INSTITUTO MAUÁ DE TECNOLOGIA

PROJETO SEMESTRAL - SISTEMAS DE CONTROLE I

SIMULAÇÃO DE CONTROLE DE POSIÇÃO E VELOCIDADE DE UM MOTOR DC COM PARÂMETROS INTERATIVOS E PRÉ-CALCULADOS

Leonardo Oneda Galvani (20.00196-7)

Guilherme Nami Bortolozi (20.00333-0)

Henrique Fortuna Accorinti (20.00080-4)

Matheus Ferreira Palú (20.00332-3)

São Caetano do Sul

2023

Sumário

1. Introdução

Este projeto consiste no desenvolvimento de um simulador interativo que deverá controlar a posição angular e velocidade de uma roda de inércia acoplada a um motor DC. O projeto foi desenvolvido utilizando os conhecimentos das matérias estudadas na 4ª série do curso de Engenharia de Controle e Automação do Instituto Mauá de Tecnologia, sendo estas: Sistemas de Controle, Programação Orientada a Objetos e Banco de Dados, Instrumentação, Microcontroladores. Este simulador irá permitir a experimentação e compreensão de conceitos importantes relacionados a essas disciplinas.

Para isso, o projeto foi dividido em duas fases. A primeira fase consiste no desenvolvimento do ambiente virtual, subdividida em duas etapas: a primeira etapa abrange o controle interativo dos ganhos de um controlador PID, que controlará a posição em graus (°) e a velocidade em rotações por minuto (RPM); a segunda etapa abrange análise o comportamento do sistema com controladores pré-projetados para controle apenas da posição em graus (°). Em ambas as etapas haverá a visualização dos resultados das simulações. A segunda fase do projeto consiste na construção do *hardware* necessário para realizar o que for requisitado pelo simulador, envolvendo a especificação dos componentes e a estrutura entre eles e, por fim, a programação para o sistema funcionar de acordo com o especificado pelo usuário do simulador.

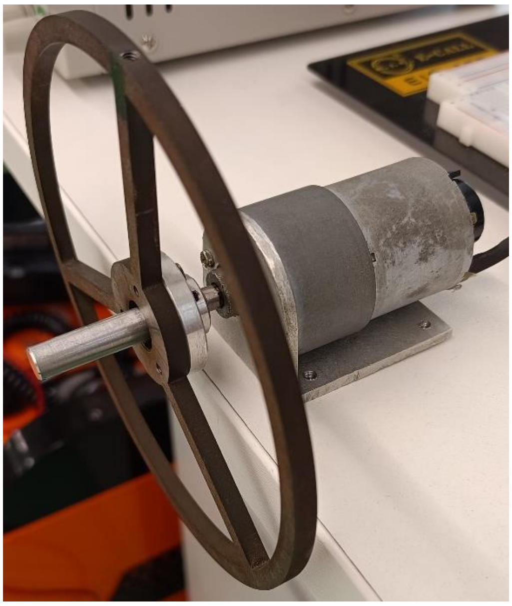
2. Funcionamento

Antes de entrar nos estudos da teoria de controle que envolve o projeto, será abordado o funcionamento das etapas que envolvem o ambiente virtual, os componentes presentes e a lógica entre eles.

2.1. Hardware

Conforme discutido anteriormente, o componente central do projeto é o motor DC, no qual está acoplada uma roda de inércia ao eixo de saída de uma caixa de redução. A razão para sua inclusão no sistema é a criação de uma carga adicional, proporcionando assim um maior desafio no controle do motor. No mesmo conjunto motor, encontramos o rotor do motor acoplado ao eixo de entrada da caixa de redução, bem como um encoder de efeito Hall, conforme ilustrado na imagem a seguir:

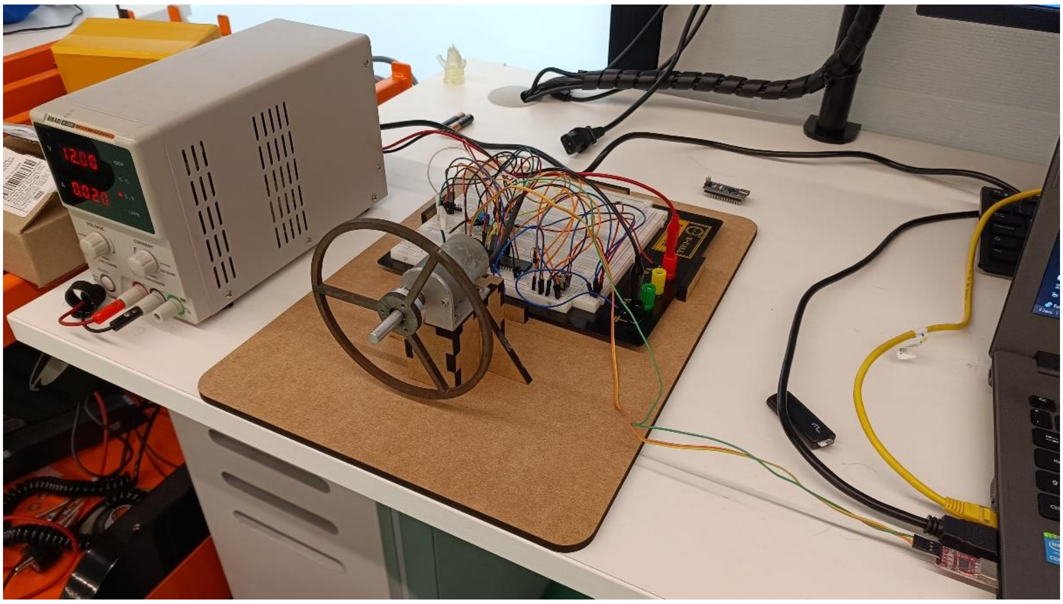
Figura 1 - Planta de controle



Fonte: Autoria própria

Porém além dele existe toda a eletrônica que torna possível o uso e controle do sistema, esse conjunto é chamada de planta, sendo a imagem a seguir:

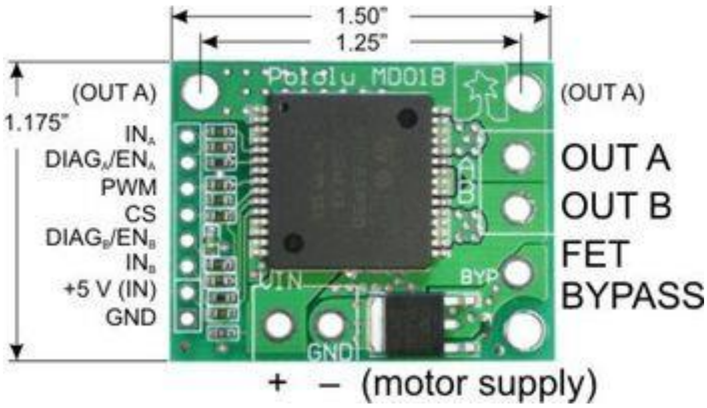
Figura 2 - Planta de controle completa



Fonte: Autoria própria

Quando se trata de controle, a primeira consideração é dada às grandezas analógicas. Por exemplo, foi necessário ajustar a tensão aplicada a um motor devido às suas diferentes velocidades. Para isso, é utilizada uma ponte H, a qual gerencia a tensão e a direção do motor por meio de dois pinos digitais e um sistema PWM (Modulação por Largura de Pulso). A configuração específica da ponte H empregada no projeto está ilustrada abaixo:

Figura 3 - Ponte H

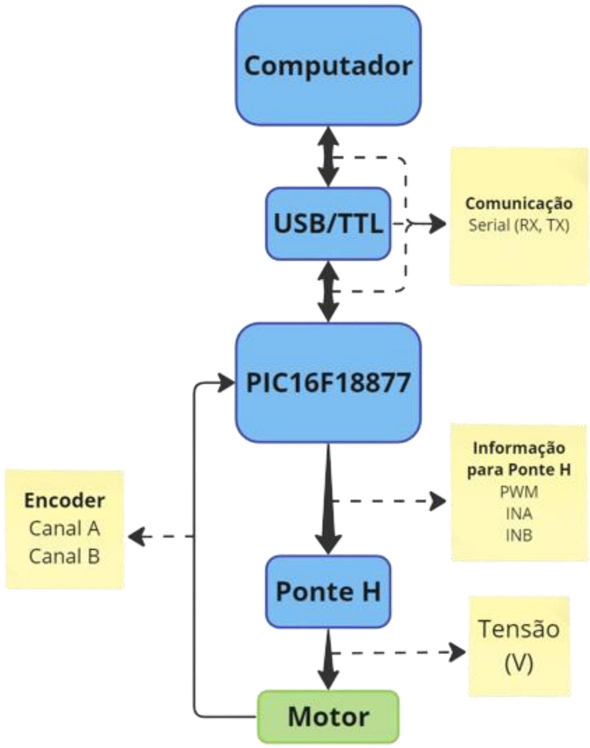


Fonte: Pololu, 2023

Visto que para controlar a tensão e a direção do motor é necessário pinos digitais e um PWM, foi inserido no projeto um microcontrolador, o PIC16F18877. Ele irá controlar a ponte H enviando os dados de acordo com o controlador implementado. Além de estar conectado na ponte H, o microcontrolador ainda será responsável por contar os pulsos gerados no encoder e identificar, por meio do canal A e B, o sentido de rotação do motor e ao final atribuir ao seu valor corretamente.

