Diskrete Regelung

Johannes Kasberger
Markus Klein
Simon Sperl

Juli 2011

1 Füllstandsregelung eines Viertanksystems

Diese Aufgabe basiert auf dem Buch *Modellbildung und Simulation dynamischer Systeme* von Helmut E. Scherf.

Es ist folgender Aufbau gegeben: Vier Tanks sind miteinander durch Röhren verbunden. Ein Motor pumpt Wasser in den ersten Tank und das Wasser fließt im letzten Tank ab. Der Querschnitt der Röhren ist grundsätzlich einstellbar, bleibt jedoch während des Betriebs konstant. Das Ziel ist es, den Füllstand im letzten Tank auf eine gewünschte Höhe zu regeln. Ein kontinuierlicher Regler der diese Aufgabe erfüllt, ist im Buch Modellbildung und Simulation dynamischer Systeme von H.E. Schärf beschrieben. Unsere Aufgabe ist es dieses kontinuierlichen Regler in einen diskreten Regler umzuwandeln. Das Model der Tanks haben wir vom Buch übernommen und in einen Simulink Block gekapselt.

Der Regler ist im Buch als PID Regler ausgeführt. Um daraus einen diskreten Regler bauen zu können, muss die Übertragungsfunktion diskretisiert werden. Das geschieht mit der Matlab Funktion c2d. Wenn man die Übertragungsfunktion eines idealen Reglers so transformiert, funktioniert der digitale Regler aber nicht richtig. Es muss noch eine parasitäre Zeitkonstante zum kontinuierlichen Regler hinzugefügt werden (somit wird er zum realen PID Regler) und dieser kann dann diskretisiert werden. Diese Rechnung passiert in der Datei *koef.m* . Darin werden auch alle benötigten Konstanten definiert, daher muss diese Datei ausgeführt werden um das Simulink Model lauffähig zu machen.

Listing 1: koef.m für Viertanksystem

```
% Koef. fuer Simulation des Viertanksystems
g=9.81;
A=0.00278;
A12=18.54e-6;
A23=17.5e-6;
A34=18.75e-6;
A4=13.44e-6;
hmax=0.338;

T_a = 1;
steps = 1800/T_a;

T_n=127;
T_v=12.7;
K_p=5;

% Koef. fuer Diskreten Regler
T_1 = T_n/2 + sqrt(T_n^2/4 -T_n*T_v);
T_2 = T_n/2 - sqrt(T_n^2/4 -T_n*T_v);
T_p = min(T_1,T_2)*0.05;

K_pid = K_p/T_n;
```

```
G_par = K_pid*tf([T_1*T_2  T_1+T_2  1],[T_p  1  0]);

H = c2d(G_par,T_a,'tustin');

b = cell2mat(H.num);
a = cell2mat(H.den);

set_param('diskr_matlab','IgnoredZcDiagnostic','none');
```

Um das Verhalten des diskreten und des kontinuierlichen Reglers vergleichen zu können, haben wir die Zwei gleichzeitig simuliert und die Ergebnisse in einem Plot dargestellt (Abbildung 1). Der Plot zeigt, dass das Verhalten der zwei Regler gut übereinstimmt. Die Abweichung der Stellgrößen der Regler beträgt maximal 3 und pendelt sonst immer um 0 (ist im mittleren Plot in Abbildung 1). Der Füllstand im oberen Plot und auch die Stellgrößen im unteren Plot sind weitgehend deckungsgleich.

Der diskrete Regler wurde in zwei Varianten realisiert. Einmal mit dem Simulink Block *Discrete Filter* (in Datei diskre.mdl realisiert). Die Koeffizienten für diesen Filter werden in der Datei koef.m berechnet und hier verwendet. Die andere Variante (in der Datei diskr_matlab.mdl zu finden, für Code siehe Listing 2) verwendet eine selbst geschriebene Embedded Matlab Function. Darin wird die Matlab Function *filter* verwendet, um die Inputdaten zu filtern. Die Koeffizienten sind die Gleichen wie bei der Simulink Variante. Auch diese Variante stimmt gut mit dem kontinuierlichen Regler überein.

