Trabalho Final
EEE038 - LABORATÓRIO DE CONTROLE AUTOMÁTICO II
"MECATRÔNICA'

Prof. Fernando Passold

6 de junho de 2014

1 Introdução

A idéia é modificar o firmware do Processo da Bola no Tubo acrescentando novas funcionalidades ao seu código original. Especificamente:

1. Implementar PID no formato "paralelo";
2. Implementar filtro derivativo para controlador PD e PID.

Os alunos se organizam em equipes para realizar os trabalhos:

Equipe	Trabalho	Alunos
1	Implementação PID "paralelo"	DIEGO ARRUDA DA SILVEIRA JOEL RODRIGUES DA SILVA
2	Implementação Filtro Derivativo	DIEGO ZAMBIASI JULIANO TEOBALDO MULLER DE LIMA

Deadline (data de entrega prevista para o trabalho): 27/06/2014.

2 PID formato "paralelo"

A idéia aqui é modificar o algoritmo PID de velocidade originalmente implementado no processo da Bola no Tubo para que outra arquitetura, a do PID "paralelo".

O PID originalmente implementando apesar de permitir o ajuste "individual" dos parâmetros: K_c, T_i e T_d foi implementado de tal forma (arquitetura ISA) que uma alteração no parâmetro K_c implica sutil alteração (dentro da equação de diferenças que corresponde à esta lei de controle) a mudanças na ação Integrativa e Derivativa, uma vez que este parâmetro age também sobre a equação dos termos relacionados individualmente com a parte Integrativa e Derivativa.

A idéia é propor e implementar o PID no formato paralelo onde uma alteração em qualquer um de seus parâmetros: K_c, T_i ou K_i e T_d ou K_d não implique influências em outras ações do controle PID. Isto é:

- Alterar $K_c \longrightarrow K_p$: implica somente alterar o ganho proporcional que age sobre o erro do sistema em malha-fechada;
- Alterar T_i ou $K_i \longrightarrow$ implique somente modificar o ganho da parte Integrativa da ação do PID;
- Alterar T_d ou $K_d \longrightarrow$ implique modificar apenas o comportamento da parte Derivativa do PID.

Para tanto, a equipe pode partir da lei de controle PID tradicionalmente expressa no plano-s e usar o método de Tustin para obter a correspondente expressão no plano-z (e a partir desta expressão obter a correspondente equação de diferenças) ou pode-se (usando as tradicionais relações de Euler – aproximações numéricas para integrais e derivadas) para a correspondente equação de diferenças que corresponda ao PID no formato "paralelo". Ressalta-se que neste formato, os termos K_c (ou K_p), T_i (ou T_i) e T_d (ou K_d) devem aparecer de forma isolada uns dos outros na equação de controle, desta foram, para cada ação do controlador PID: Proporcional + Integral + Derivativo, corresponderá um termo de ajuste que não influencia (quando modificado) as outras ações.

Comumente a função transferência contínua do controlador PID ideal é dada por:

$$C(s) = K_p + \frac{K_i}{s} + K_d s \tag{1}$$

Existem várias maneiras para mapear do plano-s para o plano-z. A maneira mais precisa é através da definição da transformada Z:

$$z = e^{Ts} \tag{2}$$

Mas é impossível se obter a função transferência discreta do controlador PID desta forma, pois o grau do polinômio do numerador resulta maior que o grau do polinômio do denominador o que implica numa equação de controle não realizável (surgem termos futuros na equação de diferenças). Então pode-se empregar a seguinte transformação bilinear:

$$s = \frac{2}{T\,s} \cdot \frac{(z-1)}{(z+1)} \tag{3}$$

Este método de transformação bilinear é conhecido como método de Tustin. O método de Tustin pode ser acessado no MATLAB usando-se o comando `c2d(·)`, ver exemplo à seguir:

```
>> Kp = 100; % segue-se exemplos arbitrarios para Kp, Ki e Kd
>> Ki = 200;
>> Kd = 10;
>> num = [ 1 0 ]; % polinomio no plano-s que corresponde ao numerador do PID
>> den = [ Kd Kp Ki ]: % polinomio no plano-s que corresponde ao denominador do PID
>> C = tf (num,den) % cria um objeto transfer funcition continuo no MATLAB
C =
            s
  ---------------------
  10 s^2 + 100 s + 200
Continuous-time transfer function.
>> T = 0.047; % especifica o periodo de amostragem a ser adotado (em segundos)
>> Cd=c2d(C,T,'tustin') % determina C(z) - versao discreta do PID
Cd =
  0.001886 z^2 - 0.001886
  -----------------------
  z^2 - 1.587 z + 0.6228
Sample time: 0.047 seconds
Discrete-time transfer function.
>> zpk (Cd) % apresenta C(z) fatorado (polos e zeros separados)
ans =
  0.001886 (z+1) (z-1)
  --------------------
  (z-0.878) (z-0.7093)
Sample time: 0.047 seconds
Discrete-time zero/pole/gain model.
>>
```

3 Filtro Derivativo

A idéia aqui é melhorar o comportamento do PID atuando sobre o processo da bola no tubo – ver figura 1.

Para tanto, a idéia é agir somente sobre a ação Derivativa do controlador PID incorporando um filtro para a parte relacionado com o cálculo da derivada do erro – ver figura 2.

Para tanto, um simples filtro Passa-Baixa de primeira ordem permite acrescentar esta funcionalidade ou pode-se usar a seguinte aproximação:

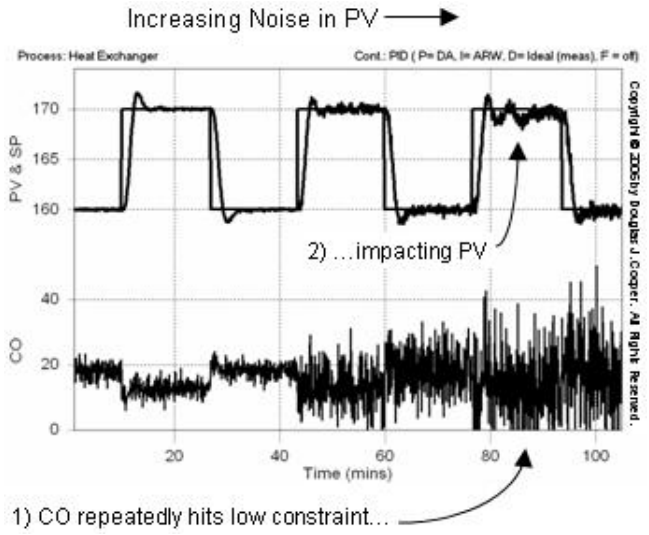
$$D(s) = \frac{K_c\,T_d\,s}{1 + \frac{T_d}{N}s} \tag{4}$$

onde $3 \leq N \leq 20$. Normalmente se emprega $N = 10$. Com isto, o ganho em baixas frequências (para a parte derivativa da ação de controle PID, $D(s)$) é praticamente mantido inalterado, mas em altas frequências (quando $s \to \infty$) fica limitado à $K_c\,N$ [1](pág. 7–8).

De todas as formas, não importa a forma como o filtro derivativo passa-baixa de 1a-ordem foi obtido, chega-se a uma equação do tipo:

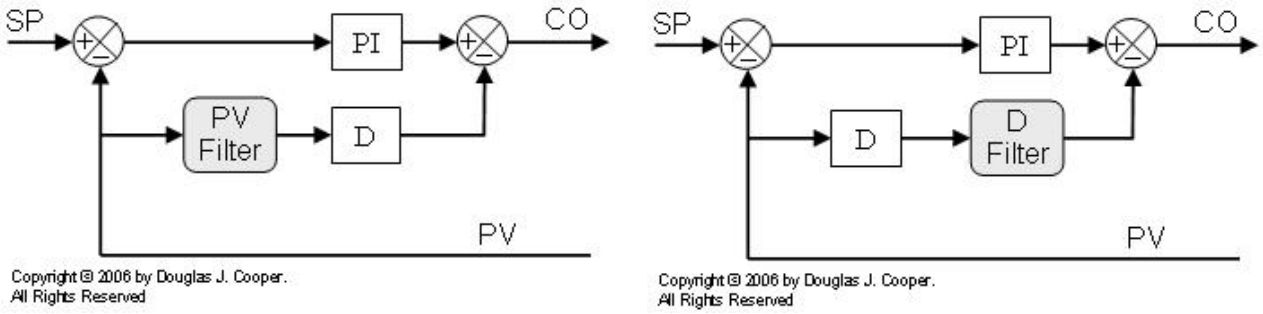
$$D[k] = a_i\,D[k-1] + b_i\,(y[k] - y[k-1]) \tag{5}$$