Neste momento, o *hardware* já teria capacidade de executar o controle, porém a ideia do projeto é fazer um simulador. Para isso, o *hardware* existente deve ser conectado ao computador. Assim, foi utilizado a comunicação serial entre o microcontrolador e o computador, de acordo com o diagrama de comunicação entre *hardwares* apresentados a seguir:

Figura 4 - Fluxograma

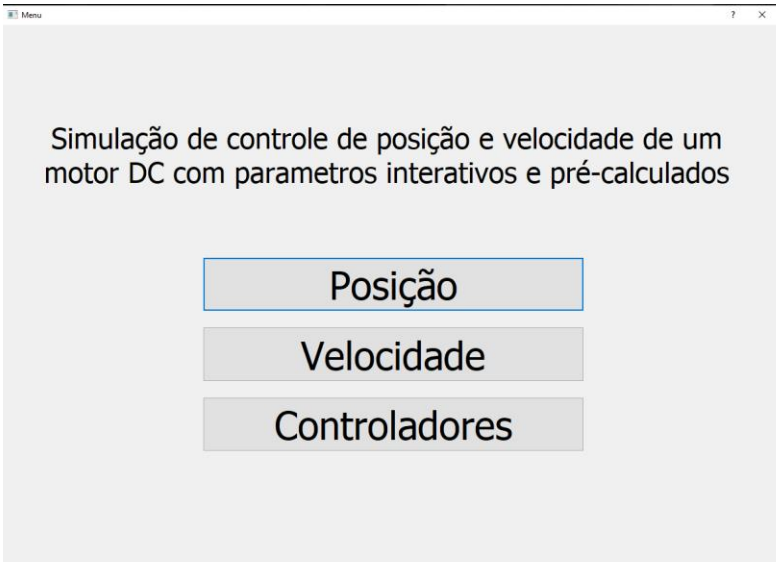


Fonte: Autoria própria

2.2. Interface

Como dito anteriormente, o simulador conta com duas funcionalidades principais: o controle de posição e controle de velocidade. Para acessar os recursos disponíveis, foi desenvolvido uma interface gráfica usando a linguagem de programação Python, utilizando principalmente o PyQt5, que é uma biblioteca com diversas ferramentas para criar janelas de uma interface, podendo colocar gráficos, botões e comando interativos. A sua documentação é excelente abordando todos os tópicos e com a ajuda do livro “Qt5 Python GUI Programming Cookbook” foi possível desenvolver todos os módulos, sendo o primeiro a tela “Menu” mostrada a seguir:

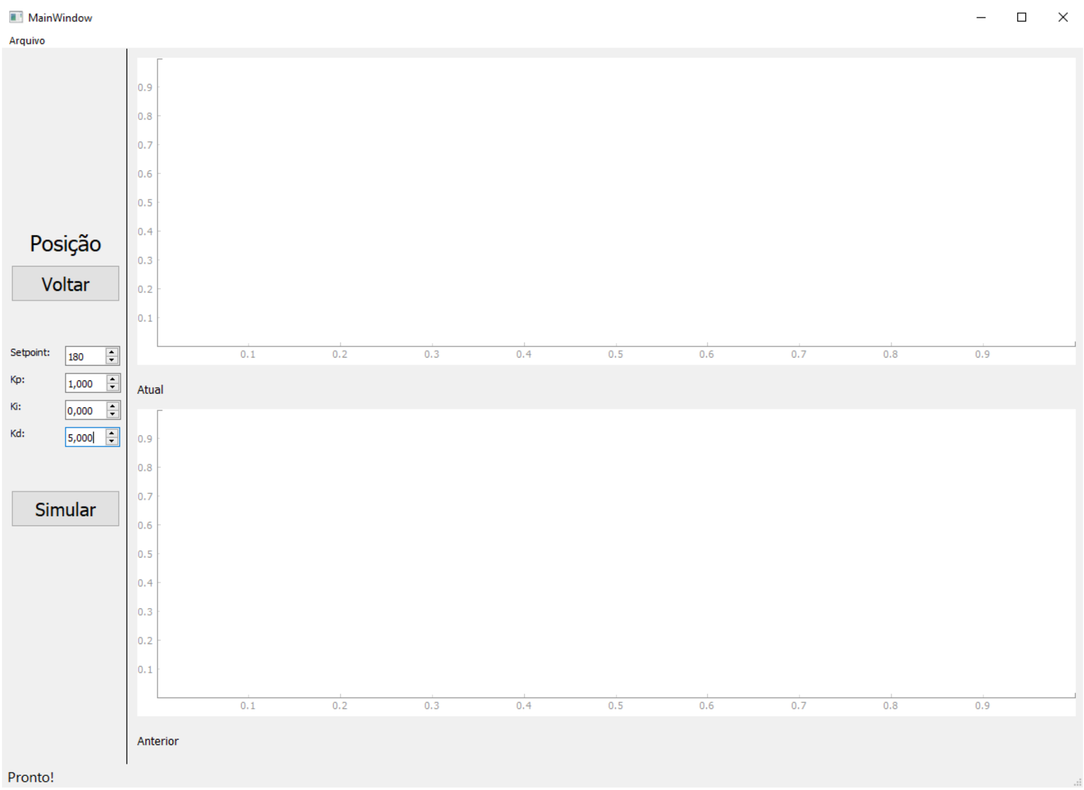
Figura 5 - Menu da interface



Fonte: Autoria própria

Nela, há as opções “Posição” e “Velocidade”, que abrem janelas que permitem configurar e visualizar a simulação, como mostrado na figura a seguir:

Figura 6 - Ambiente de simulação interativo



Fonte: Autoria própria

Além dessas duas opções, ainda há a opção de simular alguns controladores calculados, por meio da aba “Controladores”. Esta funcionalidade será abordada com mais detalhes no decorrer do relatório.

Figura 7 - Ambiente de simulação dos controladores

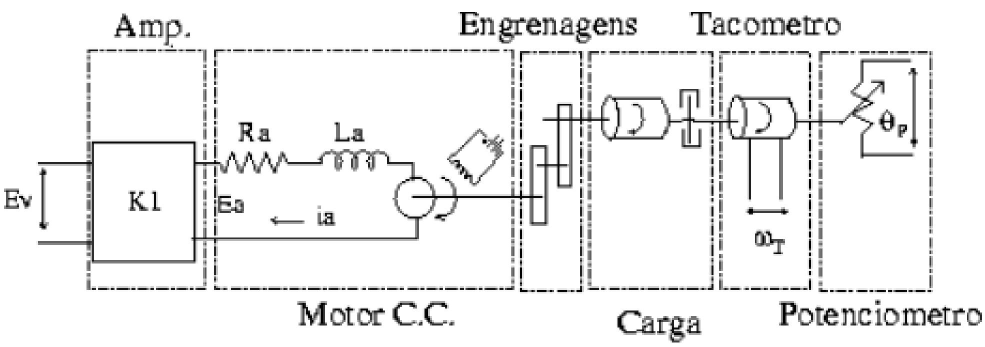
Fonte: Autoria própria

3. Identificação do Sistema

A primeira etapa para projetar um controlador é saber o que ele deve controlar. Em termos matemáticos, devemos conhecer a função de transferência que será trabalhada. No caso do projeto, deve-se encontrar a função de transferência do sistema motor e da roda de inércia.

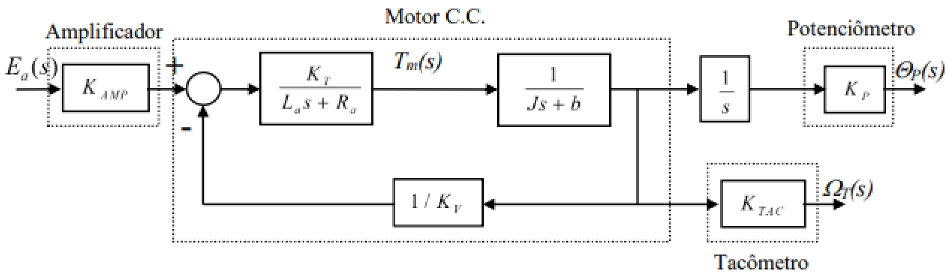
Por se tratar de um motor DC, sua modelagem é bem desenvolvida, sendo possível encontrar diagramas completos, como o mostrado na figura a seguir:

Figura 8 - Diagrama de uma planta com motor CC



Fonte: Material didático

Figura 9 - Diagrama de blocos da planta com o motor CC



Fonte: Material didático

Ao estudar o modelo e suas simplificações, é possível aproximar o modelo do motor para controle de velocidade em um sistema de primeira ordem, e para controle de posição em um sistema de segunda ordem, descritos a seguir:

$$G_\omega(s) = \frac{\Omega_T(s)}{E_V(s)} = \frac{K_W}{Ts + 1}$$

$$G_\theta(s) = \frac{\theta_P(s)}{E_V(s)} = \frac{K_\theta}{s(Ts + 1)}$$

Há diversas maneiras de adquirir essa função de transferência. Uma delas é por meio da modelagem do motor, utilizando suas características físicas e elétricas para encontrar a função desejada. No entanto, o motor em questão não dispõe das informações necessárias para essa modelagem, tornando esse método inviável para a aplicação em questão. Dessa forma, optou-se pela obtenção da função de transferência através do método de ensaio, ou seja, levantamento da curva do motor.

