Zur Modellbildung, Simulation und Regelung eines inversen doppelten Pendels

M. Bai

Forschungsbericht Nr. 7/94

Meß-, Steuer- und Regelungstechnik

Übersicht: In diesem Forschungsbericht wird über die Modellbildung, Simulation und Regelung starrer Roboter am Beispiel eines inversen doppelten Pendels berichtet, die in der objektorientierten Programmiersprache C++ unter Verwendung der Programmierumgebung „Mobile“ erfolgt. Hierbei wird ein kombiniertes Regelungskonzept in Form eines PD-Reglers mit Kompensator angewendet, wodurch Rückführung und Vorsteuerung verbunden werden. Die Stabilität des gesamten Systems wird anhand der direkten Methode von Ljapunow bewiesen.

Gerhard-Mercator-Universität - GH Duisburg
Meß-, Steuer- und Regelungstechnik
Prof. Dr.-Ing. H. Schwarz

Inhaltsverzeichnis

Nomenklatur

$\boldsymbol{A}$	Systemmatrix
$\boldsymbol{B}$	Eingangsmatrix
$\boldsymbol{b}(\boldsymbol{q}, \dot{\boldsymbol{q}})$	verallgemeinerte Kreisel- und Zentrifugalkräfte
$\boldsymbol{C}$	Ausgangsmatrix
$\boldsymbol{C}_1(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}}$	verallgemeinertes Kreisel- und Zentrifugalmoment
f	Anzahl der Freiheitsgrade
$\boldsymbol{g}(\boldsymbol{q})$	Gewichtsmoment
h	Schrittweite
$\boldsymbol{I}$	Einheitsmatrix
$\boldsymbol{K}_D$	Geschwindigkeitsrückführungsmatrix
$\boldsymbol{K}_P$	Positionsrückführungsmatrix
$l_{1,2}$	Länge
$m_{1,2}$	Masse
$\boldsymbol{M}(\boldsymbol{q})$	Massenmatrix
$\boldsymbol{q}$	verallgemeinerte Koordinaten oder Minimalkoordinaten
$\tilde{\boldsymbol{q}}$	Regelabweichungen
$\boldsymbol{Q}(\boldsymbol{q}, \dot{\boldsymbol{q}})$	verallgemeinerte eingeprägte Kräfte
$\mathbb{R}^f$	Menge der $f \times 1$-Vektoren
$\mathbb{R}^f \times \mathbb{R}^f$	Menge der $f \times f$-Matrizen
t	Zeit
$\boldsymbol{u}(t)$	Steuervektor
$\boldsymbol{w}(t)$	Sollvektor
$\boldsymbol{y}(t)$	Ausgang
$\boldsymbol{x}(t)$	Zustandsgröße
$\beta_{1,2}$	Winkel
$\mathbf{0}$	Nullmatrix

1 Einleitung

Mit dem zunehmenden Einsatz von Robotern wird eine immer höhere Funktionalität gefordert. Auf der einen Seite ist die Gewichtsreduktion der Roboter zu nennen, die z. B. für Aufgabenstellungen der Raumfahrt unerläßlich ist. Auf der anderen Seite sollen vielseitig einsetzbare Großroboter entwickelt werden, die sich durch große Reichweiten auszeichnen und deshalb aus einer Hintereinanderschaltung mehrerer Arme bestehen.

Um für solche Mehrkörpersysteme geeignete Regelungskonzepte zu entwerfen, muß zunächst ein geeignetes Modell gefunden werden. Hierzu können Newton-Euler- und Lagrange-Gleichungen aufgestellt und ausgewertet werden. Normalerweise ist es aber schwer, solche flexiblen Systeme mit mehreren Armen zu beschreiben, selbst wenn nur die Steifigkeit berücksichtigt wird. Ferner sind solche Systeme nichtlinear und zeitvariant, d. h. die Systemstruktur verändert sich mit der Zeit und den Systemzuständen. Zur Modellbildung werden die Körper, aus denen sich ein System zusammensetzen läßt, als Objekte programmiert. Die Programmumgebung „Mobile“ (Kecskeméthy 1993) erzeugt dann automatisch ein Modell des gesamten Systems. „Mobile“ ist in C++ geschrieben und erlaubt die objektorientierte Modellierung der Dynamik von Mehrkörpersystemen.

Zur Regelung dieser Systeme ist der Entwurf eines nichtlinearen Reglers notwendig. Hier wurde ein PD-Regler mit einer nichtlinearen Vorsteuerung angewendet. Die Regelstrecke ist ein inverses doppeltes Pendel. Um die Stabilität des gesamten Systems nachzuweisen, wird die Ljapunow Methode eingeführt. Der vorliegende Bericht dokumentiert die Simulationsergebnisse der Modellierung und Regelung des inversen doppelten Pendels. Für die Modellbildung wurde „Mobile“ verwendet. Die Berechnung des Zeitverlaufs erfolgte mit dem Runge-Kutta-Verfahren 4. Ordnung.

2 Beschreibung starrer Roboter

Ein starrer Roboter ist durch die Bewegungsgleichung

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{b}(\boldsymbol{q},\dot{\boldsymbol{q}}) + \boldsymbol{Q}(\boldsymbol{q},\dot{\boldsymbol{q}}) = \boldsymbol{u}(t) \tag{2.1}$$

zu beschreiben (in Anlehnung an Nijmeijer und van der Schaft 1990, Schneider 1993). Dabei ist $\boldsymbol{q} \in \mathbb{R}^f$ der Vektor der verallgemeinerten Koordinaten oder Minimalkoordinaten, $\boldsymbol{M} \in \mathbb{R}^f \times \mathbb{R}^f$ die symmetrische, positiv-definite Massenmatrix, $\boldsymbol{b} \in \mathbb{R}^f$ der Vektor der verallgemeinerten Kreisel- und Zentrifugalkräfte und $\boldsymbol{Q} \in \mathbb{R}^f$ der Vektor der verallgemeinerten eingeprägten Kräfte ohne den Steuervektor $\boldsymbol{u}(t) \in \mathbb{R}^f$. Hierbei bezeichnet f die Anzahl der Freiheitsgrade. Mit der Lagrange-Methode zu modellieren bedeutet, daß die kinetische und die potentielle Energiefunktion zu berechnen und mehrmals zu differenzieren ist. Bei der Newton-Euler-Methode müssen die Zwangskräfte berücksichtigt werden, was für einen mehrachsigen oder einen flexiblen Roboter sehr aufwendig ist. Deswegen ist es nötig, ein Programm zur automatischen Generierung der Bewegungsgleichungen zu entwickeln.

