Institut für Mechanik und Regelungstechnik - Mechatronik
Arbeitsgruppe Mess- und Regelungstechnik
Prof. Dr.-Ing. Oliver Nelles

Skript zum Fachlabor

Systemdynamik und Regelungstechnik

Stand: 23. Januar 2017

Betreuer: Dipl.-Ing. Julian Belz
Dr.-Ing. Geritt Kampmann

Universität Siegen
Fachbereich Maschinenbau
Paul-Bonatz-Straße 9-11
D-57068 Siegen

Einleitung

Dieses Fachlabor dient der Anwendung und Vertiefung grundlegender Konzepte zur Reglerauslegung. Mithilfe von MATLAB/SIMULINK werden diese Regelungskonzepte in der Simulation erprobt und anhand eines real existierenden mechatronischen Systems praktisch durchgeführt. Neben der Vorgabe gewünschter dynamischer Eigenschaften basieren die Konzepte zur Reglerauslegung auf der sogenannten Polvorgabe. Neben dem Standardregelkreis kommt die Zustandsregelung mit und ohne Beobachter zur Anwendung.

Wesentliche Inhalte des Fachlabors sind:

- Einführung in SIMULINK
- Wiederholung relevanter theoretischer Grundlagen der Regelungstechnik
- Vergleich von Steuerung und Regelung
- Standardregelkreis (PD- und PID-Regler)
- Zustandsregelung
- Luenberger-Beobachter

Wünschenswerte Voraussetzungen sind:

- Grundkenntnisse im Fach Regelungstechnik
- Grundkenntnisse in der Programmiersprache MATLAB

Inhaltsverzeichnis

1 Einführung

In diesem Fachlabor wird ein Versuchsaufbau verwendet, der gut geeignet ist, die Grundlagen der praktischen Regelungstechnik am Beispiel einer Positionsregelung zu verstehen. Im einfachsten Fall wird hierzu ein elektrisch angetriebener Wagen verwendet, der auf einer Zahnstange verfahren und dessen Position gemessen werden kann. Dieses Grundsystem ist erweiterbar, z.B. durch einen zweiten Wagen, der über eine Feder an den ersten gekoppelt wird, oder durch das Anbringen eines Pendels an den Wagen, um die Regelungsaufgabe schwerer zu gestalten.

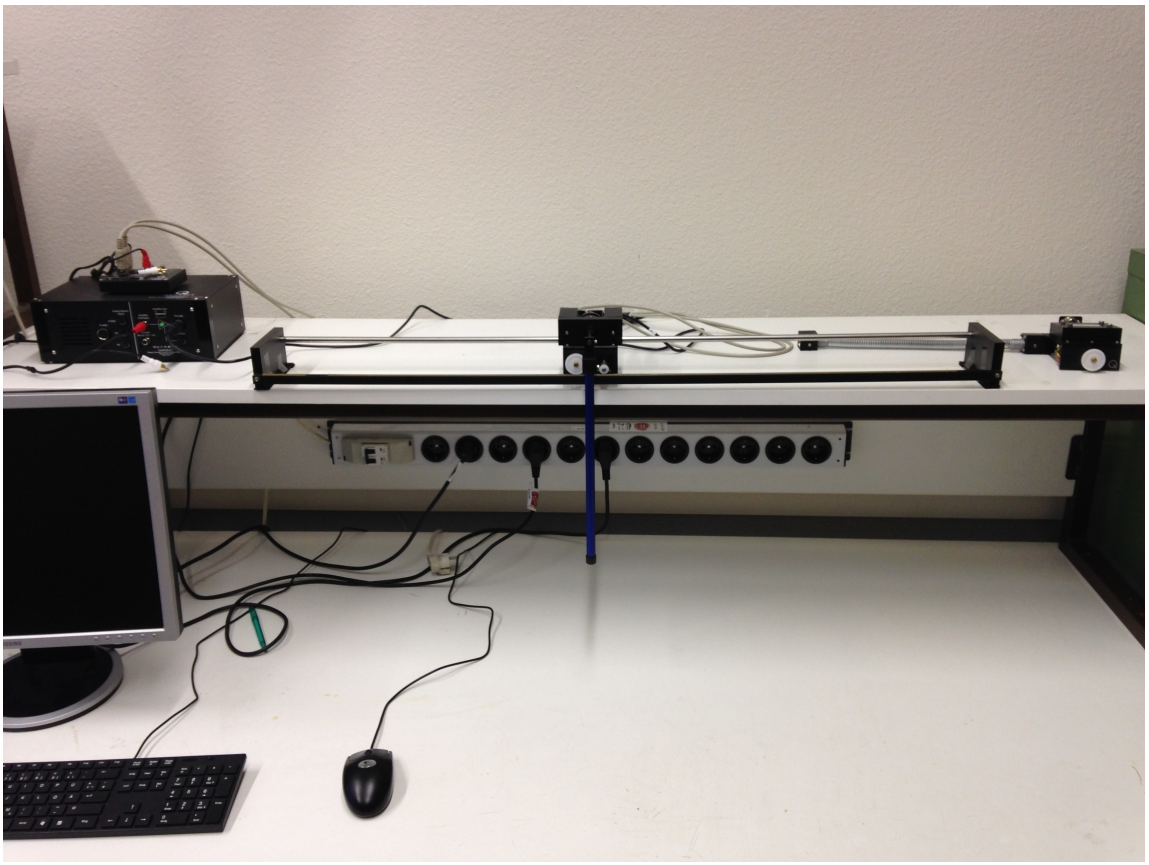


Bild 1.1: Versuchsaufbau mit Bediencomputer, Leistungsverstärker, Ansteuerungs- und Messeinheit sowie diversen Anbauteilen.

1.1 Erklärung des Versuchsstands

Der Versuchsstand besteht im Wesentlichen aus fünf Elementen. Dazu zählen der Wagen, das Schienensystem, der Leistungsverstärker, die Ansteuerungs- und Messeinheit sowie ein PC. Das Schienensystem und der Wagen sind in den Bildern 1.2 (a) bis (d) zu sehen, die Ansteuerungs- und Messeinheit sowie der Leistungsverstärker in den Bildern 1.3 (a) und (b). In Tabelle 1.1 sind die in den Bildern mit Ziffern versehenen Komponenten aufgelistet.

Nr.	Beschreibung	Nr.	Beschreibung	Nr.	Beschreibung
1	Wagen	10	Positionsgeber-anschluss	23	Endplattenschraube
2	Edelstahlschaft	11	Winkelgeber-anschluss	24	Schienenunter-brechung
3	Schiene	12	Motoransteuerungs-anschluss	25	Mini-USB An-schluss
4	Messrad	13	Gleichstrommotor	26	Messaufnehmer Eingänge
5	Antriebsrad	14	Getriebegehäuse	27	Analogsignal-ausgänge
6	Antriebswelle	15	Linearführung	28	Analogsignal-eingänge
7	Pendelachse	16	Pendelaufnahme	29	Energie- bzw. Überwachungs-LED
8	Wegaufnehmer	17	Zusatzgewicht	30	Digitale I/O An-schlüsse
9	Winkelaufnehmer	22	Endplatte	31	Erde

Tabelle 1.1: Komponentenbezeichnungen aus den Bildern 1.2 und 1.3 (a).

Die Anschlüsse des Leistungsverstärkers, in Bild 1.3 (b) zu sehen, sind in Tabelle 1.2 erläutert. Der Leistungsverstärker zusammen mit der Ansteuerungs- und Messeinheit bieten die Möglichkeit Messsignale des Wagens zu empfangen und Stellsignale an den Aktor/Motor zu senden. Als Bedienoberfläche wird in diesem Fachlabor dazu die Software MATLAB/SIMULINK genutzt. Eine kurze Einführung in SIMULINK wird im nächsten Abschnitt gegeben. Die Verkabelung des Wagens mit dem Leistungsverstärker sowie der Ansteuerungs- und Messeinheit ist im Anhang in Bild C.1 zu

Anschluss	Beschreibung	Spannungsbereich
S1,S2	Über diesen Kanal können bis zu zwei externe, analoge Sensoren angeschlossen werden. Zudem kann eine Versorgungsspannung von ±12V geliefert werden.	Eingangsbereich: -10V bis 10V
S3	Dieser Kanal kann die Daten eines externen, analogen Sensor verarbeiten. Er kann eine Versorgungsspannung von ±12V liefern.	Eingangsbereich: -10V bis 10V
S4	Dieser Kanal kann die Daten eines externen, analogen Sensor verarbeiten. Er kann eine Versorgungsspannung von ±12V liefern.	Eingangsbereich: -10V bis 10V
To ADC	Die eingelesenen Spannungen der Kanäle S1 bis S4 können über diesen Anschluss an ein externes Datenerhebungsmodul weitergegeben werden.	Ausgangsbereich: -10V bis 10V
Amplifier Command	Dieser Eingang wird mit einem analogen Spannungssignal beaufschlagt. Mit diesem lässt sich direkt der Ausgang des Verstärkers V_{out} steuern.	Eingangsbereich: -8V bis 8V
Current Sense	Gibt eine Spannung aus, die proportional zum vom Motor gezogenen Strom ist.	1 A/V
To Load	Hier wird die Last (in unserem Fachlabor der Gleichstrommotor) angeschlossen. Die Ausgangsspannung V_{out} ergibt sich aus einem Faktor C, der entweder den Wert 1 oder 3 besitzt und der an *Amplifier Command* anliegenden Spannung V_{AmpCom} zu: $V_{out} = C \cdot V_{AmpCom}$.	-24V bis 24V
LED	LED aus: System überhitzt / Keine Stromversorgung / Notaus. LED an: Verstärker einsatzbereit.	
Gain Toggle Switch	Beeinflusst den Faktor C (siehe Tabellenzeile *To Load*). In der linken Stellung ist $C = 1$, in der rechten ist $C = 3$.	

Tabelle 1.2: Erläuterungen der Leistungsverstärkeranschlüsse.

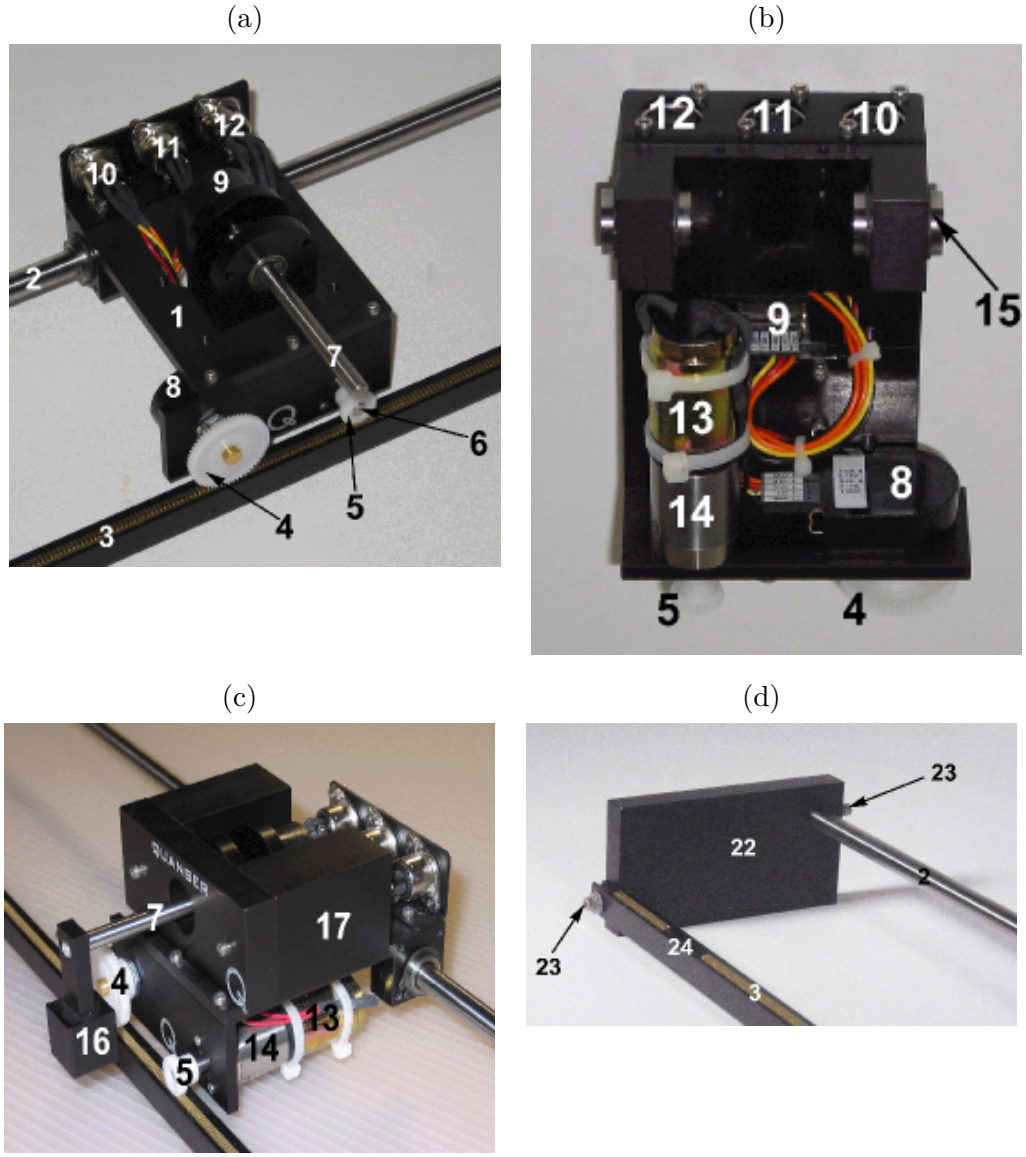
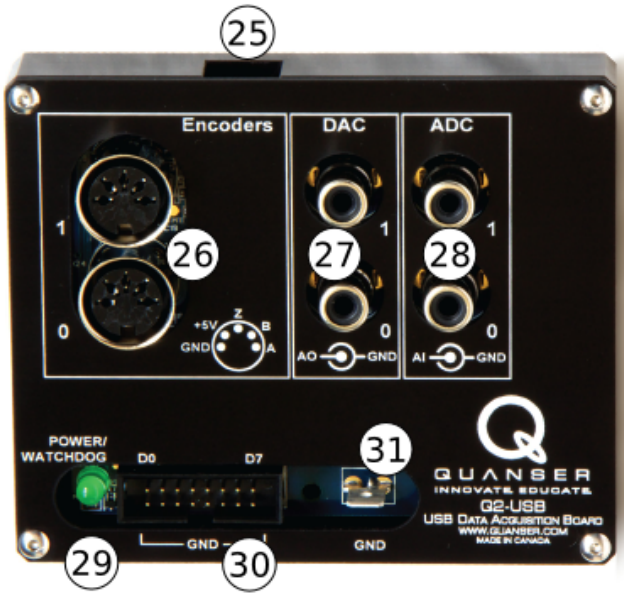


Bild 1.2: Elemente des Wagens und Schienensystems.

sehen. Nähere Informationen dazu finden sich zudem in der Gebrauchsanleitung von Quanser.

Bei sämtlichen Arbeiten mit dem Versuchsaufbau gilt es stets größte (!) Vorsicht walten zu lassen! Neben sich bewegenden Teilen, die eine nicht unerhebliche Kraft ausüben können, ist bei der Verkabelung darauf zu achten, dass die Stromversorgung erst ganz am Ende anzuschließen ist. Neben dem Achten auf die eigene körperliche Unversehrtheit, gilt es zudem mit den Komponenten des Versuchsstands pfleglich umzugehen. Insbesondere ist darauf zu achten, dass die Ansteuerungssignale für den Gleichstrommotor nicht zu hochfrequent sind. Dies kann zu einer Beschädigung des

(a)



(b)

Bild 1.3: Ansteuerungs- und Messeinheit (a) sowie Leistungsverstärker (b).

Getriebes oder der Motorbürsten führen. Das Ansteuerungssignal für den Gleichstrommotor sollte daher nie eine höhere Frequenz als **50 Hz** aufweisen. Insbesondere beim Zurückführen abgeleiteter Größen ist dafür Sorge zu tragen.

1.2 SIMULINK

Innerhalb dieses Labors wird die Software Simulink zum Einsatz kommen. Hierbei handelt es sich um eine vom Hersteller MATHWORKS entwickelte Toolbox, die in

deren Programm MATLAB-Anwendung findet. Da es sich um eine erweiternde Toolbox für MATLAB handelt, wird dieses zwingend benötigt, um SIMULINK betreiben zu können.

In diesem Kapitel soll zunächst erläutert werden, welche Einsatzmöglichkeiten diese Software besitzt. Anschließend folgt eine grundlegende Beschreibung der Bedienung. Abschließend werden einige Beispiele besprochen und Hausaufgaben gestellt, welche von den Studenten zwingend zu bearbeiten sind.

1.2.1 Einsatzmöglichkeiten für MATLAB/SIMULINK

SIMULINK ermöglicht eine hierarchische Modellierung mit Hilfe von graphischen Blöcken. Mit SIMULINK wird eine Grundausstattung von Blöcken geliefert, die eine Vielzahl von Aufgaben, sowohl bei zeitkontinuierlichen als auch zeitdiskreten Anwendungen, erfüllen können (z.B. Integration, Filtern, Darstellung von Daten etc). Zusätzliche, meist komplexere und an die jeweilige Aufgabe angepasste Blöcke können, je nach Anwendungsbereich, von MATHWORKS oder anderen Herstellern bezogen werden. Die Blöcke, welche zur graphischen Programmierung verwendet werden, bestehen aus MATLAB-Dateien, welche im mitgelieferten Edtior bearbeitet werden können. Der entscheidende Vorteil ist, dass der Benutzer sich nicht mehr mit der textbasierten Programmierung auseinander setzen muss, sondern auf diese vorgefertigten Blöcke/Algorithmen zur Programmierung zurückgreifen kann. Grundsätzlich ist es auch möglich selbst geschriebene M-Files als Block in die Simulink-Bibliothek hinzuzufügen. Die graphische Programmierung besteht nun in der Auswahl der Blöcke und dem Einzeichnen von Verbindunglinien zwischen diesen, um den benötigten Informationsfluss herzustellen. Abschließend kann das erstellte System simuliert werden, wobei verschiedene numerische Lösungsverfahren („Solver“) benutzt werden können. Mit Hilfe einer geeigneten Erweiterung für SIMULINK (Realtime Workshop), ist es möglich aus den erstellten Blockschaltbildern fertigen C-Code zu erzeugen und kompilieren, welcher dann auf externer Hardware (Microcontroller, wie z.B. die Quanser Ansteuerungs- und Messeinheit) und nicht auf dem PC ausgeführt wird. Dies ist sinnvoll, da es auf dem PC aufgrund der Vielzahl laufender Anwendungen schwierig ist, einen Echtzeitbetrieb sicherzustellen. Also dafür zu sorgen, dass möglichst exakt in festgelegten (kleinen) Abtastintervallen Daten gemessen und Stellgrößen berechnet werden. SIMULINK unterstützt alle Integer-, Gleit- und Festkommatypen (float und fixed point) in der Simulation und Codegenerierung, wobei für (skalierte) Festkommatypen eine zusätzliche Toolbox-Lizenz erforderlich ist.

1.2.2 Beschreibung des Arbeitsablaufes

Die grundsätzliche Vorgehensweise bei der Erstellung eines Modells mit SIMULINK beginnt mit dem Starten der Software. Dazu sollte eine voll funktionstüchtige Version von MATLAB inklusive SIMULINK installiert sein. SIMULINK kann, wie oben beschrieben, nur geöffnet werden, wenn MATLAB schon im Betrieb ist, vergleiche Bild 1.4. SIMULINK wird durch anklicken des dazugehörigen Symbols in der Befehlszeile (1) oder durch die Eingabe „simulink“ in der Kommandozeile von MATLAB gestartet (2). Es öffnet sich nun der „Simulink Library Browser“ (3), welcher alle im Pfad vorhandenen SIMULINK-Blöcke enthält. Ergänzt man eigene Blöcke, so müssen diese in den Pfad mit integriert werden, um sie vernünftig benutzen zu können. Nun muss ein

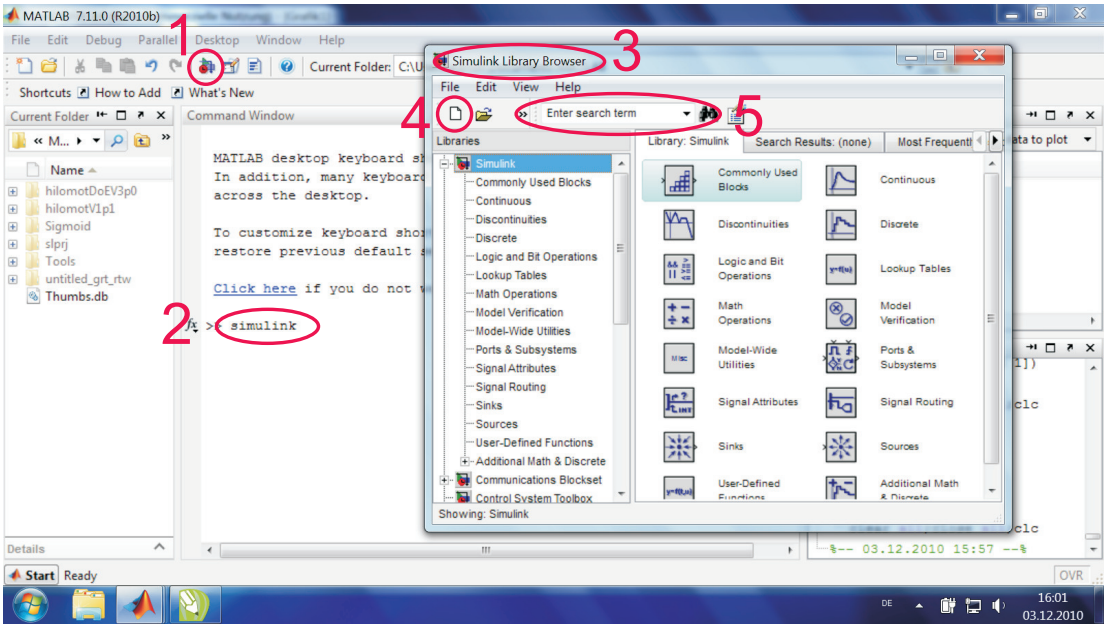


Bild 1.4: MATLAB-Oberfläche mit dazugehörigem SIMULINK Library Browser.

neues Modell erzeugt werden (4). Per „Drag and Drop“-Prinzip werden die Blöcke, welche das System beschreiben sollen, in das Modellfenster eingefügt, das heißt dass man sie mit der linken Maustaste anklickt und sie mit gedrückter Taste auf den gewünschten Bereich zieht. Lässt man die Taste los, wird der Block genau an dieser Stelle eingefügt. Wenn man nicht weiß, in welcher Bibliothek ein bestimmter Block hinterlegt ist, gibt es dazu in SIMULINK eine Suchoption (5). Sollten also in den nachfolgenden Beispielen genaue Angaben der Bibliotheken fehlen, können die Namen der Blöcke als Suchbegriffe verwendet werden. Als nächstes müssen die Blöcke miteinander verbunden werden, damit der gewünschte Signalfluss entsteht. Es ist weiterhin erforderlich, dass die einzelnen Blöcke richtig eingestellt werden. Das heißt,

dass die Parameter der Blöcke an das vorliegende Problem angepasst werden müssen. Dazu muss man die Parameterfenster der Blöcke mit dem Doppelklick öffnen. Es sind diverse Einstellmöglichkeiten gegeben. Diese Parameterfenster können sich stark unterscheiden, da sie vom Programmierer des Blockes abhängig sind. Blöcke, welche von MATHWORKS bereitgestellt werden, sind in der Regel ähnlich programmiert und dementsprechend gleichen sich die Einstellungsfenster. Um abschließend eine Simulation der Schaltung durchzuführen, muss im Modellfenster auf den Startknopf gedrückt werden. Dieser befindet sich, wie die anderen Einstellmöglichkeiten des numerischen Lösungsverfahrens in der Befehlszeile, welche in Bild 1.5 rot eingekreist sind. Möchte der Benutzer noch eine Kompilierung des SIMULINK-Blockschaltbildes

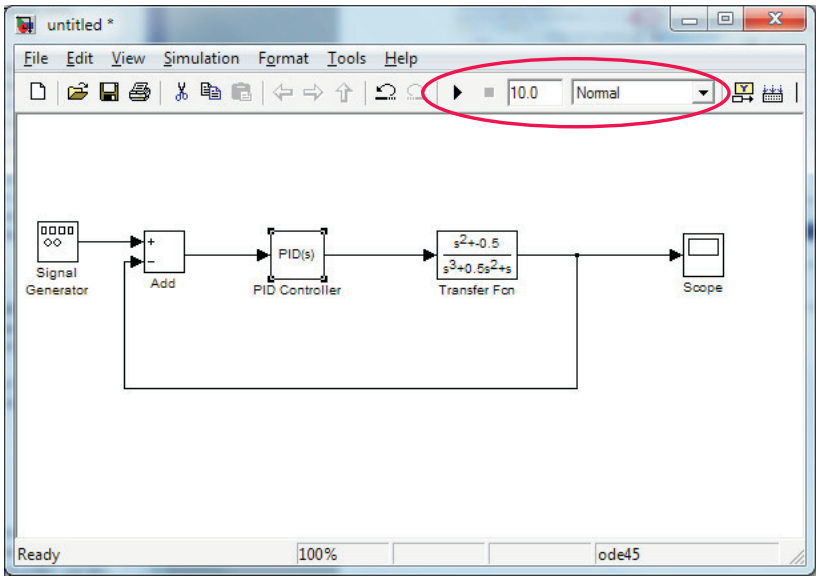


Bild 1.5: Modell eines einfachen Regelkreises.

in C/C^{++}-Code durchführen, muss der „Realtime-Workshop“ sowie eine geeignete Software zur Kompilierung, wie zum Beispiel MICROSOFT VISUAL, installiert sein. Die Beispiele und Hausaufgaben in den folgenden Unterkapiteln werden eine genauere Vorstellung des Ablaufs einer Modellerstellung liefern.

1.2.3 Anwendungsbeispiele zum Erlernen der Grundlegenden Funktionsweisen von SIMULINK

In diesem Unterabschnitt werden verschieden Beispiele abgehandelt, welche als Einführung in die Gebrauchsweise von SIMULINK dienen. So soll die Verwendung von

Blöcken und ihren Parametereinstellungen erklärt und das Zusammenspiel der einzelnen Blöcke miteinander erläutert werden.