Para esse ensaio, foi decidido que serão realizados 5 degraus com os seguintes valores:

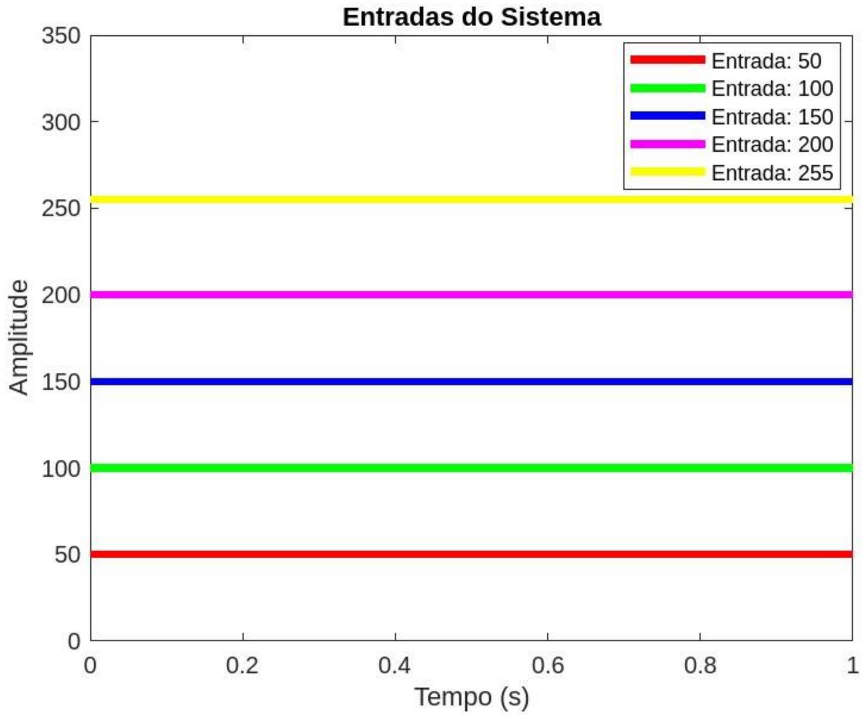
Tabela 1 - Dados de entrada dos ensaios

PWM (8-bit)	Porcentagem (%)
50	19,6%
100	39,2%
150	58,8%
200	78,4%
255	100%

Fonte: Autoria própria

Com esses dados, é possível definir a entrada do ensaio (U):

Figura 10 - Gráfico das entradas dos ensaios

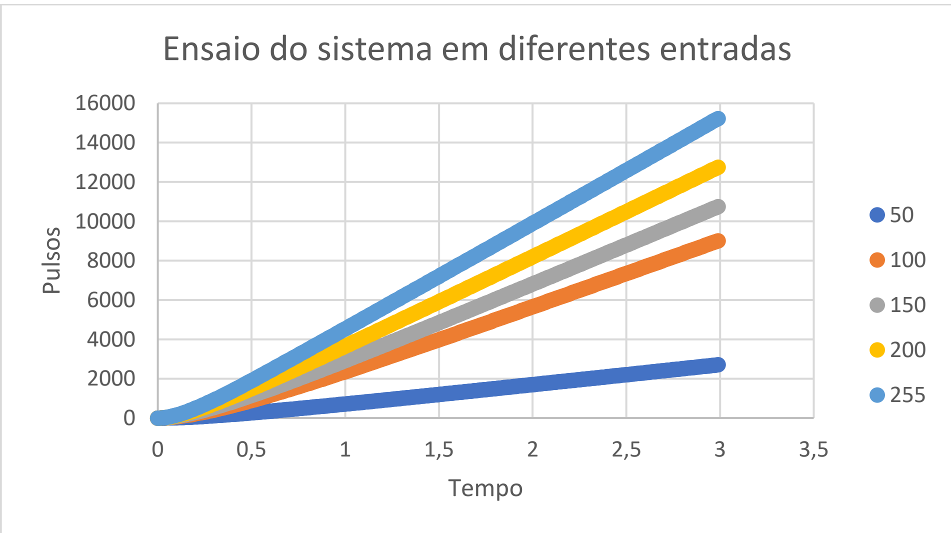


Fonte: Autoria própria

Os valores foram selecionados com base na resolução do PWM do microcontrolador. Uma vez que as entradas estão definidas, o experimento pode ser conduzido. A proposta consistiu em registrar o número de pulsos do encoder ao longo de 3 segundos, período suficiente para estabilizar o sistema. A análise dos dados será abordada de quatro maneiras: duas análises analíticas, utilizando os dados de posição e velocidade, e duas técnicas computacionais, empregando o software MatLab com os mesmos conjuntos de dados.

Ao fim dos ensaios, os seguintes dados foram registrados:

Figura 11 - Gráfico dos ensaios de posição

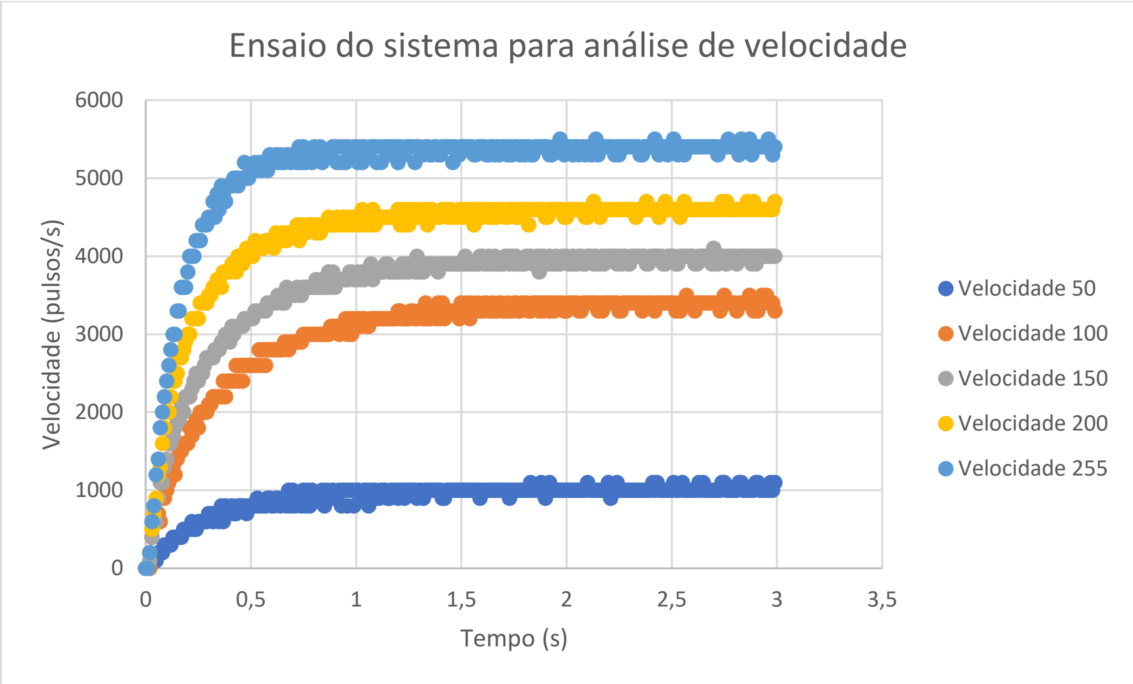


Fonte: Autoria própria

Com os dados registrados, foram definidas as unidades das grandezas a fim de reduzir o processamento necessário no microcontrolador. Dessa maneira, a unidade de velocidade será de pulsos por segundo (pulsos/s) e posição em pulsos (pulsos).

