DINCON'10
9th Brazilian Conference on Dynamics,
Control and their Applications
June 07-11, 2010

LABORATÓRIO DE CONTROLE USANDO O AMBIENTE TEMPO REAL DO MATLAB PARA CURSOS DE GRADUAÇÃO

Manoel L. Aguiar, Vilma A. Oliveira, Alessandro R. Locatti, César Domingues

Universidade de São Paulo, São Carlos, SP
aguiar, vilma, arlocati, cesard@sc.usp.br

RESUMO: Neste trabalho, material didático para o projeto e a implementação em tempo real de controladores a partir de modelos Simulink para ser utilizado em disciplinas básicas de laboratório de controle é fornecido. A praticidade do ambiente de implementação de controladores digitais com o Simulink para ensinar controle realimentado é demonstrada. A ênfase é dada na conexão da planta a ser controlada com o modelo Simulink. O experimento com uma planta real é adequado para ser dado no 3o. ano de cursos de engenharia auxiliando o aluno a compreender os componentes de hardware e software de um sistema de controle realimentado. Um motor de corrente contínua (CC) juntamente com um tacogerador como transdutor comumente encontrados em laboratórios de graduação são usados para ilustrar o ambiente de implementação.

Palavras-Chave: Ensino de controle, tempo real, automação.

1. INTRODUÇÃO

Disciplinas de laboratório de controle fazem parte do currículo de diversos cursos de engenharia e tem como objetivo fixar conceitos aprendidos de análise de sistemas de controle e implementar controladores básicos. Controladores são utilizados em diversas áreas como robótica, automação de processos industriais, indústria aero espacial e em sistemas de transportes [1]. A implementação de controladores digitais em tempo real é uma tarefa que pode levar muito tempo devido à geração e compilação de códigos executáveis, e comunicação com a planta física a ser controlada [2]. Para facilitar a implementação em tempo real de controladores, o Real-Time Workshop (RTW) e o Real-Time Windows Target (RTWT), módulos da The Mathworks podem ser usados [3, 4]. Usando modelos Simulink, o RTW fornece um ambiente para automaticamente gerar e compilar códigos executáveis e o RTWT para prototipar e testar o controlador ligado à aplicação em tempo real. O RTWT e o RTW substituem o uso de um programa para gerar e executar um código em C ou outra linguagem de baixo nível para automação.

Para facilitar a utilização do material apresentado, os principais diagramas Simulink e circuitos construídos são apresentados.

2. DESCRIÇÃO DO AMBIENTE DE TEMPO REAL

O ambiente de tempo real para implementação de controladores é formado pelos softwares Matlab, Simulink, Real-Time Workshop e o Real-Time Windows Target da The Mathwork. A planta a ser controlada é um motor CC conectado a um tacogerador com acionamento feito por um amplificador de potência e um dispositivo de aquisição de dados multi função da National Instruments. A Figura 1 ilustra a composição da bancada de experimento disponível no Laboratório de Ensino de Controle e Robótica do Departamento de Engenharia Elétrica da USP São Carlos.

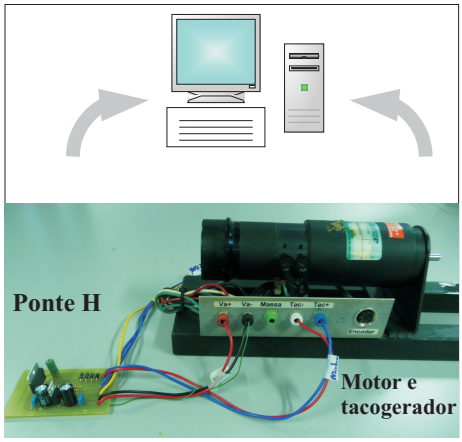


Figura 1 – Componentes da bancada de experimento.

O experimento começa com a simulação do sistema realimentado com o modelo do motor para testar o desempenho do controlador projetado. Em seguida, o código executável é gerado com o compilador Open Watcom C/C++ e o contro-

lador pode então ser implementado usando o modo externo do Simulink. A integração do modo externo do Simulink com o RTWT possibilita o uso do diagrama Simulink como uma interface gráfica para o usuário.

O RTWT forma um ambiente para tornar o PC um hospedeiro e um alvo. Com o RTWT, sinais de plantas reais são enviados para o modelo Simulink onde são processados e retornados à planta física através de uma interface [3].

