MTP2 Praktikumsbericht

Karolina Bernat, Olliver Steenbuck

Fakultät Technik und Informatik
Department Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

1 Weltraummissionen

1.1 Erdumkreisung, Fluchtgeschwindigkeit und geostationäre Bahn

Im Folgenden wird beschrieben, wie die antriebslose Phase eines Satelliten verläuft, der vermittels einer Trägerrakete in eine Startposition x_0 gebracht wird um dann antrieblos weiter zu fliegen. Der Einfluss des Satelliten auf die Erde wird hier vernachlässigt.

Im Modell folgende Konstanten sind bekannt:
Erdradius: $r_E = 6378km$
Erdmasse: $m_E = 5.9736 \cdot 10^{24} kg$
Gravitationskonstante: $G = 66.743 \cdot 10^{-12} \frac{m^3}{kg \cdot s^2}$

Startpositionsvektor

Zur Errechnung der Startposition des Satellitens nutzen wir:

$$\overrightarrow{x_0} = [\cos(\delta \cdot (r + h_0)), \sin(\delta \cdot (r + h_0))]$$

Die Funktion ***"Startposition"*** sieht folgendermaßen aus:

```
function x0 = Startposition(deltaDegree, h0)

r = 6378000;    % Erdradius [m]
h = r + h0;     % Hypothenuse
x0 = [cosd(deltaDegree) * h ; sind(deltaDegree) * h ];
```

Startgeschwindigkeitsvektor

Im nächsten Schritt wird der Startgeschwindigkeitswektor $\overrightarrow{v_{0,Welt}}$ berechnet. Hierfür werden zunächst die Einheitsvektoren in Tangential- sowie Normalrichtung konstruiert. Danach werden die Tangential- und Normalkomponente der Startgeschwindigkeit berechnet, wobei daraus die Startgeschwindigkeit zusammengebaut wird. Die Funktion ***"vStart"***, die diese Schritte durchführt, sieht folgendermaßen aus:

```
function v0welt = vStart(v0, thetaDegree, x0)

% v0 - Startgeschwindigkeit
% thetaDegree Winket theta in Grad
% x0 STartposition

nE = x0 / norm(x0);      % Einheitsvektor in Normalrichtung fuer x0
tE = [nE(2) ; -nE(1)];   % Einheitvektor in Tangentialrichtung fuer x0

vt = cosd(thetaDegree) * v0;     % Tangentialkomponente fuer v0
vn = sind(thetaDegree) * v0;     % Normalkomponente fuer v0

v0welt = tE * vt + nE * vn;      % Geschwindigkeitsvektor
```

Beschleunigung

Zur Berechnung der Beschleunigung des Satellitens wird die aktuelle Position des Satellitens benötigt. Durch die Summierung der Kräfte:

$$\sum F = m \cdot a$$

ergibt sich:

$$a = \frac{\sum F}{m}$$

Dabei ist m die Masse des Satellitens, die hier vernachlässigt werden kann. Somit ergibt sich:

$$a = F_{SE}$$

und folglich:

$$a = G \cdot \frac{m_E \cdot m_S}{r_{SE}^2} \cdot \overrightarrow{e_{SE}}$$

wobei F_{SE} die Kraft die von der Erde zum Satelliten wirkt, r_{SE} der Abstand von der Erdmitte zum Satelliten und $\overrightarrow{e_{SE}}$ der Einheitsvektor vom Satelliten zur Erde ist und m_S (Masse des Satellitens) vernachlässigt wird.

Die Funktion ***"Beschleunigung"*** sieht folgendermaßen aus:

```
function a = Beschleunigung(xAktuell)

xE = xAktuell / norm(xAktuell);    % Einheitsvektor fuer xAktuell
eSE = xE * (-1);    % Einheitsvektor vom Satelliten zu Erde - umgekehrte
    Richtung zu xE

```

```
r = norm(xAktuell); % Abstand von Erdmitte zum Satellit

mE = 5.9736 * 10^24;     % Erdmasse in [kg]
G = 66.743 * 10^-12;     % Gravitationskonstante in [m^3/kg*s^2]

a = G * mE / r^2 * eSE; % Beschleunigung
```

Kontakt

Sobald der Satellit die Erde berührt soll die Simulation beendet werden. Hierfür wurde die Funktion ***"Kontakt"*** geschrieben, die dies kontrolliert und wie folgt aufgebaut ist:

```
function endBedingung = Kontakt(xAktuell)

rErde = 6378000;
xAbstand = norm(xAktuell);

if (xAbstand > rErde)
        endBedingung = 0;
else
    endBedingung = 1;
end
```

Gesamtmodell

Das vollständige Modell wurde in Simulink modelliert und kann der folgenden Abbildung entnommen werden:

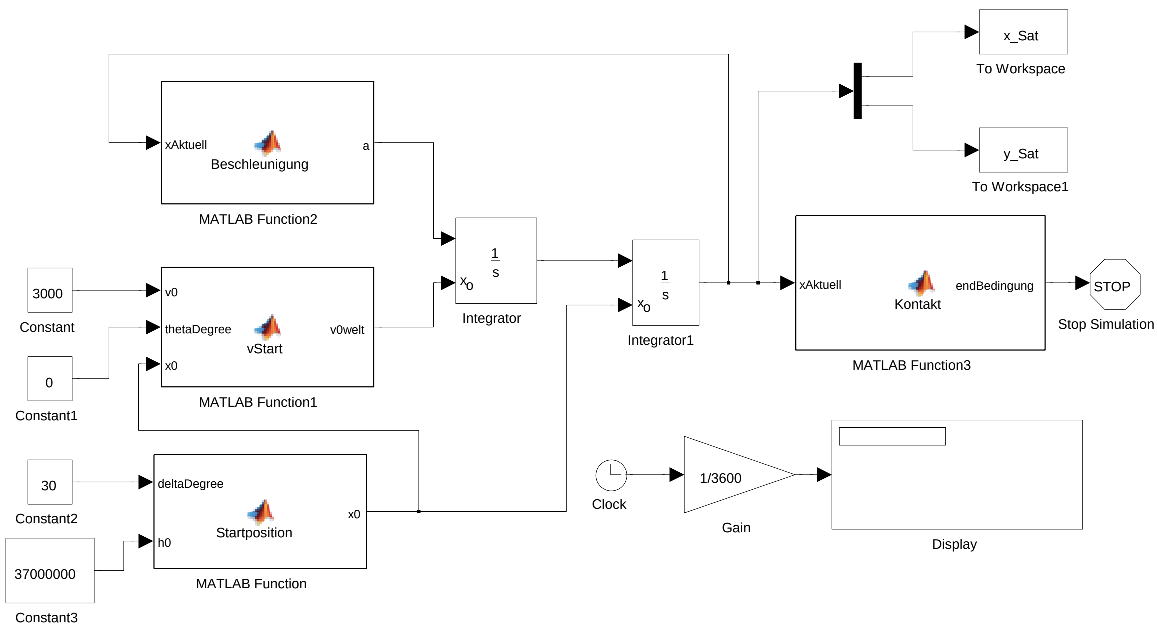


Abbildung 1.1: Gesamtmodell in Matlab/Simulink

Versuchsdurchführung

- Eine Kreisbahn in gleicher Höhe

Folgende Voreinstellungen wurden für die Simulation gewählt:

$$\delta = 30^{\circ},\, h_0 = 400km,\, \theta = 0^{\circ}$$

Bei einer Startgeschwindigkeit $v_0 = 7{,}65\frac{km}{s}$ und der Simulationszeit von $1{,}542h$ umkreist der Satellit die Erde bei einer konstanten Höhe genau ein Mal wie in der folgenden Abbildung zu sehen ist:

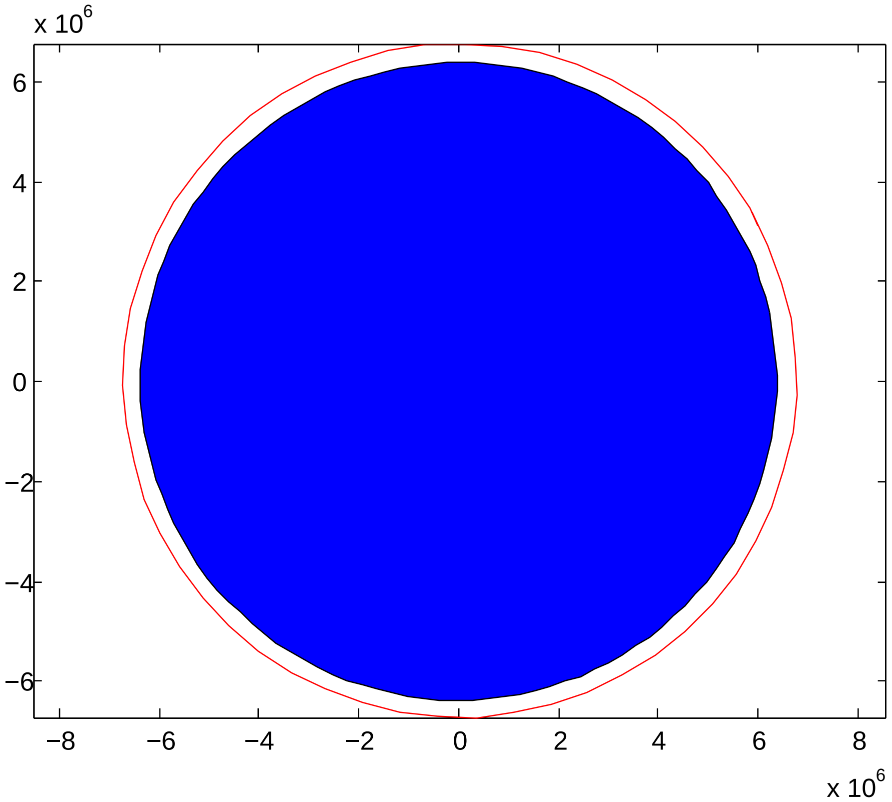


Abbildung 1.2: Eine Kreisbahn in gleicher Höhe

- Entfliehung der Erdanziehungskraft

Folgende Voreinstellungen wurden für die Simulation gewählt:

$$\delta = 30^{\circ}, h_0 = 400km, \theta = 0^{\circ}$$

Bei einer Startgeschwindigkeit $v_0 = 10{,}85\frac{km}{s}$ und der Simulationszeit von $1.000.000s$ entflieht der Satellit der Erde, wie in der nachfolgenden Grafik zu sehen ist:

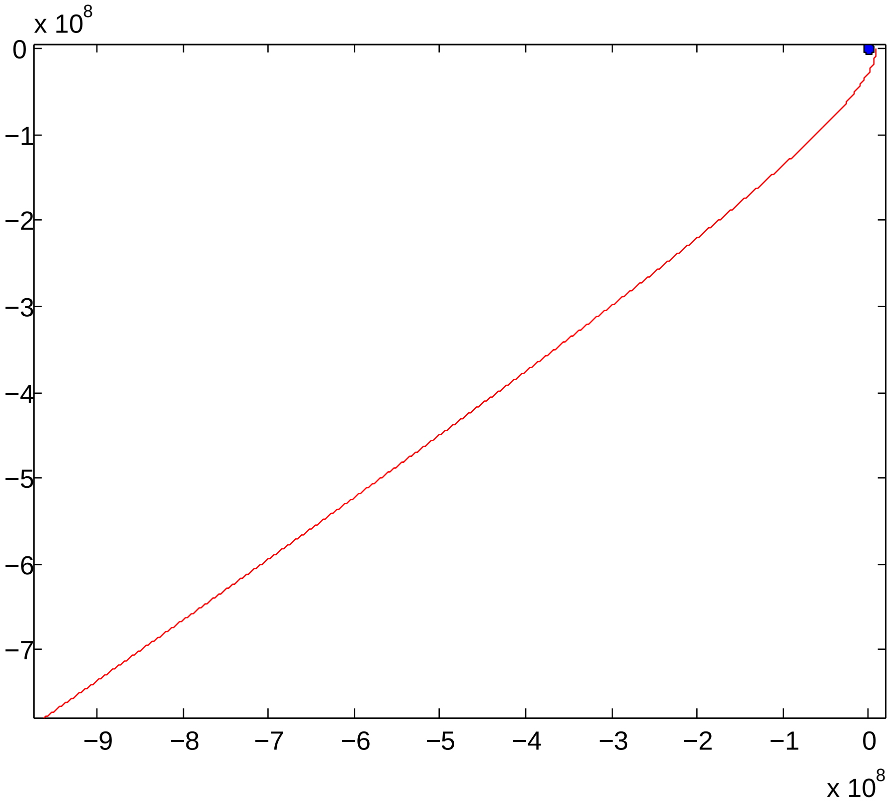


Abbildung 1.3: Entfliehen der Erde

- Eine Kreisbahn innerhalb eines Tages

$$\delta = 30^{\circ}, \theta = 0^{\circ}$$

Bei einer Starthöhe von $h_0 = 37.000km$ und der Startgeschwindigkeit von $v_0 = 2{,}995\frac{km}{s}$ umkreist der Satellit die Erde ein Mal in Simulierten $24h$, wie in der nachfolgenden Grafik zu sehen ist:

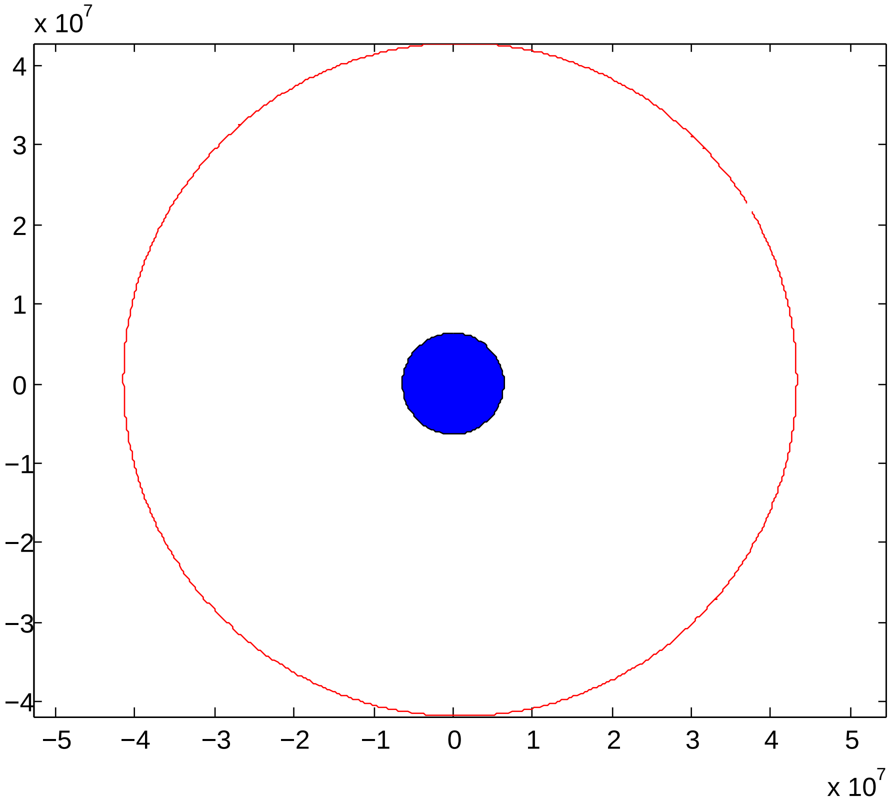


Abbildung 1.4: Eine Kreisbahn innerhalb eines Tages

1.2 Mondumkreisung

In diesem Abschnitt wird die Simulation eines Satelliten, der von der Erde zum Mond fliegt vorgestellt. Hier wird angenommen, dass sich der Mond nicht bewegt.

Folgende Kennzahlen sind bekannt:
Mondposition (fest): $x_M = (0, -380.000)^T km$
Mondmasse: $m_M = 7{,}3480 \cdot 10^{22} kg$

Beschleunigung

Für das Matlab-Modell wird die Funktion ***"Beschleunigung"*** ergänzt, so das die vom Mond auf den Satelliten wirkende Kraft F_M berücksichtigt wird. Diese wird wie folgt berechnet:

$$F_{SE} = G \cdot \frac{m_E}{r_{SE}^2} \cdot \overrightarrow{e_{SE}}$$

$$F_{SM} = G \cdot \frac{m_M}{r_{SM}^2} \cdot \overrightarrow{e_{SM}}$$

wobei m_M die Mondmasse, r_{SM} die Entfernung vom Satelliten zum Mond und $\overrightarrow{e_{SM}}$ der Einheitsvektor vom Satelliten zu Mond ist.

Um die Beschleunigung des Satelliten zu errechnen nutzen wir:

$$\sum F = m \cdot a$$

wobei m die Masse des Satelliten ist und im Modell vernachlässigt wird. Daraus folgt:

$$a = F_{SE} + F_{SM}$$

Die angepasste Matlab-Funktion sieht wie folgt aus:

```
function a = Beschleunigung(xAktuell)

xE = xAktuell / norm(xAktuell);   % Einheiitsvektor fuer xAktuell
eSE = xE * (-1);         % Einheitsvektor vom Satelliten zu Erde -
    umgekehrte Richtung zu xE

xM = [0; -380000000];    % Mondposition
vSM = xM - xAktuell;     % Vektor vom Satelliten zum Mond
rSM = norm(vSM);
eSM = vSM / norm(vSM);   % Einheitsvektor vom Sat. zu Mond

r = norm(xAktuell);      % Abstand von Erdmitte zum Satellit

mE = 5.9736 * 10^24;     % Erdmasse in [kg]
mM = 7.3480 * 10^22;     % Mondmasse in [kg]
G = 66.743 * 10^-12;     % Gravitationskonstante in [m^3/kg*s^2]

F_SE = G * mE / r^2 * eSE; % Kraft auf Sat. von der Erde
F_SM = G * mM / rSM^2 * eSM; % Kraft auf Sat. vom Mond

a = F_SE + F_SM;         % Beschleunigung
```

Das Gesamtmodell kann der folgenden Abbildung entnehmen:

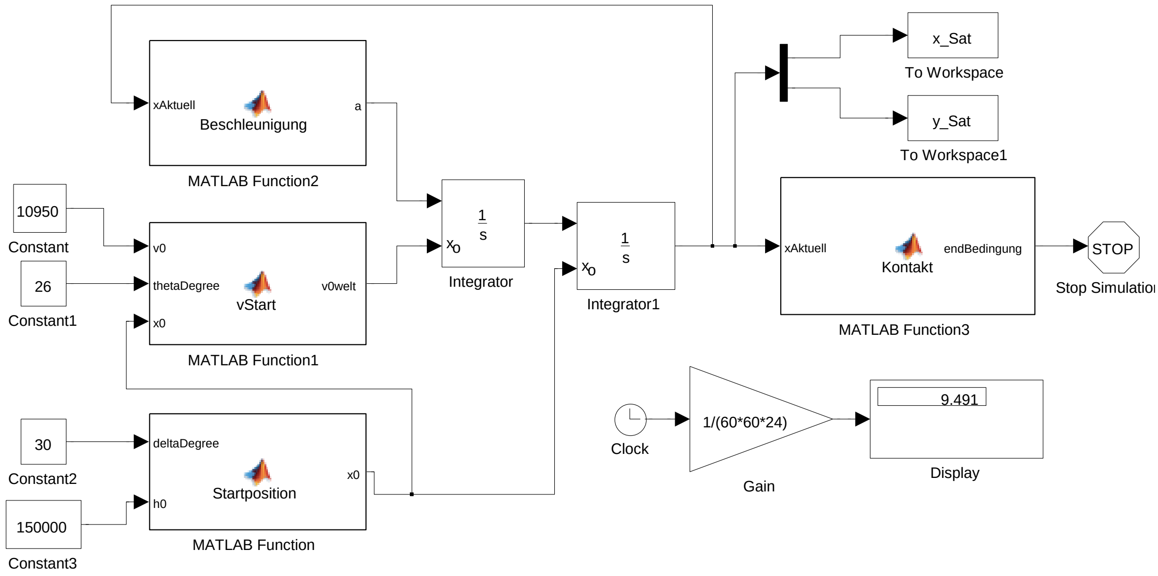


Abbildung 1.5: Gesamtmodell in Matlab/Simulink

Versuchsdurchführung

Für die Durchführung der Simulation wurden folgende Einstellungen verwendet:

$$\delta_0 = 30^{\circ}, h_0 = 150km$$

Bei einer Startgeschwindigkeit $v_0 = 10{,}95\frac{km}{s}$ und einer Neigung $\theta = 26^{\circ}$ fliegt der Satellit von der Erde um den Mond in einer 8-förmigen Schleife. Dabei dauert die Mission 9,491 Tage:

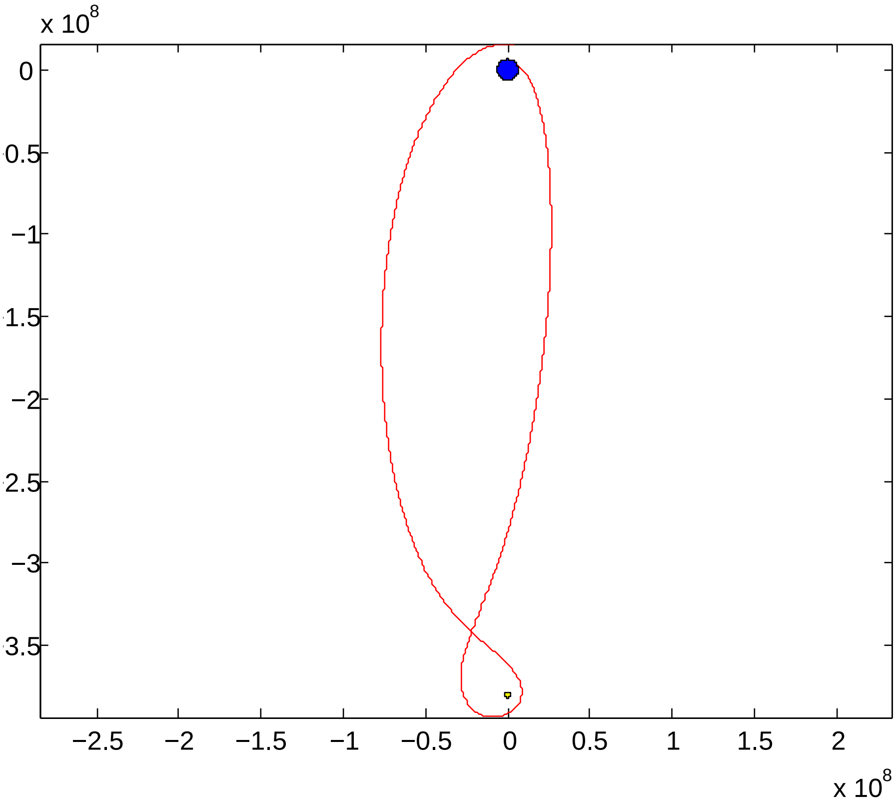


Abbildung 1.6: Mondmission

2 Crazy Pendulum

Mit dem Modell des Crazy Pendulums soll simuliert werden, wie ein Pendel (Quaderform), dass an einer Seiltrommel mit zwei Federn befestigt ist, sich bewegt. Dabei werden die Einflüsse der Pendelstange sowie der Seiltrommel in der Simulation vernachlässigt. Das System ist in der untenstehenden Grafik abgebildet:

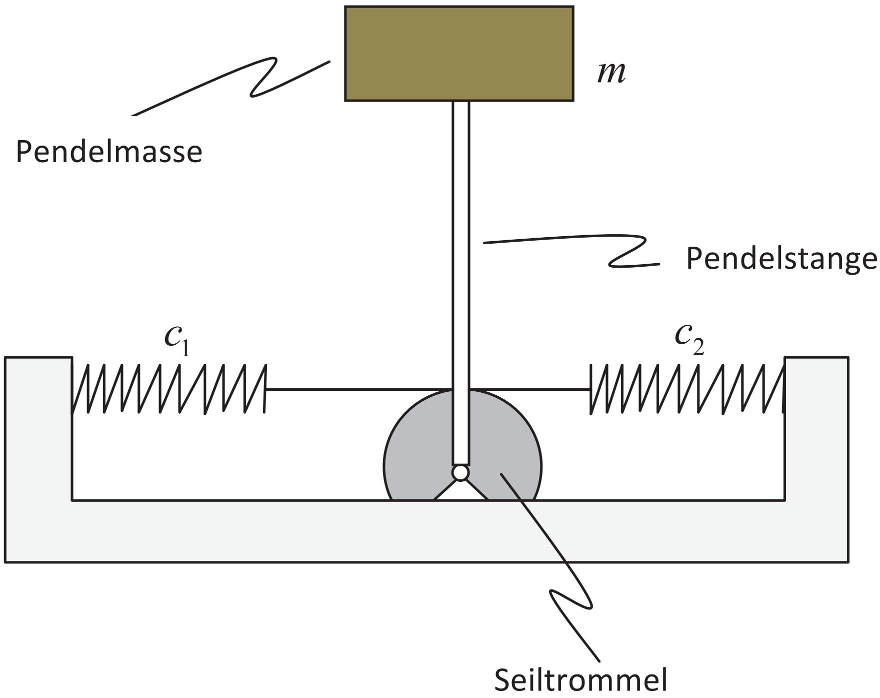


Abbildung 2.1: Crazy Pendulum

Im Modell bekannt gegeben werden: Pendelmasse $m[kg]$, Abstand $L[m]$, Quadermaße $w[m]$,$h[m]$, Trommelradius $r[m]$, Anfangsauslenkung des Pendels $\varphi_0[^\circ]$, Federkonstanten $c_1[\frac{N}{m}]$, $c_2[\frac{N}{m}]$. Zu bestimmen ist die Winkelbeschleunigung $\varphi(t)$, die über ein Matlab-Scope sowie in einem virtuellem Modell ausgegeben werden soll.

Folgende Kräfte wirken auf das modellierte Pendel:

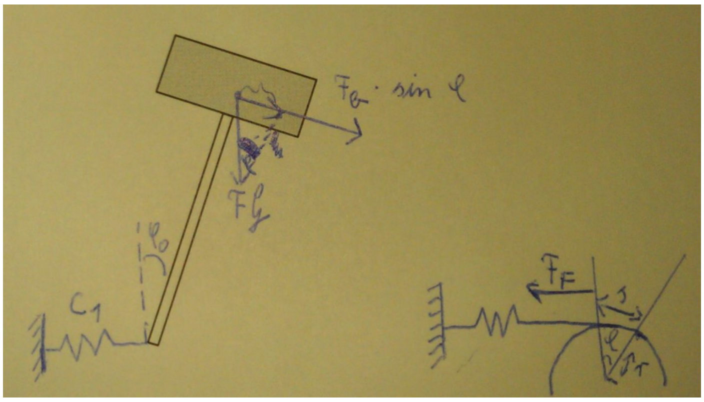


Abbildung 2.2: Freikörperbild des Pendels

Drehmomente

Die auf das Pendel wirkende Kraft ist die Gravitationskraft:

$$F_G = m \cdot g$$

wobei $g = 9{,}81[\frac{m}{s^2}]$. Da das Pendel um den Winkel φ geneigt ist, berechnen wir das wirkende Moment M_G wie folgt:

$$M_G = F_G \cdot sin(\varphi) \cdot (L + \frac{h}{2})$$

Die Kraft, die auf eine gerade aufgespannte Feder wirkt, errechnet sich wie folgt:

$$F_F = c_i \cdot f$$

wobei $i = \{1{,}2\}$ und f die Auslenkung der Feder ist sie kann wie folgt errechnet werden:

$$f = \varphi \cdot r$$

Das auf die gerade aufgespannte Feder wirkende Drehmoment errechnen wir wie folgt:

$$M_F = F_F \cdot r$$

Trägheitsmoment

Das Massenträgheitsmoment J_S für das Pendel errechnen wir wie folgt (Berechnung bei einer Quaderform und unter Vernachlässigung der Pendelstange):

$$J_S = \frac{1}{12} \cdot m(w^2 + h^2)$$

Dadurch, dass das Pendel nicht um den eigenen Mittelpunkt gedreht wird, muss das Trägheitsmoment J_A nach dem Satz von Steiner errechnet werden:

$$J_A = J_S + m \cdot (L + \frac{h}{2})^2$$

Differenzialgleichung

Aus den oben aufgeführten Schritten und unter Berücksichtigung des Drallsatzes kann hergeleitet werden:

$$\sum M = J_A \cdot \ddot{\varphi}$$

durch Umstellung der Formel kann die Winkelbeschleunigung $\ddot{\varphi}$ bestimmt werden:

$$\ddot{\varphi} = \frac{M_G - M_F}{J_A}$$

Die Matlab-Funktion zur Berechnung von $\ddot{\varphi}$ sieht wie folgt aus:

```
function phiPP = fcn(m, l, w, h, r, phi, c)

g = -9.81;  % Beschleinigung durch Gravitation
FG = m * g; % Gravitationskraft
MG = FG * sin(phi) * (l + h/2); % Drehmoment des Pendels

s = phi * r;    % Auslenkung
FF = c * s;     % Federkraft
MF = FF * r;    % Drehmoment der Feder

JS = 1/12 * m * (w^2 + h^2);    % Massentraegheitsmoments des Quaders
JA = JS + m * (l + h/2)^2;      % Mass.tr.moment nach Satz von Steiner

phiPP = (MG - MF)/JA;
```

Das vollständige Simulink Schaltbild kann der unten stehenden Abbildung entnommen werden:

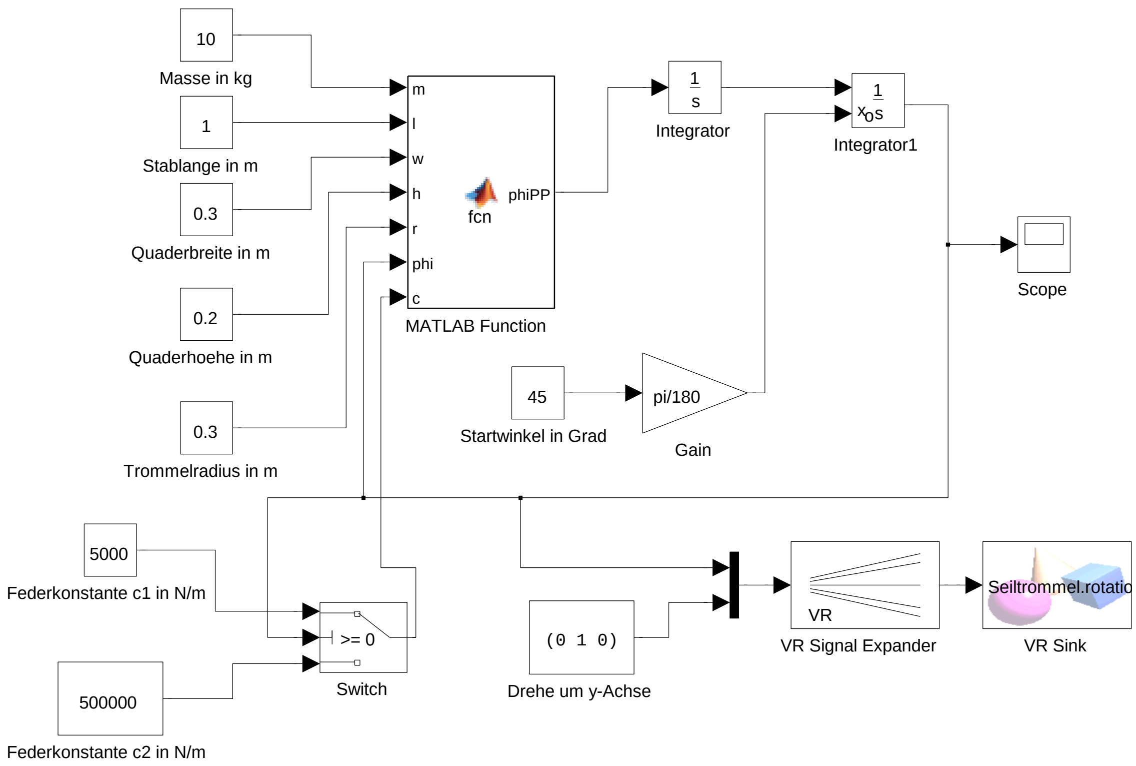


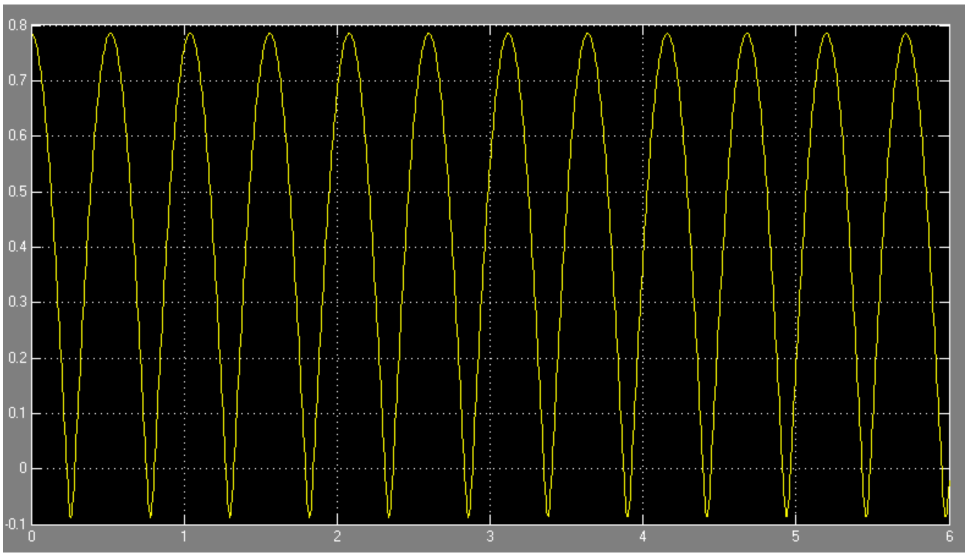
Abbildung 2.3: Gesamtmodell in Matlab/Simulink

Versuchsdurchführung

Bei der Durchführung des Versuches galten folgende Voreinstellungen:

$L = 1m$, $r = 0.3m$, $w = 0.3m$, $h = 0.2m$, $c_1 = 5000\frac{N}{m}$, $c_2 = 500000\frac{N}{m}$, $\varphi = 45^{\circ}$

Bei einer Simulationszeit von 6s ergibt sich das folgende Scope-Abbild:



In der 3-D Visualisierung sieht man deutlich, dass das Pendel sehr stark von der Feder mit der größeren Federkonstante beeinflusst wird:

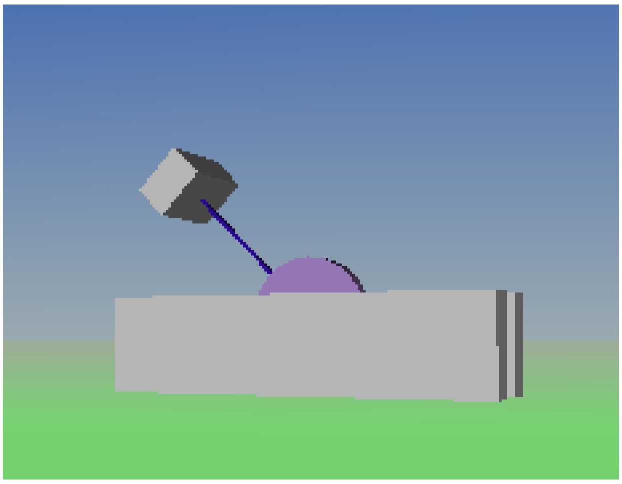


Abbildung 2.4: Pendel 3-D

3 Schwingungsgedämpfter Tisch

In der Modellierung des schwingungsgedämpfen Tisches handelt es sich um zwei Tischplatten, die mit jeweils zwei Federn sowie mit zwei Dämpfern verbunden sind (siehe Bild unten).