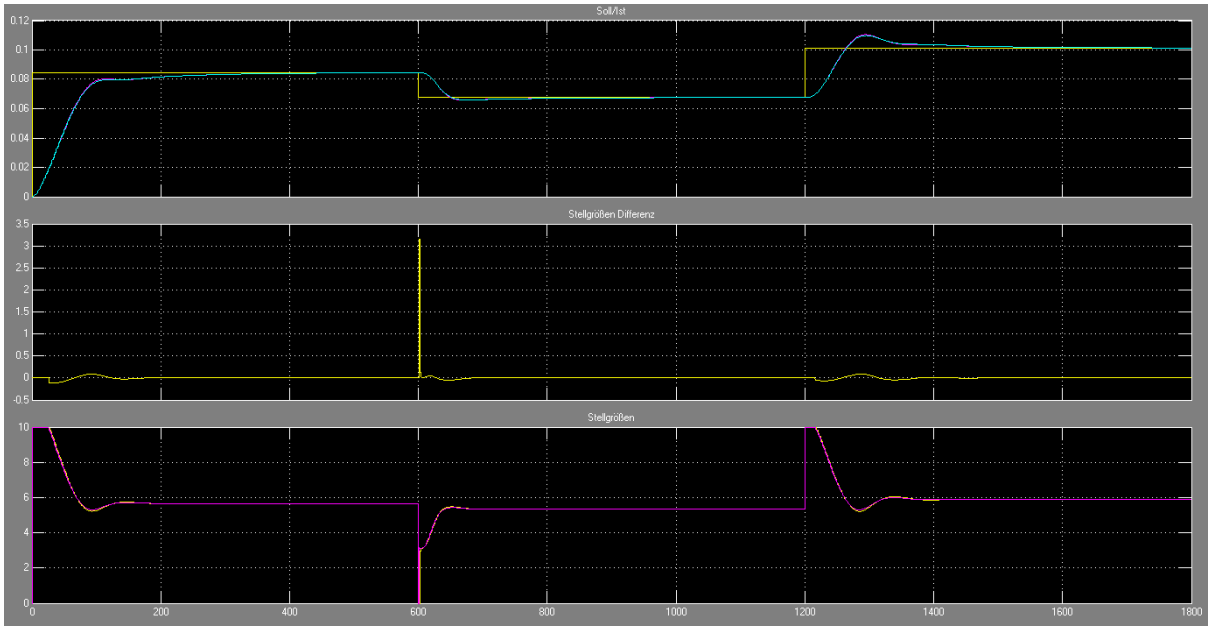

Abbildung 1: Vergleich Kontinuierlicher/Diskreter Regler für das Viertanksystem

Listing 2: Code für Matlab Function für Viertanksystem

```
% b,a sind die Koeffizienten fuer den Filter
% steps gibt an wieviele Simulationsschritte
% gemacht werden, wird benoetigt
% um Liste zu initialisieren
% soll und ist werden uebergeben
function stell = PID(ist,soll,b,a,steps) %#eml
    persistent X

    eml.varsize('X',[1  steps+1]);
```

```
% Differenz zwischen Soll und Ist Ausrechnen
diff = soll-ist;
diff = diff/0.0338;

% Am Anfang die History initialisieren
if isempty(X)
    X = diff;
% neue Werte hinzuf gen
else
    X = [X diff];
end;

res = filter(b,a,X);

% letztes Ergebnis des Filters ausgeben
% auf 10 limitieren
stell = min(res(end),10);
```

2 Aufheizen eines Werkstücks in einem Glühofen

Diese Aufgabe basiert auf dem Buch *Modellbildung und Simulation dynamischer Systeme* von Helmut E. Scherf.

Bei diesem Beispiel geht es um einen Glühofen in den ein Werkstück mit der Masse m und einer Ausgangstemperatur in den Ofen gelegt. Nun soll eine bestimmte Temperatur im Ofen erreicht und gehalten werden. Die Oberfläche A und der Wärmeübergangskoeffizient α des Werkstücks sind bekannt. Im Buch wird ausgeführt wie die Differentialgleichungen aufgestellt werden müssen. Dabei lassen sich alle Variablen die das Werkstück beschreiben in die Formel $T = \frac{cm}{\alpha A}$ zusammen fassen. Dieser Gleichung die die Temperatur (ϑ_G) im Ofen beschreibt ist $T * \dot{\vartheta_W} + \vartheta_W = \vartheta_G$. Die Temperatur des Werkstückes wird mit ϑ_W bezeichnet.

