Michele Vivaldelli, Florian Zurbriggen, Gehlen Manuel,
Simon Wieser, Jasmina Saxer, Jan Mathys,
Dominic Seibold

Wippe

Praktikum Mess- und Regeltechnik
Anleitung zum Versuch

Institute for Dynamic Systems and Control
Swiss Federal Institute of Technology Zurich

February 19, 2024

IDSC

Contents

1 Einführung **1**

2 Hausaufgaben **3**
2.1 Modell Herleitung 3
2.2 Lineares Modell 6
2.3 Positionsregler 7

3 Lab Praxis **9**
3.1 SISO Regler Anwendung 9
3.2 LQ-Regler 10
3.3 Observer - LQG 12
3.4 Vollständiger Zustandsraum - LQG 14
3.5 Integral Verhalten - LQGI 15

1 Einführung

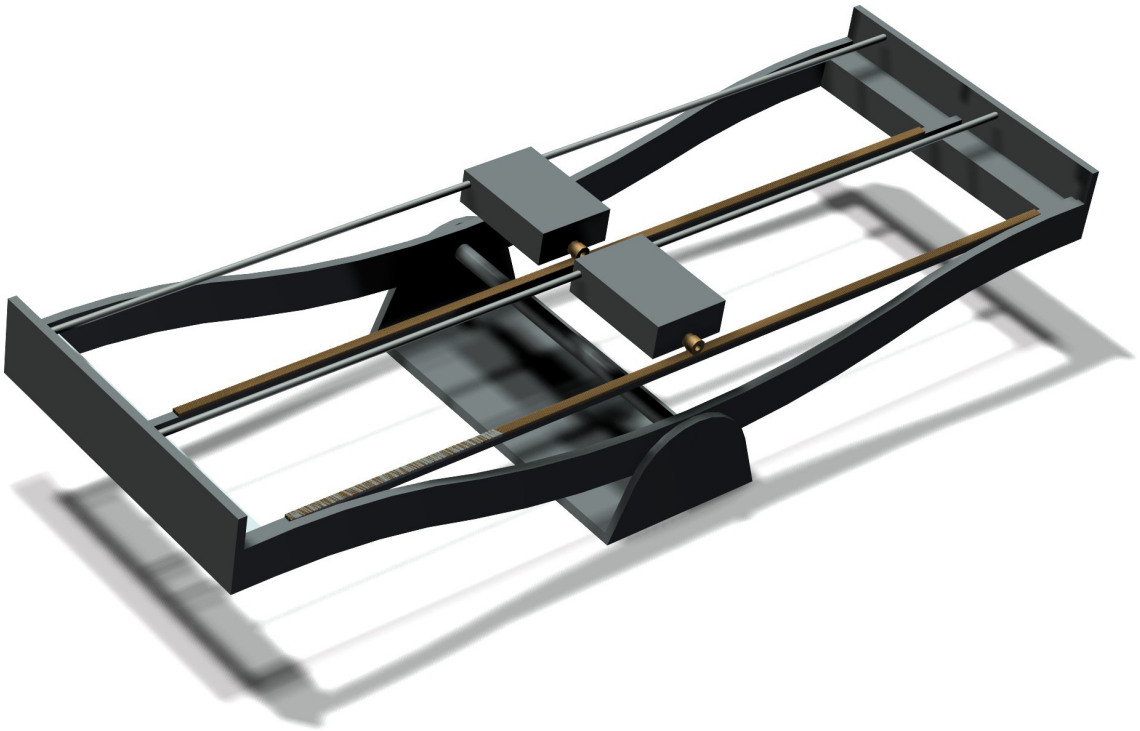


Figure 1: Wippe Hardware

Ziel dieses Laborpraktikum ist es, dass sich Student*innen mit diversen Kontrolltechniken vertraut machen, indem sie diese auf das Modell einer Wippe anwenden.

Das System *Seesaw* (Wippe) besteht aus einem Körper (der Wippe selbst), welche bezogen auf die zentrale Lagerung frei rotieren kann. Auf der Wippe sind zwei Wagen angebracht, welche mithilfe von Ansteuerung der Elektromotoren bewegt werden können. Die benutzten Sensoren sind drei Encoder. Zwei davon werden benutzt, um die Position der Wagen zu bestimmen, der dritte bestimmt den Auslenkungswinkel der Wippe. Folglich ist das System mit 2 Inputs (Strom auf die beiden Wagen, die danach ein Drehmoment erzeugen) und 3 Outputs (Position der beiden Wagen und Winkel) definiert.

Am Ende des Versuchs soll die Wippe bezüglich eines Winkels stabilisiert werden können, indem ein Wagen das System ausbalanciert, währenddessen der andere Wagen als Störfaktor agiert. Diese Zielsetzung soll erreicht werden, indem verschiedene Schritte des folgenden Skripts durchgegangen werden. Das Skript und der gesamte Versuch dienen dazu, einige Kontrollprobleme genauer anzuschauen und zu bewältigen. Das Kapitel 2 beinhaltet die genaue Herleitung des Modells und einige Aufgaben, welche die Student*innen im Voraus lösen sollen. Im Kapitel 3 wird der genaue Versuchsablauf während des Laborpraktikums erläutert.

Das Praktikum beinhaltet genauer:

- System Analyse und Verständnis;
- SISO Regler und Loop Shaping;
- LQ-Regler;
- Observer Design;
- Integral Verhalten für Steady-State Fehler;
- Kritische Auseinandersetzung von verschiedenen Reglerlösungen.

