UNIVERSIDADE FEDERAL DO PARÁ
CAMPUS UNIVERSITÁRIO DE TUCURUÍ
FACULDADE DE ENGENHARIA ELÉTRICA

RAISSA STEPHANY TEIXEIRA PAIVA

FILTRO SALLEN-KEY COMO POCKET-LAB PARA ENSINO DE IDENTIFICAÇÃO E CONTROLE DE SISTEMAS DINÂMICOS

TUCURUÍ

2025

RAISSA STEPHANY TEIXEIRA PAIVA

FILTRO SALLEN-KEY COMO POCKET-LAB PARA ENSINO DE IDENTIFICAÇÃO E CONTROLE DE SISTEMAS DINÂMICOS

Trabalho de Conclusão de Curso, apresentado como requisito parcial para a obtenção de grau de Bacharel em Engenharia Elétrica, pela Universidade Federal do Pará.

Orientador:
Prof. Dr. Raphael Barros Teixeira.

TUCURUÍ

2025

Dados Internacionais de Catalogação na Publicação (CIP) de acordo com ISBD
Sistema de Bibliotecas da Universidade Federal do Pará
Gerada automaticamente pelo módulo Ficat, mediante os dados fornecidos pelo(a) autor(a)

T266f TEIXEIRA, RAISSA.
FILTRO SALLEN-KEY COMO POCKET-LAB PARA ENSINO DE IDENTIFICAÇÃO E CONTROLE DE SISTEMAS DINÂMICOS / RAISSA TEIXEIRA. — 2025.
XI, 55 f. : il. color.

Orientador(a): Prof. Dr. Raphael Barros Teixeira Teixeira
Trabalho de Conclusão (Graduação) - Universidade Federal do Pará, Campus Universitário de Tucuruí, Faculdade de Engenharia Elétrica, Tucuruí, 2025.

1. SALLEN-KEY. 2. SISTEMAS LINEARES. 3. ARDUINO. I. Título.

CDD 629.8312

RAISSA STEPHANY TEIXEIRA PAIVA

FILTRO SALLEN-KEY COMO POCKET-LAB PARA ENSINO DE IDENTIFICAÇÃO E CONTROLE DE SISTEMAS DINÂMICOS

Trabalho de Conclusão de Curso, apresentado como requisito parcial para a obtenção de grau de Bacharel em Engenharia Elétrica, pela Universidade Federal do Pará.

Data de aprovação: 23/09/2025

Banca Examinadora:

gov.br
Documento assinado digitalmente
RAPHAEL BARROS TEIXEIRA
Data: 03/10/2025 11:20:21-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. Raphael Barros Teixeira
Orientador - FEE/CAMTUC/UFPA

gov.br
Documento assinado digitalmente
ANDRE FELIPE SOUZA DA CRUZ
Data: 02/10/2025 15:08:00-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. André Felipe Souza da Cruz
Avaliador Interno - FEE/CAMTUC/UFPA

gov.br
Documento assinado digitalmente
DENILSON COSTA DA SILVA
Data: 26/09/2025 20:20:00-0300
Verifique em https://validar.iti.gov.br

Eng. Denilson Costa da Silva
Avaliador Externo - Vale

TUCURUÍ

2025

Dedico todo este trabalho e trajetória a Deus,
pois sem ele nada disso seria possível.

AGRADECIMENTOS

Agradeço, primeiramente a Deus, pois sem ele nada disso seria possível.

Aos meus pais, Fabrício Coutinho e Cristiane Teixeira, por todo o apoio que me ofereceram e por acreditarem em mim. Sou eternamente grata.

Ao meu orientador, Prof. Dr. Raphael Teixeira, pela paciência, orientação e ótimos ensinamentos que me fizeram caminhar nessa jornada acadêmica.

A minha banca composta pelo Prof. Dr. André Cruz e o Eng. Denilson Silva, pela disponibilidade de avaliar o meu trabalho e apresentação.

Aos professores da UFPA-CAMTUC, que têm excelentes profissionais, que compartilham seus conhecimentos para a formação de novos profissionais.

Aos meus amigos que estiveram comigo durante minha trajetória acadêmica, onde compartilhamos conhecimentos e momentos únicos. Mesmo nas dificuldades sempre nos ajudavamos em união. Sempre me lembrarei de todos vocês, dos momentos bons e difíceis.

RESUMO

Este trabalho aborda o desenvolvimento de um Pocket-Lab, ou laboratório de bolso, que é baseado em um filtro de Sallen-Key. Esta ferramenta tem a finalidade do estudo de sistemas dinâmicos. Foi projetado para não ter a necessidade de uma fonte externa de alimentação para o circuito, sendo implementado apenas um microcontrolador Arduino e a linguagem Python, o que torna uma tecnologia mais compacta, acessível e de fácil manuseio. Foi apresentada a utilização do filtro para estudo de identificação de sistemas, onde um modelo matemático do filtro foi gerado a partir de dados reais e posteriormente é confrontado com o modelo nominal, modelado a partir da teoria de circuitos, de onde se obtiveram ótimos resultados de ajuste. Foram realizados também experimentos de projeto de controladores lineares do tipo PI e PID, visando demonstrar a funcionalidade do filtro para o desenvolvimento de habilidades de projeto de controle. A ideia do projeto é fornecer uma ferramenta de auxílio para o estudo e aprendizado dos alunos nas competências de identificação, análise de controle de sistemas lineares. Com essa tecnologia, espera-se que professores e alunos possam interagir com sistemas físicos de forma prática e intuitiva, sem a necessidade de estarem em um laboratório convencional. O que possibilita atividades tanto em aulas presenciais quanto em ambientes remotos, assim promovendo uma experiência mais imersiva e acessível no estudo de controle de sistemas.

Palavras Chave: Sallen-Key; Sistemas lineares; Arduino.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

SUMÁRIO

1 Introdução

O controle de sistemas dinâmicos é uma área de grande relevância para a engenharia elétrica, com elevado grau de abstração teórica-matemática, oferecendo desafios ao processo de ensino e aprendizagem nos estágios iniciais de formação nos cursos de graduação. Neste contexto, laboratórios experimentais tornaram-se grandes aliados de professores e alunos para a integração de conceitos teóricos com a prática, de modo a estimular a interação dos discentes com sistemas reais, ao criar situações mais palpáveis para a implementação de técnicas de engenharia (MORAIS et al., 2014).

Essa relevância em torno dos laboratórios experimentais para o ensino não é recente, pois ao longo da história, várias instituições de ensino reconheceram a necessidade de um estudo mais experimental. As precursoras desse tipo de ensino foram as escolas técnicas ocidentais no final do século XVIII. Já para o ensino nas universidades, o primeiro laboratório experimental foi para o estudo de química em 1824 e para física em 1869 (TAKAHASHI, 2011). Com essa trajetória histórica, observa-se a importância da prática experimental no aprendizado. Nos dias atuais, com o avanço de novas tecnologias que buscam ser cada vez mais compactas e acessíveis, surgem técnicas e ferramentas que permitem manter essa tradição de ensino experimental de forma inovadora. É o caso do Pocket-Lab ou Laboratório de bolso, que é um laboratório compacto e portátil, cuja portabilidade é a principal vantagem de promover um ambiente de aprendizagem sem limitações de espaço (BAYAR; KURT, 2021).

Para apresentar a importância dessa ferramenta, foi realizado um estudo comparativo com estudantes de física, utilizando duas ferramentas de estudo, os estudantes foram divididos em dois grupos, o primeiro grupo utiliza a ferramenta *Vernier Science Education Software* que é composta por vários sensores específicos, e que necessitam de outras ferramentas para fazer a montagem experimental, enquanto o outro grupo utilizou o *PocketLab Science Everywhere System* que é um dispositivo com várias funções, portátil e sem a utilização de fios, com a capacidade de medir diferentes variáveis em tempo real. O resultado do estudo foi que o grupo que utilizou o PocketLab teve mais flexibilidade, percepção e motivação maior de conexão com o mundo real. Foi observado que 93% dos participantes que utilizaram essa ferramenta reconheceram a utilidade e possibilidade de adaptar os experimentos e relacionar a prática do cotidiano, em comparação com 71% do grupo que utilizou Vernier. foi feita a comparação de aprovação, na qual o PocketLab obteve mais de 90% de aprovação, enquanto o Vernier teve 71%. Esses dados mostrados desse estudo sugerem que o PocketLab, além de proporcionar facilidade de coleta e análise de dados, também promove uma experiência de aprendizagem mais envolvente, palpável, real, autônoma e mais alinhada aos interesses dos alunos (KETSMAN; SANTANA, 2024).

Outro trabalho que trata desse tipo de dispositivo portátil para estudo, foi o artigo *"Tools and Control Experiments using TCLab Arduino Kit"* que aborda diversos kits de laboratório portáteis, utilizados para o estudo de engenharia de controle. Dentre esses kits, um em especial foi o *Temperature Control Laboratory* (TCLab) que foi escolhido para fazer um estudo mais aprofundado. O TCLab foi utilizado para projetar controladores e identificação de modelos. O TCLab é um laboratório portátil com a finalidade de reforçar a teoria de controle para os estudantes. Diversas universidades já utilizaram esse tipo de laboratório para o estudo de controle (AGUIAR, 2020). Esses dois estudos reforçam a importância da utilização de um dispositivo portátil para o ensino, como o Pocket-Lab, que é portátil, compacto, simples e acessível para o estudo e aprendizagem de controle.

Dentro deste cenário, o trabalho desenvolvido apresenta o projeto e a implementação de um Pocket-Lab, aplicado na área de controle, fundamentado no sistema dinâmico do filtro Sallen-Key. Este clássico sistema linear e invariante no tempo (LIT) é um filtro de segunda ordem implementado com dispositivos eletrônicos básicos com custo acessível, como resistores, capacitores e amplificadores operacionais. O sistema oferece a possibilidade de execução de experimentos com dados reais nas principais habilidades de engenharia de controle, como: (i) identificação, que trata da criação de modelos matemáticos a partir de dados experimentais medidos; (ii) análise, que aborda a verificação da resposta dinâmica do sistema, da estabilidade e da interpretação do comportamento dinâmico no domínio da frequência; e (iii) projeto de controladores PI e PID, que se refere à síntese de algoritmos para o controle do sistema dinâmico a partir de diferentes abordagens.

O objetivo do trabalho é o desenvolvimento de um Pocket-Lab, que permita o uso com portabilidade em sala de aula, em laboratório ou mesmo em atividades não presenciais. Esta característica demonstra-se crucial principalmente em cenários que dificultam o acesso presencial, como ocorreu durante a pandemia da COVID-19, onde, dada a suspensão das atividades presenciais, o ensino de laboratório tornou-se impraticável considerando a impossibilidade de acesso aos sistemas reais tradicionais.

1.1 Objetivos

Objetivo Geral: Implementar um Pocket-Lab para a área de controle, como um dispositivo portátil e de fácil acesso. Seu desenvolvimento foi baseado no filtro de Sallen-Key, visando auxiliar o ensino e o aprendizado de alunos no estudo de Sistemas Dinâmicos.

Objetivos Específicos:

- Implementar o Pocket-Lab como ferramenta que auxilie no estudo de sistemas dinâmicos, como: observar o comportamento do sistema, analisar os polos, resposta em frequência, estabilidade entre outros;

- Projetar o Pocket-Lab como um dispositivo compacto, com fácil acesso, baixo custo, construção simples e sem a necessidade de utilizar uma fonte de tensão externa;
- Utilizar o Pocket-Lab para estudos de identificação de modelo e projetar controladores.

1.2 Proposta do TCC:

A proposta desse trabalho é o desenvolvimento do Pocket-Lab como uma ferramenta que auxilie os professores e alunos com estudos de sistemas lineares e invariantes no tempo (LTI) sem a necessidade de uma fonte externa, ou a necessidade de estar em um laboratório. Sendo possível trabalhar de qualquer lugar e contribuir para o estudo de engenharia de controle por meio de um recurso didático, portátil, de fácil produção e com baixo custo, com o objetivo de auxiliar a compreensão de conceitos como: polos, resposta em frequência, resposta transitória, estabilidade, comportamento do sistema, projeto de controladores como: PI e PID que foram apresentados no trabalho, e outros.

Este trabalho pretende aproximar a teoria da prática, tornando o conteúdo menos abstrato e contribuindo com o ensino em sala de aula, mas também com projetos de extensão e ensino remoto. Sendo assim, ampliam-se as várias possibilidades de aplicação e aprendizado na área de controle.

1.3 Apresentação do documento

Este trabalho descreve o desenvolvimento de um projeto de monitoria, com a participação da discente em todas as etapas, desde a concepção até a implementação prática. O documento está dividido em seis capítulos, com a estrutura do trabalho apresentada da seguinte forma:

- **Capítulo 1:** Apresenta a parte introdutória, composta também com os objetivos, metodologia, proposta do TCC e apresentação do documento;
- **Capítulo 2:** Aborda a parte teórica do trabalho, abordando a modelagem por leis físicas, parametrização, polos e resposta dinâmica, resposta transitória do sistema, resposta senoidal e resposta em frequência.
- **Capítulo 3:** Aborda as Simulações e prototipagem do Pocket-Lab, composto pelos temas de implementação do filtro de Sallen-Key, componentes e placa PCB, conexões do Pocket-Lab com o Arduino, Software, dispositivos eletrônicos e o Pocket-Lab;

- **Capítulo 4:** Apresenta a parte de identificação do sistema, abordando a identificação por mínimos quadrados, identificação do modelo com dados reais e a análise no tempo do modelo identificado;
- **Capítulo 5:** Aborda o tema de controle de sistemas, o qual é composto pelo projeto de controladores PI e PID;
- **Capítulo 6:** Aborda a parte de conclusão, falando os pontos positivos, finalidades, dificuldades e futuras melhorias, além de discutir os resultados da identificação e dos controladores.

2 O filtro de Sallen Key

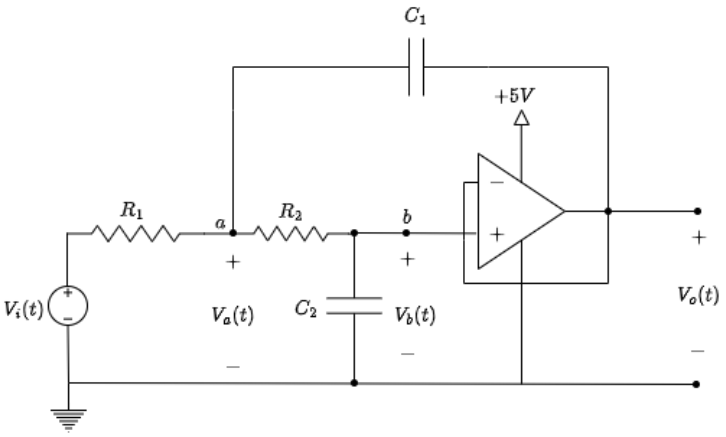
Para o processamento de sinais é importante o uso de filtros, sendo eles passivos ou ativos, pela motivação de selecionar os sinais com características necessárias e a atenuação de sinais de características indesejáveis, utilizando como referência uma faixa de frequência, de maneira que qualquer sinal que esteja fora dessa faixa será atenuado (SANDIM, 2023).

