Instituto Tecnológico de Aeronáutica – ITA
Controle para Sistemas Computacionais – CMC-12
Laboratório 3 – Projeto de Servomotor de Velocidade

Professor: Marcos Ricardo Omena de Albuquerque Maximo

25 de maio de 2020

Observação: por questões de compatibilidade, este laboratório deve ser feito utilizando Simulink R2019b.

1 Introdução

Neste laboratório, projetar-se-á um controlador PI com pré-filtro para um servomotor de velocidade. Além da verificação por simulação da teoria aprendida nas aulas teóricas, o laboratório tem como objetivo abordar as seguintes questões:

- Comparar uso de controlador PI com P + *feedforward* para o servomotor de velocidade.
- Analisar por simulação o efeito de perturbações na malha.
- Verificar por simulação o efeito da adição de integrador sobre o erro em regime.
- Analisar por simulação a validade de uma aproximação por polos dominantes.
- Como implementar um controlador PI com pré-filtro em computador.
- Avaliar por simulação os efeitos introduzidos pela discretização no tempo.
- Avaliar necessidade de implementação de *anti-windup*.

Este laboratório trabalha com um modelo de um servomotor de velocidade, que é um conjunto formado por um motor elétrico, um sensor de velocidade, uma caixa de redução (formada por engrenagens) e um controlador. Também é possível incrementar mais ainda o servomotor dependendo da aplicação, e.g. pode-se incluir um sensor de corrente para adição de uma malha de corrente. Com isso, tem-se um sistema que autonomamente rastreia uma referência de velocidade angular. Um servomotor de velocidade pode ser usado em diversas aplicações que requerem controlar a velocidade de rotação de um eixo:

- Controlar a velocidade de rotação de uma hélice de propulsão de um avião ou de um barco.

- Controlar as velocidades de rotação das hélices de um quadricoptero.
- Controlar a velocidade de rotação de uma roda de um robô ou de um carro autônomo.

No caso de um sistema autônomo, o servomotor de velocidade recebe referência de um outro sistema de controle. Por exemplo, no caso do robô do laboratório 2, os comandos de velocidade calculados pelo seguidor de linha criam referências para os servomotores das rodas do robô. Neste laboratório, implementar-se-á um servomotor de velocidade para a roda de um robô. No caso, o exemplo considerado é o servomotor utilizado na roda do robô GLaDOS da ITAndroids, mostrado na Figura 1.

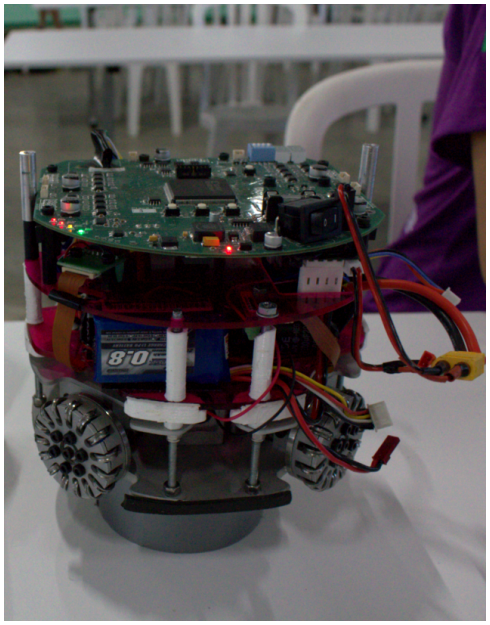

Figura 1: Robô GLaDOS da ITAndroids.

Conforme já visto nas aulas de teoria, um motor elétrico é um sistema eletromecânico, conforme apresentado no diagrama da Figura 2, e é regido por

$$\begin{cases} J_m\dot{\omega}_m + B_m\omega_m = K_t i = \tau_m, \\ V = L\dot{i} + Ri + K_t\omega_m. \end{cases} \tag{1}$$

em que ω_m é a velocidade de rotação do motor, i é a corrente que circula pelo motor, J_m é a inércia do motor, B_m é o coeficiente de atrito viscoso do eixo do motor, K_t é a constante de torque, τ_m é o torque gerado pelo motor, V é a tensão aplicada nos terminais do motor, L é a indutância e R é a resistência.