Die Assistenzperson wird die genaue Handhabung der Wippe und die Initialisierung der jeweiligen Arbeitsschritte erklären. Zudem ist das System mit folgenden Sicherheitsvorkehrungen ausgestattet:

- Sensoren sind so eingestellt, dass sie die Wagen abbremsen, um zu verhindern, dass sie mit hoher Geschwindigkeit auf die Wände treffen.
- Der rote Sicherheitsknopf neben der Wippe soll als Notaus benutzt werden. Er stoppt umgehend alle Inputs zu den Wagen. Benutzt werden soll er, wenn das System nicht wie erwünscht reagiert. Während des Versuchs sollte immer eine Hand am Sicherheitsknopf sein, falls etwas schieflaufen sollte. Zum erneuten aktivieren der Elektronik muss der Notausknopf im Gegenuhrzeigersinn herausgedreht werden.
- Es ist immer möglich, das System von der Simulink Oberfläche her zu stoppen *('Stop')*, wenn das System nicht wie erwünscht reagiert.

2 Hausaufgaben

2.1 Modell Herleitung

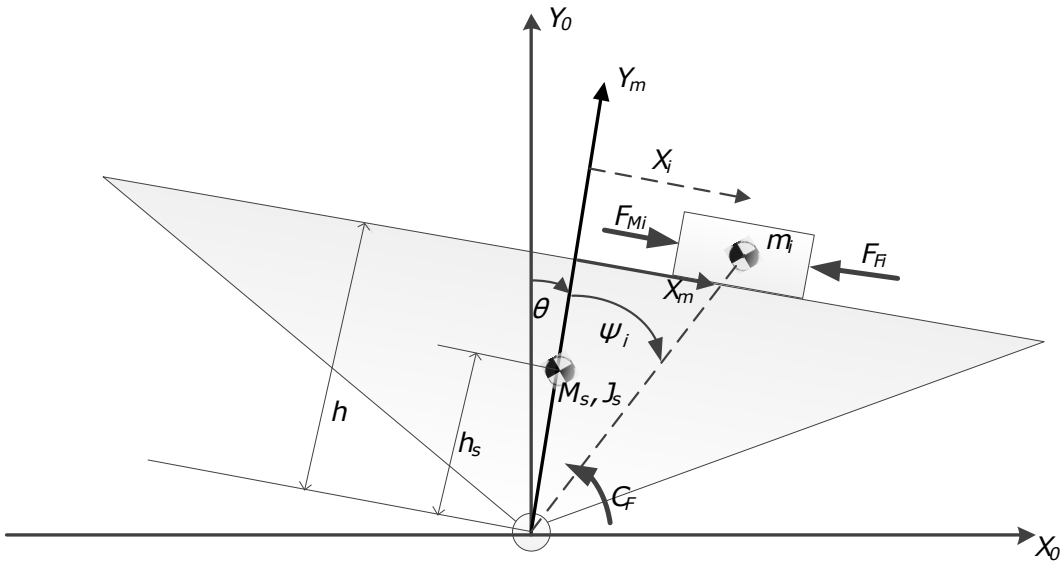


Figure 2: System Modellierung. Grössen die nicht in Tabelle 1 aufgeführt sind: θ: Auslenkwinkel der Wippe, ψ_i: Auslenkwinkel Wagen, F_{Mi}: Kraft Motoren, F_{Fi}: Dynamische Reibung Wagen, C_F: Dynamisches Reibungsdrehmoment Wippe, X_i: Distanz Wagen von der Mitteposition,

Das Bild 2 beschreibt alle relevanten Angaben und Kräfte des Systems. Um das mechanische System mathematisch zu beschreiben, wurde ein Kraft- und Momentengleichgewichtsansatz gemacht. Dieser vereinfacht das komplexe Modell, sodass dennoch das reale Verhalten approximativ beschrieben werden kann.

Um das Modell beschreiben zu können, werden folgende Vereinfachungen getroffen:

- Die Kräfte der Motoren werden als linear und abhängig von Strom und Geschwindigkeit angenommen:
  $F_{Mi} = C_{ui} I_i - C_{\dot{x}i} \dot{x}_i \qquad i \in (1, 2)$
- Die Reibungskräfte werden in einen geschwindigkeitsabhängige (dynamischen) und in eine geschwindigkeitsunabhängige Kraft geteilt. Die dynamischen Reibungskräfte werden als Dämpfung angenommen, welche proportional zu der Geschwindigkeit des jeweiligen Wagens ist:
  $F_{Fi} = \mu_i \dot{x}_i \qquad i \in (1, 2)$
- Das dynamische Reibungsdrehmoment ist proportional zur Winkelgeschwindigkeit:
  $C_F = \mu_s \dot{\theta}$

Tabelle 1 beinhaltet alle Parameter des Modells und deren Werte.

Table 1: Modellparameter

Parameter	**Beschreibung**	**Wert**	**Einheit**
m_1	Masse Wagen 1	0.696	kg
m_2	Masse Wagen 2	0.690	kg
M_s	Masse der Wippe und Schwerpunkt	4.872	kg
J_s	Trägheit der Wippe im Schwerpunkt	0.8	kg·m^2
h_s	Position des Schwerpunktes der Wippe (Höhe)	0.077	m
h	Höhe der Wippe	0.12	m
r_1	Dynamischer Reibungskoeff. für Wagen 1 ($C_{\dot{x}1} + \mu_1$)	2.32	N·s/m
r_2	Dynamischer Reibungskoeff. für Wagen 2 ($C_{\dot{x}2} + \mu_2$)	3	N·s/m
C_{u1}	Motorkonstante Wagen 1	3.944	N/A
C_{u2}	Motorkonstante Wagen 2	5.7	N/A
μ_s	Dynamischer Reibungskoeffizient der Wippe	0.4	N·m·s/rad
F_c	Coulombkraft (Reibungskraft)	??	??