Beispiel 1 : Erstellung einer einfachen Ausgabe

Die erste Aufgabe besteht darin, einen „Step“-Block und einen „Scope“-Block in ein neues Modell einzufügen und diese miteinander zu verbinden. Dazu wählt man die genannten Blöcke aus und zieht diese in das Modell. Nun kann man die Blöcke verbinden, indem man in Verbindungsrichtung die Blöcke nacheinander anklickt und dabei „Strg“ gedrückt hält. Eine andere Art die Blöcke zu verbinden ist, mit der linken Maustaste das kleine Dreieck des „Step“-Bolck anzuklicken und mit gedrückt gehaltener Taste eine Verbindung bis zum „Scope“-Block zu ziehen. Bei der Wahl des Beginns der Verbindung, wird aus dem Mauszeiger ein Fadenkreuz, sobald ein anklicken möglich ist. Ist man am Ende angekommen, wird beim Loslassen der Maustaste nur dann eine korrekte Verbindung eingerichtet, wenn aus dem Fadenkreuz ein doppeltes Fadenkreuz geworden ist. Um die Parameter zu verändern, muss ein Doppelklick auf den jeweiligen Block erfolgen. Ziel ist es nun einen Sprung am Zeitpunkt fünf Sekunden von minus zwei auf plus drei zu generieren und diesen mit dem Oszilloskop anzuzeigen („Scope“ ist die Kurzform von „oscilloscope“). Das fertige Modell ist in Bild 1.6 dargestellt.

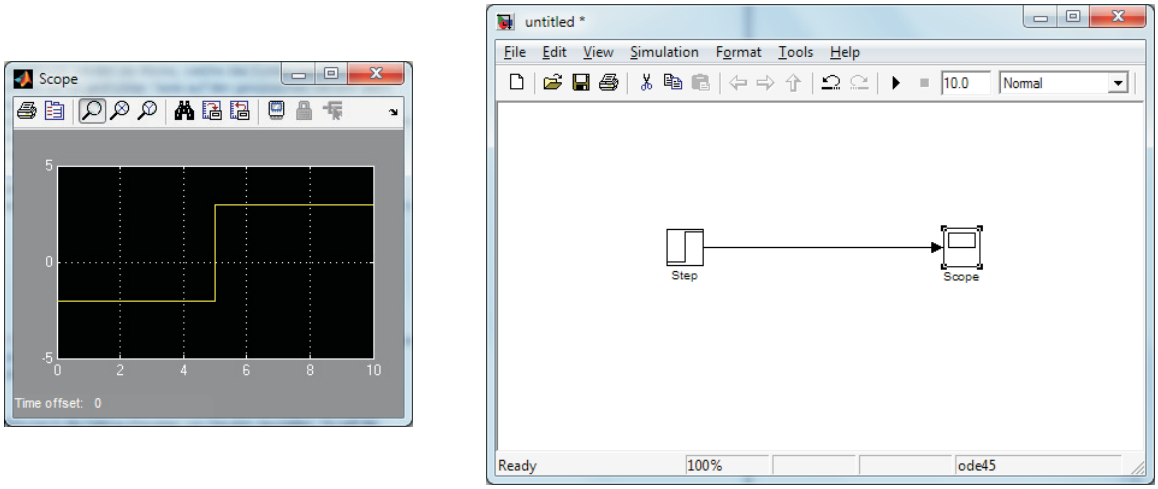


Bild 1.6: Erstes Beispielmodell zur Einführung in MATLAB/SIMULINK: Einfaches ausgeben verschiedener Signaltypen.

Beispiel 2 : Variieren der Signalquellen und deren Superposition

Wurde das erste Beispiel erfolgreich durchgeführt, sollen nun einige andere Signalquellen ausprobiert werden. Dazu ist der Sprungblock mit anderen Blöcken aus der Bibliothek *Simulink/Source* zu ersetzen. Diese Aufgabe ist selbsterklärend und bedarf keiner weiteren Darstellung.

Als sehr praktisch hat sich der „Signal Generator“-Block erwiesen. Hiermit können Signale wie Sinus, Rechteck, Sägezahn und zufälliges Rauschen („random“) erzeugt werden. Die Aufgabe besteht nun darin sich mit den Parametern dieses Blockes vertraut zu machen und verschiedene Einstellungen vorzunehmen. Eine besonders hervorzuhebende Einstellmöglichkeit ist die Rauschoption, da es mit ihrer Hilfe möglich ist ein typisches Messsignal zu erzeugen. Dieses kann dann in Simulationen eingesetzt werden, um die implementierte Schaltung hinsichtlich ihrer Robustheit zu überprüfen und eventuelle Schwachstellen zu offenbaren. Ein mögliches Messsignal soll zur Übung erzeugt werden und besteht aus der Kombination eines reinen Sinus, welcher der eigentlichen Messung - dem Nutzsignal - entspricht, und eines zufälligen Rauschens, was als Störung dient. Es wird also ein Sinusblock und der „Signal Generator“-Block benötigt, sowie eine Verbindungsstelle. Anstelle des Sinusblocks kann auch der „Signal Generator“-Block benutzt werden. Für die Verbindungsstelle bieten sich der „Add“- und der „Sum“-Block an. Beide werden mit Hinzufügen von Plus- und Minus-Zeichen in den Parameter „List of signs“ um die benötigten Eingänge erweitert. Sind die Signale miteinander kombiniert, erkennt man auf der „Scope“-Ausgabe einen verrauschten Sinus, was einem Messsignal mit Rauschen ähnelt, vergleiche Bild 1.7. Das Signal ist ein Sinus mit der Amplitude zehn und der Frequenz von einem Hertz, der mit einem Rauschen beaufschlagt wurde. In der Simulationsdauer von zehn Sekunden erkennt man deutlich einen verrauschtes Sinussignal.

Sollten die Kurven nicht den in den Bildern entsprechen, kann dies daran liegen, dass die „Solver“-Einstellungen verändert werden müssen. Hierzu muss in der Menüzeile „Simulation“ ausgewählt und der Unterpunkt „Configuration Parameters“ im Drop-Down-Menü ausgewählt werden. Dort wird dann unter „Select“ der Löser ausgewählt und zum Beispiel der Typ des Lösungsverfahrens auf „Fixed-Step“ auf die Schrittweite 0.001 gesetzt. Die Bedeutung der Schrittweite soll am nächsten Beispiel genauer gezeigt werden.

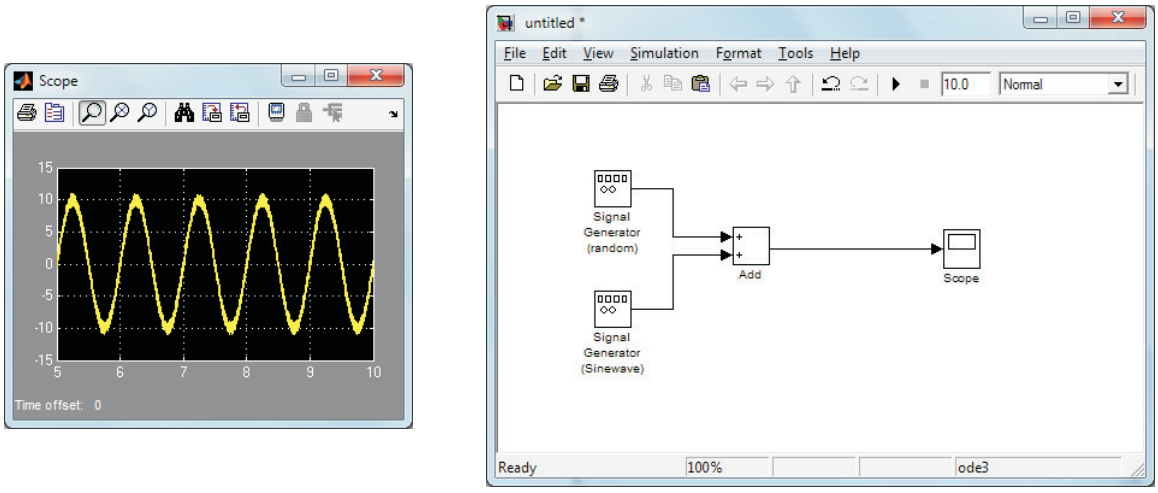


Bild 1.7: Zweites Beispielmodell zur Einführung in MATLAB/SIMULINK: Superposition eines Sinussignals mit Rauschen.

Beispiel 3 : Multiplizieren verschiedener Signale

Das dritte Beispiel unterscheidet sich vom zweiten nur dadurch, dass hier der Sinus nicht mit dem Rauschen addiert, sondern multipliziert wird. Im Prinzip muss der „Sum“- oder „Add“-Block, je nachdem was in dem vorherigen Beispiel gewählt wurde, mit dem „Dot Product“-Block ersetzt werden. Für das in Bild 1.8 gezeigte Beispiel wurde ein Sinus mit der Amplitude von zehn und der Frequenz von 0.25 Hertz mit dem Rauschen multiplikativ verknüpft.

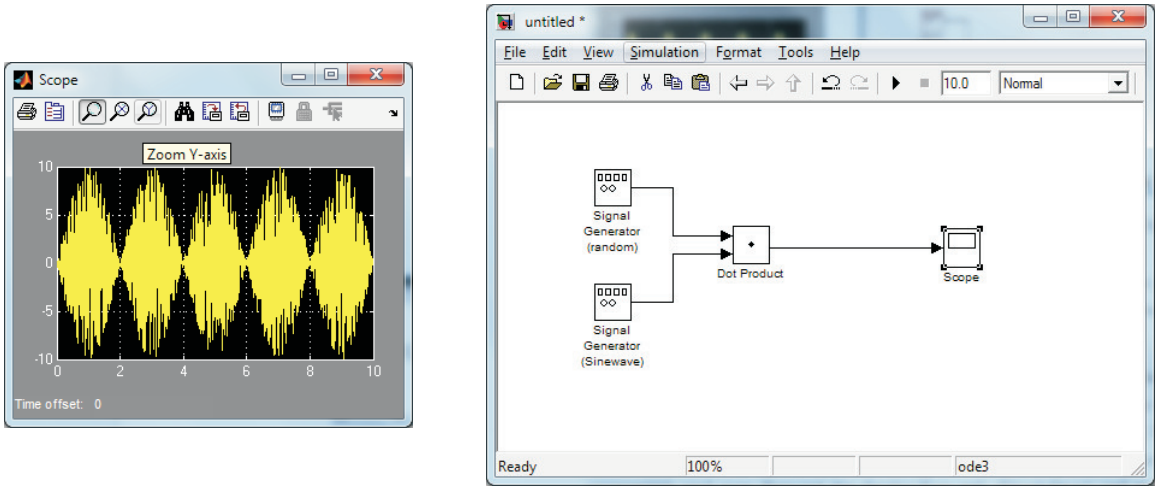


Bild 1.8: Drittes Beispielmodell zur Einführung in MATLAB/SIMULINK: Multiplikation eines Sinussignals mit Rauschen

Diese Berechnung wurde mit einer Schrittweite von 0.005 durchgeführt. Man erkennt

deutlich, dass der Sinus einen Bereich umschließt, indem mögliche Punkte liegen. Das Rauschen sorgt somit dafür, dass keine genaue Aussage getroffen werden kann, wo exakt der Signalwert zum fraglichen Zeitpunkt liegt (außer bei den Nullstellen des Sinus). Es kann lediglich der Bereich eingeschränkt werden, indem ein Punkt möglich ist.

Stellt man das numerische Lösungsverfahren auf die voreingestellte, automatische Schrittweitenanpassung, so erhält man ein anderes Bild, wie in Bild 1.9 gezeigt.

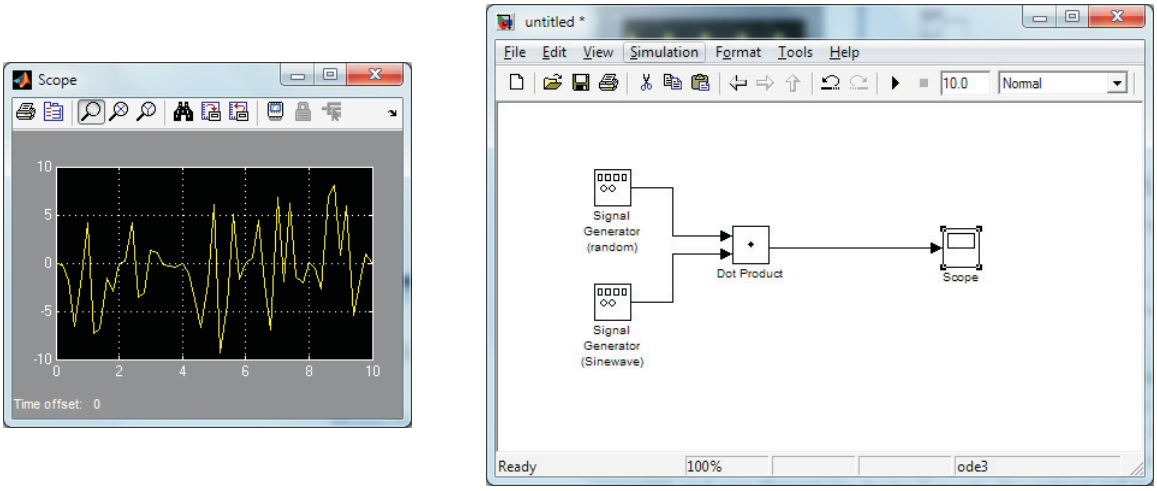


Bild 1.9: Drittes Beispielmodell mit veränderter Einstellung des numerischen Lösungsverfahrens

Hierbei ist der Zusammenhang, welcher noch aus Bild 1.8 ersichtlich war, unkenntlich. Das Signal wurde ungenau *abgetastet* und somit fehlen Informationen zur exakten Darstellung. Es ist nicht möglich den Sinus als Hüllkurve des Signal zu identifizieren.

Abtasttheorem/Aliasing-Effekt

SIMULINK ist ein computerbasiertes Simulationsprogramm und arbeitet somit mit numerischen Lösungsverfahren. Diese sind zeitdiskret. Um ein kontinuierliches Signal vernünftig abbilden zu können, muss es mit einer geeigneten Frequenz *abgetastet* werden. Die Regel, welches eine unzureichende Abtastfrequenz verhindern soll, ist das Abtasttheorem nach Shannon:

$$f_{Abtast} > 2 \cdot f_{max} \ . \tag{1.1}$$

Hierbei steht f_{max} für die höchste im analogen Signal vorkommende Frequenz, die noch korrekt durch ein Abtasten dieses Signals wiedergegeben werden kann. In der Praxis wählt man die Abtastfrequenz üblicherweise $f_{Abtast} = 5 \ldots 10 \cdot f_{max}$, um nicht nur die Frequenz, sondern auch die Signalform korrekt wiedergeben zu können. Ist das Shannonsche Abtasttheorem verletzt, werden Frequenzen $f > 1/2 \cdot f_{Abtast}$ des Signals in den Bereich niedrigerer Frequenzen gespiegelt. Auf diese Weise können hochfrequente Störsignale großen Schaden anrichten und niederfrequente Nutzsignale überlagern - diesen Effekt nennt man *Aliasing*. Ein typisches Beispiel für diesen Effekt ist die Videoaufnahme (übliche Bildwiederholraten liegen bei $f = 24$ Hz) eines beschleunigten Speichenrades. Fährt das Fahrzeug los und beschleunigt, nimmt die Winkelgeschwindigkeit des Rades zu. Ab einer gewissen Winkelgeschwindigkeit scheint diese abzunehmen, bleibt schließlich sogar scheinbar stehen und beginnt sich anschließend augenscheinlich rückwärts zu drehen, obwohl das Fahrzeug kontinuierlich beschleunigt wird. Diese optische Täuschung ist allein dem Aliasing geschuldet. Da vorab bei realen Systemen oftmals keine maximale Frequenz angegeben werden kann, liegt es in der Entscheidung des Anwenders eine sinnvolle Obergrenze anzusetzen und mit einem analogen Filter - bevor das Signal digitalisiert wird - entsprechend einzugreifen. Sind entsprechend hohe Frequenzen aus dem ursprünglichen analogen Signal gefiltert, kann dieses Signal aus dem abgetastetem Signal rekonstruiert werden, sofern das Abtasttheorem eingehalten worden ist. Ein Informationsverlust wird somit vermieden. Da analoge Filter recht teuer sind, entfällt in der Praxis die Filterung des analogen Signals oftmals. Aliasing tritt in diesem Fall mit sehr hoher Wahrscheinlichkeit auf, da die meisten realen Signale keine definierte maximale Frequenz besitzen. Enthält das Signal beispielsweise eine Rampe, sind alle Frequenzen von null bis unendlich vertreten.

Bei der Verwendung von SIMULINK ist es nun wichtig, die Schrittweite - welche der Abtastrate entspricht - des Lösungsverfahrens so zu wählen, dass der beschriebene Aliasing-Effekt nicht durch eine zu große Schrittweite des Gleichungslösers verursacht wird. In den meisten von uns betrachteten Beispielen funktionieren die Voreinstellungen des SIMULINK Gleichungslösers, sodass eine Anpassung meist nicht erforderlich ist. Dennoch sollte man diese mögliche Fehlerquelle im Hinterkopf behalten.

Beispiel 4 : Erstellung eines Blockschaltbildes

Diese ersten Beispiele sollten dazu dienen, den Umgang mit SIMULINK zu trainieren. In den folgenden Beispielen wird das Wissen um die Bedienungsweise von

SIMULINK vorausgesetzt und nicht mehr in einzelnen Unterpunkten beschrieben.

Das vierte Beispiel soll zeigen wie ein dynamisches System in SIMULINK simuliert werden kann. Dynamische Systeme werden durch Differentialgleichungen beschrieben, die in der Regelungstechnik üblicherweise, sollte das System linear sein, in Übertragungsfunktionen überführt werden. Solche Übertragungsfunktionen können in SIMULINK direkt mit Hilfe des „Transfer Fcn"-Blocks simuliert werden, wie später noch gezeigt wird. Hier soll aber zunächst vorgeführt werden, wie sich Differentialgleichungen auch direkt in SIMULINK darstellen lassen. Die allgemeine Differentialgleichung erster Ordnung lautet:

$$a_1 \cdot \dot{y} + a_0 \cdot y = b_0 \cdot u \ . \tag{1.2}$$

Für die Implementierung dieser Gleichung in SIMULINK empfiehlt es sich, diese nach der höchsten vorkommenden Ableitung wie folgt umzustellen:

$$\dot{y} = \frac{b_0}{a_1} \cdot u - \frac{a_0}{a_1} \cdot y \ . \tag{1.3}$$

Um eine bessere Übersicht zu gewährleisten, werden die skalaren Faktoren vereinfacht mit $b = \frac{b_0}{a_1}$ und $a = \frac{a_0}{a_1}$ abgekürzt:

$$\dot{y} = b \cdot u - a \cdot y \tag{1.4}$$

In SIMULINK ist es möglich ein solches System mit Hilfe eines „Integrator"-Blocks zu beschreiben. Dieser ist in der Bibliothek *Simulink/Continuous* zu finden. Ein weiterer Block, der benötigt wird, ist der „Gain"-Block (Verstärkung), der ein Signal mit einer einstellbaren Konstante multipliziert. Die Aufgabe besteht nun darin ein neues Modell zu generieren und eine Signalquelle, einen Integrations-, einen Additions-Block, zwei Gain-Blöcke und eine Scope-Anzeige hinzuzufügen. Es ist möglich, nach der Auswahl des betreffenden Blocks, diesen mit der Tastenkombination „Strg + i" umzudrehen. Das Blockschaltbild des Systems ist in Bild 1.10 gezeigt. Bei der Eingabe der Verstärkungsparameter in den „Gain"-Blöcken können Zahlenwerte oder Variablen eingetragen werden, die zuvor im MATLAB-Command Window definiert wurden. Die Parameter sind für das Beispiel wie folgt gewählt: $a = b = 1$. Der Sprung verändert seinen Wert von zwei auf sechs zum Zeitpunkt vier Sekunden.

Es ist zu erkennen, dass die Simulation einen Startwert von null besitzt. Obwohl ein Sprung von zwei auf sechs vorgegeben wird, treten tatsächlich zwei Sprünge auf. Einer beim Zeitpunkt 0 Sekunden und einer beim Zeitpunkt 4 Sekunden. Bei null

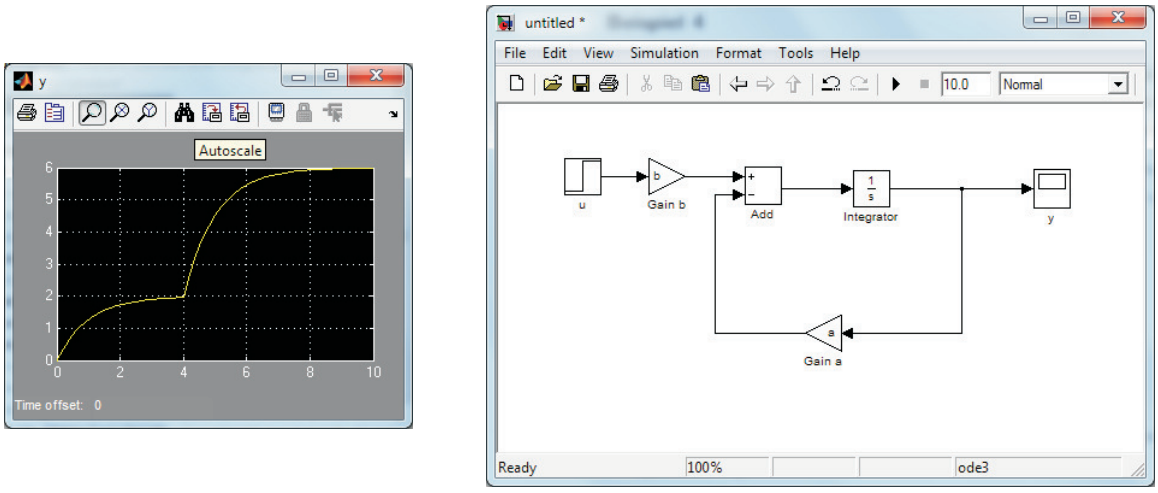


Bild 1.10: Viertes Beispielmodell zur Einführung in MATLAB/SIMULINK: Differentialgleichung erster Ordnung.

Sekunden springt der vordefinierte Startwert der Simulation auf die Ausgangsgröße unseres Sprungs, die den Wert zwei besitzt. Der zweite Sprung ist der beabsichtigte Sprung von zwei auf sechs. Der erste, ungewollte Sprung resultiert aus der Voreinstellung des Integrator-Blocks. Dieser ist mit einer Anfangsbedingung von Null versehen und somit ist der Wert für y im ersten diskreten Schritt des Lösungsverfahrens gleich null. Es gibt nun zwei Möglichkeiten den Startwert zu verändern. Zum Ersten kann man den Wert direkt auf zwei setzen. Dies ist dann allerdings nur verwendbar für Sprünge mit diesem Ausgangswert. Zum Anderen kann der Integrator-Block mit einem weiteren Eingang versehen werden, der ihm die Anfangsbedingung vorgibt. Dazu muss die Option „Initial condition source“ von „internal“ auf „external“ gesetzt und der neue Eingang mit dem „Step“-Block verbunden werden, siehe Bild 1.11.

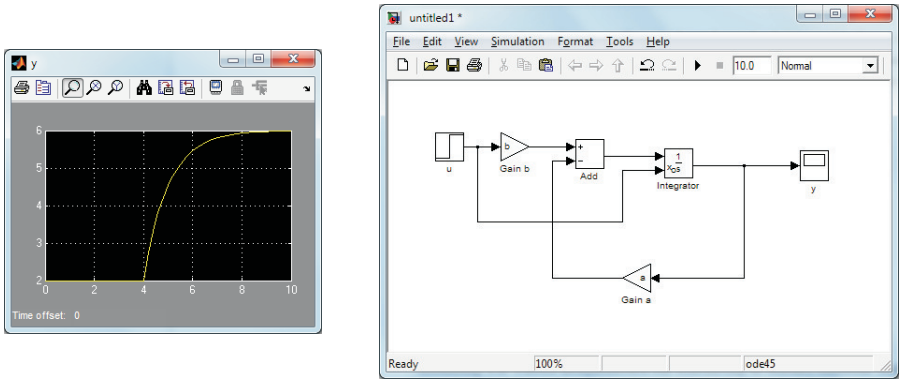


Bild 1.11: Viertes Beispielmodell zur Einführung in MATLAB/SIMULINK: Veränderung der Anfangsbedingung.

Um SIMULINK-Schaltbilder übersichtlicher zu gestalten, gibt es die Möglichkeit sogenannte *Subsysteme* zu erstellen. Dazu zieht man einen Kasten mit der linken Maustaste um alle Blöcke und Verbindungen, die innerhalb des Subsystems sein sollen, und klickt mit der rechten Maustaste auf einen der ausgewählten Blöcke. Wählt man nun im Drop-Down-Menü „Create Subsystem“ aus, wird ein Untersystem erstellt, vergleiche Bild 1.12.

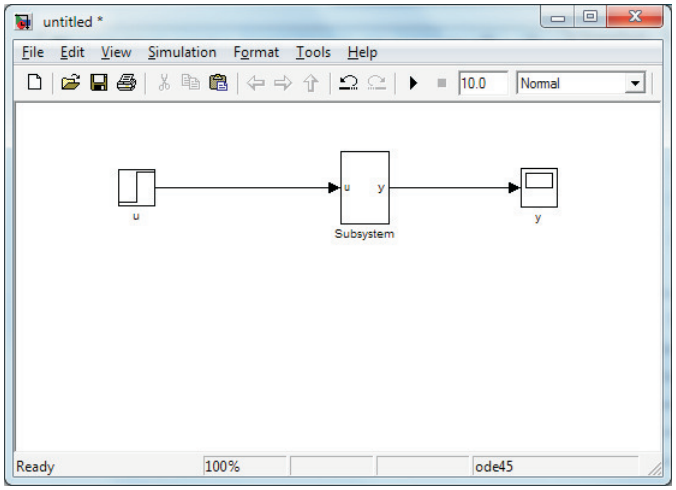


Bild 1.12: Viertes Beispielmodell zur Einführung in MATLAB/SIMULINK: Subsystem der Blockstruktur

Abschließend wird die Übertragungsfunktion für die Differentialgleichung 1.4 hergeleitet und in SIMULINK implementiert, damit ein Vergleich mit dem dazugehörigen Blockschaltbild aus Bild 1.10 durchgeführt werden kann. Um die Übertragungsfunktion aufstellen zu können, muss die Gleichung in den Frequenzbereich transformiert werden. Hierbei findet die Laplace-Transformation Verwendung. Aus Gleichung 1.4 folgt dann:

$$\dot{y} = b \cdot u - a \cdot y \tag{1.5a}$$

$$s \cdot Y = b \cdot U - a \cdot Y \tag{1.5b}$$

$$\frac{Y}{U} = \frac{b}{s+a} = G(s) \tag{1.5c}$$

Die Übertragungsfunktion $G(s)$ kann in SIMULINK direkt mit dem Block „Transfer Fcn“ implementiert werden. Hierzu muss der besagte Block aus der Bibliothek *Simulink/Continuous* dem Modell hinzugefügt werden. Weiterhin wird natürlich eine Signalquelle und eine Signalausgabe benötigt. Bei den Parametern des Blocks „Transfer Fcn“ sind die Koeffizienten des Zählers sowie des Nenners hinzuzufügen.