Entretanto, um motor é muito fraco para a maioria das aplicações, de modo que é comum o uso de engrenagens para aumentar o torque na saída. Para compreender esse fenômeno, considere o par de engrenagens apresentado na Figura 3. Conforme a figura sugere, numere como 1 e 2 as engrenagens da esquerda e da direita, respectivamente. As velocidades lineares das duas engrenagens nos pontos de contato dos dentes devem ser iguais:

$$v_1 = v_2, \tag{2}$$

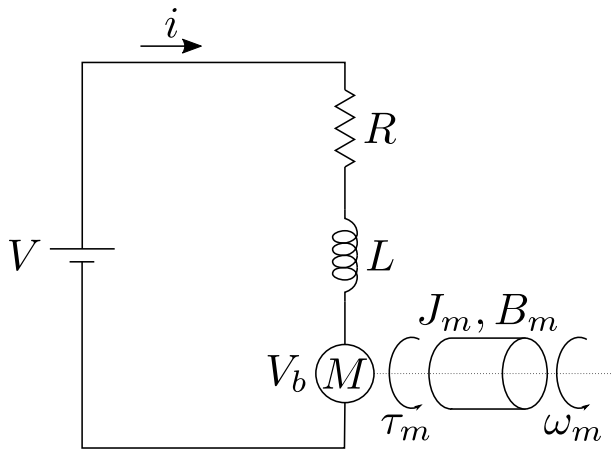

Figura 2: Motor elétrico.

logo

$$\omega_1 R_1 = \omega_2 R_2, \tag{3}$$

em que ω_1 e ω_2 são as velocidades angulares das engrenagens, enquanto R_1 e R_2 representam seus respectivos raios. Já N_1 e N_2 denotam os números de dentes das engrenagens. Com os dentes devem ter o mesmo tamanho, tem-se que R_1 e R_2 proporcionais a N_1 e N_2, respectivamente. Portanto

$$\omega_1 N_1 = \omega_2 N_2 \Rightarrow \frac{\omega_2}{\omega_1} = \frac{N_2}{N_1} = \frac{1}{N}, \tag{4}$$

em que

$$N = \frac{N_1}{N_2} \tag{5}$$

é chamado fator de redução, pois o par de engrenagens em geral é utilizado para reduzir a velocidade de rotação do motor. Para entender a vantagem em reduzir a velocidade de rotação, perceba que, por conservação da energia, deve-se ter a mesma potência mecânica nas duas engrenagens:

$$P_1 = P_2 \Rightarrow \tau_1 \omega_1 = \tau_2 \omega_2 \Rightarrow \frac{\tau_2}{\tau_1} = \frac{\omega_1}{\omega_2} = N, \tag{6}$$

ou seja, tem-se uma troca de velocidade por torque (reduz-se a velocidade, mas aumenta-se o torque). Na realidade, há perdas na transmissão de torque, geralmente refletida num parâmetro chamado eficiência da transmissão η, de modo que as relações tornam-se

$$\frac{\omega_2}{\omega_1} = \frac{1}{N}, \ \frac{\tau_2}{\tau_1} = N\eta. \tag{7}$$

Dependendo da aplicação, pode ser necessário o uso de várias engrenagens em série para atingir a redução N desejada, que são organizadas no que se chama de caixa de redução. Com isso, considere que o motor foi acoplado a uma carga (em inglês, *load*) com inércia J_l. No nosso caso, a carga é a roda do robô. Além disso, seja B_l o coeficiente de atrito viscoso do eixo de saída. Como (1) foi escrita sem considerar a presença de carga, deve-se analisar como essa equação deve ser adaptada para levar em conta a carga.

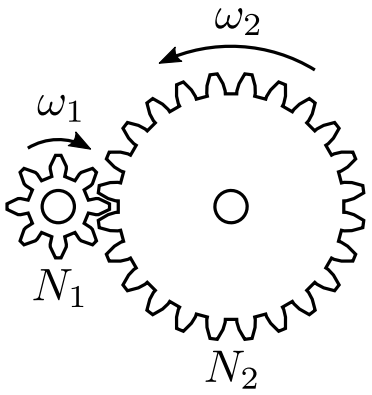

Figura 3: Par de engrenagens.