Diese Differentialgleichung wurde im Buch als kontinuierlicher Regler realisiert. Wie im ersten Beispiel auch haben wir auch hier die Übertragungsfunktion diskretisiert. Hier haben wir verschiedene Optionen der c2d Funktion ausprobiert und die Ergebnisse vergleichen (siehe Abbildung 2). Wir haben hier die Algorithmen Zero Order Hold, Triangle approximation und Impulse invariant discretization verwendet. Die Ergebnisse der verschiedenen Algorithmen sind in der Abbildung 3 zu sehen. Die Abweichung zum kontinuierlichen Regler ist mit allen Diskreten Reglern sehr klein.

Auch hier wurde der Regler einmal als Simulink und einmal als Embedded Matlab Function realisiert. Der Code für die Embedded Matlab Function ist im Listing 3 angegeben.

Listing 3: Code für Matlab Function für Glühofen

```
function tmp = PI(wert,T,step)
    persistent X

    if isempty(X)
        X = 0;
    end

    X = X + wert/T*step;
    tmp = X;
```

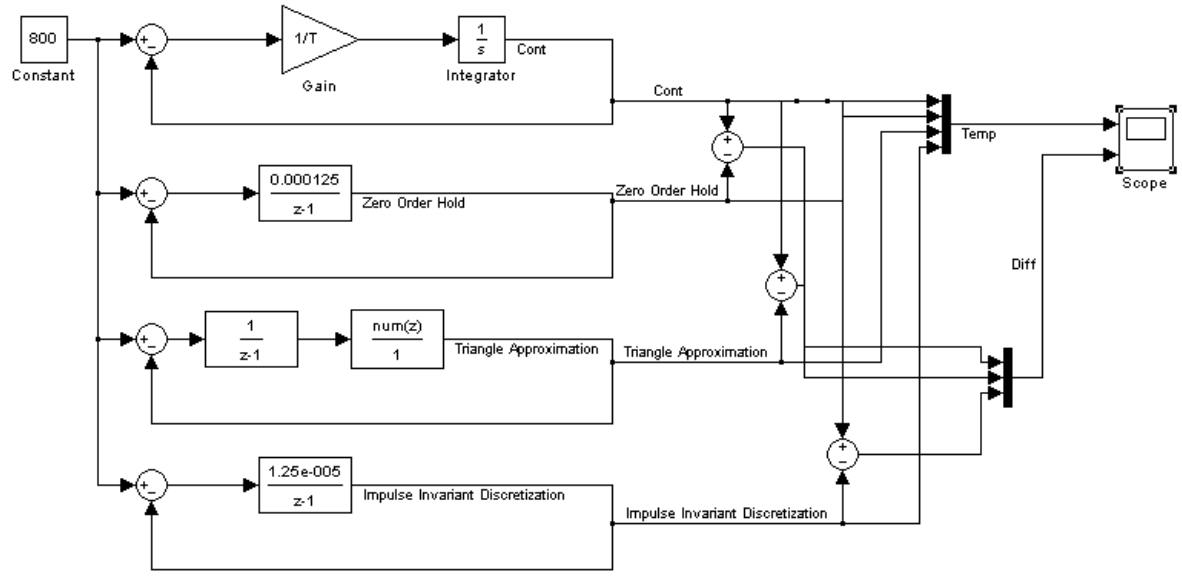

Abbildung 2: Implementierung der Regler für den Glühofen

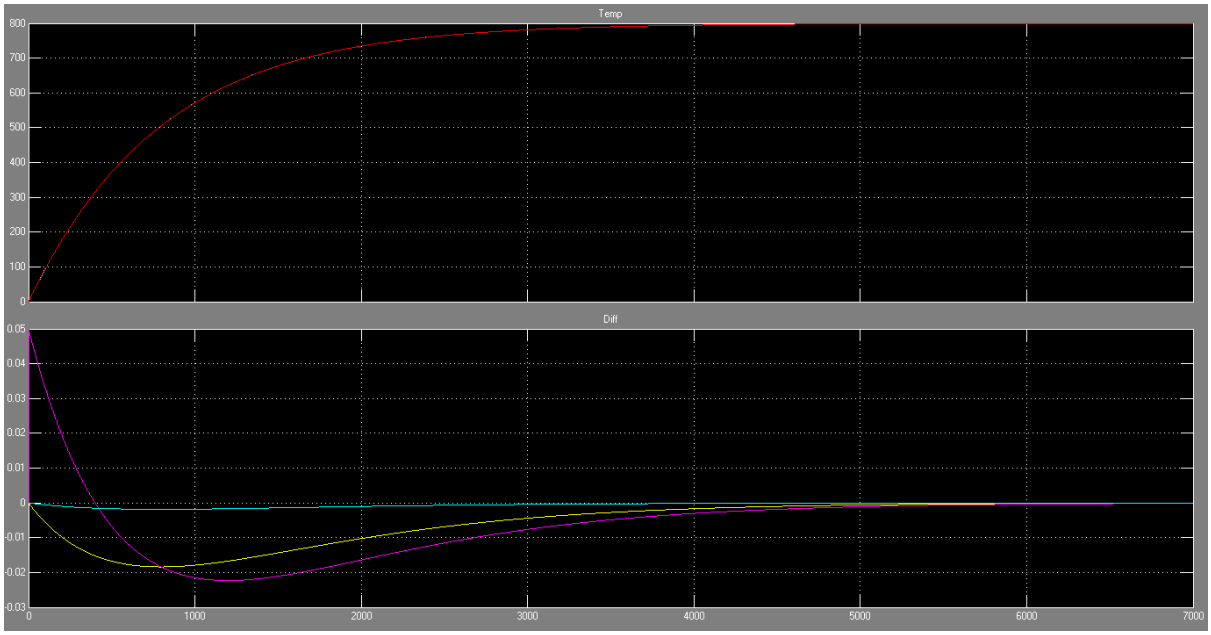

Abbildung 3: Vergleich Kontinuierlicher/Diskreter Regler für den Glühofen

3 Fahrradmodell

Aufgabe war die Implementierung und Regelung eines vereinfachten Fahrradmodells (Eyckhoff, 1974).

$$z_1 = Lenkausschlag \quad (1)$$
$$z_2 = Fahrerneigung \quad (2)$$
$$p = Fahrradneigung \quad (3)$$

$$dmv\dot{z_1} + mv^2 z_1 \quad = \quad J\ddot{p} - mhgp + (J * + h^2 m)\ddot{z_2} - 1/2mgz_2 \tag{4}$$

In Worten bedeuted die Formel:

Gyroskopisches Moment des Vorderrads + Zentrifugalmoment der Gesamtmasse= Trägheitsmoment des Fahrrads - Gravitationsmoment des Fahrrads + Trägheitsmoment des Fahrers - Gravitationsmoment des Fahrers

Für die Parameterstudie verwenden wir realistische Werte:

g	v	m_{bike}	m_{biker}	h	
$9.81m/s^2$	$3m/s$	$5kg$	$50kg$	$0.9m$	siehe *koeff.asv*

Während der Studie war es wichtig zu beachten das für z_i, z_2, p, wegen der mehrmals gewählte Vereinfachung $sin(z_1) \approx z_1$, nur Werte im Bereich $[-\frac{\pi}{4}, \frac{\pi}{4}]$ angenommen werden. Für die Simulation bedeuted dies; wenn Fahrer oder Fahrrad zu stark geneigt sind, dann ist kein realistisches Verhalten mehr zu erwarten.

Die Angabe verwendet nun zwei PID Regler zur Steuerung von z_1 und z_2 um p wieder in Ausgangslage zu bringen. Persönlich würde eine Regelung von z_2 abhängig von Lenkausschlag z_1 und Momentangeschwindigkeit v die es vermeidet das *man hinfällt*. Im Sinne von Je schneller ich fahre und je enger die Kurve desto mehr muss ich mich reinlegen.

Bei einer kontinuerlichen Störung des Lenkauschlags z_1 mittels einer schwachen Sinusschwingung Amplitude $0.05rad$ und Frequenz $0.2Hz$, und allen Parametern der PID-Regler auf 0.0001 bleibt das Fahrrad stabil. Wobei sich Ki im Bereich von 0.05-0.0001 bewegen darf, die Kp und Kd sind wesentlich empfindlicher. Die Vereinfachung von $sin(x) = x$ f¸r kleine x hat sich f¸r die Parameterstudie als stark problematisch herausgestellt, da sie den Übergang zwischen etwas falschen und völlig falschen Parametern stark verkleinert.