onde o que muda é o valor dos coeficientes a_i e b_i. Caon [1](pág. 15) realiza alguns comentários relacionados a forma como foi obtida a equação de diferenças da ação derivativa. Sugere-se que a equipe use o método de Tustin (como mostrado no item anterior) para eventualmente obter a correspondente equação de diferenças a



Onde: PV = *Process Variable* (saída do processo), CO = *controller output* (sinal de controle).

Figura 1: Impacto do ruído numa ação de controle PID (extraído de [2]).



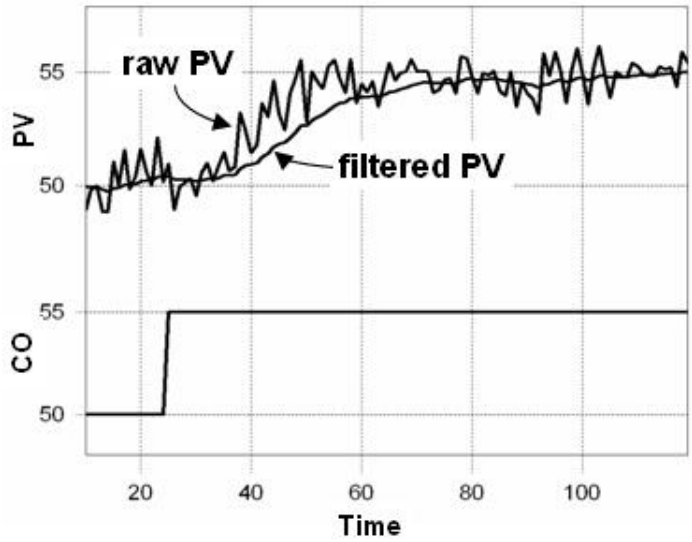
(a) Pré-Filtro. (b) Pós-Filtro.

Onde: SP = *set point value* (Referência), PV = *Process Variable* (saída do processo), CO = *controller output* (sinal de controle).

Figura 2: Introdução de Filtros Derivativos numa malha de controle (extraído de [2]).

partir da equação do filtro no plano-s.

Mais informações sobre Filtros Derivativos pode ser obtido em: [2](item Using Signal Filters In Our PID Loop: `http://www.controlguru.com/wp/p82.html`) – ver figura 3.



Onde: PV = *Process Variable* (saída do processo), CO = *controller output* (sinal de controle).

Figura 3: Comportamento de um filtro derivativo (extraído de [2].

Na prática de trabalhos com este kit, já foi percebido que a simples introdução de um filtro passa-baixa de 1a-ordem com frequência de corte em 0,6 Hz já permite alcançar resultados interessantes – ver figura 4.