Para isso, perceba que o torque gerado pelo motor deve mover tanto o próprio motor quanto a carga, de modo que pode-se escrever

$$\begin{cases} \tau_m = J_m\dot{\omega}_m + B_m\omega_m + \tau_{t,i}, \\ \tau_{t,o} = J_l\dot{\omega}_l + B_l\omega_l. \end{cases} \tag{8}$$

em que ω_m e ω_l são as velocidades de rotação do motor e da carga, respectivamente, e $\tau_{t,i}$ e $\tau_{t,o}$ são torques de transmissão na entrada e na saída da caixa de redução, respectivamente. Seja N a relação de redução, tem-se

$$\omega_l = \frac{\omega_m}{N},\ \tau_{t,o} = N\eta\tau_{t,i}. \tag{9}$$

Substituindo (9) em (8), tem-se

$$\begin{cases} \tau_m = J_m\dot{\omega}_m + B_m\omega_m + \tau_{t,i} \\ N\eta\tau_{t,i} = J_l\dfrac{\dot{\omega}_m}{N} + B_l\dfrac{\omega_l}{N} \end{cases} \Rightarrow \underbrace{\left(J_m + \frac{J_l}{N^2\eta}\right)}_{J_{eq}}\dot{\omega}_m + \underbrace{\left(B_m + \frac{B_l}{N^2\eta}\right)}_{B_{eq}}\omega_m = \tau_m, \tag{10}$$

em que J_{eq} e B_{eq} são chamados inércia e coeficiente de atrito viscoso equivalentes, conforme visto pelo motor, respectivamente. Perceba que também há também a equação equivalente conforme vista pela carga:

$$\left(N^2\eta J_m + J_l\right)\dot{\omega}_l + \left(N^2\eta B_m + B_l\right)\omega_l = \tau_l, \tag{11}$$

em que $\tau_l = N\eta\tau_m$ é o torque gerado pelo motor na carga. Nesse caso, define-se também inércia e coeficiente de atrito viscoso equivalentes conforme visto pela carga. Em geral, o que dita se (10) ou (11) será usada para projeto do sistema de controle é se o sensor de velocidade está no eixo do motor ou da carga, i.e. se a velocidade medida para retroalimentação é ω_m ou ω_l. Há vantagens e desvantagens em colocar o sensor no motor ou na carga. No caso da GLaDOS, o sensor (*encoder*) está no motor, logo (10) será usada, de modo que tem-se

$$\begin{cases} J_{eq}\dot{\omega}_m + B_{eq}\omega_m = \tau_m, \\ V - K_t\omega_m = L\dot{i} + Ri. \end{cases} \tag{12}$$

Finalmente, seja τ_e um torque externo (distúrbio) aplicado na carga. Transferindo para o motor, as equações que regem a dinâmica do servomotor são

$$\begin{cases} J_{eq}\dot{\omega}_m + B_{eq}\omega_m = \tau_m + \dfrac{\tau_e}{N\eta}, \\ V - K_t\omega_m = L\dot{i} + Ri. \end{cases} \tag{13}$$

em que V é a entrada, ω_m é a saída e τ_e é um entrada de perturbação. De acordo com o discutido, a Figura 4 mostra um diagrama da planta (dinâmica) do servomotor.

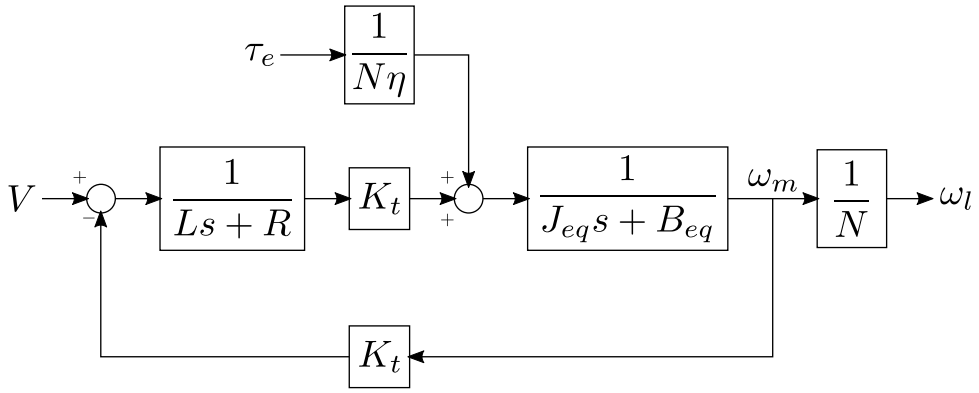

Figura 4: Diagrama de blocos da planta do servomotor.