Das Gesamtsystem hat drei Freiheitsgrade: Die Position des ersten Wagens (x_1), die Position des zweiten Wagens (x_2) und der Winkel der Wippe (θ). Deswegen müssen drei Bewegungsgleichungen formuliert werden. Basierend auf dem Kräfte- und Momentengleichgewicht ergeben sich folgende Gleichungen:

$$
\begin{aligned}
m_1\ddot{x_1} &= C_{u1}I_1 - C_{\dot{x}1}\dot{x_1} - \mu_1\dot{x_1} + m_1 g\sin(\theta) - sign(\dot{x_1})F_c && (1)\\
m_2\ddot{x_2} &= C_{u2}I_2 - C_{\dot{x}2}\dot{x_2} - \mu_2\dot{x_2} + m_2 g\sin(\theta) - sign(\dot{x_2})F_c && (2)\\
J_s\ddot{\theta} &= M_s g\sin(\theta)h_s - \mu_s\dot{\theta} - h(C_{u1}I_1 - C_{\dot{x}1}\dot{x_1} - \mu_1\dot{x_1} - sign(\dot{x_1})F_c) + \cdots \\
&\quad -h(C_{u2}I_2 - C_{\dot{x}2}\dot{x_2} - \mu_2\dot{x_2} - sign(\dot{x_2})F_c) + m_1 g(h\sin(\theta) + x_1\cos(\theta)) + \cdots \\
&\quad +m_2 g(h\sin(\theta) + x_2\cos(\theta)) && (3)
\end{aligned}
$$

Die ersten beiden Gleichungen ergeben sich aus dem Kräftegleichgewicht der jeweiligen Wagen und die dritte Gleichung aus dem Momentengleichgewicht um die Lagerung der Wippe.

Der geschwindigkeitsunabhängige Reibungsteil wird als Coulombsche Reibung modelliert (siehe Abbildung 3)

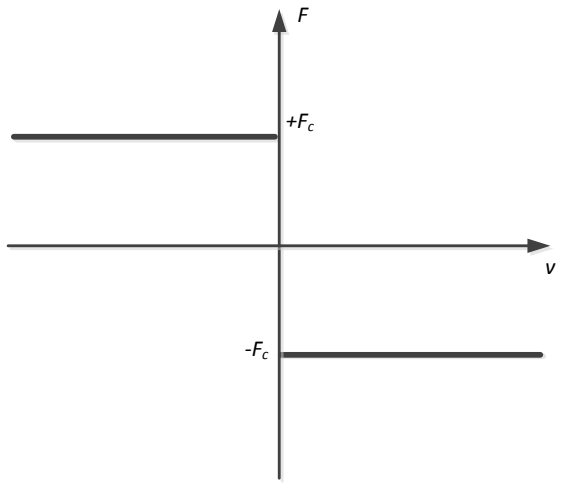


Figure 3: Coulomb'sche Reibung

Hausaufgabe 1: *Coulomb'sche Reibungskraft:* In der Auflistung der Parameter ist alles bekannt ausser die Coulomb'sche Reibungskraft F_c. Bekannt ist, dass der Strom, der benötigt wird um den Wagen 1 zu bewegen, mindestens 0.2 A sein muss. Finden Sie den Wert für F_c.

Hausaufgabe 2: *Gleichgewichtszustände:* Berechnen Sie die Gleichgewichtszustände des Systems von dem gegebenen Modell. Erklären Sie, wieso es unendliche viele Gleichgewichtspositionen gibt und wieso diese alle instabil sind.

2.2 Lineares Modell

Das System im Gleichgewichtszustand $\vec{x} = \vec{0}$ zu linearisieren führt zu der folgenden linearen Zustandsraumdarstellung.

$$\begin{cases} \dot{\vec{x}} = \tilde{A}\vec{x} + \tilde{B}\vec{u} \\ \vec{y} = \tilde{C}\vec{x} + \tilde{D}\vec{u} \end{cases}$$

mit:

$$\vec{x} = \begin{bmatrix} x_1 & x_2 & \theta & \dot{x_1} & \dot{x_2} & \dot{\theta} \end{bmatrix}', \qquad \vec{u} = \begin{bmatrix} I_1 \\ I_2 \end{bmatrix}, \qquad \vec{y} = I\vec{x}$$

$$\tilde{A} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & g & -\frac{r_1}{m_1} & 0 & 0 \\ 0 & 0 & g & 0 & -\frac{r_2}{m_2} & 0 \\ \frac{gm_1}{J_s} & \frac{gm_2}{J_s} & \frac{M_s g h_s + m_1 g h + m_2 g h}{J_s} & \frac{hr_1}{J_s} & \frac{hr_2}{J_s} & -\frac{\mu_s}{J_s} \end{bmatrix}$$

$$\tilde{B} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ \frac{C_{u1}}{m_1} & 0 \\ 0 & \frac{C_{u2}}{m_2} \\ -\frac{C_{u1}h}{J_s} & -\frac{C_{u2}h}{J_s} \end{bmatrix} \tag{4}$$

Die Eigenwerte der Matrix $\tilde{A}$ sind:

$$3.5216; \quad -1.3054 + 2.708i; \quad -1.3054 - 2.708i; \quad 0; \quad -5.2775; \quad -3.8145$$

Folglich hat das System einen instabilen Pol bei $s = 3.5216$.

