Projecto de Controlo Digital – 2013/2014

Modelação, controlo e simulação de um motor com trem de engrenagens e carga

Trabalho Prático Nº1

José Carlos Direito
2010129901

José Pedro Medeiros
2010129934

Luís Miguel Rocha
2010127532

Introdução

Neste trabalho foi feita a análise e comparação de três modelos. Esta análise e comparação incide sobre o modelo do motor + engrenagens + carga, sobre o modelo da função de transferência e sobre o modelo em espaço de estados. Todos estes modelam um servomotor.

Posteriormente avaliámos a resposta de cada um dos modelos a uma referência em rampa e a uma perturbação na entrada. Esta avaliação foi feita quando cada um destes modelos era controlado por um controlador PD e PID contínuo, e pelas respectivas aproximações discretas dos mesmos controladores. Foi ainda testado o efeito da introdução de um filtro no termo derivativo de cada um dos controladores acima referidos.

De seguida foram projectados e comparados dois observadores de estados aumentado, um com dinâmica Deadbeat e outro com dinâmica de segunda ordem, escolhida por nós.

Foi ainda efectuado um teste de robustez aos controladores acima referidos para duas situações: Uma perturbação aditiva no binário; E para variações na carga.

Por fim foi projectado um controlador em espaços com realimentação dos estados estimados pelo observador e analisámos a sua resposta a uma perturbação na entrada e no binário.

Tabela de conteúdos

1 Modelação do motor DC

1.1 Diagrama de blocos do servomotor DC com modelo mecânico de engrenagens e carga

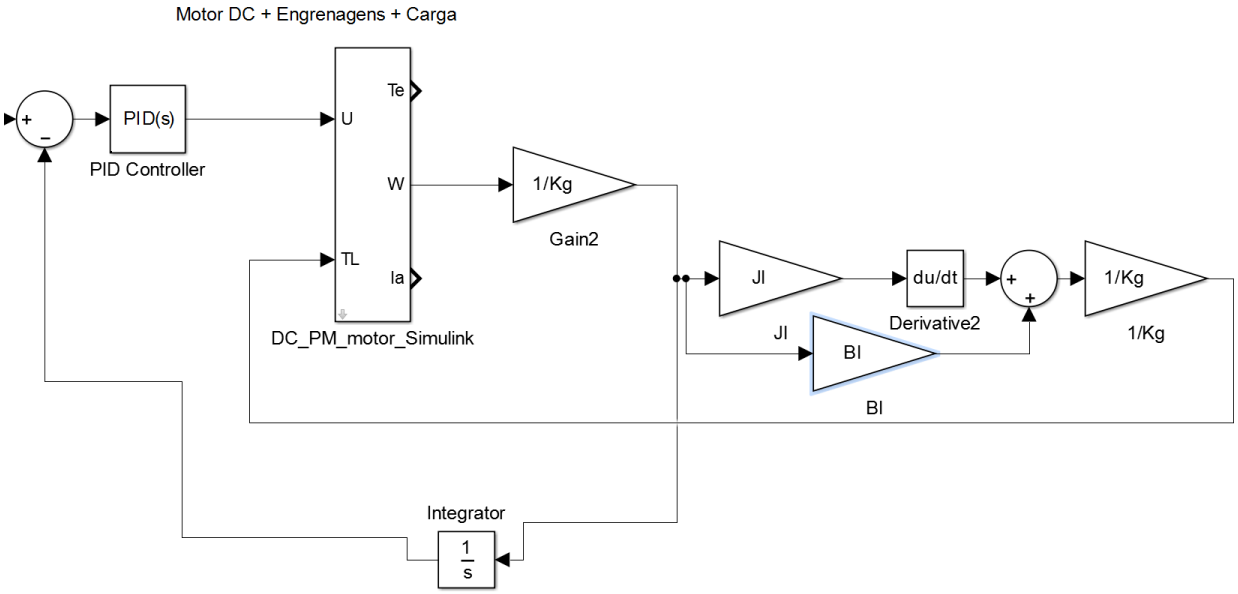

Figura 1: Diagrama do motor DC

Na figura 1 apresentamos o diagrama de blocos do motor DC com caixa de engrenagens e uma carga. Este bloco foi usado para correr várias simulações no *Simulink*.

1.2 Modelo de circuito do servomotor DC com modelo mecânico de engrenagens e carga

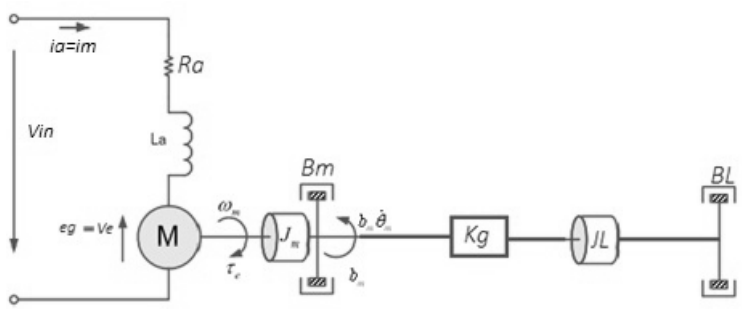

Figura 2: Modelo mecânico do motor DC

Na figura 2 apresentamos o circuito representativo do motor DC com caixa de engrenagens e uma carga aplicada ao mesmo. Este modelo está na base da análise deste trabalho sendo por isso importante analisar todas as grandezas e principais expressões matemáticas. Essa análise encontra-se na próxima secção deste relatório.

1.3 Diagrama de grandezas e expressões matemáticas

Nesta secção apresentamos as grandezas envolvidas no estudo do motor DC assim como todas as expressões matemáticas associadas presentes na figura 2. O binário gerado electricamente no veio do motor é dado em função da corrente do induzido:

$$\tau_e(t) = Km \times ia(t) \qquad \text{(Binário)} \tag{1}$$

em que Km é uma constante de acoplamento electromagnético. A velocidade do veio gera uma força contra-electromotriz induzida:

$$ve = eg = Km \times Wm \qquad \text{(Força contra-electromotriz)} \tag{2}$$

As equações 1 e 2 não são lineares. Para as linearizar podemos optar por:

1. Motor controlado pelo induzido
2. Motor controlado por campo

A expressão matemática da tensão de entrada no motor de corrente contínua é expressa da seguinte forma:

$$V_{in} = Ra \times ia + Km \times Wm \qquad \text{(Tensão no motor de corrente contínua)} \tag{3}$$

Quanto á velocidade do veio do motor (velocidade angular) pode ser expressa da seguinte maneira:

$$W_m = Kg \times W_L = Kg \times \dot{\theta}_{Local} \qquad \text{(Velocidade angular do motor)} \tag{4}$$

, onde Kg é expresso da seguinte forma:

$$K_{\mathrm{g}} = K_{\mathrm{ge}} \times K_{\mathrm{gi}} \qquad \text{(Relação desmultiplicadora)} \tag{5}$$

Sendo K_{ge} e K_{gi} a relação entre engrenagens do trem externo e a relação entre engrenagens da caixa redutora respectivamente.

Para o cálculo do momento de inércia equivalente usa-se a seguinte expressão:

$$J_{eq} = K_g^2 \times J_m + J_l \qquad \text{(Momento de inércia equivalente)} \tag{6}$$

, onde J_m representa o momento de inércia do veio do motor. Quanto ao momento de viscosidade equivalente é expresso da seguinte maneira:

$$B_{\mathrm{eq}} = {K_{\mathrm{g}}}^2 \times B_{\mathrm{m}} + B_{\mathrm{l}} \qquad \text{(Momento de viscosidade equivalente)} \tag{7}$$

O binário na carga é expresso da seguinte forma:

$$\tau_{\mathrm{l}} = J_{\mathrm{eq}} \times \dot{\omega_{\mathrm{c}}} \qquad \text{(Binário na carga)} \tag{8}$$

Todas as expressões acima descritas são importantes para o controlo do motor, sendo estas usadas neste trabalho para efeitos de simulação.

2 Função de transferência do servomotor

Considerando $B = 0$, o binário eléctrico é igual ao somatório dos binários opostos:

$$\tau_{\mathrm{e}}(s) = J_{\mathrm{m}} K_{\mathrm{g}} s^2 \theta_{\mathrm{l}}(s) + \frac{J_{\mathrm{l}}}{K_{\mathrm{g}}} s^2 \theta_{\mathrm{l}}(s) \qquad \text{(Binário eléctrico)} \tag{9}$$

De (1), (9) e (3) obtemos:

$$G(s) = \frac{\theta_{\mathrm{l}}(s)}{V_{\mathrm{in}}(s)} = \frac{1}{s^2 \frac{R_{\mathrm{a}} J_{\mathrm{eq}}}{K m K_{\mathrm{g}}} + s K_{\mathrm{m}} K_{\mathrm{g}}} \qquad \text{(Função de transferência do servomotor)} \tag{10}$$

3 Modelo em espaço de estados do servomotor

Analisando o sistema em *malha aberta*, sabemos que $G(s) = \frac{\theta_l(s)}{V_{in}(s)} = \frac{\theta_l(s)}{\theta_d(s)}$ e utilizando a função de transferência do servomotor (10), deduz-se:

$$\frac{\theta_{\mathrm{l}}(s)}{\theta_{\mathrm{d}}(s)} = \frac{1}{0.0111 s^2 + 0.5460 s} \Leftrightarrow \theta_{\mathrm{d}}(s) = \theta_{\mathrm{l}}(s) \times 0.0111 s^2 + \theta_{\mathrm{l}}(s) \times 0.5460 s \tag{11}$$

, fazendo a transformada inversa obtém-se:

$$\theta_{\mathrm{d}} = \ddot{\theta}_{\mathrm{l}} 0.0111 + \dot{\theta}_{\mathrm{l}} 0.5460 \text{ com:} \tag{12}$$

$$x_1 = y = \theta_{\mathrm{l}} \tag{13}$$

$$x_2 = \dot{x_1} = \dot{\theta}_{\mathrm{l}} \tag{14}$$

$$\dot{x_2} = \ddot{\theta}_{\mathrm{l}} \tag{15}$$

$$\theta_{\mathrm{d}} = u \tag{16}$$

Substituindo (15) e (14) em (12) obtemos a seguinte relação: $\dot{x2} = \ddot{\theta}_{\mathrm{l}} = \frac{\theta_{\mathrm{d}} - 0.5460 x2}{0.0111}$

Colocando na forma matricial, construimos o seguinte modelo em Espaços de Estados:

$$\begin{bmatrix} \dot{x_1} \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & \frac{-0.5460}{0.0111} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{0.0111} \end{bmatrix} \begin{bmatrix} u \end{bmatrix} \tag{17}$$

$$\begin{bmatrix} y \end{bmatrix} = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \end{bmatrix} \begin{bmatrix} u \end{bmatrix} \tag{18}$$

A equação (17) é conhecida como sendo a equação dinâmica, onde **A** é a matriz do sistema, **B** é a matriz de controlo e a (18) é a equação da saída, onde **C** é a matriz de saída e **D** é a matriz que permite que a entrada afecte directamente a saida do sistema.

4 Controlador PD

4.1 Projecto do controlador

Um controlador PD rege-se por duas equações, sendo estas:

$$V_{\text{in}}(t) = K_{\text{p}}e(t) + K_{\text{d}}\dot{e(t)} \qquad (19) \qquad\qquad e(t) = (\theta_{\text{d}} - \theta_{\text{l}}) \qquad (20)$$

Na equação (19) encontra-se a equação geral do controlador, com as constantes proporcional e derivativa a actuarem no erro e na derivada do erro respectivamente. A equação (20) define o erro, sendo este a relação entre a saída e a entrada do sistema.

Para obter os parâmetros K_p e K_d do controlador PD fazemos uso da seguinte formula:

$$G(s) = \frac{b_0}{s^2 + a_1 s + a_0} \qquad (21)$$

, onde

$$b_0 = \frac{K_{\text{m}}K_{\text{g}}}{R_{\text{m}}J_{\text{eq}}} = 90.3862 \qquad (22)$$

$$a_0 = 0 \qquad (23)$$

$$a_1 = \frac{(K_{\text{m}}K_{\text{g}})^2}{R_{\text{m}}J_{\text{eq}}} = 49.3509 \qquad (24)$$

são obtidos a partir da comparação entre (10) e de (21).

Após o cálculo desses três coeficientes, calcula-se as contantes K_p e K_d:

- $K_{\text{p}} = \frac{2P\xi\omega_{\text{n}}+{\omega_{\text{n}}}^2-a_0}{b_0} = 64.6032$
- $K_{\text{d}} = \frac{2\xi\omega_{\text{n}}+P-a_1}{b_0} = 0.9461$

4.2 Resultados do controlador PD

Para testar o controlador PD usamos o bloco PID do *Simulink* e atribuimos o valor zero á constante K_i. Aplicámos o controlador PD separadamente aos três modelos do servomotor, utilizando todas as combinações das condições propostas. Nas simulações que efectuamos utilizamos os seguintes valores:

- Referância em rampa: Amplitude = 1 ; $\text{t}_\text{i} = 1$; $\text{t}_\text{f} = 6$
- Perturbação na entrada: Amplitude = 1 ; t = 8
- Filtro no termo derivativo: $K_{\text{p}} + K_{\text{D}} \times \frac{N}{1+N\frac{1}{s}}$,com N = 100

4.2.1 Comparação dos três modelos

Para efeitos de simulação, testamos o desempenho do controlador PD sem filtro na parte derivativa e com perturbação na entrada.

