Trabalho de Conclusão de Curso

Controle PID aplicado a um sistema de geração elétrico

Lilian Fabrício Marques Neves

orientado por
Prof. Dr. Ícaro Bezerra Queiroz de Araújo

Universidade Federal de Alagoas
Instituto de Computação
Maceió, Alagoas
6 de dezembro de 2024

UNIVERSIDADE FEDERAL DE ALAGOAS
Instituto de Computação

CONTROLE PID APLICADO A UM SISTEMA DE GERAÇÃO ELÉTRICO

Trabalho de Conclusão de Curso apresentado ao Instituto de Computação da Universidade Federal de Alagoas como requisito parcial para a obtenção do grau de Bacharel em Engenharia de Computação.

Lilian Fabrício Marques Neves

Orientador: Prof. Dr. Ícaro Bezerra Queiroz de Araújo

Banca Examinadora:

Ícaro Bezerra Queiroz de Araújo	Prof. Dr., IC-UFAL
Glauber Rodrigues Leite	Prof. Dr., IC-UFAL
Andressa Martins Oliveira	MsC, IC-UFAL

Maceió, Alagoas
6 de dezembro de 2024

Catalogação na Fonte
Universidade Federal de Alagoas
Biblioteca Central
Divisão de Tratamento Técnico
Bibliotecário: Cláudio Albuquerque Reis – CRB-4 – 1753

N518c Neves, Lilian Fabrício Marques.
Controle PID aplicado a um sistema de geração elétrico / Lilian Fabrício Marques Neves. – 2024.
27 f. : il.

Orientador: Ícaro Bezerra Queiroz de Araújo.

Monografia (Trabalho de conclusão de curso em Engenharia de Computação) – Universidade Federal de Alagoas. Instituto de Computação. Maceió, 2024.

Bibliografia. f. 19.
Apêndice. f. 20-27.

1. Engenharia de computação. 2. Controle de processos. 3. Controladores PID. I. Título.

CDU: 004.451.25

UNIVERSIDADE FEDERAL DE ALAGOAS/UFAL
Instituto de Computação – IC
Campus A. C. Simões - Av. Lourival de Melo Mota, BL 12
Tabuleiro do Martins, Maceió/AL - CEP: 57.072-970
Telefone: (082) 3214-1401

Trabalho de Conclusão de Curso - TCC

Formulário de Avaliação

Nome do Aluno
Lilian Fabrício Marques Neves

Nº de Matrícula
19211367

Título do TCC (Tema)
Controle PID aplicado a um sistema de geração elétrico

Banca Examinadora

Prof. Dr. Ícaro Bezerra Queiroz de Araújo
Orientador

gov.br Documento assinado digitalmente
ICARO BEZERRA QUEIROZ DE ARAUJO
Data: 06/12/2024 17:12:49-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. Glauber Rodrigues Leite
Membro da banca

gov.br Documento assinado digitalmente
GLAUBER RODRIGUES LEITE
Data: 07/12/2024 09:19:52-0300
Verifique em https://validar.iti.gov.br

Profa. Me. Andressa Martins Oliveira
Membro da banca

gov.br Documento assinado digitalmente
ANDRESSA MARTINS OLIVEIRA
Data: 08/12/2024 15:59:49-0300
Verifique em https://validar.iti.gov.br

Data da Defesa	Nota Obtida
06/12/2024	8,5 (Oito e meio)

Observações

Coordenador do Curso
De Acordo

gov.br Documento assinado digitalmente
JOBSON DE ARAUJO NASCIMENTO
Data: 06/02/2025 11:36:32-0300
Verifique em https://validar.iti.gov.br

Assinatura

Resumo

Este trabalho apresenta a implementação e análise comparativa dos controlador PID aplicados ao controle de velocidade de um sistema motor-gerador de corrente contínua (CC) em uma bancada experimental de baixo custo. Utilizando componentes acessíveis, como Arduino Nano, transistor TIP120, sensores de velocidade e uma fonte de alimentação de 5V, a bancada permite explorar o ajuste manual dos parâmetros de controle, proporcionando uma visão prática das diferenças entre as estratégias de controle. Para cada controlador, foi avaliado o desempenho em termos de tempo de resposta, estabilidade e precisão. O sistema foi programado na Arduino IDE e, através do monitor serial, foram coletados dados para plotagem dos gráficos de medição, atuação e referência, permitindo uma análise detalhada da resposta do sistema em diferentes condições operacionais.

***Palavras-chave*: Controle de processos, Controlador PID, Sistema Motor-Gerador, Sistema de Controle.**

Abstract

This work presents the implementation and comparative analysis of PID controller applied to the speed control of a direct current (DC) motor-generator system in a low-cost experimental setup. Using accessible components such as an Arduino Nano, TIP120 transistor, speed sensors, and a 5V power supply, the setup enables the manual tuning of control parameters, providing a practical understanding of the differences between control strategies. The performance of each controller was evaluated in terms of response time, stability, and accuracy. The system was programmed using the Arduino IDE, and data were collected via the serial monitor for plotting measurement, control, and reference graphs, enabling a detailed analysis of the system's response under various operational conditions.

Keywords*: *Process Control*, *PID Controller*, *Motor-Generator System*, *Control Systems.

Lista de Figuras

Lista de Abreviações

P *Proporcional*

PI *Proporcional Integral*

PID *Proporcional Integral Derivativo*

PWM *Pulse Width Modulation (Modulação por largura de pulso)*

Sumário

Capítulo 1

Introdução

O controle de processos é um campo essencial da engenharia, responsável pela regulação e otimização do comportamento de sistemas dinâmicos, garantindo que operem de forma eficiente e estável. Esse controle é de extrema importância em diversas indústrias, onde a precisão e a confiabilidade são fatores cruciais para o desempenho dos processos produtivos e a qualidade dos produtos.

Dentre os métodos mais utilizados para o controle de sistemas industriais, destaca-se o controlador PID (Proporcional, Integral e Derivativo), amplamente empregado na regulação de motores de corrente contínua (CC). Esses motores desempenham um papel fundamental em inúmeras aplicações, que vão desde a automação industrial até dispositivos eletrônicos e sistemas mecatrônicos.

Neste contexto, o kit experimental formado por um par motor-gerador de motores CC, um transistor TIP120 e uma placa Arduino Nano se apresenta como uma ferramenta educativa relevante. Sua escolha se justifica pela acessibilidade e flexibilidade, permitindo que estudantes explorem, de forma prática e objetiva, os princípios de controle de processos. A possibilidade de realizar ajustes em tempo real facilita a compreensão tanto da teoria quanto da aplicação prática dos conceitos envolvidos.

O objetivo deste trabalho é implementar um sistema didático de baixo custo que permita a experimentação com o controle PID. Quando devidamente ajustado, esse controlador pode fornecer um controle preciso da velocidade de um par motor-gerador composto por motores CC, contribuindo para uma melhor compreensão dos conceitos de controle de processos.