Neste projeto foi utilizado um filtro ativo da topologia Sallen-Key (Filtro-SK) também podendo ser chamado por *voltage control voltage source* (VCVS) o qual significa fonte de tensão controlada por tensão. Sua primeira aparição foi em 1955 no laboratório Lincoln do MIT (Massachusetts Institute of Technology) por R. P. Sallen e E. L. Key (JR, 2015). Esse filtro é bastante utilizado por razões de simplicidade e possui uma menor dependência no desempenho do filtro e do amplificador operacional(amp-op), pois o amp-op está configurado como amplificador e não como integrador, que reduz o ganho de largura de banda (JUNG, 2005).

Neste trabalho o Filtro-SK é utilizado como um sistema dinâmico linear e invariante no tempo (LTI), no contexto de estudo de sistemas dinâmicos e controle. Desta forma, o filtro constitui um sistema para se estudar a identificação de modelos matemáticos, a análise de comportamentos dinâmicos e o projeto de controladores lineares.

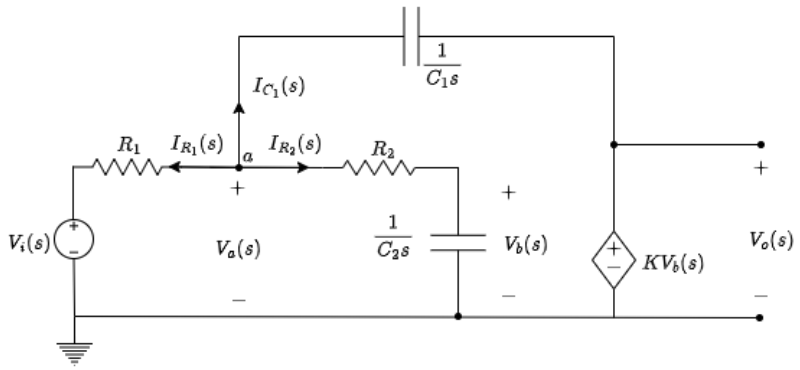
No que segue será apresentado o filtro desenvolvido no projeto, na Figura 1 mostra um circuito *sallen-key* de segunda ordem com ganho unitário. A partir desse circuito, foi elaborado um circuito equivalente, como mostra a Figura 2 para melhor determinar o modelo matemático, que pode ser representado por uma função de transferência.

Figura 1 – Circuito *sallen-key* com ganho unitário



Fonte: Elaboração própria

Figura 2 – Circuito equivalente



Fonte: Elaboração própria

2.1 Modelagem por leis físicas

O circuito Sallen-Key tem sua representação matemática a partir de uma função de transferência. Para encontrar essa função, foi analisada a Figura 2 que é o circuito equivalente e facilita na análise. Para poder determinar a função de transferência foi utilizada a proposta de modelagem baseada em (LATHI, 2006), pela qual pode-se definir a função de transferência $H(s)$ como:

$$H(s) = \frac{V_o(s)}{V_i(s)} \tag{1}$$

onde $V_o(s)$ é a tensão de saída e $V_i(s)$ a de entrada.

As tensões são substituídas por suas transformadas de Laplace, e os elementos substituídos pelas suas respectivas impedâncias. É utilizada então a análise nodal, com duas tensões desconhecidas, $V_a(s)$ e $V_b(s)$, conforme apresentados na Figura 2. Então, aplicam-se duas equações de nó.

Nó a: Serão analisadas as correntes do nó a, a primeira é a corrente no resistor R_1 chamada I_{R_1}, a equação de corrente é assim $I = \frac{V}{R}$, esse V é a tensão que sai do maior potencial para o menor, então $V = V_a(s) - V_i(s)$, então a corrente de R_1 será:

$$I_{R_1} = \frac{V_a(s) - V_i(s)}{R_1}$$

Para a corrente do R_2 tem a mesma análise, então sua corrente é igual a:

$$I_{R_2} = \frac{V_a(s) - V_i(s)}{R_2}$$

Para a corrente do capacitor, será primeiro mostrada a equação da tensão com capacitor que é $V = \frac{1}{C(s)}I$, isolando a corrente a sua equação é $I()s = V(s) \cdot Cs$, analisando as tensões que vão do maior potencial para o menor, sua tensão é $V = V_a(s) - KV_b(s)$, então sua equação é:

$$I_{c_1(s)} = [V_a(s) - KV_b(s)] \cdot C_1 s$$

Agora analisando o caminho das correntes, todos estão saindo do nó a, e segundo a lei de Kirchhoff das correntes (LKC) "A soma das correntes elétricas que fluem a partir de um nó é igual a zero"(NISE, 2012), isso significa que $I_{R_1} + I_{R_2} + I_{C_1(s)} = 0$ e substituindo as correntes por suas equações, fica:

$$\frac{V_a(s) - V_i(s)}{R_1} + \frac{V_a(s) - V_b(s)}{R_2} + [V_a(s) - KV_b(s)] \cdot C_1 s = 0$$

Em seguida, é feita a distribuição das frações, além da distribuição também sumiu o K e isso ocorre porque o ganho é unitário, então $K = 1$:

$$\frac{V_a(s)}{R_1} + \frac{V_a(s)}{R_2} - \frac{V_i(s)}{R_1} - \frac{V_b(s)}{R_2} + V_a C_1 s - V_b C_1 s = 0$$

Então aplica-se a fatoração nos termos comuns:

$$\left[\frac{1}{R_1} + \frac{1}{R_2} + C_1 s\right] V_a(s) - \left[\frac{1}{R_2} + + C_1 s\right] V_b(s) = \left[\frac{1}{R_1}\right] V_i(s)$$

Nó b: A análise nodal do nó b é similar ao nó a, onde apenas as correntes de R_2 e $C_1 s$ que entram no nó b, para a corrente de R_2, utiliza daquela mesma equação, a diferença agora é na diferença de tensão que é $V = V_b(s) - V_a(s)$, então a equação fica:

$$I_{R_2} = \frac{V_b(s) - V_a(s)}{R_2}$$

Para a corrente de $C_2 s$ a equação é parecida com a de $C_1 s$, sendo sua tensão apenas em $V_b(s)$, então fica:

$$I_{c_2(s)} = V_b(s) \cdot C_2 s$$

Aplicando novamente a LKC, a soma das correntes no nó é igual a zero, resultando em $I_{R_2} + I_{C_2(s)} = 0$, substituindo as correntes por suas equações, fica:

$$\frac{V_b(s) - V_a(s)}{R_2} + V_b(s)C_2 s = 0$$

Em seguida, é feita a distribuição das frações:

$$\frac{V_b}{R_2} + V_b(s)C_2 s - \frac{V_a}{R_2} = 0$$

Depois realiza a fatoração:

$$\frac{1}{R_2}V_a + \left[\tfrac{1}{R_2} + C_2 s\right] V_b = 0$$

Então foi utilizada a regra de Cramer, que é utilizada para resolver sistemas de equações lineares, mas antes foram temporariamente substituídos alguns termos para facilitar o uso da regra de Cramer, e os termos substituídos estão a seguir:

$$\begin{aligned} G_1 &= \frac{1}{R_1} \\ G_2 &= \frac{1}{R_2} \\ V_o(s) &= KV_b(s) = V_b(s) \\ H(s) &= \frac{V_b(s)}{V_i(s)} \end{aligned}$$

Substituindo para G_1 e G_2, e aplicando a regra de Cramer:

$$\begin{bmatrix} G_1 + G_2 + C_1 s & -(G_2 + C_1 s) \\ -G_2 & (G_2 + C_2 s) \end{bmatrix} \begin{bmatrix} V_a(s) \\ V_b(s) \end{bmatrix} = \begin{bmatrix} G_1 V_i(s) \\ 0 \end{bmatrix}$$

Realizando uma multiplicação cruzada entre os termos de cima e de baixo:

$$\Delta = (G_1 + G_2 + C_1 s)(G_2 + C_2 s) - (G_2)(G_2 + C_1 s)$$

Fazendo a distributiva e depois eliminando os termos iguais com sinais diferentes resulta em:

$$\Delta = C_1C_2s^2 + (G_1C_2 + G_2C_2)s + G_1G_2$$

Agora fazendo o determinante de $V_b(s)$, onde é feita a substituição dos valores da coluna de $V_b(s)$ da matriz pelo resultado $\begin{bmatrix} G_1V_i(s) \\ 0 \end{bmatrix}$, ficando assim:

$$\Delta V_b(s) = \begin{bmatrix} G_1 + G_2 + C_1s & G_1V_i(s) \\ -G_2 & 0 \end{bmatrix} = G_1G_2V_i(s)$$

Fazendo a divisão entre o determinante de $V_b(s)$ e o determinante principal:

$$\frac{\Delta V_b(s)}{\Delta} = \frac{G_1G_2V_i(s)}{C_1C_2s^2 + (G_1C_2 + G_2C_2)s + G_1G_2}$$

Para transformar a função de transferência na forma padrão, foi multiplicado o numerador e denominador por $\frac{1}{C_1C_2}$ para que o coeficiente de s^2 seja igual a 1.

$$\frac{V_b(s)}{V_i(s)} = \frac{G_1G_2}{C_1C_2s^2 + (G_1C_2 + G_2C_2)s + G_1G_2} \cdot \frac{\frac{1}{C_1C_2}}{\frac{1}{C_1C_2}}$$

Depois da multiplicação e também substituindo o $V_b(s)$ por $V_o(s)$:

$$\frac{V_o(s)}{V_i(s)} = \frac{\frac{G_1G_2}{C_1C_2}}{s^2 + \left[\frac{G_1C_2+G_2C_2}{C_1C_2}\right]s + \left[\frac{G_1G_2}{C_1C_2}\right]}$$

Após multiplicação, foram definidos os parâmetros convencionais de um sistema de segunda ordem padrão, ou seja, a frequência natural ω_n e o coeficiente de amortecimento ξ, os quais podem ser relacionados com os parâmetros da função de transferência desenvolvida fazendo (LATHI, 2006):

$$\begin{aligned} \omega_n^2 &= \frac{G_1G_2}{C_1C_2} \\ \omega_n^2 &= \frac{1}{C_1C_2R_1R_2} \end{aligned} \tag{2}$$

$$2\xi\omega_n = \frac{G_1C_2 + G_2C_2}{C_1C_2}$$
$$2\xi\omega_n = \frac{1}{R_1C_1} + \frac{1}{R_2C_1} \tag{3}$$

Substituindo os valores da equação por ω_n^2 e $2\xi\omega_n$, então a função de transferência de malha aberta do filtro é definida por:

$$\frac{V_o(s)}{V_i(s)} = H(s) = \frac{\omega_n^2}{s^2 + 2\xi\omega_n s + \omega_n^2} \tag{4}$$

Que expressa em função dos parâmetros do filtro resulta:

$$H(s) = \frac{\frac{1}{C_1C_2R_1R_2}}{s^2 + (\frac{1}{R_1C_1} + \frac{1}{R_2C_1})s + \frac{1}{C_1C_2R_1R_2}} \tag{5}$$

2.2 Parametrização - Escolha dos componentes

Antes de montar o circuito, foi necessário fazer simulações utilizando Python, para saber seu comportamento e proceder com a escolha do valor dos componentes eletrônicos. Então, foi utilizado um método para a escolha dos resistores e capacitores. Primeiro, utilizou-se o cálculo do tempo de acomodação t_s desejável para o circuito, que é definido, no caso de um sistema de segunda ordem, como:

$$t_s = \frac{4}{\xi\omega_n} \tag{6}$$

Buscando encontrar ω_n, foi isolado este parâmetro e realizada a definição de valor para ξ, o que resulta:

$$\omega_n = \frac{4}{\xi t_s} \tag{7}$$

A partir dos valores destes parâmetros, define-se $H(s)$ completamente.

Agora é possível descobrir os valores dos componentes, primeiro escolher os valores de capacitores e de um dos resistores, pois o outro vai ser determinado pela Equação 2 apenas fazendo o isolamento do resistor, pode ser qualquer um, por exemplo o R_2, a equação fica:

$$R_2 = \frac{1}{C_1(2\xi\omega_n - \frac{1}{R_1C_1})} \tag{8}$$

Foi escolhido o valor do fator de amortecimento como $\xi = 0,52$ e o tempo de acomodação ($t_s = 0,3\ s$). Foi determinado o valor da frequência natural com a Equação 7 como $\omega_n = 25,64\ rad/s$.