In unserem Fall $[b]$ für den Zähler und $[1\,a]$ für den Nenner. Wird die selbe Konfiguration wie zuvor verwendet, ergibt sich der in Bild 1.13 dargestellte Zusammenhang. Eine Einflussnahme auf die Anfangsbedingung des Systems gibt es bei der Verwendung des „Transfer Fcn“-Blocks nicht.

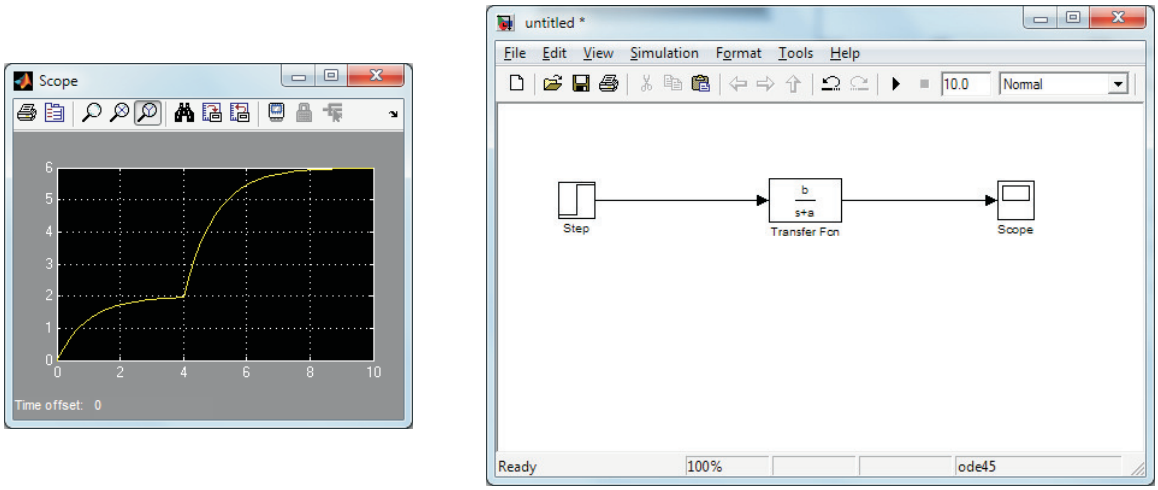


Bild 1.13: Viertes Beispielmodell zur Einführung in MATLAB/SIMULINK: Transferfunktion in SIMULINK.

Nun hat man das System kompakt dargestellt und kann zum Beispiel einen Regelkreis damit entwerfen, wie es in der Mess- und Regelungstechnikvorlesung mit Hilfe der Übertragungsfunktionen durchgeführt wurde.

Erstellung und Verwendung von konfigurierbaren Subsystemblöcken

Weiterhin ist es möglich einen variablen Block zu generieren, so dass man mit wenigen Mausklicks, zwischen den einzelnen Subsystemen umschalten kann. Das vierte Beispiel mit seinen drei möglichen Subsystemen soll nun mit variablen Subsystemblöcken erstellt werden. Zu Beginn muss eine neue Bibliotheksdatei angelegt werden. Dazu wählt man in der Befehlszeile des „Simulink Library Browsers“ *File→New→Library*. Es öffnet sich ein neues Fenster mit der Bezeichnung „Library“. In dieses Fenster müssen nun alle Subsysteme, zwischen denen umgeschaltet werden soll, eingefügt werden. Das erste Subsystem enthält die DGL 1. Ordnung, wobei der Integrator-Block einen externen Anfangszustand besitzt. Das zweite Subsystem enthält die DGL 1. Ordnung, bei der der Integrator-Block einen internen Anfangszustand besitzt. Und das letzte Subsystem besteht nur aus der Übertragungsfunktion (trotzdem muss

ein Subsystem erstellt werden). Der Block „Configurable Subsystem“ ist ebenfalls hinzuzufügen. Die Subsysteme innerhalb der Bibliothek können jederzeit mit einem Doppelklick geöffnet und Veränderung vorgenommen werden. Eine fertige Bibliothek ist in Bild 1.14 gezeigt. Es ist darauf zu achten, dass die einzelnen Subsysteme glei-

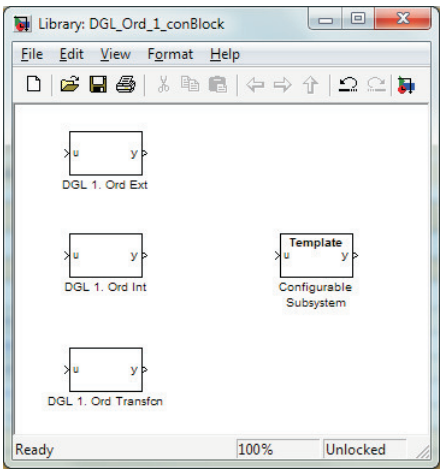


Bild 1.14: Einfache Bibliothek in SIMULINK

che Aus- und Eingänge besitzen (alle Eingänge sind in diesem Beispiel mit u, alle Ausgänge mit y bezeichnet). Grundsätzlich ist es auch möglich unterschiedliche zu verwenden, aber um eine einfache und verständliche Implementierung zu gewährleisten, sollte man in diesem Fall darauf verzichten. Dem „Template“-Block sind die Subsysteme, welche durch ihn repräsentiert werden sollen, durch setzen von Haken in den Parametereinstellungen (Doppelklick!) zuzuordnen. Ist der Bibliothekseintrag korrekt hinzugefügt, kann das variable Subsystem - beispielsweise via Kopieren/Einfügen - im vierten Beispiel verwendet werden, siehe Bild 1.15. Mit einem Rechtsklick kann im „Drop-Down-Menü“ der Eintrag „Block Choice“ zwischen den einzelnen Subsystemen umgeschaltet werden. Dies ermöglicht eine schnelles Umschalten und eine gute Vergleichbarkeit der einzelnen Systeme.

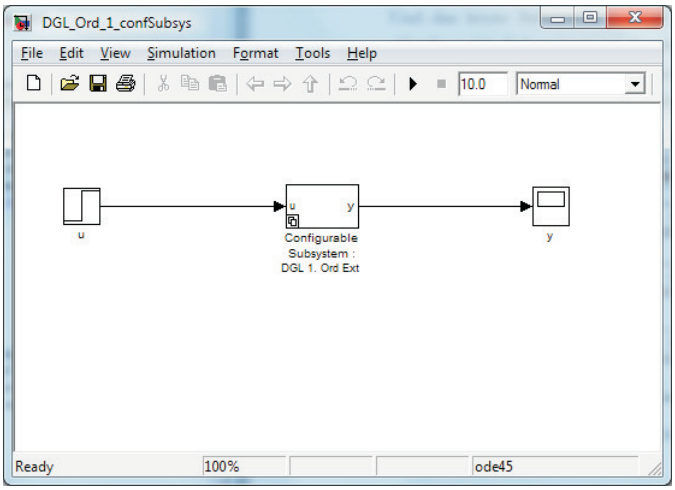


Bild 1.15: Konfigurierbares Subsystem

2 IP02-Wagen

Das erste zu untersuchende System des Fachlabors ist der IP02-Wagen. Für die in diesem Kapitel durchzuführenden Untersuchungen ist der Wagen mit einem Zusatzgewicht bestückt - es befindet sich jedoch kein Pendel an der Pendelachse. Zunächst gilt es in Abschnitt 2.1 die Bewegungsgleichung des IP02 herzuleiten. Abschnitt 2.2 erläutert die praktische Ansteuerung des Aktuators und das Auslesen der Wagen-Sensoren. In Abschnitt 2.3 sollen zwei Realisierungen einer Positionssteuerung ausgearbeitet werden und anschließend sowohl simulativ als auch praktisch miteinander verglichen werden. Die Realisierung einer Positionsregelung ist Thema von Abschnitt 2.4.

2.1 Bewegungsgleichungen der Regelstrecke

Um Experimente am Versuchsaufbau durchführen zu können, ist es zunächst nötig einige Gleichungen herzuleiten, die das System beschreiben. In diesem Kapitel soll der Wagen beschrieben werden, der mittels Elektromotor auf der Zahnstange bewegt wird und dessen Herstellerbezeichnung *IP02* lautet. Dazu soll zunächst eine beschreibende Gleichung für das System hergeleitet werden. Als Hilfestellung zur Herleitung der Bewegungsgleichung(en) sollen die Bilder 2.1 und 2.2 dienen.

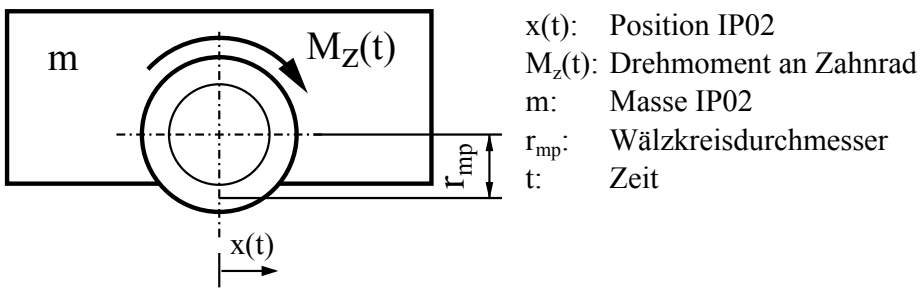


Bild 2.1: Frontansicht IP02 mit Erläuterung wichtiger Größen.

Hinweise zur Aufstellung der Bewegungsgleichung

Der Ausgangspunkt der Bewegungsgleichung ist das 2. Newtonsche Axiom, was für unser Beispiel folgende Basis liefert:

$$m\ddot{x}(t) + F_\theta(t) = F_z(t) - F_\omega(t) \ , \tag{2.1}$$

mit der (Stell-) Kraft $F_z(t)$, die am Zahnrad aufgrund des Motordrehmomentes $M_M(t)$ angreift. Der letzte Term von Gleichung 2.1 beschreibt eine Motordrehzahl abhängige Dämpfung des Systems (dies ist keine Coloumbsche Reibung - diese soll bei der Herleitung der Bewegungsgleichung vernachlässigt werden). Zur Berechnung dieser Kraft dient eine Konstante, die vom Hersteller des Versuchsstandes angegeben ist: $F_\omega(t) = B_{eq}\dot{x}(t)$. $F_\theta(t)$ ist eine Folge des Massenträgheitsmoments des Elektromotors.

Das Motordrehmoment $M_M(t)$ lässt sich mit der Vorschrift

$$M_M(t) = \eta_M \, K_t \, I_M(t) \tag{2.2}$$

berechnen. Dabei entspricht η_M dem Wirkungsgrad des Motors und K_t einer Motorkonstanten, die angibt wie viel Drehmoment pro Stromstärke generiert wird. Da die eigentliche Stellgröße die Motorspannung $U_M(t)$ ist, soll mit Hilfe von Bild 2.2 die Stromstärke $I_M(t)$ in Abhängigkeit der Spannung ausgedrückt werden. Die Induktivität L ist dabei so gering, dass man diese vernachlässigen kann ($L \approx 0$). Die Motorwiderstandsspannung $U_{wM}(t)$ berechnet sich wie folgt:

$$U_{wM}(t) = K_m \omega_M(t) \ . \tag{2.3}$$

Dabei bezeichnet K_m eine weitere Konstante des Elektromotors und $\omega_M(t)$ die Winkelgeschwindigkeit der Motorwelle.

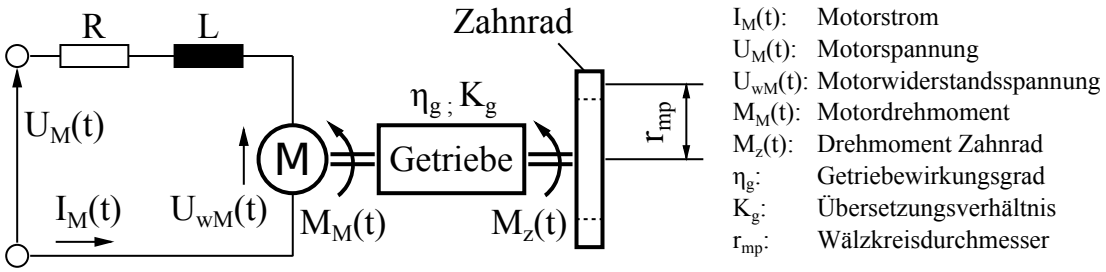


Bild 2.2: Zusammenhang zwischen Motoreingangsspannung und Moment an der Abtriebswelle.

Aufgabe 1

a) Leiten Sie die Bewegungsgleichungen im Zeitbereich mit den zuvor gegebenen Hilfestellungen her.

b) Überführen Sie die Bewegungsgleichung in den Laplace Bereich und berechnen Sie anschließend die Übertragungsfunktion

$$G(s) = \frac{x(s)}{U_M(s)} \ . \tag{2.4}$$

Welche Eigenschaft lässt sich anhand der Übertragungsfunktion sofort erkennen?

c) Implementieren Sie die aufgestellte Bewegungsgleichung zunächst lediglich im Bildbereich in MATLAB. Machen Sie sich dazu mit dem MATLAB-Befehl *tf* vertraut. Schauen Sie sich sowohl die Impuls- als auch die Sprungantwort an, die dazugehörigen Befehle sind in MATLAB *impulse* und *step*. Können Sie anhand der Antworten Aussagen über die Stabilität der Regelstrecke treffen?

d) Bestimmen Sie mithilfe des Befehls *pole* die Lage der Pole der Regelstrecke und überprüfen Sie, ob ihre Annahmen bezüglich der Systemstabilität bestätigt werden. Lassen Sie sich anschließend mit dem Befehl *rlocus* die Wurzelortskurve und mit dem Befehl *bode* das entsprechende Bode-Diagramm anzeigen.

e) Erstellen sie ein SIMULINK-Modell, welches das Übertragungsverhalten wiedergibt. Als Eingang dient die Motorspannung $U_M(t)$ [V] und der Ausgang der Regelstrecke sei die Position $x(t)$ [m].

f) Vergleichen Sie zur Kontrolle die Sprung- und Impulsantworten der SIMULINK-Modelle mit der in MATLAB definierten Übertragungsfunktion.

g) Implementieren Sie das Modell der Regelstrecke ohne vereinfachten Motorschaltkreis, d.h. falls die Vereinfachung $L_M \approx 0$ nicht getroffen wird. Vergleichen Sie die beiden Modelle, d.h. mit und ohne Vereinfachung.

2.2 Praktische Ansteuerung und Messwertaufnahme

Zur Interaktion mit dem IP02-Wagen besitzt das Vehikel einen Aktor, den Elektromotor, und zwei Sensoren in Form von Drehwinkelgebern. Im folgenden soll gezeigt werden, wie der Elektromotor innerhalb eines SIMULINK-Modells angesteuert und die Drehwinkelgeber ausgelesen werden können. Sobald man den Elektromotor ansteuern oder einen Drehwinkelgeber auslesen möchte, muss innerhalb des SIMULINK-Modells zwingend der *HIL-Initialize*-Block in das entsprechende Modell eingefügt werden, siehe Bild 2.3. Der *HIL-Initialize*-Block ist im SIMULINK Libra-

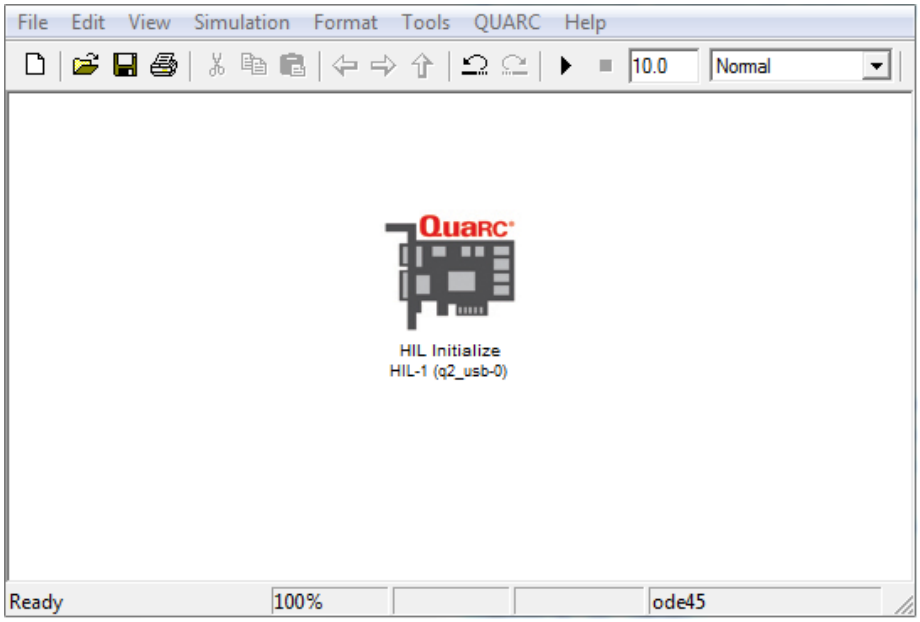


Bild 2.3: Initialisierungsblock zum Ansteuern und zur Messwerterfassung des IP02-Wagens.

ry Browser unter *QUARC Targets - Data Acquisition - Generic - Configuration* zu finden. In diesem Block muss die zum Einsatz kommende Ansteuerungs- und Messeinheit korrekt ausgewählt werden. Nach einem Doppelklick auf den Block öffnet sich das entsprechende Auswahlfenster. Unter *Board type* ist *q2_usb* auszuwählen, da dies die in diesem Labor verwendete Hardware ist. Nach der Auswahl der verwendeten Ansteuerungs- und Messeinheit steht diese in Klammern in der Beschriftung des SIMULINK-Blocks, wie in Bild 2.3 zu sehen ist. Ist die Initialisierung mithilfe diesen SIMULINK-Blocks erfolgt, kann mit den im folgenden näher beschriebenen SIMULINK-Blöcken innerhalb des Simulationsmodells auf die Messwerte zugegriffen werden bzw. der Elektromotor angesteuert werden.

Ansteuerung des Elektromotors

Als erstes geht es um die Ansteuerung des Elektromotors. Der SIMULINK-Block zum Ansteuern des Elektromotors nennt sich *HIL Write Analog* und findet sich unter *QUARC Targets - Data Acquisition - Generic - Immediate I/O*. Es sollte sichergestellt sein, dass in den Block-Einstellungen der richtige Kanal eingestellt ist. Dies ist Kanal 0, entsprechend der Verkabelung, die im Anhang in Bild C.1 dargestellt ist. Der soeben eingefügte Block ist eine Senke, die den Eingangswert als Spannung in Volt interpretiert und diese Spannung an den Elektromotor anlegt. Um den Elektromotor nicht zu beschädigen, sollten lediglich Spannungen zwischen -5 V und +5 V ausgegeben werden. Dazu wird in das SIMULINK-Modell ein *Saturation*-Block eingefügt, der genau diese beiden Grenzwerte berücksichtigt (Doppelklick auf *Saturation*-Block und *Upper limit* auf 5 sowie *Lower limit* auf -5 setzen). Um das SIMULINK-Modell lauffähig zu machen und die Ansteuerung des IP02-Wagens zu testen soll nun ein sinusförmiger Spannungsverlauf auf den Elektromotor gegeben werden. Die Frequenz des Spannungsverlaufs soll 1 rad/sec betragen und eine Amplitude von ±1.5 V aufweisen. Das finale SIMULINK-Modell sollte dem in Bild 2.4 dargestellten Modell ähneln. Um den IP02-Wagen mithilfe des erstellten SIMULINK-

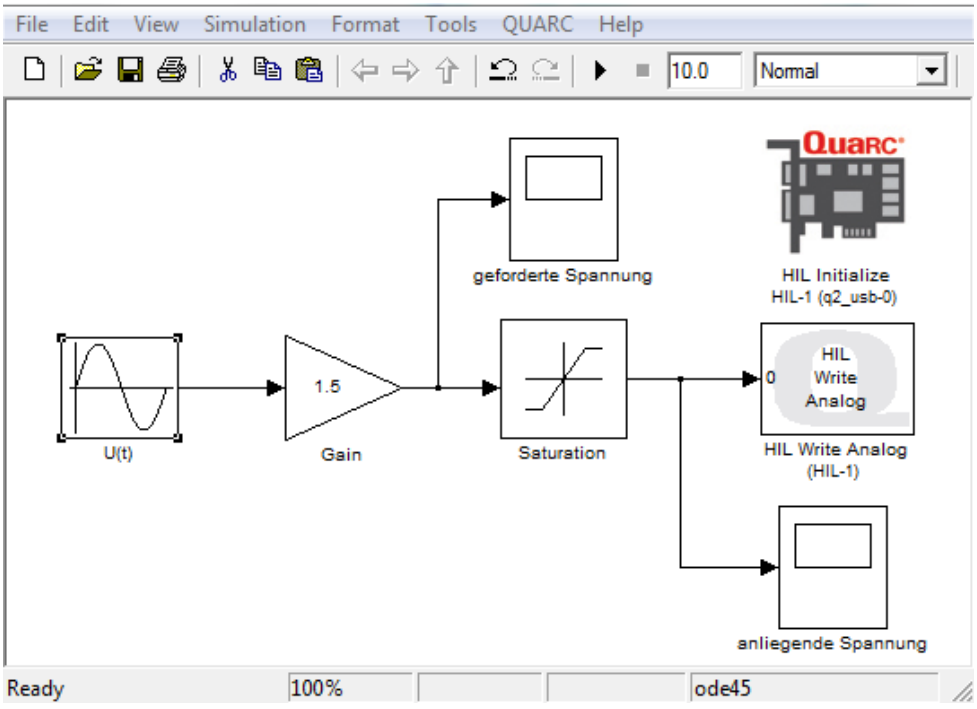


Bild 2.4: SIMULINK-Modell zur Ansteuerung des IP02 Aktuators.

Modells anzusteuern, müssen noch Standardeinstellungen gesetzt werden. Dazu muss in der Menü-Leiste auf den Eintrag *QUARC* und anschließend auf *Set default options*

geklickt werden. Der nächste Schritt besteht darin, das erstellte SIMULINK-Modell in eine für die Hardware verständliche Form der Maschinensprache zu übersetzen und lokal auf der Ansteuerungs- und Messeinheit zu speichern. Dazu wählt man in der Menü-Leiste unter *QUARC* den Eintrag *Build* aus. Hat die Übersetzung erfolgreich geklappt, sollte im MATLAB-Command Window als letzter Eintrag zu lesen sein, dass das Programm sich nun auf der Ansteuerungs- und Messeinheit befindet (*Model <model name> has been downloaded to target: ...*). **Bevor das Programm nun ausgeführt wird, muss sichergestellt sein, dass der IP02-Wagen in etwa in er Mitte der Zahnstange steht.** Dies verhindert eine Kollision des Wagens mit den seitlichen Streckenbegrenzungen. Bevor das Programm gestartet wird, muss noch der Leistungsverstärker eingeschaltet werden, damit der Motor auch eine Antriebsspannung erhält. Der Start des Programms erfolgt über die Menü-Leiste *QUARC - Start*. Während der Wagen verfährt soll nun der Faktor im *Gain*-Block verkleinert werden - zunächst auf 1 und schließlich auf 0.5. Die Geschwindigkeit des Wagens sollte abnehmen ebenso wie die zurückgelegte Strecke. Prinzipiell ist auch eine Vergrößerung des Faktors möglich, dies sollte innerhalb diesen Versuchs jedoch nicht durchgeführt werden um das Kollisionsrisiko sowie die Materialbeanspruchung gering zu halten.

Auslesen der Drehwinkelgeber

Zum Auslesen der Drehwinkelgeber dient wieder ein Modell, in dem sich der *HIL-Initialize*-Block befindet, wie bereits in Bild 2.3 dargestellt. Der IP02-Wagen besitzt zwei Drehwinkelgeber, einen zur Positionsmessung und einen zur Messung der Winkelstellung der Pendelaufnahme. Auch wenn wir (vorerst) nur die Position des Wagens benötigen, gilt es in einem SIMULINK-Modell nun beide Sensoren auszulesen. Dazu benötigen wir zwei *Scope*-Blöcke sowie zwei *Gain*-Blöcke - alle beide Blocktypen sollten bereits aus der Einführung in SIMULINK bekannt sein. Des weiteren wird ein *HIL Read Encoder Timebase*-Block benötigt, der unter *QUARC Targets - Data Acquisition - Generic - Timebases* im SIMULINK Library Browser zu finden ist. Per Doppeklick auf diesen zuletzt genannten Block, erhalten wir Einblick auf dessen Einstellparameter. Unter *Channels* fügen wir alle verfügbaren Kanäle zu den ausgewählten hinzu, indem wir auf den Button mit den drei Punkten klicken, die noch nicht ausgewählten Kanäle anwählen und per Klick auf » zu den ausgewählten Kanälen hinzufügen. Die restlichen Einstellungen können auf den Standardeinstellungen belassen werden. Nun sollte der *HIL Read Encoder Timebase*-Block zwei Ausgänge besitzen, einer mit 0 und der andere mit 1 bezeichnet. Der mit 0 bezeich-

nete Ausgang liefert das Signal des Drehwinkelgebers zur Positionsmessung, der mit 1 bezeichnete entpsrechend das Signal des Drehwinkelgebers zur Winkelmessung der Pendelachse. Die sich nun im SIMULINK-Modell befindlichen Blöcke müssen nun wie in Bild 2.5 gezeigt miteinander verbunden werden. Der Faktor für den *Gain*-

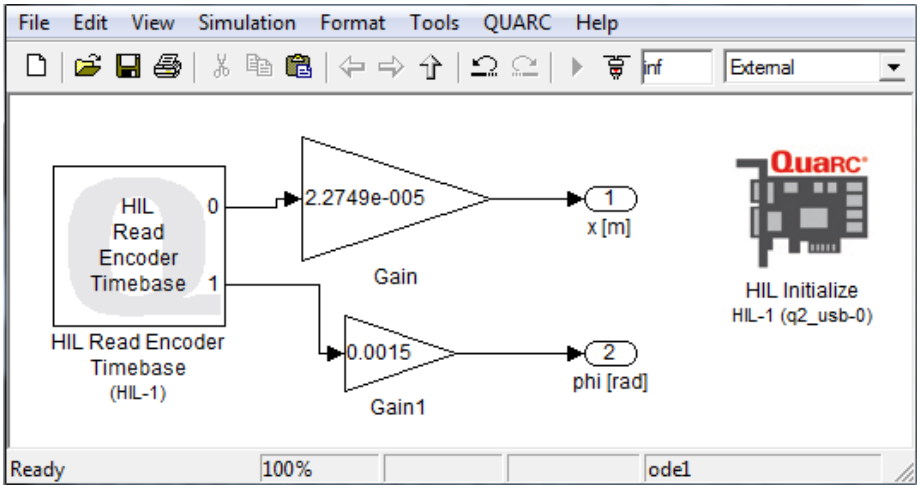


Bild 2.5: SIMULINK-Modell zum Auslesen der beiden Drehwinkelgeber.