2.1. Configurando o kernel

O RTWT trabalha com um kernel de tempo real para comunicar com o sistema operacional Windows a uma taxa de amostragem selecionada. Com o kernel, o código executável tem prioridade sobre outras tarefas do sistema operacional.

Para fazer a comunicação entre Simulink e a planta, o Simulink deve ser configurado no modo *External* para passar parâmetros para a planta física e receber em tempo real a sua saída que pode ser visualizada ou armazenada em arquivo. Este modo externo interage com kernel e é usado para começar a aplicação de tempo real, mudar parâmetros do modelo Simulink e recuperar dados do visualizador.

Durante a execução em tempo real, o kernel irá interferir quando necessário para garantir que o modelo Simulink tenha prioridade no uso da CPU quando ocorre uma atualização. Quando a atualização estiver completa, o kernel libera a CPU para outras tarefas. É necessário a instalação do Real-Time Windows Target kernel para garantir prioridade de execução para o arquivo executável em tempo real. Para a instalação do kernel, os seguintes passos devem ser seguidos.

```
>>rtwintgt -install
```

a mensagem aparecerá:

```
You are going to install the Real-Time Windows
Target kernel. Do you want to proceed? [y] :

>> y
```

e uma mensagem de confirmação aparecerá:

```
The Real-Time Windows Target kernel has been
successfully installed.
```

Reinicie seu computador. Para conferir se o kernel foi instalado corretamente, digite

```
<<rtwho
```

Se o kernel estiver instalado corretamente, informações sobre o RTW e sobre o kernel serão dadas:

```
Real-Time Windows Target version 3.2.0 (C)
The MathWorks, Inc. 1994-2008
Running on Multiprocessor APIC computer.
MATLAB performance = 100.0}
Kernel timeslice period = 1 ms
```

O kernel só precisa ser instalado uma única vez. Uma vez instalado, se torna ativo na execução do modelo Simulink com recursos do Real Time Workshop e é desativado no término da execução.

2.2. Configurando e executando no ambiente Simulink

No menu do modelo Simulink escolher 'configuration parameters' então 'hardware implementation' e selecionar '32-bit Real-Time Windows Target'. Novamente, no menu 'configuration parameters' selecionar 'Real Time Workshop' e escolher rtwin.tlc. No menu Tools selecionar 'Real Time Workshop' então 'build model' e o 'Real-Time Workshop' gera os arquivos fontes de códigos C rtwin-model.c e rtwin-model.h. Para a implementação das entrada/saida escolher 'Simulation' então 'External' então 'Connect to target'. Finalmente, do menu do modelo Simulink escolher 'Start real-time code' e o Scope mostra o sinal de saida com a aparência da Figura 2. O sinal de saída mostrado é a tensão do tacogerador obtida para uma entrada ao degrau de 12V a ser usada na identificação do modelo da planta.

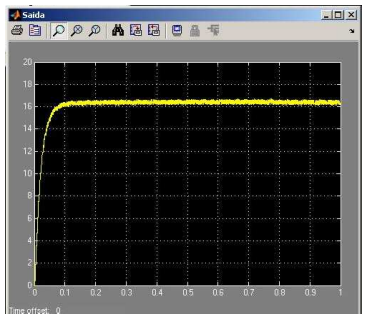


Figura 2 – Exemplo de um sinal de saída mostrado na ambiente Simulink.

3. IDENTIFICAÇÃO DO GANHO E CONSTANTES DE TEMPO DO MOTOR CC

Os parâmetros de plantas simples descritas por modelos entrada-saída podem ser obtidos a partir de respostas a entradas típicas. A função de transferência para um motor CC com entrada a tensão aplicada na armadura v_a em volts e saída a velocidade w em rad/s descrito pelo diagrama de blocos da Figura 3 é da forma

$$\frac{w(s)}{v_a(s)} = \frac{K_m}{(s+\gamma)(s+\delta)} \tag{1}$$

onde

$$K_m = \frac{1}{K_e K_t \tau_m \tau_a}, \gamma + \delta = \frac{1}{\tau_a} + \frac{1}{\tau_b} \text{ e } \gamma\delta = \frac{1}{\tau_m \tau_a} + \frac{1}{\tau_a \tau_b}$$

com

$$\tau_a = \frac{L}{R}, \tau_b = \frac{J}{B} \text{ e } \tau_m = \frac{RJ}{K_e K_t}.$$

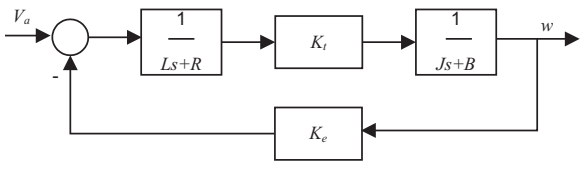


Figura 3 – Diagrama de blocos do modelo do motor CC.