2.1 Zustandsmodell eines inversen doppelten Pendels

Als ein einfaches Beispiel starrer Mehrkörpersysteme wird ein inverses doppeltes Pendel betrachtet (Bild 2.1), dessen Position durch zwei Gelenkwinkel $\boldsymbol{q}^T = [\beta_1, \beta_2]$ beschrieben wird. $\boldsymbol{u}(t) \in \mathbb{R}^2$ bezeichnet den Vektor der auf die Gelenke ausgeübten Stellmomente. Die Dynamik des Pendels ist durch die Gleichung

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}_1(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{u}(t) \tag{2.2}$$

zu beschreiben (Slotine und Li 1991), wobei $\boldsymbol{C}_1(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} \in \mathbb{R}^2$ dem Kreisel- und Zentrifugalmoment und $\boldsymbol{g}(\boldsymbol{q}) \in \mathbb{R}^2$ dem Gewichtsmoment entspricht.

Der Zustandsvektor $\boldsymbol{x} \in \mathbb{R}^4$ des Systems wird aus den Minimalkoordinaten $\boldsymbol{q}$ und deren ersten zeitlichen Ableitungen $\dot{\boldsymbol{q}}$ gebildet:

$$\boldsymbol{x} = \begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \end{bmatrix} = \begin{bmatrix} \boldsymbol{q} \\ \dot{\boldsymbol{q}} \end{bmatrix}. \tag{2.3}$$

Durch Auflösen der Gleichung (2.2) nach $\ddot{\boldsymbol{q}}$ kann der Vektor $\dot{\boldsymbol{x}}$ gebildet werden:

$$\dot{\boldsymbol{x}} = \begin{bmatrix} \dot{\boldsymbol{x}}_1 \\ \dot{\boldsymbol{x}}_2 \end{bmatrix} = \begin{bmatrix} \dot{\boldsymbol{q}} \\ \ddot{\boldsymbol{q}} \end{bmatrix} = \begin{bmatrix} \dot{\boldsymbol{q}}^T \\ \boldsymbol{M}^{-1}\left(\boldsymbol{u} - \boldsymbol{C}_1\dot{\boldsymbol{q}} - \boldsymbol{g}\right) \end{bmatrix}. \tag{2.4}$$

Dann gilt für das Zustandsmodell

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{u} - \boldsymbol{B}\boldsymbol{g} \quad , \tag{2.5}$$