Block, der mit Ausgang 0 verbunden wird, beträgt $2.2749 \cdot 10^{-5}$ und sorgt dafür, dass das gemessene Wegsignal in Metern ausgegeben wird. Der andere *Gain*-Block erhält den Wert 0.0015, sodass der ausgegebene Winkel die Einheit rad besitzt. Um das SIMULINK-Modell und damit die Messwertaufnahme zu starten, müssen zunächst abermals die Standardeinstellungen für den Quanser Versuchsaufbau gesetzt werden. Dazu muss in der Menü-Leiste unter *QUARC* auf *Set default options* geklickt werden. Anschließend ist das Modell in Maschinensprache zu übersetzen (Menü-Leiste - *QUARC* - *Build*) und kann letztendlich gestartet werden (Menü-Leiste - *QUARC* - *Start*). Öffnet man nun die beiden *Scope*-Blöcke und bewegt den Wagen und/oder die Pendelachse, sollte dies zu sehen sein. Gegebenenfalls muss per *Autoscale*-Button der angezeigte Bereich der *Scopes* angepasst werden.

Aufgabe 2

a) Erstellen Sie ein SIMULINK-Blockschaltbild, dass zum Ansteuern des realen IP02-Wagens dienen kann, sowie die aktuelle Position und den Winkel der Pendelachse ausgibt. Erzeugen Sie aus diesem Schaltbild ein Subsystem, bei dem Sie sicherstellen müssen, dass der *HIL Initialize*-Block mit darin enthalten ist.

b) Erstellen Sie ein SIMULINK-Modell, in dem das reale System mit dem physikalischen Modell verglichen wird. Als Anregungssignal soll für beide Systeme ein Sinus der Frequenz 1 rad/sec und einer Amplitude von 1 dienen.

2.3 IP02 Positionssteuerung

Bevor es nun im Speziellen um die Steuerung des IP02 Wagens geht, soll zunächst eine kurze Wiederholung zum Thema Steuerung erfolgen. Steuern ist die vorwärtsgerichtete Beeinflussung eines Systems, also ohne Kontrolle der Ausgangsgröße $y(t)$ durch Messung (Rückkopplung). Wie in Bild 2.6 zu sehen, berechnet die Steuerung aus der Führungs- oder Wunschgröße $w(t)$ die Stellgröße $u(t)$. Daraus sollte sich die gewünschte Zielgröße $y(t)$ ergeben, die durch äußere Störeinflüsse $d(t)$ beaufschlagt sein kann. Optimalerweise folgt die Zielgröße exakt der Führungsgröße, d.h.: $y(t) = w(t)$.

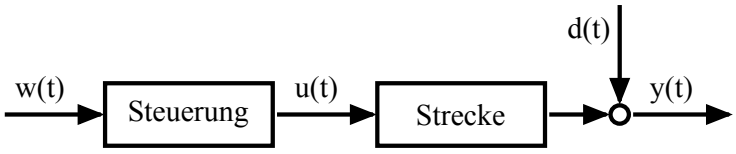


Bild 2.6: Steuerung zur Beeinflussung des Systemverhaltens.

Daraus ergibt sich die optimale Steuerung als Inverse der Übertragungsfunktion der Strecke. Lediglich die Störung $d(t)$ ist durch dieses Vorgehen noch nicht beseitigt, siehe Bild 2.7. Falls die Störung messbar ist, lässt sich die Steuerung weiter verbessern,

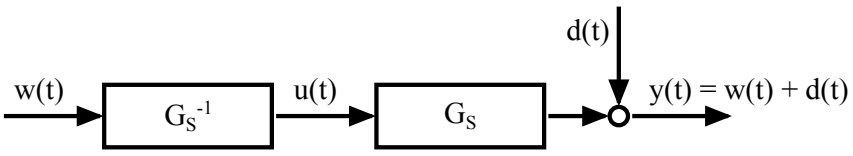


Bild 2.7: Inverse der Strecke als Steuerung des Prozesses.

indem man sie von der Führungsgröße subtrahiert, wie in Bild 2.8 gezeigt. Neben der Tatsache, dass sich viele Störungen in der Realität nicht messen lassen, bestehen weitere Probleme, die zu Einschränkungen für die optimale Steuerung führen. Durch die Invertierung der Streckenübertragungsfunktion kann die resultierende Übertragungsfunktion der Steuerung $G_{St} = G_S^{-1}$ instabil oder akausal werden. Was praktisch jedoch so gut wie immer auftritt, ist ein rein differenzierendes Verhalten, welches in

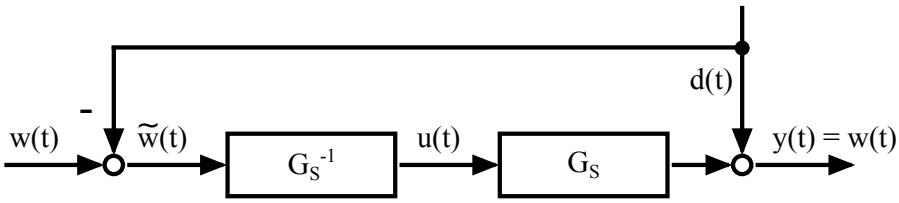


Bild 2.8: Kompensation der Störung durch Subtraktion von der Führungsgröße.

der Realität nicht realisierbar ist. Dies liegt daran, dass fast alle realen Strecken nicht sprungfähig sind, d.h. der Zählergrad der Streckenübertragungsfunktion ist kleiner als der Nennergrad. Die Realisierbarkeit der Inversen kann durch Multiplikation von Verzögerungsgliedern mit kleiner Zeitkonstante erreicht werden:

$$G_{St}^{(opt,real)}(s) = G_S^{-1}(s) \cdot \frac{1}{(1+Ts)^{nn-nz}} \ . \tag{2.5}$$

Dabei bezeichnet T die zu wählende Zeitkonstante, nn den Nennergrad der Streckenübertragungsfunktion und nz den entsprechenden Zählergrad.
Ein genereller Nachteil von Steuerungen ist ihre Sensibilität bezüglich Modellungenauigkeiten. Es ist unmittelbar klar, dass die Steuerung nur so gut sein kann, wie die mathematische Beschreibung des zu steuernden Prozesses. Auftretende Ungenauigkeiten werden (aufgrund fehlender Rückkopplung der Zielgröße) nicht erfasst und können somit auch nicht berücksichtigt werden. Wie sich beim Vergleich des IP02 Wagens mit seinem Modell herausgestellt haben sollte, bestehen zwischen Modell und Realität erhebliche Abweichungen. Grund dafür sind nichtlineare Reibungseffekte, die eine tote Zone der Stellgröße verursachen. Bevor nun eine Steuerung für unseren IP02 Wagen entworfen werden kann, muss Aufgabe 3 bearbeitet sein.

Aufgabe 3

Treffen Sie Maßnahmen, die zu einer besseren Übereinstimmung zwischen Modell und Realität führen. Verändern Sie dazu die Ansteuerung des realen Motors. Eine näherungsweise Kompensation des nichtlinearen Verhaltens sollte durch Aufschalten von konstanten Stellgrößen möglich sein, um lineares Modellverhalten zu erzeugen. Eine Anpassung der Ansteuerung erlaubt es bei einem linearen mathematischen Modell der Strecke zu bleiben, was den Entwurf der Steuerung vereinfacht. **Hinweis**: Erstellen Sie ein Modell, bei dem Sie die Spannung der Motoransteuerung über einen sogenannten *Slider Gain*-Block (zu finden unter *Simulink Library Browser - Simulink - Math Operations*) zunächst zwischen 0 V

und 1 V einstellen können. Beginnen Sie bei einer Ansteuerungsspannung von 0 V und erhöhen Sie diese stufenweise.

Sofern die Übereinstimmung zwischen Realität und Modell nun verbessert ist, kann mit dem Entwurf einer Steuerung begonnen werden. Dazu sollen zwei Ansätze zum Einsatz kommen. Zum einen die bereits erwähnte Realisierung der inversen Streckenübertragungsfunktion und zum anderen die Berechnung einer Solltrajektorie. Bei der Solltrajektorie berechnet man zu welchen Zeitpunkten welche Positionen vom IP02-Wagen erreicht sein sollten und kann aus diesen Informationen auf den Stellgrößenverlauf zurück schließen. Ein einfaches Beispiel soll dies veranschaulichen. Wir gehen von einem System aus, welches der Differentialgleichung

$$\dot{y}(t) + 2y(t) = u(t) \tag{2.6}$$

folgt. Der Verlauf der Zielgröße $y(t)$, soll einer Sinusschwingung mit vorgegebener Frequenz ω entsprechen:

$$y(t) = \sin \omega t \ . \tag{2.7}$$

Ist dieser Sollverlauf bekannt, kann der zeitliche Verlauf der Stellgröße $u(t)$ berechnet werden, um die Sollvorgaben zu erfüllen. Dazu ermittelt man alle benötigten zeitlichen Ableitung der Sollvorgabe, in diesem Beispiel von Gl. 2.7, und setzt diese in die Differentialgleichung, hier Gl. 2.6, ein. Im gewählten Beispiel setzt man sowohl Gl. 2.7 als auch deren zeitliche Ableitung

$$\dot{y}(t) = \omega \cos \omega t \ . \tag{2.8}$$

in Gl. 2.6 ein und erhält für den Stellgrößenverlauf:

$$u(t) = \omega \cos \omega t + 2 \sin \omega t \ . \tag{2.9}$$

Für die folgenden Aufgaben ist die Möglichkeit innerhalb von SIMULINK sogenannte Masken bzw. Maskierungen zu erstellen von Vorteil, daher soll die Vorgehensweise dazu näher erläutert werden. Zur Maskierung benötigt man zunächst ein SIMULINK-Subsystem, wie es beispielhaft in Bild 2.9 zu sehen ist. Im SIMULINK-Subsystem ist eine Variable $t0$ enthalten, die sowohl für den *Constant*-Block als *Constant value*

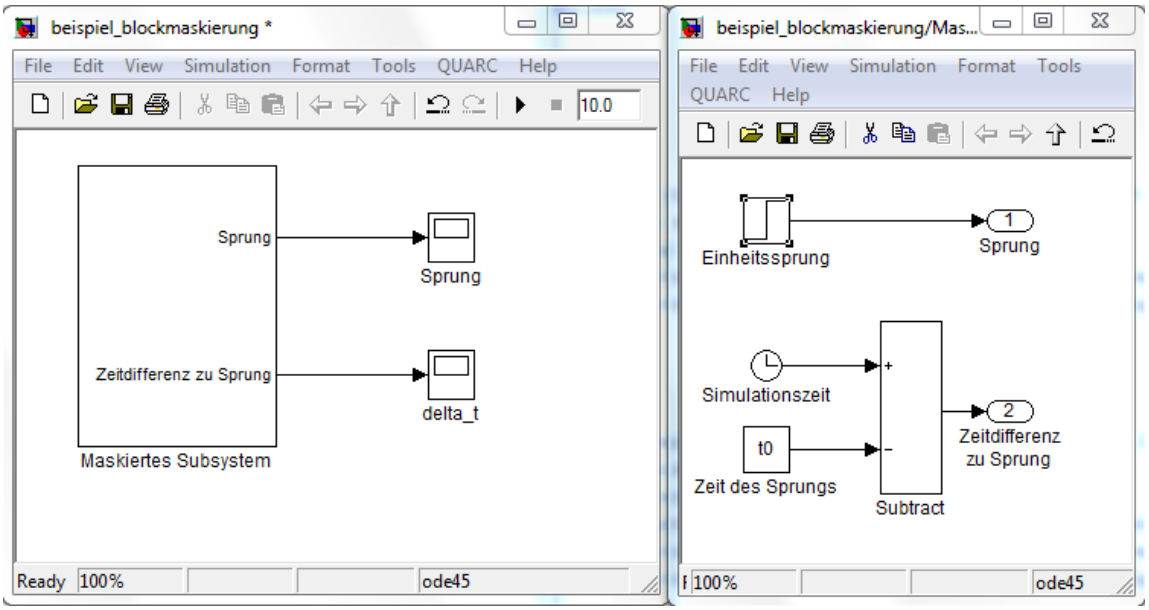


Bild 2.9: Beispiel eines SIMULINK-Subsystems.

dient, als auch innerhalb des *Step*-Blocks als *Step time* gesetzt ist, siehe Bild 2.10 (a). Die Maskierung erfolgt über einen Rechtsklick auf das Subsystem und einen Linksklick im sich öffnenden Auswahlmenü auf *Mask Subsystem...*. Im sich öffnenden Fenster fügt man nun unter der Registerkarte *Parameters* die Variable $t0$, wie in Bild 2.10 (b) gezeigt, hinzu. Klickt man nun auf *OK* und anschließend doppelt auf das nun maskierte Subsystem, öffnet sich eine Maske, in der ein Wert für die Variable $t0$ eingetragen werden kann. Man ändert somit durch eine Eingabe einen Wert im Subsystem an mehreren Stellen.

Aufgabe 4

a) Entwerfen Sie eine Positionssteuerung für den IP02-Wagen auf der Grundlage der inversen Streckenübertragungsfunktion. Überprüfen Sie simulativ die entstehenden Stellgrößen für unterschiedliche Zeitkonstanten T und wählen Sie T sinnvoll für die Ansteuerung des realen IP02-Wagens. Maskieren Sie dazu einen *Transfer Fcn*-Block, in dem Sie die Übertragungsfunktion der Steuerung in Abhängigkeit von T definieren und T innerhalb der Parameter-Maske einstellen können.

b) Ermitteln Sie eine analytische Funktion, die sowohl die Solltrajektorie als auch die Stellgrößen zum Einhalten der Solltrajektorie ausgibt. Zur Berech-

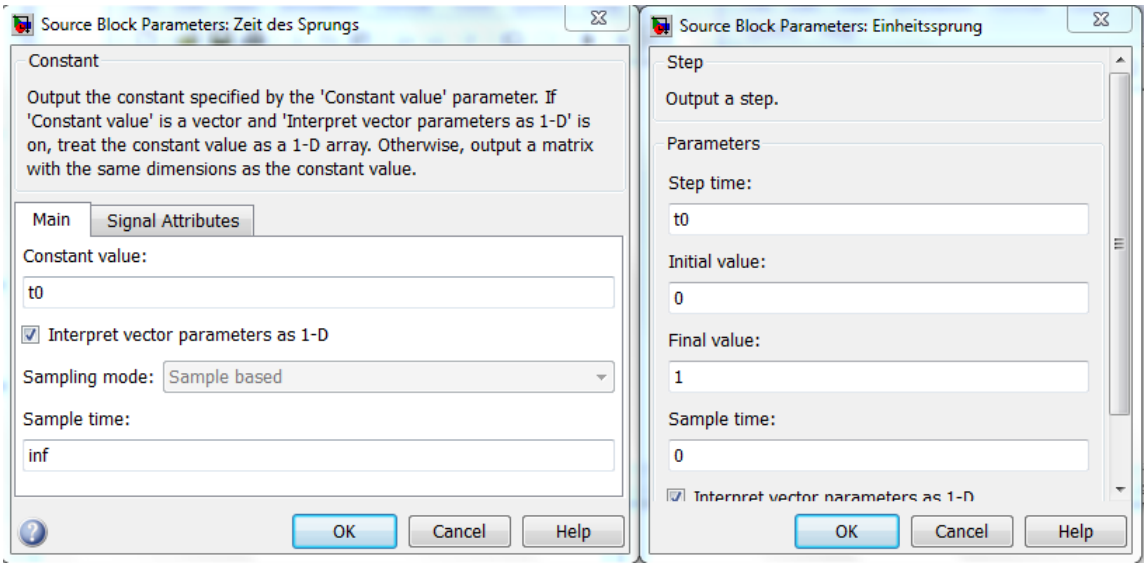
nung der Solltrajektorie und der entsprechenden Stellgrößen sollen folgende Informationen zugrunde liegen: Eine Durchschnittsgeschwindigkeit v_d, die aktuelle Ist-Position x_0, die Zielposition x_{soll} sowie der Zeitpunkt t_0, zu dem die Positionsänderung gewünscht wird. Machen Sie sich Gedanken um geeignete Funktionen zur Beschreibung der Trajektorie unter dem Gesichtspunkt sinnvollerweise einzuhaltender Bedingungen. Beispielsweise sollte die Geschwindigkeit des Wagens zu Beginn und beim Erreichen der Soll-Position gleich Null sein.

c) Implementieren Sie die analytische Funktion zur Trajektorien-Steuerung des IP02-Wagens aus Aufgabenteil b) mithilfe des *Embedded Matlab Function*-Blocks, zu finden unter *SIMULINK - User-Defined Functions*.

d) Vergleichen Sie zunächst lediglich simulativ die beiden Varianten zur Positionssteuerung des IP02-Wagens. Neben dem Vergleich der Ist-Positionen ist insbesondere der Vergleich der Stellgrößenverläufe von Interesse. Lässt sich bereits einschätzen, welche Strategie für den realen Anwendungsfall besser funktionieren wird?

e) Steuern Sie den realen IP02-Wagen mit beiden entworfenen Steuerungen. Speichern Sie sich für beide Varianten sowohl die gemessenen Ist-Positionen als auch die dazugehörigen Stellgrößen ab. Visualisieren Sie sich beide Ist-Positionsverläufe sowie Stellgrößenverläufe in jeweils einem Graphen mithilfe eines MATLAB-Skripts.

(a) Verwendung des Parameters $t0$ innerhalb des Subsystems.



(b) Hinzufügen der Variablen $t0$ zur Subsystem-Maske.

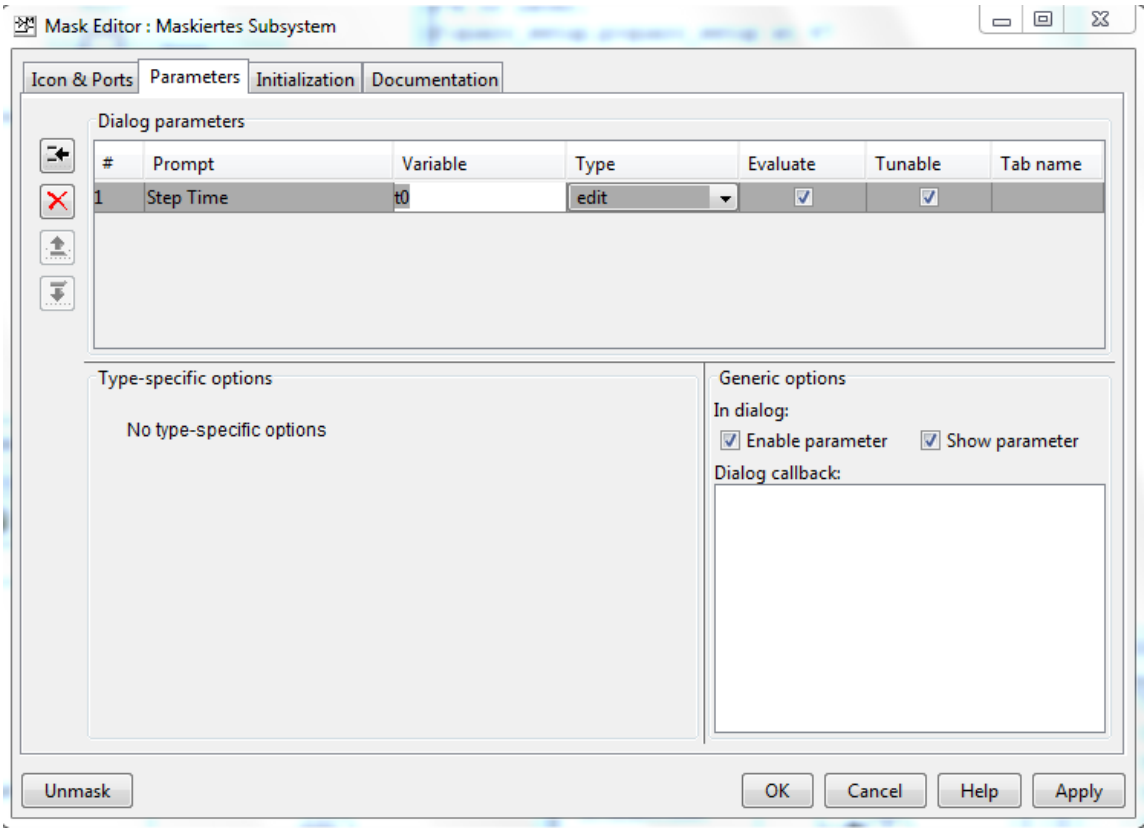
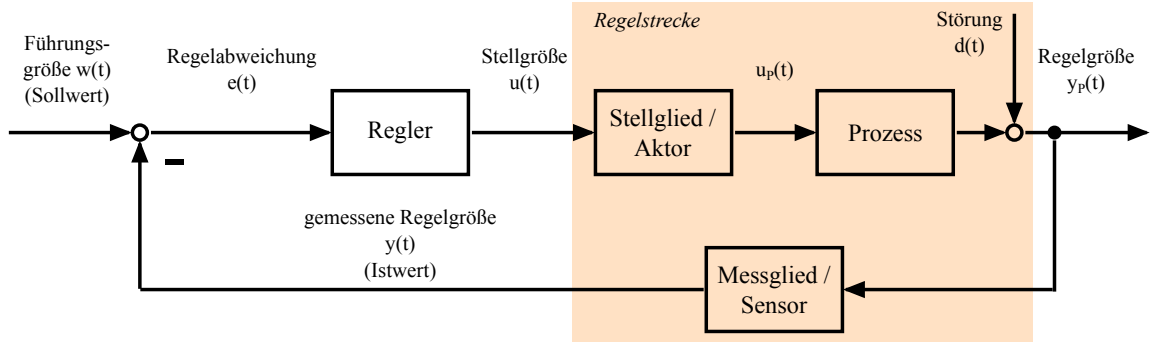


Bild 2.10: Definition der Variablen $t0$ für ein maskiertes Subsystem.

2.4 IP02 Positionsregelung

An dieser Stelle erfolgt zunächst wieder eine kurze Wiederholung der wichtigsten Grundlagen zum Thema Regelung. Unter einer Regelung versteht man einen technischen Vorgang in einem festgelegten System, bei dem diejenige Prozessgröße, die beeinflusst werden soll (Regelgröße), gemessen und ständig mit einem Vorgabewert verglichen wird. Als Ergebnis des Vergleiches erfolgt eine Stellgrößenausgabe, die eine Angleichung an den Vorgabewert (Führungsgröße oder Sollwert) als Ziel hat. Bild 2.11 (a) zeigt einen Regelkreis in Form eines Blockschaltbildes in allgemeiner Form mit wichtigen Begriffen der Regelungstechnik. Das Beispiel einer Geschwindigkeitsregelung in Bild 2.11 (b) soll dem besseren Verständnis der allgemeinen regelungstechnischen Begrifflichkeiten dienen. Im Unterschied zur Steuerung wird bei

(a) Allgemeines Blockschaltbild eines Regelkreises.



(b) Blockschaltbild einer Geschwindigkeitsregelung.

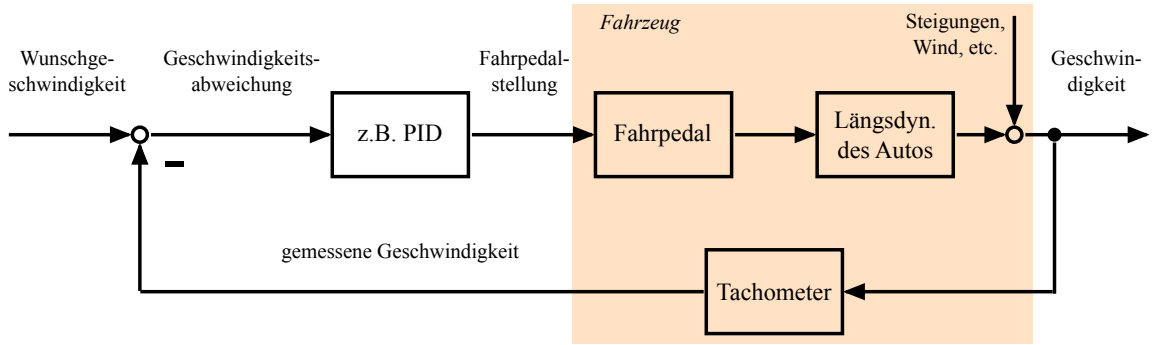


Bild 2.11: Blockschaltbild eines allgemeinen Regelkreises (a) und einer Geschwindigkeitsregelung (b).

der Regelung die Zielgröße gemessen und die aktuelle Regelabweichung bestimmt. Die Stellgröße greift aufgrund der Regelabweichung korrigierend ein, um Führungs- und Zielgröße anzunähern. Durch dieses Vorgehen ist es prinzipiell möglich Modellungenauigkeiten zu kompensieren sowie Störungen auszuregeln. Das bedeutet eine

Regelung weist eine gewisse Robustheit bezüglich Störungen und Modellungenauigkeiten auf.
Wichtige Übertragungsfunktionen des geschlossenen Regelkreises sind zum einen die Führungsübertragungsfunktion $G_{w\to y}(s)$ sowie die Störgrößenübertragungsfunktion $G_{d\to y}(s)$. Die Führungsübertragungsfunktion sollte über einen großen Frequenzbereich den Wert Eins aufweisen (die Regelgröße sollte der Führungsgröße folgen), die Störgrößenübertragungsfunktion sollte bestenfalls gleich Null sein (die Störung sollte die Regelgröße nicht beeinflussen). Um eine bestimmte Übertragungsfunktion im Regelkreis aufzustellen, ist folgende Regel hilfreich:

$$G(s) = \frac{\text{Vorwärtszweig}}{1 + G_0} \ . \tag{2.10}$$

Dabei bezeichnet den offenen Regelkreis $G_0 = G_R G_u G_S G_M$, für den Fall, dass die Übertragungsfunktionen entsprechend Bild 2.12 benannt sind. Oftmals ist die Über-

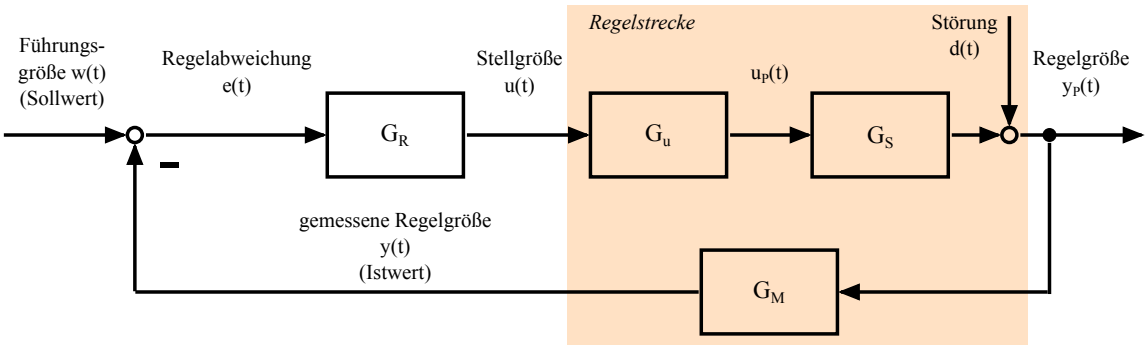


Bild 2.12: Blockschaltbild eines allgemeinen Regelkreises mit den Übertragungsfunktionen der einzelnen Blöcke.

tragungsfunktion des Stellglieds G_u bereits in der Übertragungsfunktion der Strecke G_S enthalten, sodass dieser Multiplikator zur Beschreibung des offenen Regelkreises G_O entfällt. Es bleibt noch zu klären, was in Gl. 2.10 mit *Vorwärtszweig* gemeint ist. Als Beispiel soll die Führungsübertragungsfunktion dienen, die mithilfe der in Bild 2.12 gewählten Bezeichnungen im Folgenden beschrieben wird. Für die Übertragungsfunktion von $w \to y$ startet man bei dem Signal w und schaut sich an, welche Blöcke auf dem Weg zum Signal y passiert werden. Alle diese Blöcke bilden zusammen den *Vorwärtszweig*, sodass sich die Führungsübertragungsfunktion zu

$$G_{w\to y}(s) = \frac{G_R G_u G_S}{1 + G_R G_u G_S G_M} \tag{2.11}$$

ergibt. Diese Übertragungsfunktion wird uns später noch nützlich sein, um Reglerparameter zu wählen respektive zu berechnen.