Identificar o motor CC resume-se então na obtenção das constantes de tempo γ e δ e ganho constante K_m. A partir da resposta à um degrau de tensão aplicado na armadura e

a interface gráfica do Matlab denotada 'ident'. A resposta é obtida em volts como a saída de um tacogerador acoplado. Para converter a resposta para rad/s deve-se obter o modelo do tacogerador representado por uma constante denotada K_{tg}. Assim, tem-se $w = \frac{V_{tg}}{K_{tg}}$.

As linhas de comando para definir os dados de entrada e saída para a interface 'ident' CC são mostradas a seguir.

1. Carregar os dados do ensaio degrau de tensão [t,V_{tg}]

```
>>load dados.dat;
>>Ktg=0.15;
>>w=dados(:,2)/Ktg;
>>T = 1/2000; %taxa de amostragem
```

2. Definir o objeto motor

```
>>motor = iddata;
>>motor.Tstart = 0;
>>motor.T = T [s];
>>motor.InputData=12*ones(size(dados(:,1));
>>motor.OutputData = w [rad/s];
```

3. Chamar a interface gráfica ident

```
>>ident
```

4. Importar dados via objeto 'motor', escolher modelo e executar o comando 'Estimate'
5. Digitando P1 na área de trabalho aparece

```
>> P1

Process model with transfer function

             K
G(s) = ------------------
       (1+Tp1*s)(1+Tp2*s)

with    K = 1.2077
      Tp1 = 0.0432
      Tp2 = 0.0010

Estimated using PEM from data set motor
```

6. Salvar a seção como motorid.sid
7. Para chamar a seção salva previamente digitar

```
>>ident('motorid.sid')
```

4. SISTEMA DE CONTROLE

4.1. Projeto via rltool

O método do lugar das raízes (LR) pode ser utilizado para o projeto de controladores. O controlador largamente utilizado em processos industriais é o controlador PID descrito por

$$\frac{u(s)}{e(s)} = K(s) = K_p + \frac{K_I}{s} + K_D s \qquad (2)$$

O controlador fornece um termo proporcional e um termo integrativo e um termo derivativo. O efeito individual dos termos K_P, K_I e K_D são resumidos na Tabela 1. A função de transferência $K(s)$ não é realizável e descreve o PID ideal. O PID implementável possui um segundo polo para filtrar a ação derivativa e para evitar a amplificação de ruído de alta frequência.

Tabela 1 – Efeito individual dos termos do PID na resposta

	Tempo de subida	Sobresinal	Tempo de resposta	Erro	Estabilidade
Aumento K_P	Diminui	Aumenta	Aumenta pouco	Diminue	Piora
Aumento K_I	Diminue pouco	Aumenta	Aumenta	Diminue muito	Piora
Aumento K_D	Diminue pouco	Diminue	Diminui	Pouco muda	Melhora

Tabela 2 – Tabela verdade para as entradas da ponte H

Entradas		Saídas rotação motor CC
$V_{enable} = H$	$C = H\ D = L$	Horário
	$C = L\ D = H$	Anti-horário
	$C = D$	Não ocorre
$V_{enable} = L$	$C = X\ D = X$	Roda livre

L=Baixo H=Alto X=Qualquer

Em geral o controlador PID é muito útil para zerar o erro de regime permanente e melhorar a resposta transitória. A seleção dos três coeficientes do PID é basicamente um problema de busca em um espaço tri-dimensional. Pontos no espaço de busca correspondem a diferentes escolhas dos termos do PID. O problema é como relacionar este termos com as especificações das propriedades de robustez desejadas.