A partir desses valores pode-se escolher os componentes, alguns foram pré-definidos e outros foram calculados, o que resultou em: $R_1 = 1\ K\Omega$, $C_1 = 3,3\ \mu F$, $C_2 = 3,3\ \mu F$. Para determinar R_2, foi utilizada a Equação 8, e fazendo a substituição o valor será de $R_2 \approx 11363,63\ \Omega$. No entanto, o valor comercial adotado foi $R_2 = 10\ K\Omega$, e assim foi realizada a escolha dos valores. E com esses valores é possível montar o modelo analítico no tempo contínuo que é apresentado na Equação 9, nessa equação foram utilizados os valores reais dos componentes, sendo o $R_1 = 984\ \Omega$, $R_2 = 9,88\ K\Omega$, $C_1 = 3,303\ \mu F$ e $C_2 = 3,353\ \mu F$.

$$H(s) = \frac{9288}{s^2 + 337,9s + 9288} \tag{9}$$

Após encontrar o modelo analítico no tempo contínuo, é necessário fazer a discretização no sistema, pois será utilizado mais à frente. Então, para discretizar o sistema, foi utilizado o método de ZOH, que é multiplicar $\frac{1-e^{-sT}}{s}$ pelo $H(s)$, como na Equação 10. A equação a seguir é baseada em (NISE, 2012).

$$H(z) = \frac{1 - e^{-sT}}{s} \cdot \frac{9288}{s^2 + 337,9s + 9288} \tag{10}$$

Mandando o s para multiplicar com o denominador do $H(s)$ como mostra na Equação 11, como o $H(s)$ recebeu um s, então será definida essa nova equação de $H_2(s)$.

$$H(z) = 1 - e^{-sT} \cdot \frac{9288}{s(s^2 + 337,9s + 9288)} \tag{11}$$

Foi analisado o e^{-sT} como mostra a Equação 12:

$$e^{sT} = z \ \therefore \ e^{-sT} = z^{-1} \tag{12}$$

Foi analisado $1 - e^{-sT}$ como mostra a Equação 13:

$$1 - e^{-sT} = 1 - z^{-1} = 1 - \frac{1}{z} = \frac{z-1}{z} \tag{13}$$

Substitui o $1 - e^{-sT}$ por $\frac{z-1}{z}$ e aplica a transformada Z no $H(s)$, representado na Equação 14:

$$H(z) = \frac{z-1}{z}\mathcal{Z}\left\{\frac{9288}{s(s^2+337,9s+9288)}\right\} \tag{14}$$

Aplicando a decomposição em frações parciais na $H_2(s)$ mostrado na Equação 15.

$$H_2(s) = \frac{9288}{s(s^2+337,9s+9288)} = \frac{K_1}{s} + \frac{K_2}{(s+30,18)} + \frac{K_3}{(s+307,68)} \tag{15}$$

Primeiro foram divididas as frações em 3 partes, pois esse é o número de polos que o $H_2(s)$ tem, além disso temos que simplificar encontrando os polos, o primeiro é s que vale 0, os outros dois, correspondem aos polos de $H(s)$, que são dados por $-30,18$ e $-307,68$, então: $\frac{K_1}{s} + \frac{K_2}{(s+30,18)} + \frac{K_3}{(s+307,68)}$, primeiro vamos descobrir o valor de K_1, que é pegar a equação 15 e retirar o s do denominador, ficando apenas $(s^2+337,9s+9288)$, pois ele é o denominador de K_1, e então substituir o valor do s pelo valor do polo em K_1, como mostra a Equação 16:

$$K_1 = \frac{9288}{(s^2+337,9s+9288)}|s=0 \ = 1 \tag{16}$$

Para encontrar K_2 é preciso retirar o denominador de K_2 da Equação 15 e substituir o s pelo polo de K_2 que é $-30,18$, como mostra na Equação 17:

$$K_2 = \frac{9288}{s(s+307.68)}|s=-30,18 \ = -1,1088 \tag{17}$$

E para encontrar o K_3 é da mesma forma que foi feito para K_1 e K_2 que é retirar o denominador de K_3 da Equação 15 e substituir o s pelo polo de K_3 que é $-307,68$, como mostra na Equação 18:

$$K_3 = \frac{9288}{s(s+30,18)}|s=-307,68 \ = 0,1088 \tag{18}$$

Então substitui os valores de K_1, K_2 e K_3 como mostra na Equação 19.

$$H_2(s) = \frac{1}{s} - \frac{1,1088}{(s+30,18)} + \frac{0,1088}{(s+307,68)} \tag{19}$$

Fazendo a transformada pela Tabela 1.

Tabela 1 – Tabela de transformadas Z e Laplace

Nř	$F(s)$	$F(t)$ ou $F(k)$	$F(z)$ onde $k = 0, 1, 2...$
1	$\frac{1}{s}$	Degrau unitário - $1(t)$	$\frac{z}{z-1}$
2	$\frac{1}{s+a}$	e^{-aT}	$\frac{z}{z-e^{-aT}}$

Fonte: Baseado em Nize (2012)

Então, depois de usar a tabela, é assim que fica a equação $H_2(z)$ como mostra a Equação 20.

$$H_2(z) = \frac{z}{z-1} - \frac{1,1088z}{(z - e^{(-30,18 \cdot 0,00156)})} + \frac{0,1088z}{(z - e^{(-307,68 \cdot 0,00156)})} \tag{20}$$

Então na Equação 21 mostra a equação depois de calcular o e^{-aT}.

$$G_2(z) = \frac{z}{z-1} - \frac{1,1088z}{(z - 0,6244)} + \frac{0,1087z}{(z - 0,00823)} \tag{21}$$

Depois foi feita manipulação algébrica e substituindo o $H_2(z)$ na Equação 22:

$$H(z) = \frac{z-1}{z} \cdot H_2(z) \tag{22}$$

Após todas as manipulações matemáticas resulta na Equação 23

$$G(z) = \frac{0,3085z + 0,06394}{z^2 - 0,6327z + 0,00514} \tag{23}$$

2.3 Polos e resposta dinâmica

Os polos significam raízes do denominador da função de transferência, valores necessários para se definir o perfil da resposta dinâmica e analisar a estabilidade do sistema (LATHI, 2006). Como verificado, $H(s)$ possui um par de polos reais dados por:

$$s_1 = -303,7$$
$$s_2 = -30,18$$

Com base nesses resultados, pode-se analisar detalhadamente o comportamento do sistema dinâmico.

Em sistemas de controle, a parte real dos polos é fundamental para a definição da estabilidade. Se a parte real de qualquer polo for positiva, isso indica que a resposta do sistema crescerá exponencialmente com o tempo e isso significa a instabilidade do sistema. No entanto, quando os polos são negativos, a resposta tende a se dissipar ao longo do tempo, convergindo para o ponto de equilíbrio, e isso define a estabilidade do sistema (NISE, 2012).

No caso em questão, os polos possuem partes negativas, declarando sua estabilidade independentemente de uma excitação inicial. O sistema retornará, após um tempo, ao seu estado de equilíbrio, de modo que o Filtro-SK é um sistema estável.

Outra característica importante na análise dos polos é a sua posição no plano complexo. A distância dos polos em comparação com a origem tem impacto direto na dinâmica do sistema, pois quando os polos estão distantes em relação ao eixo imaginário, isso indica que a resposta do sistema é mais rápida, já que a parte real mais negativa se torna uma exponencial de decaimento mais acentuada (NISE, 2012).

Como pode ser observado, os polos apresentados estão bem longe da origem, o que significa que o sistema tem uma resposta rápida, com o decaimento exponencial das respostas, acontecendo em um curto período de tempo, o que é típico de circuitos eletrônicos.

Foram apresentados os polos no tempo contínuo, mas serão apresentados também os polos do modelo analítico no tempo discreto que são: $0,6244$ e $0,0082$. Para encontrar os polos no tempo discreto foi da mesma forma que os polos no tempo contínuo. Para um sistema no tempo discreto, os polos devem estar dentro do círculo unitário ($|z| < 1$) para serem estáveis; então, para fazer essa verificação, precisa apenas fazer o módulo dos polos menor que 1, o que é o caso (NISE, 2012).

Sendo assim, a análise dos polos é importante para confirmar a estabilidade e a velocidade do sistema.

2.4 Resposta transitória do sistema

No estudo de sistemas de controle, o tempo de acomodação é um dos aspectos importantes que podem ser analisados para conhecer o desempenho de um sistema. Ao aplicar um degrau na entrada, é possível notar como o sistema responde, observando características como: tempo de subida, *overshoot* e, em especial, o tempo de acomodação, que é o aspecto que será analisado.

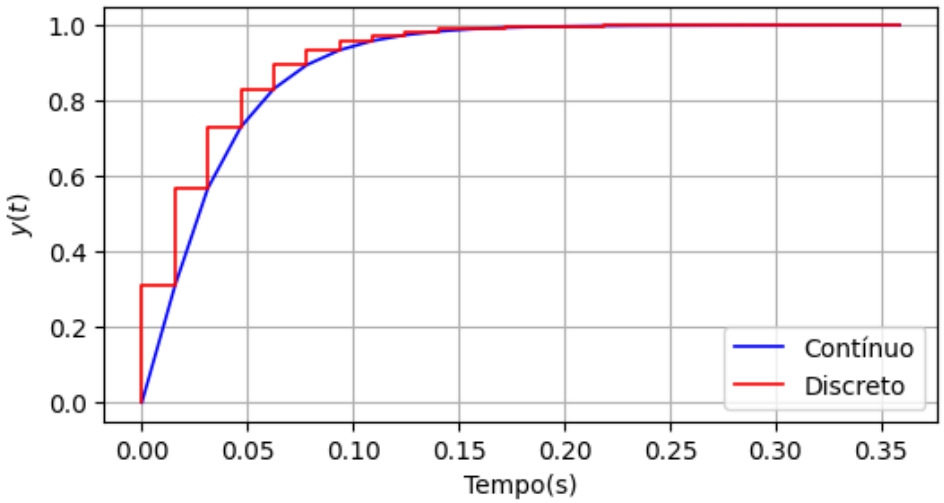
O tempo de acomodação significa "O tempo necessário para que as oscilações amortecidas transitórias alcancem e permaneçam dentro de uma faixa de ś2% em torno do valor em regime permanente"(NISE, 2012).

Na figura 3 mostra a resposta ao degrau do sistema contínuo, representado pela linha azul, e do sistema discreto, representado pela linha vermelha. É possível notar a excelente correspondência entre as curvas, o que significa que a discretização está adequada para representar o sistema. Ademais, as duas curvas convergem para 1, evidenciando que o sistema tem ganho unitário. Aliás, na figura pode-se notar que a resposta do sistema tem uma curva que se estabiliza rapidamente, sem oscilações perceptíveis. Além disso, ainda observando o gráfico, parece que o sistema atinge o regime permanente em torno

de $0,10$ s a $0,15$ s, o que indica que seu tempo de acomodamento é curto. Utilizando a função `ct.step_info` é possível visualizar o valor do tempo de acomodamento que é $0,134$ s. Essa resposta caracteriza que o sistema é bem amortecido e estável.

A resposta ao degrau apresentada demonstra que o sistema projetado tem rapidez na transição para o estado estacionário e que atende aos requisitos de estabilidade, o que mostra ser positivo de acordo com características a serem desejadas em sistemas de controle eficientes.

Figura 3 – Gráfico da resposta ao degrau do modelo analítico no tempo contínuo e discreto



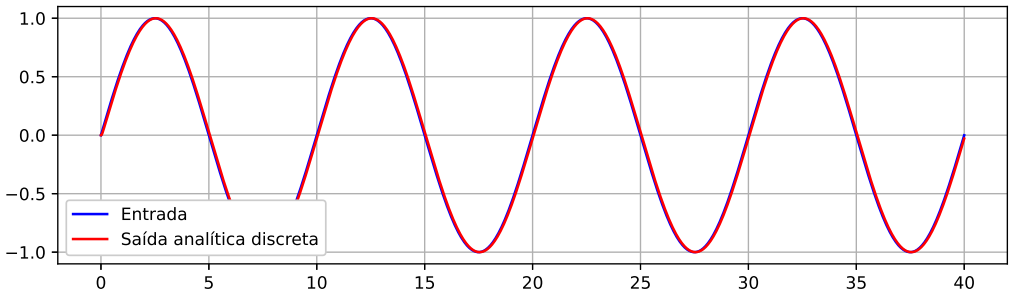
Fonte: Elaboração própria

Este foi a resposta ao degrau no tempo contínuo, no tempo discreto os gráficos são bem semelhantes como é apresentado na Figura 3, mas o tempo de acomodação é um pouco diferente com o valor de $0,1404$ s.

2.5 Resposta senoidal

Foi aplicada uma senoide na entrada do sistema analítico discreto como mostra na Figura 4, com uma frequência de $0,1$ Hz com a amplitude entre -1 e 1. A finalidade de aplicar essa senoide é para uma análise mais simplificada, e o primeiro a ser analisado é o fato de a entrada e saída estarem tão bem alinhadas, e isso remete que a entrada está dentro da região da passagem do filtro (LATHI, 2006). No gráfico, dá para ver que o sinal de saída não tem defasamento nem atenuação, pois a frequência do sinal não atingiu ainda a faixa onde acontece essa defasagem.

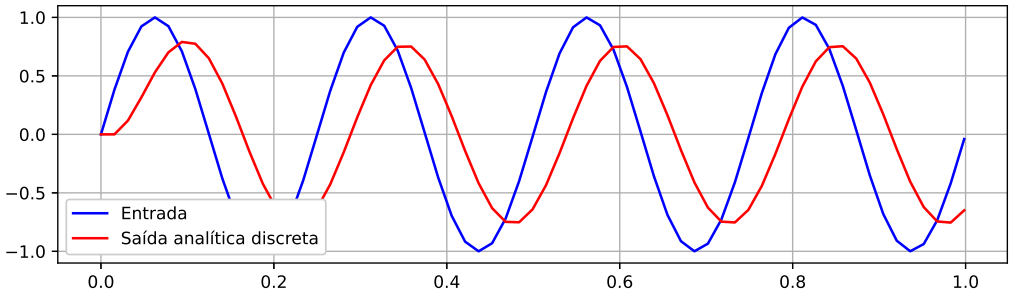
Figura 4 – Gráfico da senoide do modelo analítico no tempo discreto



Fonte: Elaboração própria

Na figura 5 mostra como fica a saída quando se aplica na entrada uma frequência acima da frequência de corte, percebe-se uma atenuação e defasamento, isso significa que como está acima da frequência de corte o sinal atenua cada vez que aumenta a frequência, pois o filtro é passa-baixa, a frequência aplicada na entrada foi de 3 Hz.

Figura 5 – Gráfico da senoide do modelo analítico no tempo discreto com a frequência acima da frequência de corte



Fonte: Elaboração própria

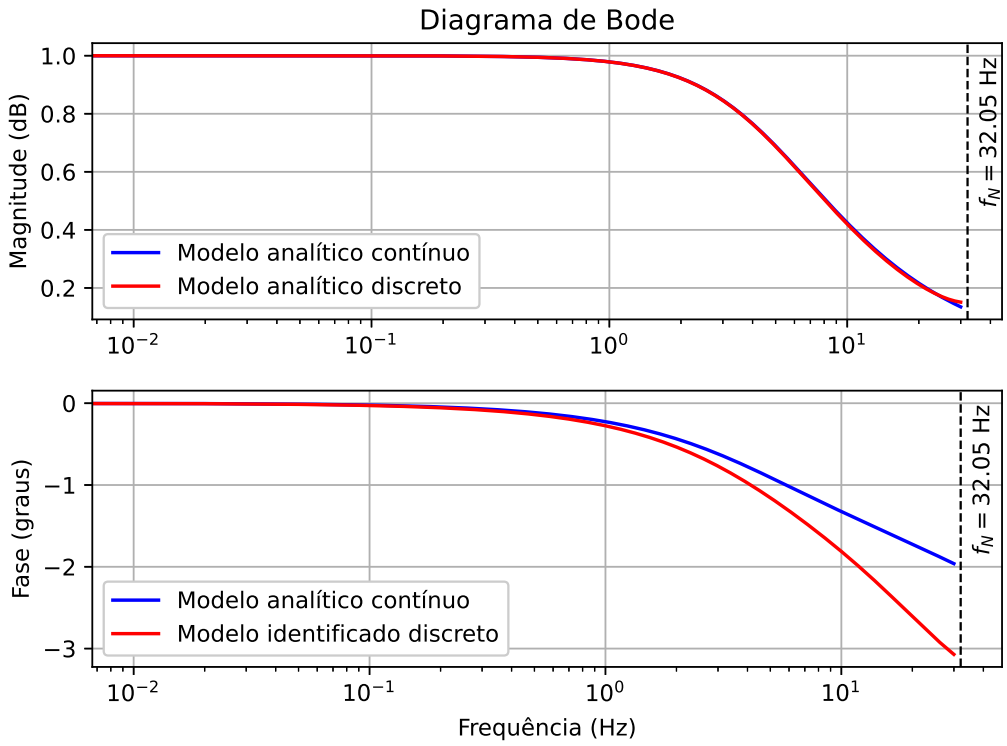
2.6 Resposta em frequência: frequência de corte - Bode

O diagrama de Bode é constituído por dois gráficos: um da amplitude em (dB) e o outro da fase em graus, ambos em função da escala logarítmica. Uma de suas vantagens é converter a multiplicação dos módulos em soma, além de permitir esboços aproximados utilizando assíntotas, o que facilita a análise da resposta em frequência (OGATA, 2020).

