Regelung eines Hebelarms

Aufgabenstellung

Freiwilliges praktisches Projekt
Systemtheorie II
Periode 1

WS 22/23

Kontaktadresse:

acs-teaching-sys2@eonerc.rwth-aachen.de

Inhaltsverzeichnis

1 Einleitung

Vorsicht: Es besteht Quetschgefahr. Stellen Sie jederzeit sicher, dass sich der an dem Motor befestigte Hebelarm frei bewegen kann.

Verwenden Sie die freie Fläche hinter dem Arduino, um den Aufbau mit einem Gewicht (z.B. einem Buch) zu stabilisieren.

Ein seperates Dokument mit Hilfestellungen zur Konfiguratrion der Hard- und Software finden Sie im Moodle Lernraum (Setup-Hilfe).

Ein seperates Dokument mit einer Einführung zu MATLAB Simulink finden Sie im Moodle Lernraum (MATLAB-Simulink-Einführung).

Die Simulation ist von der verwendeten Computerhardware abhängig. Es wird empfohlen, vor dem Simulationsstart alle gewünschten Scopes zu öffnen und während der Simulation den Computer so wenig wie möglich auszulasten (insbesondere keine rechenaufwendigen Programme, wie etwa Zoom). Andernfalls kann es zum Aufschwingen des Modells kommen. Abhilfe schafft das Neustarten der Simulation.

Diese Aufgabe ist zum Verständnis und zur Vertiefung der in der Vorlesung und Übung Systemtheorie II gelehrten Inhalte gedacht. Bei erfolgreichem Bearbeiten der Aufgabe erhalten die Studierenden Bonuspunkte in Höhe von 0,3 Notenstufen für die Klausur der Systemtheorie II (Winteresemester 22/23 und Sommersemester 2023). Die Bonuspunkte werden den in der Klausur erzielten Punkte aufaddiert, auch wenn die Klausur ohne Anrechnung der Bonuspunkte nicht bestanden wurde. Teilweise gehen die Inhalte der Aufgaben über diejenigen der Vorlesung hinaus. Trifft das zu, werden entsprechende Hinweise bzw. Lösungshilfen gegeben.

Die gesamte Hardware ist Eigentum des ACS und sie muss unbeschädigt und unverändert am Lehrstuhl wieder abgegeben werden. Beschädigte Teile müssen auf eigene Kosten ersetzt werden.

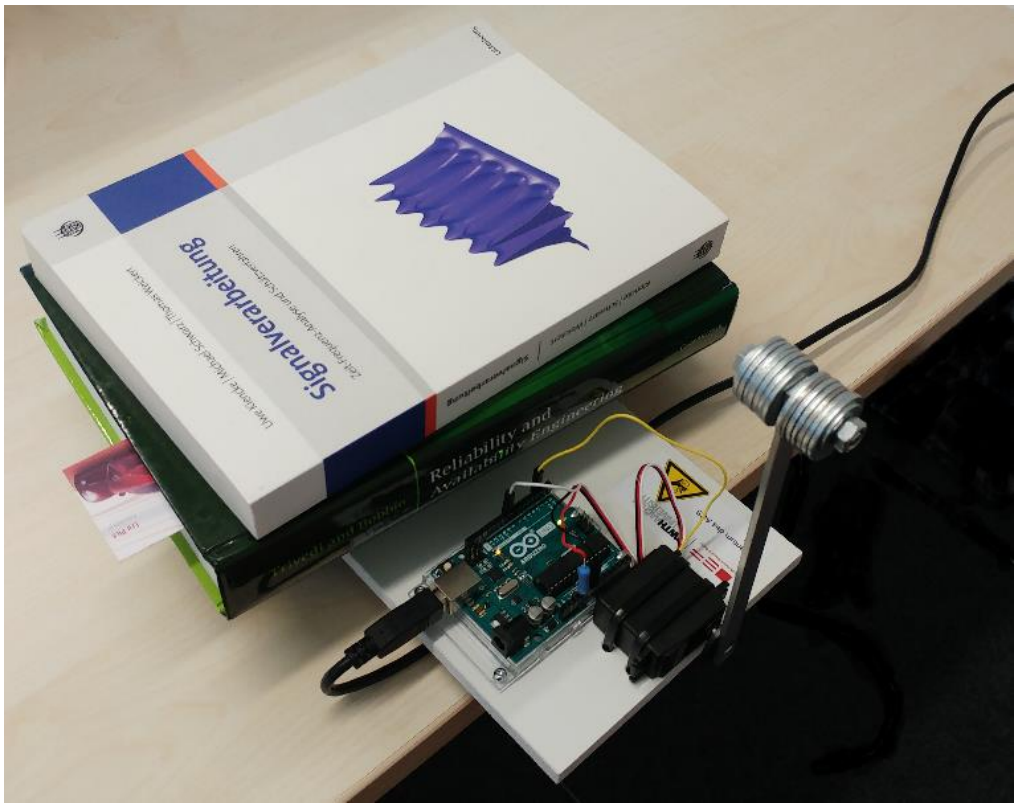
Die Bearbeitung erfolgt in zwei Zeiträumen, 14.11.2022-14.12.2022 und 16.12.2022-31.01.2023. Für den ersten Zeitraum kann die Hardware am 10.11. im ACS Hauptgebäude in der Mathieustr. 10 abgeholt werden und am 15.12. zurückgegeben werden. Für den zweiten Zeitraum erfolgt die Hardwareausgabe am 15.12. und die Rückgabe am 03.02.2022.