$$\boldsymbol{y} = \boldsymbol{C}^T\boldsymbol{x} \quad . \tag{2.6}$$

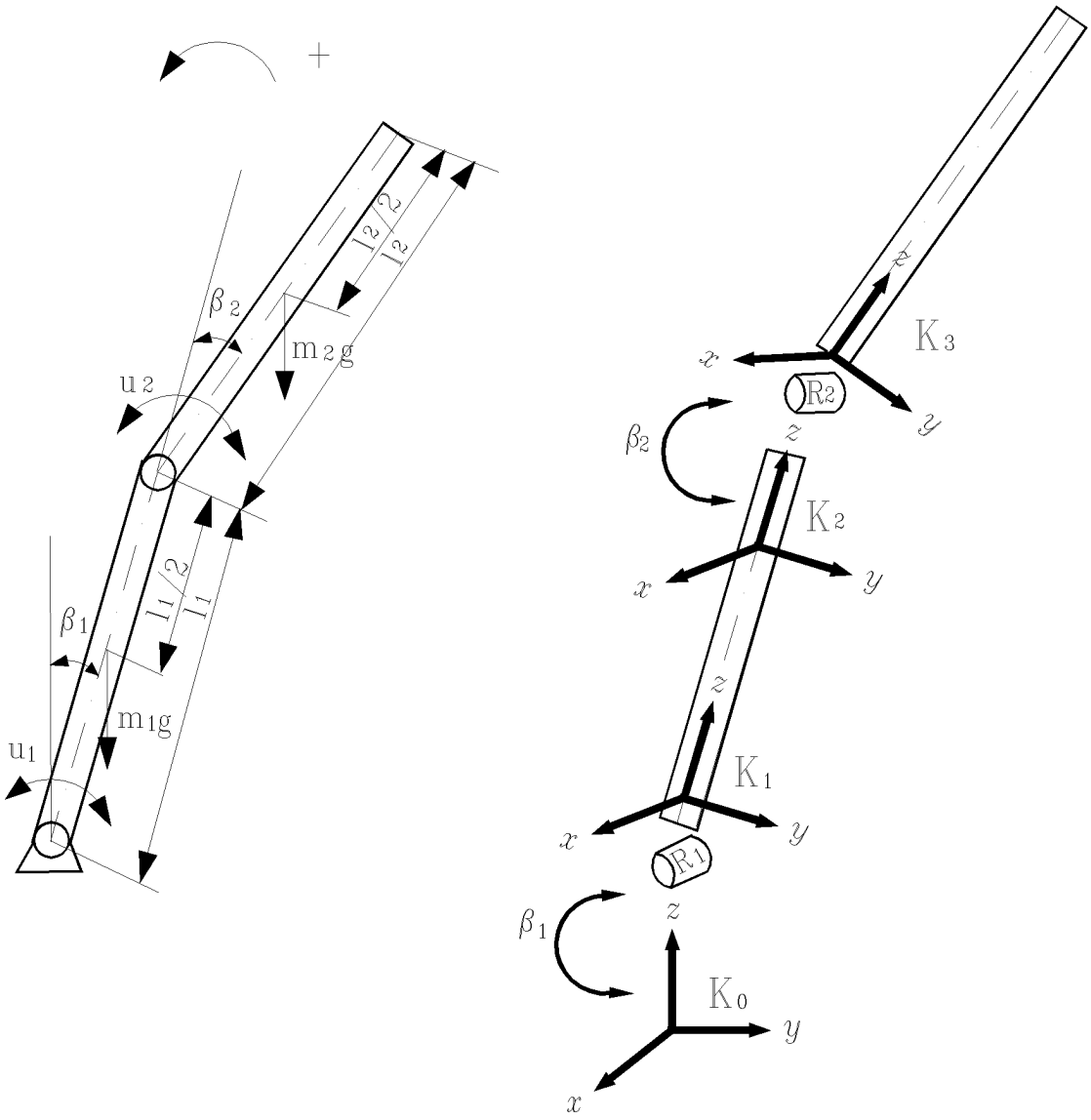

Bild 2.1: Ein inverses doppeltes Pendel

Die Matrizen des Modells sind erklärt zu

$$\boldsymbol{A} = \begin{bmatrix} \mathbf{0} & \boldsymbol{I} \\ \mathbf{0} & -\boldsymbol{M}^{-1}\boldsymbol{C}_1 \end{bmatrix}, \quad \boldsymbol{B} = \begin{bmatrix} \mathbf{0} \\ \boldsymbol{M}^{-1} \end{bmatrix} \text{ und } \boldsymbol{C}^T = \begin{bmatrix} \boldsymbol{I} & \mathbf{0} \end{bmatrix}. \tag{2.7}$$

Es ist offensichtlich, daß das inverse doppelte Pendel ein zeitvariantes und stark nichtlineares MIMO-System darstellt. Um $\boldsymbol{M}(\boldsymbol{q})$, $\boldsymbol{C}_1(\boldsymbol{q}, \dot{\boldsymbol{q}})$ und $\boldsymbol{g}(\boldsymbol{q})$ zu bestimmen, können auch die Euler-Lagrange-Gleichungen verwendet werden. Ein solches Vorgehen ist aber sehr aufwendig, wenn ein Roboter mehrere Freiheitsgrade hat. Deswegen wurde die Modellbildung unter der Programmierumgebung „Mobile“ durchgeführt.

2.2 Modellbildung und Simulation mit „Mobile“

Mit „Mobile“ kann eine Modellierung der Statik, Kinematik und Dynamik von Mehrkörpersystemen durchgeführt werden. Dazu müssen alle Glieder des Systems mit C++ definiert und die Verbindungen zwischen den Gliedern beschrieben werden. Obwohl „Mobile“ kein symbolisches Modell anbieten kann, läßt sich die Dynamik durch numerische Berechnung aller Arbeitspunkte bestimmen. Beispielweise können bei gegebenen $\boldsymbol{q}$, $\dot{\boldsymbol{q}}$, verallgemeinerten eingeprägten Kräften $\boldsymbol{Q}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ und Steuervektor $\boldsymbol{u}(t)$ die Massenmatrix $\boldsymbol{M}(\boldsymbol{q})$ und die verallgemeinerten Kreisel- und Zentrifugalkräfte $\boldsymbol{b}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ berechnet sowie nach $\ddot{\boldsymbol{q}}$ aufgelöst werden. Um die dynamischen Eigenschaften des Mehrkörpersystems zu erreichen, muß während jeder Schrittweite Modellierung, Auflösung und Integration von $\ddot{\boldsymbol{q}}$ durchgeführt werden. Das Flußdiagramm des Programmes ist in Bild 2.2 dargestellt. Als Beispiel wurde