Weil das System mit einem Zustandsraumregler (LQR) kontrolliert wird, ist es wichtig dieses zu normalisieren. Für das gegebene System ist es nicht möglich, das System an dem Punkt zu normalisieren, an dem es linearisiert worden ist, weil in dieser Gleichgewichtsposition alle Zustände gleich null sind. Somit wird eine Normalisierung gewählt, welche den Maximalwert jeder Variable benutzt:

x_{1max}	=	0.47	[m]			
x_{2max}	=	0.47	[m]			
θ_{max}	=	15	[deg]	=	0.2618	[rad]
$\dot{x}_{1max}$	=	5	[m/s]			
$\dot{x}_{2max}$	=	5	[m/s]			
$\dot{\theta}_{max}$	=	30	[deg/s]	=	0.5236	[rad/s]
I_{1max}	=	4.8	[A]			
I_{2max}	=	4.8	[A]			

Hausaufgabe 3: *Wagen Input:* Bis jetzt wurde das System hergeleitet, linearisiert und normalisiert. Bevor mit dem nächsten Teil weitergemacht werden kann, ist es wichtig zu verstehen, warum die beiden Ströme der Motoren als Inputs betrachtet werden. Wir nehmen also an, dass die Drehmomente (welche direkt proportional zu den Strömen sind) zu jedem Zeitpunkt definiert werden können. Wieso und unter welchen Voraussetzungen kann man diese Annahme treffen?

2.3 Positionsregler

Der erste Regler, welcher für das System bestimmt werden muss, ist ein Positionsregler für den Wagen 1. Dieser wird benutzt, wenn die Wippe in der horizontalen Position fixiert wird. Dabei wird Loop-Shaping für den SISO-Regler angewendet. Während des Praktikums wird der resultierende Regler benutzt werden, um Störungen zu provozieren und die Robustheit des LQ-Reglers (dieser wird auf den anderen Wagen angewendet, um die Wippe zu stabiliseren) zu testen.

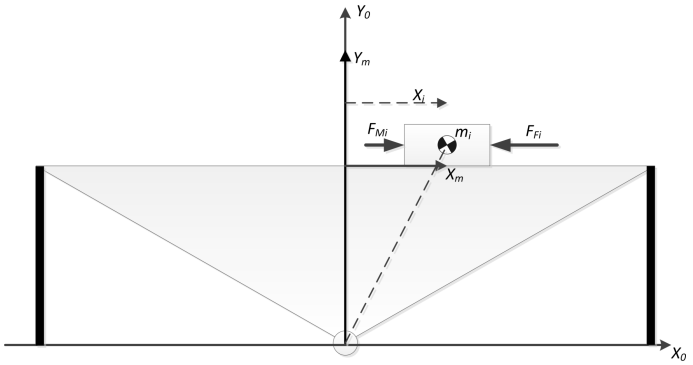


Figure 4: Horizontale Position

Wenn die Wippe in der horizontalen Position (siehe Grafik 3) fixiert wird, ist die Übertragungsfunktion zwischen Strom und Position des Wagens 1 folgende:

$$\frac{X_1(s)}{I_1(s)} = \frac{1}{s} \cdot \frac{K_i}{1 + s \cdot T_i} \qquad K_i = 1.7 \qquad T_i = 0.3 \tag{5}$$

Hier wurde das nicht-normalisierte System benutzt. Die Normalisierung wird später eingeführt, wenn mit Zustandsreglern gearbeitet wird.

Hausaufgabe 4: *PID Tuning:* Wieso ist das klassiche Ziegler-Nichols Tuning Vorgehen nicht möglich für dieses Modell? Betrachte den Nyquist-Plot der Übertragungsfunktion der Anlage.

Hausaufgabe 5: *SISO Regler:*

- Laden Sie die Simulation Files von der Website *'Prelab.zip'* herunter, entpacken Sie diese in einen Ordner und setzen Sie diesen Ordner im Matlab auf 'Current Folder';
- Initialisieren Sie `s` für die Übertragungsfunktion in Matlab, indem sie im Command Window `s=tf['s']` eingeben
- Laden Sie die Übertragungsfunktion vom Wagen 1 (5) mit dem Command Window in den Workspace und benennen Sie diese `P`;
- Öffnen Sie das `sisotool` GUI mit dem Befehl `sisotool(P)`. Per Default werden Bode Plots, Root Locus und die Step Response angezeigt. Mit der Funktion *'New Plot'* ist es möglich, einige zusätzliche Plots anzeigen zu lassen, beispielsweise den Nyquist Plot. Betrachten Sie die Matlab Dokumentation für weitere Funktionen vom `sisotool`;
- Per Rechtsklick finden Sie die Funktion *'Edit compensator...'*. Diese kann den Regler per Hinzufügen von Pol- und Nullstellen modifizieren. Versuchen Sie einen passenden Regler zu konstruieren, welche die folgenden Bedingungen erfüllt:
  - $\omega_c \sim 10$ [rad/sec] cross-over frequency;
  - $\phi_m \sim 60$ [deg] phase margin;
- Lassen Sie die Simulation laufen mit dem Matlab-File *'RunSimulation.m'*. Sie werden vermutlich sehen, dass aufgrund der Reglerstruktur ein statischer Nachlauffehler zwischen der realen Position des Wagens und der geforderten Position auftritt. Wieso gibt es ein solches Verhalten?

- Notieren Sie sich den konstruierten Regler und bringen Sie diesen zum Praktikum mit.