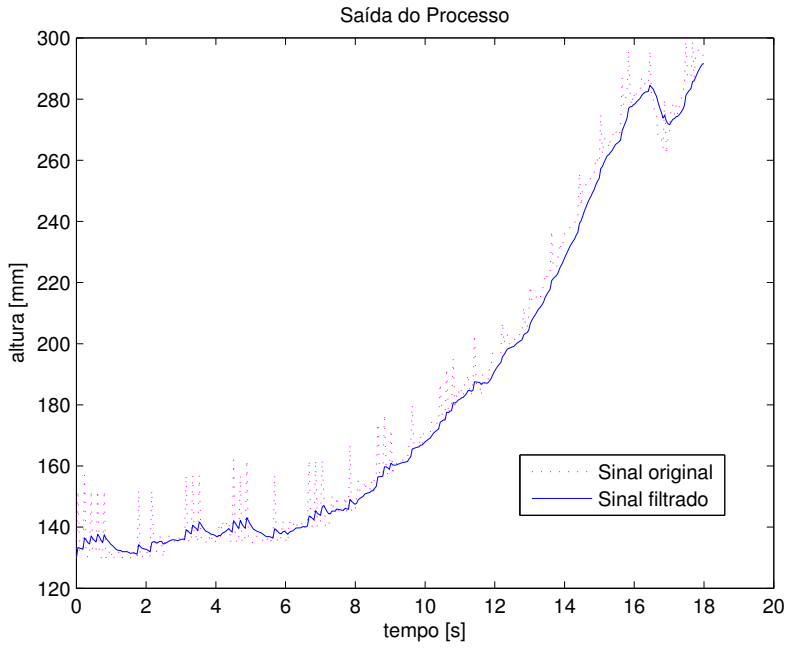


Figura 4: Exemplo de atuação de filtro com $f_c = 0,6$ (Hz) aplicado no processo da Bola no Tubo.

4 Outras Referências

Outras referências para implementação dos algoritmos exigidos neste trabalho podem ser obtidas em: [2], [3],

A Filtro Passa Baixa Digital

Um filtro PB de 1a-ordem no formato digital é dado por:

$$y[k] = \alpha\, x[k] + (1-\alpha)\, y[k-1] \tag{6}$$

onde: $y[k]$=saída atual do filtro; $y[k-1]$=amostra atrasada da saída do filtro (de um período de amostragem), $x[k]$=sinal de entrada (sem atraso); α=fator de amortecimento ($0 \leq \alpha \leq 1$) e onde α é calculado como:

$$\alpha = \frac{T_s}{(RC + T_s)}$$

onde: T_s=valor do período de amostragem adotado e RC=constante de tempo do filtro (a mesma que seria usada num filtro PB analógico formado por um circuito TC). Este tipo de filtro é dito também de média móvel de ponderação exponencial.

Este filtro (eq. 6) no formato de um algoritmo para MATLAB (`lowpass.m`) fica:

```
% Filtro passa-baixa de 1a-ordem
% Media movel de ponderacao exponencial (EWMA)
%
% function y=lowpass(x,Ts,RC)
%
% Parmetros de entrada:
%   RC=1/(2*pi*fc)=constante de tempo do filtro
%   Ts=periodo de amostragem adotado (em s)
% onde:
%   fc=freq. de corte do filtro (em Hz)
% Internamente ser calculado 'alpha':
% alpha = Ts/(RC+Ts) = parmetro do filtro
%   importante que Ts<= RC/5, seno este filtro apresenta
% um comportamento muito diferente do esperado.
function y=lowpass(x,Ts,RC)
    alpha=Ts/(RC+Ts)
    fprintf('RC/Ts=%7.4f (esperado >= 5)\n',RC/Ts);
    if (Ts>(RC/5))
        fprintf('Warning: the filter may behave quite differently from the original continuous-time
            filter\n');
    end
    y(1)=x(1);
    amostras=length(x);
    for i=2:amostras
        % y(i)= alpha*x(i) + (1-alpha)*y(i-1); % eq. (1)
        y(i)= y(i-1) + alpha*(x(i) - y(i-1)); % eq. (2)
    end
% return y
```

lowpass.m

A equação (6) pode ser re-escrita no formato:

$$y[k] = y[k-1] + \alpha \cdot (x[k] - y[k-1]) \tag{7}$$

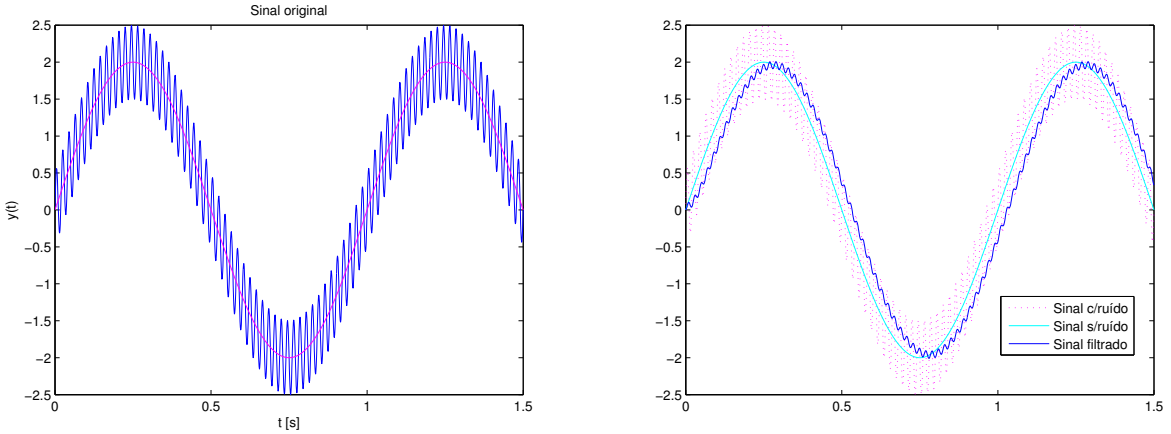
Realizando um teste: suponha que o sinal a ser filtrado seja dado por: $y(t) = 2\,\sin(2\,\pi\,1) + 1/4\sin(2\,\pi\,50)$ - conforme mostra a figura 5(a), ou seja, o sinal principal oscila a 1 Hz enquanto sobreposto a este sinal há um ruído de25% de amplitude e frequência de 50 Hz. Se for aplicado um FPB com $f_c = 5$ [Hz], no formato digital trabalhando com período de amostragem de $T_s = 1$[ms] ($f_s = 1$ KHz), obteremos algo como mostrado na figura 5(b). Aplicando esta informação sobre a rotina de teste mostrada abaixo (`sinal.m`), obtêm-se o resultado mostrado na figura 5(b).

```
% sinal.m
% montando sinal x(:,1) no tempo para teste do filtro PB
% entradas:
%   Ts= periodo de amostragem
%    f = frequencia base da senoide
clear x
i=0; % contador do numero da amostra
f=1;
T=1/f;
for k=0:Ts:(1.5*T) % gera 2,5 perodos
    i=i+1;
    x(i,1)=k; % varivel tempo
    x(i,2)=2*sin(2*pi*1*k)+0.5*sin(2*pi*50*k); % sinal base + "rudo"
    x(i,3)=2*sin(2*pi*1*k); % sinal base
end
figure; plot(x(:,1),x(:,2),'b-', x(:,1),x(:,3),'m-')
```

sinal.m

Comandos usados no MATLAB:

```
>> Ts=1e-3;
>> sinal
>> RC=1/(2*pi*5)
RC =
    0.0318
>> y=lowpass(x(:,2),Ts,RC);
alpha =
    0.0305
RC/Ts=31.8310
>> figure; plot(x(:,1),x(:,2),'m:', x(:,1),x(:,3),'c-', x(:,1),y,'b-')
>> legend('Sinal c/ruído','Sinal s/ruído', 'Sinal filtrado')
```



(a) Sinal original.

(b) Sinal filtrado.

Figura 5: Teste do filtro digital FPB.

Note pela figura 5(b) que como a frequência do ruído está uma década acima da frequência de corte do filtro, sua amplitude decaiu de 20 dB ou seja de 1/10.

Referências

[1] CAON JUNIOR, José Roberto. Controladores PID industriais com sintonia automática por realimentação a relê. 1999. Dissertação (Mestrado em Engenharia Elétrica) - Escola de Engenharia de São Carlos, Universidade de São Paulo, São Carlos, 1999. Disponível em: `http://www.teses.usp.br/teses/disponiveis/18/18133/tde-04042001-172532/`. Acesso em: 2014-06-06.

[2] Cooper, Doug J., Practical Process Control: Proven Methods and Best Practices for Automatic PID Control, `http://www.controlguru.com/pages/table.html`. Acesso em 2014-06-06.

[3] BERTO, Maria Isabel; SA, Fabiana Rodrigues de and SILVEIRA JR., Vivaldo. Avaliação de controles PID adaptativos para um sistema de aquecimento resistivo de água. Ciênc. Tecnol. Aliment. [online]. 2004, vol.24, n.3, pp. 478-485. ISSN 1678-457X. `http://dx.doi.org/10.1590/S0101-20612004000300030` ou `http://www.scielo.br/pdf/cta/v24n3/21946.pdf`. Acessado em 2014-06-06.