Começando pela análise da velocidade, os dados foram tratados, chegando ao seguinte gráfico:

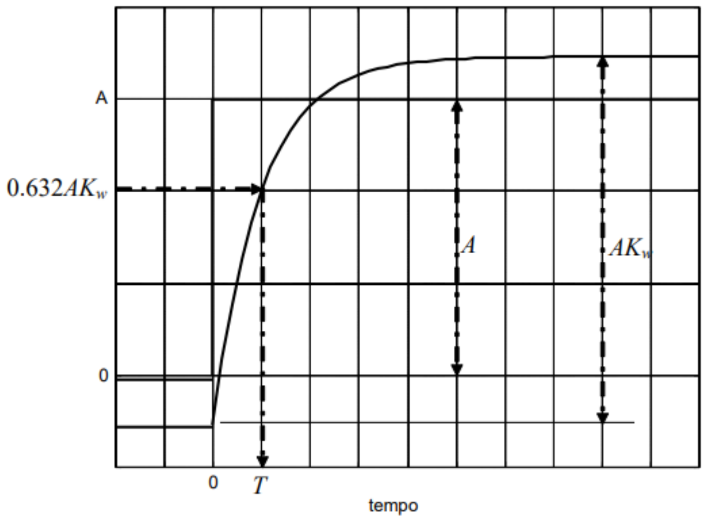
Figura 12 - Gráfico dos ensaios de velocidade



Fonte: Autoria própria

Com esses dados, foi feita a análise separada de cada saída, seguindo o modelo:

Figura 13 - Modelo de cálculo para ensaios de velocidade



Fonte: Material didático

Usando os dados e os conceitos aprendidos na matéria de Sistemas de Controle I, obtemos as seguintes funções de transferência e seu modelo médio:

Tabela 2 - Funções de transferência da velocidade pelo método analítico

Função de Transferência	Constante de tempo (s)	Ganho
$G_{50} = \dfrac{20}{0{,}3s + 1}$	0,3	20
$G_{100} = \dfrac{34}{0{,}3s + 1}$	0,3	34
$G_{150} = \dfrac{26{,}67}{0{,}27s + 1}$	0,27	26,67
$G_{200} = \dfrac{23}{0{,}19s + 1}$	0,19	23
$G_{255} = \dfrac{21{,}176}{0{,}18s + 1}$	0,18	21,176
$G_{M} = \dfrac{24{,}97}{0{,}248s + 1}$	0,248	24,97

Fonte: Autoria própria

Usando outro método de obtenção, pelo *software* MatLab e sua parte de identificação de sistemas, chegamos nas seguintes funções:

Tabela 3 - Funções de transferência da velocidade pelo software

Função de Transferência	Constante de tempo (s)	Ganho
$G_{50} = \dfrac{20{,}128}{0{,}321s + 1}$	0,321	20,128
$G_{100} = \dfrac{33{,}697}{0{,}353s + 1}$	0,353	33,697
$G_{150} = \dfrac{26{,}26}{0{,}295s + 1}$	0,295	26,26
$G_{200} = \dfrac{22{,}776}{0{,}21s + 1}$	0,21	22,776
$G_{255} = \dfrac{21{,}04}{0{,}16s + 1}$	0,16	21,04
$G_{M} = \dfrac{24{,}7802}{0{,}2678s + 1}$	0,2678	24,7802

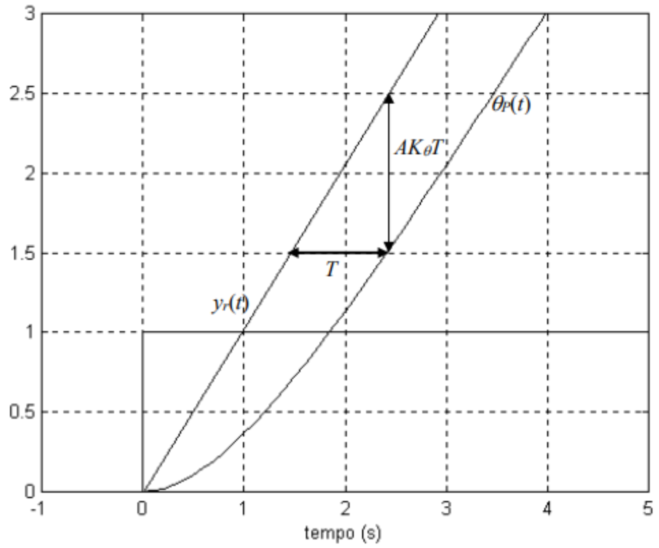
Fonte: Autoria própria

Ao se analisar os resultados obtidos, percebe-se que as funções de transferência médias dos dois métodos exibem parâmetros próximos, com um erro percentual de 0,76% para o ganho e 7,4% para a constante de tempo. Dessa forma, torna-se possível considerar um modelo médio por meio da análise da velocidade.

$$G(s) = \frac{24{,}8751}{0{,}2579s + 1}$$

Além da análise dos dados de velocidade para a estimativa da função de transferência, propõe-se, neste projeto, uma abordagem adicional para obtenção da curva característica. Os dados de posição do motor serão utilizados, permitindo a comparação posterior dos resultados obtidos por ambas as metodologias. O processo seguirá uma abordagem semelhante ao realizado para a velocidade, seguido pela análise no MatLab, conforme o modelo apresentado abaixo:

Figura 14 - Modelo de cálculo para ensaios de posição



Fonte: Material didático

Tabela 4 - Funções de transferência da posição pelo método analítico e software

Analítica			*Software*
Função de transferência	Cte. De tempo	Ganho	Função de transferência
$G_{50} = \frac{19{,}93}{s(0{,}3s+1)}$	0,3	19,93	$G_{50} = \frac{63{,}67}{s^2+3{,}342s+4{,}64*10^{-9}}$
$G_{100} = \frac{33{,}68}{s(0{,}31s+1)}$	0,31	33,68	$G_{100} = \frac{88{,}53}{s^2+2{,}748s+1{,}368*10^{-9}}$
$G_{150} = \frac{26{,}17}{s(0{,}27s+1)}$	0,27	26,17	$G_{150} = \frac{74{,}97}{s^2+2{,}978s+2{,}315*10^{-7}}$
$G_{200} = \frac{23{,}4}{s(0{,}2s+1)}$	0,2	23,4	$G_{200} = \frac{97{,}9}{s^2+4{,}43s+2{,}003*10^{-9}}$
$G_{255} = \frac{20{,}78}{s(0{,}16s+1)}$	0,16	20,78	$G_{255} = \frac{135{,}7}{s^2+6{,}521s+4{,}108*10^{-10}}$
$G_M = \frac{24{,}794}{s(0{,}248s+1)}$	0,248	24,794	$G_M = \frac{92{,}154}{s^2+4{,}002s+4{,}8*10^{-8}}$

Fonte: Autoria própria

A primeira conclusão importante observada a partir da comparação dos métodos é a função de transferência sem o integrador livre obtida usando as ferramentas do MatLab, diferente do modelo simplificado obtido analiticamente. O modelo sem o integrador livre será utilizado para o cálculo analítico do controlador PID, que será abordado futuramente.

A segunda conclusão possível é: considerando que a função de transferência da velocidade integrada resulta na função de transferência da posição para a mesma entrada, chegamos com funções de transferência semelhantes, com um erro da constante de tempo em 3,8% e no ganho em 0,33%.

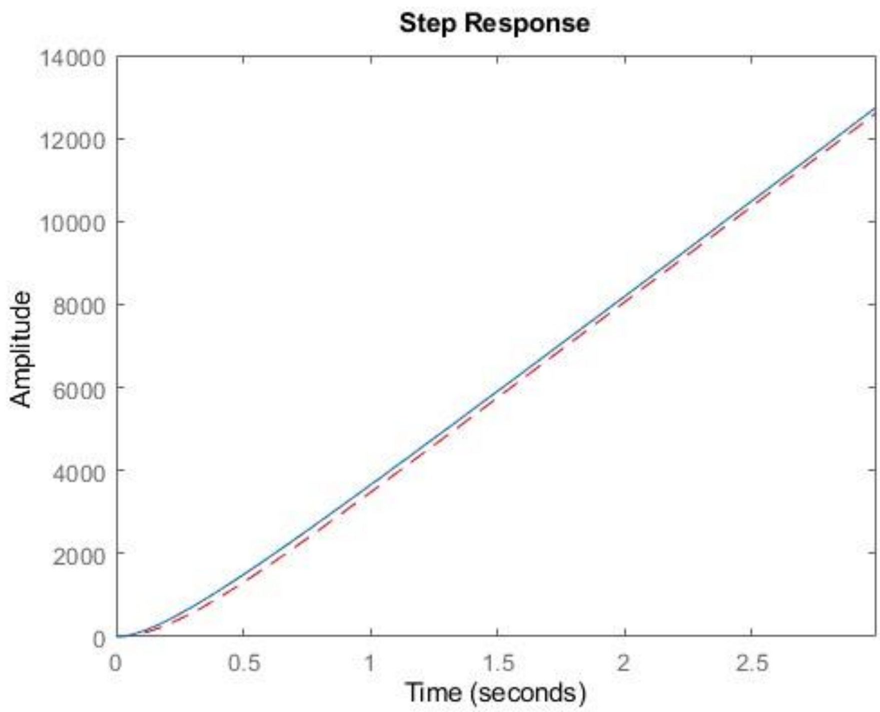
Com todos os dados obtidos e calculados é possível obter um modelo médio do motor estudado:

$$G(s) = \frac{24{,}8345}{s(0{,}2579s + 1)}$$

4. Validação do Sistema

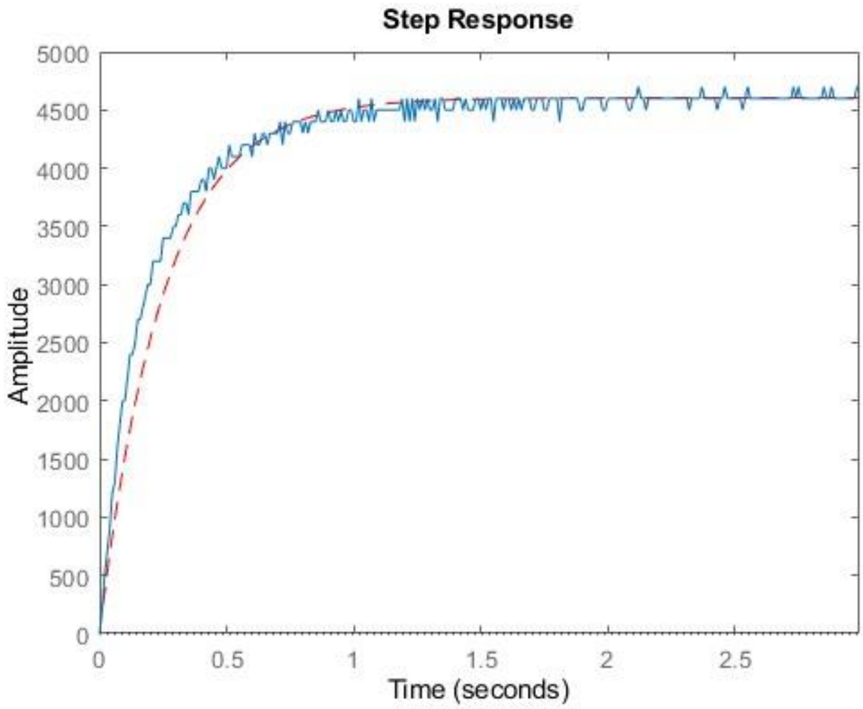
Com a obtenção da função de transferência que descreve o sistema de forma matemática, é possível realizar uma comparação entre a resposta simulada e a resposta real do sistema. A seguir, apresentam-se alguns testes que incluem as curvas correspondentes às simulações e à execução prática.

Figura 15 - Comparação dos modelos de posição e real



Fonte: Autoria própria

Figura 16 - Comparação dos modelos de velocidade e real



Fonte: Autoria própria

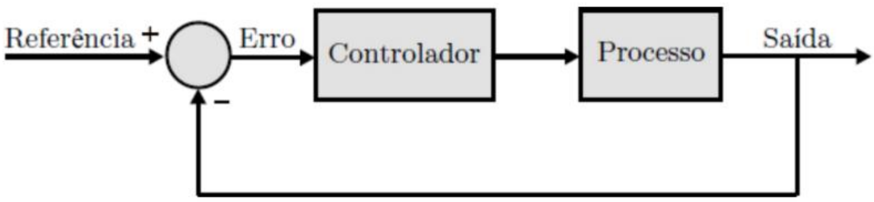
A curva azul representa o que o motor realmente realizou, e em vermelho tracejado está a simulação pela função de transferência obtida nos ensaios.

5. Proposta de controle do sistema

O projeto incorpora uma seção de controle interativo fundamentada em um controlador PID, além da seção dedicada aos controladores projetados. Em relação a esta última, o propósito primordial consiste na comparação entre os controladores calculados e aqueles projetados com o auxílio do MatLab. Estes incluem um controlador de avanço de fase, um controlador PID e um controlador PD, sendo este último concebido no MatLab.

Primeiramente, todos os controladores devem seguir o mesmo diagrama de blocos:

Figura 17 - Diagrama de blocos



Fonte: Material didático

O primeiro controlador projetado será o avanço de fase para o controle da posição do motor. Para isso, foram determinados os requisitos do projeto: um tempo de assentamento ($T_S(2\%)$) de 1 segundo e um sobressinal ($M_{Pt}(\%)$) de 1%. Sabendo disso, é possível começar o projeto do controlador. Sabendo que:

$$G_C(s) = \frac{k_c(s+z_c)}{s+p_c}, \qquad z_c < p_c$$

O primeiro passo para projetar o controle é definir os polos desejados:

$$\left.\begin{array}{r} \zeta = \sqrt{\dfrac{\ln^2 M_p}{\pi^2 + \ln^2 M_p}} = \sqrt{\dfrac{\ln^2 0{,}01}{\pi^2 + \ln^2 0{,}01}} = 0{,}826 \\ T_S(2\%) = \dfrac{4}{\sigma} = \dfrac{4}{\zeta\omega_n} \rightarrow 1 = \dfrac{4}{0{,}826 * \omega_n} \therefore \omega_n = 4{,}842\ rad/s \end{array}\right\} p_d = -\zeta\omega_n \pm \omega_n\sqrt{1-\zeta^2}j = -4 \pm 2{,}7293j$$

Depois de calcular os polos desejados, é possível começar o cálculo do controlador. O primeiro passo é assumir o zero do controlador igual a parte real do polo desejado, ou seja, $z_c = 4$. Portanto, temos:

$$G_C(s) = \frac{k_c(s+4)}{s+p_c}$$

O próximo passo é utilizar o critério de fase para encontrar o valor de p_c:

$$\angle\left(\left(\frac{K_c\ *(s+4)}{s+8{,}46}\right) * \left(\frac{24{,}8345}{s*(0{,}2579s+1)}\right)\right)_{s=-4+2{,}7293j} = -180 \rightarrow \therefore p_c = 8{,}46$$

Com o polo do controlador calculado, encontra-se o ganho do controlador, k_c. Para isso, será usado o critério do módulo:

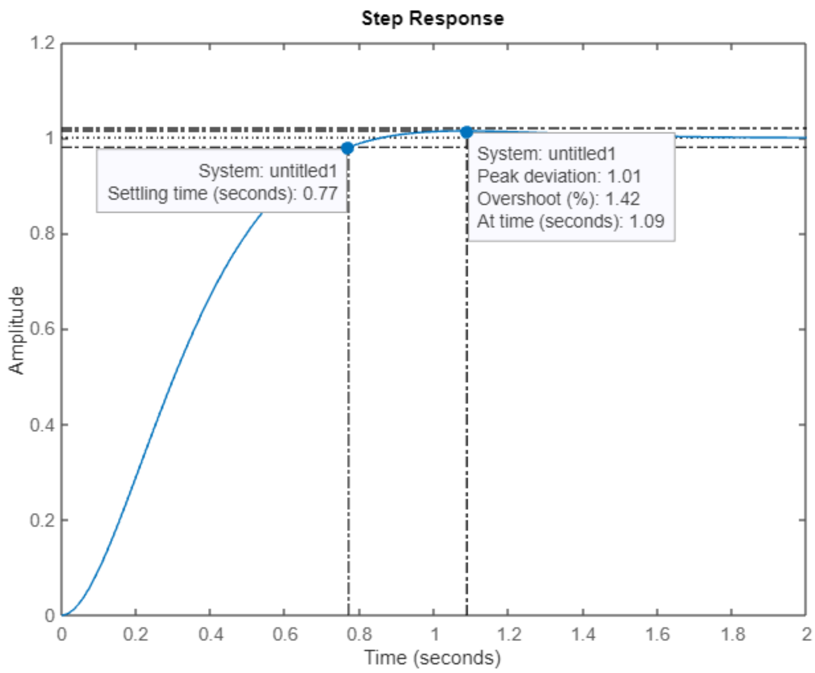
$$\left|\left(\frac{K_c\ *(s+4)}{s+8{,}46}\right) * \left(\frac{24{,}8345}{s*(0{,}2579s+1)}\right)\right|_{s=-4+2{,}7293j} = 1 \rightarrow \therefore k_c = 0{,}2632$$

Depois de obter seus parâmetros, foi obtido o seguinte controlador:

$$G_C(s) = \frac{0{,}2632(s+4)}{s+8{,}46}$$

Para avaliar se o controlador calculado atende aos parâmetros do projeto, é possível simular sua resposta no MATLAB.

Figura 18 - Resposta do sistema controlado



Fonte: Autoria própria

Outro controlador proposto é o PID, mas para este será necessário utilizar outro modelo, aquele que contêm dois polos diferentes de 0, como mencionado anteriormente:

$$G_M = \frac{92{,}154}{s^2 + 4{,}002s + 4{,}8 * 10^{-8}} = \frac{92{,}154}{(s+4{,}002)(s+1{,}2*10^{-8})}$$

O controlador PID pode ser escrito como visto abaixo, e ao final do processo os valores de Kp, Ti e Td devem ser definidos.

$$G_{PID} = k_p * \left(1 + \frac{1}{T_i s} + T_D s\right) = k_P \left(\frac{T_i T_D s^2 + T_i s + 1}{T_i s}\right)$$

O primeiro passo é encontrar os valores de Ti e Td a fim de “cancelar” os polos da planta. Para isso, é necessário normalizar o denominador de G_M:

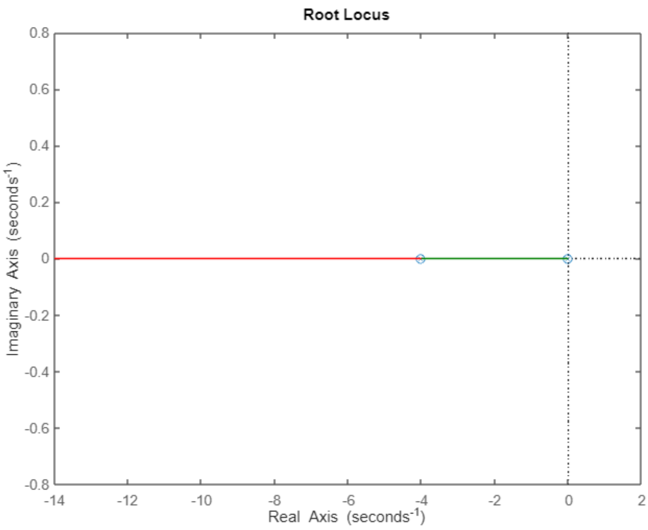
$$s^2 + 4{,}002s + 4{,}8 * 10^{-8} \rightarrow \frac{s^2}{4{,}8 * 10^{-8}} + \frac{4{,}002}{4{,}8 * 10^{-8}} s + 1$$