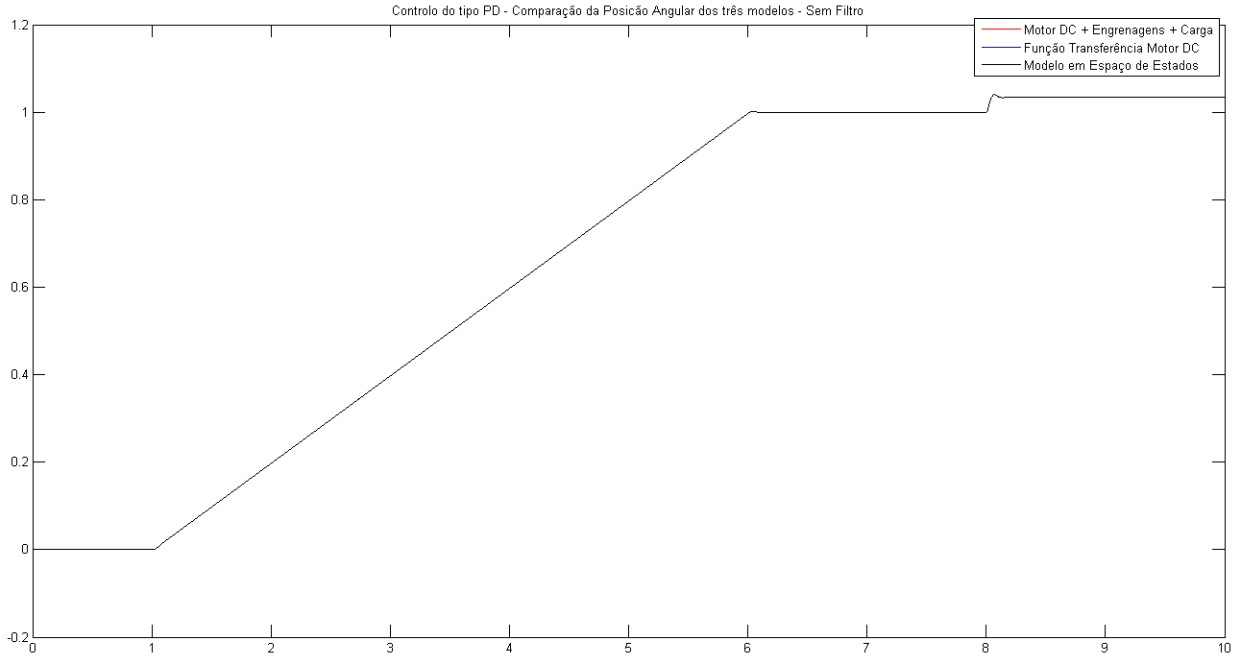

Figura 3: Sobreposição da resposta dos 3 modelos - Posição

Como podemos ver na figura 3, ambos os modelos (Motor Dc + Engrenagens + Efeito da Carga, Função Transferência e o modelo de Espaço de Estados) se sobrepõem. Este resultado é importante para podermos concluir que ambos os três modelos modelam bem o motor DC. No instante t = 8s é injectada a perturbação na carga, analisando a resposta dos três modelos verificamos que o PD não consegue eliminar o erro em regime final. Este efeito é devido ao facto do PD não conter parte integrativa, o que implica que o controlador não consegue eliminar a perturbação injectada.

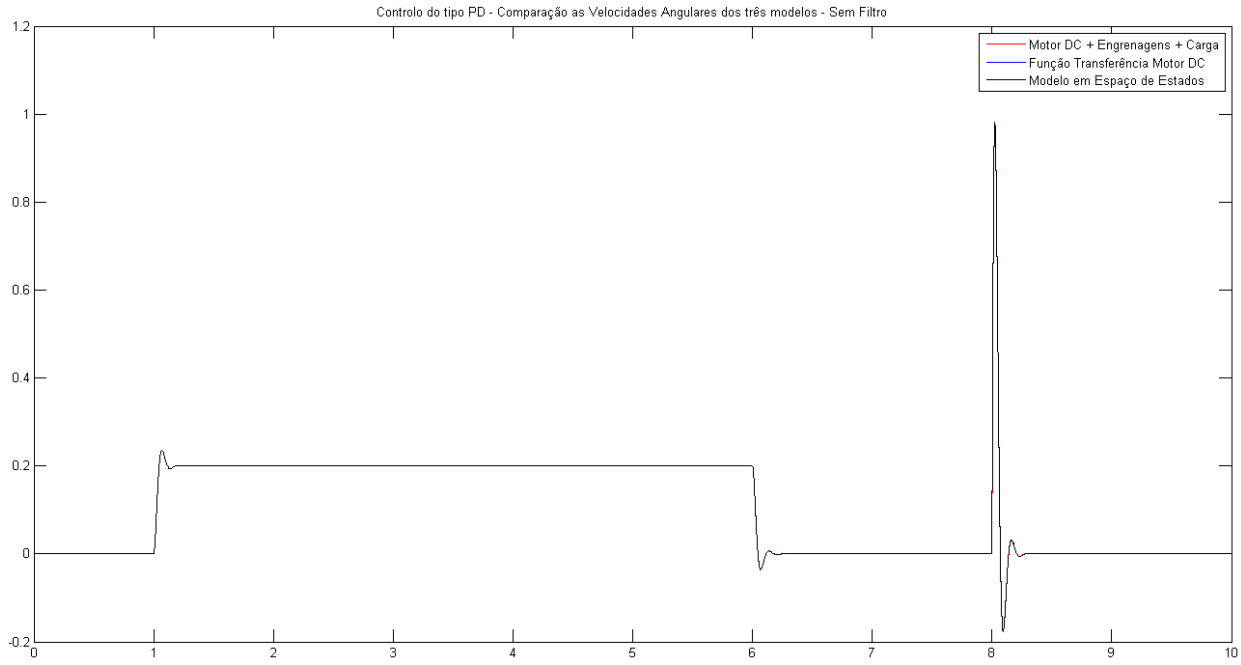

Figura 4: Sobreposição da resposta dos 3 modelos - Velocidade.

Na figura 4 observamos um comportamento normal na velocidade angular do motor exceptuando quando é injectada uma perturbação na carga. Nesse instante a velocidade angular sofre um pico elevado voltando a estabilizar passado algum tempo. De frisar que esta figura sobrepoem os três modelos sendo que ambos se comportam de forma semelhante.

Como era de se esperar, ambos os três modelos têm comportamentos/respostas semelhantes tanto na posição como na velocidade angular.

4.2.2 Referência em rampa e perturbação na entrada

Sem filtro no termo derivativo

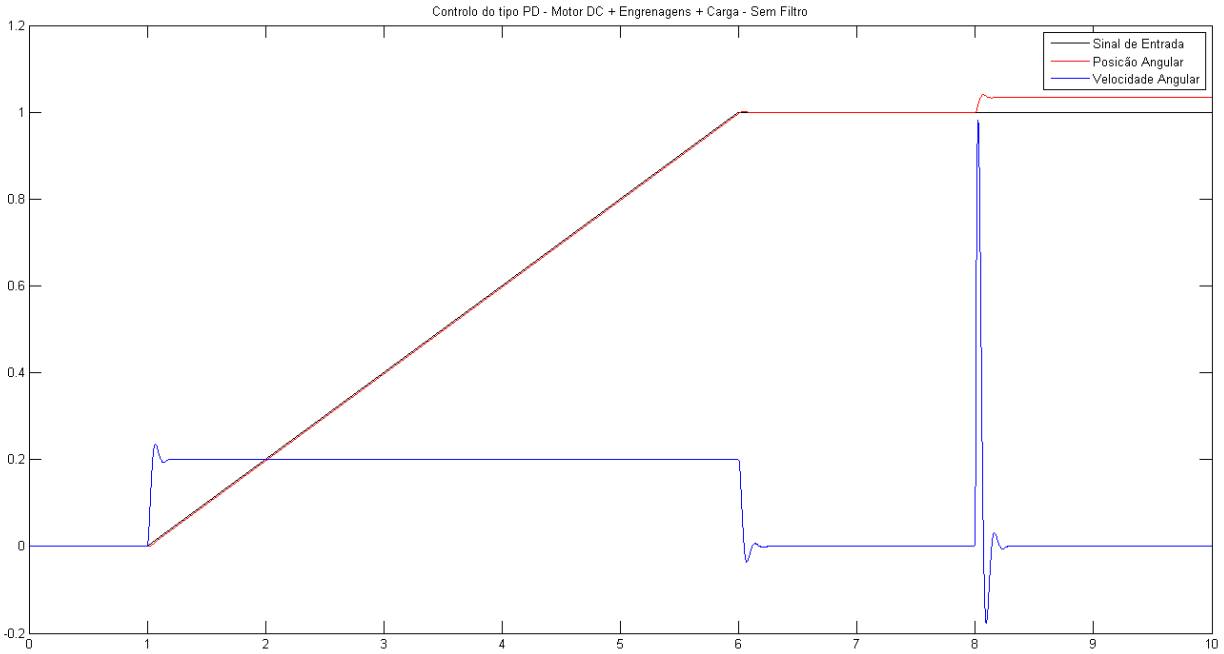

Figura 5: Resposta de qualquer um dos três modelos

O controlador PD segue eficazmente a entrada em rampa, com um pequeno overshoot e um erro em regime permanente nulo. Na resposta à perturbação no instante $t = 8$, pelo contrário, o controlador exibe um erro em regime permanente não desprezível. Esse erro não é eliminado pelo controlador PD sendo que o controlador não converge em regime final para o valor correcto.

O controlador exibe ainda transitórios significativos nos instantes de variação da velocidade quando a perturbação é injectada. Esses transitórios tem valores elevados e indesejados visto que que o motor pode ser danificado com esse efeito.

Com filtro no termo derivativo

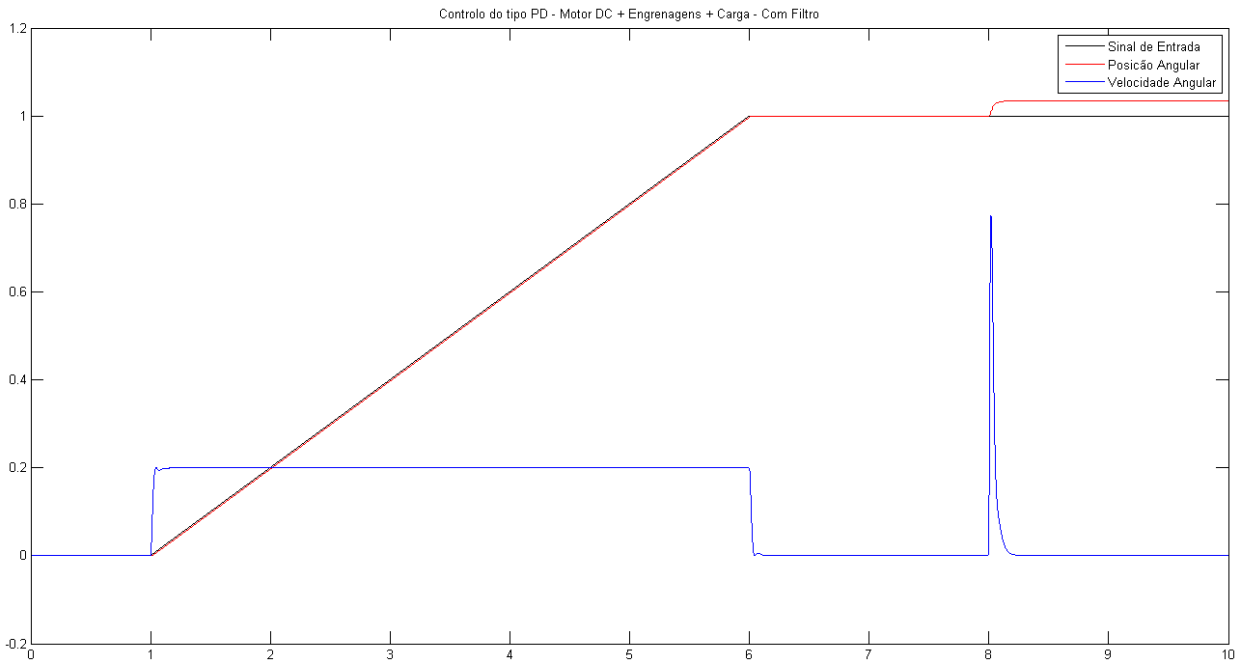

Figura 6: Resposta de qualquer um dos três modelos

Novamente concluímos que ao analisar que a resposta a um sinal de referência em rampa que a resposta dos três modelos para as mesmas condições é a mesma. De frisar que o filtro no termo derivativo do controlador elimina os transitórios da velocidade que eram visíveis na figura 5 (sem filtro). Concluímos que a reposta do sistema é mais suave quando é usado este tipo de filtro.

5 Controlador PID

5.1 Projecto do controlador

Um controlador PID rege-se pelas equações:

$$V_{\mathrm{in}}(t) = K_{\mathrm{p}} e(t) + K_{\mathrm{D}} \dot{e}(t) + \frac{K_{\mathrm{I}}}{e(t)} \tag{25}$$

e pela equação (20).

Na equação (25) encontra-se a equação geral do controlador, com as constantes proporcional, derivativa e integrativa a actuarem no erro, na derivada do erro e no integral do erro respectivamente. A equação (20) define o erro, sendo este a relação entre a saída e a entrada do sistema.

Para obter os parâmetros $\mathrm{K_p}$, $\mathrm{K_D}$ e $\mathrm{K_I}$ do controlador PID, fazemos uso de (22), (23) e (24).

Substituindo esses três parametros nas três equações abaixo, obtém-se:

- $K_{\mathrm{p}} = \frac{2P\xi\omega_{\mathrm{n}} + {\omega_{\mathrm{n}}}^2 - a_0}{b_0} = 64.6032$
- $K_{\mathrm{d}} = \frac{2\xi\omega_{\mathrm{n}} + P - a_1}{b_0} = 0.9461$
- $K_{\mathrm{I}} = \frac{P{\omega_{\mathrm{n}}}^2}{b_0} = 1005.7$

5.2 Resultados do controlador PID

Aplicámos o controlador PID separadamente aos três modelos do servomotor, utilizando todas as combinações das condições propostas.