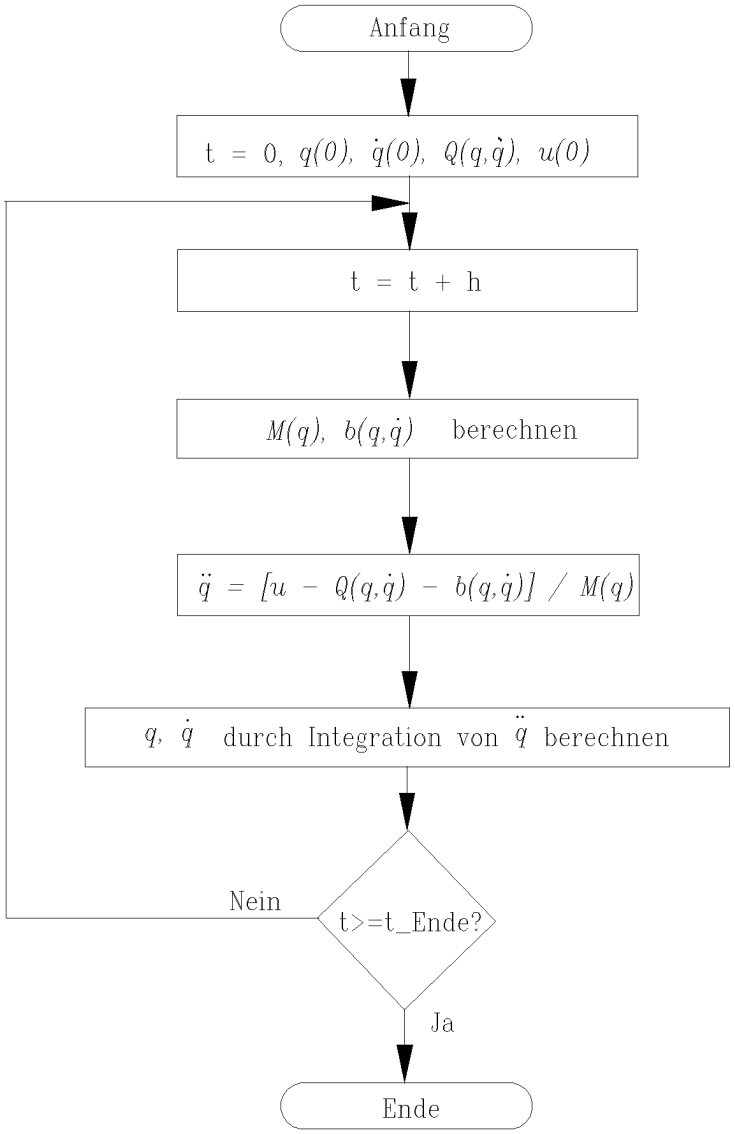

Bild 2.2: Flußdiagramm des Simulationsprogramms

das in Bild 2.1 dargestellte inverse doppelte Pendel betrachtet. Es besteht aus zwei Dreh-

gelenken R_1 und R_2, deren Achsen in x-Richtung zeigen. Die zwei Stangen des Pendels sind homogene steife Körper, deren Massenmittelpunkte auf ihren geometrischen Mittelpunkten liegen. Die Referenzsysteme sind hier willkürlich mit K_1, K_2 und K_3 bezeichnet. Die Gelenkvariablen sind die Winkel β_1 und β_2. Hier beträgt $l_1 = l_2 = 1\text{m}$, $m_1 = m_2 = 1\text{kg}$. Außer den Stellmomenten wurde nur die Gewichtskraft berücksichtigt. Der Quelltext des Programms steht im Anhang.

Simulationsergebnisse des inversen doppelten Pendels ohne Regler($\boldsymbol{u}(t) = 0$) sind in Bild 2.3 und Bild 2.4 dargestellt. Um die Eigendynamik des Systemes deutlich darzustellen, werden die Beträge der Winkel $\beta_1(t)$ und $\beta_2(t)$ durch 360° nomiert. Hier sind $\beta_1(0) = 10°$ und $\beta_2(0) = 20°$. Die Simulationsergebnisse zeigen, daß die beiden Winkel eine aperiodische Dynamik haben.