Com o denominador normalizado é possível estabelecer o seguinte sistema:

$$\begin{cases} T_i T_D = \dfrac{1}{4{,}8 * 10^{-8}} \\ T_i = \dfrac{4{,}002}{4{,}8 * 10^{-8}} \end{cases} \rightarrow \begin{cases} T_D = 0{,}2499 \\ T_i = 83375000 \end{cases}$$

Nese momento, foi obtido os valores de Ti e Td, faltando calcular o valor de Kp. Porém, antes disso, deve-se olhar o diagrama do lugar raízes admitindo $k_p = 1$:

Figura 19 - Lugar geométrico das raízes em malha aberta do sistema controlado



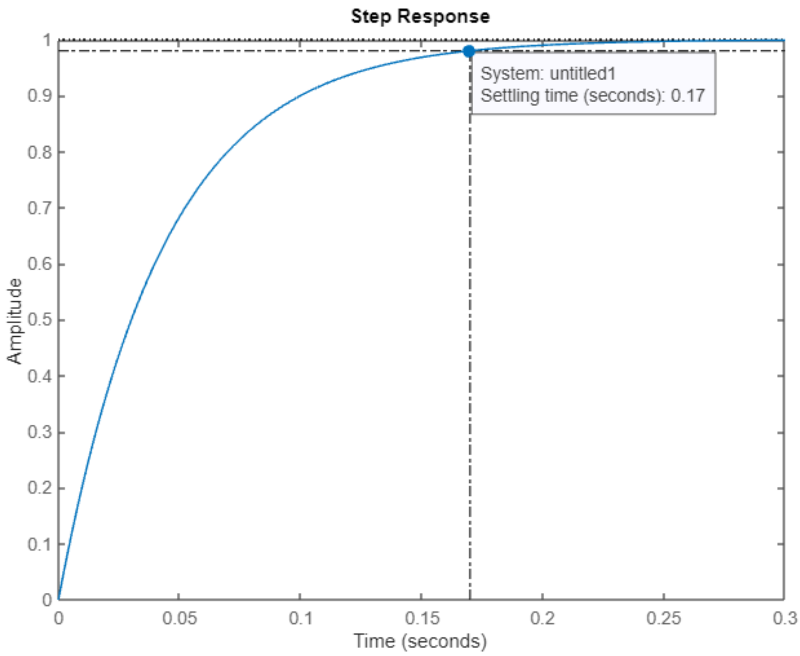
Fonte: Autoria própria

É possível verificar que, independentemente do valor do k_p, a resposta do sistema nunca será instável e nunca oscilará, portanto, a única influência de k_p está no tempo de resposta do sistema. Portanto, $k_p = 1$ já satisfaz os requisitos do projeto.

Admitindo o controlador projetado, temos o seguinte controlador e a resposta simulada no MatLab:

$$G_{PID} = \left(1 + \frac{1}{83375000 * s} + 0{,}2499 * s\right)$$

Figura 20 - Resposta do sistema controlado



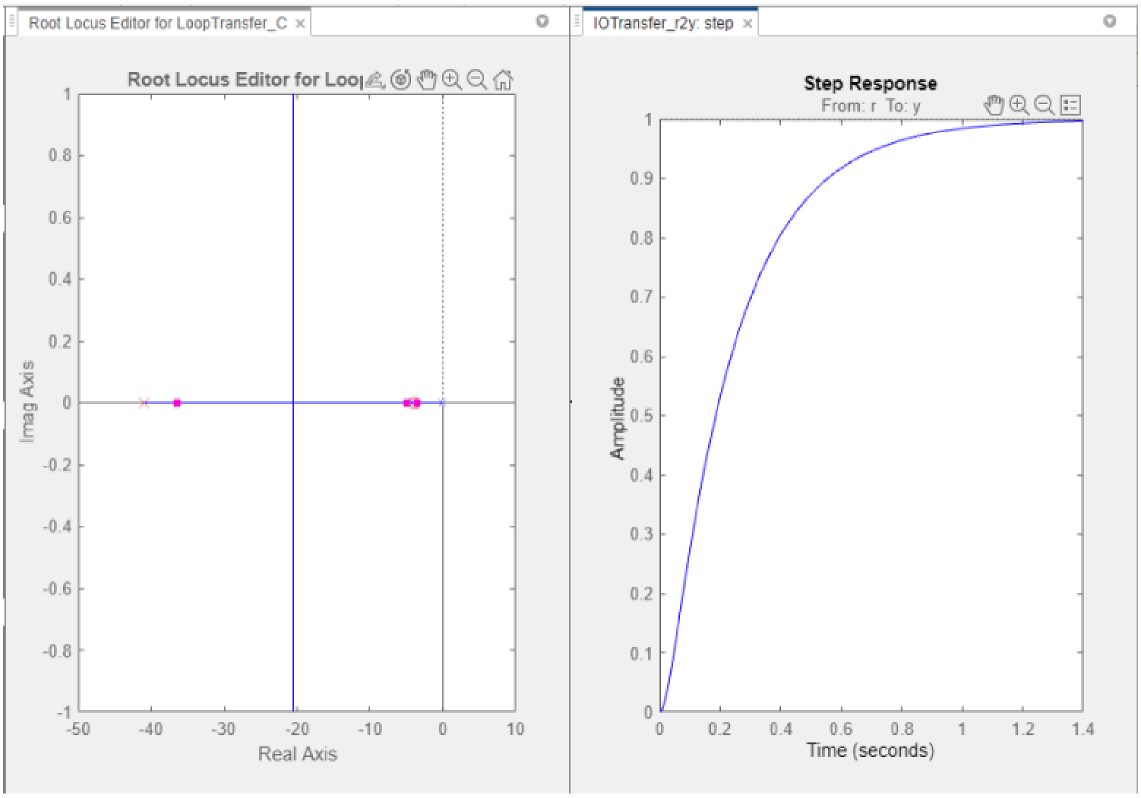
Fonte: Autoria própria

O resultado obtido é satisfatório para projeto do controlador.

Além dos controladores de avanço de fase e PID mencionados, será apresentada uma comparação com um controlador PD desenvolvido por meio das ferramentas do MatLab, especificamente o RLTOOL. A otimização do resultado foi realizada de acordo com os requisitos do projeto, considerando a sensibilidade ao ganho do controlador. Os resultados obtidos incluem a configuração específica do controlador e a saída correspondente da planta controlada por esse dispositivo.

$$G_C(s) = 0{,}17016 * \frac{(1 + 0{,}25s)}{(1 + 0{,}024s)}$$

Figura 21 - Ferramenta do MatLab para projetar o controlador

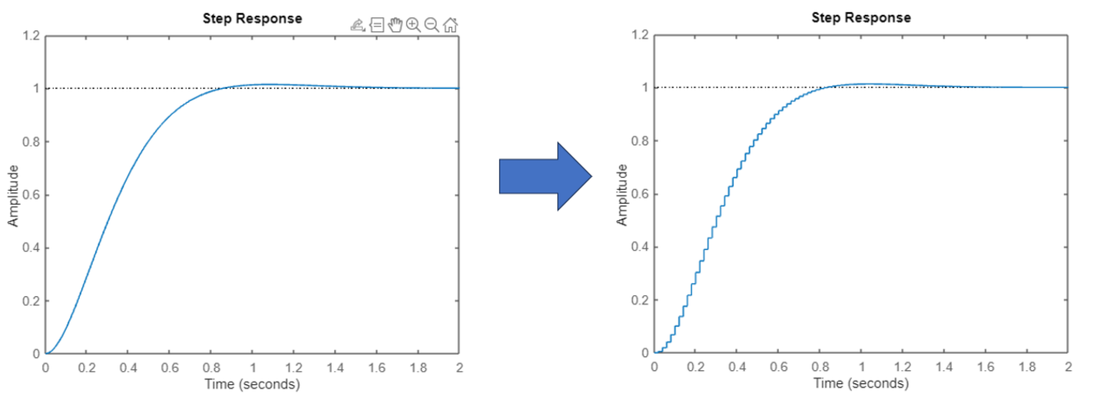


Fonte: Autoria própria

6. Controle Embarcado

Com o objetivo de integrar um controlador em um microcontrolador ou CLP, por exemplo, torna-se necessário discretizar o controlador projetado em tempo contínuo. Essa discretização implica que o controlador será atualizado em intervalos específicos definidos por T, denominado tempo de amostragem. A seguir, apresenta-se um exemplo de um sistema discretizado:

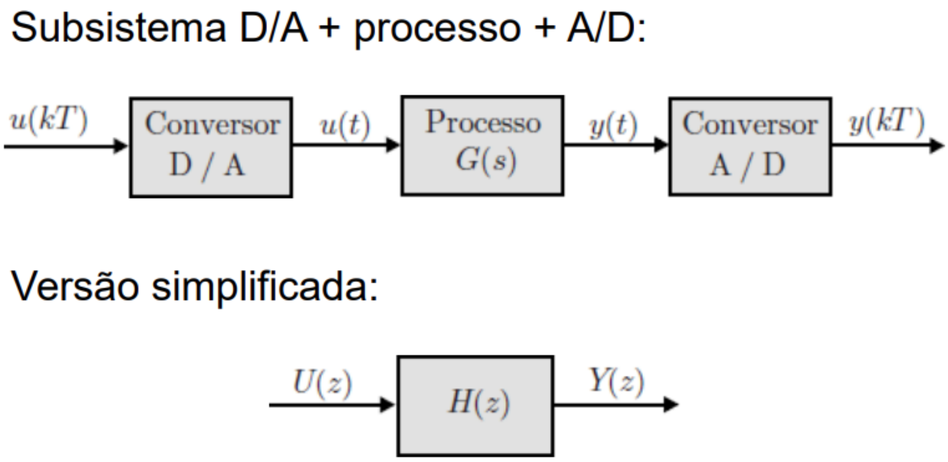
Figura 22 - Representação do sistema discreto



Fonte: Autoria própria

Para isso, utilizaremos a função de transferência do nosso controlador e a discretização será feita usando a transformada Z e a simplificação dos diagramas com os conversores analógicos/digitais e digitais/analógicos, como na imagem e equação abaixo:

Figura 23 - Diagrama de blocos de um sistema discretizado



Fonte: Material didático

$$\text{com } H(z) = (1 - z^{-1})\mathrm{Z}\left[\frac{G(s)}{s}\right]$$

No primeiro momento, o cálculo será demonstrado utilizando o controlador de avanço de fase. Porém, os outros dois controladores serão construídos utilizando a ferramenta do MatLab.

$$H_{af}(z) = (1 - z^{-1})\mathrm{Z}\left[\frac{\frac{0{,}2632(s+4)}{s+8{,}46}}{s}\right] = (1 - z^{-1})\mathrm{Z}\left[\frac{0{,}2632(s+4)}{s(s+8{,}46)}\right]$$

Para realizar a transformada Z foi utilizado o método da expansão em frações parciais:

$$\mathrm{Z}\left[\frac{0{,}2632(s+4)}{s(s+8{,}46)}\right] = \mathrm{Z}\left[\frac{A}{s} + \frac{B}{(s+8{,}46)}\right]$$

$$\begin{cases} A = \lim_{s\to 0}\left(\frac{0{,}2632(s+4)}{s(s+8{,}46)}\right) * s = \frac{0{,}2632(0+4)}{(0+8{,}46)} = 0{,}1244 \\ B = \lim_{s\to -8{,}46}\left(\frac{0{,}2632(s+4)}{s(s+8{,}46)}\right) * (s+8{,}46) = \frac{0{,}2632(-8{,}46+4)}{-8{,}46} = 0{,}13876 \end{cases}$$

$$Ƶ\left[\frac{0,2632(s+4)}{s(s+8,46)}\right] = Ƶ\left[\frac{0,1244}{s} + \frac{0,13876}{(s+8,46)}\right] = 0,1244 * \frac{z}{z-1} + 0,13876 * \frac{z}{z-e^{-8,46*T}}$$

$$H_{af}(z) = (1-z^{-1})\left(0,1244 * \frac{z}{z-1} + 0,13876 * \frac{z}{z-e^{-8,46*T}}\right)$$

$$H_{af}(z) = \left(\frac{z-1}{z}\right)\left(\frac{0,1244 * z * (z-e^{-8,46*T}) + 0,13876 * z * (z-1)}{(z-1)(z-e^{-8,46*T})}\right)$$

$$H_{af}(z) = \frac{0,1244 * (z-e^{-8,46*T}) + 0,13876 * (z-1)}{(z-e^{-8,46*T})}$$

Considerando um tempo de amostragem para os controladores de 10ms, $T = 0.01$, chegamos a seguinte função discreta:

$$H_{af}(z) = \frac{0,1244 * (z-e^{-8,46*0,01}) + 0,13876 * (z-1)}{(z-e^{-8,46*0,01})} = \frac{0,2632z - 0,2531}{z - 0,9189}$$

Para otimizar o processo foi utilizada a função c2d, que resulta na função já discretizada a partir da função de transferência em tempo contínuo e o tempo amostral. O resultado está descrito nas seguintes funções:

$$H_{PD}(z) = \frac{1,772z - 1,715}{z - 0,6592}$$

$$H_{PID}(z) = \frac{1,509z^2 - 2,94z + 1,431}{z^2 - 1,607 * z + 0.6065}$$

O próximo passo é encontrar a equação de diferenças utilizando a manipulações algébricas e a propriedade do atraso, $Ƶ\{\mathrm{u[k-1]T}\} = z^{-1}U(z)$. O cálculo será novamente desenvolvido para o controlador de avanço de fase, e os outros dois controladores terão exatamente o mesmo cálculo.

$$\frac{Y(z)}{U(z)} = \frac{0,2632z - 0,2531}{z - 0,9189} = \frac{z(0,2632 - 0,2531z^{-1})}{z(1 - 0,9189z^{-1})}$$

$$Y(Z) - 0{,}9189z^{-1}Y(z) = 0{,}2632U(z) - 0{,}2531z^{-1}U(z)$$

$$y_{af}(kT) = 0{,}2632u[kT] - 0{,}2531u[(k-1)T] + 0{,}9189y_{af}[(k-1)T]$$

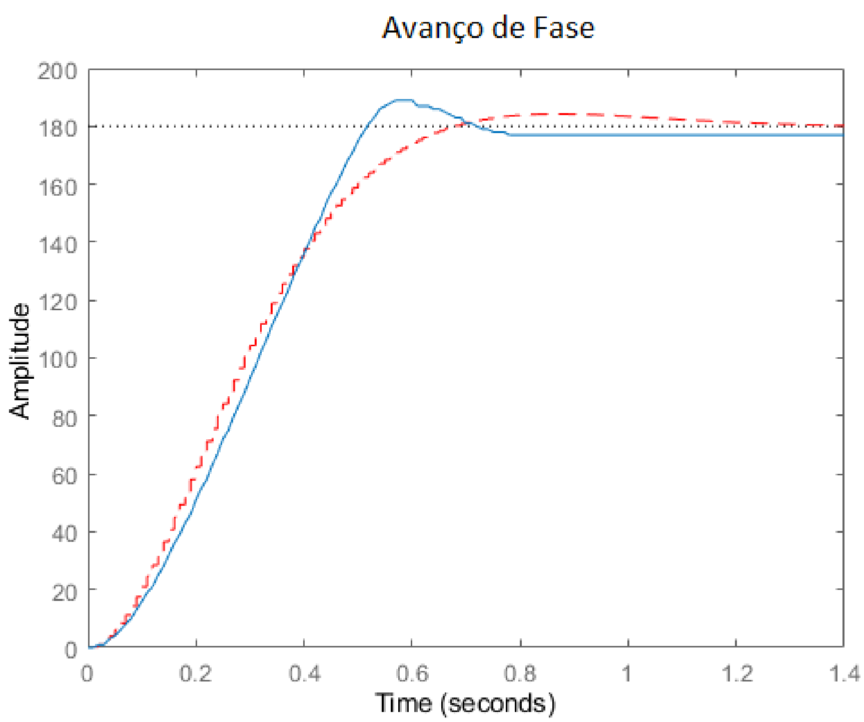
Sendo “u” a entrada, o erro, e “y” a saída do sistema, o PWM, aplicada no motor. Aplicando o mesmo cálculo para os outros controladores achamos as seguintes equações de diferenças:

$$y_{PD}(kT) = 1{,}772u[kT] - 1{,}715u[(k-1)T] + 0{,}6592y_{PD}[(k-1)T]$$

$$y_{PID}(kT) = 1{,}509u[kT] - 2{,}94u[(k-1)T] + 1{,}431u[(k-2)T] + 1{,}607y_{af}[(k-1)T] - 0{,}6065y_{PID}[(k-2)]$$

