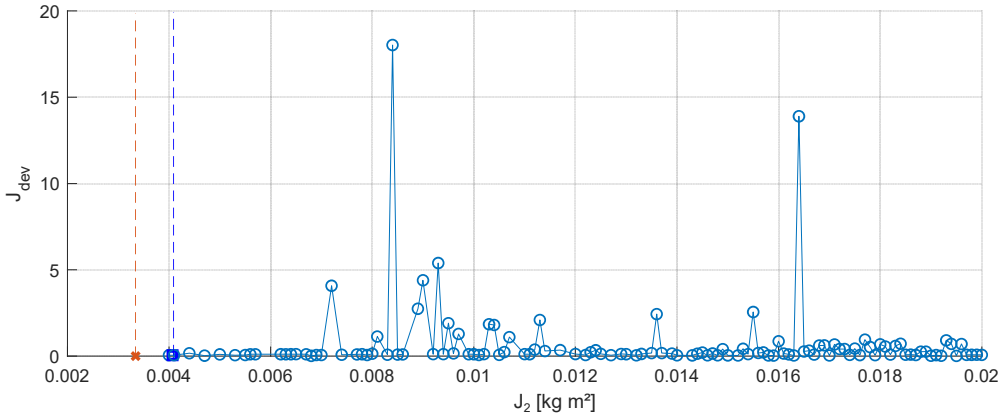


Abbildung 4.9: Variation J_2

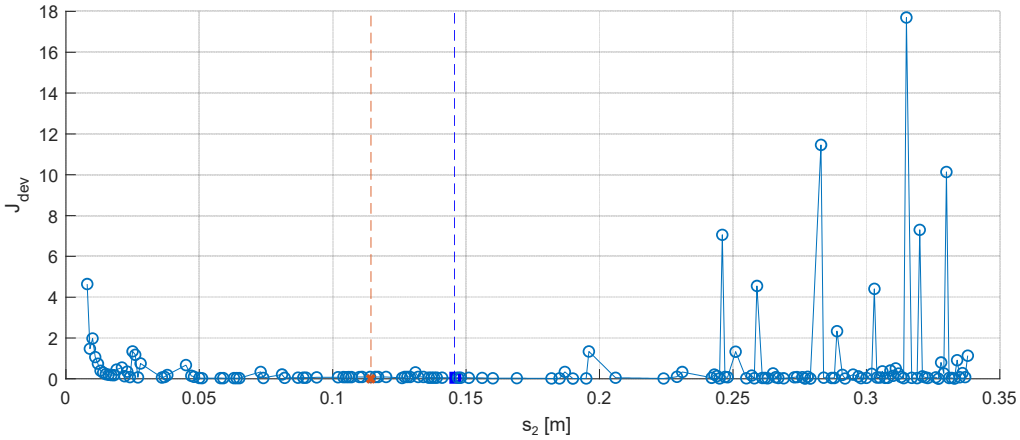


Abbildung 4.10: Variation s_2

5 Fazit und Ausblick

5.1 Zusammenfassung und Fazit

5.2 Ausblick

A AP-Regelung Systemparametertests

Literatur

[1] Jürgen Adamy. *Systemdynamik und Regelungstechnik II*. Shaker, 2019.

[2] Stefanie Apprich. *Konstruktion und Regelung eines inversiven Doppelpendels*. Studienarbeit. Technische Universität Darmstadt, 2009.

[3] Andreas Binder. *Elektrische Maschinen und Antriebe - Grundlagen, Betriebsverhalten*. Darmstadt: Springer-Verlag Berlin Heidelberg, 2012.

[4] Daniel Brehl. *Implementierung und Vergleich verschiedener Reglervarianten zur Regelung eines inversen Doppelpendels*. Bachelorarbeit. Technische Universität Darmstadt, 2014.

[5] Nicolas Fauvé. *Nichtlineare MPC-Regelung eines Doppelpendels*. Studienarbeit. Technische Universität Darmstadt, 2020.

[6] Andreas Franke. *Modellierung und Regelung eines mechatronischen Systems - Eine Realisierung des invertierten Pendels mit momentgeregeltem Antriebsmotor*. Diplomarbeit. Technische Universität Clausthal, 1997.

[7] Andreas Kämmerer. *Inbetriebnahme und Regelung eines inversen Doppelpendels*. Studienarbeit. Technische Universität Darmstadt, 2009.

[8] Viktor Kisner. *Trajektorienfolgerelung eines inversen Doppelpendels*. Bachelorarbeit. Technische Universität Darmstadt, 2011.

[9] Ulrich Konigorski. *Modellbildung und Simulation, Skript*. Darmstadt, 2019.

[10] Ulrich Konigorski. *Praktikum MATLAB/Simulink II*. Darmstadt, 2019.

[11] Eric Lenz. *Modellierung Pendel*. Technische Universität Darmstadt, Fachgebiet rtm.

[12] Karl Max Crépin Noupa. *Reibkompensation am Doppelpendel*. Bachelorarbeit. Technische Universität Darmstadt, 2011.

[13] Igor de Sousa Ribeiro; Thomas Gassen; Vitaliy Omelchuk. *Steuerung und Regelung eines inversen Doppelpendels*. Projektseminar. Technische Universität Darmstadt, 2020.

[14] Feiyu Chang; Nicolas Fauvé; Antonia Gießler; Charles Studd Tchoutchui. *Konstruktion und Regelung eines inversen Doppelpendels*. Projektseminar. Technische Universität Darmstadt, 2019.