A Figura 6 mostra o modelo analítico contínuo e discreto no mesmo diagrama, no eixo horizontal dessa figura representa a frequência em Hertz na escala logarítmica, e no eixo vertical da figura representa o ganho do sistema em diferentes frequências. À medida que a frequência aumenta, o ganho começa a cair, o que indica que o sinal está atenuando. Em relação à frequência de corte, é o ponto onde o ganho cai 3 dB em relação ao ganho máximo. Analisando a figura, quando a frequência chega em $10^0\,Hz$ ou $1\,Hz$, o ganho começa a cair. Isso significa que a frequência de corte está nessa região próxima de $1\,Hz$, mostrando que o sistema se comporta como um filtro passa-baixa.

Nessa mesma figura, a parte do modelo analítico discreto aparece uma linha tracejada na vertical que se chama frequência de Nyquist, que significa o limite superior da frequência que pode ser representada sem *aliasing*. O *aliasing* é um problema crítico em sistemas de amostragem que pode levar à distorção e perda de informação (DANTAS et al., 2015). Observando no gráfico, parece que a frequência de Nyquist está aproximadamente entre $30\,Hz$ e $35\,Hz$ e na Figura 6 mostra o valor exato que é $32,05\,Hz$. Para encontrar o valor da frequência de Nyquist foi utilizada algumas equações, e a mesma foi baseada em (DANTAS et al., 2015). Analisando as duas curvas, é perceptível uma certa semelhança com a magnitude do diagrama de Bode do sistema em tempo contínuo e discreto, a frequência de corte também está próxima de $1\,Hz$.

Figura 6 – Diagrama de Bode do modelo analítico no tempo contínuo e discreto



Fonte: Elaboração própria

Para encontrar a frequência de Nyquist (ω_N), primeiro tem que ser encontrada a frequência de amostragem (ω_{Ts}), que segundo o Shannon-Nyquist, o valor de ω_{Ts} deve ser maior ou igual ao dobro do valor de ω_N (DANTAS et al., 2015).

$$\omega_{Ts} \geq 2 \cdot \omega_N \tag{24}$$

O ω_{Ts} pode ser representado da seguinte forma:

$$\omega_{Ts} = \frac{2\pi}{T_s} \tag{25}$$

O valor do período de amostragem é $0,0156\ s$, então, substituindo na Equação 25:

$$\omega_{Ts} = \frac{2\pi}{0,0156} = 402,77\ rad/s \tag{26}$$

O valor de ω_{Ts} já foi obtido, então falta a equação da frequência de Nyquist, que é representada pela Equação 27:

$$\omega_N = \frac{\omega_{Ts}}{2} \tag{27}$$

Fazendo a substituição dos valores, a frequência de Nyquist fica igual a $\omega_N = 201,385\ rad/s$. No entanto, no trabalho é utilizada a frequência em Hertz e não em radiano. Para converter para Hertz, precisa apenas dividir essa frequência por 2π. Além disso, trocar o parâmetro ω_N por f_N para simbolizar que agora a frequência é em Hertz, então depois da divisão o valor é apresentado na Equação 28:

$$f_N = \frac{201,385}{2\pi} = 32,05\ Hz \tag{28}$$

2.7 Metodologia

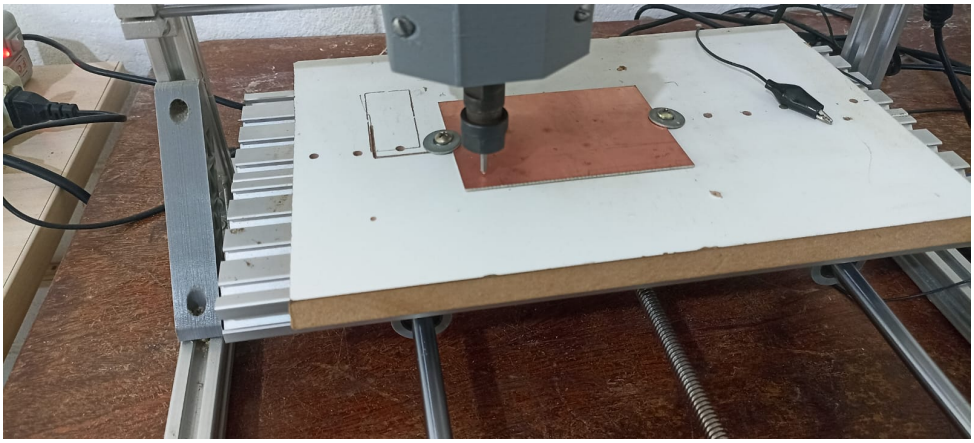
O projeto foi desenvolvido no laboratório de inteligência, controle e eletrônica (LINCE) e teve uma duração total de dez meses para a produção do filtro Sallen-Key. Neste período o desenvolvimento do projeto envolveu as seguintes atividades: (a) levantamento e estudo de referências bibliográficas, (b) análise matemática do filtro, (c) simulações computacionais, (d) construção e teste de protótipo, (f) identificação e controle do filtro, (g) aplicação funcional do produto final, conforme apresentado na Tabela 2.

Tabela 2 – Cronograma do Projeto

Cronograma do Projeto - 2023	
Atividade	**Período**
Levantamento e estudo de referências bibliográficas	Março e Abril
Análise matemática do filtro	Abril
Simulações computacionais	Maio e Junho
Construção e teste de protótipo	Julho e Setembro
Identificação e controle do filtro	Outubro e Novembro
Aplicação funcional do produto final	Dezembro

Fonte: Elaboração própria

Figura 7 – Fresadora CNC do laboratório FabLab



Fonte: Elaboração própria

Para a construção do Pocket-Lab, foram utilizados os seguintes instrumentos e componentes: três resistores, três capacitores, um amplificador operacional, uma placa fenolite, três bornes conector KRE e quatro fios. A impressão da placa de circuito impresso (PCB), foi realizada utilizando uma fresadora CNC disponível no laboratório FabLab como mostra a Figura 7. A abordagem adotada no desenvolvimento baseou-se na análise do filtro de Sallen-Key através de sua modelagem matemática embasada na teoria de circuitos elétricos, o que conduziu a um modelo por função de transferência, referido como modelo nominal (LATHI, 2006). Em seguida foram realizadas simulações computacionais utilizando o Python. Permitindo a seleção dos valores dos componentes eletrônicos e a avaliação do comportamento do sistema por meio da análise dos polos e de gráficos de resposta. Após as simulações, os componentes foram montados em uma protoboard para realizar os testes. Após todos os testes concluídos, foi projetado o layout da placa PCB a partir do site EasyEDA. Com o layout pronto, a impressão na placa foi realizada por meio da fresadora CNC, seguido pela soldagem dos componentes na placa fenolite para a finalização do circuito. O orçamento da confecção desse Pocket-Lab é apresentado na Tabela 3.

Tabela 3 – Orçamento dos materiais

Orçamento		
Quantidade	**Componente**	**Valor (R$)**
3	Resistores	2,40
3	Capacitores	3,60
1	Amp-op	3,20
3	Bornes conector KRE	3,30
4	Fios	1,80
1	Placa fenolite	20,99
1	Arduino	49,90
Valor total: R$ 85,19		

Fonte: Elaboração própria

Em conjunto com o filtro de Sallen-Key, é utilizado uma placa Arduino, de modo a servir como interface de hardware entre o sistema dinâmico e o computador pessoal. Isto permitiu a utilização da modulação por largura de pulso do sinal de entrada e a conversão analógica-digital do sinal de saída do sistema. Esta geração e aquisição dos sinais de entrada e saída são fundamentais para a realização de identificação de sistemas e para projetar e implementar controladores. A escolha da linguagem Python, como interface de software, livre e gratuita, mostrou-se favorável por apresentar sintaxe simples e legível, versatilidade e uma ampla gama de bibliotecas disponíveis. Estas garantiram a geração e aquisição de sinais, bem como a implementação de algoritmos de análise e controle, de maneira simples e bem fundamentada nas técnicas teóricas. Uma preocupação importante do projeto foi a portabilidade do protótipo. Deste modo, buscou-se realizá-lo fisicamente em uma placa de pequena dimensão, integrando os componentes fundamentais do circuito e utilizando como alimentação a energização da placa Arduino. Isto mostrou-se importante por prescindir de uma fonte DC externa e também uma estratégia segura do ponto de vista energético, tendo em conta que o filtro é um circuito de baixa potência. A interface desenvolvida em Python foi pensada para a geração e aquisição de sinais necessários à identificação e à implementação dos algoritmos de controle. A escolha da linguagem também permite ao projeto a geração de roteiros em arquivos jupyter, que incluem tanto os códigos computacionais quanto texto, figuras e formulação matemática necessárias ao desenvolvimento das atividades pelos usuários.

3 O protótipo Filtro Sallen-Key

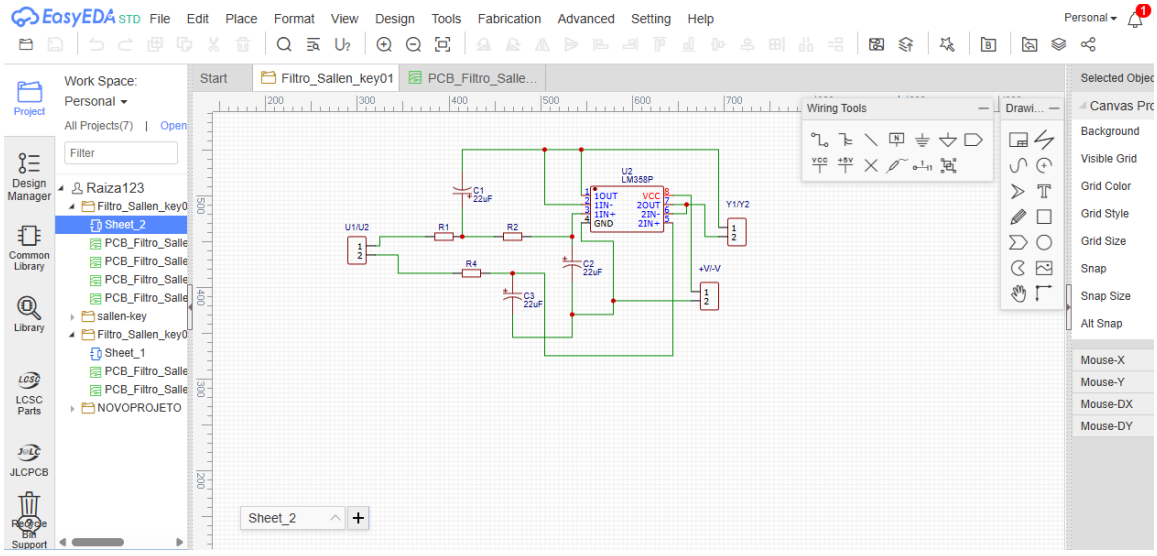
Este capítulo apresenta a prototipagem e a implementação física do Filtro-SK, a partir do projeto desenvolvido no capítulo anterior. A construção segue a ideia de conceber uma ferramenta para estudo de sistemas dinâmicos e controle.

3.1 Componentes e placa PCB

Os componentes utilizados para a montagem do Pocket-Lab foram definidos no segundo capítulo, que são: dois resistores $R_1 = 1\ k\Omega$ e $R_2 = 10\ k\Omega$, dois capacitores $C_1 = 3,3\ \mu F$ e $C_2 = 3,3\ \mu F$, e um amplificador operacional do modelo LM358P, esse modelo é do tipo não *rail-to-rail* (não alcança toda a faixa dos trilhos de alimentação), ou seja, ele não consegue trabalhar com toda a faixa de tensão que é aplicada na sua alimentação, ele acaba reduzindo essa tensão, por exemplo, uma alimentação de $5\ V$ na saída o valor pode chegar a até $3,5\ V$ (INSTRUMENTS, 1976). A escolha dos valores foi realizada a partir de simulações feitas em Python.

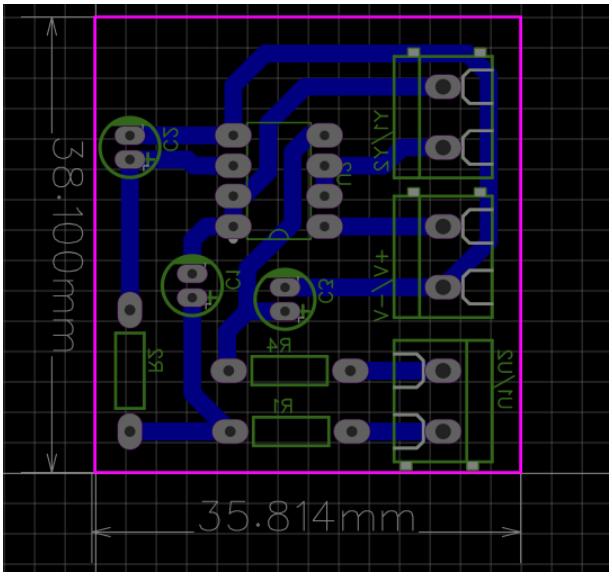
Após a definição desses componentes, o próximo passo foi projetar o modelo da placa de circuito impresso (PCB, do inglês "Printed Circuit Board"). Para projetar o modelo, foi utilizado um site de design de PCB chamado EasyEDA, apresentado na figura 8, que disponibiliza ferramentas de criação e simulação de circuitos. Na Figura 9 é apresentado o modelo em 2D da placa que será impressa, já na Figura 10 é apresentado o modelo 3D do projeto, que promove uma melhor visualização dos componentes distribuídos pela placa.