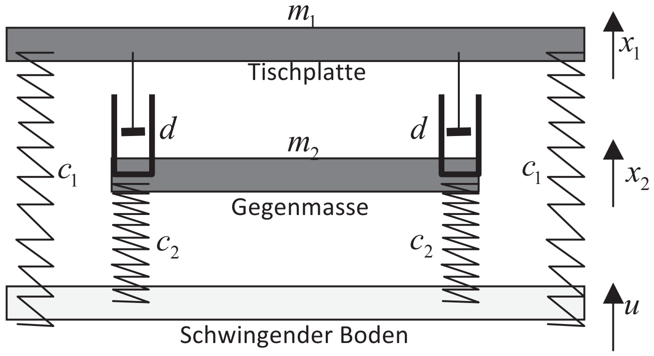


Abbildung 3.1: Schwingungsgedämpfter Tisch-Modell

Diese sind auf einem Untergrund befestigt, der in der Simulation durch einen Impuls in Bewegung gesetzt wird, wodurch eine Schwingung der beiden Tischplatten beobachtet werden kann. Ein Freikörperbild des Systems mit den angreifenden Kräften kann der unten stehenden Abbildung entnommen werden:

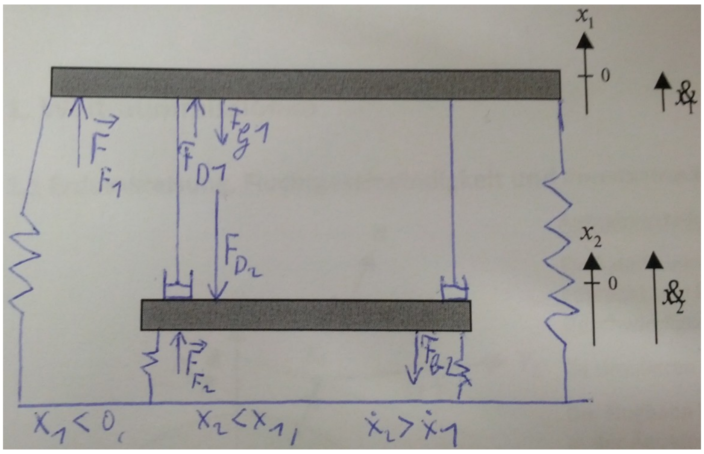


Abbildung 3.2: Schwingungsgedämpfter Tisch im Freikörperbild

Die für die Simulation vorgegebenen Kenngrößen sind: Masse der oberen Tischplatte $m_1[kg]$, Masse der zweiten Tischplatte $m_2[kg]$, Federkonstante der äußeren Feder $c_1[\frac{N}{m}]$, Federkonstante der inneren Feder $c_2[\frac{N}{m}]$, Dämpfungskonstante des Dämpfers $d[\frac{N \cdot s}{m}]$.

Differenzialgleichungen

Die auf das System wirkenden Kräfte sind folgende

- Gravitationskraft: $F_G = m \cdot g$, wobei g die Erdbeschleunigung ist und ca. $9{,}81[\frac{m}{s^2}]$ beträgt.
- Federkraft: $F_F = c \cdot x$, wobei x die Auslenkung der Feder beschreibt
- Dämpfungskraft $F_D = d \cdot \dot{x}$, wobei $\dot{x}$ die Geschwindigkeit des Dämpfers beschreibt

Aus dem Schwerpunktsatz ergibt sich:

$$\sum F = m \cdot \ddot{x}$$

Für unser System (siehe Freikörperbild) können demnach folgende Differenzialgleichungen abgeleitet werden:

$$-F_{G_1} - F_{F_1} + F_D = \ddot{x_1} \cdot m_1$$

$$-F_{G_2} - F_{F_2} + F_D = \ddot{x_2} \cdot m_2$$

woraus sich ergibt:

$$\ddot{x}_1 = \frac{-F_{G_1} - F_{F_1} + F_D}{m_1}$$

$$\ddot{x}_2 = \frac{-F_{G_2} - F_{F_2} + F_D}{m_2}$$

Durch das Einsetzen der Formeln erahlten wir:

$$\ddot{x}_1 = \frac{-m_1 \cdot g - 2 \cdot c_1 \cdot (x_1 - u) + 2 \cdot d \cdot (\dot{x}_2 - \dot{x}_1)}{m_1}$$

$$\ddot{x}_2 = \frac{-m_2 \cdot g - 2 \cdot c_2 \cdot (x_2 - u) - 2 \cdot d \cdot (\dot{x}_2 - \dot{x}_1)}{m_2}$$

dabei ist u der Abstand, um den das System schlagartig bewegt wird und somit ins Schwingen kommt.

Die Matlab-Funktionen sehen dann folgendermaßen aus:

```
function x1PP = fcn(m1, c1, d, x1, x1P, x2P, u)

g = 9.81;   % Erdbechleunigung

x1PP = (-m1*g - 2*c1*(x1 - u) + 2*d*(x2P - x1P)) / m1;
```

```
function x2PP = fcn(m2, c2, d, x2, x1P, x2P, u)

g = 9.81;   % Erdbechleunigung

x2PP = (-m2*g - 2*c2*(x2 - u) - 2*d*(x2P - x1P)) / m2;
```

Das gesamte Simulink-Schaltbild kann der untenstehenden Abbildung entnommen werden:

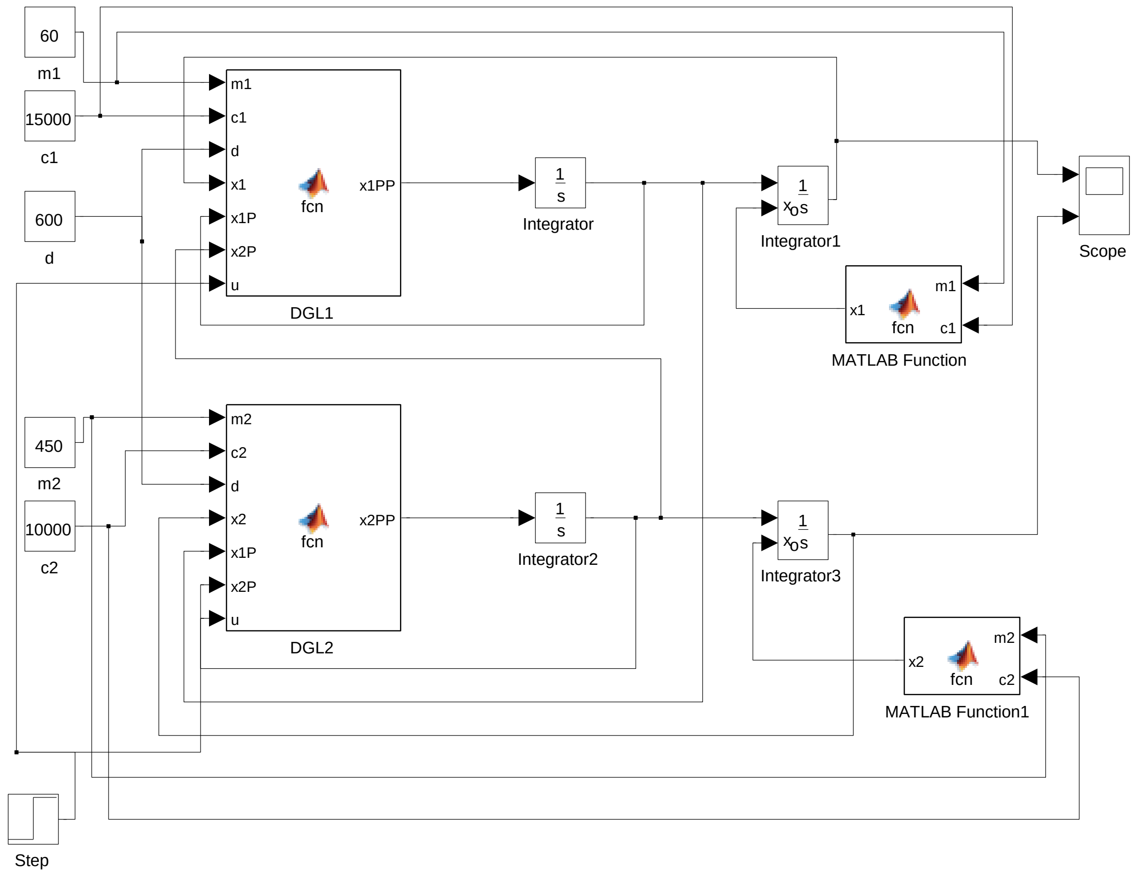


Abbildung 3.3: Gesamtmodell in Matlab/Simulink

System in Ruhelage

Solange das System nicht von dem Sprungsignal in Bewegung versetzt wird, soll es in Ruhelage bleiben. Dies bedeutet, dass keine Bewegungen stattfinden. Daraus ergibt sich:

$$\dot{x}_1 = 0, \ddot{x}_1 = 0$$

$$\dot{x}_2 = 0, \ddot{x}_2 = 0$$

Durch das Einsetzen in die DGL für $\ddot{x}_1$ und $\ddot{x}_2$ erhalten wir:

$$\frac{-m_1 \cdot g - 2 \cdot c_1 \cdot (x_1 - u) + 2 \cdot d \cdot (0-0)}{m_1} = 0$$

$$\frac{-m_2 \cdot g - 2 \cdot c_2 \cdot (x_2 - u) - 2 \cdot d \cdot (0-0)}{m_2} = 0$$

und folglich (bei der Annahme, dass zum Simulationsstartpunkt $u = 0$ ist):

$$x_1 = -\frac{m_1 \cdot g}{c_1}$$

$$x_2 = -\frac{m_2 \cdot g}{c_2}$$

Versuchsdurchführung

Bei der Durchführung des Versuchs gelten folgende Voreinstellungen:

$c_1 = 15000\frac{N}{m}$, $c_2 = 10000\frac{N}{m}$, $d = 600\frac{N \cdot s}{m}$, $m_1 = 60kg$, $m_2 = 450kg$

Das System wird mit einem Sprung auf den Tisch $u = 1mm$ in Bewegung versetzt.

Folgende Ausgabe ergibt das Scope der Simulink-Schaltung nach 6s Simulation (obere Ausgabe für die obere und untere Ausgabe für die untere Tischplatte):

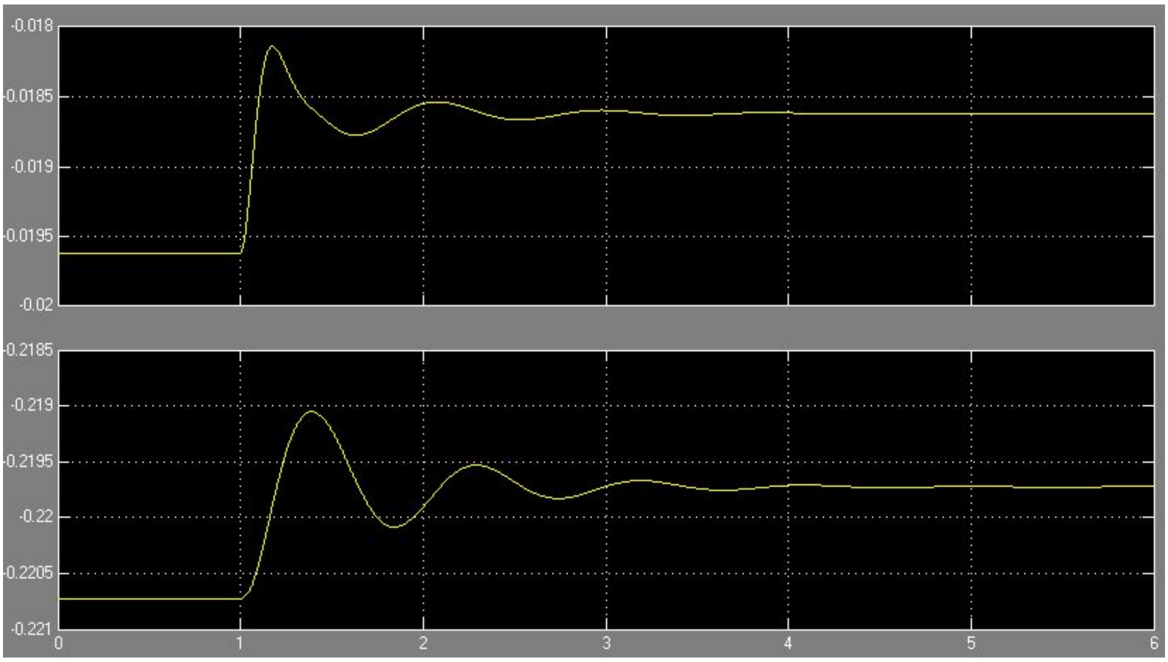


Abbildung 3.4: Scope-Ausgabe