Die wichtigsten Forderungen an einen Regelkreis sind im Folgenden kurz aufgeführt:

1. Stabilität - dies sollte selbsterklärend sein.

2. Kompensation von Störungen und Folge der Führungsgröße (statische Anforderungen). Die statischen Anforderungen betreffen den eingeschwungenen Zustand, für den die Regelgröße $y(t)$ der Führungsgröße $w(t)$ exakt folgen sollte:

$$\lim_{t\to\infty}(w(t) - y(t)) = 0 \ . \tag{2.12}$$

   Um diese Anforderung erfüllen zu können, sollte im offenen Regelkreis die Laplace-Transformierte der Führungsgröße enthalten sein. Man spricht dabei vom sogenannten *Inneren Modell Prinzip*. Ist keine Laplace-Transformierte der Führungsgröße in der Strecke enthalten, sollte man diese im Regler realisieren. Zudem sollten - soweit vorhanden - Informationen zu vorhandenen Störgrößen verwendet werden. Ist beispielsweise eine bestimmte Störfrequenz bekannt, kann die entsprechende Laplace-Transformierte im Regler implementiert werden. Üblicherweise liegen solche Informationen jedoch nicht vor.

3. Dynamische Anforderungen: Sie betreffen beispielsweise das maximale Über- oder Unterschwingen, eine möglichst kurze Anstiegszeit sowie eine kurze Beruhigungszeit. Inwiefern die gewählten Reglerparameter diese Eigenschaften beeinflussen, kann bei gegebenem Führungsgrößenverlauf aus der Zeitlösung des gesamten Regelkreises ermittelt werden.

4. Robustheit, d.h. *genügend* Abstand zur Stabilitätsgrenze. Die Forderung nach Robustheit bezieht sich im Wesentlichen auf den Abstand des Regelkreises zur Stabilitätsgrenze. Ein gewisser Abstand wird in der Regel gefordert, um zu verhindern, dass durch Modellungenauigkeiten das reale System instabil wird. Zu diesen Ungenauigkeiten können sowohl nicht berücksichtigte physikalische Effekte beitragen, als auch zeitliche Änderungen der Regelstrecke, die aufgrund von Alterungsprozessen stattfinden. Aufschluss über den Abstand gibt beispielsweise die Ortskurve des offenen Regelkreises (Nyquist Kriterium), mit dessen Hilfe sich sowohl Amplitudenrand als auch Phasenrand bestimmen lassen.

Sollte man sich in den zuvor angesprochenen Themen nicht mehr so gut auskennen, lohnt ein Blick in das Regelungstechnik Skript. Interessant sind insbesondere Kapitel 10 (*Inneres Modell Prinzip* und dynamische Anforderungen) sowie Kapitel 8 und 9 ([qualitative] Stabilitätskriterien).

Mit den aufgefrischten Grundlagen zum Thema Regelung, wollen wir nun eine Positionsregelung für den IP02-Wagen entwerfen.

Aufgabe 5

a) Welche Struktur muss der zum Einsatz kommende Regler besitzen, um gemäß dem Prinzip des inneren Modells einen Sprung ohne bleibende Regelabweichung folgen zu können?

b) Berechnen Sie die Führungsübertragungsfunktion des in Bild 2.13 gezeigten Regelkreises, mit den Reglerparametern K_P und K_D.

c) Setzen Sie in der soeben aufgestellten Führungsübertragungsfunktion $K_D = 0$ und bestimmen Sie die Reglerverstärkung K_P, ab der der Regelkreis schwingungsfähig wird. Zeigen Sie außerdem rechnerisch, dass für einen Führungssprung keine bleibende Regelabweichung zu erwarten ist.

d) Realisieren Sie die Positionsregelung des IP02-Wagens zunächst lediglich mit dem Simulationsmodell des Prozesses, wobei K_D unberücksichtigt bleibt ($K_D = 0$). Geben Sie dazu als Führungsgröße ein Rechteck-Signal vor und überprüfen Sie das Verhalten für verschiedene Reglerparameter K_P, u.a. auch für Verstärkungen die zu schwingungsfähigem Verhalten führen.

e) Machen Sie nun Gebrauch von beiden zur Verfügung stehenden Reglerparametern. Bestimmen Sie die Reglerparameter K_P und K_D für eine allgemein vorgegebene maximale relative Überschwingweite Δm und Einschwingzeit T_r. Mit der Wahl von Δm und T_r legen Sie die Dämpfung D und die Eckfrequenz ω_0 einer allgemeinen Differentialgleichung 2. Ordnung fest:

$$\ddot{y}(t) + 2D\omega_0\dot{y}(t) + \omega_0^2 y(t) = K\omega_0^2 u(t) \ . \tag{2.13}$$

Verwenden Sie zum Berechnen dieser beiden Größen die folgenden beiden Gleichungen. Die erste Gleichung beschreibt unter Vorgabe einer maximalen

relativen Überschwingweite Δm die Dämpfung D:

$$D = \sqrt{\frac{\ln^2 \Delta m}{\ln^2 \Delta m + \pi^2}} \ . \tag{2.14}$$

Die zweite Gleichung beschreibt die Eckfrequenz ω_0 in Abhängigkeit der geforderten Einschwingzeit T_r. Zur Bestimmung der Einschwingzeit benötigt man eine eindeutige Definition, ab wann das Signal als eingeschwungen gilt. Dazu legt man um den Endwert des Systems ein Toleranzband $\pm\Delta\%$ und definiert die Sprungantwort als eingeschwungen, wenn dieses Toleranzband nicht mehr verlassen wird, siehe Bild 2.14. Die Gleichung zur Berechnung der Eckfrequenz lautet dann:

$$\omega_0 = \frac{\ln\left(\frac{\Delta\%}{100\%}\sqrt{1-D^2}\right)}{-DT_r} \ . \tag{2.15}$$

Setzen Sie für $\Delta\% = 5\%$ ein.

f) Implementieren Sie den berechneten PD-Regler in SIMULINK und betrachten Sie die Ergebnisse der Simulation für verschiedene Vorgaben von Δm und T_r. Lassen Sie sich neben der Führungs- und Regelgröße ebenfalls die Stellgröße ausgeben.

g) Schätzen Sie die Robustheit der sich ergebenden Regelkreise (d.h. mit P- und PD-Regler) anhand des sich ergebenden Amplituden- und/oder Phasenrandes ab.

h) Erstellen Sie schließlich ein SIMULINK-Modell zur Regelung des Realen IP02-Wagens und testen Sie sowohl die Variante mit reinem P-Regler als auch die Variante mit PD-Regler. Verwenden Sie als Führungsgrößen sowohl ein Rechtecksignal (Frequenz $f = 1/10$ Hz, Amplitude $A = 0.05$ m) als auch einen Sinusverlauf mit fester Amplitude $A = 0.1$ m, bei dem Sie die Frequenz sukzessive erhöhen (beginnend bei $f = 1/10$ Hz).

Bei den vorigen Untersuchungen sollte aufgefallen sein, dass beide Regelungsvarianten eine bleibende Regelabweichung nicht vermeiden können. Wie bereits zuvor bei der Steuerung des Systems festgestellt, besteht die Problematik, dass das System für kleine Stellgrößen nicht reagiert. Um die nicht gewünschte bleibende Regelabweichung zu beseitigen muss daher ein I-Anteil im Regler realisiert werden.

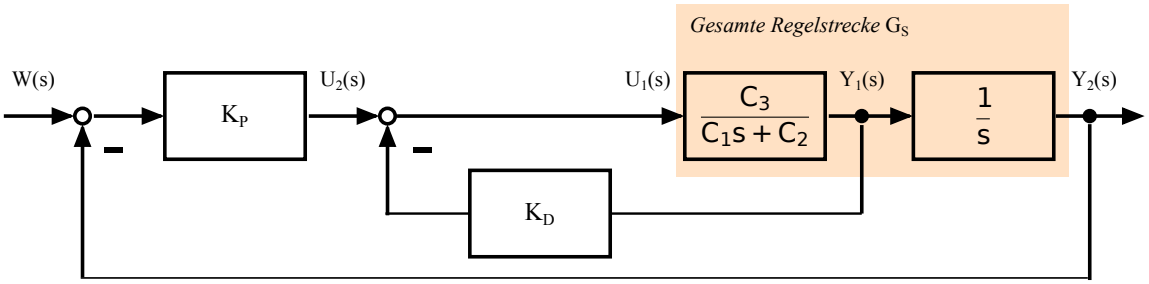


Bild 2.13: Blockschaltbild eines PD-Reglers, wobei der D-Anteil nicht im Vorwärtszweig des Regelkreises enthalten ist. Dies ist insbesondere bei sprunghaftem Verhalten der Führungsgröße von Vorteil.

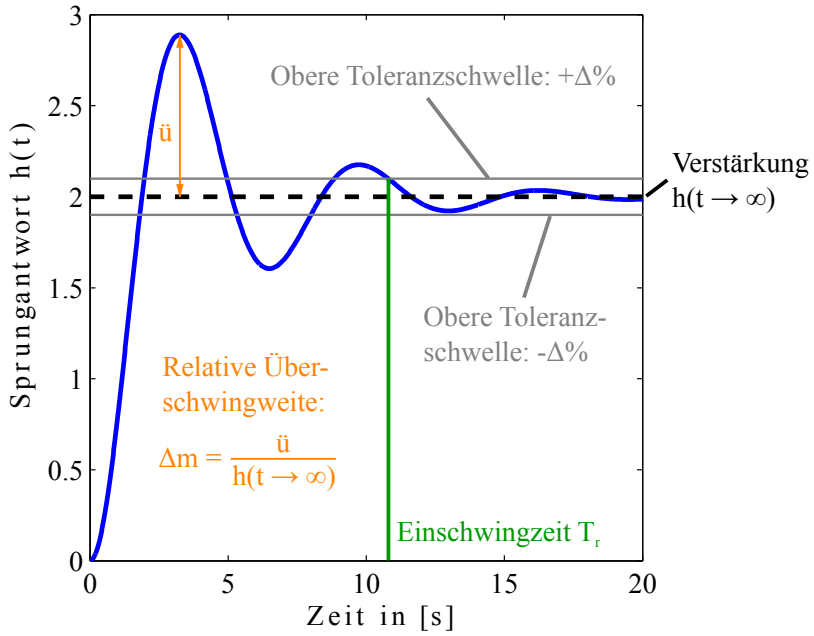


Bild 2.14: Illustration diverser Größen der Sprungantwort eines dynamischen Systems.

Aufgabe 6

a) Setzen Sie einen PID-Regler zur Positionsregelung des IP02-Wagens ein, wie er in Bild 2.15 zu sehen ist. Berechnen Sie dazu die Führungsübertragungsfunktion des sich ergebenden Regelkreises mit den Reglerparametern K_P, K_I und K_D.

b) Geben Sie zur Bestimmung der Reglerparameter Pole für die Führungsüber-

tragungsfunktion vor und bestimmen Sie die Reglerparameter K_P, K_I sowie K_D in Abhängigkeit der Pole. Schreiben Sie dazu ein MATLAB Skript, dass zu gegebenen Polen die entsprechenden Reglerparameter berechnet.

c) Testen Sie die gefundenen Reglerparameter in einem SIMULINK-Modell und überprüfen Sie die sich ergebenden Stellgrößen.

d) Lesen Sie anhand des Bode-Diagramms die Bandbreite des Regelkreises für die in der Simulation gefundenen Reglerparameter ab.

e) Realisieren Sie die Positionsregelung via PID-Regler für das reale System. Verwenden Sie als Führungsgrößen sowohl ein Rechtecksignal (Frequenz $f = 1/10$ Hz, Amplitude $A = 0.05$ m) als auch ein sogenanntes *Chirp*-Signal, bei dem die Frequenz linear mit der Zeit erhöht wird. Die Amplitude des Chirp-Signals sollte dabei $A = 0.075$ m betragen. Die langsamste Frequenz sollte bei $f_{Start} = 0.2$ Hz liegen, die höchste bei $f_{max} = 2$ Hz.

f) Bestimmen Sie (näherungsweise) die Bandbreite des realen Regelkreises mithilfe des Chirp-Signals. Passen Sie die Start- und Endfrequenz des Chirps aus Aufgabenteil e) sinnvoll an die abgelesene Bandbreite aus Aufgabenteil d) an (abgelesene Bandbreite $\pm$ 0.2 Hz).

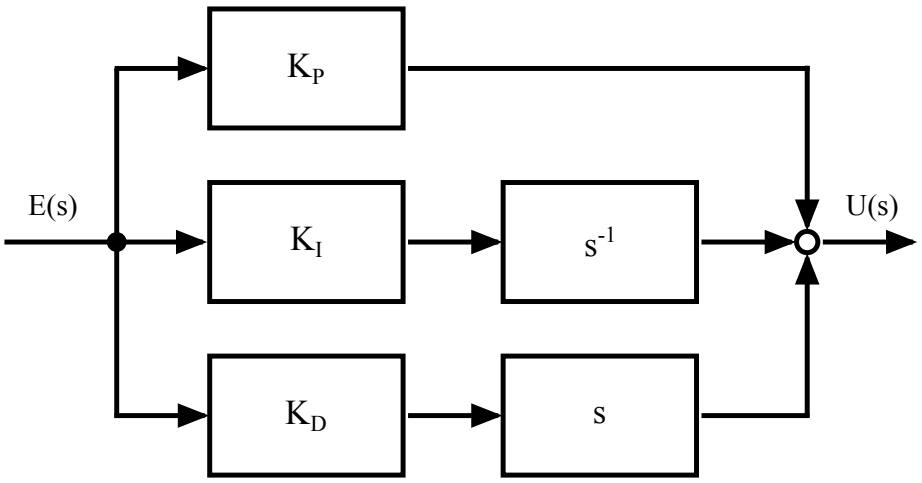


Bild 2.15: Blockschaltbild eines PID-Reglers.

3 IP02-Wagen mit gefederter Zusatzmasse

Im Folgenden ist das bereits aus Kapitel 2 bekannte System um einen zusätzlichen Wagen erweitert, der, wie in Bild 3.1 zu sehen, über ein Federelement angebunden ist. Die regelungstechnische Aufgabe besteht in der Positionsregelung des Zusatzwagens. Zu diesem Zweck wird das System in Abschnitt 3.1 in Zustandsraumdarstellung beschrieben. Abschnitt 3.2 behandelt die Ansteuerung und das Auslesen der entsprechenden Messwertaufnehmer. Schließlich behandelt Abschnitt 3.3 den Entwurf des Zustandsreglers.

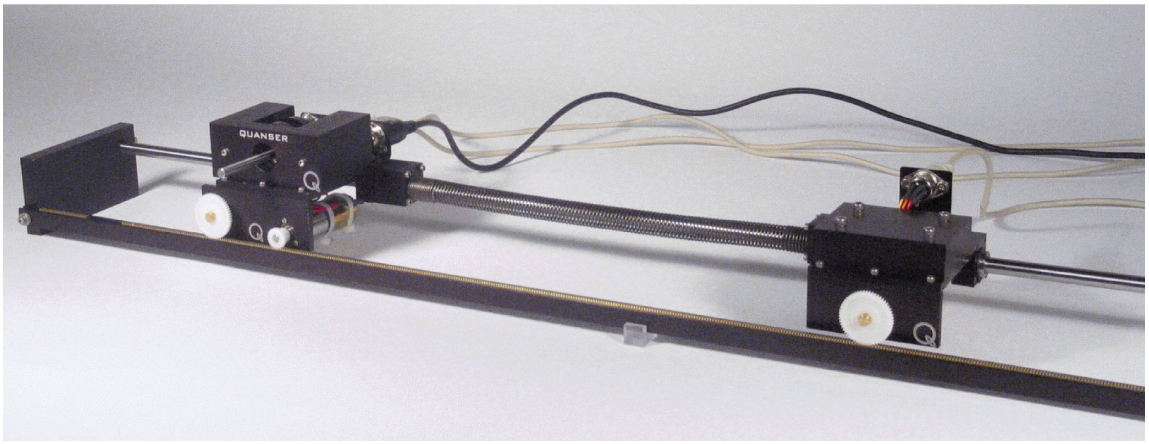


Bild 3.1: IP02-Wagen mit verbundenem Zusatzwagen.

3.1 Bewegungsgleichungen der Regelstrecke

Bevor es um die Bewegungsgleichungen der Regelstrecke an sich geht, erfolgt ein kurzer Einschub, der die wichtigsten Grundladen zur Beschreibung dynamischer Systeme im Zustandsraum beleuchtet. Dabei überführt man eine Differentialgleichung n-ter

Ordnung in n Differentialgleichungen 1. Ordnung. Das sich ergebende Gleichungssystem kann übersichtlich in einer Matrix-Vektor-Schreibweise dargestellt werden. Dies führt dazu, dass die Mächtigkeit der linearen Algebra voll ausgenutzt werden kann, für die bereits robuste numerische Verfahren (in MATLAB) verfügbar sind. Weitere Vorteile der Zustandsraumdarstellung sind die gute Erweiterbarkeit auf Mehrgrößensysteme, zeitvariante Systeme und nichtlineare Systeme. Zudem lassen sich komplexe Systeme hoher Ordnung übersichtlicher darstellen.

Das Ein-/Ausgangsverhalten eines linearen dynamischen Systems n-ter Ordnung mit Eingang $u(t)$ und Ausgang $y(t)$ lässt sich durch folgende Übertragungsfunktion beschreiben:

$$G(s) = \frac{b_m s^m + b_{m-1} s^{m-1} + \cdots + b_1 s + b_0}{s^n + a_{n-1} s^{n-1} + \cdots + a_1 s + a_0} \tag{3.1}$$

Im Zustandsraum lässt sich ein solches System wie folgt beschreiben:

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{b}u(t) \tag{3.2}$$

$$y(t) = \boldsymbol{c}^{\mathrm{T}}\boldsymbol{x}(t) + du(t)\ . \tag{3.3}$$

$\boldsymbol{A}$ bezeichnet dabei die Systemmatrix, $\boldsymbol{b}$ den Eingangsvektor, $\boldsymbol{c}^{\mathrm{T}}$ den Ausgangsvektor und d ist der Durchgriff, der bei nicht sprungfähigen Systemen entfällt. Der Zustandsvektor $\boldsymbol{x}$ beinhaltet sämtliche Zustände des beschriebenen Systems, $\dot{\boldsymbol{x}}$ die entsprechenden zeitlichen Ableitungen. Ein Blockschaltbild der Zustandsraumdarstellung findet sich in Bild 3.2. Mithilfe des Zustandsvektors, der alle Zustände des Systems beinhaltet, den Anfangsbedingungen $\boldsymbol{x}_0$ und dem Verlauf der Eingangsgröße $u(t)$ lässt sich der Verlauf der Ausgangsgröße $y(t)$ berechnen.

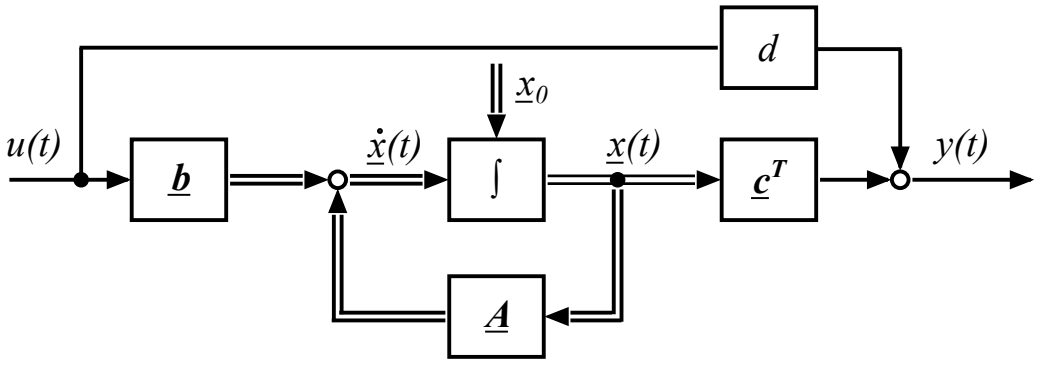


Bild 3.2: Zustandsraumdarstellung als Blockschaltbild mit den Anfangszuständen $\boldsymbol{x}_0$.

Zur Veranschaulichung der Zustandsraumdarstellung, sei als nächstes eine allgemeine Differentialgleichung 3. Ordnung

$$a_3 \cdot \dddot{y} + a_2 \cdot \ddot{y} + a_1 \cdot \dot{y} + a_0 \cdot y = b_0 \cdot u \tag{3.4}$$

in die Zustandsraumdarstellung überführt. Gleichung 3.4 wird derart umgeformt, dass nach der höchsten vorkommenden zeitlichen Ableitung freigestellt wird:

$$\dddot{y} = \frac{b_0}{a_3} \cdot u - \frac{a_2}{a_3} \cdot \ddot{y} - \frac{a_1}{a_3} \cdot \dot{y} - \frac{a_0}{a_3} \cdot y \ . \tag{3.5}$$

Die Zustände des Systems sind, etwas salopp formuliert, alle Signale rechtsseitig des Gleichheitszeichens, die nicht Eingangssignale oder Störgrößen sind. Daraus ergibt sich der Zustandsvektor

$$\boldsymbol{x}(t) = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} y \\ \dot{y} \\ \ddot{y} \end{bmatrix} \tag{3.6}$$

und dessen zeitliche Ableitung

$$\dot{\boldsymbol{x}}(t) = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} \dot{y} \\ \ddot{y} \\ \dddot{y} \end{bmatrix} \ . \tag{3.7}$$

Für jede einfach abgeleitete Zustandsgröße wird eine Gleichung benötigt. Gleichung 3.5 liefert diese für die zeitliche Ableitung von Zustand drei ($\dot{x}_3$). Die verbleibenden Gleichungen lauten:

$$\dot{x}_1 = x_2 \tag{3.8}$$

$$\dot{x}_2 = x_3 \ . \tag{3.9}$$

Mithilfe dieser Gleichungen lassen sich nun die Systemmatrix $\boldsymbol{A}$, der Eingangsvektor $\boldsymbol{b}$, der Ausgangsvektor $\boldsymbol{c}^T$ sowie der Durchgriff d bestimmen. Die vollständige

Zustandsraumdarstellung der Differentialgleichung 3. Ordnung lautet somit:

$$\underbrace{\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\\ \dot{x}_3\end{bmatrix}}_{\dot{\boldsymbol{x}}(t)} = \underbrace{\begin{bmatrix}0 & 1 & 0\\ 0 & 0 & 1\\ -\frac{a_0}{a_3} & -\frac{a_1}{a_3} & -\frac{a_2}{a_3}\end{bmatrix}}_{\boldsymbol{A}} \underbrace{\begin{bmatrix}x_1\\ x_2\\ x_3\end{bmatrix}}_{\boldsymbol{x}(t)} + \underbrace{\begin{bmatrix}0\\ 0\\ \frac{b_0}{a_3}\end{bmatrix}}_{\boldsymbol{b}} u(t) \tag{3.10}$$

$$y(t) = \underbrace{\begin{bmatrix}1 & 0 & 0\end{bmatrix}}_{\boldsymbol{c}^T} \underbrace{\begin{bmatrix}x_1\\ x_2\\ x_3\end{bmatrix}}_{\boldsymbol{x}(t)} . \tag{3.11}$$

Da das System nicht sprungfähig ist, ist der Durchgriff $d = 0$.
Zwei wichtige Eigenschaften der Zustandsgleichungen sind die sogenannte *Steuerbarkeit* und die *Beobachtbarkeit*. Als steuerbar gilt ein System, wenn es in endlicher Zeit von jedem beliebigen Anfangszustand durch einen geeignet gewählten Verlauf der Eingangsgröße in jeden beliebigen Endzustand überführt werden kann. Um ein System auf Steuerbarkeit zu überprüfen muss der Rang der *Steuerbarkeitsmatrix* $\boldsymbol{S}_S$ gleich der Anzahl n an Differentialgleichungen 1. Ordnung sein (n: Anzahl der Zeilen der Systemmatrix $\boldsymbol{A}$ oder des Eingangsvektors $\boldsymbol{b}$), die das System beschreiben. Die Steuerbarkeitsmatrix ist definiert als:

$$\boldsymbol{S}_S = \begin{bmatrix}\boldsymbol{b} & \boldsymbol{A}\boldsymbol{b} & \boldsymbol{A}^2\boldsymbol{b} & \cdots & \boldsymbol{A}^{n-1}\boldsymbol{b}\end{bmatrix} . \tag{3.12}$$

Als beobachtbar gilt ein System, wenn der Anfangszustand aus dem über einem endlichen Intervall bekannten Verlauf der Ein- und Ausgangsgröße bestimmt werden kann. Um ein System auf Beobachtbarkeit zu überprüfen, bestimmt man den Rang der *Beobachtbarkeitsmatrix* $\boldsymbol{S}_B$, der ebenfalls mit n, d.h. der Anzahl an Differentialgleichungen 1. Ordnung, übereinstimmen muss. Die Beobachtbarkeitsmatrix ist definiert als:

$$\boldsymbol{S}_B = \begin{bmatrix}\boldsymbol{c}^T\\ \boldsymbol{c}^T\boldsymbol{A}\\ \boldsymbol{c}^T\boldsymbol{A}^2\\ \cdots\\ \boldsymbol{c}^T\boldsymbol{A}^{n-1}\end{bmatrix} . \tag{3.13}$$

An dieser Stelle sei noch erwähnt, dass Systeme in einen steuerbaren, einen beobachtbaren, einen nicht steuerbaren und nicht beobachtbaren Teil zerlegt werden

können - dies würde an dieser Stelle aber zu weit führen. Weitere Details bezüglich der Zustandsraumdarstellung und den Eigenschaften der Zustandsgleichungen sind entsprechender Fachliteratur zu entnehmen.