---

3 Lab Praxis

3.1 SISO Regler Anwendung

Zuerst wird im Lab als erster Task der als Hausaufgabe konstruierte Positionsregler an der Hardware getestet. Hierfür muss die Wippe in eine horizontale Position gebracht werden, wie es in Figure 5 gezeigt wird.

Die Assistenzperson wird Ihnen den Versuchsaufbau erklären und die Handhabung von Simulink und Hardware während des Versuchs erläutern. Wann immer Sie unsicher sind, fragen Sie einfach nach. Die korrekte Initialisierung des Versuchs ist entscheidend und unten zusätzlich beschrieben.

Hinweis: 'Cart 1' bzw. 'Wagen 1' ist der hintere Wagen, 'Cart 2' bzw. 'Wagen 2' ist derjenige, der nahe am roten Sicherheitsknopf ist.

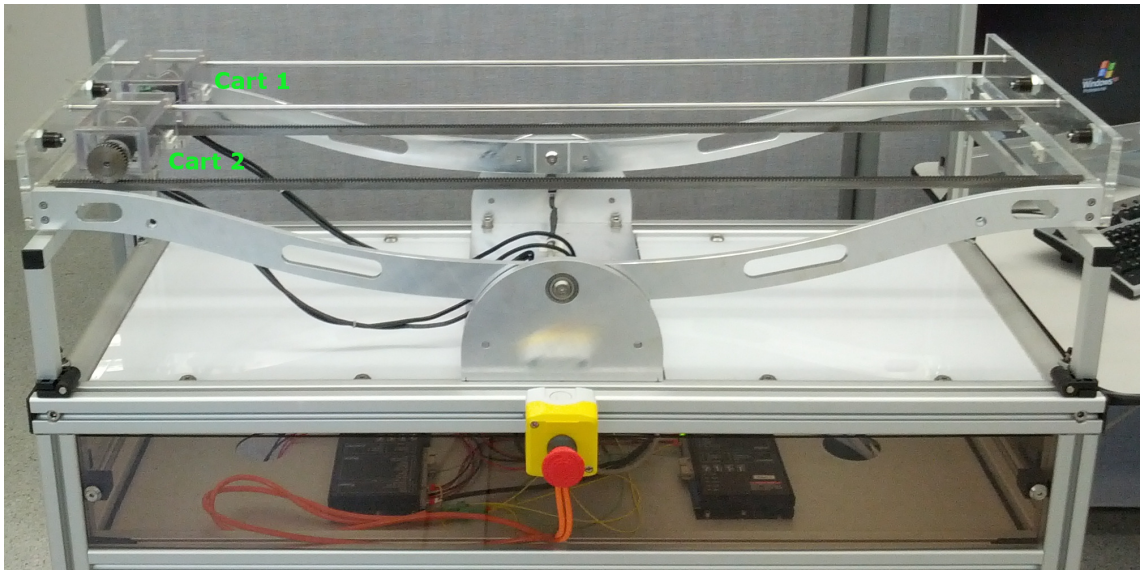
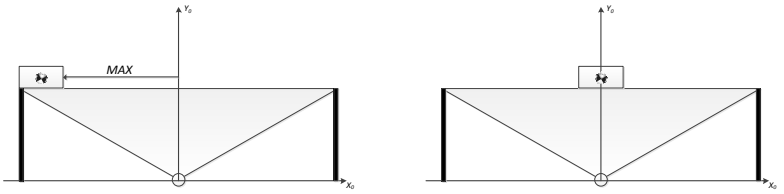


Figure 5: Wippe in der horizontalen Position

Task: Test des Positionreglers

- Öffnen Sie den Ordner *'3_1_PositionController'* im Matlab;
- Fügen Sie nun Ihre konstruierte Übertragungsfunktion aus der Hausaufgabe in den Workspace hinzu. Hierfür geben Sie "s = tf('s')" ein und danach Ihren Regler, den Sie "C" benennen;
- Öffnen Sie das Simulink-File *'PositionController'*;
- Nun können Sie die Simulation auf der Wippe laufen lassen. Fragen Sie die Assistenzperson um Hilfe, die Ihnen das Prozedere erklären wird;
- Versichern Sie sich, dass das Modell kompiliert ist. Benutzen Sie hierfür den Shortcut Ctrl + B;
- Bevor Sie nun das Simulink-File mit der Hardware verbinden, versichern Sie sich, dass die richtige Startposition wie in Figure 6(a) eingestellt ist.



(a) Initialisation (b) Initial Position

Figure 6: Horizontale Position - Startprozedere

- Während den ersten 10 Sekunden ist der Regler noch ausgeschaltet. Währenddessen müssen Sie den Wagen von Hand in die Mitte fahren. Sie sehen die Position in Figure 6(b). Führen Sie dafür genau das folgende durch:
  1. Klicken Sie auf *'RUN'* (1) im Simulink;
  2. Verschieben Sie den Wagen in die Mitte während den ersten 10 Sekunden (diese Position wird *'Mitteposition'* genannt). Die Zeit wird Ihnen unten rechts im Simulink angezeigt;
  3. Klicken Sie den roten Not-Aus Knopf, falls das System nicht reagiert wie erwartet!
- Nach 10 Sekunden wird der Wagen starten, eine bestimmte Strecke abzufahren. Wenn der Wagen wieder die Startposition erreicht hat, wird die Simulation automatisch stoppen;
- Wenn die Simulation gestoppt hat, werden Plots angezeigt. Vergleichen Sie die Strecke, die der Wagen abgefahren hat, mit der Referenzstrecke. Ist die Performance von Ihrem Regler zufriedenstellend? Testen und vergleichen Sie die verschiedenen Regler, die Sie erstellt haben;
- Simulink integriert den absoluten Fehler (actual position vs required position) auf und zeigt den finalen Wert rechts oben an. Achten Sie auf diesen Wert, und vergleichen Sie diesen jeweils während des Labs. Ein guter Startwert für diesen Regler ist 52.