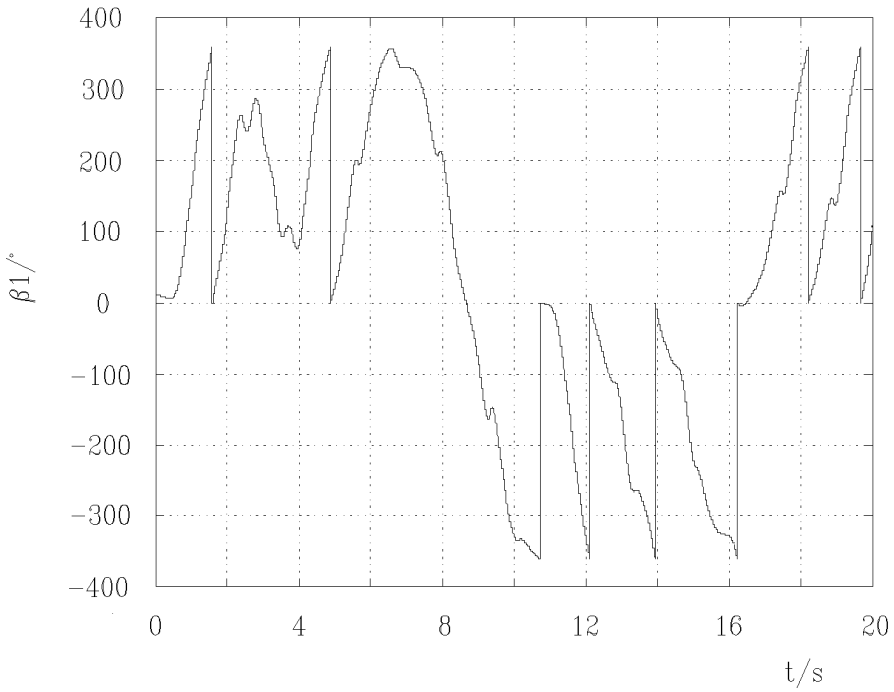

Bild 2.3: Zeitverlauf des Winkels β_1

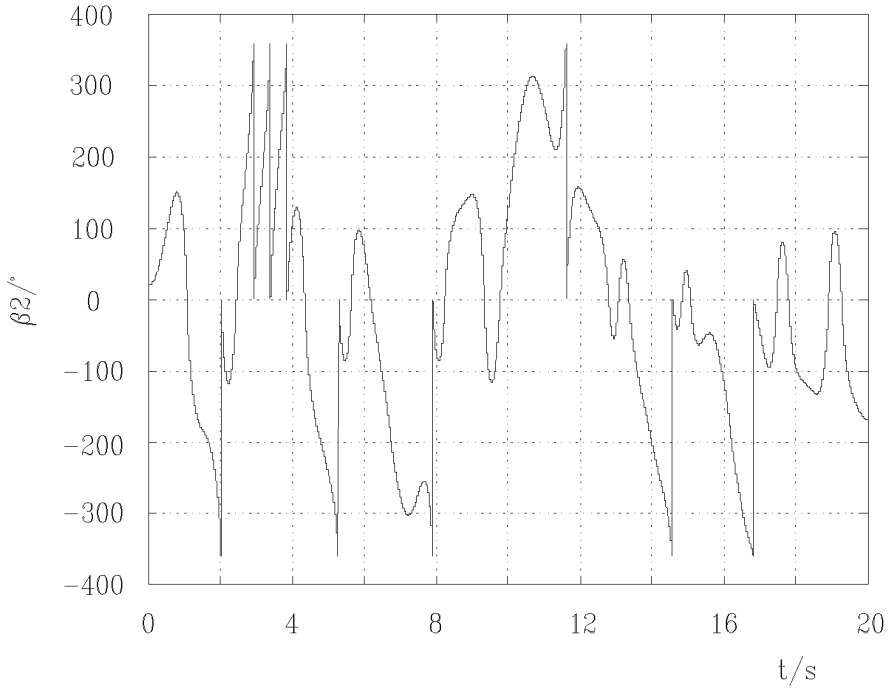

Bild 2.4: Zeitverlauf des Winkels β_2

3 Analyse und Synthese steifer Roboter

Die Analyse eines Systems ist sehr wichtig für die Simulation und den Entwurf eines Regelungskonzepts, das normalerweise wiederum von der Analysemethode abhängt. Wie oben erwähnt, sind die Mehrkörpersysteme nichtlineare Systeme, die schwierig zu analysieren sind. Trotzdem wurden einige Theorien zur Analyse der nichtlinearen Systeme entwickelt, wie z. B. die Ljapunowtheorie, Beschreibungsfunktionen usw. Die Ljapunowtheorie bezieht sich oft auf die direkte Methode von Ljapunow.

3.1 Die direkte Methode von Ljapunow

Die direkte Methode von Ljapunow stellt die wichtigste bisher bekannte Methode zur Stabilitätsanalyse bei nichtlinearen Systemen dar. Sie bietet die Möglichkeit, eine Aussage über die Stabilität der Ruhelage eines dynamischen Systems zu machen, ohne dazu die das System beschreibende Differentialgleichung zu lösen. Da es häufig bei nichtlinearen Systemen nicht möglich ist, explizite Lösungen anzugeben, ist dies ein entscheidender Vorteil. Die Ruhelage eines physikalischen Systems ist dadurch gekennzeichnet, daß die Gesamtenergie, also die Summe aus kinetischer und potentieller Energie gleich Null ist. In jedem anderen Bewegungszustand dagegen ist sie positiv, und die zeitliche Änderung der Gesamtenergie des Systems wird in der Umgebung der Ruhelage nie positiv sein. Gelingt es nun, die Energie als Funktion der Zustandsgrößen darzustellen und für diese skalare Funktion $V(\boldsymbol{x})$ zu zeigen, daß

1. $V(\boldsymbol{x}) > 0 \ \forall \boldsymbol{x} \neq \mathbf{0}$,

2. $V(\boldsymbol{x}) = 0$ für $\boldsymbol{x} = \mathbf{0}$ und
3. $\dot{V}(\boldsymbol{x}) \leq 0$,

dann ist das betrachtete System stabil (Unbehauen 1989).

3.2 Regelungsentwurf mit der direkten Methode von Ljapunow

Es gibt zwei Wege zur Auslegung eines Regelungskonzeptes mit der direkten Methode von Ljapunow: Eine besteht darin, zuerst einen Regler anzunehmen und dann eine Ljapunow-Funktion zu finden. Die andere funktioniert umgekehrt: Nach der Annahme einer Ljapunow-Funktion wird ein Regler so gewählt, daß die Funktion wirklich die Eigenschaften der Ljapunow-Funktion (Punkt 1., 2. und 3.) aufweist. Für die Positionsregelung von Robotern werden PD-Regler angewendet. Es gibt aber keinen allgemeinen Beweis für die Stabilität solcher Systeme, weil die Dynamik stark nichtlinear ist (Slotine und Li 1991).

Der Steuervektor $\boldsymbol{u}$ besteht aus einem PD-Regler und einem Gewichtskraftkompensator:

$$\boldsymbol{u} = -\boldsymbol{K}_D\dot{\boldsymbol{q}} + \boldsymbol{K}_P\tilde{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}), \tag{3.1}$$

wobei $\boldsymbol{K}_D$ und $\boldsymbol{K}_P \in \mathbb{R}^f \times \mathbb{R}^f$ konstante, positiv-definite Matrizen sind. Die Regelabweichung berechnet sich zu $\tilde{\boldsymbol{q}} = \boldsymbol{w} - \boldsymbol{q}$. Das PD-Glied des Reglers ist die Rückführungsregelung und der Gewichtskraftkompensator die Vorsteuerung, die die folgende Form hat:

$$g_1(\boldsymbol{q}) = m_1 g\frac{l_1}{2}\sin q_1 + m_2 g(\frac{l_2}{2}\sin q_2 + l_1 \sin q_1) \quad , \tag{3.2}$$

$$g_2(\boldsymbol{q}) = m_2 g\frac{l_2}{2}\sin q_2 \quad . \tag{3.3}$$

Bild 3.1 zeigt das Blockschaltbild des gesamten Systems. Aufgrund der physikalischen Gegebenheiten kann folgende Ljapunow-Funktion für das System gefunden werden:

$$V = \frac{1}{2}[\dot{\boldsymbol{q}}^T\boldsymbol{M}\dot{\boldsymbol{q}} + \tilde{\boldsymbol{q}}^T\boldsymbol{K}_P\tilde{\boldsymbol{q}}] > 0 \quad . \tag{3.4}$$