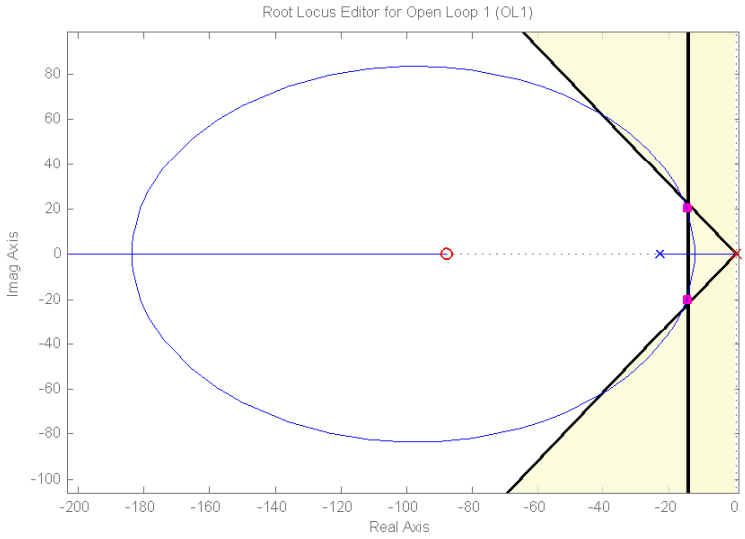
O procedimento via o método do lugar das raízes envolve:

1. Selecionar a freqüência natural w_n e coeficiente de amortecimento ξ através da especificação do tempo de acomodação T_s e sobresinal do modo dominante desejado.
2. Usar a interface gráfica $rltool$ para encontrar os zeros do compensador para atender a especificação de sobresinal e tempo de acomodação desejados para uma entrada degrau, numa forma de tentativa e erro para considerar a entrada do PWM no intervalo $[0\ 1]$ e efeito dos zeros do PID.

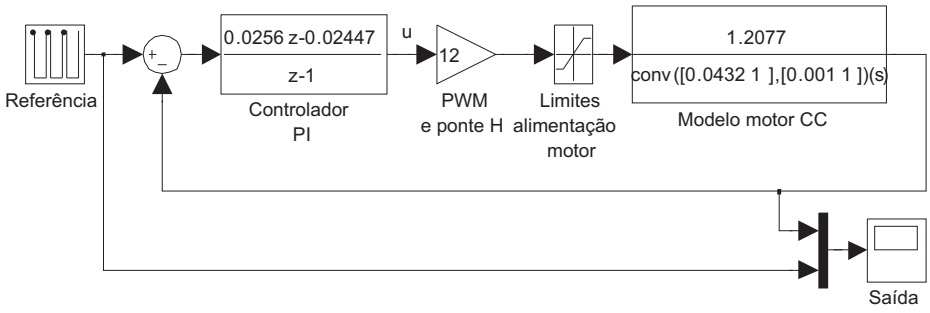
O Simulink possui uma função PID que pode ser utilizada para simular o sistema compensado. Esta função pode ser encontrada em *Simulink Extras/Additional Linear.* A Figura 4(a) ilustra o uso da interface $rltool$ para obter os parâmetros do controlador PI utilizado e a Figura 4(b) apresenta o diagrama de simulação do sistema realimentado.

4.2. Implementando o sistema de controle

O sistema de controle é implementado com um acionamento via uma ponte H. A Tabela 2 apresenta a tabela verdade para a entrada 'enable' da ponte H com o componente L298 [5] implementada como na Figura 5(a) com entradas geradas conforme Figura 5(b). Observe que a saída do controlador deve ficar no intervalo [-1 1] para evitar a sua sa-



(a)



(b)

Figura 4 – (a) Diagrama do ambiente rltool para projeto do controlador PI e (b) diagrama de simulação.

turação e erro na geração da razão cíclica necessária para fornecer a correta ação do controlador.

O diagrama de blocos mostrado na Figura 6 apresenta os componentes do sistema de controle realimentado. Os blocos do RTW usados 'Frequency Output', 'Analog Input' e 'Digital Output' devem ser configurados para operar como saída, entrada e saída, respectivamente, com a mesma taxa de amostragem. Neste ponto, a placa I/O a ser usada deve ser selecionada de uma lista de placas compatíveis com o RTW. Utilizou-se o modelo DAQ AT-MIO-16E-10 da National Instruments com 16 entradas analógicas simples ou 8 diferenciais e 2 saídas analógicas com resolução de 12 Bits e 8 entradas/saídas digitais. Pode-se usar uma placa I/O não constante da lista, mas, tem-se que instalar o driver desta.

5. RESULTADOS EXPERIMENTAIS E DE SIMULAÇÃO

A obtenção da resposta de velocidade para a identificação dos parâmetros do conjunto motor CC, gerador e ponte H pode ser feita usando o diagrama Simulink da Figura 6 sem o controlador e com um valor de K_h pequeno e razão cíclica δ fixada no valor desejado. Foram usados $K_h = 0.001$ e $\delta = 0.8$.