Frage: Statischer Nachlauffehler

Sie sollten, wie in der Simulation zu Hause, eine Abweichung des Reglers in den Messungen feststellen können. Nehmen Sie an, dass die reale Hardware Reibung aufweist, welche als Coulomb'sche Reibung modelliert werden kann (Figure 3). Welche Probleme stellt diese Art von Reibung dar, wenn wir lineare Regelungstechnik anwenden wollen?

Frage: Lösungen zum statischen Nachlauffehler

Welches sind mögliche Lösungen, um den statischen Nachlauffehler zu minimieren oder gar zu eliminieren?
Könnten Ihre Lösungen neue Probleme oder Nebeneffekte auslösen?
Diskutieren Sie die Vor- und Nachteile mit der Assistenzperson.

Task: Test auf der Hardware

Wenn eine Lösung diskutiert worden ist, implementieren Sie diese in das File *'PositionController.mdl'* und testen Sie sie auf der Hardware. Eine gute Lösung sollte einen Fehler zwischen 30 und 35 aufweisen.

3.2 LQ-Regler

Ab jetzt können Sie die Stützen der Wippe entfernen, damit sie 'wippen' kann. Ab sofort ist es stets das Ziel, die Wippe auszubalancieren, indem man nur den Cart 2 benutzt. Cart 1 wird dazu

genutzt, um Störungen einzuführen und somit die Robustheit des Reglers testen zu können.

Da der Regler nur am Cart 2 angreifen wird, ist es möglich anzunehmen, dass Cart 1 stets in der Mitte der Wippe gehalten wird. Das heisst also, dass keine Störungen angenommen werden. Diese Annahme führt dazu, dass wir das linearisierte Modell aus (4) vereinfachen können und sich die Dimension des Zustandsraumes von 6 auf 4 verkleinert.

Task: Reduziertes Modell Gehen Sie in den Ordner *'3_2_LQR'* und öffnen Sie das Matlab-File *'simulation_ControllerDesign.m'*. Leiten Sie die A- und B-Matrix für das Reduzierte Modell aus dem vollständigen Modell in (4) her und tragen Sie diese im Matlab-file ein.
Hinweis: Die Matrizen können mit den Parametern befüllt werden (keine Zahlenwerte).

Im nächsten Task soll ein LQ-Regler designt werden. Ein LQR muss stets den kompletten Zustandsvektor, also auch die Geschwindigkeiten und Winkelgeschwindigkeiten, kennen. Da es nur möglich ist, die Positionen (x_2 and θ) mit den Encoders zu messen, muss man einen Weg finden, um die Geschwindigkeiten zu bestimmen.

In einem ersten Versuch können die Geschwindigkeiten einfach bestimmt werden, indem die Positionssignale abgeleitet werden. Der folgende Task soll zeigen, dass dieser Lösungsansatz nicht effizient ist und zusätzliche Probleme in die Reglerstruktur einbringen kann.

Task: LQR Simulation

- Bevor Sie den Regler auf der Hardware testen, soll er anhand von Simulationen getestet werden. Wichtig: Die Simulation-Files sind im Matlab auch mit *'simulation.m'* benannt, diese können jeweils **nicht** auf die Hardware zugreifen;
- Wählen Sie nun im file *'simulation_ControllerDesign.m'* die Gewichtungen für die Q- und R-Matrix (Die R-Matrix ist in diesem Fall ein Skalar) und erklären Sie der Assistenzperson weshalb Sie diese so gewählt haben. Beachten Sie die Probleme, die auftreten können bei dieser Art der Geschwindigkeitsbestimmung. Behalten Sie die Durchtrittsfrequenz bei 20 rad/s;
- Lassen Sie das File *'simulation_ControllerDesign.m'* laufen. Die Resultate der Simulation werden als Plots angezeigt. Wiederholen Sie das Prozedere bis Sie einen zufriedenstellenden Plot erhalten haben.

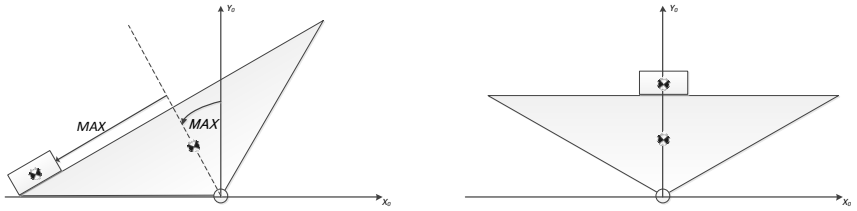
Frage: Derivative Verhalten

Die Geschwindigkeiten werden aus abgeleiteten Werten der Position bestimmt, was für den Regler in der Simulation als kein Problem erscheint. Was aber macht Probleme, wenn man Geschwindigkeiten auf diese Art bestimmt *('derivation')*? Was wird passieren, wenn man denselben Regler auf der realen Hardware testen wird?