Figura 8 – Circuito do Pocket-Lab no site EasyEDA



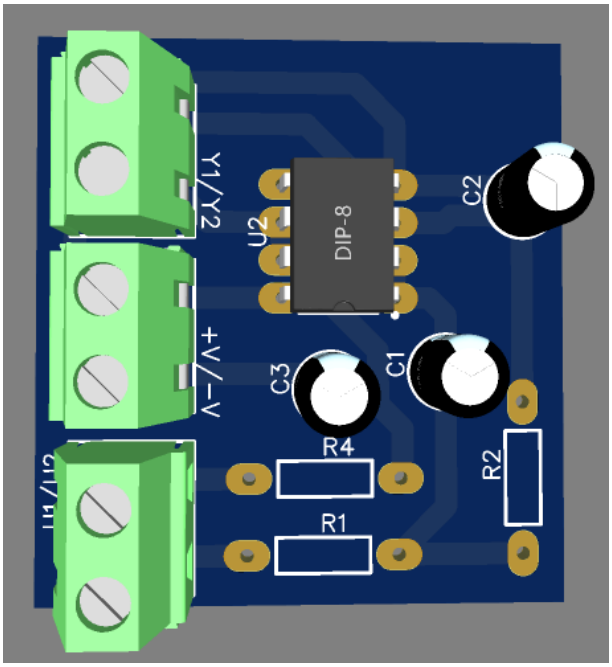
Fonte: Elaboração própria

Figura 9 – Modelo da placa PCB feito no EasyEDA



Fonte: Elaboração própria

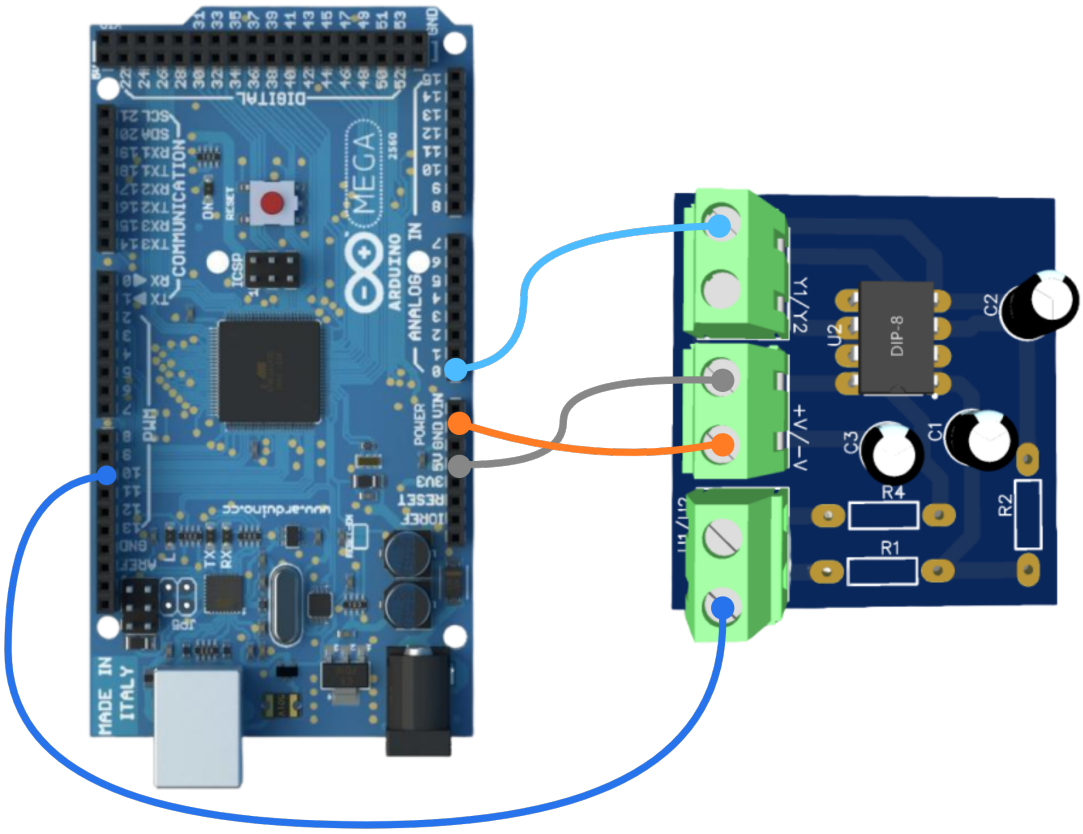
Figura 10 – Modelo da Placa PCB em 3D



Fonte: Elaboração própria

3.2 Conexões do Pocket-Lab com o Arduino

Figura 11 – Conexões do Pocket-Lab com o Arduino



Fonte: Elaboração própria

As conexões entre o Pocket-Lab e o Arduino são bem simples, são necessárias apenas quatro portas, duas são para a alimentação, e as outras duas são para entrada e saída:

- ligação da alimentação (+V) no pino +5;
- ligação da alimentação (-V) no pino GND;
- ligação da entrada (U1) no pino PWM 10;
- ligação da saída (Y1) no pino analógico 0;

3.3 Softwares - Interface Python e Arduino

3.4 O que é o Arduino

O Arduino é um microcontrolador que disponibiliza a criação e os testes em projetos de forma simples. Em sua composição, há uma placa de microcontrolador juntamente com um sistema de programação simples, que é o IDE. O Arduino é muito utilizado em

projetos de eletrônica, controle e estudos acadêmicos por ser versátil e ter uma programação intuitiva.

3.4.1

ectionComo o Arduino foi utilizado no projeto

Para a realização deste projeto, uma das principais ferramentas utilizadas foram os softwares Arduino IDE e o Visual Studio Code (VScode) que comporta a linguagem Python, ambos selecionados por suas funcionalidades complementares e específicas. O Arduino IDE é encarregado de compilar e carregar o código para a placa do Arduino. No código abaixo, é possível visualizar o código desenvolvido para o Arduino, que inclui a configuração das portas de entrada e saída e o comando de comunicação serial, que é responsável por estabelecer a conexão entre o Arduino e o computador.

```
int pinoSaida = 10;
int pinoEntrada = A0;
float valorPWM = 0;
float tensaoSaida = 0;

void setup() {
  pinMode(pinoSaida, OUTPUT);
  pinMode(pinoEntrada, INPUT);
  Serial.begin(115200);
  Serial.setTimeout(5);
}

void loop() {
  if (Serial.available() > 0) {
    valorPWM = Serial.parseFloat();
    analogWrite(pinoSaida,valorPWM);

    tensaoSaida = analogRead(pinoEntrada);
    Serial.println(tensaoSaida*5./1023.);
  }
}
```

Listing 1 – Código no Arduino

Observando melhor o Código 1 pode ser feita uma análise um pouco mais detalhada. No bloco de `setup()` foram configurados os pinos do Arduino e inicializada a conexão serial com uma taxa de 115200 bps, o qual disponibiliza uma transmissão de dados entre o Arduino e o computador de forma mais rápida e eficiente.

No bloco `loop()`, a função `Serial.available()` averigua se há dados recebidos na porta serial. Caso haja, então a função `Serial.parseFloat()` interpreta o valor numérico

enviado via serial e o armazena em `valorPWM`, que é utilizado logo para controlar a saída PWM (`Pulse Width Modulation` que significa Modulação por Largura de Pulso) pelo comando `analogWrite(pinoSaida, valorPWM)`.

Logo após, o pino analógico `A0` faz a leitura da tensão, onde o valor é armazenado `tensaoSaida`. O valor lido foi convertido em uma tensão real e, sendo multiplicado por $\frac{5}{1023}$, o valor 5 significa a tensão de referência ($5V$) e o 1023 representa a resolução ADC (conversor analógico-digital, 10 bits), e, no final, manda para a porta `Serial.println()`.

Essa implementação possibilita que o software em Python trabalhe junto com o Arduino para o compartilhamento dos dados que são enviados pelo Arduino. Tornando-o assim possível uma interface eficiente entre os dois ambientes.

3.5 O que é Python

Python é uma linguagem de programação simples e muito popular justamente por sua versatilidade e facilidade. Ela pode ser utilizada em várias áreas, como análise de dados, inteligência artificial, desenvolvimento de jogos, automação e outras. Uma das vantagens dessa linguagem é a variedade de bibliotecas disponíveis que ajudam a resolver problemas complexos com pouco esforço.

3.5.1 Como foi utilizado o Python no projeto

A integração entre o Arduino e o Python foi um passo crucial para a coleta de dados no sistema proposto. Neste projeto, a conexão entre esses dois ambientes foi feita a partir da biblioteca `serial`, a qual possibilita a troca de dados por meio da porta `COM`. Para que ocorra o envio de comando e recebimento de dados em tempo real, foi estabelecida uma interface robusta a partir dessa comunicação.

O Python foi responsável pela geração de sinais, no caso o sinal PRBS que significa sequência binária pseudoaleatória (PRBS do inglês *Pseudo-Random Binary Sequence*), utilizado para excitar o sistema em experimentos de identificação. No Código 2 mostra as bibliotecas importadas, como `matplotlib`, `scipy` e `numpy`, e outras essenciais para a visualização de dados, otimização e cálculos matemáticos. Para garantir a precisão na coleta de dados e controle do processo, foram definidos parâmetros como o `setpoint` e período de amostragem (Ts) que, apesar de seu valor ser $0,0156\ s$ e no código estar como $0,0115\ s$, esse valor do código foi utilizado apenas para ajuste, devido ao atraso causado pelo Arduino e por conta da conexão serial, então foi escolhido um número menor para que, no final, se obtivesse o valor real do `Ts`. O sinal PRBS foi gerado por meio da biblioteca `scipy.signal`, como mostra o código.

```
from scipy import signal as sg
import matplotlib.pyplot as plt
import numpy as np
import scipy.optimize
import control as ct
import serial
import time as tempo
from scipy.signal import square


Ts = 11.5e-3      #período de amostragem
setpoint = 2.5


##########_Geração do sinal PRBS_###############
U = 2*(sg.max_len_seq(8)[0]-0.5)
Ns = 3
V = np.ones(Ns)
u = np.kron(U,V) + setpoint                    #u = Sinal de Entrada
NA = len(u)                                    #NA = Número de Amostras
y = np.zeros(NA).astype(float)                 #y = Sinal de Saída
toc = np.zeros(NA)
u = u[:NA]
tempo_medido = np.zeros(len(y)).astype(float)
```

Listing 2 – Código Python geração de sinal

No código 3 mostra a parte da comunicação entre o Python e o Arduino. Após a configuração da porta serial `COM7`, o sinal PRBS gerado foi enviado ao Arduino para excitar o sistema e, a cada interação, o Arduino retornava os dados da resposta do sistema, que são lidos pela função da biblioteca serial chamada `readline()`. Para análises futuras, esses dados foram armazenados em vetores.

```
####################-COLETA E ENVIO DE DADOS-########################
print('\estabelecendo conexão.')
conexao = serial.Serial(port='COM7',baudrate=115200, timeout=0.005)

tempo.sleep(1)
print('\n iniciando coleta.')
# ESTABELECENDO UMA CONEXÃO DO ARDUINO COM O PYTHON:
for n in range(NA):
    tic = tempo.time()
    sinal_pwm = (u[n]*255)/5
    conexao.write(str(round(sinal_pwm)).encode())

    if (conexao.inWaiting()>0):
        y[n] = conexao.readline().decode()

    tempo.sleep(Ts)
    tempo_medido[n] = tempo.time()-tic


conexao.write('0'.encode())
print('\nFim da coleta.')
conexao.close()
```

Listing 3 – Coleta e envio de dados

Nessa última etapa, no Código 4 está a plotagem dos gráficos de sinal de entrada e saída por meio da biblioteca `matplotlib`, e o armazenamento de dados foi feito em um arquivo `.npy` para análises posteriores.

```
####################-PLOTAGEM DOS GRÁFICOS-#########################
t = Ts*np.arange(0,len(u))
plt.figure(figsize=(10,10))
plt.subplot(211)
plt.plot(t,u,'-b',linewidth=1.2)
plt.xlabel('Tempo(s)')
plt.ylabel('Sinal de Entrada')
plt.grid()
plt.legend(loc='lower right', labels=('Tempo','Sinal de entrada'))

plt.subplot(212)
plt.plot(t,y,'-r',linewidth=1.2)
plt.xlabel('Tempo(s)')
plt.ylabel('sinal de saida')
plt.grid()
plt.legend(loc='lower right', labels=('Tempo','Sinal de saida'))
plt.savefig('coleta_dados_2.png', bbox_inches='tight')
plt.show()

####################_SALVANDO OS DADOS_#################################
Dados = np.stack((tempo_medido,u,y), axis=-1)


np.save(r"Dados_SK_02.npy",Dados)
```

Listing 4 – Plotagem do gráfico e salvando dados

3.6 Dispositivos eletrônicos e o Pocket-Lab

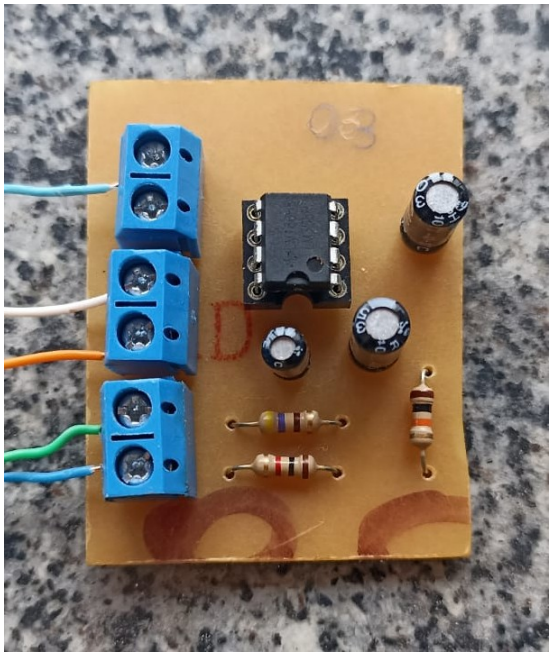
Os dispositivos eletrônicos utilizados no filtro Sallen-Key foram: dois resistores, dois capacitores, um amplificador operacional. Como apresenta a Tabela 4.