Foi projetado um controlador PI adotando $\xi \geq 0.6$, tempo de acomodação $T_s \leq 0.25$ e erro de regime nulo. Apenas a velocidade é medida via o tacogerador. Utiliza-se a ferramenta de projeto rltool para o projeto do controlador via o método do lugar das raízes. Neste estágio do curso o aluno deve rever o método do lugar das raízes [6–8]. O controlador obtido é então discretizado para poder ser implementado no RTW do Matlab.

6. ELABORAÇÃO DO RELATÓRIO

O relatório deve ser visto como parte do aprendizado na disciplina de laboratório para fixar os conhecimentos adquiridos e aprofundar nos tópicos cobertos nas aulas de laboratório. Mas, o relatório tem de ser valorizado para o aluno dedicar um pouco do seu tempo na sua preparação para sanar dúvidas e completar o seu entendimento sobre os tópicos desenvolvidos em sala de aula. O relatório deve ser conciso e escrever as passagens principais para chegar aos resultados obtidos. Apresentar ao aluno diretrizes para a organização e formatação dos relatórios valoriza essa atividade. Os relatórios devem seguir normas de escrita científica com formatação adequada para as figuras, tabelas e incluir a citação dos livros e webpage consultados.

7. CONCLUSÃO

O material didático apresentado aborda o problema de implementação de controle realimentado para plantas reais utilizando a plataforma de desenvolvimento RTW e RTWT da Mathworks. O kernel do RTWT instalado torna possível o controle em tempo real através do Matlab e Simulink pois dá prioridade à implementação em relação aos outros processos do PC. Foram apresentados com detalhamento os componentes e a configuração necessária da implementação em tempo real. Resultados de simulação e experimentais foram apresentados para uma planta formada pelo conjunto motor CC e tacogerador. O material didático apresentado utiliza componentes comumente encontrados nos laboratórios de ensino e pode ser usado no 3o. ano de cursos de engenharia.

Neste projeto foram estudadas a implementação de técnicas de controle em plantas reais utilizando a plataforma de desenvolvimento Real-Time Workshop e xPC Target. Para isto, foi utilizado um kernel, o qual torna possível o controle em tempo real através do Matlab e Simulink pois dá prioridade a aplicação em relação aos outros processos do PC. Foi apresentado com detalhes a configuração necessária para a implementação de controle em tempo real. Para testar a configuração de controle, utilizou-se um motor CC com controlador P. Foi constatado que a saída da planta real foi compatível com a simulada e apresentada em um PC-alvo através da ferramenta xPC Target, a qual permite o controle e monitoração de plantas reais a distância. Com essas duas ferramentas é possível realizar o controle de diversas plantas, inclusive a distância.

A. ROTEIRO DO EXPERIMENTO

1. Ensaio para obter a constante do tacogerador K_{tg} via um encoder

   Com auxílio de um osciloscópio e um encoder óptico de 1024 linhas aplicar tensão na armadura denotada V_a e medir a freqüência fornecida nos terminais do encoder denotada f_e em [Hz] e a saída do tacogerador em volts para pelo menos 4 valores de V_a. A freqüência de uma rotação do motor é dada por $f_e/1024$. Utilizando os valores de V_{tg} e ω em rad/s aproximar os pontos a uma reta passando pela origem cuja inclinação fornece a constante K_{tg}.

2. Ensaio para identificação dos parâmetros do motor CC

   A partir do diagrama Simulink da Figura 6 retirar o controlador e com um valor de K_h pequeno e razão cíclica δ fixada no valor desejado obter a resposta de velocidade para o conjunto motor CC gerador com a ponte H. Para organizar os dados a serem utilizados na interface 'ìdent' usar o procedimento apresentado na Seção 3.

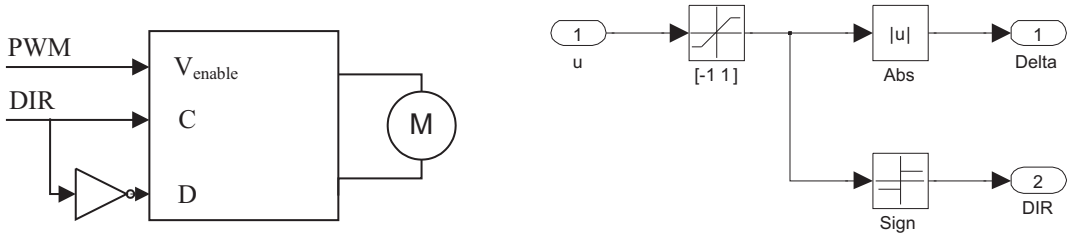
3. Projeto na interface gráfica rltool

   (a) Na tela da interface rltool, marcar a região do plano s para os valores de s do modo dominante descrito como $s_d = -\xi w_n \pm j w_n \sqrt{1-\xi^2})$ satisfazerem as especificações de sobresinal e tempo de acomodação dadas (ver Figura 4(a)).