Como dito anteriormente, o exemplo considerado é o servomotor de velocidade da GLaDOS, robô da ITAndroids. O motor utilizado é um Maxon EC-45 30 W 200142 com tensão nominal de 12 V. A redução é implementada por um único par de engrenagens com $N = 3$. A carga considerada é uma roda omnidirecional, projetada pela ITAndroids. O sensor de velocidade utilizado é um *encoder* ótico de quadratura US Digital ET4 com 360 contagens por revolução. Todos os parâmetros associado a essa planta são fornecidos em código pela função `obterPlantaServo`.

2 Tarefas

2.1 Controlador P + *Feedforward*

Inicialmente, você implementará um controlador P + *feedforward* para o servomotor de velocidade, conforme mostrado na Figura 5. Perceba que, embora a malha fechada considere grandezas no motor, a referência r_l para o servomotor é em termos de velocidade da roda, a qual é convertida para referência de velocidade do motor $r_m = Nr_l$.

Determine a função de transferência de malha fechada para o sistema de controle

$$G_R(s) = \frac{\Omega_l(s)}{R_l(s)}. \tag{14}$$

Ademais, determine a função de transferência do distúrbio (torque externo) para a saída

$$G_D(s) = \frac{\Omega_l(s)}{\mathrm{T}_e(s)}. \tag{15}$$

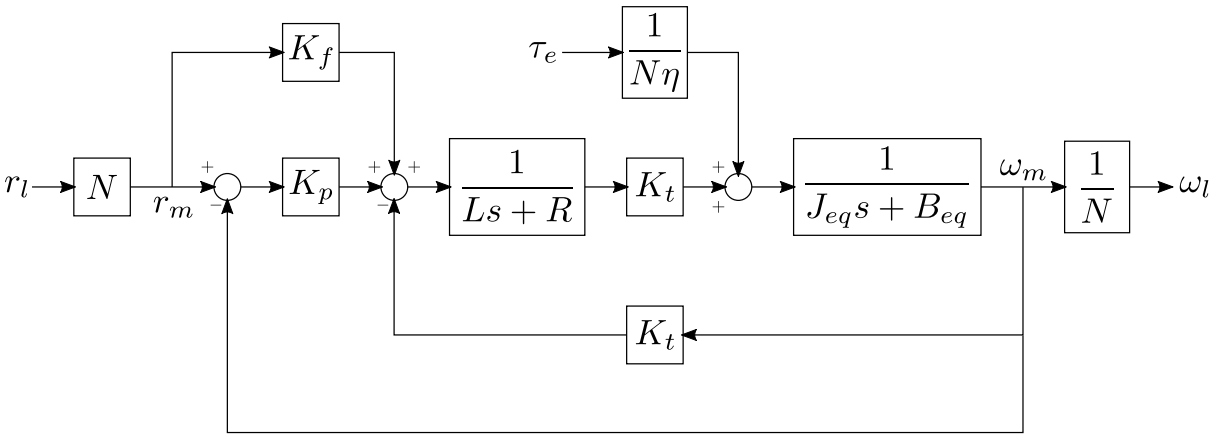

Figura 5: Diagrama de blocos do servomotor com controlador P + *feedforward*.

Com isso, negligenciando o efeito do indutor (i.e. considerando $L \approx 0$), determine expressões para os ganhos K_p e K_f para que o sistema tenha constante de tempo τ e erro nulo em regime para entrada degrau unitário. Mostre que o sistema projetado não possui erro em regime para entrada degrau unitário mesmo quando o indutor é considerado. Inclua essas deduções no relatório. A partir disso, implemente a função de MATLAB `projetarControladorPFeedforward`.