- Referância em rampa: Amplitude = 1 ; $t_i = 1$; $t_f = 6$
- Perturbação na entrada: Amplitude = 1 ; t = 8
- Filtro no termo derivativo: $K_p + \frac{K_I}{s} + K_D \times \frac{N}{1+N\frac{1}{s}}$,com N = 100

5.2.1 Comparação dos três modelos

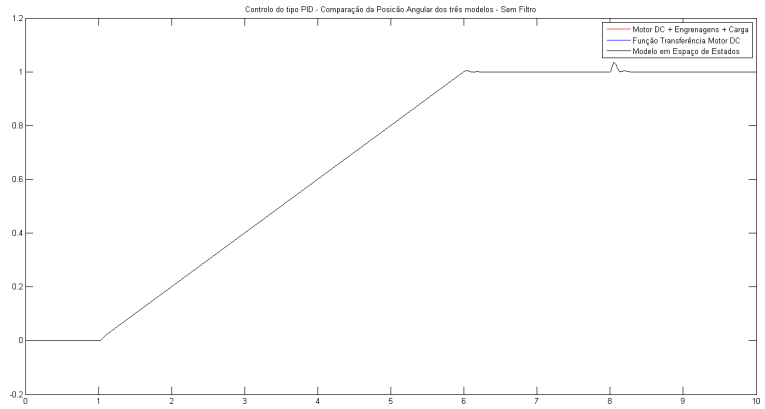

Figura 7: Sobreposição da resposta dos 3 modelos - Posição

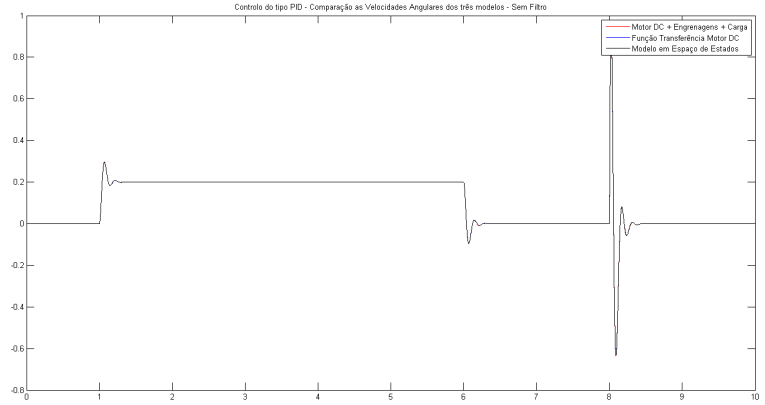

Figura 8: Sobreposição da resposta dos 3 modelos - Velocidade.

Observando as figuras 7 e 8 concluímos que os três modelos têm respostas equivalentes tanto na posição como na velocidade angular. O pico que se observa na figura da velocidade angular é devido á injeção de uma perturbação na entrada nesse instante.

5.2.2 Referência em rampa e perturbação na entrada

Sem filtro no termo derivativo

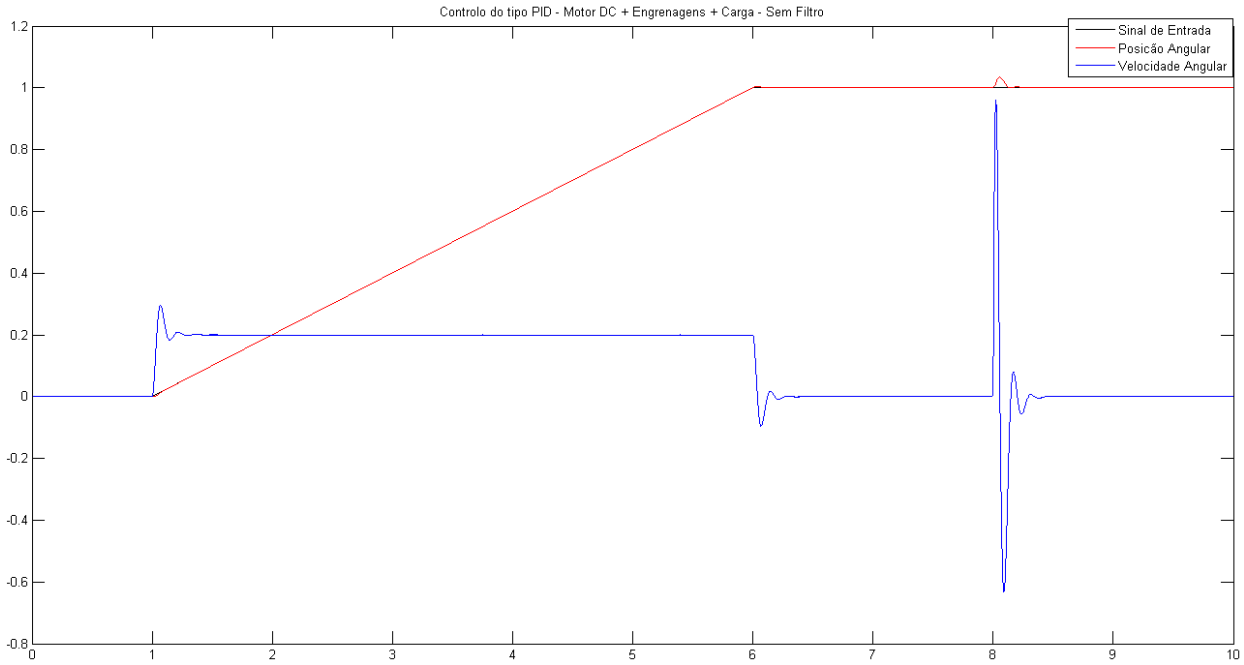

Figura 9: Resposta de qualquer um dos 3 modelos

Como já foi referido todos os modelos têm a mesma resposta à referência em rampa e à perturbação na entrada. Pela análise da velocidade angular e da posição angular é possível verificar que o controlador PID tem uma resposta rápida e que converge rapidamente, não exibindo erro em regime permanente. O mesmo acontece quando se introduz uma perturbação na entrada. A perturbação na entrada quase não afecta a posição angular no entanto gera uma grande variação na velocidade angular que é corrigida pelo controlador. A efeito da perturbação é eliminado através da parte integrativa do PID que elimina o erro em regime final.

Com filtro no termo derivativo

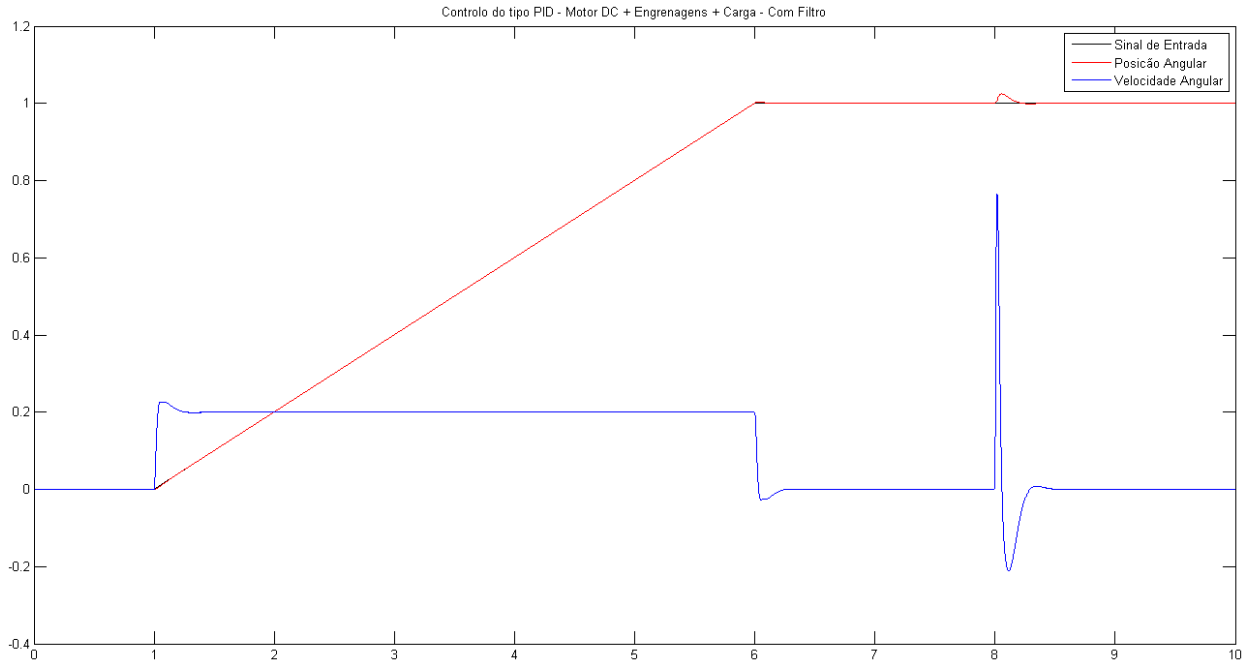

Figura 10: Resposta de qualquer um dos 3 modelos

À semelhança com o comportamento do controlador PD, o filtro no termo derivativo suaviza a resposta do controlador. Entre o instante 5.5 a 6.1 verifica-se um pequeno overshoot que pode ser desprezado. No instante t = 8s é injectada a perturbação que faz disparar a velocidade angular. Após o PID reagir, o erro em regime final é nulo sendo que o PID reage tem o comportamento desejado. Ao contrário do PD, o PID consegue eliminar o erro inserido pela perturbação.
Verificou-se ainda uma semelhança na resposta dos 3 modelos em estudo.

6 Comparação entre controlador Contínuo vs Aproximação Discreta

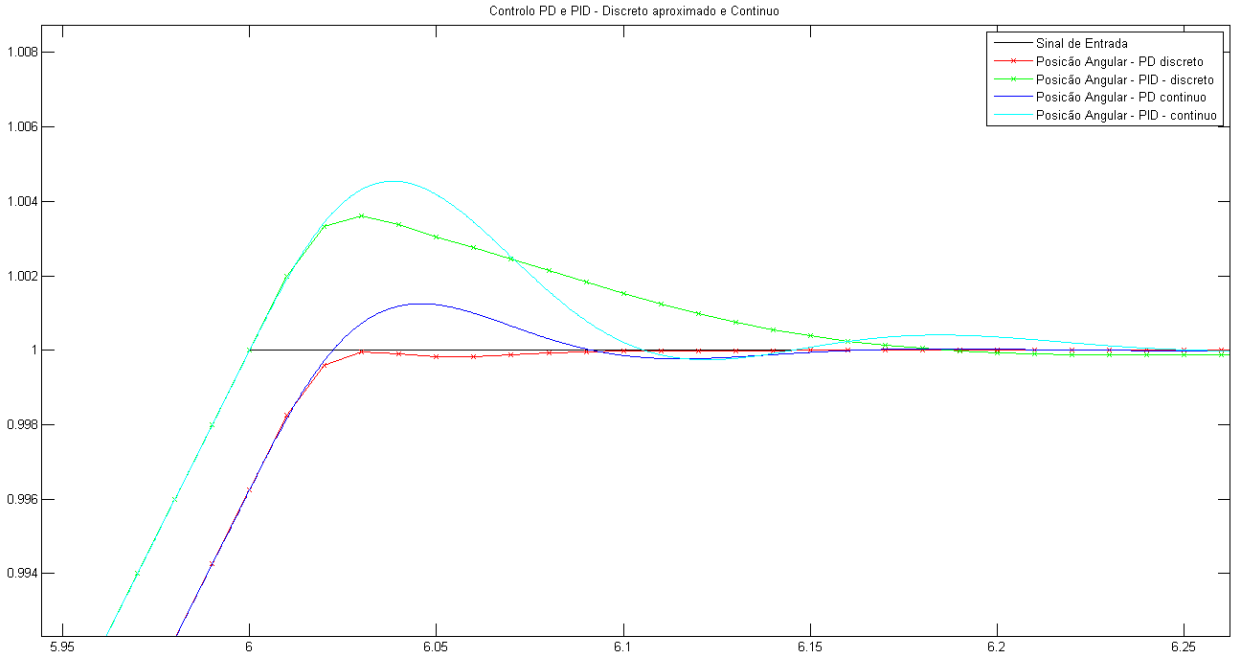

Figura 11: Overshoot na resposta a rampa.

Na figura 11 observamos diferentes overshoots dos diferentes tipos de controladores. O controlador PID discreto e continuo acompanham/seguem bem o a rampa (sinal de referência) enquanto que o PD discreto e continuo não conseguem acompanhar perfeitamente a subida da rampa. Após ocorrer o overshoot, ambos as configurações establizam para o valor de referência final de valor 1.