1.1 Objetivos

1.1.1 Objetivo Geral

Demonstrar a aplicabilidade do controlador PID no controle de velocidade de motores CC em um sistema motor-gerador, utilizando uma bancada experimental de baixo custo.

1.1.2 Objetivos Específicos

1. Estudar os conceitos fundamentais de controle de processos, como malha aberta e malha fechada, e aplicá-los ao sistema motor-gerador;
2. Implementar a bancada experimental do sistema motor-gerador CC;
3. Implementar o controlador PID no Arduino para regular a velocidade do motor;
4. Analisar o desempenho do controle em presença de perturbações, ruído de medição e saturação dos atuadores;
5. Comparar os resultados obtidos com o controlador PID, nos diferentes tipos de malha, analisando o impacto de cada um no desempenho do sistema de controle de velocidade.

1.2 Organização do trabalho

Este trabalho foi organizado em capítulos que detalham os passos seguidos durante a implementação do sistema de controle de velocidade com controladores P, PI e PID, aprofundando-se em conceitos teóricos de controle de processos e definições matemáticas que servem como base para o modelo de controle utilizado.

O Capítulo 2 introduz os conceitos fundamentais de controle de processos, controlador PID e controle digital com o uso de microcontroladores. Explica também a modelagem de sistemas dinâmicos, a partir da descrição matemática dos motores CC.

O Capítulo 3 descreve a implementação da bancada experimental, quais componentes foram utilizados e os métodos utilizados para a coleta de dados e testes.

O Capítulo 4 apresenta o desempenho do controlador nos diferentes tipos de malha, compara em termos de tempo de resposta, estabilidade e precisão no controle da velocidade e mostra as vantagens e limitações de cada sistema.

Capítulo 2

Fundamentação Teórica

2.1 Controle de Processos

Controle de processos é um campo fundamental na engenharia, responsável por monitorar e ajustar variáveis de um sistema dinâmico para alcançar o comportamento desejado de forma eficiente. Esses sistemas incluem uma ou mais variáveis de controle, como a velocidade de um motor ou o nível de líquido em um tanque, que precisam ser mantidas dentro de limites específicos para garantir a operação adequada.

A implementação de sistemas de controle pode ocorrer em malha aberta ou malha fechada. Em uma malha aberta, o sistema realiza uma ação sem levar em conta a saída, logo, sem *feedback*. Esse tipo de controle é simples, mas pode não ser robusto, pois não responde a mudanças externas que podem alterar o desempenho do sistema. Por outro lado, sistemas em malha fechada utilizam a realimentação (*feedback*) para monitorar continuamente a saída e ajustá-la conforme necessário para reduzir a diferença em relação ao valor desejado. Esse tipo de controle permite respostas adaptáveis, sendo amplamente utilizado na indústria para garantir estabilidade e precisão em condições variáveis (Ogata (2010)).

O controle de processos desempenha, na indústria, um papel essencial para garantir qualidade, eficiência e segurança. Segundo Ogata (2010), o controle de variáveis críticas é essencial para assegurar a qualidade e eficiência em ambientes industriais, onde pequenas variações podem impactar diretamente na produtividade e segurança. O controle de velocidade em motores CC, por exemplo, é uma aplicação prática que ilustra como o controle em malha fechada pode manter a operação do sistema dentro dos parâmetros desejados, mesmo sob a influência de fatores externos.

2.2 Controladores PID

O controlador PID (Proporcional-Integral-Derivativo) é bastante utilizado em sistemas de controle industrial por causa da sua capacidade de ajustar variáveis de forma eficiente e robusta. Esse tipo de controlador trabalha com base em três ações fundamentais que atuam em conjunto para minimizar o erro entre a saída do sistema e o valor desejado (setpoint). Essas três ações são: a ação proporcional (P), a ação integral (I) e a ação derivativa (D). Conforme descrito por Franklin et al. (2013), cada componente tem uma função específica na resposta do sistema: a parte proporcional reage ao erro atual, a integral acumula os erros passados e a derivativa antecipa as futuras tendências do erro, contribuindo para uma resposta mais estável e precisa.

A equação geral do controlador PID pode ser expressa como:

$$u(t) = K_p \cdot e(t) + K_i \int e(t)\, dt + K_d \frac{de(t)}{dt} \tag{2.1}$$

Onde $u(t)$ é a saída do controlador, $e(t)$ é o erro, que é a diferença entre o setpoint e a saída real), K_p, K_i e K_d são, respectivamente, os ganhos proporcional, integral e derivativo.

Esses ganhos determinam a intensidade da ação de cada componente sobre o sistema. No controle de velocidade de motores CC, como no kit experimental implementado neste trabalho, o controlador PID ajusta a largura do sinal PWM (modulação por largura de pulso), que controla a tensão média aplicada ao motor e, consequentemente, a sua velocidade.

Para ajustar os parâmetros K_p, K_i e K_d de modo a obter um desempenho adequado, métodos de sintonia como o de Ziegler-Nichols são frequentemente aplicados. Este método consiste em observar a resposta do sistema a uma condição de oscilação sustentada e, a partir daí, determinar valores iniciais para os parâmetros PID. Embora esse método forneça uma solução prática e rápida, ele pode introduzir oscilações iniciais ou leve overshoot, o que pode ser aceitável dependendo da aplicação Dorf (2017).

A sintonia do PID afeta diretamente o comportamento dinâmico do sistema, influenciando aspectos como o tempo de subida (tempo para a variável de controle atingir o valor desejado), o erro em regime permanente (diferença entre o valor de saída e o setpoint após a estabilização) e a estabilidade (capacidade do sistema de retornar ao equilíbrio após uma perturbação). Ajustes precisos dos parâmetros PID podem reduzir significativamente o erro em regime e melhorar a estabilidade do sistema, porém, ajustes inadequados podem gerar oscilações indesejadas ou até mesmo tornar o sistema instável. Como ressaltado por Franklin et al. (2013), a combinação dessas três ações permite que o controlador PID alcance um equilíbrio entre precisão e robustez.

2.3 Modelagem de Sistemas Dinâmicos

Segundo Ogata (2010), a modelagem matemática de sistemas dinâmicos envolve a criação de um conjunto de equações que descreve a dinâmica do sistema de forma precisa ou razoavelmente representativa. Um dos atuadores mais comuns utilizados no controle de sistemas é o motor CC, que permite o controle de movimento rotativo com precisão (Franklin et al. (2013)).

2.3.1 Modelagem do Motor CC

A modelagem de motores de corrente contínua (CC) utiliza equações dinâmicas que descrevem tanto os aspectos elétricos quanto os mecânicos do sistema. A análise começa a partir da Lei de Kirchhoff, que permite formular equações que representam o comportamento mecânico do motor, dividindo-o em elétrico e mecânico.