Assim, execute o *script* `avaliarPFeedforward`, que realiza uma simulação através do modelo de Simulink `servomotor_p_feedforward.slx` (já implementado) com os seguintes parâmetros:

- Controlador projetado para $\tau = 0{,}01\ s$.
- Referência degrau com amplitude de 50 rad/s e início em 0 s.
- Distúrbio degrau com amplitude de 0,2 Nm e início em 0,1 s.

Além disso, o *script* traça os seguintes gráficos:

- Velocidade do motor ω_m.
- Velocidade da roda ω_l.
- Tensão comandada V.

Pede-se:

- Verifique se a relação entre velocidades do motor e da roda condiz com o esperado.
- Discuta qualitativamente o resultado obtido com base na ordem do sistema e nas expressões das funções de transferência $G_R(s)$ e $G_D(s)$. Comente sobre erro em regime antes e após o início da perturbação.
- Usando o princípio da superposição e o teorema do valor final, verifique que a saída obtida na simulação em regime antes e após o início da perturbação condiz com o previsto pela teoria.

- Verifique se o valor da tensão comandada em regime antes e após o início da perturbação condiz com o previsto pela teoria.

Inclua os gráficos e as discussões no seu relatório.

2.2 Controlador PI com Pré-Filtro

Nessa seção, você implementará um controlador PI com pré-filtro para o servomotor de velocidade, conforme mostrado na Figura 6. Lembre-se que um controlador PI possui compensador dado por

$$C(s) = K_p + \frac{K_i}{s} = \frac{K_p s + K_i}{s}. \tag{16}$$

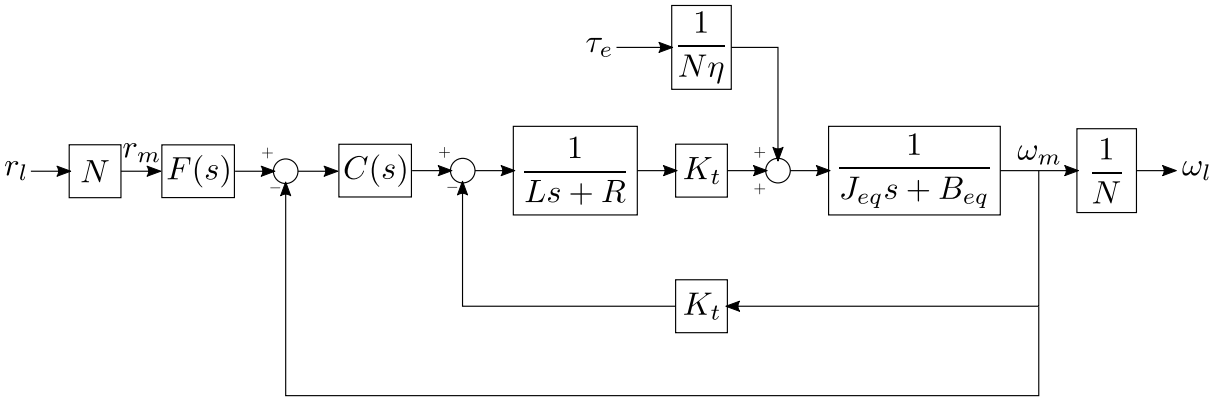

Figura 6: Diagrama de blocos do servomotor usando controlador PI com pré-filtro.

A função de transferência do pré-filtro $F(s)$ é

$$F(s) = \frac{K_i}{K_p s + K_i} \tag{17}$$

Com isso, determine a função de transferência em malha fechada

$$G_R(s) = \frac{\Omega_l(s)}{R_l(s)}. \tag{18}$$

Obtenha ainda, para esse caso, a função de transferência do distúrbio para a saída

$$G_D(s) = \frac{\Omega_l(s)}{\mathrm{T}_e(s)}, \tag{19}$$

incluindo-a no relatório. Então, ignorando a dinâmica da corrente (i.e. $L \approx 0$), projete o sistema de controle para atender a requisitos de tempo de subida de 0 a 100% $t_r|_0^{100\%}$ e sobressinal M_p. Mostre que o sistema projetado não possui erro em regime para entrada degrau unitário nem para perturbação degrau unitário, mesmo quando o indutor não é negligenciado. Inclua todas as deduções solicitadas no seu relatório.