   (b) Alocar o pólo na região marcada acima e ajustar o zero do controlador PI para atenuar o efeito do zero na sáida de velocidade e para o sinal de controle u respeitar a faixa de entrada do PWM.

4. Simulação do sistema realimentado

   (a) Considerar o sistema a malha fechada inicialmente com um controlador do tipo proporcional $K(s) =$



(a) (b)

Figura 5 – (a) Entradas para a ponte H e (b) diagrama Simulink para gerar as entradas para a ponte H.

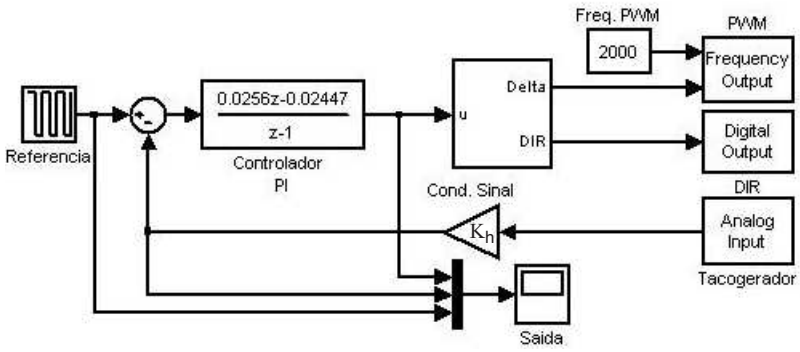
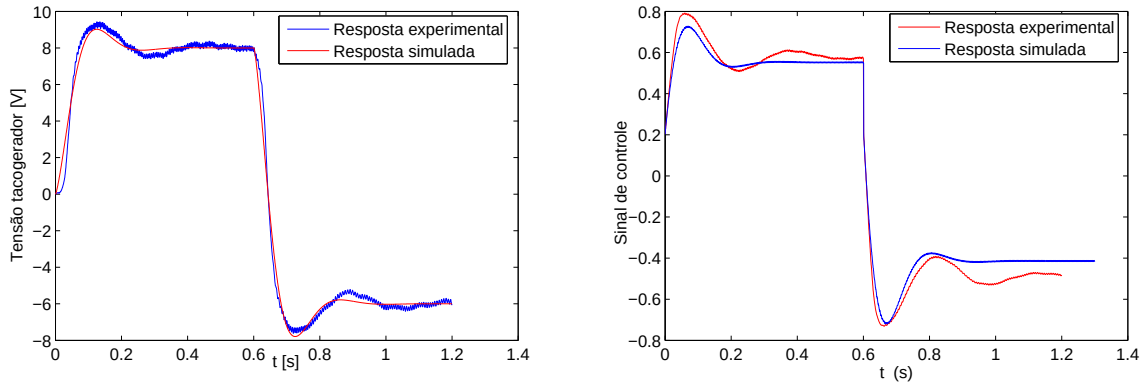


Figura 6 – Diagrama Simulink do sistema realimentado para implementação.



(a) (b)

Figura 7 – (a) Resposta e (b) sinal de controle para um degrau de tensão [V].

K_p. Verificar o valor máximo de K_p permitido, considerando a limitação da alimentação do motor CC. Prever no diagrama de simulação uma função não-linear do tipo saturação $(12 - 12V)$.

(b) Considerar um controlador do tipo PI e analisar o efeito do integrador na anulação do erro de regime a entrada degrau.

(c) Indicar o diagrama de blocos do sistema a malha fechada e dar sua função de transferência para $K_p = 1$, explicitando o valor dos pólos do sistema a malha fechada. Utilizar

```
>> [A,B,C,D]=linmod('diagrama')
>> T=ss(A,B,C,D)
```

com 'diagrama' o nome do 'arquivo mdl' contendo o diagrama de blocos do sistema a malha fechada.