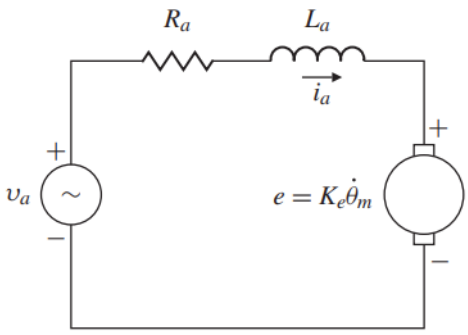

Figura 2.1: Circuito elétrico da armadura. Fonte: Franklin et al. (2013)

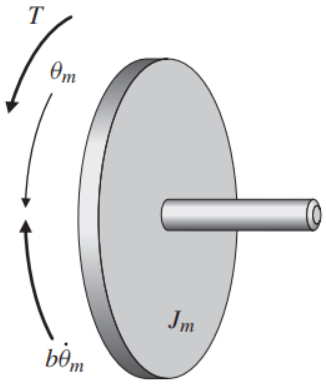

Figura 2.2: Diagrama de corpo livre do rotor. Fonte: Franklin et al. (2013)

Aspecto Elétrico

A equação dinâmica do aspecto elétrico do motor CC considera a tensão de entrada Vi(t) aplicada à armadura. Essa tensão é a soma das quedas de tensão causadas pela resistência e pela indutância da armadura, além da força contra eletromotriz (FCEM):

$$V_i(t) = R_a \cdot i_a + L_a \cdot i_a + V_{\text{cem}} \tag{2.2}$$

Onde R_a é a resistência da armadura, i_a é a corrente da armadura, L_a é a indutância da armadura e V_{cem} é a força contra eletromotriz, que é proporcional à velocidade angular do motor.

Essa equação reflete como a tensão na resistência, a resposta dinâmica da indutância e a força contra eletromotriz gerada pela rotação do motor.

Aspecto Mecânico

A equação dinâmica do aspecto mecânico do motor CC relaciona o momento de inércia do rotor com a variação da velocidade angular:

$$J \cdot \dot{\omega}(t) = T_g(t) - T_f(t) - T_l(t) \tag{2.3}$$

Onde J é o momento de inércia do rotor, $\omega(t)$ é a velocidade angular, T_g, T_f e T_l são, respectivamente, os torques gerado pelo motor, de fricção e de carga.

Essa equação descreve que a variação do momento angular do motor depende do torque gerado menos as forças de resistência, como fricção e torque de carga. O torque Tg(t) é proporcional à corrente de armadura, conectando os aspectos elétricos e mecânicos do motor.

Funções de transferência para motor CC

Segundo Ogata (2010), a função de transferência é muito utilizada na análise de sistemas dinâmicos lineares, porque permite relacionar a entrada e a saída do sistema no domínio da frequência por meio da transformada de Laplace, sendo essencial para projetar e analisar controladores.

$$G(s) = \frac{Y(s)}{U(s)} \tag{2.4}$$

Onde $G(s)$ é a função de transferência, $Y(s)$ e $U(s)$ são, respectivamente, a saída e a entrada do sistema no domínio de Laplace e s é a variável complexa do domínio de Laplace.

Para motores CC, a função de transferência é utilizada para descrever a dinâmica do motor, considerando seus aspectos elétricos e mecânicos. Isso permite prever como o motor responderá a diferentes sinais de controle, como a aplicação de uma tensão, e é fundamental para projetar controladores eficientes, como P, PI e PID.

As equações dinâmicas 2.2 e 2.3 são transformadas para o domínio da frequência utilizando a Transformada de Laplace, assim o sistema é representado de forma algébrica. No domínio de Laplace, as derivadas das variáveis no tempo são convertidas em multiplicações pela variável complexa s, o que facilita a manipulação e resolução das equações diferenciais.

$$V_i(s) = R_a I_a(s) + L_a I_a(s) \cdot s + k\Omega(s) \tag{2.5}$$

$$J\Omega(s) \cdot s = kI_a(s) - T_f(s) - T_l(s) \tag{2.6}$$

Com base nas equações 2.5 e 2.6, determinam-se duas funções de transferência nas

formas:

$$\frac{\Omega(s)}{V_i(s)} \tag{2.7}$$

$$\frac{\Omega(s)}{T_l(s)} \tag{2.8}$$

A equação 2.7 representa o comportamento dinâmico da velocidade do eixo do motor em função de variações na tensão de armadura, enquanto a equação 2.8 indica como variações no conjugado de carga afetam a velocidade do eixo do motor.

Para encontrar as duas funções de transferência acima, as equações 2.5 e 2.6 são usadas como bases e, nesse caso, como não é necessário saber a corrente da armadura (I_a), essa variável pode ser eliminada. Como ela está presente nas duas equações, é preciso achar uma expressão para I_a a partir da equação 2.5 e, depois, substituí-la na equação 2.6.

$$I_a = \frac{[V_i(s) - k\Omega(s)]}{R_a - L_a \cdot s} \tag{2.9}$$

Substituindo a equação na 2.6:

$$\Omega(s) = \frac{k}{JL_as^2 + JL_as + k^2} \cdot V_i(s) - \frac{R_a + L_a}{JL_as^2 + JL_as + k^2} \cdot T_l(s) \tag{2.10}$$

Logo,

$$\frac{\Omega(s)}{V_i(s)} = \frac{k}{JL_as^2 + JL_as + k^2} \tag{2.11}$$

$$\frac{\Omega(s)}{T_l(s)} = \frac{-R_a + L_a}{JL_as^2 + JL_as + k^2} \tag{2.12}$$

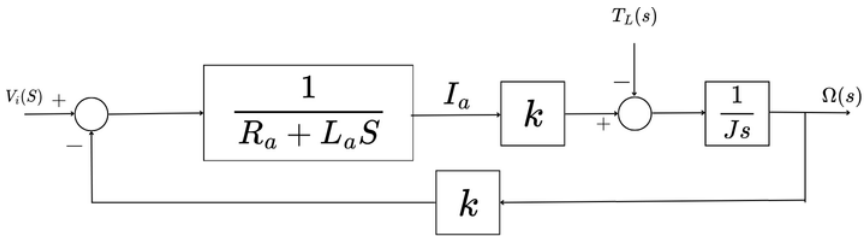

Figura 2.3: Diagrama de blocos. Fonte: autora, 2024

2.4 Modulação PWM (Pulse Width Modulation)

A modulação por largura de pulso (PWM) é bastante utilizada no controle de sistemas eletrônicos e eletromecânicos. Essa modulação permite a variação da potência média fornecida a uma carga, ajustando a largura dos pulsos de uma onda retangular, enquanto

a frequência é mantida constante. No contexto de controle de motores CC, a PWM é uma ferramenta eficaz para regular a velocidade do motor, fornecendo energia de maneira ajustável.