Portanto, use o *script* `avaliarControladorPI`, que realiza uma simulação através do modelo de Simulink `servomotor_pi.slx` (já implementado) com os seguintes parâmetros:

- Controlador projetado para $t_r|_0^{100\%} = 0{,}02\ s$ e $M_p = 0{,}046$.
- Referência degrau com amplitude de 50 rad/s e início em 0 s.
- Distúrbio degrau com amplitude de 0,2 Nm e início em 0,1 s.

Ademais, os mesmos gráficos da simulação da seção anterior são traçados. Como análise, pede-se:

- Verifique se o sistema atende aos requisitos de tempo de subida e sobressinal.
- Verifique se na simulação o sistema de fato apresenta erro nulo em regime para entrada e distúrbio do tipo degrau.
- Analise a tensão comandada em regime, antes e após o distúrbio, e responda: qual dos termos do controlador PI está sendo responsável por fornecer a tensão necessária para eliminar o erro em regime? Justifique.
- Mencione quais fenômenos necessitam ser "cancelados" pela tensão comandada para que o sistema tenha erro nulo em regime.

2.3 Aproximação por Polos Dominantes

No projeto do controlador PI, ignorou-se o efeito do indutor. Essa aproximação remove um polo da função de transferência em malha fechada, de modo que essa assume uma forma de sistema de segunda ordem padrão, o que é muito conveniente para projeto. Pode-se justificar negligenciar o indutor como uma aproximação por polos dominantes.

Para analisar o efeito do indutor, foi fornecido o *script* `avaliarIndutor`, que apresenta resposta ao degrau e posições dos polos de malha fechada para diferentes valores de L. Esse *script* considera:

- Usa-se controlador PI projetado para $t_r|_0^{100\%} = 0{,}02\ s$ e $M_p = 0{,}046$.
- Usa-se referência degrau com amplitude 50 rad/s.
- Não há perturbação.
- Seja $L_{nom} = 0{,}56\ mH$ o valor nominal de indutância. O *script* considera $L \in \{L_{nom}/2, L_{nom}, 2L_{nom}, 4L_{nom}, 8L_{nom}\}$.

Para que `avaliarIndutor` funcione corretamente, você deve implementar a função `obterPolosPI`, que determina os polos de malha fechada para o servomotor de velocidade usando controlador PI com pré-filtro. Dica: crie a função de transferência no MATLAB e obtenha os polos com uso de `pole`. Como análise, pede-se:

- Verifique se os polos complexo-conjugados estão aproximadamente nas posições esperadas quando L é baixo.
- Discuta como os polos de malha fechada se movem à medida que o valor de L aumenta e relacione isso com o critério usado para aproximação por polos dominantes.

- Discuta o aumento de L influencia a resposta ao degrau e relacione isso com as posições dos polos.

Inclua os gráficos e as discussões no seu relatório. Finalmente, perceba que a influência de L não é absoluta, mas depende também dos ganhos utilizados no controlador. Para verificar isso, use o *script* `avaliarTempoSubida`, que apresenta resposta ao degrau e posições dos polos de malha fechada com o controlador projetado levando em conta diferentes valores de tempo de subida de 0 a 100% $t_r|_0^{100\%}$. Esse *script* considera:

- Controlador PI projetado com $M_p = 0{,}046\ s$. Para o tempo de subida, seja $t_{nom} = 0{,}02\ s$, então considera-se $t_r|_0^{100\%} \in \{t_{nom}, t_{nom}/2, t_{nom}/4, t_{nom}/8\}$.
- Referência degrau com amplitude 50 rad/s.
- Não há perturbação.
- Os parâmetros da planta são os nominais.

Discuta como o aumento de $t_r|_0^{100\%}$ influencia a resposta ao degrau e as posições dos polos. Inclua discussões e gráficos no seu relatório.