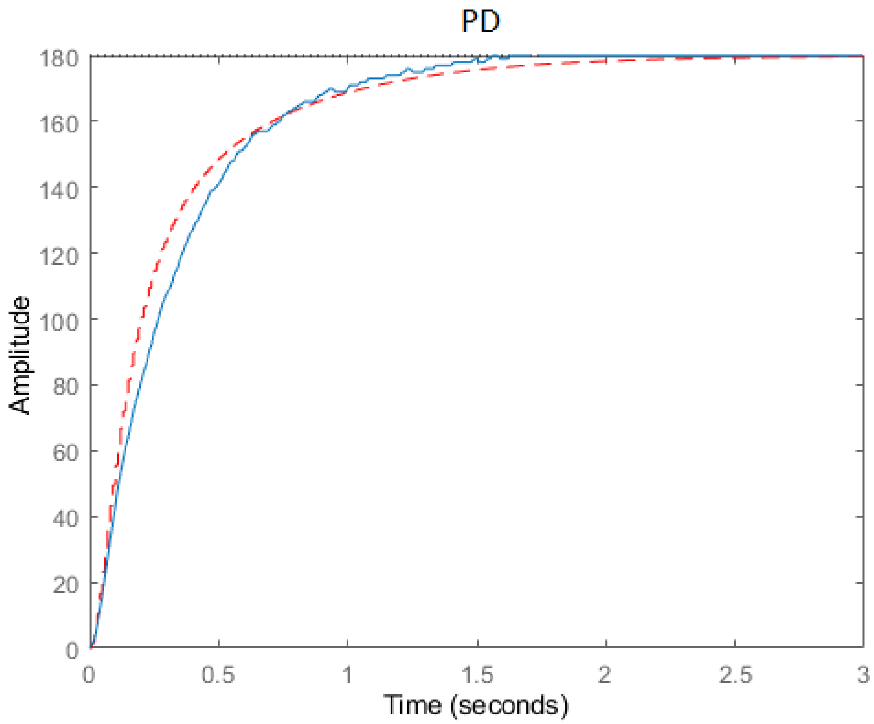
Com essas equações de diferenças é possível implementá-las no microcontrolador e testá-las a fim de compará-las com os resultados simulados, como demonstrado nos seguintes gráficos:

Figura 24 - Comparação do sistema



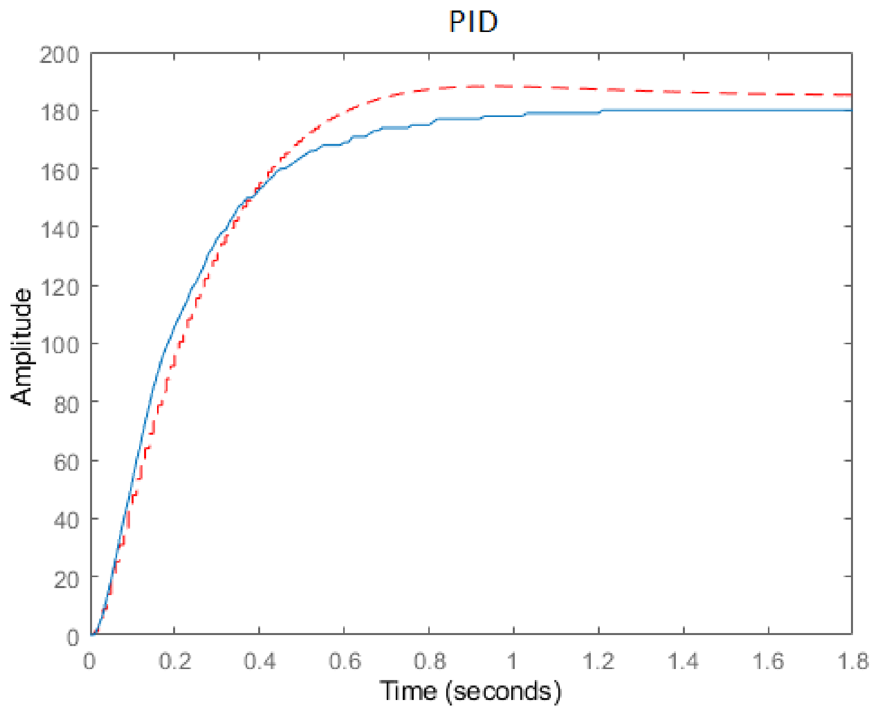
Fonte: Autoria própria

Figura 25 - Comparação do sistema



Fonte: Autoria própria

Figura 26 - Comparação do sistema



Fonte: Autoria própria

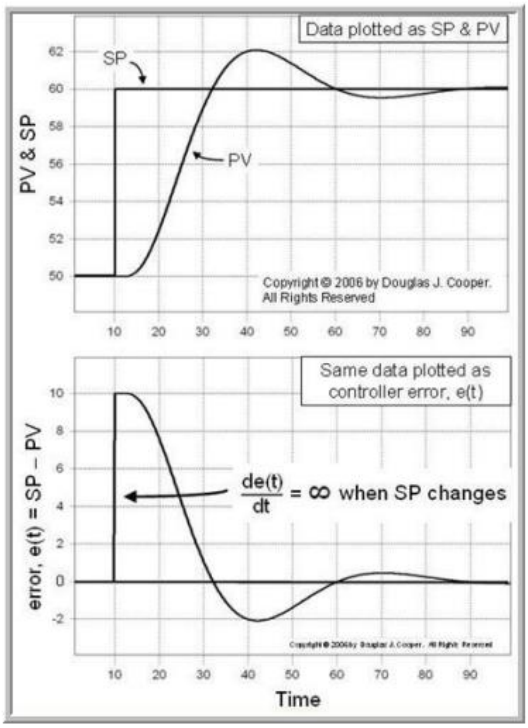
A linha vermelha tracejada representa a curva simulada e a linha azul representa a curva do sistema real.

Outro controle discretizado que é importante ser mencionado está compreendido na primeira parte do projeto, no qual os ganhos de um controlador PID são interativos. Nessa parte, existe um controlador discreto implementado da seguinte forma:

$$u(t) = K_p e(t) + K_i \int_0^t e(t)dt + K_d \frac{d}{dt} e(t)$$

A componente integrativa é obtida pelo somatório do produto do erro pela variação do tempo (Δt), enquanto a componente derivativa resulta da diferença dividida pelo mesmo intervalo de tempo. Normalmente, essa diferença é calculada entre o erro atual e o anterior, mas essa abordagem pode levar a uma derivada que tende ao infinito. Para contornar esse problema, opta-se pela variação da variável do processo, utilizando a posição em vez da diferença entre os erros. Assim, realiza-se a subtração entre a posição atual e a posição anterior, invertendo o comportamento da derivada. Para corrigir essa inversão, a parcela da derivada é subtraída em vez de somada, como evidenciado na comparação a seguir:

Figura 27 - Comparações dos métodos



Fonte: Material didático

Dessa forma, obtemos a seguindo função:

$$u(t) = K_p e(t) + K_i \int_0^t e(t)dt + K_d \frac{d}{dt} PV(t)$$

7. Conclusões

Como conclusão inicial, é possível ver que utilizando o método analítico ou por software, analisando velocidade ou posição gerou modelos próximos, com pequenos erros, como comentado anteriormente, portanto ambos os métodos atendem o objetivo, obtendo o modelo do motor utilizando os ensaios.

Outra conclusão identifica-se disparidades na resposta do sistema, influenciadas por diversos fatores, tais como a folga existente na caixa de redução, que impacta a dinâmica do sistema. Dada a impossibilidade de sua eliminação completa, considera-se a utilização de uma caixa de redução alternativa para aprimorar a concordância com os resultados desejados.

Outro elemento relevante destacado refere-se à zona morta do motor, caracterizada por um intervalo de tensões entre 0V e um determinado valor, no qual o motor permanece inerte devido a considerações mecânicas, predominantemente associadas ao atrito estático e momentos.

Apesar das sutis discrepâncias observadas, é evidente que o projeto atendeu às expectativas, possibilitando a comparação da eficiência entre controladores e métodos, além de proporcionar uma compreensão abrangente dos conceitos abordados nas disciplinas pertinentes.

8. Referências Bibliográficas

LIN, Paul-I-Hai; HWANG, Sentai; CHOU, J. **Comparison on fuzzy logic and PID controls for a DC motor position controller.** In: Proceedings of 1994 IEEE Industry Applications Society Annual Meeting, Denver, CO, USA, 1994. p. 1930-1935 vol.3. DOI: 10.1109/IAS.1994.377695.

BAR-KANA, I.; FISCHL, R.; KALATA, P. **Direct position plus velocity feedback control of large flexible space structures.** In: IEEE Transactions on Automatic Control, vol. 36, no. 10, Oct. 1991. p. 1186-1188. DOI: 10.1109/9.90232.

MAHMUD, M.; MOTAKABBER, S. M. A.; ALAM, A. H. M. Z.; NORDIN, A. N. **Adaptive PID Controller Using for Speed Control of the BLDC Motor.** In: 2020 IEEE International Conference on Semiconductor Electronics (ICSE), Kuala Lumpur, Malaysia, 2020. p. 168-171. DOI: 10.1109/ICSE49846.2020.9166883.

BALAMURUGAN, S.; UMARANI, A. **Study of Discrete PID Controller for DC Motor Speed Control Using MATLAB.** In: 2020 International Conference on Computing and Information Technology (ICCIT-1441), Tabuk, Saudi Arabia, 2020. p. 1-6. DOI: 10.1109/ICCIT-144147971.2020.9213780.

Pololu - VNH2SP30 Motor Driver Carrier MD01B. Disponível em: <https://www.pololu.com/product/706>. Acesso em: 10 nov. 2023.

HARWANI, B. M. **Qt5 Python GUI Programming Cookbook Building responsive and powerful cross-platform applications with PyQt.** [s.l.] Birmingham ; Mumbai Packt July, 2018.

LGALVANI007. **Projeto Semestral**. Disponível em: <https://github.com/lgalvani007/ProjetoSemestral>. Acesso em: 14 nov. 2023.