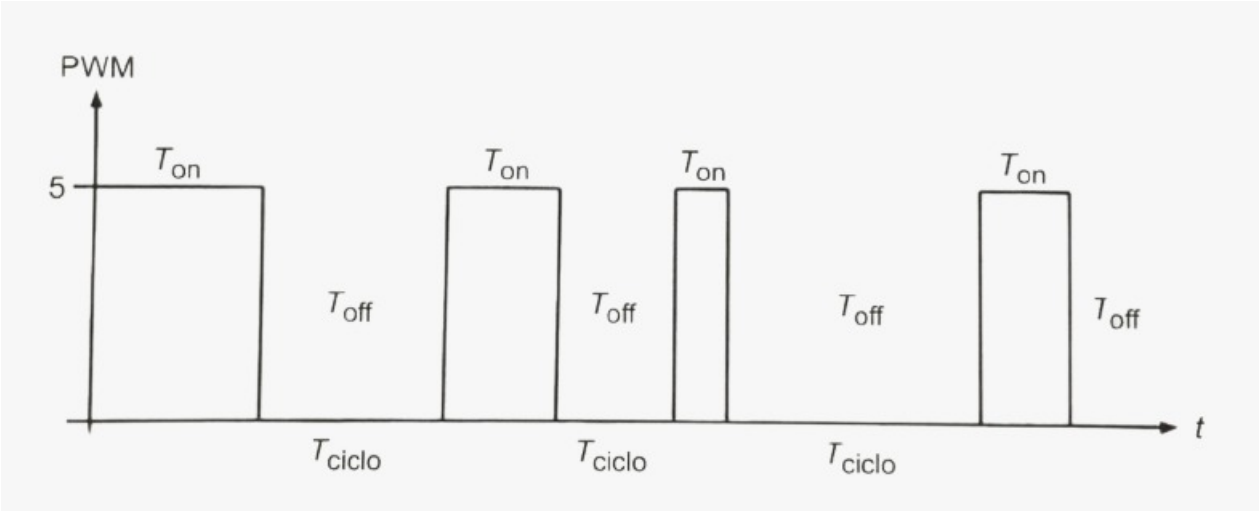

Figura 2.4: Sinal do tipo PWM. Fonte: Normey-Rico, J. E.; Morato, M. M. (2021)

Capítulo 3

Metodologia

3.1 Circuito de controle de velocidade

O sistema didático implementado nesse trabalho é composto por um Arduino Nano, que atua como unidade de controle, um transistor TIP120 para amplificar o sinal de controle PWM, dois motores de corrente contínua de 12V conectados em uma configuração motor-gerador, e componentes adicionais: diodo, resistor, protoboard e cabos jumpers. Esse circuito permite controlar a velocidade do motor de acionamento enquanto o segundo motor atua como gerador, produzindo uma tensão proporcional à rotação, que é usada como feedback para o sistema de controle.

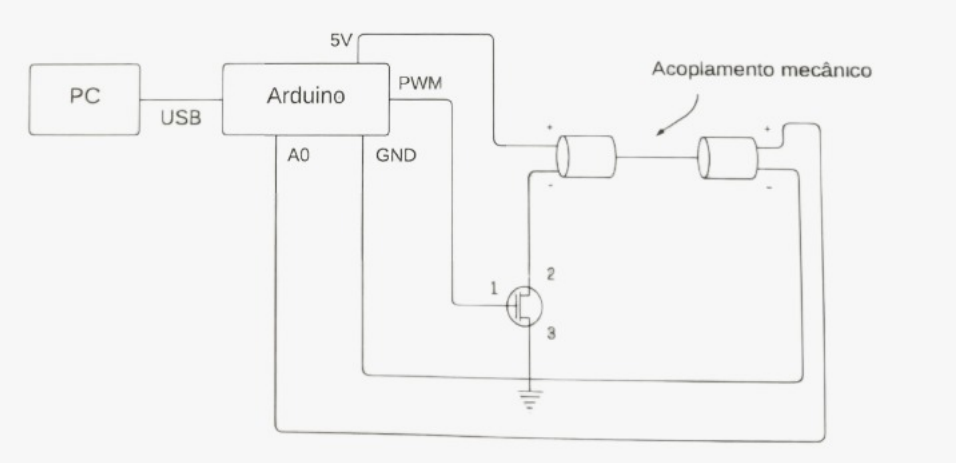

Figura 3.1: Circuito de implementação. Fonte: Normey-Rico, J. E.; Morato, M. M. (2021)

3.1.1 Arduino Nano

O Arduino Nano é uma placa compacta da Arduino, projetada para uso em protoboards, com conectores de pinos que facilitam sua conexão e um conector USB Mini-B para programação e alimentação (Arduino (2023)).

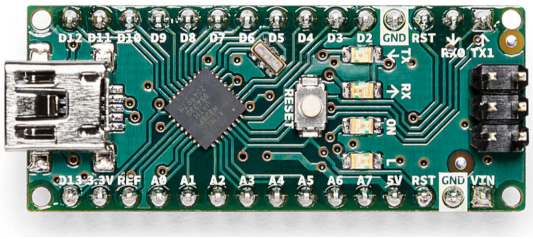

Figura 3.2: Arduino Nano. Fonte: Arduino (2023).

3.1.2 Motores CC de 12V

Dois motores CC de 12V são utilizados, onde um motor atua como o motor de acionamento e o outro como motor gerador. O motor de acionamento é controlado pelo Arduino através de um sinal PWM, enquanto o motor gerador é acoplado mecanicamente ao motor de acionamento, produzindo uma tensão proporcional à sua rotação. Essa tensão é medida e usada como feedback para o sistema de controle.

3.1.3 Transistor TIP120

O TIP120 é um transistor Darlington NPN que funciona como um interruptor controlado pelo Arduino, permitindo amplificar o sinal PWM para acionar o motor de acionamento. Como o Arduino não consegue fornecer corrente suficiente para acionar diretamente o motor, o TIP120 atua como um amplificador, modulando a corrente do motor com base no sinal de controle PWM gerado pelo Arduino. O TIP120 é capaz de controlar correntes maiores do que um transistor comum, tornando-o ideal para essa aplicação.

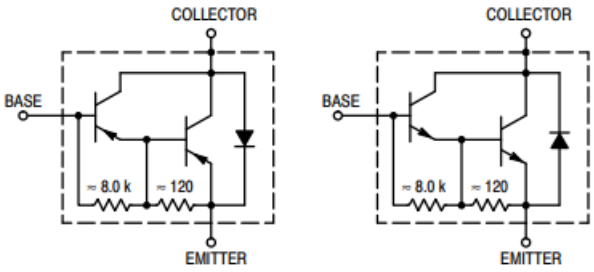

Figura 3.3: Circuito esquemático do Darlington. Fonte: Onsemi (2022).

3.1.4 Diodo

O diodo é colocado em paralelo com o motor de acionamento para proteger o TIP120 de picos de corrente reversa que ocorrem quando o motor é desligado.