3 Implementação Digital

Conforme discutido em sala, para implementação em processador digital (microcontrolador ou computador), é necessário discretizar as funções de transferência usadas no controlador. Quanto discutiu-se controlador PI, apresentou-se uma fórmula para discretização de um compensador PI baseado em aproximar a integral por trapézios:

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau)d\tau \Rightarrow u[k] \approx K_p e[k] + K_i \sum_{i=1}^{k} \left(\frac{e[i-1] + e[i]}{2} \right) T. \tag{20}$$

em que $e[k]$ e $e[k]$ representam o comando e o erro no instante de amostragem (tempo discreto) k. Pode-se escrever essa equação em um formato mais conveniente para implementação. Para isso, perceba que

$$\begin{aligned} u[k-1] = K_p e[k-1] + K_i \sum_{i=1}^{k-1} \left(\frac{e[i-1] + e[i]}{2} \right) T \Rightarrow \\ K_i \sum_{i=1}^{k-1} \left(\frac{e[i-1] + e[i]}{2} \right) T = u[k-1] - K_p e[k-1], \end{aligned} \tag{21}$$

de modo que pode-se escrever

$$\begin{aligned} u[k] = K_p e[k] + K_i \sum_{i=1}^{k-1} \left(\frac{e[i-1] + e[i]}{2} \right) T + K_i \frac{e[i-1]}{2} T + K_i \frac{e[i]}{2} T \Rightarrow \\ u[k] = u[k-1] + \left(K_p + \frac{K_i T}{2} \right) e[k] + \left(-K_p + \frac{K_i T}{2} \right) e[k-1]. \end{aligned} \tag{22}$$

Para discretizar o pré-filtro, perceba que

$$F(s) = \frac{Y(s)}{U(s)} = \frac{K_i}{K_p s + K_i} = \frac{a}{s+a} \Rightarrow sY(s) + aY(s) = aU(s) \Rightarrow$$
$$\dot{y} + ay = au \Rightarrow \left(\frac{y[k] - y[k-1]}{T}\right) + ay[k] \approx au[k] \Rightarrow \tag{23}$$
$$y[k] = \frac{1}{1+aT}\left(y[k-1] + aTu[k]\right),$$

em que

$$a = \frac{K_i}{K_p}. \tag{24}$$

Implemente essas equações no modelo de Simulink `controlador_pi_discreto.slx`. Por enquanto, ignore o parâmetro `Vmax` presente no compensador. Então, execute o *script* `avaliarControladorDiscreto`, que realiza simulações de acordo com o seguinte:

- Controlador PI projetado para $t_r|_0^{100\%} = 0{,}02\ s$ e $M_p = 0{,}046$.
- Referência degrau com amplitude 50 rad/s.
- Não há perturbação.
- Discretização do controlador PI com $f_s = 1/T \in [1k, 500, 200, 100]\, Hz$.

Analise por simulação como a discretização degrada o desempenho do controlador de acordo com a taxa de amostragem $f_s = 1/T$ utilizada. Discuta se o comportamento observado condiz com o esperado. Inclua os gráficos e as discussões no seu relatório.

4 Projeto de Anti-windup

Até o momento, considerou-se que não há limite para a tensão aplicada nos terminais do motor. Porém, na prática, o comando de tensão é limitado pela fonte de tensão utilizada, que no caso da GLaDOS é uma bateria de lítio polímero (LiPo) de 3 células. Dependendo da carga da bateria, sua tensão varia de 11,1 a 12,4 V.

Como a técnica de projeto assume um modelo linear, não é possível levar em conta essa saturação durante o projeto. Esse descasamento é acentuado quando há um integrador na malha, de modo que surge um efeito chamado *windup*. O *windup* surge porque o integrador acumula mais erro do que deveria, dado que a saturação faz com que a referência seja atingida mais lentamente do que o esperado pelo modelo linear.

Esse acúmulo excessivo de erro no integrador aumenta o *overshoot* e pode até mesmo desestabilizar o controlador. Para mitigar o problema, deve-se implementar um *anti-windup*. Quando se tem um controlador em formato digital como mostrado em (22), uma forma muito simples de se implementar um *anti-windup* consiste em limitar explicitamente o valor do comando $u[k]$. No caso, aplica-se

$$u[k] = \begin{cases} u[k],\ -V_{max} \leq u[k] < V_{max}, \\ -V_{max},\ u[k] < -V_{max}, \\ V_{max},\ u[k] > V_{max}. \end{cases} \tag{25}$$

logo após o cálculo de $u[k]$ de acordo com (22). Com isso, esse valor limitado é usado como $u[k-1]$ na próxima iteração, evitando o acúmulo de integral. Implementa essa estratégia de *anti-windup* no compensador em `servomotor_pi_discreto.slx`.