Der erste Teil entspricht der kinetischen Energie des Pendels, während der zweite Teil die potentielle Energie wiedergibt, die sich wie eine virtuelle Feder im Regler auswirkt. Nach dem mechanischen Energiesatz ist die Änderung der kinetischer Energie gleich der von den eingeprägten Kräften entwickelten Leistungen, nämlich

$$\dot{V} = \dot{\boldsymbol{q}}^T(\boldsymbol{u}(t) - \boldsymbol{g}(\boldsymbol{q})) + \dot{\tilde{\boldsymbol{q}}}^T\boldsymbol{K}_P\tilde{\boldsymbol{q}}, \tag{3.5}$$

$$\dot{V} = \dot{\boldsymbol{q}}^T(-\boldsymbol{K}_D\dot{\boldsymbol{q}} + \boldsymbol{K}_P\tilde{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) - \boldsymbol{g}(\boldsymbol{q})) + \dot{\tilde{\boldsymbol{q}}}^T\boldsymbol{K}_P\tilde{\boldsymbol{q}}. \tag{3.6}$$

Wenn $\boldsymbol{w}$ konstant ist, dann gilt $\dot{\tilde{\boldsymbol{q}}} = -\dot{\boldsymbol{q}}$ und

$$\dot{V} = -\dot{\boldsymbol{q}}^T\boldsymbol{K}_D\dot{\boldsymbol{q}} \leq 0 \quad . \tag{3.7}$$

Damit ist das System global asymptotisch stabil (in Anlehnung an Yuan, Book und Huggins 1993).

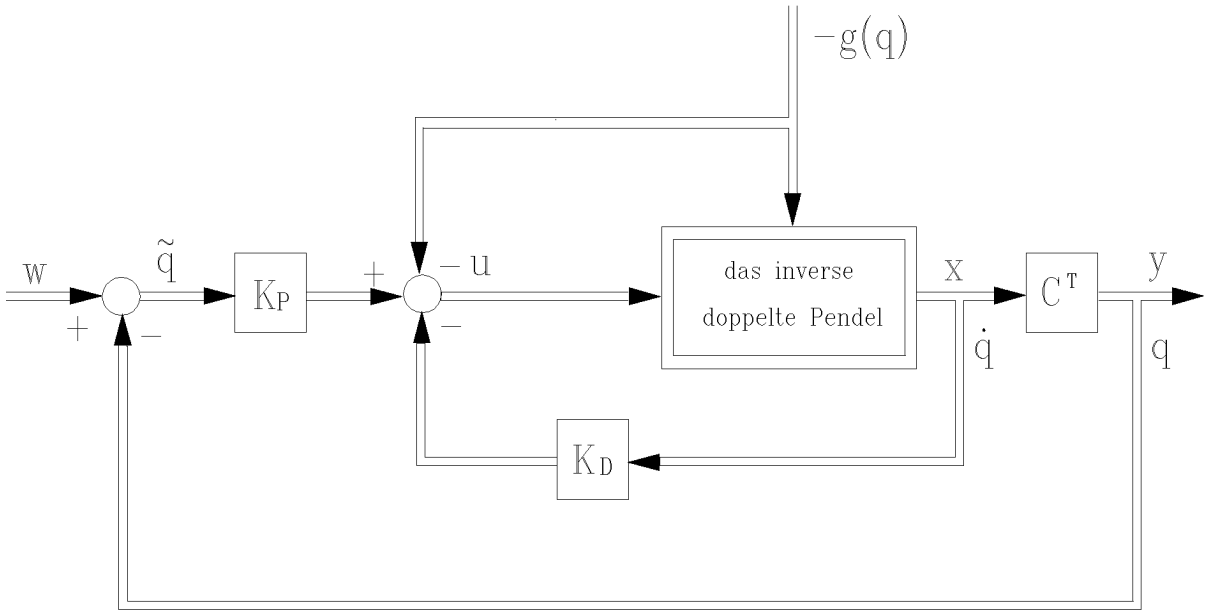

Bild 3.1: Blockschaltbild des gesamten Systems

3.3 Simulationsergebnisse

Betrachtet wurden die Rückführungsmatrizen $\boldsymbol{K}_D = 100\boldsymbol{I}$, $\boldsymbol{K}_P = 20\boldsymbol{K}_D$. Als Anfangswerte wurden $\beta_1(0) = 10°$, $\beta_2(0) = 20°$ und als Sollwert $\boldsymbol{w}(t) = \boldsymbol{0}° \in \mathbb{R}^2$ verwendet. Den Zeitverlauf der Winkel $\beta_1(t)$ und $\beta_2(t)$ und des Steuervektors $\boldsymbol{u}(t)$ zeigen Bild 3.2 bis 3.5. Mit dem PD-Regler wurde eine stabile Positionsregelung erreicht.