3.1.5 Resistor de 1kΩ

O resistor de 1kΩ é utilizado entre o pino de controle do Arduino e a base do TIP120 para limitar a corrente de entrada no transistor. Isso ajuda a proteger o Arduino e o TIP120 de sobrecarga e oscilações indesejadas no sinal de controle.

3.2 Softwares utilizados

A Arduino IDE foi utilizada como ambiente de desenvolvimento para a programação do Arduino Nano. Com ela, foi desenvolvido o código para implementar o controle de velocidade do sistema motor-gerador com controlador PID e enviar o código para o Arduino, configurando o sinal PWM, que controla a velocidade do motor de acionamento, e ajustando os parâmetros de controle conforme a resposta do sistema. Também foi utilizado o Serial Monitor e do Serial Plotter para visualizar os dados em tempo real. Isso incluiu gráficos de medição, atuação e referência.

O MATLAB, na versão R2021B, foi usado para importar e analisar os dados coletados pela Arduino IDE, para gerar gráficos detalhados da resposta do sistema para cada tipo de controlador e compará-los quanto a eficiência.

3.3 Experimento

O experimento começou com a seleção e preparação dos componentes para a implementação do sistema de controle. O microcontrolador Arduino Nano foi escolhido por sua versatilidade e capacidade de executar algoritmos de controle de forma eficiente. A amplificação do sinal PWM foi realizada por meio do transistor TIP120, que permitiu o controle da corrente fornecida ao motor de acionamento, garantindo que o Arduino pudesse controlar o motor sem sobrecarregar suas saídas.

A montagem do sistema utilizou dois motores CC de 12V, onde um funcionou como o motor de acionamento principal e o outro como motor gerador, responsável por fornecer um feedback de tensão, fundamental para monitorar a resposta em tempo real. Também foi utilizada uma protoboard para facilitar a conexão entre os componentes.

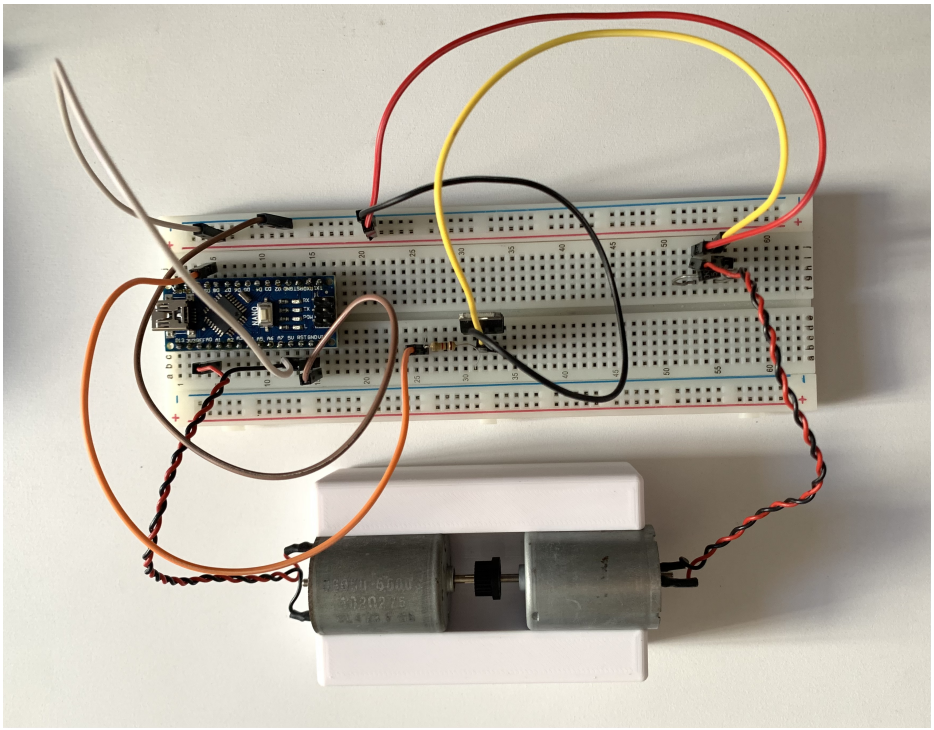

Figura 3.4: Foto do sistema didático. Fonte: autora, 2024.

O circuito foi alimentado por uma fonte de 5V, conectada ao pino de alimentação do Arduino, que por sua vez distribuía a tensão para o motor de acionamento. A conexão do TIP120 foi realizada de forma que o sinal PWM, gerado pelo pino de saída do Arduino, controlasse sua base, modulando a corrente que passava pelo motor de acionamento. A tensão gerada pelo motor gerador foi direcionada ao Arduino para fornecer o feedback de medição necessário para os algoritmos de controle implementados, garantindo um loop de controle em malha fechada.

O algoritmo de controle foi desenvolvido na Arduino IDE, implementando as estratégias de controle proporcional-integral-derivativo (PID). A programação começou com a leitura do sinal de feedback do motor gerador, conectado ao pino analógico A0 do Arduino. Esse sinal era convertido de ADC para tensão em volts e comparado com o valor de referência, previamente definido, para calcular o erro. Com base nesse erro, os ganhos K_p, K_i e K_d eram aplicados pelo controlador para gerar o sinal de controle proporcional ao comportamento desejado do motor. Esse sinal era convertido em um pulso PWM no pino D9, modulando a tensão aplicada ao motor de acionamento.

O ajuste dos ganhos K_p, K_i e K_d foi realizado manualmente, começando com o controlador proporcional. Para isso, valores baixos de K_p foram testados e gradualmente aumentados até que a medição se aproximasse da referência, mesmo que apresentasse oscilações. Em seguida, o controlador PI foi ajustado adicionando K_i para eliminar o erro em regime permanente. Por fim, o controlador PID foi configurado, utilizando K_d para reduzir oscilações e melhorar a estabilidade.

Os dados coletados durante o experimento foram enviados do Arduino para o MATLAB via monitor serial, permitindo a captura de variáveis como medição (em ADC e volts), erro, sinal de controle (PWM) e referência (em volts). Esses dados foram armazenados em arquivos CSV para facilitar a análise posterior. No MATLAB, gráficos foram gerados para avaliar a evolução da medição, do sinal de controle e do erro ao longo do tempo. Essa abordagem possibilitou uma análise detalhada do desempenho do controlador PID, destacando diferenças em termos de tempo de resposta, estabilidade e precisão.

Capítulo 4

Resultados e Discussões

A análise comparativa dos sistemas em malha aberta e malha fechada foi realizada para compreender as vantagens e limitações de cada abordagem no controle de sistemas dinâmicos. Inicialmente, o sistema foi configurado em malha aberta, alimentando o motor de acionamento com um sinal PWM fixo em 200 (considerando a escala de 0 a 255) e sem utilizar a realimentação para ajustes. Em seguida, foi implementada a configuração de malha fechada, que integrou um mecanismo de realimentação para adaptar continuamente o sinal de controle, também utilizando um sinal PWM fixo em 200. Por fim, um controlador PID foi aplicado para melhorar a resposta do sistema, buscando maior estabilidade e precisão.