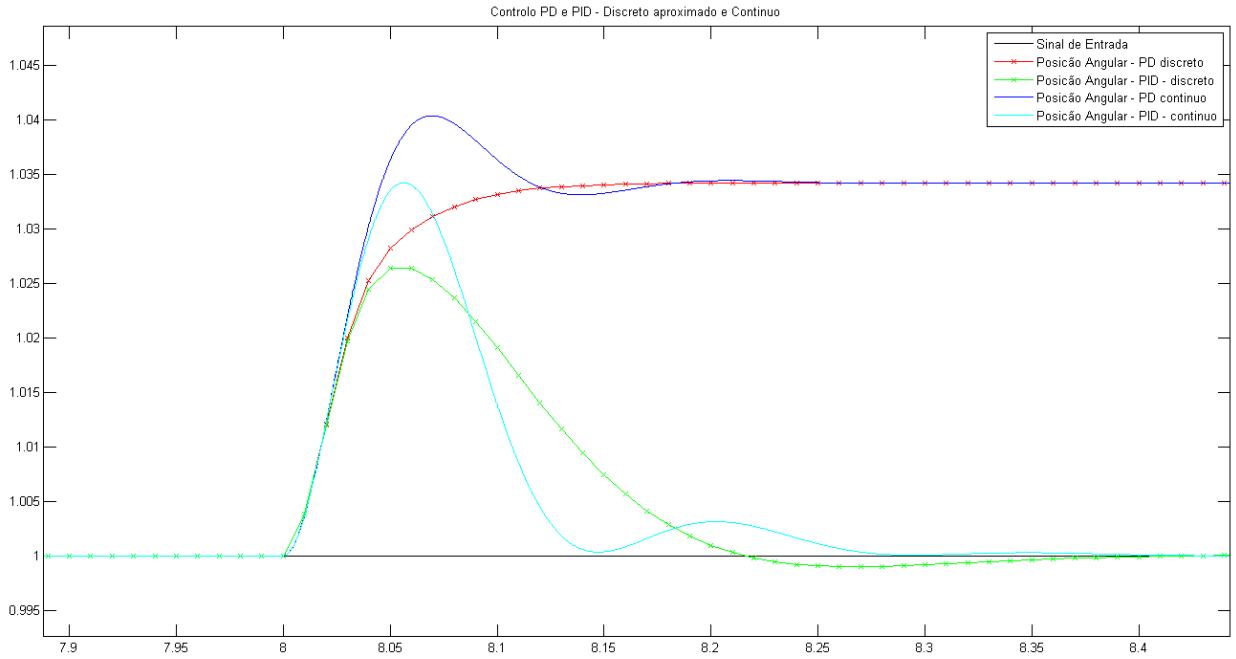

Figura 12: Resposta a perturbação.

Como podemos analisar na figura 12 (resposta á perturbação) verifica-se que os controladores PID discreto e continuo conseguem eliminar o erro em regime final sendo que eliminam perfeitamente a perturbação injectada. Por outro lado, os controladores PD discreto e continuo são ineficazes na eliminação do erro em regime final sendo o valor final da posição angular diferente do da referência. Este tipo de comportamento era expectável.

7 Projecto de observador de estados aumentado

7.1 Dinâmica Deadbeat

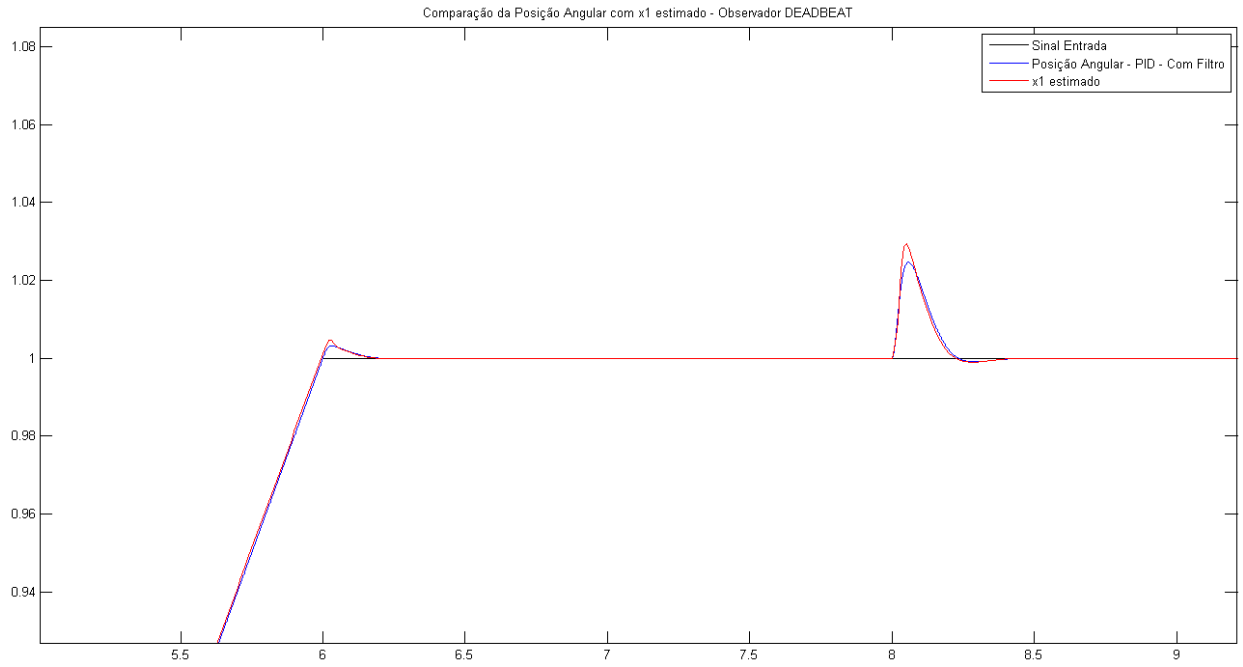

Figura 13: Deadbeat - Estado x1 vs estado x1 estimado.

Na figura 13 verifica-se um pequeno *overshoot* tanto na posição angular do PID como na estimada pelo observador. Após injectarmos uma perturbação na entrada, no instante 8, de magnitude de 1 observa-se um pico na posição angular que após um pequeno intervalo de tempo é corrigido pelo PID. Esse pico era de se esperar devido á introdução da perturbação nesse mesmo instante. Podemos concluir que o PID corrige bem a perturbação.

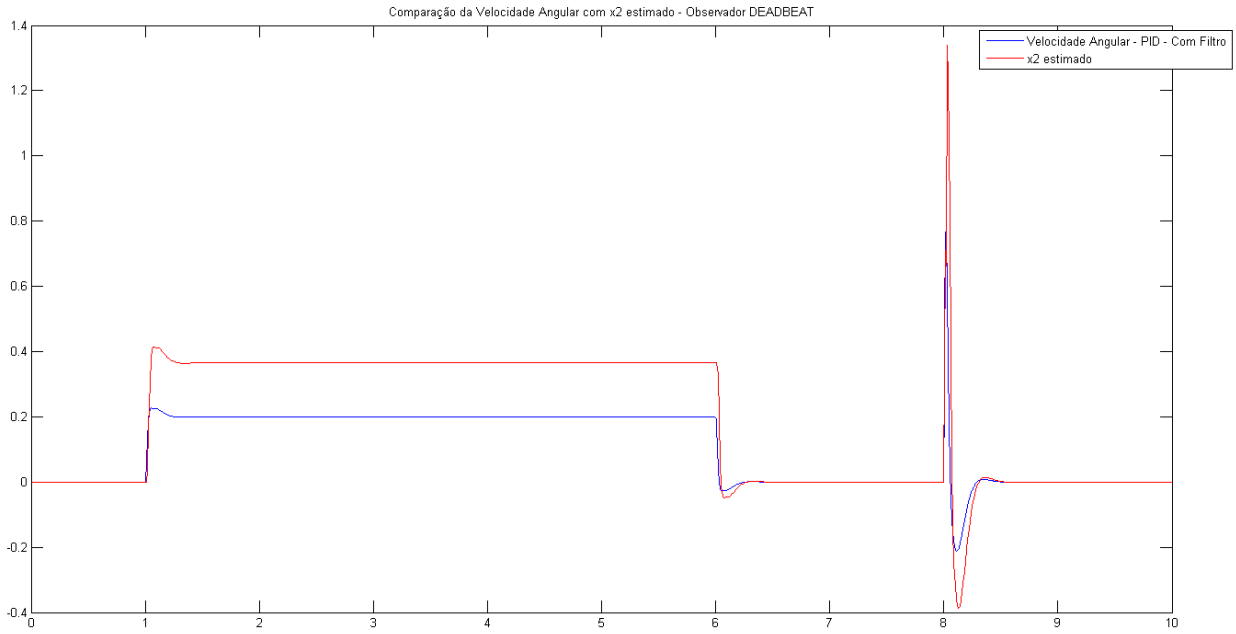

Figura 14: Deadbeat - Estado x2 vs estado x2 estimado.

Na figura 14 verifica-se uma diferença entre a velocidade angular do sistema controlado pelo PID e a velocidade estimada pelo observador. Essa diferença é de aproximadamente 0.2 relativamente ao intervalo de instantes 1 a 6s. Essa pequena diferença pode ser ignorada visto que não é de grande ordem. Quando é injectada a perturbação, no instante 8s, observamos um pico na velocidade angular que é corrigido no instante 8.4 para a frente. Comparando a velocidade angular estimada e a velocidade angular obtida através do sistema podemos inferir que a estimação feita pelo observador de *deadbeat* é boa.

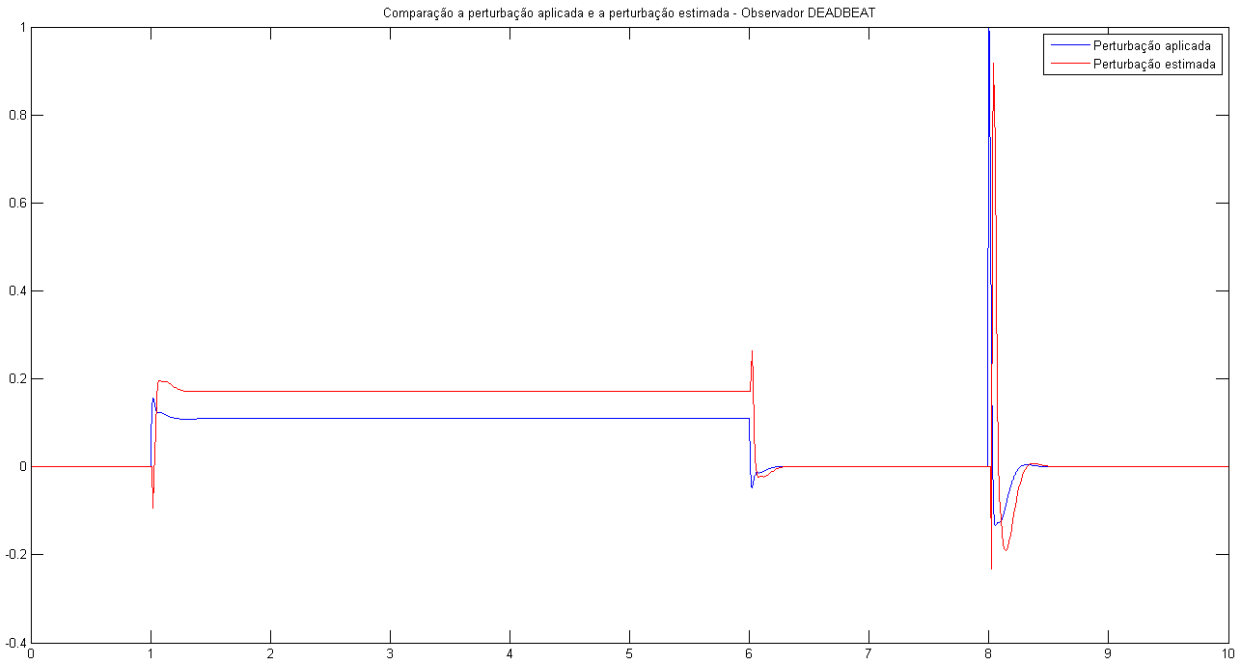

Figura 15: Deadbeat - Perturbação vs Perturbação estimada.

Na figura 15 observa-se a perturbação que está a ser efectivamente aplicada á entrada e a sua estimação através do estimador de estado aumentado. No intervalo de 1 a 6 é aplicado o degrau em rampa de amplitude 1. Visualizamos uma pequena diferença entre a perturbação aplicada e a estimada nesse periodo o que é normal. No instante 8 injectamos a perturbação na entrada, sendo que tanto o estado da perturbação estimado como a perturbação sofrem um pico severo nesse momento. Após esse pico a perturbação estabiliza para um valor nulo como era de se esperar, ou seja, é eliminada a perturbação.

Código matlab:

```
%Observador preditor de estado aumentado
phiw = 1;
phixw = gama;
phi_a = [phi phixw; 0 0 phiw];
gama_a = [gama;0];
C_a = [C 0] ;
%Matriz de observabilidade
w0=[C_a; C_a*phi_a; C_a*phi_a^2];
w0i=inv(w0);
%Vector de ganhos do observador DEADBEAT
Ko=phi_a^3*w0i*[0; 0; 1] ;
%valor inicial do estado observado
xobs=[0;0;0];
```

7.2 Dinâmica dominante de segunda ordem

Para fins de comparação, decidimos usar a mesma dinâmica dominante tanto no PID como no observador.

Equação Caracteristica do sistema com dinâmica de Segunda Ordem:

$$(s^2 + 2\zeta\omega_n s + \omega_n^2)(s + P) = 0 \tag{26}$$

$$= s^3 + (2\zeta\omega_n s + P)s^2 + (\omega_n^2 + 2\zeta\omega_n P)s + \omega_n^2 P \tag{27}$$

Equação Caracteristica do Sistema com PID:

$$s^3 + (a_1 + b_0 Kd)s^2 + (a_0 + b_0 Kp)s + b_0 Ki \tag{28}$$

Sendo que:

- $\zeta = 0.7$
- $\omega_n = 21.9955$
- $P = 70$

Por comparação das equações (27) e (28) determinamos as constantes Ki, Kp e Kd do PID. Para o observador fazemos uso da equação (26) e utilizamos a função *conv* do Matlab para a multiplicação.
Esperamos nos resultados que as variaveis de posição e velocidade e os estados estimados tenham comportamentos semelhantes visto que o observador e o PID foram projectados para o sistema ter uma dinãmica de segunda ordem idêntica.