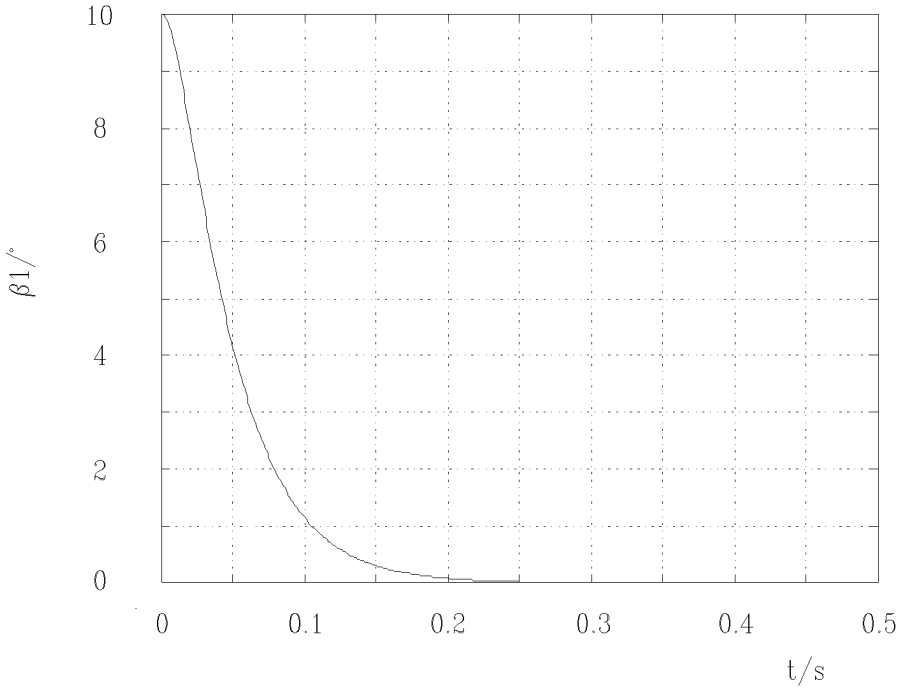

Bild 3.2: Zeitverlauf des Winkels Pendels $\beta_1(t)$

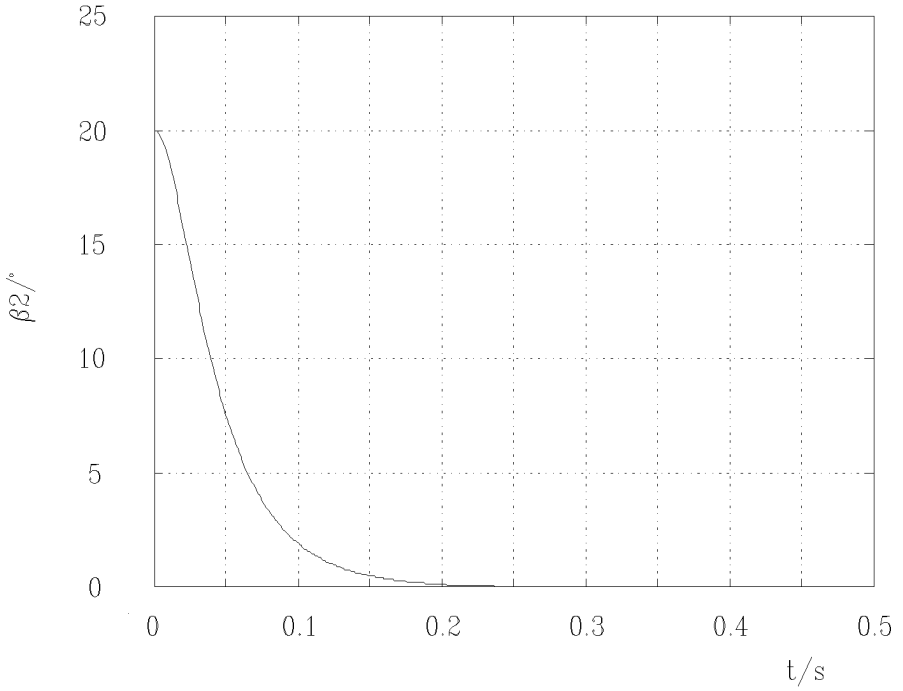

Bild 3.3: Zeitverlauf des Winkels Pendels $\beta_2(t)$

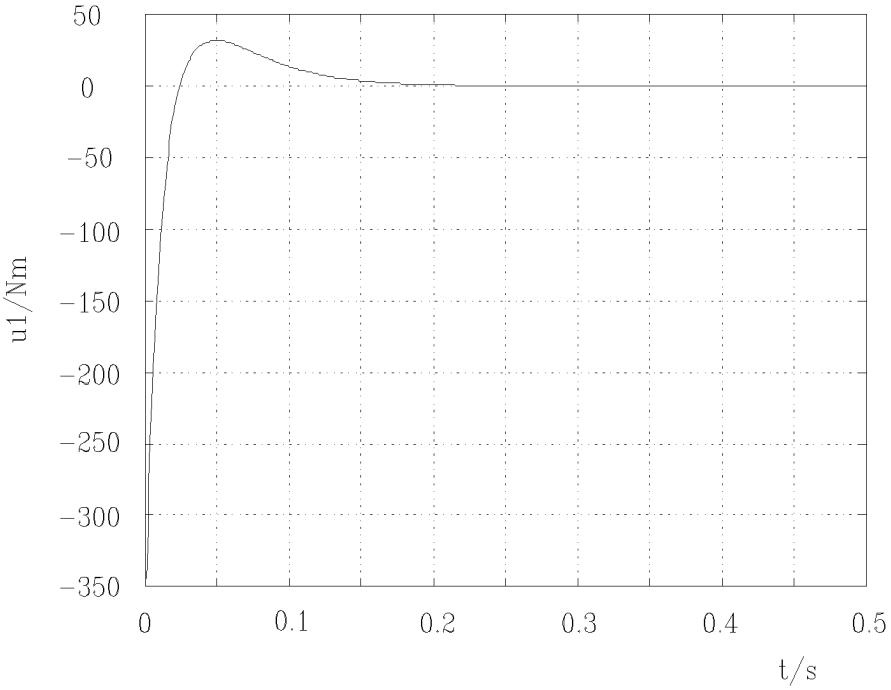

Bild 3.4: Zeitverlauf der Stellgröße $u_1(t)$

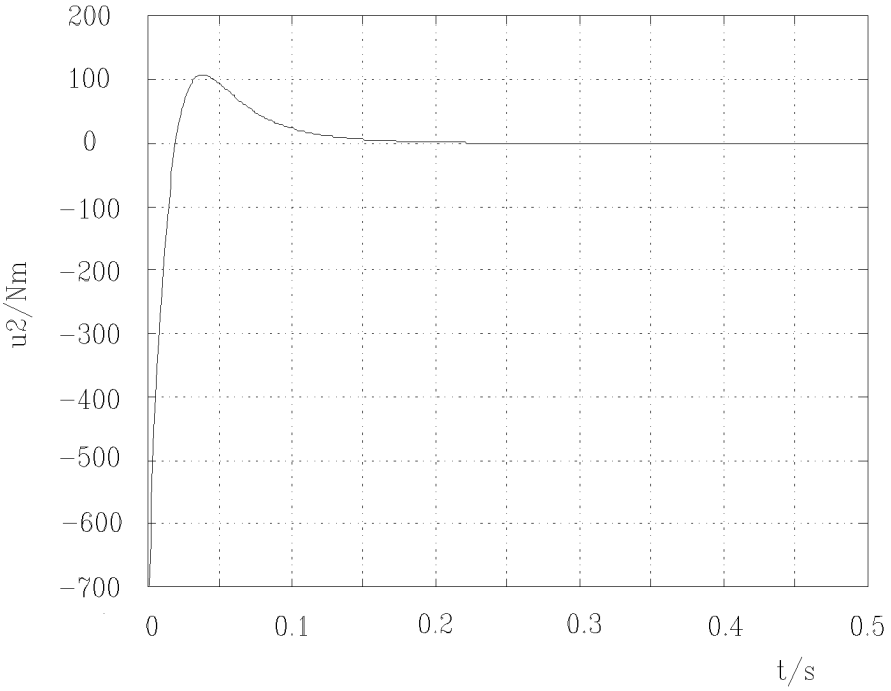

Bild 3.5: Zeitverlauf der Stellgröße $u_2(t)$

4 Zusammenfassung und Ausblick

Der vorliegende Bericht informiert über den Stand der Arbeiten in dem Projekt „Modellbildung, Simulation und Regelung flexibler Roboter“. Die Modellierung und Simulation eines starren Roboters, hier am Beispiel eines inversen doppelten Pendels, erfolgten in der objektorientierten Programmiersprache C++ unter Verwendung der Programmierumgebung „Mobile“. Um mit „Mobile“ zu modellieren, sind sowohl C++- Programmier- als auch Kinematikkenntnisse notwendig. Nach der Einarbeitung in „Mobile“ kann allerdings viel Zeit gespart werden, insbesondere bei komplexen mechanischen Systemen. Eine Bestimmung der Differentialgleichungen ist nicht nötig, stattdessen erfolgt eine Definition aller Glieder des Systems und eine Beschreibung ihrer Verbindungen in C++. Zwar kann man kein symbolisches Modell erhalten, die Dynamik aller Arbeitspunkte ist aber lösbar. Zur Regelung eines starren Roboters wurde mit Hilfe der direkte Methode von Ljapunow ein PD-Regler mit einem nichtlinearen Vorsteuerungskompensator ausgelegt und die globale Stabilität des geregelten Systems nachgewiesen. Simulationsergebnisse für das exemplarisch betrachtete inverse doppelte Pendel wurden vorgestellt.

In der Fortführung dieser Arbeit können

1. die Verbindung zwischen „Mobile“ und elektrischen Antriebsmotoren oder hydraulischen Antriebselementen,

2. die Regelung einer Regelstrecke, die mehr als zwei Freiheitsgrade besitzt, sowie

3. alternative Regelungskonzepte, wie z. B. adaptive Regler oder Sliding-Mode Regelungen

untersucht werden.

5 Literaturverzeichnis