Die Lösung ist als schriftliche Ausarbeitung bis spätestens zum **14. Dezember 2022** für den ersten Zeitraum und bis zum **31 Januar 2022** für den zweiten Zeitraum über eine entsprechende Upload-Funktion auf RWTH Moodle einzureichen. Einreichungen nach dieser Frist werden nicht akzeptiert! Die Lösung muss je nach Aufgabenteil eine nachvollziehbare Lösung (mit Lösungsweg), Screenshots von Plots, Simulink Schaltplänen und/oder Beschreibungen enthalten. Der Gesamtumfang der eingereichten Dokumente sollte 15 Seiten nicht überschreiten. Ein Template im Word-Format zur Anfertigung der schriftlichen Ausarbeitung wird auf RWTH Moodle bereitgestellt. Die Einreichung (d.h. das Hochladen) der Lösung sollte dabei vorzugsweise im PDF-Format erfolgen.

1.1 Versuchsbeschreibung

In dieser Aufgabe soll mit MATLAB Simulink eine Positionsregelung für einen Hebelarm ausgelegt, implementiert und dann in einem „Hardware-in-the-Loop“ (HiL) Aufbau getestet werden. Der erste Aufgabenteil dient als Einführung in die Hardware. Im zweiten Aufgabenteil werden verschiedene Störungen auf den Hebelarm identifiziert und modelliert. Teil drei der Aufgabe befasst sich mit der Entwicklung eines Reglers zur Positionsregelung des Hebelarms und in Aufgabenteil vier werden verschiedene Störungssignale mit dem Sollwert überlagert, welche dann kompensiert werden müssen. Es wird empfohlen, die Aufgaben in dieser Reihenfolge zu bearbeiten.

Es wird ein Arduino und ein Servomotor[1], montiert auf einer Kunststoffplatte (siehe Bild unten), bereitgestellt. Der Motor ist elektrisch in seinem maximalen Drehmoment begrenzt, um eine Zerstörung bei Überlastung zu vermeiden. Überschreiten der Drehmomentgrenze hat einen sogenannten Brownout der Motorsteuerung zu Folge. Bei einem Brownout schaltet die Motorsteuerung ab und nach wenigen Sekunden wieder ein. Der Brownout ist zu vermeiden, da sich der Motor trotz Schutzbeschaltung stark erhitzen kann.

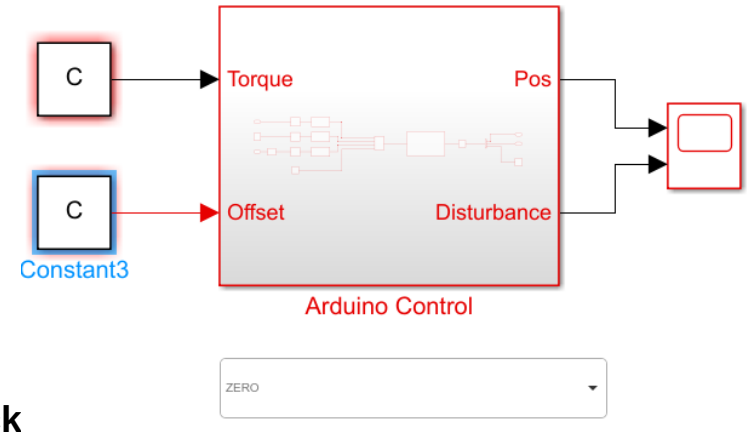


Exemplarisches Hardware-Setup aus dem WS19/20: Hebelarm als „Inverses Pendel“ mit Gewichten