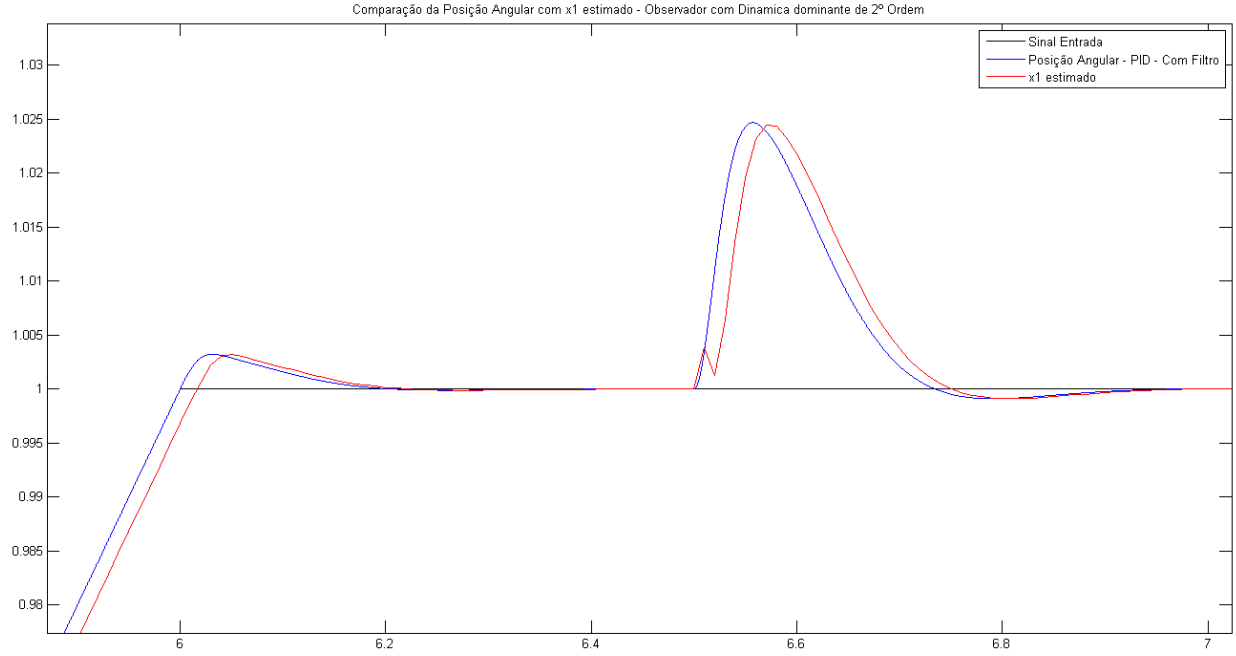

Figura 16: Obs. de 2ª ordem - Estado x1 vs estado x1 estimado.

Na figura 16 observa-se um pequeno overshoot de valor aproximadamente 1 em ambas as configurações (PID e Observador). Após o sistema ter estabilizado entre 6.2 e 6.4s é injectada uma perturbação na entrada (degrau unitário) no instante 6.5. Vemos que a posição angular e a sua estimação são afectadas pela perturbação entre o instante 6.5 e o instante 6.8. Após esse instante o PID corrige a posição angular sendo que esta estabiliza para o valor nominal de 1 como é esperado. A parte integradora do PID anula o erro em regime final.

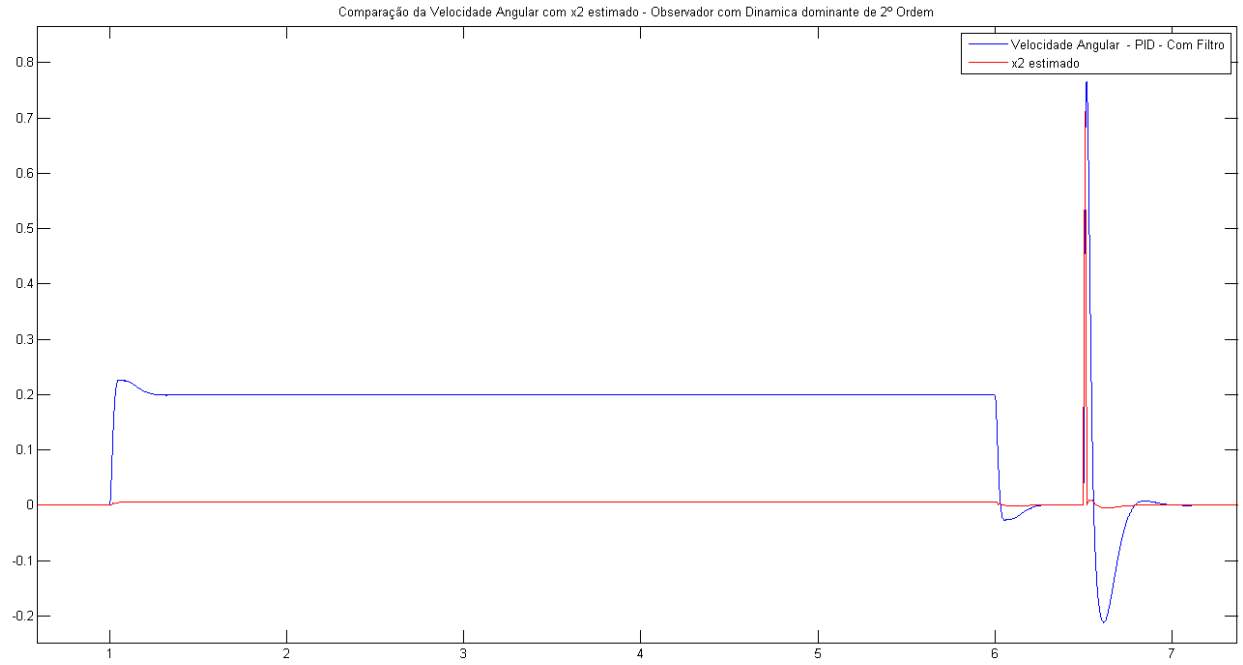

Figura 17: Obs. de 2ª ordem - Estado x2 vs estado x2 estimado.

Na figura 17 vemos uma pequena diferença entre o estado observado e a velocidade angular

do PID de valor 0.2 entre 1 e 6s. Essa pequena diferença deve-se ao facto da entrada (degrau em rampa) se dar entre o intervalo 1 a 6. Quando injectamos a perturbação na entrada, instante 6.5, vemos um pico na velocidade angular que é corrigido quase de imediato pelo PID. Comparando o estado estimado e a velocidade angular podemos inferir que o estimador comporta-se de forma adequada e estima bem a velocidade angular.

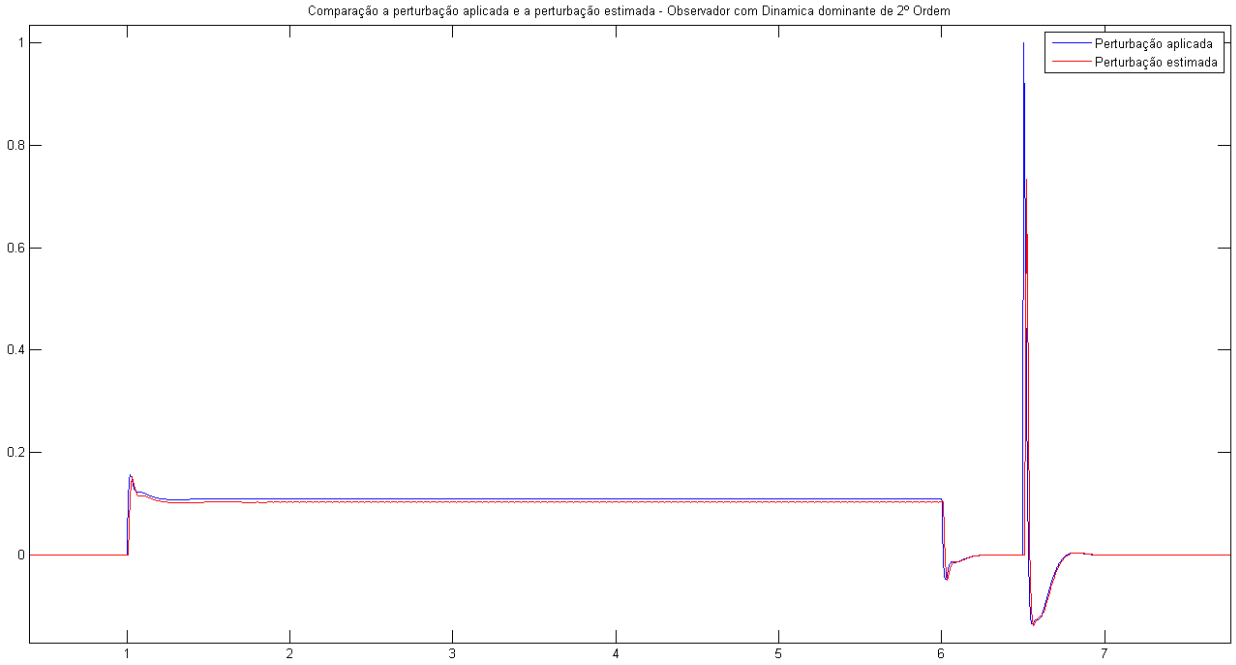

Figura 18: Obs. de 2ª ordem - Perturbação vs Perturbação estimada.

Na figura 18 juntamos a perturbação que aplicamos na entrada e a perturbação estimada pelo observador preditor de estado aumentado. Como é possivel observar a perturbação estimada coincide quase perfeitamente com a perturbação aplicada na entrada. Podemos concluir que o observador está a estimar bem o estado da perturbação.

8 Teste de robustez

Utilizando o observador Dedbeat com filtro e sem perturbacao na entrada.

8.1 Perturbação aditiva no binário

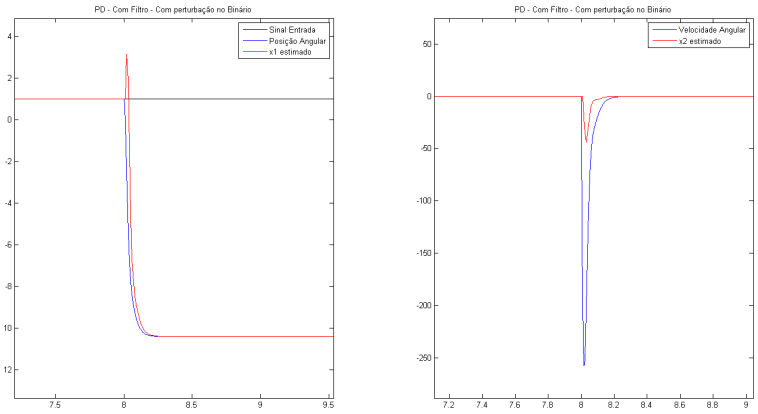

Figura 19: PD - Perturbação no binário.

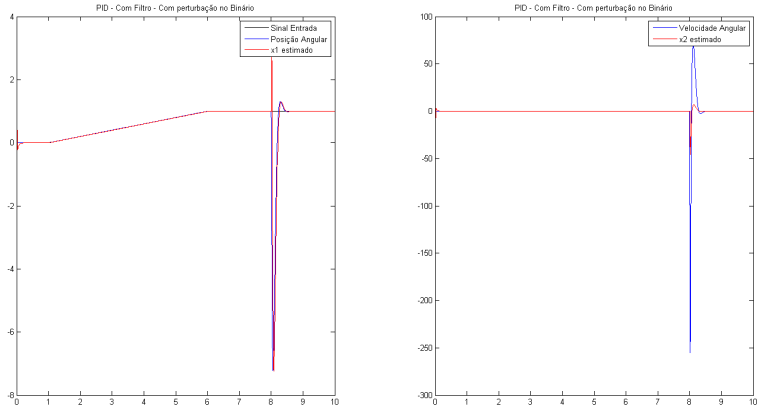

Figura 20: PID - Perturbação no binário.

- **Controlador PD**
  Verifica-se um overshoot na estimação da posição, mas que converge em regime permanenente.
  Resposta da velocidade estimada mais rápida que a velocidade real.

- **Controlador PID**
  Observa-se um overshoot na estimação da posição.
  Pico negativo na estimação da velocidade.

8.2 Variações na carga

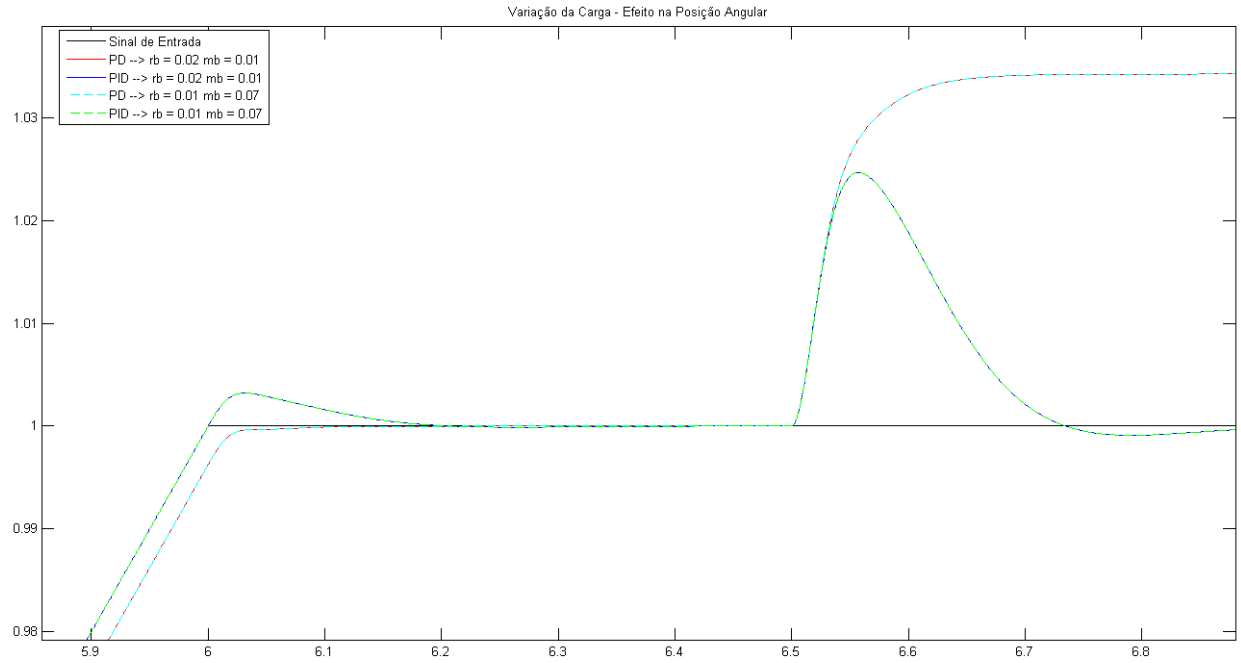

Figura 21: Variação da carga - Efeito na posição angular.