Kecskeméthy, A. 1993. *Objektorientierte Modellierung der Dynamik von Mehrkörpersystemen mit Hilfe von Übertragungselementen.* VDI Forschrittberichte, Reihe 20, Nr. 88. Düsseldorf: VDI.

Nijmeijer, H. und A. J. van der Schaft. 1990. *Nonlinear Dynamical Control System.* Berlin: Springer.

Schneider, M. und M. Hiller 1993. *„Regelung von Großrobotern".* Forschungsnotiz Fachgebiet Mechatronik. Universität-GH-Duisburg.

Slotine, J.-J. E. und W. Li. 1991. *Applied Nonlinear Control.* Englewood Cliffs, New Jersey: Prentice-Hall.

Unbehauen, H. 1989. *Regelungstechnik II.* Braunschweig: Vieweg.

Yuan, B.-S., W. J. Book und J. D. Huggins. 1993. Dynamik of Flexible Manipulator Arms: Alternative Derivation, Verification, and Characteristics for Control. *Transactions of the ASME 115.* 394-404.

A Quelltext des Programms

Ein inverses doppeltes Pendel läßt sich durch das folgende Möbile-Programm beschreiben:

```
//reference frames
Frame K0, K1, K2, K3 ;

//defining variables
AngularVariable beta_1,beta_2 ;

//defining joints
Joint R1( K0, K1,beta_1, x_axis ) ;
Joint R2( K2, K3,beta_2, x_axis ) ;

//vectors
Vector a_1, r_1, a_2, r_2;

//defining links connecting joints
RigidLink arm_a (K1, K2, a_1) ;

//defining masses
real m1, m2 ;
InertiaTensor THETA_d1, THETA_d2;
MassElement M1( K1, THETA_d1 , m1, r_1 ) ;
MassElement M2( K3, THETA_d2 , m2, r_2 ) ;

//defining parameter values
a_1 = a_2 = NullState ;
a_1.z = 1.0 ;
r_1.z = a_1.z/2 ;
a_2.z = 1.0 ;
r_2.z = a_2.z/2 ;

m1 = m2 = 1.0 ;
THETA_d1 = THETA_d2 = NullState ;
THETA_d1 = ( 0.0,0.0,0.0833 ) ;
THETA_d2 = ( 0.0,0.0,0.0833 ) ;

//building up the Pendulum
MapChain pendulum ;
pendulum << R1 << arm_a << R2 << M1 << M2 ;
VariableList q ;
q << beta_1 << beta_2 ;

//defining gravitation
Vector gravitation ( 0.0,0.0,-9.8 ) ;
```

```
//values of the independent variables
beta_1.q =DEG_TO_RAD * 30 ;
beta_2.q =DEG_TO_RAD * 60 ;

EqmSolver system ( q, pendulum, K0, gravitation ) ;
system.buildEquations () ;
system.saveEquations () ;
system.solveEquations () ;
```