Das Ergebnis des Parameterstudie ist im großen und ganzen schön gleichmäsige Störungen und ganz vorsichtig den Körper neigen dann geht im Modell alles gut.

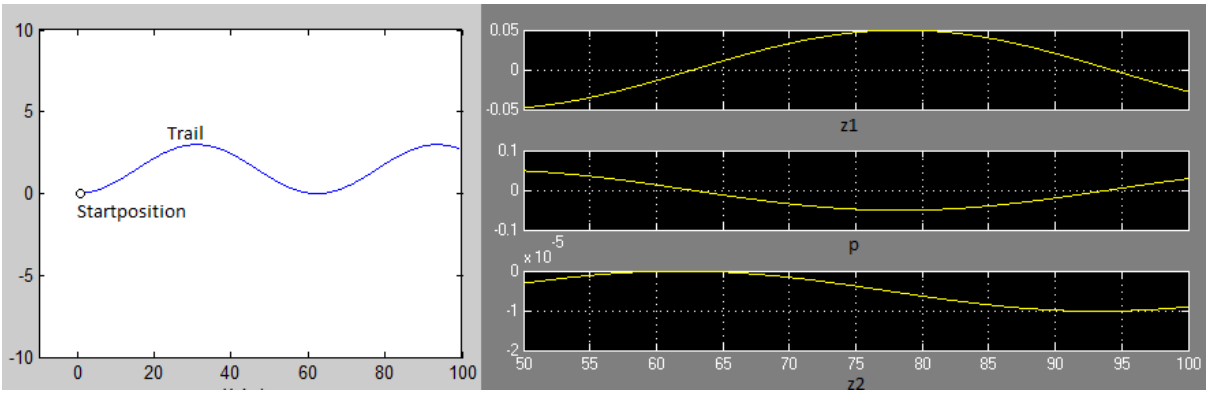

Abbildung 4: Testfahrt, mit sinusartig störschlingenden Lenker, *siehe Fahrrad2.sim*

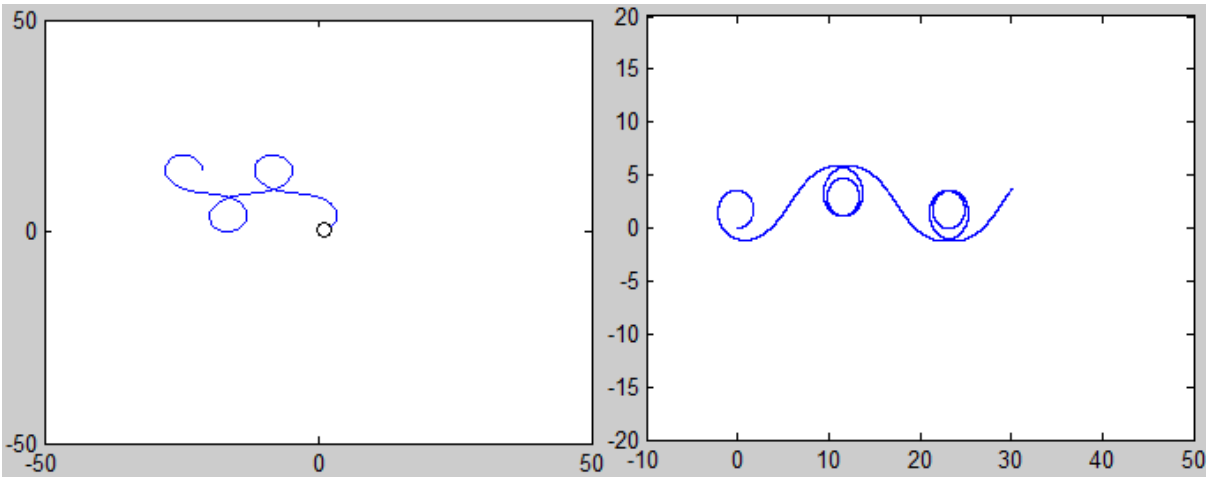

Abbildung 5: Hübsche Fahrradfahrten, erzeugt durch veränderte Frequenzen und Amplituden der Störschwingungen