5. Discretização do controlador necessária já que o RTW opera em modo discreto

   Usar

```
>> Cz=c2d(C,T,'zoh'),
```

   com $T = 1/2000$ e 'zoh' o método de aproximação usado, aproximação exata da resposta ao degrau nos pontos de discretização [9, 10].

6. Implementar o sistema de controle no ambiente RTW e RTWT via diagrama da Figura 6.

7. Verificar o sinal de erro quando ocorre uma perturbação de torque. Comentar sobre o efeito da realimentação na redução do erro de regime.

8. Obter a resposta transitória da saída do sistema controlado. Medir alguns valores de tempo e velocidade destacando tempo de pico, tempo de acomodação e o valor de regime da velocidade. Verificar se a especificação de tempo de acomodação e de erro de regime nulo foram atendidas.

9. Obter o sinal de controle $u(t)$. O sinal de controle deve ser constante em regime permanente.

10. Apresentar a função de transferência do sistema com controlador a malha fechada, explicitando o valor dos pólos do sistema a malha fechada.

B. DISCRETIZAÇÃO DO CONTROLADOR

No caso ideal, obtém-se o sinal discretizado como sendo dado pela convolução do sinal com um trem de pulsos unitários de período T

$$u_T(t) = u(t) * p(t) = u(t) \sum_{k=0}^{\infty} \delta(t - kT). \qquad (3)$$

Utilizando a transformada de Laplace obtém-se

$$\begin{aligned} u_T(s) &= \mathcal{L}(x(t) * p(t)) \\ &= \sum_{k=0}^{\infty} x(kT) e^{-kTs}. \end{aligned} \qquad (4)$$

Assim, a transformada de Laplace dos sinais discretizados resulta numa seqüência representada no somatório em (4). Da mesma forma, considerando o somatório de convolução

$$y(k) = \sum_{m=0}^{k} g(k-m)u(m), k = 0, 1, \cdots, \qquad (5)$$

com auxílio de (4), obtém-se a representação discretizada da função de transferência

$$\begin{aligned} G_T(s) &= \frac{y(s)}{u(s)} \\ &= \sum_{k=0}^{\infty} g(kT) e^{-kTs}. \end{aligned} \qquad (6)$$

A função de transferência relacionando entrada e saída discretas é então dada em termos de uma seqüência infinita descrita no somatório em (6) e não na formulação algébrica de razão de polinômios como no caso contínuo. A análise de sistemas discretos, portanto, com o uso da transformada de Laplace não oferece simplificação do tratamento matemático comum aos sistemas contínuos. A solução para isto é o uso da transformada Z definida a seguir.

A transformada Z é uma aplicação matemática que faz corresponder a cada seqüência de números, uma função da variável complexa z. A variável complexa z é definida como

$$z = e^{Ts}. \qquad (7)$$

Usando (7) em (4), obtém-se a seguinte representação do sinal discreto na variável z

$$\begin{aligned} u(z) &= \mathcal{Z}[u_T(t)) \\ &= \sum_{k=0}^{\infty} u(kT) z^{-k}. \end{aligned} \qquad (8)$$

$x(z)$ é então uma série infinita de potências da variável z denominada transformada Z do sinal discretizado em (4). Da mesma forma, usando (7) obtém-se a transformada Z da função discretizada em (6).

Os principais métodos de discretização são os métodos de aproximação por integração numérica, equivalência de mapeamento de polos e zeros do planos-s para o plano-z e equivalência com segurador (*sample hold* em inglês), também conhecido como método da resposta invariante ao degrau [9, 11]. No último método, ambas as funções de transferência em s e em z, $G(s)$ e $G(z)$, respectivamente, apresentam a característica de que as suas respostas ao degrau $y(t)$ e $y(k)$ são iguais em $t = kT$. Isto é conseguido quando

$$\mathcal{Z}^{-1}\left[G(z)\frac{1}{1-z^{-1}}\right] = \mathcal{L}^{-1}\left[G(s)\frac{1}{s}\right]_{t=kT} \qquad (9)$$

onde o lado esquerdo de (9) é igual a $y(kT)$ e o lado direito é igual a $y(t)$ em $t = kT$. Todos os outros métodos têm forte dependência do período de amostragem. Aplicando a transformada $\mathcal{Z}$ em (9) obtém-se

$$\begin{aligned} G(z) &= (1 - z^{-1})\mathcal{Z}\left[\frac{G(s)}{s}\right] \\ &= \mathcal{Z}\left[\frac{1 - e^{-sT}}{s}G(s)\right] \\ &= \mathcal{Z}[G_h(s)G(s)] \end{aligned} \quad (10)$$