Aufgabe 7

a) Überführen Sie folgende Differentialgleichung in die Zustandsraumdarstellung:

$$a_2 \cdot \ddot{y} + a_0 \cdot y = b_0 \cdot u \ . \tag{3.14}$$

b) Beschreiben Sie in Zustandsraumdarstellung einen Doppelintegrator.

c) Prüfen Sie die beiden Systeme aus Aufgabenteil a) und b) auf Steuerbarkeit und Beobachtbarkeit.

Da nun die theoretischen Grundlagen bezüglich der Zustandsraumdarstellung aufgefrischt sind, widmen wir uns nun der regelungstechnischen Aufgabe: Positionsregelung der gefederten, nicht angetriebenen Masse. Die das Gesamtsystem beschreibenden Bewegungsgleichungen sind im Folgenden gegeben:

$$\ddot{x}_c = -\frac{K_s}{M_c}x_c + \frac{K_s}{M_c}x_2 - \frac{B_{eq}}{M_c}\dot{x}_c + \frac{F_c}{M_c} \ , \tag{3.15}$$

$$\ddot{x}_2 = \frac{K_s}{M_{c2}}x_c - \frac{K_s}{M_{c2}}x_2 - \frac{B_{eq2}}{M_{c2}}\dot{x}_2 \ . \tag{3.16}$$

Die Antriebskraft F_c resultiert aus einer Spannung U_M, die am Elektromotor des IP02-Wagens anliegt. Die Gleichung zur Berechnung der Kraft aus einer Spannung sollte bereits aus Kapitel 2 bekannt sein, sei aber trotzdem hier noch mal aufgeführt:

$$F_c = -\frac{\eta_g K_g^2 \eta_m K_t K_m}{R_m r_{mp}^2}\dot{x}_c + \frac{\eta_g K_g \eta_m K_t}{R_m r_{mp}}U_M \ . \tag{3.17}$$

Zur übersichtlicheren Darstellung und einfacheren Implementierung in den Programm-

code bzw. SIMULINK werden nachfolgend einige Hilfskonstanten definiert.

$$H_1 = -\frac{K_s}{M_c} \tag{3.18}$$

$$H_2 = \frac{K_s}{M_c} \tag{3.19}$$

$$H_3 = \frac{B_{eq}}{M_c} \tag{3.20}$$

$$H_4 = \frac{K_s}{M_{c2}} \tag{3.21}$$

$$H_5 = -\frac{K_s}{M_{c2}} \tag{3.22}$$

$$H_6 = -\frac{B_{eq2}}{M_{c2}} \tag{3.23}$$

Nun werden die soeben definierten Konstanten in Gleichung 3.15 und 3.16 zusammen Gleichung 3.17 eingesetzt.

$$\ddot{x}_c = H_1 x_c + H_2 x_2 + H_3 \dot{x}_c + \frac{1}{M_c}\left[-\frac{\eta_g K_g^2 \eta_m K_t K_m}{R_m r_{mp}^2}\dot{x}_c + \frac{\eta_g K_g \eta_m K_t}{R_m r_{mp}} U_M\right] , \tag{3.24}$$

$$\ddot{x}_2 = H_4 x_c + H_5 x_2 + H_6 \dot{x}_2 . \tag{3.25}$$

Zwei weitere Konstanten seien durch

$$H_7 = -\frac{\eta_g K_g^2 \eta_m K_t K_m}{M_c R_m r_{mp}^2} \text{ und} \tag{3.26}$$

$$H_8 = \frac{\eta_g K_g \eta_m K_t}{M_c R_m r_{mp}} \tag{3.27}$$

gegeben. Damit vereinfacht sich Gleichung 3.24 weiter, sodass sich das Gesamtgleichungssystem wie folgt darstellt:

$$\ddot{x}_c = H_1 x_c + H_2 x_2 + (H_3 + H_7)\dot{x}_c + H_8 U_M . \tag{3.28}$$

$$\ddot{x}_2 = H_4 x_c + H_5 x_2 + H_6 \dot{x}_2 . \tag{3.29}$$

Aufgabe 8

a) Definieren Sie die Zustandsraummatrizen und -vektoren, die die Regelstrecke

beschreiben in MATLAB.

b) Prüfen Sie, ob die Regelstrecke vollständig steuerbar ist.

c) Prüfen Sie, ob die Regelstrecke vollständig beobachtbar ist.

d) Definieren Sie in MATLAB ein Zustandsraummodell und lassen Sie sich mit dem *pole*-Befehl die Pole der Regelstrecke anzeigen.

3.2 Praktische Ansteuerung und Messwertaufnahme

Zur Ansteuerung und Messwertaufnahme sind bereits in Abschnitt 2.2 alle nötigen Grundlagen genauer erläutert worden. Im Unterschied zu den dort beschriebenen Voraussetzungen erhält man durch das Auslesen des Encoder-Ausgangs 1 (siehe Bild 3.3) nicht den aktuellen Winkel der Pendelachse, sondern die Position x_2 des über eine Feder angebundenen Wagens. Da der gleiche Sensor im gefederten wie auch im angetriebenen Wagen zum Einsatz kommt, muss in dem entsprechenden *Gain*-Block, wie in Bild 3.3 zu sehen, der gleiche Faktor eingetragen sein.

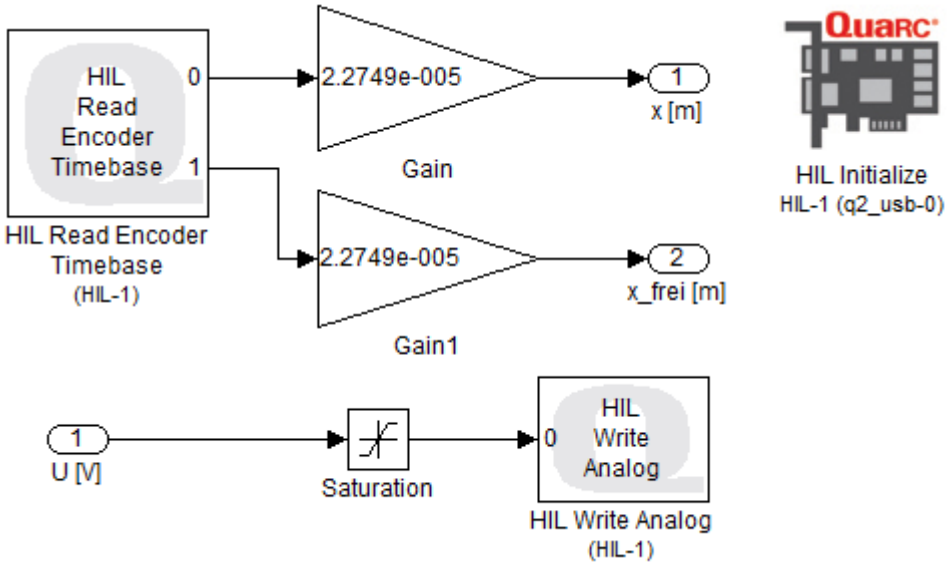


Bild 3.3: SIMULINK-Schaltbild zum Ansteuern der Regelstrecke und zum Auslesen der Wegsensoren.

Aufgabe 9

Vergleichen Sie das Modell mit dem realen System. Erstellen Sie zu diesem Zweck eine SIMULINK-Datei, bei der Sie als Anregungssignal einen Sprung von 0 auf 5 Volt verwenden, der nach 1.3 Sekunden wieder von 5 auf 0 Volt zurückspringt. Verwenden Sie zur Erzeugung des Eingangssignals den *Signal Builder*-Block (zu finden unter *SIMULINK* → *Sources*). Vergleichen Sie alle Zustände des simulierten und realen Systems.

3.3 Positionsregelung Zusatzmasse

Das für das vorliegende System zum Einsatz kommende Verfahren zur Reglerauslegung ist bereits aus vorigen Abschnitten bekannt. Es soll eine Reglerparametrierung durch Polvorgabe erfolgen. Zu diesem Zweck wird zunächst etwas Theorie bezüglich des Zustandsreglers wiederholt. Die Grundidee der Zustandsregelung ist es die Zustände des Systems rückzukoppeln - im Gegensatz zum Standardregelkreis, bei dem nur die Regelgröße zurückgeführt wird. Somit kann man einen Zustandsregler als eine Art erweiterte Kaskadenregelung auffassen. Das Blockschaltbild einer Zustandsregelung ist in Bild 3.4 zu sehen. Im folgenden sind lediglich Systeme ohne Durchgriff behandelt. Eine Erweiterung der Theorie auf Systeme mit Durchgriff ist der entsprechenden Literatur zu entnehmen. Wie in Bild 3.4 zu erkennen ist, werden alle

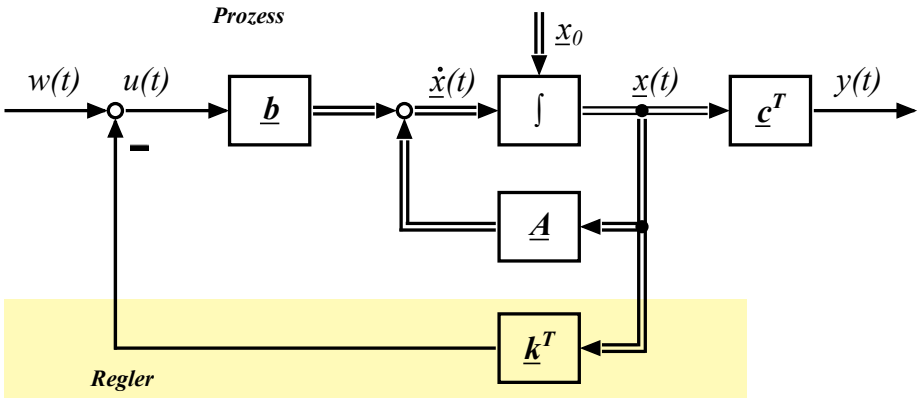


Bild 3.4: Blockschaltbild einer Zustandsregelung eines Systems ohne Durchgriff.

Zustände zurückgekoppelt und mit den Reglerparametern gewichtet aufsummiert.

Durch dieses Vorgehen ist eine Interpretation der einzelnen Reglerparameter so gut wie nicht möglich. Dennoch lässt sich sagen, dass es sich um eine Art PD_{n-1}-Regler handelt, mit einem entscheidenden Unterschied: Es wird nicht die Regelabweichung abgeleitet, sondern es werden die Regelgröße und deren Ableitungen zurückgeführt. D.h. falls entsprechende Sensoren vorhanden sind, die die zeitlichen Ableitungen der Regelgröße direkt messen können, müssen die Ableitungen nicht explizit berechnet werden. Sind nicht für alle Zustände Sensoren vorhanden, kann entweder ein Beobachter zum Einsatz kommen (dazu mehr im nächsten Kapitel) oder eine explizite Ableitungsberechnung ist unumgänglich (diese Strategie verfolgen wir in diesem Abschnitt, da keine Sensoren zur Messung der Wagengeschwindigkeiten vorhanden sind).
Um nun die Reglerparameter via Polvorgabe durchführen zu können, wird der Zusammenhang der Reglerparameter mit der charakteristischen Gleichung benötigt. Die Zustandsgleichungen, die das in Bild 3.4 gezeigte (zustandsgeregelte) System beschreiben, lauten wie folgt:

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{b}u(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{b}\left(w - \boldsymbol{k}^T\boldsymbol{x}\right) \tag{3.30}$$

$$y(t) = \boldsymbol{c}^{\mathrm{T}}\boldsymbol{x}(t)\ . \tag{3.31}$$

Die Zustandsgleichung (Gl. 3.30) lässt sich weiter zu

$$\dot{\boldsymbol{x}}(t) = \left(\boldsymbol{A} - \boldsymbol{b}\boldsymbol{k}^T\right)\boldsymbol{x}(t) + \boldsymbol{b}w \tag{3.32}$$

zusammenfassen. Überführt man die Gleichung 3.31 und 3.32 in den Laplace-Bereich, ergeben sich die folgenden beiden Gleichungen:

$$s\boldsymbol{X}(s) = \left(\boldsymbol{A} - \boldsymbol{b}\boldsymbol{k}^T\right)\boldsymbol{X}(s) + \boldsymbol{b}W \tag{3.33}$$

$$Y(s) = \boldsymbol{c}^{\mathrm{T}}\boldsymbol{X}(s)\ . \tag{3.34}$$

Wir sind nun an der Übertragungsfunktion von $W \rightarrow Y$ des geschlossenen Regelkreises G_w interessiert. Dazu lösen wir zunächst Gl. 3.33 nach $X(s)$ auf und setzen das Ergebnis in Gl. 3.34 ein:

$$\boldsymbol{X}(s) = \left(s\boldsymbol{I} - \boldsymbol{A} + \boldsymbol{b}\boldsymbol{k}^T\right)^{-1}\boldsymbol{b}W \tag{3.35}$$

$$Y(s) = \boldsymbol{c}^{\mathrm{T}}\left(s\boldsymbol{I} - \boldsymbol{A} + \boldsymbol{b}\boldsymbol{k}^T\right)^{-1}\boldsymbol{b}W\ . \tag{3.36}$$

$\boldsymbol{I}$ ist dabei die Einheitsmatrix (gleiche Dimensionen wie die Systemmatrix, die lediglich auf der Hauptdiagonalen mit Einsen besetzt ist). Damit ergibt sich die Übertragungsfunktion zu:

$$G_w(s) = \boldsymbol{c}^{\mathrm{T}} \left(s\boldsymbol{I} - \boldsymbol{A} + \boldsymbol{b}\boldsymbol{k}^T \right)^{-1} \boldsymbol{b} \ . \tag{3.37}$$

Uns interessiert nun der Nenner dieser Übertragungsfunktion, der gleichbedeutend mit der charakteristischen Gleichung ist. Da sich im Allgemeinen die Inverse einer regulären Matrix $\boldsymbol{M}$ zu

$$\boldsymbol{M}^{-1} = \frac{1}{\det(\boldsymbol{M})} \cdot \mathrm{adj}(\boldsymbol{M}) \tag{3.38}$$

berechnet, und $\mathrm{adj}(\boldsymbol{M})$ für die adjunkte der Matrix $\boldsymbol{M}$ steht, entspricht die charakteristische Gleichung der Determinanten des Klammerausdrucks aus Gleichung 3.37:

$$\text{char. Gleichung} = \det \left(s\boldsymbol{I} - \boldsymbol{A} + \boldsymbol{b}\boldsymbol{k}^T \right) \ . \tag{3.39}$$

Berechnet man nun diese Determinante und setzt den sich ergebenden Ausdruck mit dem Wunschpolynom gleich, erhält man:

$$s^n + a_{n-1}s^{n-1} + \cdots + a_1 s + a_0 = s^n + n_{n-1}s^{n-1} + \cdots + n_1 s + n_0 \ . \tag{3.40}$$

Bei dieser Gleichung bezeichnen die Koeffizienten a_0 bis a_{n-1} die Vorfaktoren, die sich aus der Berechnung der Determinante ergeben. n_0 bis n_{n-1} sind die Wunschkoeffizienten, die aus den vorgegebenen Polen herrühren. Die Berechnung der a_i-Koeffizienten mit $i = 0, 1, \cdots, n-1$ und der anschließende Vergleich mit den gewünschten Koeffizienten muss jedoch nicht explizit durchgeführt werden. Mit einigen Rechenschritten und dem Umweg über eine Transformation eines beliebigen Systems in die Regelungsnormalform, lässt sich die Formel nach Ackermann herleiten, mit deren Hilfe die Reglerparameter $\boldsymbol{k}^T$ auf folgende Weise berechnet werden können:

$$\boldsymbol{k}^T = \begin{bmatrix} n_0 & n_1 & \cdots & n_{n-1} & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{s}_S^T \\ \boldsymbol{s}_S^T\boldsymbol{A} \\ \boldsymbol{s}_S^T\boldsymbol{A}^2 \\ \vdots \\ \boldsymbol{s}_S^T\boldsymbol{A}^{n-1} \\ \boldsymbol{s}_S^T\boldsymbol{A}^n \end{bmatrix} \ . \tag{3.41}$$

Der Vektor $\boldsymbol{s}_S^T$ steht dabei für die letzte Zeile der inversen Steuerbarkeitsmatrix des Originalsystems. Auf die ausführliche Herleitung der Ackermann-Formel wird an dieser Stelle verzichtet. Interessierte finden diese beispielsweise in [1].
In der hier vorgestellten Form der Zustandsregelung ($\boldsymbol{k}^T$ in der Rückführung der Zustände - es gibt auch andere Darstellungsformen) muss beim Anwendungsfall einer Folgeregelung darauf geachtet werden, dass keine bleibende Regelabweichung bestehen bleibt. Um dies sicherzustellen kann man entweder einen statischen Vorfilter verwenden, oder einen I-Anteil in den Regler integrieren (sofern dieser nicht bereits in der Regelstrecke enthalten ist). Zunächst soll lediglich die Methode mithilfe des Vorfilters näher betrachtet werden, die Integration eines I-Anteils in den Regler folgt am Ende diesen Abschnitts. Der statische Vorfilter soll die Führungsgröße derart skalieren, dass bei einem Sprung der Führungsgröße $w(t) = \sigma(t)$ keine bleibende Regelabweichung entsteht. Um diesen Faktor zu bestimmen, berechnet man die Verstärkung des geschlossenen Regelkreises und verwendet die Inverse dieses Wertes als Vorfilter v:

$$v^{-1} = y(t \to \infty) = \lim_{s \to 0} sG_w \underbrace{\frac{1}{s}}_{\text{Sprung im Bildbereich}} \tag{3.42}$$

$$= \lim_{s \to 0} \boldsymbol{c}^{\mathrm{T}} \left(s\boldsymbol{I} - \boldsymbol{A} + \boldsymbol{b}\boldsymbol{k}^T \right)^{-1} \boldsymbol{b} \tag{3.43}$$

$$= \boldsymbol{c}^{\mathrm{T}} \left(-\boldsymbol{A} + \boldsymbol{b}\boldsymbol{k}^T \right)^{-1} \boldsymbol{b} \tag{3.44}$$

$$\Leftrightarrow v = \frac{1}{\boldsymbol{c}^{\mathrm{T}} \left(-\boldsymbol{A} + \boldsymbol{b}\boldsymbol{k}^T \right)^{-1} \boldsymbol{b}} \tag{3.45}$$

Aufgabe 10

a) Berechnen Sie mithilfe der Ackermann-Formel Reglerparameter, wenn Sie alle Pole des geschlossenen Regelkreises an die gleiche Stelle p legen. Wählen Sie einen sinnvollen Startwert für p unter Berücksichtigung der Pole der Regelstrecke.

b) Definieren Sie die Übertragungsfunktion des geschlossenen Regelkreises in MATLAB mit den Reglerparametern aus Aufgabenteil b). Überprüfen Sie mithilfe der *step*-Funktion, ob der in Aufgabenteil a) bestimmte Verstärkungsfaktor zum gewünschten Führungsübertragungsverhalten führt.

c) Erstellen Sie ein SIMULINK-Modell, mit dessen Hilfe Sie die Zustandsregelung simulieren können. Überprüfen Sie die sich ergebende Stellgröße und verändern Sie den gewählten Pol p, bis sich für einen Führungssprung auf 0.15 m sinnvolle Stellgrößen ergeben.

d) Realisieren Sie die Zustandsregelung für das reale System mit den in Aufgabenteil d) gefundenen Reglerparametern.

Im Folgenden soll kurz dargestellt werden, wie man einen zusätzlichen I-Anteil in den Zustandsregler implementieren kann. Dieser wird für eine Regelung ohne bleibende Regelabweichung für das vorliegende System benötigt, da die Strecke für kleine Stellgrößen faktisch keinen I-Anteil aufweist. Der integrierende Anteil wird, wie in Bild 3.5 dargestellt, in den Vorwärtszweig der Zustandsregelung implementiert. Durch

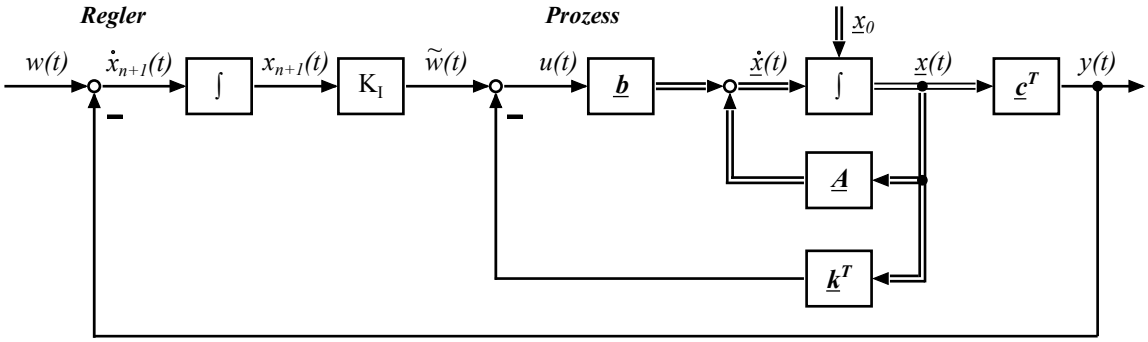


Bild 3.5: Blockschaltbild eines PI-Zustandsreglers (ohne Durchgriff).

das Hinzufügen eines weiteren Integrators entsteht ein weiterer Zustand x_{n+1}, für den man eine weitere Gleichung aufstellen kann:

$$\dot{x}_{n+1} = w - y \tag{3.46}$$

$$= w - \boldsymbol{c}^T \boldsymbol{x} \ . \tag{3.47}$$

Diese Gleichung kann man mit den Systemmatrizen und -vektoren der Gleichungen

3.2 und 3.3 in folgender Weise zu einer Matrix-Vektor-Gleichung zusammenfassen:

$$\underbrace{\begin{bmatrix} \dot{\boldsymbol{x}} \\ \dot{x}_{n+1} \end{bmatrix}}_{\dot{\widetilde{\boldsymbol{x}}}} = \underbrace{\begin{bmatrix} \boldsymbol{A} & \boldsymbol{0} \\ -\boldsymbol{c}^T & 0 \end{bmatrix}}_{\widetilde{\boldsymbol{A}}} \underbrace{\begin{bmatrix} \boldsymbol{x} \\ x_{n+1} \end{bmatrix}}_{\widetilde{\boldsymbol{x}}} + \underbrace{\begin{bmatrix} \boldsymbol{b} \\ 0 \end{bmatrix}}_{\widetilde{\boldsymbol{b}}} u + \underbrace{\begin{bmatrix} \boldsymbol{0} \\ 1 \end{bmatrix}}_{\boldsymbol{f}} w \tag{3.48}$$

$$y = \underbrace{\begin{bmatrix} \boldsymbol{c}^T & 0 \end{bmatrix}}_{\widetilde{\boldsymbol{c}}^T} \underbrace{\begin{bmatrix} \boldsymbol{x} \\ x_{n+1} \end{bmatrix}}_{\widetilde{\boldsymbol{x}}} . \tag{3.49}$$

Oder in Notation mit den neu eingeführten Schlange-Größen:

$$\dot{\widetilde{\boldsymbol{x}}} = \widetilde{\boldsymbol{A}}\widetilde{\boldsymbol{x}} + \widetilde{\boldsymbol{b}}u + \boldsymbol{f}w \tag{3.50}$$

$$y = \widetilde{\boldsymbol{c}}^T\widetilde{\boldsymbol{x}} . \tag{3.51}$$

Die Stellgröße u berechnet sich, wie aus Bild 3.5 zu erkennen ist, zu:

$$u = x_{n+1}K_I - \boldsymbol{k}^T\boldsymbol{x} , \tag{3.52}$$

was sich mithilfe des neu eingeführten Zustandsvektors $\tilde{\boldsymbol{x}}$ auch wie folgt darstellen lässt:

$$u = -\underbrace{\begin{bmatrix} \boldsymbol{k}^T & -K_I \end{bmatrix}}_{\widetilde{\boldsymbol{k}}^T} \begin{bmatrix} \boldsymbol{x} \\ x_{n+1} \end{bmatrix} . \tag{3.53}$$

Setzt man diese Reglergleichung in Gl. 3.50 ein, erhält man man schließlich die Zustandsdarstellung des PI-Zustandsreglers:

$$\dot{\widetilde{\boldsymbol{x}}} = \left(\widetilde{\boldsymbol{A}} - \widetilde{\boldsymbol{b}}\widetilde{\boldsymbol{k}}^T\right)\widetilde{\boldsymbol{x}} + \boldsymbol{f}w \tag{3.54}$$

$$y = \widetilde{\boldsymbol{c}}^T\widetilde{\boldsymbol{x}} . \tag{3.55}$$

In dieser Darstellung lassen sich alle bereits für den Standard-Zustandsregler bekannten Verfahren zur Reglerauslegung anwenden - es müssen lediglich die Schlange-Größen verwendet werden. Analog zur Herleitung der Führungsübertragungsfunktion des geschlossenen Regelkreises des Zustandsreglers ohne I-Anteil, ergibt sich diese für den PI-Zustandsregler zu:

$$G_w = \widetilde{\boldsymbol{c}}^T\left(s\boldsymbol{I} - \widetilde{\boldsymbol{A}} + \widetilde{\boldsymbol{b}}\widetilde{\boldsymbol{k}}^T\right)^{-1}\boldsymbol{f} . \tag{3.56}$$

Ebenso gelten für die Schlange-Größen die Definitionen der charakteristischen Gleichung und der Ackermann-Formel zur Polvorgabe des zu regelnden Systems.

Aufgabe 11

a) Berechnen Sie mithilfe der Ackermann-Formel für den PI-Zustandsregler Reglerparameter, wenn Sie alle Pole des geschlossenen Regelkreises an die gleiche Stelle p legen. Wählen Sie einen sinnvollen Startwert für p unter Berücksichtigung der Pole der Regelstrecke.

b) Definieren Sie die Übertragungsfunktion des geschlossenen Regelkreises in MATLAB mit den Reglerparametern aus Aufgabenteil a). Überprüfen Sie mithilfe der *step*-Funktion, ob die Führungsübertragungsfunktion eine Verstärkung von 1 besitzt.

c) Erstellen Sie ein SIMULINK-Modell, mit dessen Hilfe Sie die PI-Zustandsregelung simulieren können. Überprüfen Sie die sich ergebende Stellgröße und verändern Sie den gewählten Pol p, bis sich für einen Führungssprung auf 0.3 m sinnvolle Stellgrößen ergeben.

d) Realisieren Sie die PI-Zustandsregelung für das reale System mit den in Aufgabenteil c) gefundenen Reglerparametern.