4.1 Sistema em Malha Aberta

Em malha aberta, o sistema funciona sem a capacidade de ajustar sua saída em função de desvios ou perturbações. O sinal PWM fixo aplicado ao motor de acionamento não leva em consideração o comportamento dinâmico do sistema.

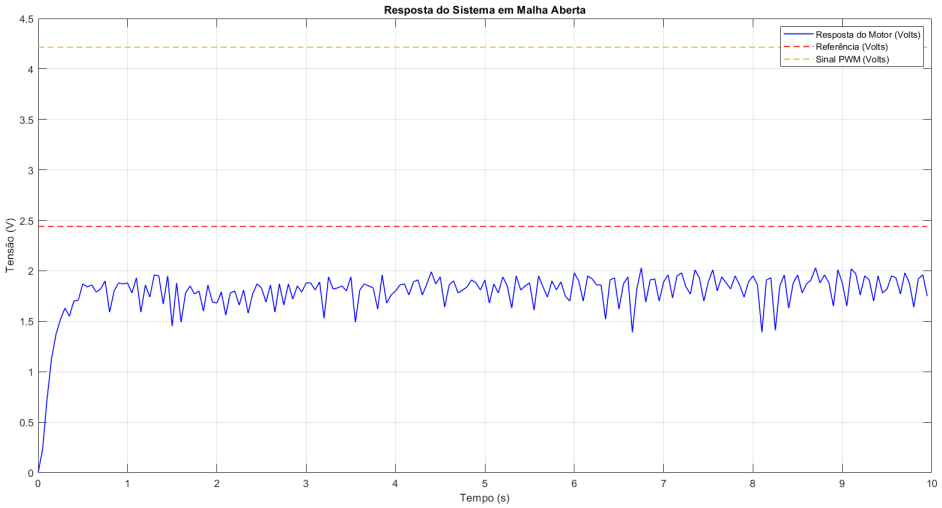

Figura 4.1: Gráfico da resposta do sistema em malha aberta. Fonte: autora, 2024.

Os resultados apresentados na figura 4.1 mostram que a tensão gerada pelo motor gerador estabilizou-se em torno de 1,5 V, valor significativamente inferior à referência estipulada de 2,44 V. Esse comportamento reflete as limitações do sistema em malha aberta, que não possui mecanismos de correção de desvios ou ajuste dinâmico para atingir valores predefinidos com precisão, comprometendo sua aplicação em sistemas que exigem maior estabilidade e controle. Além disso, observa-se que, mesmo com o aumento do PWM fixo, a tensão gerada pelo motor não se aproxima da referência. Esse problema está diretamente relacionado ao fato de o motor ser projetado para operar a 12V, enquanto o sistema é alimentado apenas com 5V pelo Arduino, limitando o desempenho e a resposta do motor às condições impostas pelo experimento.

4.2 Sistema em Malha Fechada

A transição para o sistema em malha fechada adicionou o uso de realimentação, permitindo que o sistema ajustasse de forma contínua o sinal PWM com base na diferença entre a medição e o valor de referência.

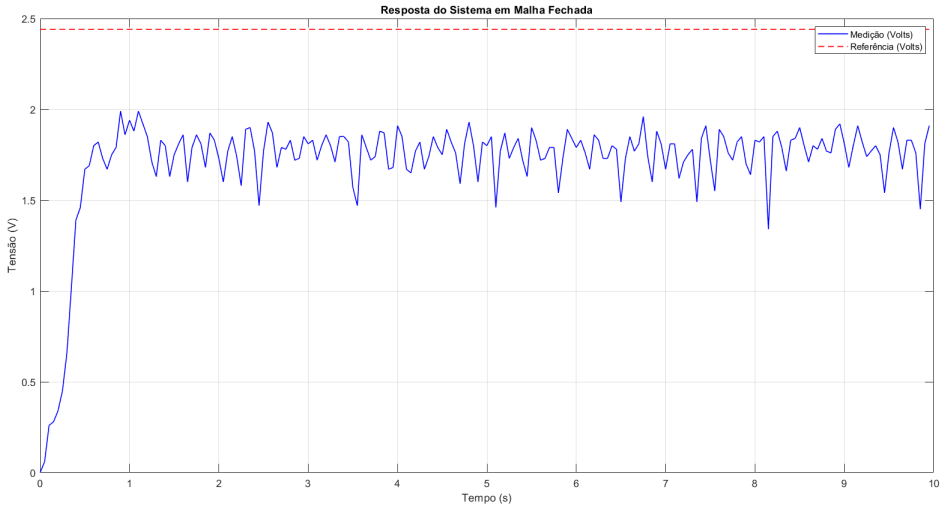

Figura 4.2: Gráfico da resposta do sistema em malha fechada. Fonte: autora, 2024.

Na figura 4.2, observou-se uma melhora significativa na resposta do sistema. A tensão gerada pelo motor gerador passou a se aproximar do valor de referência de maneira mais consistente. No entanto, oscilações iniciais foram observadas, indicando a necessidade de ajustes mais detalhados nos parâmetros de controle para reduzir o tempo de estabilização.

4.3 Desempenho com Controlador PID

Com a implementação do controlador PID com os ganhos $K_p = 1.0$, $K_i = 0.1$, $K_d = 0.2$), o sistema em malha fechada apresentou uma resposta mais próxima do valor de referência.

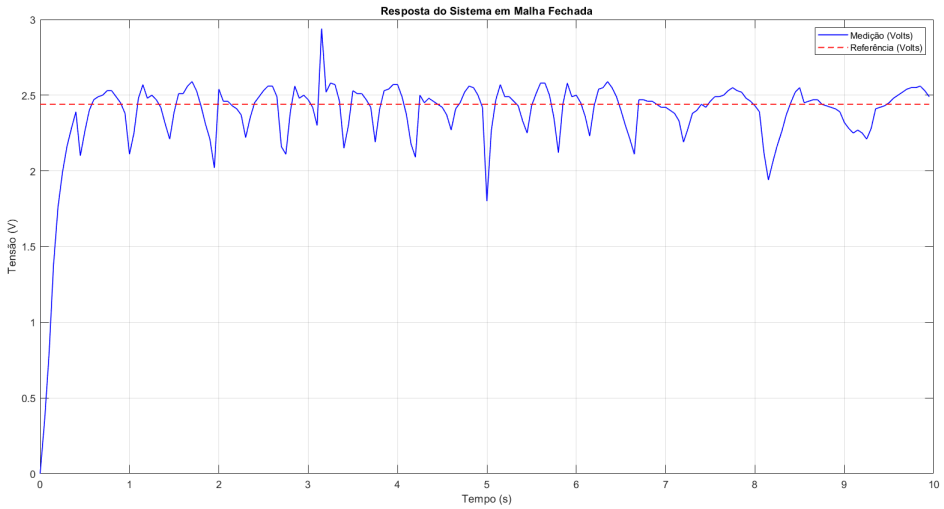

Figura 4.3: Gráfico da resposta do sistema em malha fechada com controlador PID. Fonte: autora, 2024.