Nenhum dos controladores é afectado pela alteração na carga. A resposta de ambos é igual para os dois casos.

O controlador PD exibe uma resposta lenta à rampa e um erro em regime permanente após a perturbação da entrada. Por sua vez o controlador PID converge rapidamente, tanto após a rampa, como após a perturbação, mas gera um overshoot elevado.

9 Controlador em espaço de estados com realimentação dos estados estimados por observador

9.1 Perturbação na entrada

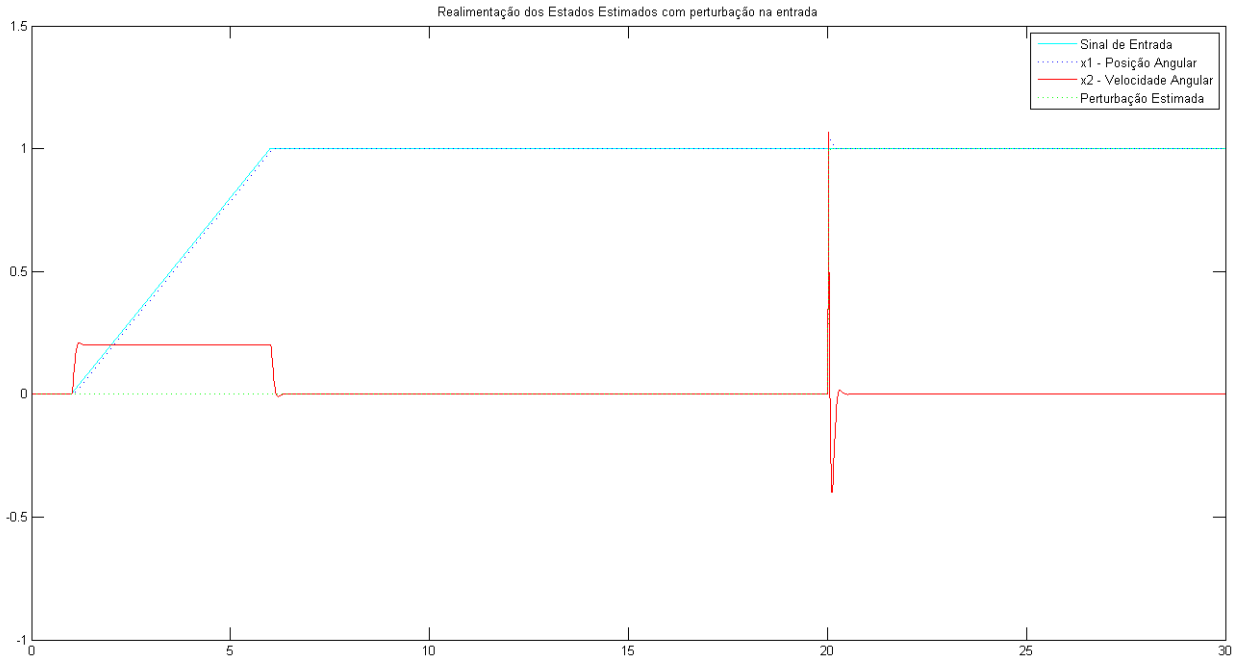

Figura 22: Realimentação dos estados estimados - Perturbação na entrada.

Na figura 22 verificamos que o estado estimado x_1 - Posição Angular - segue bem o sinal de referência (rampa). O estado x_2 - Velocidade Angular - apresenta valores coerentes sendo constante enquanto a rampa sobe e anula-se quando o efeito da rampa desaparece, tal como era esperado. Quando é injectada a perturbação na entrada no instante t = 20s (degrau unitário de valor 1) a velocidade angular dispara e a posição angular sofre um pequeno overshoot que é eliminado logo após 0.4s. Pode-se observar que o erro em regime estacionário é nulo o que indicia que o sistema realimentado pelas variáveis de estado tem comportamento integrativo do erro.

9.2 Perturbação no binário

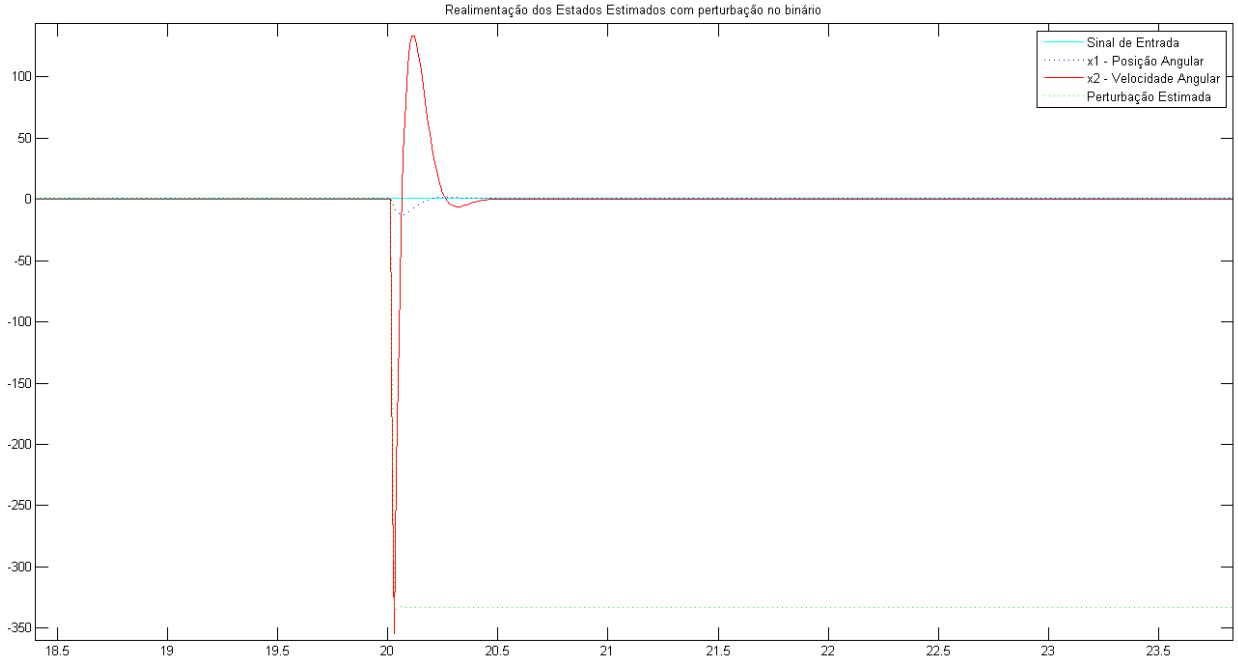

Figura 23: Realimentação dos estados estimados - Perturbação no binário.

Como era de se esperar, quando se introduz uma perturbação no binário (no instante t = 20s), a velocidade angular dispara para conseguir estabilizar o sistema. A perturbação surte o seu efeito e prolonga-se durante 0.5s até ser totalmente eliminada. Este pico na velocidade angular (Estado x_2 é elevado sendo que em termos de sistema global é preciso ter em atenção este efeito nefasto. Concluimos que a realimentação com os estados do processo tem um bom desempenho sendo que anulam o efeito da perturbação do binário rapidamente e estabilizam o sistema.

