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

Guilherme Nami Bortolozi

Henrique Fortuna Accorinti

Matheus Ferreira Palú

São Caetano do Sul

2023

Sumário

[1. Introdução 3](#_Toc150877922)

[2. Funcionamento 4](#_Toc150877923)

[*2.1.* *Hardware* 4](#_Toc150877924)

[2.2. Interface 6](#_Toc150877925)

[3. Identificação do Sistema 8](#_Toc150877926)

[4. Validação do Sistema 14](#_Toc150877927)

[5. Proposta de controle do sistema 15](#_Toc150877928)

[6. Controle Embarcado 20](#_Toc150877929)

[7. Conclusões 26](#_Toc150877930)

[8. Referências Bibliográficas 26](#_Toc150877931)

# 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.

# 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.

## *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 - Planta de controle

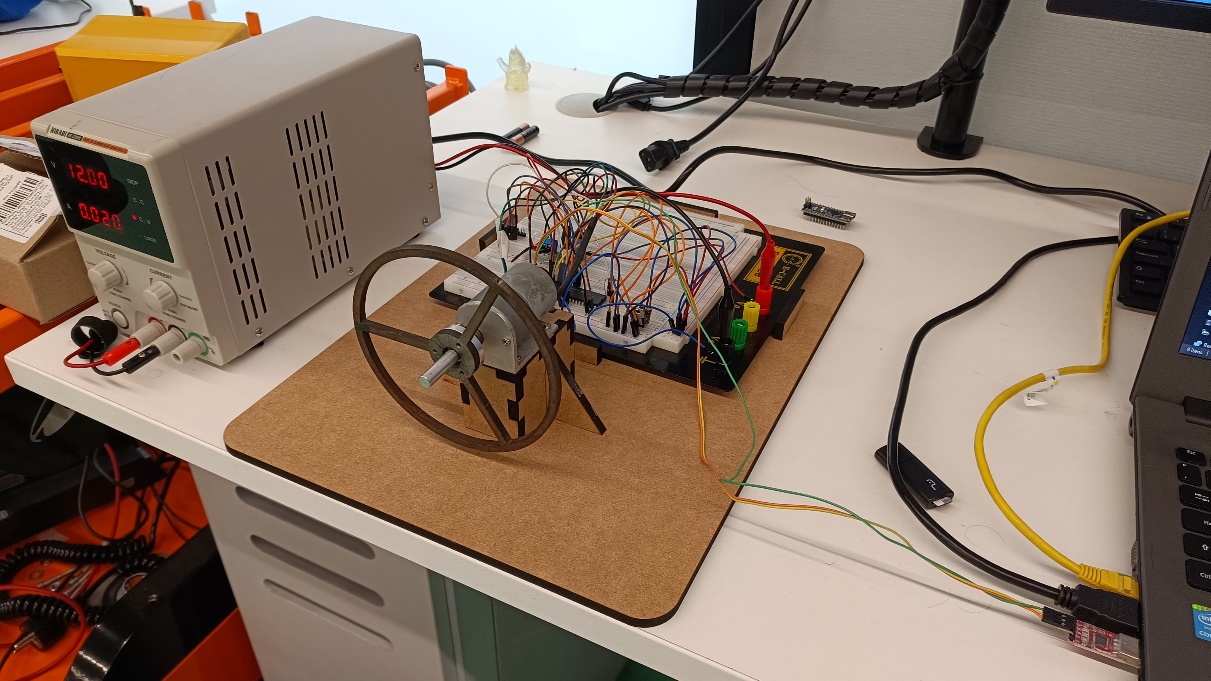
Uma imagem contendo no interior, mesa, computador, escritório

Descrição gerada automaticamente

*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 - 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 - Ponte H

Tela de um aparelho eletrônico

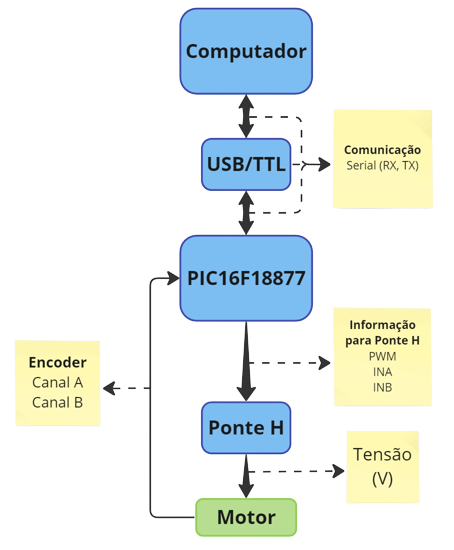
Descrição gerada automaticamente com confiança baixa

*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 - Fluxograma



*Fonte: Autoria própria*

## 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 - Menu da interface

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

*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 - Ambiente de simulação interativo