Conforme apresentado na figura 4.3, observou-se uma melhora considerável na resposta do sistema. A tensão gerada pelo motor gerador passou a se aproximar do valor de referência de maneira mais consistente. No entanto, o sistema oscilou bastante, indicando a necessidade de ajustes mais detalhados nos parâmetros de controle para reduzir o tempo de estabilização.

4.4 Comparação Geral

A comparação entre as abordagens confirma a superioridade do sistema em malha fechada, especialmente quando associado a um controlador PID. O sistema em malha aberta, embora simples de implementar, apresenta limitações claras em sua capacidade de atender aos valores de referência e responder a perturbações. Por outro lado, o sistema em malha fechada, com realimentação ativa, demonstrou maior capacidade de correção, enquanto o controlador PID refinou ainda mais a resposta, equilibrando o tempo de resposta, a estabilidade e a precisão.

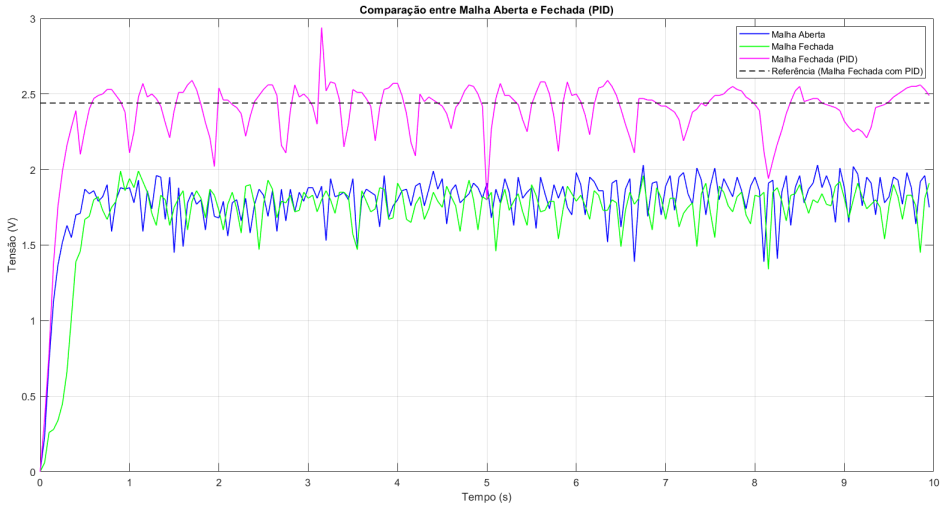

Figura 4.4: Gráfico comparativo dos três sistemas. Fonte: autora, 2024.

Capítulo 5

Conclusão

Neste trabalho, foi realizada a implementação e análise comparativa de controlador PID aplicados ao controle de velocidade de um sistema motor-gerador de corrente contínua (CC). A partir de uma bancada experimental de baixo custo, composta por um Arduino Nano, transistor TIP120, dois motores CC de 12V e outros componentes acessíveis, foi possível explorar de maneira prática os conteúdos introdutórios de controle de processos.

Os resultados evidenciaram que o sistema em malha aberta é limitado para aplicações que exigem precisão e estabilidade, devido à incapacidade de compensar desvios do valor de referência. Já na malha fechada, o sistema demonstrou melhoria significativa na capacidade de acompanhar a referência, sobretudo com a inclusão do controlador PID, que apresentou o menor tempo de resposta e o que conseguiu manter o sistema próximo à referência estabelecida.

A plataforma desenvolvida também demonstrou ser uma ferramenta educacional eficaz, permitindo a visualização prática dos conceitos teóricos de controle de processos e oferecendo suporte ao aprendizado de metodologias de ajuste e análise de controladores.

5.1 Trabalhos Futuros

Embora os objetivos principais deste trabalho tenham sido alcançados, algumas melhorias e expansões podem ser consideradas para estudos futuros:

- A implementação de sensores dedicados à medição da velocidade angular permitirá maior precisão nos dados coletados, proporcionando análises mais completas sobre o comportamento dinâmico do sistema.

- Incluir a aplicação de métodos de sintonia automática, como o método de Ziegler-Nichols, para a determinação dos ganhos dos controladores, reduzindo o tempo gasto com calibrações manuais e aprimorando o desempenho do sistema.

Referências Bibliográficas

Arduino (2023). *Arduino Nano Documentation*. Recuperado de `https://docs.arduino.cc/hardware/nano`.

Dorf, R. C.; Bishop, R. H. (2017). *Modern Control Systems*. Pearson.

Franklin, G. F., Powell, J. D., and Emami-Naeini, A. (2013). *Sistemas de controle para engenharia*. Bookman.

Normey-Rico, J. E.; Morato, M. M. (2021). *Introdução ao Controle de Processos*. Editora Blucher.

Ogata, K. (2010). *Modern Control Engineering*. Prentice Hall.

Onsemi (2022). *TIP120 - Plastic Medium-Power Complementary Silicon Transistors*. `https://www.onsemi.com/pdf/datasheet/tip120-d.pdf`.

Apêndice A

Código do Arduino - Malha Aberta

```
#define MEDICAO A0
#define ATUACAO 9

const int pwmValor = 200; //(0 - 255)

void setup() {
    Serial.begin(9600);
    pinMode(ATUACAO, OUTPUT);
    pinMode(MEDICAO, INPUT);
    Serial.println("Medicao_ADC,Medicao_Volts,PWM_Aplicado");
}

void loop() {
    analogWrite(ATUACAO, pwmValor);

    int medicao = analogRead(MEDICAO);
    float medicaoVolts = (medicao * 5.0) / 1023;

    Serial.print(medicao);
    Serial.print(",");
    Serial.print(medicaoVolts);
    Serial.print(",");
    Serial.println(pwmValor);

    delay(50); // 50ms
}
```

Apêndice B

Código do Arduino - Malha Fechada

```
#define MEDICAO A0
#define ATUACAO 9

void setup() {
    Serial.begin(9600);
    pinMode(ATUACAO, OUTPUT);
    pinMode(MEDICAO, INPUT);

    Serial.println("Medicao_ADC,Medicao_Volts,PWM_Fixo,Referencia_Volts");
}

void loop() {
    int medicao = analogRead(MEDICAO);
    float medicaoVolts = (medicao * 5.0) / 1023;

    // PWM fixo aplicado ao motor de acionamento
    int pwmFixo = 215;
    analogWrite(ATUACAO, pwmFixo);

    float referenciaVolts = 2.44;

    Serial.print(medicao);
    Serial.print(",");
    Serial.print(medicaoVolts);
    Serial.print(",");
    Serial.print(pwmFixo);
    Serial.print(",");
    Serial.println(referenciaVolts);

    delay(50);
}
```