Tabela 4 – Dispositivos eletrônicos utilizados no Filtro-SK

Dispositivos eletrônicos		
Componente	Valor nominal	Valor real
Resistor R_1	1 $K\Omega$	984 Ω
Resistor R_2	10 $K\Omega$	9,88 $K\Omega$
Capacitor C_1	3,3 μF	3,303 μF
Capacitor C_2	3,3 μF	3,353 μF
Amp-op LM358P	-	-

Fonte: Elaboração própria

Figura 12 – Pocket-Lab



Fonte: Elaboração própria

A Figura 12 mostra o protótipo do Pocket-Lab que até então estava sendo representado pelo circuito esquemático na Figura 1 e pelos modelos em 2D e 3D desenvolvidos no EasyEDA, ilustrados nas Figuras 9 e 10. A montagem da placa real proporcionou validar na prática o projeto teórico e simulado. Além disso, melhorou a aquisição de dados, reduzindo ruídos, ficou mais estável e organizado, pois o protótipo montado na protoboard não apresentava uma conexão tão adequada, por conta do uso de jumpers, tornando-o mais suscetível a interferências. Assim, a versão final apresentou-se como mais adequada para aplicações experimentais e didáticas.

4 Identificação de sistemas

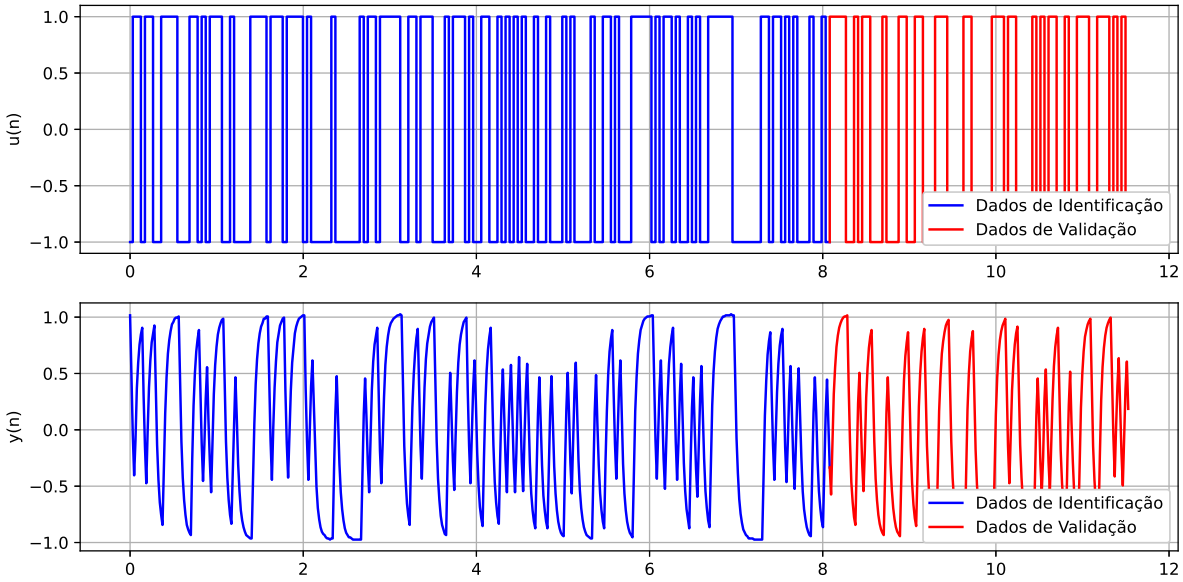
Este capítulo aborda a identificação de sistemas, demonstrada de maneira experimental utilizando o Pocket-Lab, sendo o modelo obtido por meio do método dos mínimos quadrados. No entanto, é importante salientar que existem outros métodos que podem ser utilizados para estudo.

A identificação de sistemas permite obter representações dinâmicas de sistemas lineares e não lineares, mesmo sem conhecimento prévio de suas leis físicas. Esse processo ocorre por meio da aplicação de métodos utilizando os dados experimentais obtidos (TEIXEIRA; SILVA; PARÁ,). Para realizar a identificação, é necessário, inicialmente, a coleta e pré-processamento dos dados. Esse processo começa com a coleta de dados, onde foi aplicado na entrada um sinal PRBS . Depois, tem a extração dos sinais de entrada e saída da matriz de dados; em seguida é realizado o ajuste dos sinais, removendo o *offset*, e posteriormente é definido um valor para o descarte de amostras iniciais no intuito de remover o transiente.

Para a realização da identificação, foi utilizada a linguagem Python, onde ocorreu a coleta, pré-processamento, a validação e a identificação pelo método dos mínimos quadrados.

Após o pré-processamento, é feita a divisão dos dados em dois conjuntos, um para a identificação que usa 70% dos dados, e para a validação usa 30% dos dados, que auxilia nos testes de precisão do modelo. E depois é gerado um gráfico com os sinais de entrada e de saída e mostra a divisão entre os dados de identificação e validação, mostrado na Figura 13.

Figura 13 – Sinal de entrada e de saída, destacando os dados de validação e identificação



Fonte: Elaboração própria

4.1 Identificação por mínimos quadrados

Essa identificação foi realizada por meio do modelo ARX usando o método dos mínimos quadrados (MQ), como as informações são em tempo discreto, então foi utilizado o ARX, pois ele é um modelo representado no tempo discreto, sua sigla significa Auto-regressivo com entradas externas (ARX do inglês *Autoregressive with exogenous inputs*) (AGUIRRE, 2007). Enquanto, o método MQ foi utilizado para descobrir os parâmetros do modelo. O modelo de ARX e o método MQ realizados a seguir são baseados em (AGUIRRE, 2007) e (SOUZA et al., 2023).

Para conseguir fazer a identificação, são necessários ter pelo menos dois elementos dessa Equação 29, quando foi realizada a coleta, foram recebidos os dados de entrada $\boldsymbol{u}$ e saída $\boldsymbol{y}$, esses dados são necessários para montar a matriz regressão M:

$$\boldsymbol{y} = M \cdot \boldsymbol{\theta} + \boldsymbol{\xi} \tag{29}$$

Como pode ser observado, já se dispõe da saída $\boldsymbol{y}$ e dos dados necessários para montar a matriz. Assim, os parâmetros do vetor $\boldsymbol{\theta}$ devem ser determinados para obter os valores dos coeficientes do modelo. Para encontrar esse vetor, o vetor deve ser isolado na equação futuramente, então considera-se nulo o erro $\boldsymbol{\xi}$.

Expandindo a Equação 29, resulta na Equação 30

$$\begin{bmatrix} y(n) \\ \vdots \\ \vdots \\ y(N) \end{bmatrix} = \begin{bmatrix} y(n-1) & ... & y(n-n_a) & u(n-1) & ... & u(n-n_b) \\ \vdots & & \vdots & & & \vdots \\ y(N-1) & & y(N-2) & u(N-1) & & u(N-2) \end{bmatrix} \begin{bmatrix} \theta_1 \\ \vdots \\ \vdots \\ \theta_{n_a+n_b} \end{bmatrix} \tag{30}$$

Os valores de n_a e n_b são iguais a 2, então a matriz tem N linhas e $n_a + n_b = 4$ colunas. Substituindo os valores de atraso, a Equação 31 resulta em:

$$M = \begin{bmatrix} y(n-1) & y(n-2) & u(n-1) & u(n-2) \\ \vdots & \vdots & \vdots & \vdots \\ y(N-1) & y(N-2) & u(N-1) & u(N-2) \end{bmatrix} \tag{31}$$

Agora, para encontrar os parâmetros do vetor $\boldsymbol{\theta}$, ou seja, os coeficientes do modelo identificado, será aplicado o método dos mínimos quadrados apresentado na Equação 32:

$$\boldsymbol{\theta} = (M^T M)^{-1} M^T \boldsymbol{y} \tag{32}$$

Onde:
θ: Vetor de parâmetros a serem estimados;
M: Matriz de regressão;
Y: É o vetor de saída;
M^T: Transposta da matriz de regressão;

Para montar o modelo identificado é utilizado o modelo geral, como apresenta a Equação 33 para a estrutura do ARX que será utilizada:

$$A(q)y(n) = \frac{B(q)}{F(q)}u(n) + \frac{C(q)}{D(q)}v(n) \tag{33}$$

Os valores que não são $A(q)y(n)$ e $B(q)u(n)$, serão iguais a 1. Já que só foram informados a entrada $u(n)$ e a saída $y(n)$, então o valor do ruído $v(k)$ será nulo e resulta na Equação 34:

$$A(q)y(n) = B(q)u(n) \tag{34}$$

Como a função de transferência é a saída, pela entrada, então na Equação 35 vai ser isolado $\frac{y(n)}{u(n)}$:

$$\frac{y(n)}{u(n)} = \frac{B(q)}{A(q)} \tag{35}$$

Apresentando os polinômios $A(q)$ e $B(q)$, onde $y(n)q^{-1} = y(k-1)$ e aplicando a transformada $\mathcal{Z}$, obtém-se $z^{-1}Y(z)$:

$$A(q) = 1 - a_1 q^{-1} - a_2 q^{-2} \tag{36}$$

$$B(q) = b_1 q^{-1} + b_2 q^{-2} \tag{37}$$

Então substitui os valores 37 na Equação 35 e depois aplica a transformada $\mathcal{Z}$, resultando em:

$$\frac{Y(z)}{U(z)} = \frac{b_1 z^{-1} + b_2 z^{-2}}{1 - a_1 z^{-1} - a_2 z^{-2}} \tag{38}$$

Depois multiplica na Equação 38 o numerador e o denominador z^2, para que as variáveis sejam positivas.

$$G(z) = \frac{b_1 z + b_2}{z^2 - a_1 z - a_2} \tag{39}$$

Agora só falta substituir os valores de $\boldsymbol{\theta}$ no modelo ARX apresentado na Equação 39, onde: $a_1 = \theta_1$, $a_2 = \theta_2$, $b_1 = \theta_3$ e $b_2 = \theta_4$.

$$G(z) = \frac{\theta_3 z + \theta_4}{z^2 - \theta_1 z - \theta_2} \tag{40}$$

Então esse é o modelo ARX utilizando o método MQ que é consagrado como a solução ótima do problema de identificação utilizado no trabalho (AGUIRRE, 2007).

4.2 Identificação do modelo com dados reais

Procedeu-se então com a identificação do modelo com dados reais medidos do Filtro-SK, conforme apresentados na Figura 13. O algoritmo de mínimos quadrados foi implementado em Python para obtenção dos parâmetros do modelo com estrutura previamente concebida, como mostra no código 5. Após obtido o valor de teta $\boldsymbol{\theta}$ por

mínimos quadrados, foi possível montar a função de transferência no tempo discreto. No Código 6 têm as variáveis A e B que são definidas com base no teta, sendo A o denominador e B o numerador da função de transferência; já o Gz representa a função de transferência no tempo discreto e, para ser criada, foi utilizada a biblioteca `control` com a função `ct.tf`. Após a utilização dessa função, o modelo obtido é apresentado na Equação 41.

```
# Mínimos quadrados:
theta = np.linalg.inv(M.T@M)@M.T@yr[ni]
```

Listing 5 – Método dos mínimos quadrados

```
# MONTANDO A FUNÇÃO DE TRANSFERÊNCIA NO TEMPO DISCRETO QUE FOI IDENTIFICADA:
A = np.array([1, -theta[0],-theta[1]]) # Termos do denominador do sistema Hz;
B = np.array([theta[2], theta[3]])     # Termos do numerador do sistema Gz;
Gz = ct.tf(B,A,dt = Ts)
```

Listing 6 – Montando a função de transferência identificada

$$G_z = \frac{0,2946z + 0,07316}{z^2 - 0,6401z + 0,09918} \tag{41}$$

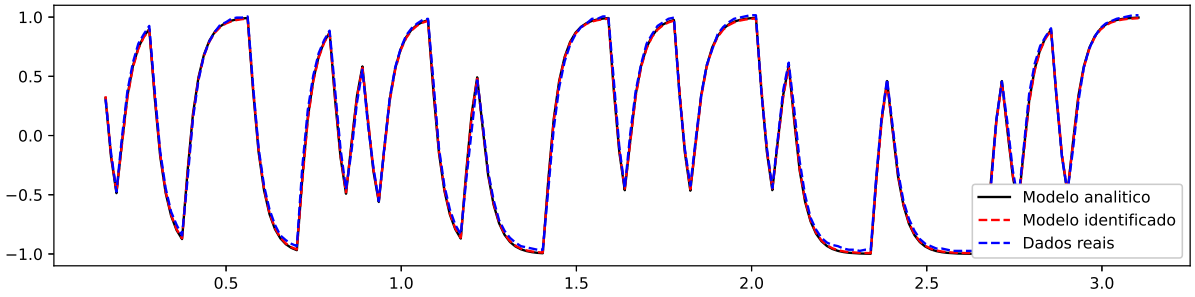
E para a verificação da identificação, foi feita a validação, aplicando um sinal de entrada (dados reais) com o modelo analítico e o modelo identificado, e isso foi feito utilizando a biblioteca `control` com a função `ct.forced_response` como mostra o Código 7, que está gerando sinais de saída dos dois modelos, e esses sinais de saída serão utilizados para comparar o desempenho dos modelos com os dados reais.

```
# APLICANDO UM SINAL DE ENTRADA ur NOS MODELOS ANALÍTICO E IDENTIFICADO
# AMBOS NO TEMPO DISCRETO:

t = Ts*np.arange(0,N)
_, ys = ct.forced_response(Hz, T=t,U = ur,X0 = y0) # Modelo analítico;
_, yp = ct.forced_response(Gz, T=t,U = ur) # Modelo identificado;
```

Listing 7 – Aplicando um sinal de entrada *ur* nos modelos

Para fazer essa comparação, foi plotado um gráfico que mostra o comportamento dos três modelos, como é apresentado na Figura 14.

Figura 14 – Gráfico dos modelos analítico, identificado e os dados reais



Fonte: Elaboração própria

```
# MOSTRANDO O AJUSTE UTILIZANDO O R2;

from sklearn.metrics import r2_score as AjusteR2

print('Ajuste Analítico x Identificado: ', AjusteR2(ys,yp).round(3))
print('Ajuste Analítico x Dados reais: ', AjusteR2(ys,yr).round(3))
print('Ajuste Identificado x Dados reais: ', AjusteR2(yp,yr).round(3))
```

Listing 8 – Ajustes utilizando o R2

Por fim, foi utilizada a função `r2_score` para observar o ajuste dos modelos com os dados reais, como mostra no Código 8 onde está havendo ajuste entre o modelo analítico e identificado, ajuste entre o modelo analítico e dados reais, ajuste entre o modelo identificado e o modelo real.

Tabela 5 – Resultado do ajuste dos modelos

Ajuste	
Modelo Analítico X Modelo Identificado	$99,9\ \%$
Modelo Analítico X Dados reais	$99,4\ \%$
Modelo Identificado X Dados reais	$99,1\ \%$

Fonte: Elaboração própria

Então, analisando esses três resultados na Tabela 5, é possível observar que os modelos estão bem próximos entre si e próximos dos dados reais. Isso significa que tanto o modelo analítico quanto o identificado, ambos são confiáveis e precisos.