10 Anexos

10.1 Código Matlab

```
warning('off','all');
clear
close all
clc
%-------------------------------------------------------------------------
%% Vari?veis
global xobs_2 Ko_2 C_a_2 phi_a_2 gama_a_2; % Realimentar
global xobs Ko C_a phi_a gama_a; % Observador de estado aumentado - Observar
Ko=[0 0 0]';
C_a = [0 0 0];
phi_a = zeros(3,3);
gama_a = zeros(3,1);
xobs=[0;0;0];
Ko_2=[0 0 0]';
C_a_2 = [0 0 0];
phi_a_2 = zeros(3,3);
gama_a_2 = zeros(3,1);
xobs_2=[0;0;0];
La = zeros(1,3);
Lc = 0;
perturbacao_fv = 1;
perturbacao_st = 8;
perturbacao_bin_fv = 0;
perturbacao_bin_st = 0;
%-------------------------------------------------------------------------
%% Alineas 1,2,3,4,5
Rm=2.6;%ohms
Km=0.0078;%Nm/A
Lm=0.00018;%H
Kgi=14; %relacao entre engrenagens e caixa redutora
Kge=5; %relacao entre engrenagens do trem externo
Jmrotor=3.9e-7; %kgm^2
Jm=4.60625e-7; %Kg.m^2
JL=6.63e-5; %Kg.m^2
Jg=5.28e-5; %Kg.m2;
Kg=Kge*Kgi;
Jeq= Kg^2*Jm +JL;
Ksi=0.7;
tp=0.2;
wd=pi/tp;
wn=wd/sqrt(1-Ksi^2);
h = 0.01;
num = 1;
den = [(Rm*Jeq)/(Km*Kg) Km*Kg];
a0 = 0;
b0 = (Km*Kg)/(Rm*Jeq);
a1 = ((Km*Kg)^2)/(Rm*Jeq);

P = 70;

Kp = ((2*P*Ksi*wn)+wn^2-a0)/(b0);
Kd = ((2*Ksi*wn)+P-a1)/(b0);
Ki = 0;

Jl=JL;
Kf=0;
Ke=Km;
```

```
Bl=0;
% Modelo do motor State Space
A = [0  1; 0  -0.5460/0.0111];
B = [0  1/0.0111]';
C = [1  0];
D = 0;

Ca = [1  0; 0  1];
Da = [0  0]';
C1 = [1  0];
C2 = [0  1];
% Fim do motor State Space
%-------------------------------------------------------------------------
%% Alinea 4
% 1)
filtro = 0;
sim('servermotor');
% Motor DC + Engrenagens + Carga
figure
plot(sinal_entrada.time, sinal_entrada.signals.values,'k',teta_1.time, teta_1.
    signals.values,'r',teta_ponto_1.time, teta_ponto_1.signals.values,'b');
legend('Sinal de Entrada','Posicao Angular','Velocidade Angular');
title('Controlo do tipo PD - Motor DC + Engrenagens + Carga - Sem Filtro')
% Funcao Transferencia Motor DC
figure
plot(sinal_entrada.time, sinal_entrada.signals.values,'k',teta_2.time, teta_2.
    signals.values,'r',teta_ponto_2.time, teta_ponto_2.signals.values,'b');
legend('Sinal de Entrada','Posicao Angular','Velocidade Angular');
title('Controlo do tipo PD - Funcao Transferencia Motor DC - Sem Filtro ')
% Espaco de Estados
figure
plot(sinal_entrada.time, sinal_entrada.signals.values,'k',x1.time, x1.signals.values
    ,'r',x2.time, x2.signals.values,'b');
legend('Sinal de Entrada','Posicao Angular','Velocidade Angular');
title('Controlo do tipo PD - Modelo em Espaco de Estados - Sem Filtro');
% Comparacao entre as Posicoes Angulares dos tres modelos
figure
plot(teta_1.time, teta_1.signals.values,'r',teta_2.time, teta_2.signals.values,'b',
    x1.time, x1.signals.values,'k');
legend('Motor DC + Engrenagens + Carga','Funcao Transferencia Motor DC','Modelo em
    Espaco de Estados');
title('Controlo do tipo PD - Comparacao da Posicao Angular dos tres modelos - Sem
    Filtro');
% Comparacao entre as Velocidades Angulares dos tres modelos
figure
plot(teta_ponto_1.time, teta_ponto_1.signals.values,'r',teta_ponto_2.time,
    teta_ponto_2.signals.values,'b',x2.time, x2.signals.values,'k');
legend('Motor DC + Engrenagens + Carga','Funcao Transferencia Motor DC','Modelo em
    Espaco de Estados');
title('Controlo do tipo PD - Comparacao as Velocidades Angulares dos tres modelos -
    Sem Filtro');
% 2)
filtro = 100;
sim('servermotor');
% Motor DC + Engrenagens + Carga
figure
plot(sinal_entrada.time, sinal_entrada.signals.values,'k',teta_1.time, teta_1.
    signals.values,'r',teta_ponto_1.time, teta_ponto_1.signals.values,'b');
legend('Sinal de Entrada','Posicao Angular','Velocidade Angular');
title('Controlo do tipo PD - Motor DC + Engrenagens + Carga - Com Filtro')
% Funcao Transferencia Motor DC
```

```
figure
plot(sinal_entrada.time, sinal_entrada.signals.values,'k',teta_2.time, teta_2.signals.values,'r',teta_ponto_2.time, teta_ponto_2.signals.values,'b');
legend('Sinal de Entrada','Posicao Angular','Velocidade Angular');
title('Controlo do tipo PD - Funcao Transferencia Motor DC - Com Filtro ')
% Espaco de Estados
figure
plot(sinal_entrada.time, sinal_entrada.signals.values,'k',x1.time, x1.signals.values,'r',x2.time, x2.signals.values,'b');
legend('Sinal de Entrada','Posicao Angular','Velocidade Angular');
title('Controlo do tipo PD - Modelo em Espaco de Estados - Com Filtro');
% Comparacao entre as Posic??es Angulares dos tres modelos
figure
plot(teta_1.time, teta_1.signals.values,'r',teta_2.time, teta_2.signals.values,'b',x1.time, x1.signals.values,'k');
legend('Motor DC + Engrenagens + Carga','Funcao Transferencia Motor DC','Modelo em Espaco de Estados');
title('Controlo do tipo PD - Comparacao da Posicao Angular dos tres modelos - Com Filtro');
% Comparacao entre as Velocidades Angulares dos tres modelos
figure
plot(teta_ponto_1.time, teta_ponto_1.signals.values,'r',teta_ponto_2.time, teta_ponto_2.signals.values,'b',x2.time, x2.signals.values,'k');
legend('Motor DC + Engrenagens + Carga','Funcao Transferencia Motor DC','Modelo em Espaco de Estados');
title('Controlo do tipo PD - Comparacao as Velocidades Angulares dos tres modelos - Com Filtro');
%-------------------------------------------------------------------------
% Alinea 5
% 1)
filtro = 0;
Ki = (P*wn^2)/(b0); % A parte integradora num PID serve para anular o erro em regime final
sim('servermotor');
% Motor DC + Engrenagens + Carga
figure
plot(sinal_entrada.time, sinal_entrada.signals.values,'k',teta_1.time, teta_1.signals.values,'r',teta_ponto_1.time, teta_ponto_1.signals.values,'b');
legend('Sinal de Entrada','Posicao Angular','Velocidade Angular');
title('Controlo do tipo PID - Motor DC + Engrenagens + Carga - Sem Filtro')
% Funcao Transferencia Motor DC
figure
plot(sinal_entrada.time, sinal_entrada.signals.values,'k',teta_2.time, teta_2.signals.values,'r',teta_ponto_2.time, teta_ponto_2.signals.values,'b');
legend('Sinal de Entrada','Posicao Angular','Velocidade Angular');
title('Controlo do tipo PID - Funcao Transferencia Motor DC - Sem Filtro ')
% Espaco de Estados
figure
plot(sinal_entrada.time, sinal_entrada.signals.values,'k',x1.time, x1.signals.values,'r',x2.time, x2.signals.values,'b');
legend('Sinal de Entrada','Posicao Angular','Velocidade Angular');
title('Controlo do tipo PID - Modelo em Espaco de Estados - Sem Filtro');
% Comparacao entre as Posic??es Angulares dos tres modelos
figure
plot(teta_1.time, teta_1.signals.values,'r',teta_2.time, teta_2.signals.values,'b',x1.time, x1.signals.values,'k');
legend('Motor DC + Engrenagens + Carga','Funcao Transferencia Motor DC','Modelo em Espaco de Estados');
title('Controlo do tipo PID - Comparacao da Posicao Angular dos tres modelos - Sem Filtro');
% Comparacao entre as Velocidades Angulares dos tres modelos
```

```
figure
plot(teta_ponto_1.time, teta_ponto_1.signals.values,'r',teta_ponto_2.time,
    teta_ponto_2.signals.values,'b',x2.time, x2.signals.values,'k');
legend('Motor DC + Engrenagens + Carga','Funcao Transferencia Motor DC','Modelo em
    Espaco de Estados');
title('Controlo do tipo PID - Comparacao as Velocidades Angulares dos tres modelos -
     Sem Filtro');
% 2)
filtro = 100;
sim('servermotor');
% Motor DC + Engrenagens + Carga
figure
plot(sinal_entrada.time, sinal_entrada.signals.values,'k',teta_1.time, teta_1.
    signals.values,'r',teta_ponto_1.time, teta_ponto_1.signals.values,'b');
legend('Sinal de Entrada','Posicao Angular','Velocidade Angular');
title('Controlo do tipo PID - Motor DC + Engrenagens + Carga - Com Filtro')
% Funcao Transferencia Motor DC
figure
plot(sinal_entrada.time, sinal_entrada.signals.values,'k',teta_2.time, teta_2.
    signals.values,'r',teta_ponto_2.time, teta_ponto_2.signals.values,'b');
legend('Sinal de Entrada','Posicao Angular','Velocidade Angular');
title('Controlo do tipo PID - Funcao Transferencia Motor DC - Com Filtro ')
% Espaco de Estados
figure
plot(sinal_entrada.time, sinal_entrada.signals.values,'k',x1.time, x1.signals.values
    ,'r',x2.time, x2.signals.values,'b');
legend('Sinal de Entrada','Posicao Angular','Velocidade Angular');
title('Controlo do tipo PID - Modelo em Espaco de Estados - Com Filtro');
% Comparacao entre as Posic??es Angulares dos tres modelos
figure
plot(teta_1.time, teta_1.signals.values,'r',teta_2.time, teta_2.signals.values,'b',
    x1.time, x1.signals.values,'k');
legend('Motor DC + Engrenagens + Carga','Funcao Transferencia Motor DC','Modelo em
    Espaco de Estados');
title('Controlo do tipo PID - Comparacao da Posicao Angular dos tres modelos - Com
    Filtro');
% Comparacao entre as Velocidades Angulares dos tres modelos
figure
plot(teta_ponto_1.time, teta_ponto_1.signals.values,'r',teta_ponto_2.time,
    teta_ponto_2.signals.values,'b',x2.time, x2.signals.values,'k');
legend('Motor DC + Engrenagens + Carga','Funcao Transferencia Motor DC','Modelo em
    Espaco de Estados');
title('Controlo do tipo PID - Comparacao as Velocidades Angulares dos tres modelos -
     Com Filtro');
%-------------------------------------------------------------------------
%% Alinea 4-5 --> Comportamento PD continuo-discreto e PID continuo-discreto
%Comparacao entre PD e PID aproximado discreto com PD e PID continuo
% PD continuo - Sem Filtro
filtro = 0;
Ki = 0;
sim('servermotor');
teta_PD = teta_1;
% PID continuo - Sem Filtro
filtro = 0;
Ki = (P*wn^2)/(b0); % A parte integradora num PID serve para anular o erro em regime
     final
sim('servermotor');
teta_PID = teta_1;
sim('Discreto');
figure
```

```
plot(sinal_entrada.time, sinal_entrada.signals.values,'k',teta_3.time, teta_3.
    signals.values,'r-x',teta_4.time, teta_4.signals.values,'g-x',teta_PD.time,
    teta_PD.signals.values,'b',teta_PID.time, teta_PID.signals.values,'c')
legend('Sinal de Entrada','Posicao Angular - PD discreto','Posicao Angular - PID -
    discreto','Posicao Angular - PD continuo','Posicao Angular - PID - continuo');
title('Controlo PD e PID - Discreto aproximado e Continuo')
%-------------------------------------------------------------------------
%% Alinea 6
%%Modelo de estado do sistema discreto:
[phi gama] = c2d(A,B,h);
%Observador preditor de estado aumentado
phiw = 1;
phixw = gama;
phi_a = [phi phixw; 0 0 phiw];
gama_a = [gama;0];
C_a = [C 0] ;
%Matriz de observabilidade
w0=[C_a; C_a*phi_a; C_a*phi_a^2];
w0i=inv(w0);
%Vector de ganhos do observador DEADBEAT
Ko=phi_a^3*w0i*[0; 0; 1] ;
%valor inicial do estado observado
xobs=[0;0;0];
% DEADBEAT
filtro = 100;
Ki = (P*wn^2)/(b0); % A parte integradora num PID serve para anular o erro em regime
    final
sim('servermotor');
figure
plot(sinal_entrada.time, sinal_entrada.signals.values,'k',teta_1.time, teta_1.
    signals.values,'b',x1_estimado.time, x1_estimado.signals.values,'r-');
legend('Sinal Entrada','Posicao Angular - PID - Com Filtro','x1 estimado');
title('Comparacao da Posicao Angular com x1 estimado - Observador DEADBEAT');
figure
plot(teta_ponto_1.time,teta_ponto_1.signals.values,'b',x2_estimado.time, x2_estimado
    .signals.values,'r-');
legend('Velocidade Angular - PID - Com Filtro','x2 estimado');
title('Comparacao da Velocidade Angular com x2 estimado - Observador DEADBEAT');
figure
plot(perturbacao.time, perturbacao.signals.values,'b',p_estimado.time, p_estimado.
    signals.values,'r-');
legend('Perturbacao aplicada','Perturbacao estimada');
title('Comparacao a perturbacao aplicada e a perturbacao estimada - Observador
    DEADBEAT');
%-------------------------------------------------------------------------
%% DINAMICA DEFINIDA ATRAV??S DA EQUA????O: (s^2+2*Ksi*wn*s + (wn)^2)*(s+70) = 0
% Queremos a dinamica igual ao do controlador PID da alinea 5.
den=conv([1 2*Ksi*wn (wn)^2], [1 70]);
[a,b,c,d]=tf2ss([0 0 0 1],den);
[phio,gamao]=c2d(a,b,h);
po=eig(phio);
Ko=acker(phi_a',C_a',po);
Ko=Ko';
xobs=[0;0;0];
filtro = 100;
Ki = (P*wn^2)/(b0); % A parte integradora num PID serve para anular o erro em regime
    final
sim('servermotor');
figure
plot(sinal_entrada.time, sinal_entrada.signals.values,'k',teta_1.time, teta_1.
    signals.values,'b',x1_estimado.time, x1_estimado.signals.values,'r-');
```

```
253 legend('Sinal Entrada','Posicao Angular - PID - Com Filtro','x1 estimado');
    title('Comparacao da Posicao Angular com x1 estimado - Observador com Dinamica
        dominante de 2?? Ordem');
255 figure
    plot(teta_ponto_1.time, teta_ponto_1.signals.values,'b',x2_estimado.time,
        x2_estimado.signals.values,'r-');
257 legend('Velocidade Angular  - PID - Com Filtro','x2 estimado');
    title('Comparacao da Velocidade Angular com x2 estimado - Observador com Dinamica
        dominante de 2?? Ordem');
259 figure
    plot(perturbacao.time, perturbacao.signals.values,'b',p_estimado.time,-p_estimado.
        signals.values,'r-');
261 legend('Perturbacao aplicada','Perturbacao estimada');
    title('Comparacao a perturbacao aplicada e a perturbacao estimada - Observador com
        Dinamica dominante de 2?? Ordem');
263 %-------------------------------------------------------------------------
    %% Alinea 7.1
265 perturbacao_bin_fv = 1;
    perturbacao_bin_st = 8;
267 perturbacao_fv = 0;
    perturbacao_st = 0;
269 % PID
    filtro = 100;
271 Ki = (P*wn^2)/(b0); % A parte integradora num PID serve para anular o erro em regime
        final
    sim('servermotor');
273 figure
    subplot(1,2,1)
275 plot(sinal_entrada.time, sinal_entrada.signals.values,'k',teta_1.time, teta_1.
        signals.values,'b',x1_estimado.time, x1_estimado.signals.values,'r-');
    legend('Sinal Entrada','Posicao Angular','x1 estimado');
277 title('PID - Com Filtro - Com perturbacao no Binario');
    subplot(1,2,2)
279 plot(teta_ponto_1.time, teta_ponto_1.signals.values,'b',x2_estimado.time,
        x2_estimado.signals.values,'r-');
    legend('Velocidade Angular','x2 estimado');
281 title('PID - Com Filtro - Com perturbacao no Binario');
    % PD
283 filtro = 100;
    Ki = 0;
285 sim('servermotor');
    figure
287 title('PD')
    subplot(1,2,1)
289 plot(sinal_entrada.time, sinal_entrada.signals.values,'k',teta_1.time, teta_1.
        signals.values,'b',x1_estimado.time, x1_estimado.signals.values,'r-');
    legend('Sinal Entrada','Posicao Angular','x1 estimado');
291 title('PD - Com Filtro - Com perturbacao no Binario');
    subplot(1,2,2)
293 plot(teta_ponto_1.time, teta_ponto_1.signals.values,'b',x2_estimado.time,
        x2_estimado.signals.values,'r-');
    legend('Velocidade Angular','x2 estimado');
295 title('PD - Com Filtro - Com perturbacao no Binario');
    %% Alinea 7.2
297 perturbacao_bin_fv = 0;
    perturbacao_bin_st = 0;
299 perturbacao_fv = 1;
    perturbacao_st = 6.5;
301 % PD
    filtro = 100;
303 Ki = 0;
```

```
% Teste 1
rb = 0.02;
mb = 0.01;
Jg = 5.28e-5; %Kg.m2;
JL = Jg + (mb*(rb^2))/2; %Kg.m^2
Jeq = Kg^2*Jm +JL;
sim('servermotor');
% Motor DC + Engrenagens + Carga
teta_PD_1 = teta_1;
% PID
Ki = (P*wn^2)/(b0); % A parte integradora num PID serve para anular o erro em regime
    final
sim('servermotor');
% Motor DC + Engrenagens + Carga
teta_PID_1 = teta_1;
% Teste 2
% PD
Ki = 0;
rb = 0.01;
mb = 0.07;
Jg = 5.28e-5; %Kg.m2;
JL = Jg + (mb*(rb^2))/2; %Kg.m^2
Jeq = Kg^2*Jm +JL;
sim('servermotor');
% Motor DC + Engrenagens + Carga
teta_PD_2 = teta_1;
% PID
Ki = (P*wn^2)/(b0); % A parte integradora num PID serve para anular o erro em regime
    final
sim('servermotor');
% Motor DC + Engrenagens + Carga
teta_PID_2 = teta_1;
figure
plot(sinal_entrada.time, sinal_entrada.signals.values,'k',teta_PD_1.time, teta_PD_1.
    signals.values,'r',teta_PID_1.time, teta_PID_1.signals.values,'b',teta_PD_2.time
    , teta_PD_2.signals.values,'c--',teta_PID_2.time, teta_PID_2.signals.values,'g--
    ');
legend('Sinal de Entrada','PD --> rb = 0.02 mb = 0.01 ','PID --> rb = 0.02 mb = 0.01
    ','PD --> rb = 0.01 mb = 0.07 ','PID --> rb = 0.01 mb = 0.07');
title('Variacao da Carga - Efeito na Posicao Angular')
%-------------------------------------------------------------------------
%% Alinea 8
Rm=2.6;%ohms
Km=0.0078;%Nm/A
Lm=0.00018;%H
Kgi=14; %relacao entre engrenagens e caixa redutora
Kge=5; %relacao entre engrenagens do trem externo
Jmrotor=3.9e-7; %kgm^2
Jm=4.60625e-7; %Kg.m^2
JL=6.63e-5; %Kg.m^2
Jg=5.28e-5; %Kg.m2;
Kg=Kge*Kgi;
Jeq= Kg^2*Jm +JL;
Jl=JL;
Kf=0;
Ke=Km;
Bl=0;
perturbacao_bin_st = 0;
perturbacao_bin_fv = 0;
perturbacao_fv = 1;
perturbacao_st = 20;
```

```
A = [0 1 ; 0 -0.5460/0.0111];
B = [0 1/0.0111]';
C = [1 0];
D = 0;
h=0.01;
[phi,gama]=c2d(A,B,h);
%Controlador: definicao do polinomio caracteristico
zeta=0.7;
wn=21.9955;
den=[1 2*zeta*wn wn*wn];
[a,b,c,d]=tf2ss([0 0 1], den);
[phi_cl,gama_cl]=c2d(a,b,h);
p_cl=eig(phi_cl);
L=acker(phi, gama, p_cl);
Lw=1;
La=[L Lw];
%Estado inicial
x0=[1;1];
%Ganho Lc para seguimento com ganho DC unitario
phic=phi-gama*L;
Lc=1/(C*inv(eye(2)-phic)*gama);
%Observador preditor de estado aumentado
phiw=1; phixw=gama;
phi_a_2=[phi phixw; 0 0 phiw];
gama_a_2=[gama;0];
C_a_2=[C 0];
%Matriz de observabilidade
w0=[C_a_2; C_a_2*phi_a_2; C_a_2*phi_a_2^2];
w0i=inv(w0);
%Vector de ganhos do observador deadbeat
Ko_2=phi_a_2^3*w0i*[0; 0; 1];
%valor inicial do estado observado
xobs_2=[0;0;0];
sim('servermotor') % chamada o modelo Simulink
figure
plot(sinal_entrada.time,sinal_entrada.signals.values,'c',x1_est.time, x1_est.signals
    .values,'b:',x2_est.time, x2_est.signals.values,'r',p_est.time, p_est.signals.
    values,'g:');
legend('Sinal de Entrada','x1 - Posicao Angular','x2 - Velocidade Angular','
    Perturbacao Estimada')
title('Realimentacao dos Estados Estimados com perturbacao na entrada');
axis([ 0 30 -1 1.5]);
perturbacao_bin_st = 20;
perturbacao_bin_fv = 1;
perturbacao_fv = 0;
perturbacao_st = 0;
sim('servermotor') % chamada o modelo Simulink
figure
plot(sinal_entrada.time,sinal_entrada.signals.values,'c',x1_est.time, x1_est.signals
    .values,'b:',x2_est.time, x2_est.signals.values,'r',p_est.time, p_est.signals.
    values,'g:');
legend('Sinal de Entrada','x1 - Posicao Angular','x2 - Velocidade Angular','
    Perturbacao Estimada')
title('Realimentacao dos Estados Estimados com perturbacao no binario');
```