[1] https://www.mouser.de/datasheet/2/321/900-00360-Feedback-360-HS-Servo-v1.2-1147206.pdf

1.2 Bereitgestellter Simulink Block

Ein seperates Dokument mit einer Einführung zu MATLAB Simulink finden Sie im Moodle Lernraum (MATLAB-Simulink-Einführung).



1.2.1 Arduino Control Block

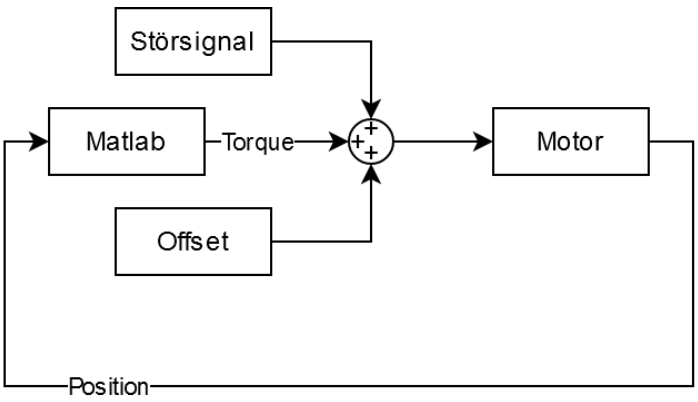
Der Arduino Block stellt ein serielles Interface zwischen MATLAB und dem Arduino bereit. Der Block verfügt über zwei Eingänge und zwei Ausgänge. *Pos* gibt die aktuelle absolute Position aus. *Disturbance* das aktuelle Störsignal. Das *Pos* Signal muss auf den Wertebereich 0-360 Grad abgebildet werden.

Der Eingang *Torque* ist ein Parameter der proportional zu dem am Motor eingestellten Drehmoment ist.

Der Eingang *Offset* korrigiert eine Asymmetrie zwischen positiven und negativen *Torque* Werten.

Wichtig: Der Eingangsdatentyp für *Torque* und *Offset* muss jeweils zu „int32“ festgelegt sein (z.B. zu definieren innerhalb der beiden „Constant“-Blöcke wie oben abgebildet)!

Der Zusammenhang zwischen den Parametern ist im folgenden Bild veranschaulicht.



2 Aufgabenstellungen

Ein seperates Dokument mit Hilfestellungen zur Konfiguratrion der Hard- und Software finden Sie im Moodle Lernraum (Setup-Hilfe).

2.1 Aufgabe 1: Aufbau und grundlegende Motorsteuerung

Verbinden Sie den Servomotor entsprechend der folgenden Tabelle mit dem Arduino.

Servomotor	Arduino
Rot mit Widerstand	+5V
Schwarz	GND
Weis	Pin 6
Gelb	Pin 3

System-Voraussetzungen:

- Windows 7/10 mit MATLAB 2020b oder neuer einschließlich des Simulink Moduls (getestet)

- Für MATLAB Simulink:

  - Simulink Coder (bei der Installation auszuwählen)
  - Embedded Coder (bei der Installation auszuwählen)
  - (möglicherweise weitere Abhängigkeiten)

- Zusätzlich werden folgende MATLAB Support Packages benötigt:

  - MATLAB Support for MinGW-w64 C/C++ Compiler (Installation über MATLAB Add-Ons)

(Es wird empfohlen für die Installation der Support Packages MATLAB als Administrator zu starten.)

- Installation der Arduino IDE (einschließlich Treiber):

  - https://www.arduino.cc/en/software

- Per USB mit dem Computer verbundener Arduino Uno mit ST2-Projekt-Firmware (siehe separates Setup-Hilfe Dokument)

Für diesen Aufgabenteil wird der Arduino Control Block verwendet. Es werden die entsprechenden Blöcke für die Auswertung und Steuerung mir einer Konstante benötigt.