Para testar o *anti-windup*, use o *script* `avaliarAntiwindup`, que considera simulações com e sem limitação de comando de tensão. Esse *script* considera:

- Controlador PI projetado para $t_r|_0^{100\%} = 0{,}02\ s$ e $M_p = 0{,}046$.
- Referência degrau com amplitude 100 rad/s. Aumentou-se a referência para tornar o *windup* mais pronunciado.
- Não há perturbação.
- Tensão máxima da bateria como 12 V, a tensão nominal do motor Maxon EC-45 200142.

Inclua os gráficos gerado no seu relatório e comente se o problema de *anti-windup* foi resolvido.

5 Instruções

- A primeira etapa do processo de correção consistirá em submeter as funções implementadas a vários casos de teste de forma automatizada. Assim, os cabeçalhos das funções devem ser seguidos **rigorosamente**. Arquivos `.m` com os nomes destas funções e os cabeçalhos já implementados foram fornecidos juntamente com este roteiro. Dê preferência a implementar seu laboratório a partir destes arquivos `.m` fornecidos para evitar erros.
- A entrega da solução desse laboratório consiste de arquivos de código (MATLAB e Simulink) e de um relatório (em `.pdf`), que devem ser submetidos no Google Classroom.
- Compacte todos os arquivos a serem submetidos em um único `.zip` (use obrigatoriamente `.zip`, e **não** outra tecnologia de compactação de arquivos) e anexe esse `.zip` no Google Classroom. Para o `.zip`, use o padrão de nome `<login_ga>_labX.zip`. Por exemplo, se seu login é `marcos.maximo` e você está entregando o laboratório 1, o nome do arquivo deve ser `marcos.maximo_lab1.zip`. **Não** crie subpastas, deixe todos os arquivos na "raiz" do `.zip`.
- O relatório deve ser sucinto, preocupe-se apenas em incluir discussões e entregáveis solicitados no roteiro. Pede-se apenas um cuidado mínimo na elaboração do relatório: responder adequadamente as perguntas, incluir figuras diretamente no relatório (ao invés de deixar como arquivos separados), figuras de boa qualidade, colocar nomes nos eixos dos gráficos, colocar legenda para diferenciar curvas num mesmo gráfico etc.
- **Não** é permitido o uso de funções ou comandos prontos do MATLAB que realizem toda a funcionalidade atribuída a uma certa função cuja implementação foi solicitada. Entretanto, o uso destas funções para verificação das implementações realizadas é encorajado. Em caso de dúvida, consulte o professor.

- A criação de *scripts* e funções auxiliares no MATLAB para execução de experimentos e geração de gráficos é fortemente recomendada para facilitar seu trabalho. Porém, não há necessidade de entregar código auxiliares.

- **Não** há necessidade de copiar e colar o código no seu relatório, dado que você também submeterá os arquivos de código. Também **não** há necessidade de explicar sua implementação no relatório, a **não** ser que o roteiro tenha solicitado explicitamente. Porém, organizar e comentar o código é muito salutar, pois ele será o foco da correção.

6 Dicas

- Nos modelos de simulação entregues, todos os blocos e os parâmetros de simulação já estão adequadamente configurados.

- Além dos blocos Simulink conhecidos de laboratórios anteriores, neste laboratório, usa-se o bloco `Delay`, que gera atraso em tempo discreto. O símbolo z^{-1} em um bloco `Delay` significa que ele gera atraso de 1 tempo de amostragem. Perceba que z é a variável da chamada transformada Z, uma transformada matemática usada para funções em tempo discreto.

- Para colocar tempo de amostragem em blocos Simulink, deve-se editar a propriedade `Sample Time` desses. Não se preocupe, pois todos os blocos já foram configurados.

- Para usar a saída do bloco `To Workspace` no formato `Structure with Time`:

  ```
  plot(out.x.time, out.x.signals.values)
  ```

- Caso queira chamar o Simulink dentro do MATLAB, use `out = sim('arquivo.slx')`. A saída do Simulink ficará na variável `out` nesse caso.