Tabela

Descrição gerada automaticamente

*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 - Ambiente de simulação dos controladores

Uma imagem contendo Tabela

Descrição gerada automaticamente

*Fonte: Autoria própria*

# 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 - Diagrama de uma planta com motor CC

Diagrama, Esquemático

Descrição gerada automaticamente

*Fonte: Material didático*

Figura - Diagrama de blocos da planta com o motor CC

Diagrama

Descrição gerada automaticamente

*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:

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:

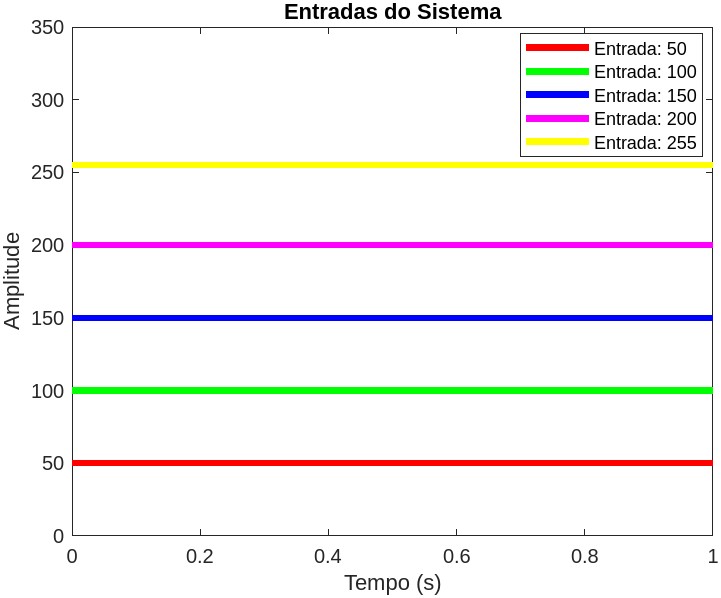
Tabela - Dados de entrada dos ensaios

|  |  |
| --- | --- |
| **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 - 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 - 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 - Gráfico dos ensaios de velocidade

*Fonte: Autoria própria*

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

Figura - Modelo de cálculo para ensaios de velocidade

Gráfico, Gráfico de linhas

Descrição gerada automaticamente

*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 - Funções de transferência da velocidade pelo método analítico

|  |  |  |
| --- | --- | --- |
| Função de Transferência | Constante de tempo (s) | Ganho |
|  | 0,3 | 20 |
|  | 0,3 | 34 |
|  | 0,27 | 26,67 |
|  | 0,19 | 23 |
|  | 0,18 | 21,176 |
|  | 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 - Funções de transferência da velocidade pelo software

|  |  |  |
| --- | --- | --- |
| Função de Transferência | Constante de tempo (s) | Ganho |
|  | 0,321 | 20,128 |
|  | 0,353 | 33,697 |
|  | 0,295 | 26,26 |
|  | 0,21 | 22,776 |
|  | 0,16 | 21,04 |
|  | 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.

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 - Modelo de cálculo para ensaios de posição

Gráfico, Gráfico de linhas

Descrição gerada automaticamente

*Fonte: Material didático*

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

|  |  |  |  |
| --- | --- | --- | --- |
| Analítica | | | *Software* |
| Função de transferência | Cte. De tempo | Ganho | Função de transferência |
|  | 0,3 | 19,93 |  |
|  | 0,31 | 33,68 |  |
|  | 0,27 | 26,17 |  |
|  | 0,2 | 23,4 |  |
|  | 0,16 | 20,78 |  |
|  | 0,248 | 24,794 |  |

*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.

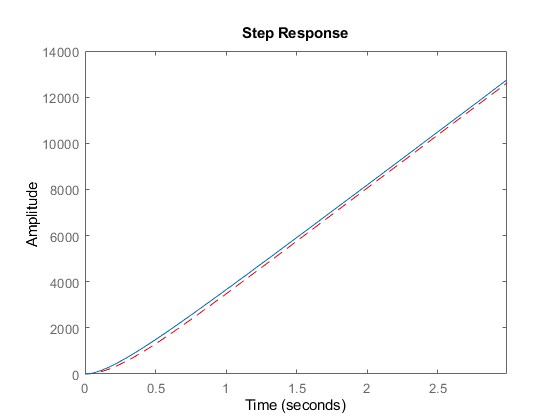
A segunda conclusão possível é: considerando que a função de transferência da velocidade integrada resulta na função de transferência da posição para a mesma entrada, chegamos com funções de transferência semelhantes, com um erro da constante de tempo em 3,8% e no ganho em 0,33%.