Apêndice C

Código do Arduino - Controlador PID

```
#define MEDICAO A0
#define ATUACAO 9

volatile const float Kp = 1.0;
volatile const float Ki = 0.1;
volatile const float Kd = 0.2;

float erroAcumulado = 0;
float erroAnterior = 0;

int referencia = 500; // (2.44V)

void setup() {
    Serial.begin(9600);
    pinMode(ATUACAO, OUTPUT);
    pinMode(MEDICAO, INPUT);

    Serial.println("Medicao_ADC,Medicao_Volts,Erro,Atuacao_PWM,Referencia_Volts");
}

void loop() {
    int medicao = analogRead(MEDICAO);
    float medicaoVolts = (medicao * 5.0) / 1023;

    float referenciaVolts = (referencia * 5.0) / 1023;

    float erro = referencia - medicao;

    float atuacao = controlePID(erro);
    atuacao = constrain(atuacao, 0, 255);
    analogWrite(ATUACAO, atuacao);

    float atuacaoVolts = (atuacao * 5.0) / 255;

    Serial.print(medicao);
    Serial.print(",");
    Serial.print(medicaoVolts);
    Serial.print(",");
    Serial.print(erro);
    Serial.print(",");
    Serial.print(atuacaoVolts);
    Serial.print(",");
```

```
    Serial.println(referenciaVolts);

    delay(50);
}

float controlePID(float erro) {
    float termoP = Kp * erro;
    erroAcumulado += erro;
    float termoI = Ki * erroAcumulado;
    float termoD = Kd * (erro - erroAnterior);
    erroAnterior = erro;

    return termoP + termoI + termoD;
}
```

Apêndice D

Códigos do MATLAB - Gráficos

```
dados = readtable('malha_aberta.csv');
dados2 = readtable('malha_fechada.csv');

tempo = (0:height(dados)-1) * 0.05; % (50ms)
medicaoVolts = dados.Medicao_Volts;
referenciaVolts = dados2.Referencia_Volts;
pwmAplicado = dados.PWM_Aplicado; % pwm aplicado ao motor

figure;
plot(tempo, medicaoVolts, 'b', 'LineWidth', 1.0,
    'DisplayName', 'Resposta do Motor (Volts)');
hold on;
plot(tempo, referenciaVolts, 'r--', 'LineWidth', 1.0,
    'DisplayName', 'Refer ncia (Volts)');
hold on;
plot(tempo, pwmAplicado * (5.0 / 255), 'Apendi--',
    'LineWidth', 1.0, 'DisplayName', 'Sinal PWM (Volts)');
hold off;

title('Resposta do Sistema em Malha Aberta');
xlabel('Tempo (s)');
ylabel('Tens o (V)');
legend('show');
grid on;

dados = readtable('malha_fechada.csv');

tempo = (0:height(dados)-1) * 0.05; % (50ms)
```

```
medicaoVolts = dados.Medicao_Volts;
referenciaVolts = dados.Referencia_Volts;
atuacaoPWM = dados.PWM_Fixo;

figure;
plot(tempo, medicaoVolts, 'b', 'LineWidth', 1.0,
    'DisplayName', 'Medi  o (Volts)');
hold on;
plot(tempo, referenciaVolts, 'r--', 'LineWidth', 1.0,
    'DisplayName', 'Refer ncia (Volts)');
hold off;
title('Resposta do Sistema em Malha Fechada');
xlabel('Tempo (s)');
ylabel('Tens o (V)');
legend('show');
grid on;

dados = readtable('dados_pid.csv');

tempo = (0:height(dados)-1) * 0.05;
medicaoVolts = dados.Medicao_Volts;
atuacaoPWM = dados.Atuacao_PWM;
referenciaVolts = dados.Referencia_Volts;

% medi  o vs refer ncia
figure;
plot(tempo, medicaoVolts, 'b', 'LineWidth', 1.0,
    'DisplayName', 'Medi  o (Volts)');
hold on;
plot(tempo, referenciaVolts, 'r--', 'LineWidth', 1.0,
    'DisplayName', 'Refer ncia (Volts)');
hold off;

title('Resposta do Sistema em Malha Fechada');
xlabel('Tempo (s)');
ylabel('Tens o (V)');
legend('show');
grid on;
```

```
% Sinal de controle (pwm)
figure;
plot(tempo, atuacaoPWM * (5.0 / 255), 'g', 'LineWidth', 1.0,
    'DisplayName', 'Sinal de Controle (Volts)');
title('Sinal de Controle (PWM)');
xlabel('Tempo (s)');
ylabel('Tens o (V)');
legend('show');
grid on;

% Medi  o, refer ncia e controle
figure;
plot(tempo, medicaoVolts, 'b', 'LineWidth', 1.0,
    'DisplayName', 'Medi  o (Volts)');
hold on;
plot(tempo, referenciaVolts, 'r--', 'LineWidth', 1.0,
    'DisplayName', 'Refer ncia (Volts)');
plot(tempo, atuacaoPWM * (5.0 / 255), 'g:', 'LineWidth', 1.0,
    'DisplayName', 'Controle (Volts)');
hold off;

title('Gr fico Combinado: Medi  o, Refer ncia e Controle');
xlabel('Tempo (s)');
ylabel('Tens o (V)');
legend('show');
grid on;

dadosMalhaAberta = readtable('malha_aberta.csv');
dadosMalhaFechada = readtable('malha_fechada.csv');
dadosPID = readtable('dados_pid.csv');

tempoAberta = (0:height(dadosMalhaAberta)-1) * 0.05;
tempoFechada = (0:height(dadosMalhaFechada)-1) * 0.05;
tempoPID = (0:height(dadosPID)-1) * 0.05;

medicaoAberta = dadosMalhaAberta.Medicao_Volts;
medicaoFechada = dadosMalhaFechada.Medicao_Volts;
referenciaFechada = dadosMalhaFechada.Referencia_Volts;
medicaoPID = dadosPID.Medicao_Volts;
```

```
referenciaPID = dadosPID.Referencia_Volts;

figure;
plot(tempoAberta, medicaoAberta, 'b', 'LineWidth', 1.0,
    'DisplayName', 'Malha Aberta');
hold on;
plot(tempoFechada, medicaoFechada, 'g', 'LineWidth', 1.0,
    'DisplayName', 'Malha Fechada');
plot(tempoPID, medicaoPID, 'm', 'LineWidth', 1.0,
    'DisplayName', 'Malha Fechada (PID)');
plot(tempoPID, referenciaPID, 'k--', 'LineWidth', 1.0,
    'DisplayName', 'Refer ncia (Malha Fechada com PID)');
hold off;

title('Compara o entre Malha Aberta e Fechada (PID)');
xlabel('Tempo (s)');
ylabel('Tens o (V)');
legend('show');
grid on;
```