main.m

```
function estim=obx(ent)
global xobs Ko C_a phi_a gama_a;
uk1=ent(1);
yok=ent(2);
```

```
%predicao do estado aumentado
xobs=phi_a*xobs+gama_a*uk1+Ko*(yok-C_a*xobs);
%Devolucao do estado observado
estim=xobs;
```

observador_preditor_aumentado.m

```
function estim=obx(ent)
global xobs_2 Ko_2 C_a_2 phi_a_2 gama_a_2;
uk1=ent(1);
yok=ent(2);
%predicao do estado aumentado
xobs_2=phi_a_2*xobs_2+gama_a_2*uk1+Ko_2*(yok-C_a_2*xobs_2);
%Devolucao do estado observado
estim=xobs_2;
```

observador_preditor_aumentado_2.m

10.2 Simulink

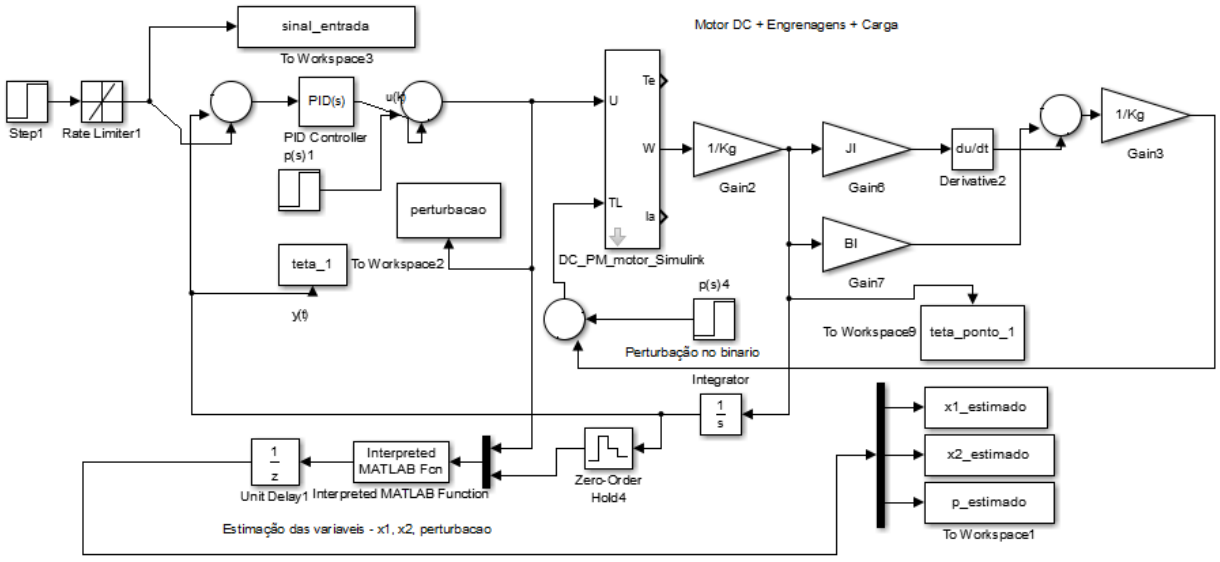

Figura 24: Modelo do Servomotor - motor DC + engrenagens + carga.

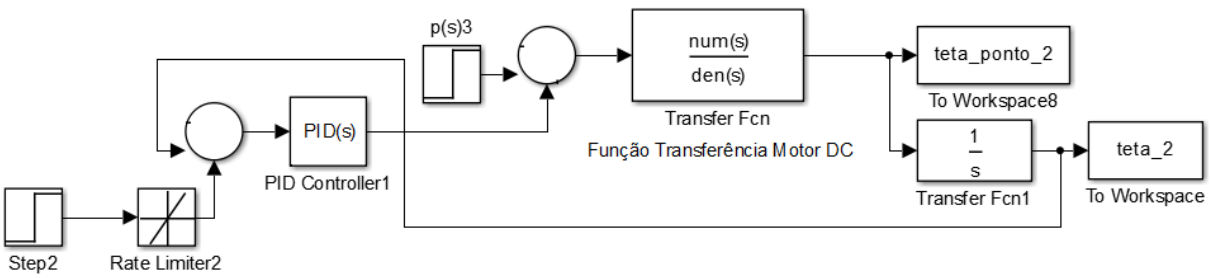

Figura 25: Modelo do Servomotor - Função de Transferência.

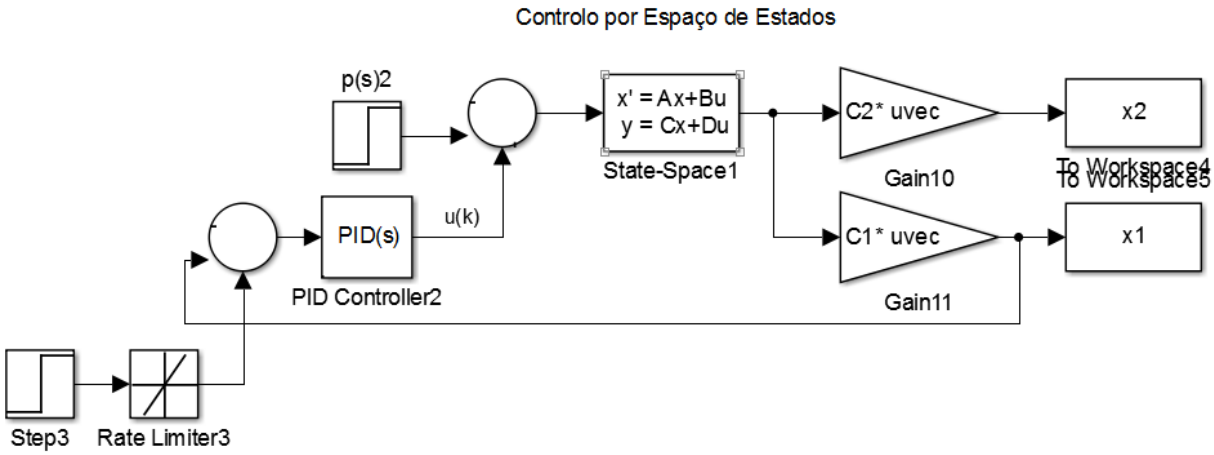

Figura 26: Modelo do Servomotor - Espaço de Estados.

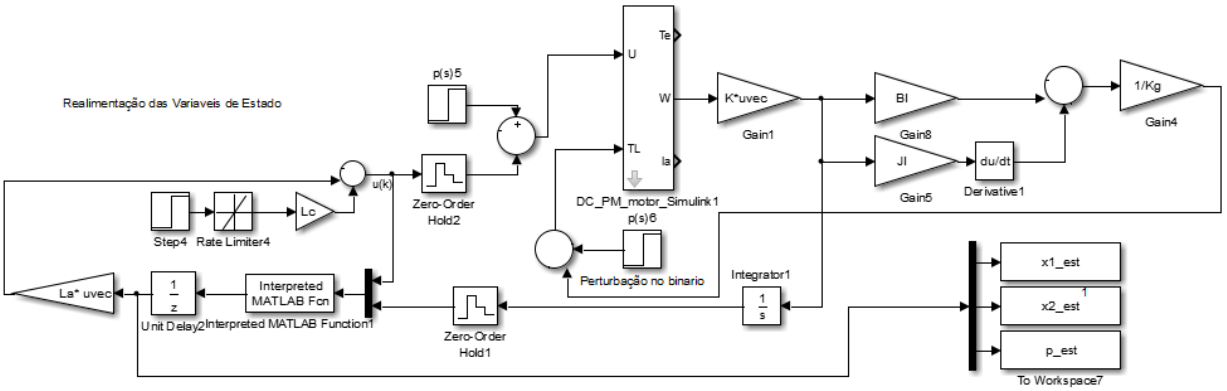

Figura 27: Realimentação de variáveis de estado estimadas.

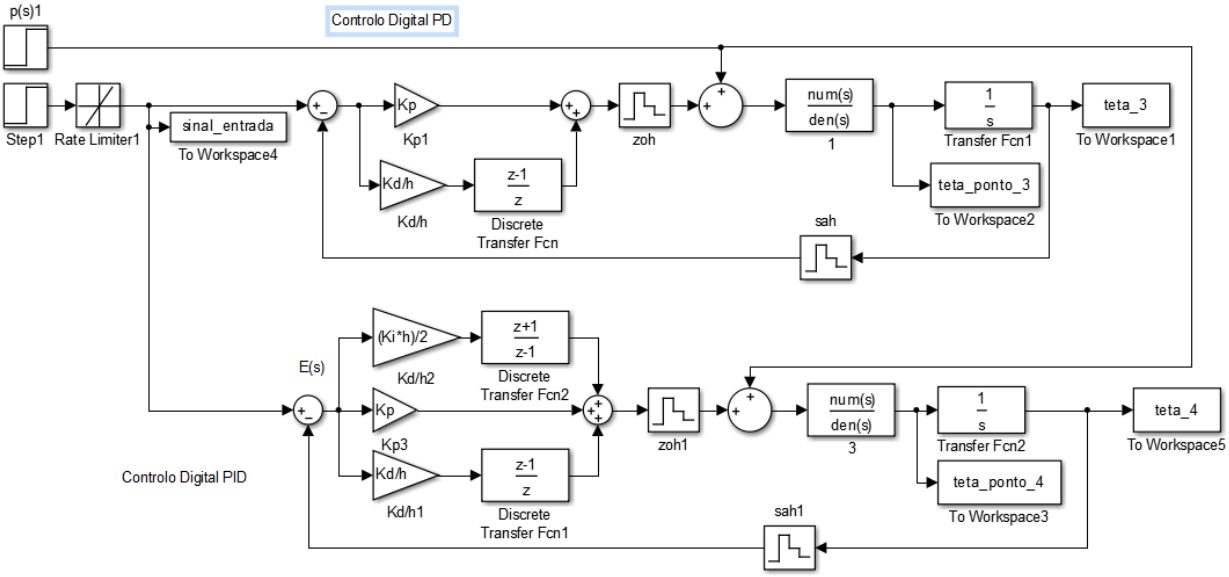

Figura 28: PD e PID contínuo vs aproximação discreta.