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

# Validação do Sistema

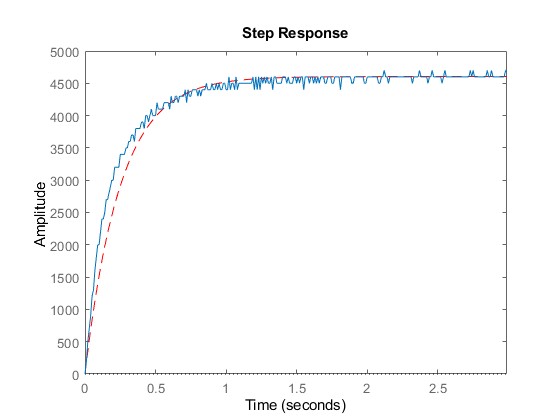
Com a obtenção da função de transferência que descreve o sistema de forma matemática, é possível realizar uma comparação entre a resposta simulada e a resposta real do sistema. A seguir, apresentam-se alguns testes que incluem as curvas correspondentes às simulações e à execução prática.

Figura - Comparação dos modelos de posição e real



*Fonte: Autoria própria*

Figura - 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.

# 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 - Diagrama de blocos

Diagrama

Descrição gerada automaticamente

*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 (2%)) de 1 segundo e um sobressinal () de 1%. Sabendo disso, é possível começar o projeto do controlador. Sabendo que:

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

Forma

Descrição gerada automaticamente com confiança média

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, . Portanto, temos:

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

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

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

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

Figura - Resposta do sistema controlado

Gráfico

Descrição gerada automaticamente

*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:

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

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 :

Com o denominador normalizado é possível estabelecer o seguinte sistema:

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 :

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

Gráfico, Gráfico de linhas, Gráfico de caixa estreita

Descrição gerada automaticamente

*Fonte: Autoria própria*

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

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

Figura - Resposta do sistema controlado

Gráfico

Descrição gerada automaticamente

*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.

Figura - Ferramenta do MatLab para projetar o controlador

Gráfico

Descrição gerada automaticamente

*Fonte: Autoria própria*

# 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 - Representação do sistema discreto

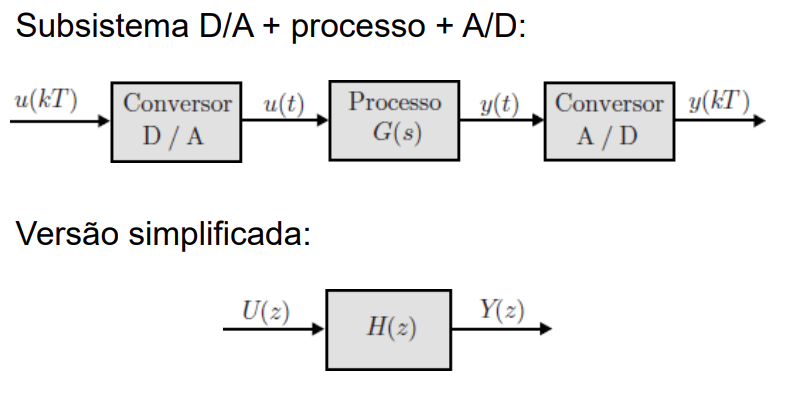
Gráfico

Descrição gerada automaticamente

*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 - Diagrama de blocos de um sistema discretizado



*Fonte: Material didático*

com

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.

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

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

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:

O próximo passo é encontrar a equação de diferenças utilizando a manipulações algébricas e a propriedade do atraso, . O cálculo será novamente desenvolvido para o controlador de avanço de fase, e os outros dois controladores terão exatamente o mesmo cálculo.

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:

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 - Comparação do sistema

Gráfico, Gráfico de linhas

Descrição gerada automaticamente

*Fonte: Autoria própria*

Figura - Comparação do sistema

Gráfico

Descrição gerada automaticamente

*Fonte: Autoria própria*

Figura - Comparação do sistema

Gráfico

Descrição gerada automaticamente

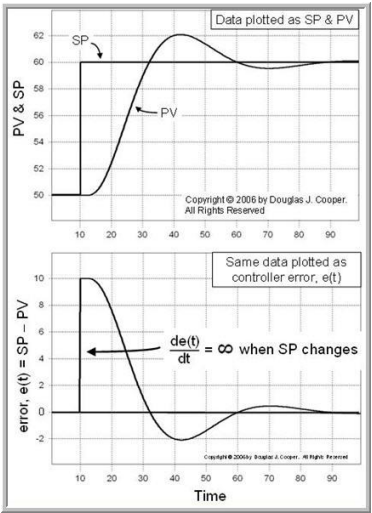
*Fonte: Autoria própria*

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:

A componente integrativa é obtida pelo somatório do produto do erro pela variação do tempo (∆t), 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 - Comparações dos métodos



*Fonte: Material didático*

Dessa forma, obtemos a seguindo função:

# 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.

# Referências Bibliográficas

LIN, Paul-l-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.

‌

‌

‌

‌