Hinweis:

Für die Messung der aktuellen Motorposition wird ein angepasster Simulink Block (Arduino Control) als Modell „base_model_2020b.slx“ bereitgestellt. Hier muss die S-Function innerhalb des Simulink Blocks angepasst werden. Es muss der entsprechende COM Port, an dem der Arduino angeschlossen ist, eingestellt werden. Weitere Informationen hierzu finden Sie im Setup-Hilfe Dokument im Moodle Lernraum.

Achtung: Die Torque-Einstellung kann aufgrund der Hardware unsymmetrisch sein.

Aufgabe 1.1

- Verbinden Sie den „Arduino Control“ Simulink Block zunächst mit Scopes sowie konstanten Signalquellen und stellen Sie als Störsignaltyp „Zero“ im Drop-Down Menü des „Arduino Control“ Simulink Blocks ein.
- Stellen Sie ein festes *Torque* (τ_{in}) von 5 ein.
- Machen Sie sich mit dem Positionssignal vertraut (Form, Frequenz …)
- Stellen Sie nun eine Umlaufzeit von 5 Sekunden ein (im Uhrzeigersinn).
- Nehmen Sie einen Screenshot des Positionssignals auf und beschreiben Sie den Verlauf sowie markante Punkte des Signals.

Aufgabe 1.2

- Bilden sie mit Hilfe eines Umrechnungsfaktors sowie einer Nullpunktverschiebung das *Pos* Signal auf 360⁰ ab. Die untere Gleichgewichtslage soll 180⁰ entsprechen.
- Dokumentieren Sie Ihre Ergebnisse.

Aufgabe 1.3

- Stellen Sie nun $\tau_{in} = 0$ ein.
- Bringen Sie den Hebelarm in die obere Gleichgewichtslage.
- Stellen Sie den *Offset* (τ_{offset}) so ein, dass sich der Hebelarm nicht aus der oberen Gleichgewichtslage entfernt.
- Bestimmen Sie mit welcher maximalen Frequenz das System geregelt werden darf. *Hinweis: Betrachten Sie das Positionssignal.*
- Was ist demnach die kleinste mögliche Regelzeitkonstante dieses Systems?

2.2 Positionsregelung des Hebelarms

Ziel ist die Regelung der Position des Hebelarms.

2.2.1 Aufgabe 2: Identifikation der Störfunktion

Falls sich in dieser Aufgabe der Hebelarm nicht dauerhaft um die untere Ruhelage bewegen sollte, korrigieren Sie bitte den Offset-Faktor aus Aufgabenteil 2.2.1 um +/-1 bezüglich des identifizierten Offset-Werts.

Bestimmen Sie zunächst den Übertragungsfaktor zwischen dem *Torque* Eingang und dem *Pos* Ausgang des Arduino Control Blocks. Untersuchen Sie folgende Werte für den Torque:

τ_{in} : -20, -10, -8, -5, 5 ,8, 10, 20

Aufgabe 2.1

- Stellen Sie *Torque* auf die verschiedenen Werte ein und untersuchen sowie dokumentieren Sie *Pos* sowie die Ableitung von *Pos*.
- Erklären Sie, wie möglicherweise auftretende Sprünge zustande kommen. *Hinweis: Betrachten Sie das Positionssignal sowie die Ableitung im Bereich der oberen Gleichgewichtslage.*

Aufgabe 2.2

- Erklären Sie, wieso die Ableitung nicht immer konstant ist, sondern zwischen Null und einem Wert x springt.
  Hinweis: Betrachten Sie das Positionssignal sowie die Ableitung im Bereich der unteren Gleichgewichtslage.

Aufgabe 2.3

- Bestimmen Sie für die verschiedenen τ_{in} die Amplitude der Ableitung im Bereich der unteren Gleichgewichtslage.
- Bestimmen Sie entsprechend einen Proportionalitätsfaktor zwischen *Torque* und der Ableitung von *Pos* im Bereich um die untere Ruhelage. Gehen Sie auf mögliche Nichtlinearitäten ein.