4 IP02-Wagen mit Pendel

In diesem Kapitel gilt es das im Folgenden kurz als *SPG* (*Single Pendulum Gantry*) bezeichnete System mathematisch zu beschreiben und zu regeln. Es handelt sich um den bereits bekannten IP02-Wagen, an dessen Pendelachse Pendel verschiedener Längen angebracht werden können. Bild 4.1 zeigt das zu regelnde System aus zwei verschiedenen Betrachtungswinkeln. Die Position des Wagens soll geregelt werden, sodass sich das Pendel beim Erreichen der Wunschposition in Ruhe befindet. Es wird für dieses Kapitel davon ausgegangen, dass eine Messung der Wagengeschwindigkeit sowie der Pendelwinkelgeschwindigkeit nicht möglich ist. Aus diesem Grund wiederholen wir Grundlagen zum Zustandsbeobachter und verwenden beobachtete Größen zur Zustandsregelung des Systems.

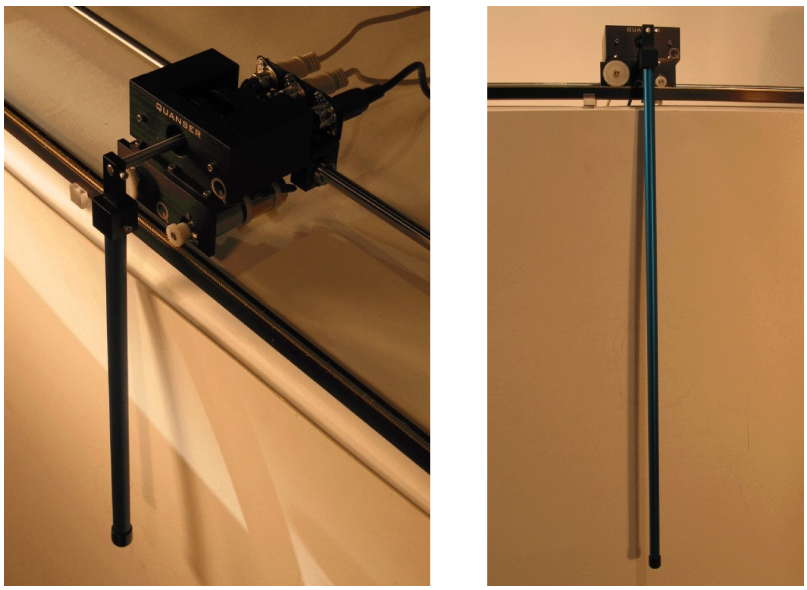


Bild 4.1: IP02-Wagen mit montiertem Pendel.

4.1 Bewegungsgleichungen der Regelstrecke

Die Bewegungsgleichungen der Regelstrecke sollen an dieser Stelle nicht explizit hergeleitet werden und lauten:

$$\begin{aligned}\ddot{x}_c = & \frac{1}{(M_c+M_p)I_p + M_cM_pl_p^2 + M_p^2l_p^2\sin^2\alpha} \cdot (\ \ldots \\ & -(I_p+M_pl_p^2)B_{eq}\dot{x}_c + (M_p^2l_p^3 + I_pM_pl_p)\dot{\alpha}^2\sin\alpha + M_pl_pB_p\dot{\alpha}\cos\alpha + \ \ldots \\ & +(I_p+M_pl_p^2)F_c + M_p^2l_p^2g\cos\alpha\sin\alpha)\end{aligned} \tag{4.1}$$

$$\begin{aligned}\ddot{\alpha} = & \frac{1}{(M_c+M_p)I_p + M_cM_pl_p^2 + M_p^2l_p^2\sin^2\alpha} \cdot (\ \ldots \\ & -(M_c+M_p)M_pgl_p\sin\alpha - (M_c+M_p)B_p\dot{\alpha} - M_p^2l_p^2\dot{\alpha}^2\sin\alpha\cos\alpha + \ \ldots \\ & M_pl_pB_{eq}\dot{x}_c\cos\alpha - F_cM_pl_p\cos\alpha)\end{aligned} \tag{4.2}$$

Das System soll in der Zustandsraumdarstellung beschrieben werden, in der per Definition lediglich lineare Gleichungen vorkommen dürfen. Aus diesem Grund muss eine Linearisierung um den Arbeitspunkt $\alpha_0 = 0$ erfolgen. Die Bedeutung der einzelnen Formelzeichen und deren numerische Werte können Anhang B entnommen werden.

Aufgabe 12

a) Linearisieren Sie die Bewegungsgleichungen 4.1 und 4.2 um den Pendelwinkel $\alpha_0 = 0$.

b) Substituieren Sie die Antriebskraft F_c mit folgender Gleichung, um als Stellgröße die Motorspannung U_M einsetzen zu können:

$$F_c = -\frac{\eta_g K_g^2 \eta_m K_t K_m}{R_m r_{mp}^2}\dot{x}_c + \frac{\eta_g K_g \eta_m K_t}{R_m r_{mp}}U_M \ . \tag{4.3}$$

c) Erstellen Sie ein MATLAB-Skript, in dem Sie die Systemmatrix $\boldsymbol{A}$, den Eingangsvektor $\boldsymbol{b}$, den Ausgangsvektor $\boldsymbol{c}^T$ und den Durchgriff $\boldsymbol{d}$ für das linearisierte Gleichungssystem definieren.

d) Erstellen Sie ein SIMULINK-Modell zur Simulation der Regelstrecke.

4.2 Praktische Ansteuerung und Messwertaufnahme

Zur Ansteuerung und Messwertaufnahme sind bereits in Abschnitt 2.2 alle nötigen Grundlagen genauer erläutert worden. Die dort beschriebene Konfiguration ist dieselbe, wie sie für das SPG vorgesehen ist - d.h. es müssen keinerlei Änderungen im Vergleich zu Abschnitt 2.2 vorgenommen werden.

4.3 Positionsregelung des IP02-Wagens mit Pendel

Wie bereits zu Anfang des Kapitels erwähnt, gehen wir davon aus die Wagengeschwindigkeit und die Pendelwinkelgeschwindigkeit nicht messen zu können. Aus diesem Grund kommt ein Zustandsbeobachter zum Einsatz, der mithilfe der gemessenen Wagenposition x_c und des gemessenen Pendelwinkels α die fehlenden Zustände schätzt. Aus diesem Grund wiederholen die folgenden Abschnitte Grundlagen zum Thema Zustandsbeobachter.
Bei der Realisierung von Zustandsreglern tritt häufig das Problem auf, dass nicht für alle Zustände entsprechende Messungen zur Verfügung stehen. Aus Kostengründen oder widrigen Umgebungsbedingungen sind diese benötigten Zustände in Form von Messungen nicht verfügbar. Ein Beobachter soll mithilfe der vorhandenen Informationen über das System die nicht gemessenen Zustände schätzen. Diese geschätzten Zustände fließen anstatt der tatsächlichen Messungen in die Rückkopplung ein. Insofern spricht man auch von einem *virtuellen Sensor*. Dazu ist ein gutes Modell der Regelstrecke zwingend erforderlich.
Wir wollen den Luenberger-Beobachter verwenden, dessen Blockschaltbild für ein System mit einem Eingang und einem Ausgang in Bild 4.2 zu sehen ist. Wie aus diesem Schaubild zu erkennen ist, besteht der Beobachter aus dem Zustandsraummodell der Regelstrecke, die Parallel zum Prozess geschaltet ist. Ein Beobachter sollte im Allgemeinen immer so aufgebaut sein, dass er dem Ausgangsfehler entgegen wirkt. Er nutzt somit - analog zu einem Regelkreis - das Prinzip der Rückkopplung. Die sich ergebende Zustandsgleichungen für den Luenberger-Beobachter lassen sich einfach aus Bild 4.2 ablesen und lauten:

$$\dot{\hat{\boldsymbol{x}}}(t) = \boldsymbol{A}\hat{\boldsymbol{x}}(t) + \boldsymbol{b}u(t) + \boldsymbol{l}e_y(t) \tag{4.4}$$

$$y(t) = \boldsymbol{c}^T\hat{\boldsymbol{x}}(t)\ . \tag{4.5}$$

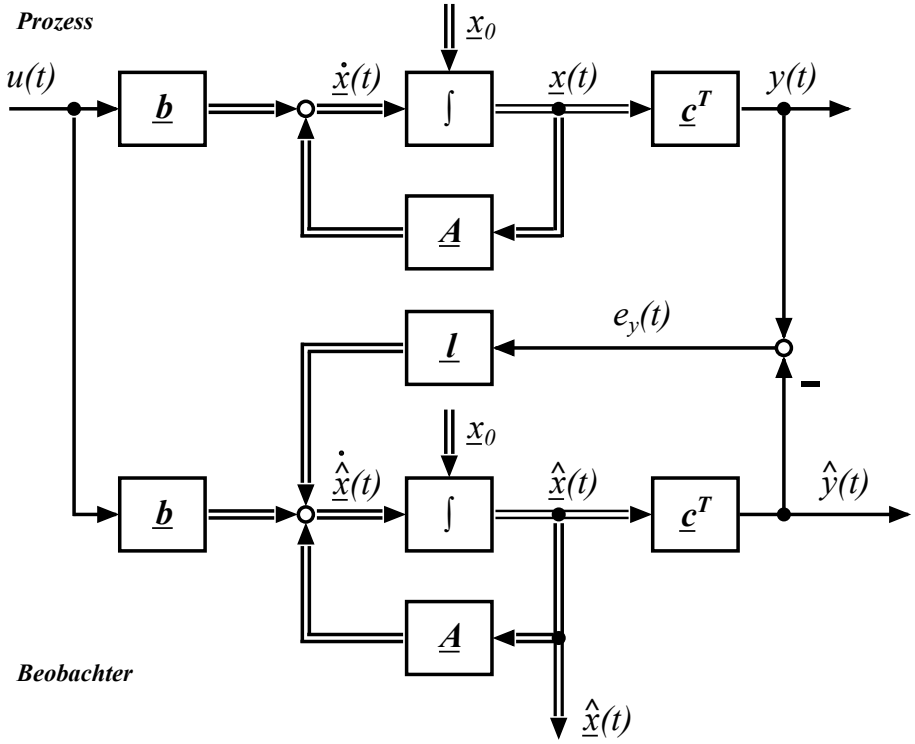


Bild 4.2: Blockschaltbild eines Luenberger-Beobachters mit einem Ein- und Ausgang.

Die ebenfalls aus Bild 4.2 zu erkennenden Beziehungen

$$e_y(t) = y(t) - \hat{y}(t) \ , \tag{4.6}$$

$$y(t) = \boldsymbol{c}^T \boldsymbol{x}(t) \text{ und} \tag{4.7}$$

$$\hat{y}(t) = \boldsymbol{c}^T \hat{\boldsymbol{x}}(t) \tag{4.8}$$

führen zusammen mit Gl. 4.4 auf die folgende Darstellung

$$\dot{\hat{\boldsymbol{x}}}(t) = \boldsymbol{A}\hat{\boldsymbol{x}}(t) + \boldsymbol{b}u(t) + \boldsymbol{l}\boldsymbol{c}^T \left(\boldsymbol{x}(t) - \hat{\boldsymbol{x}}(t)\right) \ , \tag{4.9}$$

aus der sich gut erkennen lässt, dass die Rückkopplung (d.h. der Rückkopplungsvektor $\boldsymbol{l}$) nur bei einer Abweichung zwischen geschätzten und realen Zuständen aktiv ist. In Gl. 4.9 lassen sich alle von den geschätzten Zuständen $\hat{\boldsymbol{x}}$ abhängigen Größen zusammenfassen, sodass sich schließlich die Beobachtergleichung zu

$$\dot{\hat{\boldsymbol{x}}}(t) = \left(\boldsymbol{A} - \boldsymbol{l}\boldsymbol{c}^T\right) \hat{\boldsymbol{x}}(t) + \boldsymbol{b}u(t) + \boldsymbol{l} \underbrace{\boldsymbol{c}^T \boldsymbol{x}(t)}_{=y(t)} \tag{4.10}$$

ergibt. Der Rückkopplungsvektor $\boldsymbol{l}$ ändert die Eigendynamik des Beobachters in einer ähnlichen Weise wie der Zustandsregler $\boldsymbol{k}^T$ die Eigendynamik der Regelstrecke

verändert. Der Entwurf des Beobachters kann daher sehr ähnlich zum Entwurf des Zustandsreglers durch Polvorgabe erfolgen. Man nennt den Beobachterentwurf aus diesem Grund das duale Problem zum Reglerentwurf. Für den Entwurf eines Beobachters muss der Rückkopplungsvektor $\boldsymbol{l}$ so bestimmt werden, dass die Systemmatrix des Beobachters $(\boldsymbol{A} - \boldsymbol{l}\boldsymbol{c}^T)$ die gewünschte Dynamik aufweist. Da eine allgemeine Matrix $\boldsymbol{M}$ und ihre Transponierte $\boldsymbol{M}^T$ die selben Eigenwerte aufweisen, können wir alternativ auch die Dynamik für folgende Matrix bestimmen:

$$\left(\boldsymbol{A} - \boldsymbol{l}\boldsymbol{c}^T\right)^T = \boldsymbol{A}^T - \boldsymbol{c}\boldsymbol{l}^T \ . \tag{4.11}$$

Vom Entwurf eines Zustandsreglers wissen wir bereits, wie man den Reglervektor $\boldsymbol{k}^T$ berechnen kann, um die Dynamik der Matrix $\boldsymbol{A} - \boldsymbol{b}\boldsymbol{k}^T$ zu bestimmen. Diese Verfahren können wir nun analog für den Beobachterentwurf verwenden, es muss lediglich $\boldsymbol{b}$ durch $\boldsymbol{c}$ und $\boldsymbol{A}$ durch $\boldsymbol{A}^T$ ersetzt werden, um $\boldsymbol{l}^T$ statt $\boldsymbol{k}^T$ zu erhalten. Ziel ist es dabei Abweichungen zwischen den geschätzten und realen Zuständen möglichst schnell abklingen zu lassen. Dass dies durch eine geeignete Wahl des Rückkopplungsvektors $\boldsymbol{l}$ möglich ist, zeigt sich bei Betrachtung der Gleichung zur Änderungsrate des Beobachterfehlers. Dazu wird zunächst eine Beobachterfehler wie folgt definiert:

$$\boldsymbol{e}_x(t) = \boldsymbol{x}(t) - \hat{\boldsymbol{x}}(t) \ . \tag{4.12}$$

Die Änderungsrate erhält man nun durch Ableitung von Gl. 4.12:

$$\dot{\boldsymbol{e}}_x(t) = \dot{\boldsymbol{x}}(t) - \dot{\hat{\boldsymbol{x}}}(t) \ . \tag{4.13}$$

Für die zeitlich abgeleiteten Zustandsvektoren der geschätzten und realen Zustände, setzen wir nun Gl. 3.2 und 4.9 ein und erhalten:

$$\begin{aligned} \dot{e}_x(t) &= \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{b}u(t) - \left[\boldsymbol{A}\hat{\boldsymbol{x}}(t) + \boldsymbol{b}u(t) + \boldsymbol{l}\boldsymbol{c}^T\left(\boldsymbol{x}(t) - \hat{\boldsymbol{x}}(t)\right)\right] && (4.14) \\ &= \left(\boldsymbol{A} - \boldsymbol{l}\boldsymbol{c}^T\right) \underbrace{\left(\boldsymbol{x}(t) - \hat{\boldsymbol{x}}(t)\right)}_{\boldsymbol{e}_x(t)} \ . && (4.15) \end{aligned}$$

Ist die Matrix $\boldsymbol{A} - \boldsymbol{l}\boldsymbol{c}^T$ stabil, klingt jede Anfangsabweichung exponentiell gegen Null ab. Zu erkennen ist zudem, dass es sich bei der Beschreibung des Beobachterfehlers um ein *nicht steuerbares* System handelt. Die Verläufe der Signale $u(t)$ und $y(t)$ haben keinerlei Einfluss auf den Fehlerverlauf. Das bedeutet, so lange wir stabile Pole für den Beobachter vorgeben, dass die geschätzten Zustände des Systems den realen Zuständen entgegen streben. Da es sich bei dem Beobachter um eine rei-

ne Simulation handelt, muss bei der Vorgabe der Eigenwerte nicht auf eventuelle Stellgrößenbeschränkungen oder die Schonung des Stellgliedes geachtet und die Pole könnten im Prinzip beliebig schnell gewählt werden. Allerdings sind hier wegen des stets auftretenden Messrauschens Grenzen gesetzt. Grundsätzlich gilt aber, dass die Pole für den Beobachter deutlich schneller sein sollten, als die dominierenden Pole des beobachteten Systems. Mit einer solchen Wahl ist sichergestellt, dass der Beobachterfehler deutlich schneller abklingt als die Eigendynamik des Systems.

Aufgabe 13

a) Entwerfen Sie für das SPG-System einen Beobachter. Achten Sie darauf, dass die gewählten Beobachterpole bezüglich der Pole des Prozesses sinnvoll gewählt sind.

b) Erstellen Sie ein SIMULINK-Modell, in dem Sie den Beobachter zusammen mit dem Prozess-Modell implementieren. Wählen Sie die Anfangszustände für das Prozess-Modell verschieden von denen des Beobachters und betrachten Sie in der Simulation die Verläufe der Zustandsgrößen des Prozess-Modells und der geschätzten Beobachter Zustände. Verdoppeln und halbieren Sie die gewählten Eigenwerte der charakteristischen Gleichung des Beobachters und betrachten Sie die resultierenden Zeitverläufe.

Mithilfe eines Beobachters lassen sich somit Zustände schätzen, die dann für eine Zustandsregelung verwendet werden können, wie dies in Bild 4.3 dargestellt ist. Es lässt sich zeigen (siehe Separationstheorem in [2] und/oder [1]), dass man den Beobachter und die Zustandsregelung unabhängig voneinander entwerfen kann. D.h. der Beobachter kann so entworfen werden, als ob keine Regelung vorhanden wäre und der Regler kann so entworfen werden, als ob es keinen Beobachter gäbe. Die Güte der sich ergebenden Regelung hängt natürlich von der Größe des Beobachterfehlers ab. Bei der Wahl der Lage der Pole sollten die Eigenwerte des Beobachters deutlich schneller sein als die dominierenden Pole des **geregelten Systems**. Je schneller man den Beobachter wählt, umso weniger wird die Regelgüte durch den Beobachter beeinträchtigt. Obwohl es sich bei dem Beobachter, wie bereits erwähnt, um ein rein simuliertes System handelt, besteht ein Zielkonflikt, der es verbietet die Beobachterpole zu schnell zu wählen. Die Rauschempfindlichkeit des Beobachters nimmt mit schnelleren Polen zu, sodass ein Kompromiss zwischen der Rauschempfindlichkeit und der Beobachterdynamik getroffen werden muss.

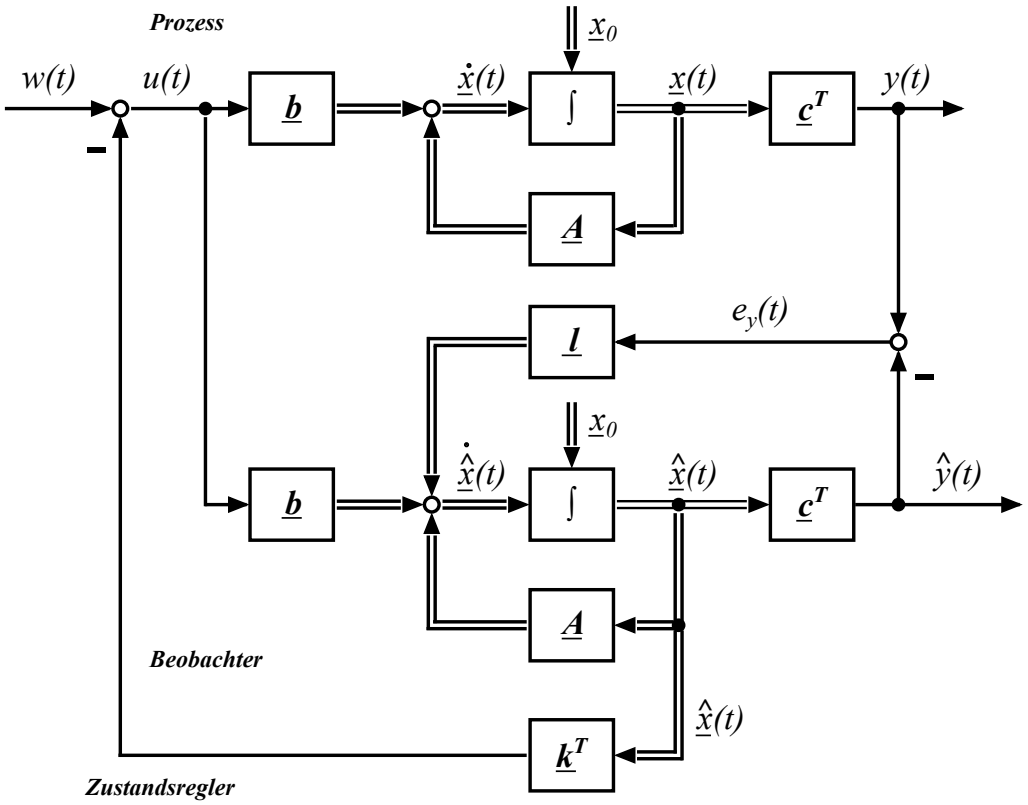


Bild 4.3: Blockschaltbild eines Luenberger-Beobachters zusammen mit einem Zustandsregler.

Aufgabe 14

a) Legen Sie einen einfachen Zustandsregler (kein I-Anteil) für das SPG-System durch Polvorgabe aus. Passen Sie die gewählten Beobachterpole gegebenenfalls an.

b) Erstellen Sie ein SIMULINK-Modell zur Simulation des geschlossenen Regelkreises mit Beobachter. Überprüfen Sie die sich ergebenden Stellgrößen für das System in der Simulation. Passen Sie gegebenenfalls die Pole des geschlossenen Regelkreises so an, dass Stellgrößen von $\max(|u|) > 5$ Volt vermieden werden.

c) Regeln Sie das reale System mit den gefundenen Regler- und Beobachterpolen.

Aufgrund der bereits bekannten Problematik der Reibung und dem damit einhergehenden fehlenden I-Verhalten für kleine Stellgrößen, besitzt das real geregelte System

eine bleibende Regelabweichung. Eine Möglichkeit die bleibende Regelabweichung zu eliminieren ist bereits zuvor vorgestellt worden, der PI-Zustandsregler. Hier soll nun eine weitere Möglichkeit aufgezeigt werden, die bleibende Regelabweichung zu beseitigen. Es handelt sich um eine Erweiterung der Beobachtergleichungen und führt zu einem sogenannten *Störbeobachter*. Man geht zunächst davon aus, dass eine Störung $z(t)$ des Stellsignals $u(t)$ vorliegt. Dies führt zunächst zu folgender Zustandsgleichung des Beobachters:

$$\dot{\hat{\boldsymbol{x}}}(t) = \boldsymbol{A}\hat{\boldsymbol{x}}(t) + \boldsymbol{b}\left[u(t) + z(t)\right] + \boldsymbol{l}e_y(t) \ . \tag{4.16}$$

Wenn man nun eine Schätzung der Störung $\hat{z}(t)$ kennt, kann man die Stellgröße derart anpassen, dass die Störung kompensiert wird:

$$u(t) = w(t) - \left(\boldsymbol{k}^T \hat{x}(t) \quad \underbrace{+\hat{z}(t)}_{\text{Stellgrößenanpassung}} \right) \ . \tag{4.17}$$

Die Schätzung der Störung $\hat{z}(t)$ nimmt man nun in den Vektor der übrigen geschätzten Zustände $\hat{x}(t)$ auf und trifft eine Annahme über die Eigenschaften der Störung, um diese im Zustandsraum beschreiben zu können. In unserem Fall gehen wir von einer konstanten Störung aus. Dies lässt sich als konstante Reibkraft interpretieren, die durch ein Plus an Stellgröße überwunden werden kann. In Zustandsraumdarstellung folgt die Störung somit der folgenden Gleichung:

$$\dot{\hat{z}}(t) = 0 \ . \tag{4.18}$$

Es ergibt sich folgende Zustandsraumdarstellung für den Störbeobachter mit den angenommenen Eigenschaften der Störung $z(t)$:

$$\underbrace{\begin{bmatrix} \dot{\hat{\boldsymbol{x}}}(t) \\ \dot{\hat{\boldsymbol{z}}} \end{bmatrix}}_{\dot{\hat{\boldsymbol{x}}}_z(t)} = \underbrace{\begin{bmatrix} \boldsymbol{A} & \boldsymbol{b} \\ \boldsymbol{0} & 0 \end{bmatrix}}_{\boldsymbol{A}_z} \underbrace{\begin{bmatrix} \hat{\boldsymbol{x}}(t) \\ \hat{z}(t) \end{bmatrix}}_{\hat{\boldsymbol{x}}_z} + \underbrace{\begin{bmatrix} \boldsymbol{b} \\ 0 \end{bmatrix}}_{\boldsymbol{b}_z} u(t) + \boldsymbol{l}_z e_y(t) \ , \tag{4.19}$$

$$y(t) = \underbrace{\begin{bmatrix} \boldsymbol{c}^T & 0 \end{bmatrix}}_{\boldsymbol{c}_z^T} \underbrace{\begin{bmatrix} \hat{\boldsymbol{x}}(t) \\ \hat{z}(t) \end{bmatrix}}_{\hat{\boldsymbol{x}}_z} \ . \tag{4.20}$$

Nähere Informationen zum Thema Störbeobachter sind in [1] und [2] zu finden, insbesondere für anders geartete Störungen. Mithilfe des so angepassten Beobachters, d.h. dem Störbeobachter, lässt sich nun die Störung $z(t)$ beobachten und deren Schätzung

$\hat{z}(t)$ zur Anpassung der Stellgröße verwenden. Bild 4.4 zeigt das Blockschaltbild einer Regelung mit Störbeobachter. An dem Entwurf des Zustandsreglers $\boldsymbol{k}^T$ ändert sich

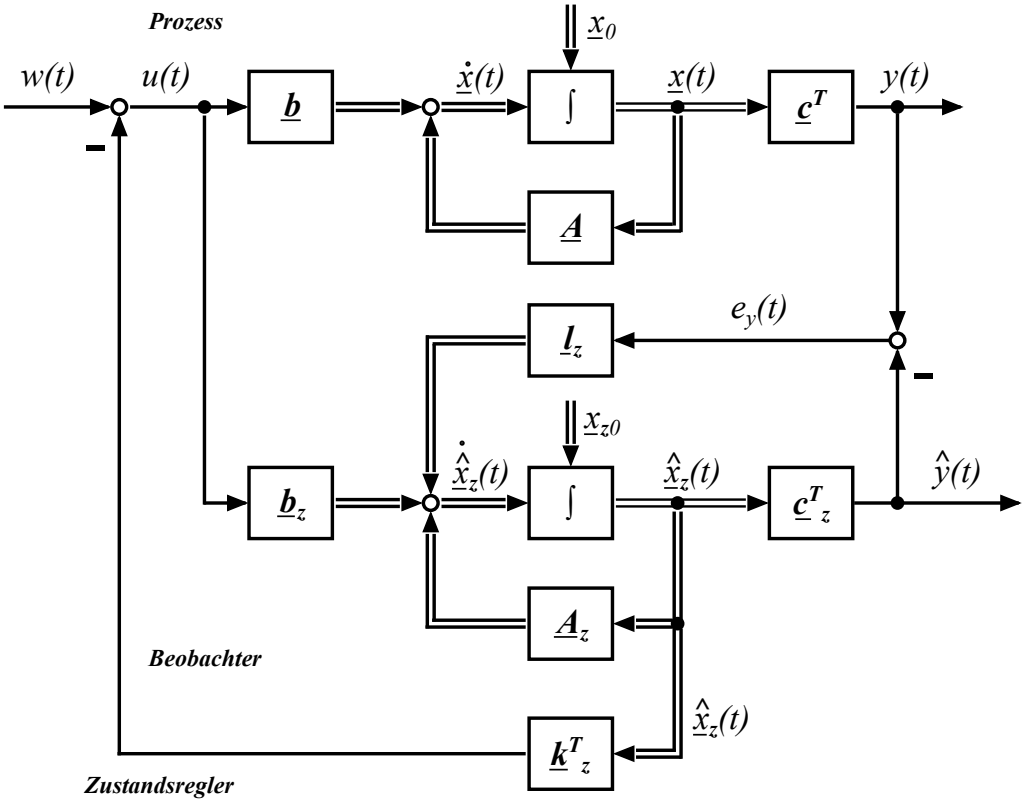


Bild 4.4: Blockschaltbild eines Störbeobachters zusammen mit einem Zustandsregler und der Stellgrößenanpassung.

prinzipiell nichts. Da der Störbeobachter jedoch nun fünf statt bisher vier Zustände zurückführt, muss eine Anpassung erfolgen, die Gl. 4.17 gerecht wird. D.h. der Rückführungsvektor erweitert sich zu $\boldsymbol{k}_z^T = \begin{bmatrix} \boldsymbol{k}^T & 1 \end{bmatrix}$.