Task: LQR Anwendung

- Vergewissern Sie sich, dass die Verstärkung des Reglers unter dem Namen `K_lqr` im Workspace vorhanden ist. Falls dies nicht der Fall sein sollte, lassen Sie das Simulation-File von zuvor nochmals im Ordner *'LQR'* laufen;
- Öffnen Sie das Simulink File *'LQR'*, welches auch auf die Hardware zugreifen kann. (Dies konnte das Simulation-File zuvor nicht!);
- Bevor Sie den Regler laufen lassen muss das Startprozedere erneut durchgegangen werden. Fragen Sie hier wieder die Assistenzperson um Hilfe;
- Startprozedere
    1. Bringen Sie das System in die Startposition, welches in Figure 7(a) aufgezeigt ist;

2. Drücken Sie *'RUN'*. Dies initialisiert alle Messungen der Encoder und verbindet den Regler mit der Hardware;
3. Für die ersten 10 Sekunden ist der Regler noch immer ausgeschaltet. Währenddessen muss manuell das System ausbalanciert werden, was in Figure 7(b) aufgezeigt ist. Halten Sie diese Balance bis der Regler startet;
4. Nach 10 Sekunden wird der LQR starten. Ab diesem Moment lassen Sie die Wippe los und beobachten den Regler;



(a) Initialisation Position (b) Center Position

Figure 7: ausbalancierte Wippe - Startprozedere

- Falls der Regler **'eskaliert'**, nutzen Sie den roten Sicherheitsknopf oder stoppen Sie den Regler auf der Simulink Oberfläche mit *'Stop'*;
- Falls der Regler wie gewünscht operiert, fügen Sie dem System manuell Störungen hinzu, indem Sie den Winkel der Wippe ändern; **Achten Sie darauf, dass niemals eine Hand in die Bahn eines Wagens kommen kann!**
- Drücken Sie *'Stop'*, wenn das Testen beendet werden soll. Nun werden die Plots der Messungen der Hardware auf dem Bildschirm angezeigt. Sie sollten bemerkt haben, dass es einen Unterschied zwischen der Performance, die die Simulation anzeigte und der realen Performance des Reglers gibt.

Frage: Encoder Initialisierung

Das Startprozedere, welches oben beschrieben wurde, ist essentiell damit der Regler auf der Hardware laufen kann. Sie wird in den folgenden Tasks genau gleich ausgeführt. Wieso ist dieses Prozedere wichtig für die Encoder? Erklären Sie, weshalb man mit solchen Sensoren keine absoluten Positionsänderungen messen kann.

3.3 Observer - LQG

Von den vorangegangenen Tests sollte klar sein, dass das numerische Bestimmen von Geschwindigkeiten via Derivation nicht immer eine optimale Lösung ist. Der folgende Task zeigt wie ein Observer verwendet werden kann um bessere Regelungsergebnisse zu erreichen.

Task: Observer

- Öffnen Sie den Ordner *'3_3_LQG_Observer'*;
- Öffnen Sie das File *'simulation_LQG_Design'*. Wählen Sie die Gewichtungen für den LQ-Regler und für den Observer mit dem LTR Prozedere. Jetzt kommen keine Begrenzungen bei der LQR Durchtrittsfrequenz vor;
- Lassen Sie das File *'simulation_LQG_Design'* laufen, um die Simulationsergebnisse zu erhalten;

- Sobald zufriedenstellende Werte im LQR und für den Observer gefunden worden sind, vergewissern Sie sich, ob `K_lqg` und die Observermatrizen `Aobs`, `Bobs`, `Cobs` und `Dobs` im Workspace vorhanden sind. Falls nicht, müsste die Simulation nochmals durchlaufen *('RUN')*;
- Öffnen Sie das Simulink-File *'LQG.slx'*;
- Startprozedere: Machen Sie nun das Startprozedere vom vorherigen Task. Falls Sie noch unsicher sind, fragen Sie erneut die Assistenzperson;
- Drücken Sie den Sicherheitsknopf, falls der Regler nicht wie gewünscht reagieren sollte. Ändern Sie anschliessend die Werte, die Sie angepasst haben, und starten Sie die Messung erneut;
- Falls der Regler wie gewünscht läuft, fügen Sie dem System Störungen hinzu, indem Sie den Winkel der Wippe manuell ändern;
- Falls der Regler die Störungen stabilisieren kann, versuchen Sie den Cart 1 von der Mitteposition weg zu bewegen. Zögern Sie hierbei nicht und wenden Sie ruhig genug Kraft auf, um den Wagen auch deutlich weg zu bewegen;
- Drücken Sie *'Stop'* im Simulink, um den Vorgang zu stoppen und die Plots zu sehen;
- Betrachten Sie die Plots: Vergleichen Sie die Plots mit denjenigen, die Sie bei vorangegangenen Tasks erhalten haben;
- Wiederholen Sie die Messung mit verschiedenen Werten, bis Sie einen zufriedenstellenden Regler erhalten haben.

Frage: Einfluss von Cart 1

Kann diese Regelungsstruktur den Winkel stabilisieren, wenn der Cart 1 nicht mehr in der Mitte ist? Falls nein, was wären Gründe hierfür?

3.4 Vollständiger Zustandsraum - LQG

Bis zu diesem Punkt wurde das vereinfachte Modell mit vier Zuständen jeweils betrachtet. Aus dem letzten Task wurde ersichtlich, dass dies nicht genügt, um den Regler robust gegen Störungen, die vom Cart 1 ausgehen, auslegen zu können.
Um die Performance des Reglers zu verbessern, wird in diesem Task der *vollständige Zustandsraum*, der in (4) gezeigt ist, betrachtet. Dies bedeutet, dass nun sechs Zustände für den LQR vorhanden sind und der Observer nun zwei Inputs mehr hat: den Strom, der in Cart 1 fliesst (I_1) und dessen Position (x_1).