4.3 Análise no tempo do modelo identificado - Confrontando com o analítico

4.3.1 Valores dos polos e expectativa da resposta dinâmica

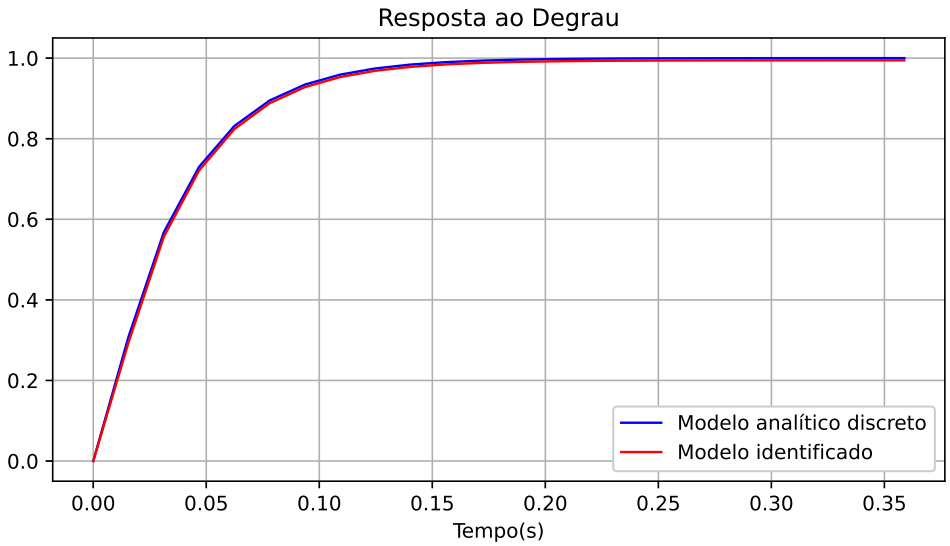
Os valores dos polos do modelo identificado são: $0,6242 + 0,j$ e $0,01589 + 0,j$ e, como já foi visto, para saber se o sistema no tempo discreto é estável, os polos precisam

estar dentro do círculo unitário. Fazendo o módulo de cada polo, eles ficam assim: $0,6242$ e $0,01589$, ambos são menores que 1, estão, consequentemente, dentro do círculo unitário. Observando os polos do modelo analítico no tempo discreto, dá para perceber a semelhança entre eles, o que pode significar que esses modelos descrevem o comportamento dinâmico do sistema de maneira muito próxima.

4.3.2 Resposta ao degrau: tempo de acomodação do sistema

Analisando a Figura 15, é possível perceber que é bastante semelhante aos gráficos da resposta ao degrau do modelo analítico, com um tempo de acomodamento aparentemente de acordo com a imagem, entre $0,10$ s a $0,15$ s, mas utilizando novamente a função `ct.step_info` é possível encontrar o valor do tempo de acomodamento que é $0,1404$ s, que é o mesmo valor do tempo de acomodamento do modelo analítico no tempo discreto.

Figura 15 – Gráfico da resposta ao degrau do modelo identificado e do modelo analítico discreto

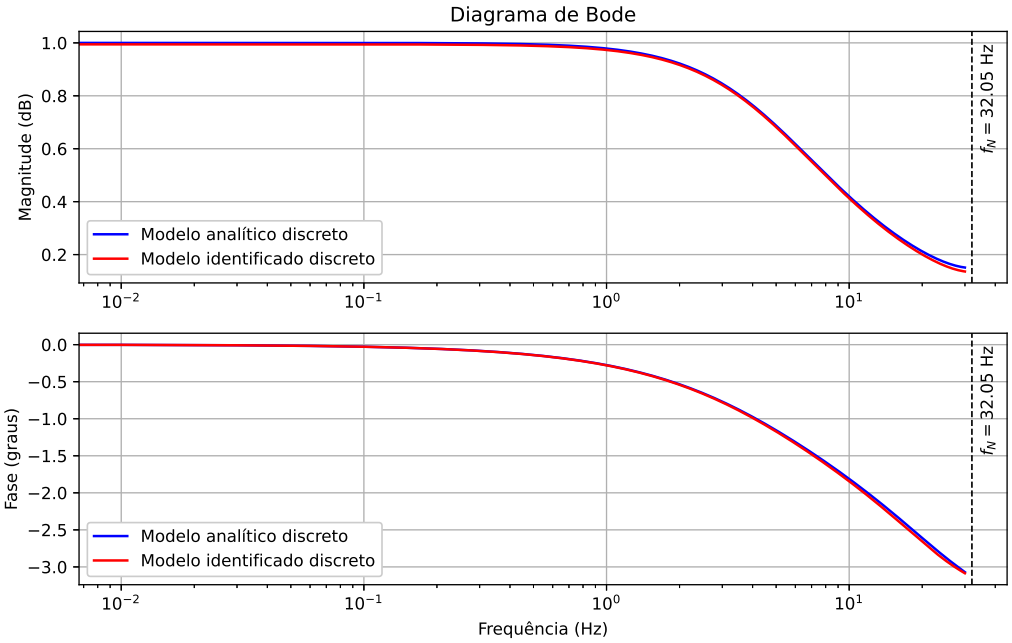


Fonte: Elaboração própria

4.3.3 Resposta em frequência: Frequência de corte - Bode

Observando a Figura 16 que está comparando os dois modelos analítico discreto e identificado discreto, percebe-se a semelhança com o diagrama de Bode do modelo analítico no tempo discreto, até a frequência de corte é aproximada a $1\,Hz$, e a frequência de Nyquist parece estar também entre $30\,Hz$ e $35\,Hz$, mas o seu valor na frequência de Nyquist é $32,05\,Hz$.

Figura 16 – Diagrama de bode do modelo identificado e do modelo analítico discreto



Fonte: Elaboração própria

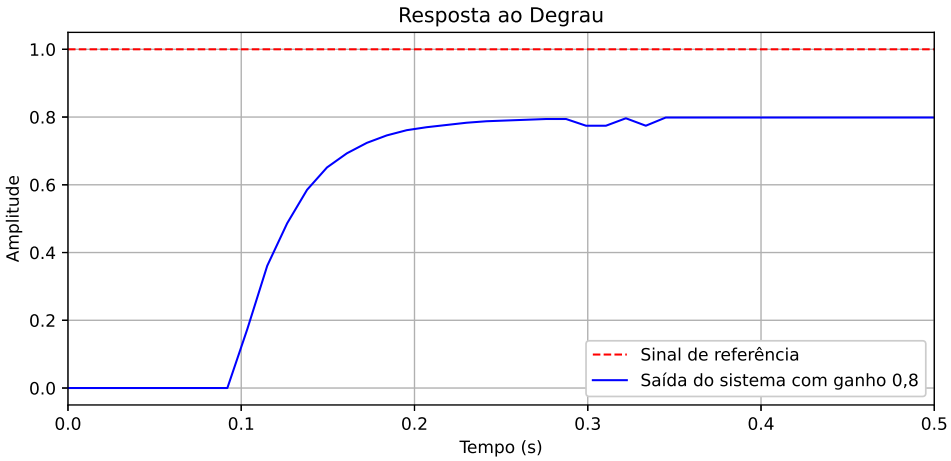
5 Controle de sistemas

5.1 Projeto de controladores

A finalidade desse capítulo de projeto de controladores, serve justamente para demonstrar como pode ser utilizado o Pocket-Lab para o auxílio do estudo. Por ser uma ferramenta compacta, portátil e acessível, o estudante não tem a necessidade de estar em um laboratório para fazer aplicações práticas. Para o estudo de controladores, por exemplo, um aluno com esse dispositivo pode realizar a implementação de controladores a qualquer momento e em qualquer lugar.

Neste capítulo foi introduzido um obstáculo nesse filtro, pois como o Filtro-SK tem o ganho unitário, não oferece nenhuma dificuldade para necessitar projetar controladores, então na sua entrada foi adicionada de forma externa um divisor de tensão com o ganho $0,8$. Na Figura 17 mostra a saída do sistema com a amplitude de $0,8$, onde foi aplicada a divisão de tensão e o objetivo de projetar os controladores é fazer o sinal de saída alcançar o sinal de referência.

Figura 17 – Saida do sistema com ganho 0,8



Fonte: Elaboração própria

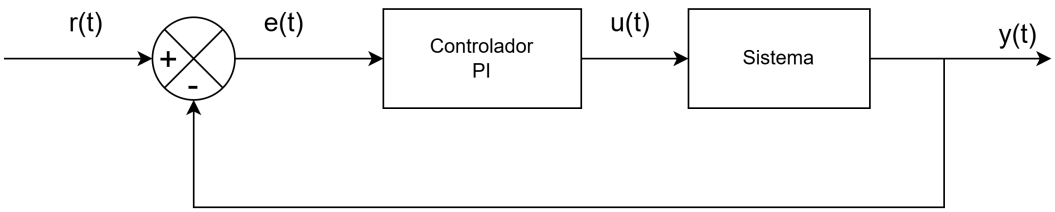
5.2 Controlador PI

A resposta transitória de um sistema linear e invariante no tempo (LTI) em malha fechada depende da localização dos polos no plano complexo que varia em função do ganho em malha aberta. Em alguns casos, só o ajuste desse ganho já pode chegar ao resultado esperado. No entanto, quando não chega ao resultado desejado, então é necessário implementar um controlador em série com a função de transferência da planta. O método do lugar geométrico das raízes (LGR) que foi utilizado é eficiente por mostrar

graficamente as mudanças que acontecem enquanto os polos em malha fechada se deslocam com a variação do ganho, permitindo visualizar seu comportamento com a adição de polos e zeros (RODRIGUES; KURASHIMA; LORDELO, 2023).

Na Figura 18 mostra o diagrama de blocos do controlador PI sendo colocado em série com o sistema.

Figura 18 – Diagrama de blocos do controlador PI



Fonte: Elaboração própria

- r(t): O sinal de referência;
- e(t): erro;
- u(t): sinal do controlador;
- y(t): Sinal de saída.

Como funciona, é aplicado o sinal de referência na entrada e esse sinal de referência é o desejável a atingir, o erro é a diferença entre o sinal de referência e o sinal de saída. Como a ideia é o sinal de saída ser o mais próximo do sinal de referência, então é utilizado o controlador.

Pela necessidade de corrigir o erro em regime permanente, foi utilizado um controlador PI, que significa controlador proporcional integral. Onde a parte do proporcional tenta reduzir o erro entre o sinal de referência e o sinal de saída representado pela Equação 42.

$$C(s) = K_p \tag{42}$$

E a parte Integral ela zera o erro, a qual é representada pela Equação 43.

$$C(s) = \frac{K_i}{s} \tag{43}$$

E junto o proporcional e o integral, a sua representação é apresentada na Equação 44.

$$C(s) = K_p + \frac{K_i}{s} \tag{44}$$

O controlador é representado pela relação entre a saída do controlador e o erro, como mostra na Equação 45

$$\frac{U(s)}{E(s)} = K_p + \frac{K_i}{s} \tag{45}$$

Que pode ser escrita como:

$$\frac{U(s)}{E(s)} = \frac{K_p s + K_i}{s} \tag{46}$$

Colocando K_p em evidência, tem-se:

$$\frac{U(s)}{E(s)} = Kp\frac{(s + K_i/K_p)}{s} \tag{47}$$

Então para ter o sinal do controlador, isolamos o $U(s)$ e o erro vai para o outro lado multiplicando.

$$U(s) = K_p E(s) + \frac{K_i}{s} E(s) \tag{48}$$

Depois de isolar o $U(s)$, foi feita a discretização.

$$U(z) = K_p E(z) + \frac{K_i T_s}{z-1} E(z) \tag{49}$$

Com a discretização realizada, fazemos uma simplificação para uma fração única, multiplicando o K_p por $z - 1$.

$$U(z) = \frac{K_p(z-1) + K_i T_s}{z-1} E(z) \tag{50}$$

No final, o denominador $z - 1$ foi para o outro lado multiplicando o $U(z)$.

$$U(z)(z-1) = (K_p(z-1) + K_i T_s) E(z) \tag{51}$$

Aplicando a propriedade distributiva no $U(z)(z - 1)$, e mandando o $U(z)$ para o outro lado com o sinal positivo.

$$zU(z) = U(z) + (K_p(z-1) + K_iT_s)E(z) \tag{52}$$

Depois foi aplicada em todo o resto da equação a propriedade distributiva e a separação dos seus semelhantes.

$$zU(z) = U(z) + zK_pE(z) + (K_iT_s - K_p)E(z) \tag{53}$$

Para deixar o $U(z)$ isolado, sem o z que o multiplica, é só multiplicar toda a equação por z^{-1} que vai se anular com o z do $U(z)$.

$$U(z) = z^{-1}U(z) + K_pE(z) + z^{-1}(K_iT_s - K_p)E(z) \tag{54}$$

Agora para finalizar, é só passar para o domínio do tempo, quem é multiplicado por z^{-1} vira $(n-1)$, exemplo, $z^{-1}U(z)$ vai ficar $u(n-1)$, para aqueles que não estão sendo multiplicados por z, então fica só (n), por exemplo $K_pE(z)$ que ficou $K_pe(n)$.

$$u(n) = u(n-1) + K_pe(n) + (K_iT_s - K_p)e(n-1) \tag{55}$$

Utilizando a ferramenta do MATLAB chamada rtool (*root locus tool*, que significa: ferramenta do lugar geométrico das raízes), essa ferramenta permite projetar controladores pelo método LGR, permitindo adicionar polos e zeros e observar os seus resultados graficamente. Então, foi possível descobrir o valor de $K_p = 0,1$ e $K_i = 13,2$. Substituindo em 55, resulta:

$$u(n) = u(n-1) + 0,1 \cdot e(n) + (13,2 \cdot T_s - 0,1)e(n-1) \tag{56}$$

Depois de ser encontrada a equação de $u(n)$, agora é só implementar no filtro de Sallen-Key como no Código 9. Primeiro gera o sinal de referência que é uma onda quadrada, representado por `r[n]`, depois utiliza o sinal de referência e a saída em condições iniciais `y_menos1` que é 0, faz a diferença entre os dois e encontra o erro representado por `e_atual` e armazena esse valor em `e[n]`. Após encontrar o erro, já pode conseguir projetar o controlador `u_atual`, o que precisa é o `Kp` ganho proporcional, `Ki` ganho integral, `Ts` o período de amostragem, o `e_atual`, `u_menos1`, `e_menos1` esses dois últimos têm a condição inicial igual a 0. Depois armazena o valor em `u[n]`. No `sinal_pwm` o `u_atual` é convertido para o intervalo PWM do Arduino (0 a 255), o `np.clip` serve para evitar

a saturação, limitando os valores. Já o `conexao.write` envia o valor do PWM por meio da porta `serial` para o Arduino. A `conexao.in_waiting` verifica se há dados na porta `serial`, em `leitura` lê os dados utilizando o `conexao.readline()`, e para o `y_atual = float(leitura)` converte a leitura em `float`. Se a leitura tiver um erro em `ValueError`, ele mantém o último valor do `y_menos1` para que não haja interrupções no `loop`. O `y[n]` guarda o valor lido. Essas variáveis: `y_menos2 = y_menos1`; `y_menos1 = y_atual`; `u_menos1 = u_atual`; `e_menos1 = e_atual` servem para guardar os últimos valores dos passos anteriores para fazer os cálculos recursivos.