Aufgabe 15

a) Definieren Sie in einem MATLAB-Skript die sich neu ergebenden Matrizen und Vektoren zur Beobachtung der als konstant angenommenen Störung $z(t)$.

b) Setzten Sie den Störbeobachter zunächst in einer Simulation ein, in der Sie einen sogenannten *Dead Zone Dynamic*-Block benutzen, um die Haftreibung des Systems näherungsweise mit zu modellieren. Schafft es der Störbeobachter die vom Benutzer gewählten Werte des *Dead Zone Dynamic*-Blocks exakt zu schätzen?

c) Regeln Sie das reale SPG-System mit dem Störbeobachter und betrachten Sie erneut die sich ergebenden Schätzungen für die Störung $z(t)$.

5 IP02-Wagen mit invertiertem Pendel

Gegenstand des folgenden Kapitels ist das bereits aus Kapitel 4 bekannte System, bei dem Pendel unterschiedlicher Länge am IP02-Wagen montiert sind. Die Aufgabe besteht darin, das Pendel durch eine Regelung in der aufrechten Position zu stabilisieren, während eine bestimmte Position entweder gehalten oder angefahren werden soll. In Bild 5.1 ist der im Folgenden als SIP-System (*Single Inverted Pendulum*) bezeichnete Aufbau zu sehen. Im linken Bild ist das kurze, im rechten Bild das lange Pendel am IP02-Wagen angebracht.

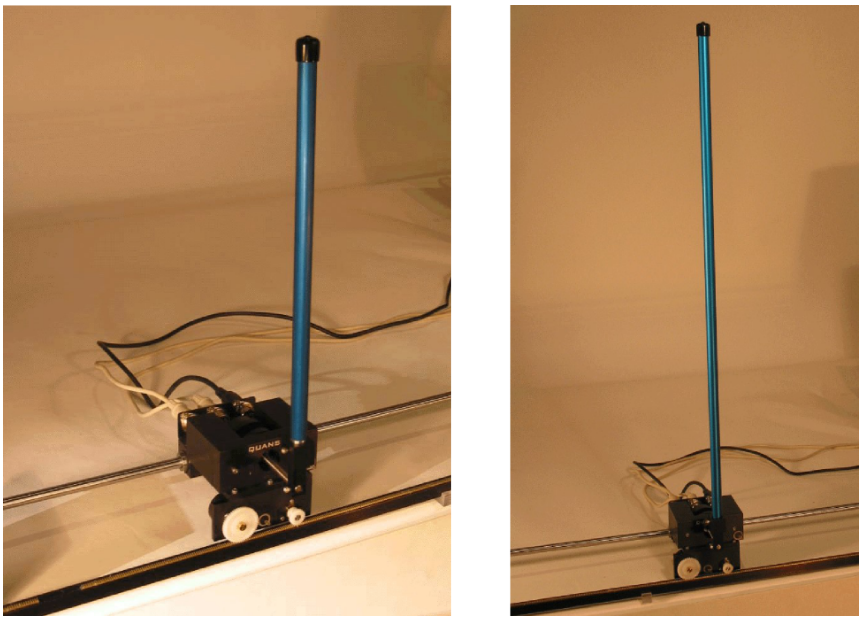


Bild 5.1: Kurzes und langes Pendel in inverser Position.

5.1 Bewegungsgleichungen der Regelstrecke

Die Bewegungsgleichungen, die das SIP-System beschreiben sind die gleichen, die bereits aus Abschnitt 5.1 bekannt sind. Der Unterschied im Vergleich zum vorherigen Kapitel ist lediglich der andere Arbeitspunkt, um den die nichtlinearen Bewegungsgleichungen linearisiert werden.

Aufgabe 16

a) Linearisieren Sie die Bewegungsgleichungen 4.1 und 4.2 um den Pendelwinkel $\alpha_0 = \pi$.

b) Substituieren Sie die Antriebskraft F_c mit folgender Gleichung, um als Stellgröße die Motorspannung U_M einsetzen zu können:

$$F_c = -\frac{\eta_g K_g^2 \eta_m K_t K_m}{R_m r_{mp}^2} \dot{x}_c + \frac{\eta_g K_g \eta_m K_t}{R_m r_{mp}} U_M \ . \tag{5.1}$$

c) Erstellen Sie ein MATLAB-Skript, in dem Sie die Systemmatrix $\boldsymbol{A}$, den Eingangsvektor $\boldsymbol{b}$, den Ausgangsvektor $\boldsymbol{c}^T$ und den Durchgriff $\boldsymbol{d}$ für das linearisierte Gleichungssystem definieren.

d) Erstellen Sie ein SIMULINK-Modell zur Simulation der Regelstrecke.

e) Benutzen Sie das zur Verfügung stehende SIMULINK-Modell der Regelstrecke, in dem die Differentialgleichungen in nichtlinearer Form implementiert sind, um zu evaluieren für welchen Winkelbereich das linearisierte Modell um $\alpha \approx \pi$ valide ist. Vergleichen Sie dazu das nichtlineare Modell mit dem linearisierten Modell aus Aufgabenteil d).

5.2 Praktische Ansteuerung und Messwertaufnahme

Zur Ansteuerung und Messwertaufnahme sind bereits in Abschnitt 2.2 alle nötigen Grundlagen genauer erläutert worden. Die dort beschriebene Konfiguration ist dieselbe, wie sie für das SIP-System vorgesehen ist - d.h. es müssen keinerlei Änderungen im Vergleich zu Abschnitt 2.2 vorgenommen werden.

Da es für die anstehende regelungstechnische Aufgabe von großer Wichtigkeit ist, dass der Pendelwinkel α sehr genau gemessen wird, soll diese Genauigkeit anhand eines Versuchs überprüft werden.

Aufgabe 17

Erstellen Sie ein SIMULINK-Modell, in dem Sie das SIP-System ansteuern können sowie die Sensoren zur Messung der Wagenposition und Winkelstellung auslesen können. Setzen Sie das Stellsignal auf 0 Volt fest. Lassen Sie sich die Zustände der Wagenposition und der Winkelstellung in jeweils einem Scope anzeigen und speichern Sie die Winkelstellungen α in einer Variablen namens *pendelwinkel* im MATLAB-Workspace ab. Starten Sie nun die Simulation und drehen Sie das Pendel händisch um genau eine Umdrehung. Achten Sie darauf, dass das Pendel seine Ruhelage erreicht hat, stoppen Sie die Simulation und überprüfen Sie anhand der *pendelwinkel*-Variablen, welchen Pendelwinkel der Sensor zuletzt ausgegeben hat. Definieren Sie gegebenenfalls einen Korrekturfaktor.

5.3 Positionsregelung des IP02-Wagens mit Pendel

Bevor die Regelung des invertierten Pendels stattfindet, müssen Vorkehrungen zur Schonung des Aufbaus getroffen werden. Um zu verhindern, dass das SIP-System mit hoher Geschwindigkeit gegen die physikalischen Anschläge fährt, gilt es Positionen und Winkelstellungen zu definieren, bei deren Erreichen ein softwaremäßiger Notaus zu betätigen ist. Beim Einschalten des Systems soll sich der Wagen stets in etwa mittig auf der Zahnstange befinden. Ausgehend von dieser Position soll es dem Wagen möglich sein, 30 cm sowohl nach links als auch nach rechts zu fahren. Daraus ergeben sich die ersten Bedingungen zur Betätigung des Notaus-Schalters ($x_c < -0.3$ m oder $x_c > 0.3$ m). Des weiteren soll der Notaus-Schalter eingreifen, sobald das Pendel den Winkelbereich verlässt, in dem das linearisierte Modell das tatsächliche Prozessverhalten ausreichend gut beschreibt. Dieser Winkelbereich sollte aus Aufgabe 16 e) bekannt sein. Zur Realisierung des softwareseitigen Notaus Schalters werden drei bisher nicht benutzte SIMULINK-Blöcke vorgestellt. Im SIMULINK Library Browser unter *Quarc Targets* → *Sinks* → *Error Handling* ist der Block *Stop with Message* zu finden, über den der Notaus aktiviert wird und ein entsprechender Warnhinweis ausgegeben wird. Der Block terminiert die Simulation, sobald am

Eingang des Blocks ein logisches Ja (*true*) anliegt. Nun fehlen vergleichende Blöcke, um Überschreitungen der Wagenposition oder des Pendelwinkels festzustellen. Diese sind unter *Simulink* → *Logic and Bit Operations* unter dem Namen *Compare to constant* zu finden. Als Ausgabe dieses Block-Typs erhält man eine Boolean Variable, die einem logischen Ja (*true*) oder Nein (*false*) entsprechen kann. Da eine Aktivierung der Winkelüberwachung erst nach dem Erreichen der aufrechten Position des Pendels gewünscht ist, sollten wir den sogenannten *Enable Subsystem*-Block verwenden, zu finden unter *Simulink* → *Ports & Subsystems*. Dieser lässt ein vorgegebenes Signal, welches am Eingang des Blocks anliegt, erst durch, nachdem ein logisches Ja am *Enable*-Port angelegen hat. Nach der Aktivierung schaltet sich das System nicht mehr aus. Zur Erläuterung ist dazu in Bild 5.2 ein SIMULINK-Modell mit dem entsprechendem Block dargestellt. Die Sinusschwingung und die Ausgabe des *Compare to constant*-Blocks sind in Bild 5.3 (a) zu sehen. Darunter ist die Ausgabe des *Enable Subsystem*-Blocks dargestellt. Man erkennt in Bild 5.3 (b), dass der *Enable Subsystem*-Block aktiv bleibt, obwohl das Signal am *Enable Port* wieder auf ein logisches Nein abfällt.

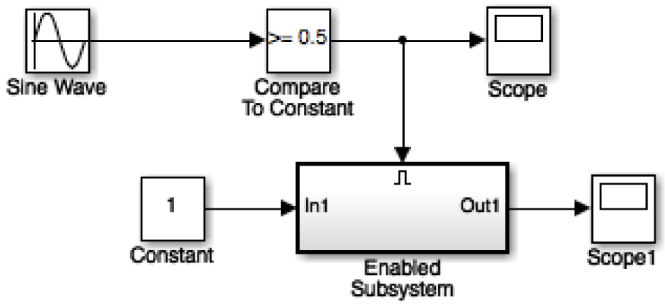
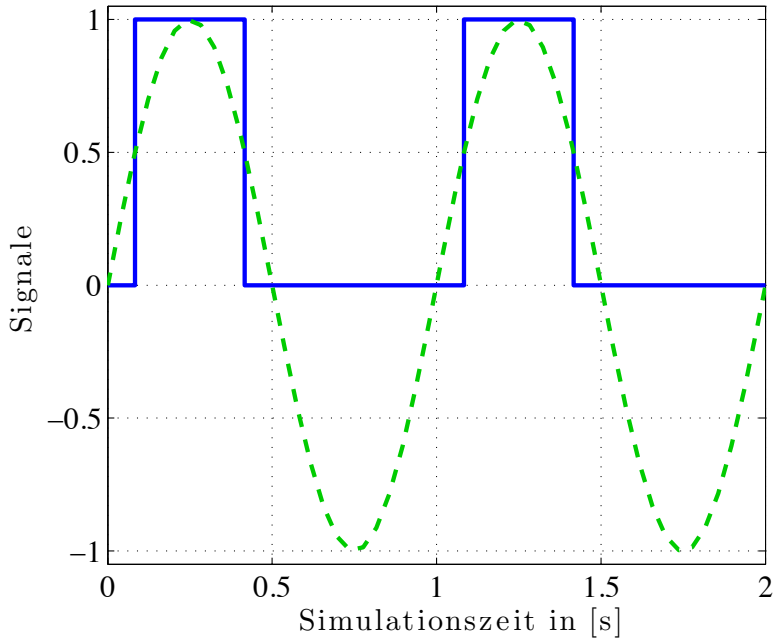


Bild 5.2: SIMULINK-Modell mit *Enable Subsystem*-Block.

Um zu gewährleisten, dass die Regelung erst nach dem Erreichen der aufrechten Pendelposition aktiv wird, soll der sogenannte *Switch*-Block zum Einsatz kommen, der unter *Simulink* → *Signal Routing* zu finden ist. Dieser soll lediglich ein Stellsignal verschieden von Null auf den Stellmotor geben, wenn der Pendelwinkel $\alpha = \pi$ einmalig erreicht ist. Mit den nun neu vorgestellten SIMULINK-Blöcken, sollten die sich anschließenden Aufgaben lösen lassen.

(a) Ausgabe des Eingangs- und Vergleichssignals.



(b) Ausgabe des *Enable Subsystem*-Blocks.

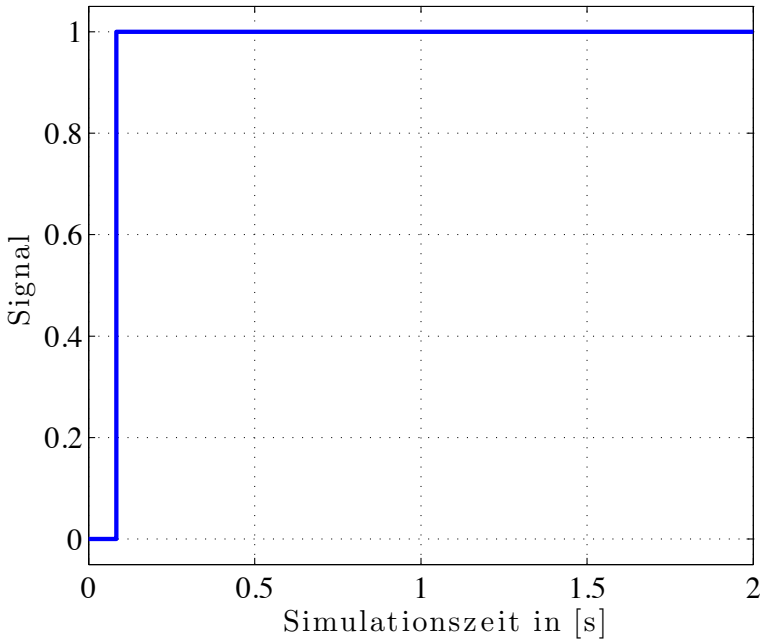


Bild 5.3: Ein- und Ausgabesignale des SIMULINK-Modells aus Bild 5.2.

Aufgabe 18

a) Erstellen Sie ein im Folgenden als *SIP-Interface* Block genanntes Subsystem, welches die gewünschten, zuvor beschriebenen Eigenschaften besitzt. Eingang des SIP-Interface Blocks ist die Motorspannung U_M, der Ausgang sollte alle Zustände des SIP-Systems umfassen. Stellen Sie sicher, dass anstatt des absoluten Pendelwinkels α entsprechend der Linearisierung um $\alpha_0 = \pi$, der Pendelwinkel $\tilde{\alpha} = \alpha - \pi$ als Zustand ausgegeben wird.

b) Überprüfen Sie alle geforderten Eigenschaften experimentell. Erst wenn alle geforderten Abschaltmechanismen wie zuvor beschrieben funktionieren, fahren Sie mit der nächsten Teilaufgabe fort.

c) Verwenden Sie zur Regelung des SIP-Systems einen gewöhnlichen Zustandsregler (ohne I-Anteil). Legen Sie diesen via Polvorgabe aus. Überprüfen Sie simulativ die sich ergebenden Stellgrößen unter Beachtung der maximal möglichen Ansteuerungsspannung.

d) Regeln Sie das reale SIP-System. Führen Sie nach dem Start des SIMULINK-Programms das Pendel behutsam gegen den Uhrzeigersinn in die aufrechte Position. Lassen Sie das Pendel erst los, wenn Sie merken, dass die Regelung aktiv ist.

A MATLAB Befehle

sys = tf(num,den)	Erzeugt ein kontinuierliches Übertragungsfunktion-Objekt *sys* mit Zähler *num* (vom englischen *numerator*) und Nenner *den* (vom englischen *denominator*). *num* und *den* sind liegende Vektoren mit absteigenden Potenzen der Laplace Variable s.
printsys(num,den,'s')	Ausgabe der kontinuierlichen Übertragungsfunktion im MATLAB Command Window.
impulse(sys)	Erzeugt einen Plot der Impulsantwort der kontinuierlichen Übertragungsfunktion *sys*.
step(sys)	Erzeugt einen Plot der Sprungantwort der kontinuierlichen Übertragungsfunktion *sys*.
pole(sys)	Berechnet die Pole einer kontinuierlichen Übertragungsfunktion *sys*.
rlocus(sys)	Ausgabe der Wurzelortskurve der kontinuierlichen Übertragungsfunktion *sys*.
bode(sys)	Ausgabe des Bode-Diagramms der kontinuierlichen Übertragungsfunktion *sys*.
sys = ss(A, b, c, d)	Erzeugt ein kontinuierliches Zustandsraummodell-Object *sys* mithilfe der Systemmatrix A, dem Eingangsvektor b, dem Ausgangsvektor c und dem Durchgriff d.

B Abkürzungen und Größen

B.1 Technische Größen

Symbol	Beschreibung	Wert	Einheit
V_{nom}	Nominelle Motoreingangsspannung	6,0	V
f_{max}	Maximale Eingangsspannungsfrequenz	50	Hz
I_{max}	Maximale Eingangsstromstärke	1,0	A
ω_{max}	Maximale Motorwinkelgeschw.	628,3	rad/s
R_M	Motorinnenwiderstand	2,6	Ω
L_m	Motorinduktivität	0,18	mH
K_t	Motormomentenkonstante	0,00767	Nm/A
η_M	Motorwirkungsgrad	100	%
K_m	Gegen-EMK Konstante	0,00767	V s / rad
J_m	Rotorträgheitsmoment	$3,9 \cdot 10^7$	kg m^2
K_g	Getriebeübersetzungsverhältnis	3,71	
η_g	Getriebewirkungsgrad	100	%
B_{eq}	Equivalente viskose Daempfung (am Motorzahnrad)	5,4	N s/m
M_{c2}	IP02 Masse	0,57	kg
M_w	IP02 Zusatzgewicht	0,37	kg
T_c	Verfahrweg	0,814	m
P_r	Zahnstangenteilung	$1,664 \cdot 10^3$	m/Zahn
r_{mp}	Wirkdurchmesser Motorzahnrad	$6,35 \cdot 10^3$	m
N_{mp}	Zähnezahl Antriebszahnrad	24	
r_{pp}	Wegaufnehmerzahnradradius	0,01482975	m
N_{pp}	Zähnezahl Wegaufnehmerzahnrad	56	
K_{EC}	Wegaufnehmerauflösung	$2,275 \cdot 10^5$	m/Inkrement
K_{EP}	Winkelaufnehmerauflösung	0,0015	rad/Inkrement

Technische Größen (Englisch)

Symbol	Description	Value	Unit
V_{nom}	Motor Nominal Input Voltage	6,0	V
f_{max}	Motor Input Voltage Maximum Frequency	50	Hz
I_{max}	Maximum input current	1,0	A
ω_{max}	Maximum motor speed	628,3	rad/s
R_M	Motor Armature Resistance	2,6	Ω
L_m	Motor Armature Inductance	0,18	mH
K_t	Motor Torque Constant	0,00767	Nm/A
η_M	Motor Efficiency	100	%
K_m	Back-ElectroMotive-Force (EMF) Constant	0,00767	V s / rad
J_m	Rotor Moment of Inertia	$3,9 \cdot 10^7$	kg m^2
K_g	Planetary Gearbox Gear Ratio	3,71	
η_g	Planetary Gearbox Efficiency	100	%
M_{c2}	IP02 Cart Mass	0,57	kg
M_w	IP02 Cart Weight Mass	0,37	kg
L_t	Track Length	0,990	m
T_c	Cart Travel	0,814	m
P_r	Rack Pitch	$1,664 \cdot 10^3$	m/tooth
r_{mp}	Motor Pinion Radius	$6,35 \cdot 10^3$	m
N_{mp}	Motor Pinion Number of Teeth	24	
r_{pp}	Position Pinion Radius	0,01482975	m
N_{pp}	Position Pinion Number of Teeth	56	
K_{EC}	IP02 Cart Encoder Resolution	$2,275 \cdot 10^5$	m/count
K_{EP}	IP02 Pendulum Encoder Resolution	0,0015	rad/count

B.2 Abkürzungen

Symbol	Beschreibung
x	Cart Linear Position
	Positions des Wagens
U_M	Motor input voltage
	Motoreingangsspannung
I_M	Motor input current

Symbol	Beschreibung
	Motoreingangsstromstärke
ω_M	Angular velocity of the motor shaft
	Winkelgeschwindigkeit der Motorwelle
F_θ	Force generated by the inertia of the e-machine
	Kraft resultierend aus dem Trägheitsmoment des E-Motors
F_z	Driving force by the e-machine
	Resultierende Antriebskraft des Motors
F_ω	Force depending on the angular velocity of the e-machine
	Widerstandskraft abhängig von der Winkelgeschw. der Motorwelle
m	Mass of the whole vehicle
	Masse des gesamten Vehikels
M_M	Generated torque of the e-machine
	Generiertes Motormoment

C Anhang

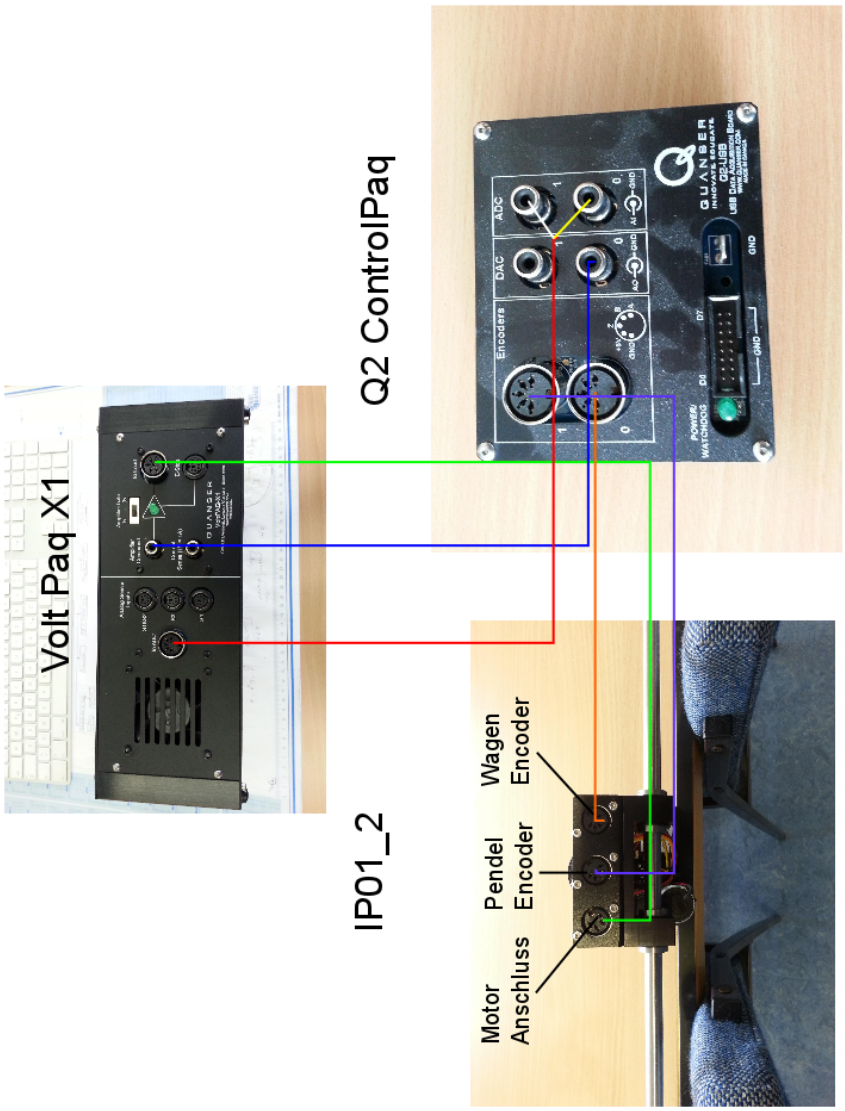


Bild C.1: Verkabelung des Wagens mit dem Leistungsverstärker sowie der Ansteuerungs- und Messeinheit.

Literaturverzeichnis

[1] Otto Föllinger. Regelungstechnik, Einführung in die Methoden und ihre Anwendungen. *Aufl. Hüthig, Heidelberg*, 1994.

[2] Graham Clifford Goodwin, Stefan F Graebe, and Mario E Salgado. *Control system design*, volume 240. Prentice Hall Upper Saddle River, 2001.

[3] O. Nelles. *Nonlinear System Identification.* Springer, Berlin, Germany, 2001.