Bestimmen Sie nun die verschiedenen Fehlersignale:

- Setzen Sie $\tau_{in} = 0$ und stellen Sie die in der Tabelle auf der nächsten Seite aufgelisteten Störsignaltypen im Simulink-Modell ein.
- Bestimmen und dokumentieren Sie für jedes Störsignal den Signaltyp, Amplitude und Frequenz gemäß der Spalte „Signalbeschreibung“ in der Tabelle.
- *Hinweis: Betrachten Sie mehrere Perioden des Störsignals und bestimmen Sie ggf. Mittelwerte, um Ungenauigkeiten des realen Systems zu verringern.*

Störsignaltyp	**Signalbeschreibung**
ZERO	Kein Störsignal vorhanden.
TYPE1	
TYPE2	
TYPE3	
TYPE4	
TYPE5	
TYPE6	
TYPE7	
TYPE8	
TYPE9	

Aufgabe 2.4

1. Welche maximale Störsignalfrequenz kann bei den verwendeten Simulationseinstellungen gemessen werden? Begründen Sie Ihre Aussage mathematisch und zeigen Sie anhand eines der Signale, welches Problem dabei auftritt.

2.2.2 Aufgabe 3: Positionsregelung

In dieser Aufgabe wird ein Regler zur Positionsregelung des Hebelarms entworfen.

Aufgabe 3.1

- Setzen Sie für τ_{offset} den in Aufgabe 2 ermittelten Wert ein.
- Stellen Sie Störsignaltyp „Zero“ im Simulink-Modell ein.
- Im ersten Schritt wird ein P-Regler entworfen, der die Position des Hebelarms in der unteren Hälfte auf eine beliebige Position einstellen kann. Wir betrachten die Regelung zuerst auf zwei Positionen und zwar 160° und 200°. Für die Implementierung des Reglers wird die Verwendung des „PID Controller“ Blocks aus der Simulink Library empfohlen.
- Stellen Sie zuerst ein Verstärkungsfaktor von 0,1 ein. Dokumentieren und erklären Sie den stationären Regelfehler.
- Erhöhen Sie in +0,1 Schritten den Verstärkungsfaktor und untersuchen Sie jeweils einen Positionswechsel zwischen 160° und 200°. Untersuchen Sie das Systemverhalten. Kann der stationäre Fehler auf diese Weise eliminiert werden? Wie ändert sich das Systemverhalten in Bezug auf den stationären Fehler sowie den Einschwingvorgang?
- Dokumentieren Sie das Systemverhalten für die verschiedenen Verstärkungsfaktoren.

Aufgabe 3.2

- Entwerfen Sie einen Regler mit Hilfe der zeitdiskreten Einstellregeln. Wählen Sie den aus Ihrer Sicht besten Reglertyp (P, PI, PID) aus und begründen Sie ihre Wahl. *Hinweis: Es sind keine mathematischen Herleitungen erforderlich und Sie können von der untenstehenden Tabelle Gebrauch machen. Als „Integrator Method“ des „PID Controller“ Blocks verwenden Sie am besten das Ihnen bekannte Verfahren nach Tustin.*
- Vergleichen, beschreiben und dokumentieren Sie die Unterschiede zwischen dem Regler nach dem ersten und Teil dieser Aufgabe.

Regler	K_R	T_I	T_D
P	$0{,}5 \cdot K_{R,krit}$	–	–
PI	$0{,}45 \cdot K_{R,krit}$	$0{,}83 \cdot T_{krit}$	–
PID	$0{,}6 \cdot K_{R,krit}$	$0{,}5 \cdot T_{krit}$	$0{,}125 \cdot T_{krit}$

2.2.3 Aufgabe 4: Kompensation einer Störung

In diesem Aufgabenteil wird untersucht, wie eine Störung am Eingang der Regelstrecke ausgeregelt werden kann.

- Verwenden Sie Ihre Reglerparameter aus Aufgabe 3.2.
- Kompensieren die verschiedenen Störungen (TYPE 1 – TYPE8), indem Sie Ihre Regelkreis-Implementierung geeignet modifizieren.
- *Hinweis: Verwenden Sie den „Disturbance“ Ausgang des „Arduino Control“ Blocks.*
- Untersuchen, vergleichen und dokumentieren Sie den Einfluss der verschiedenen Störungen (TYPE1 - TYPE8) auf die Regelung (d.h. jeweils mit und ohne Kompensation).
- Erläutern Sie, welche Rolle die Totzeit der Kommunikation zwischen dem Arduino und MATLAB Simulink bei der Kompensation der Störung spielt.