onde $G_h(s)$ é a função de transferência de um segurador de amostras de ordem zero. Um segurador de amostras de ordem zero, como o próprio nome sugere, retém o sinal amostrado por um período de amostragem. Esta operação representa a conversão digital/analógica e é representada pelo bloco D/A e a operação inversa representa a conversão analógica/digital representada pelo bloco A/D (Figura 8). A saída amostrada em $t = kT$ é dada por

$$y(kT) = \sum_k \delta(t - kT)y(t). \quad (11)$$

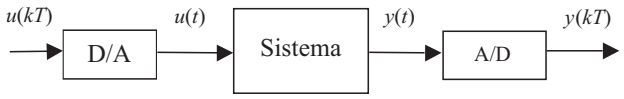


Figura 8 – Diagrama de blocos de sistema a malha aberta amostrado.

A função de transferência (10) também pode ser obtida como a função de transferência entre $u(kT)$ a $y(kT)$ incluindo os componentes A/D e D/A uma vez que a função de transferência do segurador de ordem zero pode ser obtida a partir da transformada Laplace de um trem de pulsos unitários de duração T

$$p(t) = 1(t) - 1(t - T) \quad (12)$$

onde $1(t)$ é a função degrau unitário. Assim, obtém-se

$$\mathcal{L}(p(t)) = (1 - e^{-sT})/s. \quad (13)$$

Neste método, usa-se a relação do plano s com o plano z pela definição da transformada z dada por (7). Com o período de amostragem T adequadamente escolhido de forma a atender o teorema da amostragem. Já que o método segue a definição (7), tem-se que uma faixa horizontal no semi-plano esquerdo do plano-s será mapeada no interior de um círculo unitário no plano-z como mostrado na Figura 9. uma vez que

$$\begin{aligned} z &= e^{Ts} = e^{\sigma T}e^{jwT} \\ &= re^{j\theta} \end{aligned} \quad (14)$$

para $s = jw + \sigma$ e, consequentemente, $r = e^{\sigma T}$ e $\theta = wT$.

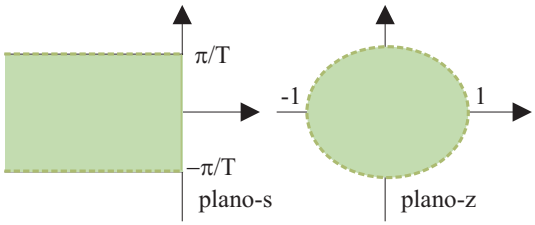


Figura 9 – Mapeamento do plano-s ao plano-z por $z = e^{Ts}$.

REFERÊNCIAS

[1] J Eker and A.Cervin. A Matlab toolbox for real-time and control systems co-design. In *Sixth International Conference on Real-Time Computing Systems and Applications*, pages 320–327, Hong Kong, P.R. China, 1999.

[2] F C Teng. Real-time control using MATLAB SIMULINK. In *International Conference on Systems, Man, Cybernetics*, volume 4, pages 2697–2702, Nashville, TN, USA, 2000.

[3] Real-time workshop getting started. The Mathworks Inc., Natick, MA, 2008.

[4] Real-time windows target 3 user's guide. The Mathworks Inc., Natick, MA, 2010.

[5] STMicroelectronics. Dual full-bridge driver. http://pdf1.alldatasheet.com/datasheet-pdf/view/22437/STMICROELECTRONICS/L298.html, consulta em abril de 2010.

[6] G. F. Franklin, J. D. Powell, and M. L. Worman. *Digital Control of Dynamic Systems*. Addison Wesley, New York, 1990.

[7] R. C. Dorf and R. H. Bishop. *Modern Control Systems*. Prentice Hall, Menlo Park, 2000.

[8] K. Ogata. *Discrete Time Control Systems*. Prentice Hall, Upper Saddle River, 1995.

[9] R. Isermann. *Digital Control Systems*. Springer Verlag, Heidelberg, 1989.

[10] V. A. Oliveira, M. L. Aguiar, and J. B. Vargas. *Sistemas de Controle: Aulas de Laboratório*. EESC/USP, São Carlos, SP, 2005.

[11] E. M. Hermely. *Controle por Computador de Sistemas Dinâmicos*. Edgard Blücher, São Paulo, 1996.