Wie zuvor ist das Ziel des Tasks, dass die Wippe um 0° ausbalanciert werden kann, indem man Cart 2 bewegt. In diesem Task gibt es zusätzliche Informationen:

- Der Observer erhält zusätzliche Informationen über den Cart 1. Somit sollte der innere Zustandsraum (internal state space) besser rekonstruiert werden können.
- Die Informationen über Cart 1 soll dem LQG helfen, besser zu performen.

Task: LQG vollständiger Zustandsraum

- Öffnen Sie den Ordner *'3_4_LQG_Full'* im Matlab;
- Öffnen Sie das File *'simulation_LQG_Full_Design'*. Wählen Sie die Gewichtungen für den LQR und den Observer.
  Beachten Sie, dass jetzt beide (LQR und Oberserver) auf den vollständigen Zustandsraum zugreifen, der in (4) gezeigt ist;
- Lassen Sie das File *'simulation_LQG_Full_Design'* laufen. Anschliessend erhalten Sie die Resultate der Simulation auf den Plots;
- Sobald Sie zufriedenstellende Werte und Plots erhalten haben, vergewissern Sie sich, dass die Verstärkung des Reglers `K_lqg` und die Matrizen des Observers `Aobs`, `Bobs`, `Cobs` und `Dobs` im Workspace vorhanden sind. Falls dies nicht der Fall sein sollte, müssen Sie das File 'simulation_LQG_Full_Design' nochmals laufen lassen;
- Öffnen Sie das Simulink-File *'LQG_Full'*;
- Startprozedere: führen Sie erneut das Startprozedere, welches im Task 3.2 beschrieben wurde, durch;
- Falls der Regler nicht wie gewünscht laufen sollte, benutzen Sie den roten Sicherheitsknopf;
- Falls der Regler funktioniert, fügen Sie dem System Störungen hinzu, indem Sie manuell den Winkel der Wippe ändern;
- Falls der Regler den induzierten Winkel stabilisieren kann, bewegen Sie den Cart 1 weg von der Mitteposition;
- Drücken Sie *'Stop'* im Simulink, um den Test zu beenden und die Plots aus den Messungen zu erhalten;
- Kann der Regler nun den Winkel der Wippe stabilisieren, wenn der Cart 1 von der Mitteposition wegbewegt worden ist?

3.5 Integral Verhalten - LQGI

Von den vorangegangenen Tests konnte man sehen, dass die Störungen im Winkel und das Verschieben des Wagen vom Regler stabilisiert werden konnten, allerdings nicht in der sogenannten Zero-Position (Startposition bei 0°). Daraus lässt sich schliessen, dass der Beitrag des Integrals in der Regelstrecke und im Regler selbst entscheidend ist. Dieser muss demnach hinzugefügt werden.

Dieser letzte Task benötigt die Implementierung des Integrals auf den Winkel und testet die Performance des Reglers auch bei einem Winkel, welcher nicht 0° ist.

Task: LQGI

- Öffnen Sie den Ordner *'3_5_LQGI_Full'* im Matlab;
- Öffnen Sie das File *'simulation_LQGI_Full_Design.m'*. Wählen Sie die Gewichtungen für den LQR und den Observer;
  Beachten Sie, dass der Regler nun mit sieben Zuständen arbeitet: sechs vom Zustandsraum (4) und einer vom Integrator von θ. Zudem basiert der Observer auf dem vollständigen Zustandsraum(4);
- Lassen Sie das File *'simulation_LQGI_Design'* laufen, um die Resultate der Simulation zu erhalten;
- Sobald LQR und der Observer zufriedenstellend reagieren, vergewissern Sie sich, dass die Reglerverstärkung `K_lqg` und die Observermatrizen `Aobs`, `Bobs`, `Cobs` und `Dobs` im Workspace vorhanden sind. Falls nicht, lassen Sie das File nochmals laufen;
- Öffnen Sie das Simulink-File *'LQGI_Full'*;
- Startprozedere: Führen Sie wieder erneur das Startprozedere aus dem Abschnitt 3.2 durch;
- Drücken Sie den roten Sicherheitsknopf, falls der Regler nicht wie gewünscht reagiert;
- Falls der Regler funktioniert, führen Sie dem System manuell Störungen hinzu, indem Sie den Winkel der Wippe ändern.
  Mit dem zusätzlichen Integrator, der nun hinzugefügt wurde, sollte das System die Wippe immer um die 0°-Position stabilisieren können;
- Ändern Sie den Referenzwinkel im Simulink von 0° zu $\pm 8^o$. Hierfür müssen Sie in der Schaltbildstruktur manuell die zusätzlichen Winkel addieren (mithilfe von *'add'* und *'constant'*). Fragen Sie nötigenfalls die Assistenzperson um Hilfe.
  Führen Sie dem System erneut Störungen hinzu und testen Sie, ob der Regler diese nun immer noch ausbalancieren kann;
- Testen Sie, ob der Regler den Winkel noch immer stabilisieren kann, wenn Cart 1 von der Mitteposition wegbewegt wird.

Frage: SISO Regler (optional)

Das gegebene Regelproblem kann als Single-Input (Strom von Cart 2) Single-Output (Winkel der Wippe) System beschrieben werden. Warum wird hier aber nicht ein standardmässiger SISO-Regler konstruiert, sondern sofort ein LQR?

Vorschlag: Bestimmen Sie die Übertragungsfunktion zwischen Strom und Winkel für den Zustandsraum (4). Was macht die SISO-Betrachtung komplex?