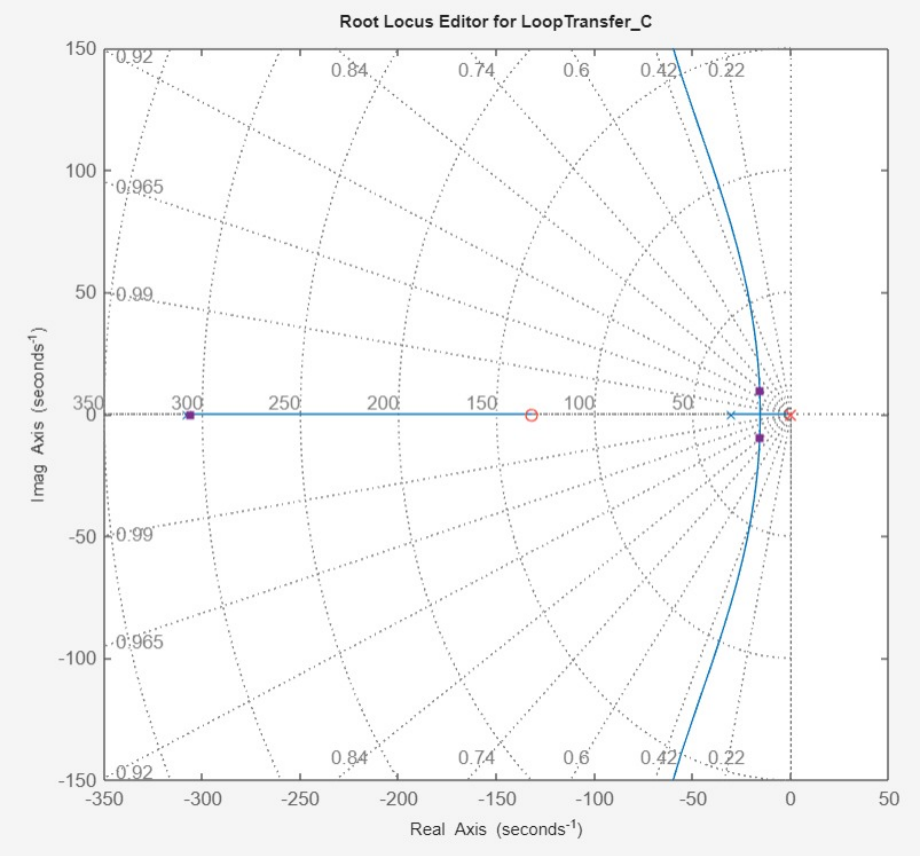
```
for n in range(Ns):
    # Geração do sinal de referência
    r[n] = Amplitude * square(2 * np.pi * freq * n * Ts) + setpoint

    # Cálculo do erro
    e_atual = r[n] - y_menos1
    e[n] = e_atual

    # Ação de controle PI
    u_atual = Kp * e_atual + u_menos1 + (Ki * Ts - Kp) * e_menos1
    u[n] = u_atual

    # Conversão para sinal PWM (0 a 255), limitando para evitar saturação
    sinal_pwm = np.clip((u_atual * 255) / 5, 0, 255)
    conexao.write(str(round(sinal_pwm)).encode())

    # Leitura do sinal da planta
    if conexao.in_waiting > 0:
        try:
            leitura = conexao.readline().decode().strip()
            y_atual = float(leitura)
        except ValueError:
            y_atual = y_menos1

    # Armazenamento da saída
    y[n] = y_atual

    # Atualização das variáveis passadas
    y_menos2 = y_menos1
    y_menos1 = y_atual
    u_menos1 = u_atual
    e_menos1 = e_atual
```

Listing 9 – Código do Controlador PI

Na Figura 19 apresenta o LGR para analisar e simular com adição de polos e zeros utilizando um compensador. Como o controlador é PI, então ele adiciona um polo e um zero, e pode ser observado na equação 47, com o zero sendo igual a $s = -\frac{K_i}{K_p}$ e

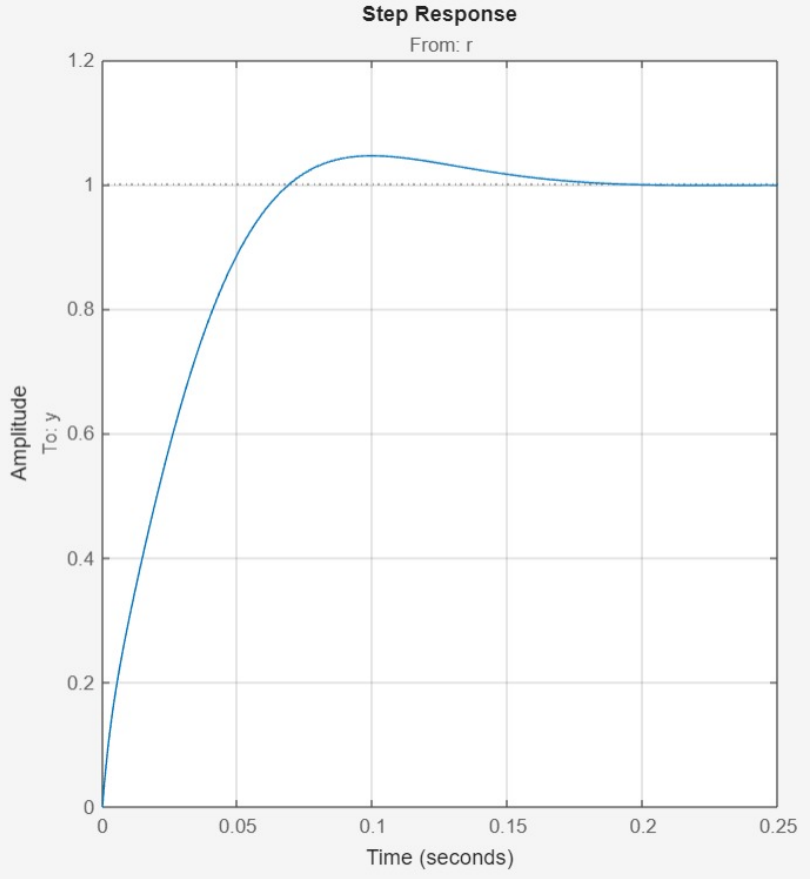
o polo é $s = 0$ o que significa que o polo está na origem. O polo adicionado na origem tem a ação integrativa que é responsável por acumular o erro e forçar a saída a seguir a referência, ou seja, garantir o erro nulo em regime permanente. Enquanto a adição do zero compensa o efeito do polo na origem, fazendo com que os polos originais permaneçam aproximadamente nos mesmos pontos em que estavam no LGR (NISE, 2012). O resultado da resposta ao degrau do controlador PI aplicado ao sistema é apresentado na Figura 20

Figura 19 – LGR do controlador PI



Fonte: Elaboração própria

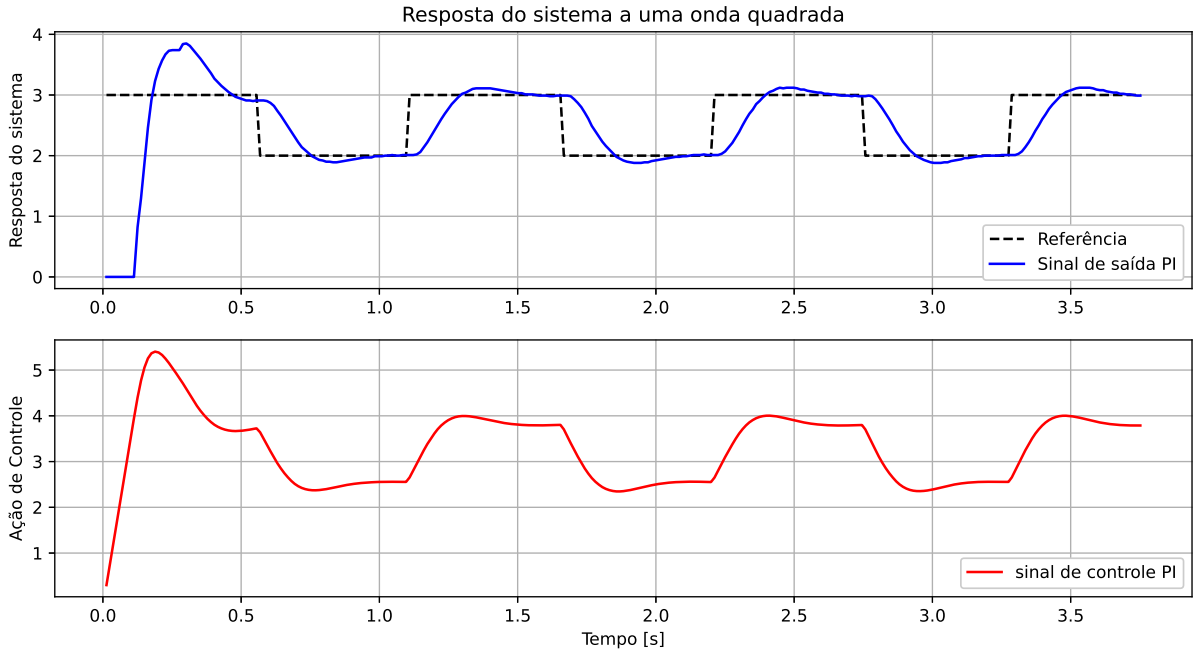
Figura 20 – Resposta ao degrau do controlador PI



Fonte: Elaboração própria

Depois de encontrar os parâmetros pelo LGR e implementar o controlador PI no Python, é gerado um gráfico que é apresentado na Figura 21.

Figura 21 – Sinal de saída usando o controlador PI

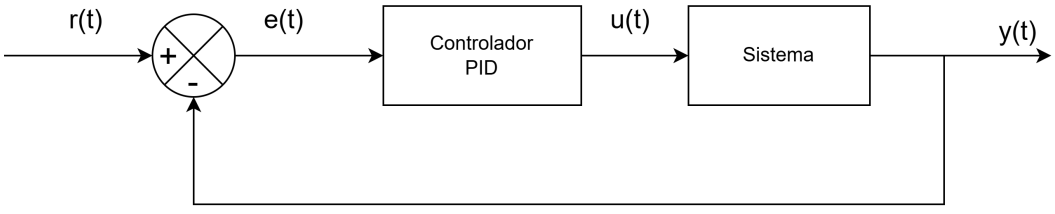


Fonte: Elaboração própria

5.3 Controlador PID

Para o controlador Proporcional Integral Derivativo (PID) é quase a mesma ideia do PI, a diferença é que adiciona o controlador derivativo, que é representado por $K_d s$. "A ação proporcional fornece uma contribuição que depende do valor instantâneo do sinal de erro. A ação integral fornece uma saída do controlador que é proporcional ao sinal de erro acumulado. A ação derivativa atua na taxa de variação do sinal de erro."(RODRIGUES; KURASHIMA; LORDELO, 2023).

Figura 22 – Diagrama de blocos do controlador PID



Fonte: Elaboração própria

- r(t): O sinal de referência;
- e(t): erro;
- u(t): sinal do controlador;
- y(t): Sinal de saída.

A Figura 22 representa um sistema de controle em malha fechada utilizando um controlador PID, e a finalidade é fazer com que o sinal de saída $y(t)$ seja igual ao sinal de referência $r(t)$. O processo começa primeiro pelo sinal de referência, que é o sinal que se deseja alcançar, e será feita a comparação entre $r(t)$ e $y(t)$ e, dessa diferença, surge o erro $e(t)$, que significa o quanto falta para a saída chegar ao sinal desejado. Esse erro é enviado para o controlador, que calcula uma resposta de controle $u(t)$. Então, esse sinal de controle que é gerado pelo PID é aplicado no sistema e a resposta do sistema é o sinal de saída, que vai ser realimentado e comparado novamente com o sinal de referência.

Na Equação 57 mostra uma equação parecida com a do controlador PI, com a adição do $K_d s$. Utilizando a ferramenta do MATLAB o rtool, obteve-se o controlador a partir da observação do LGR e analisando a resposta ao degrau.

$$\frac{U(s)}{E(s)} = K_p + \frac{K_i}{s} + K_d s \tag{57}$$

A equação do controlador PID, também pode ser representada como mostra a Equação 58:

$$\frac{U(s)}{E(s)} = \frac{K_d \left[s^2 + \frac{K_p}{K_d} s + \frac{K_i}{K_d} \right]}{s} \tag{58}$$

A Equação 59 foi obtida pelo rtool:

$$\frac{U(z)}{E(z)} = \frac{2,8057(z^2 - 0,4041z + 0,04729)}{(z-1)(z+1)} \tag{59}$$

Passa o dominador $z^2 - 1$ e $E(z)$ multiplicando no outro lado como na Equação 60:

$$(z^2 - 1)U(z) = 2,8057(z^2 - 0,4041z + 0,04729)E(z) \tag{60}$$

Aplica a distributiva como na Equação 61:

$$z^2 U(z) - U(z) = 2,8057(z^2 - 0,4041z + 0,04729)E(z) \tag{61}$$

Passa o $-U(z)$ para o outro lado com o sinal trocado como na Equação 62:

$$z^2U(z) = U(z) + 2,8057(z^2 - 0,4041z + 0,04729)E(z) \tag{62}$$

Multiplica toda a equação por z^{-2} para que $z^2U(z)$ fique isolado, como na Equação 63:

$$U(z) = z^{-2}U(z) + 2,8057(1 - 0,4041z^{-1} + 0,04729z^{-2})E(z) \tag{63}$$

Na equação 64 temos a equação da saída do controlador, pronta para ser implementada no filtro Sallen-Key e o resultado está na Figura 25.

$$u(n) = u(n-2) + 2,8057(e(n) - 1,1338e(n-1) + 0,1327e(n-2)) \tag{64}$$

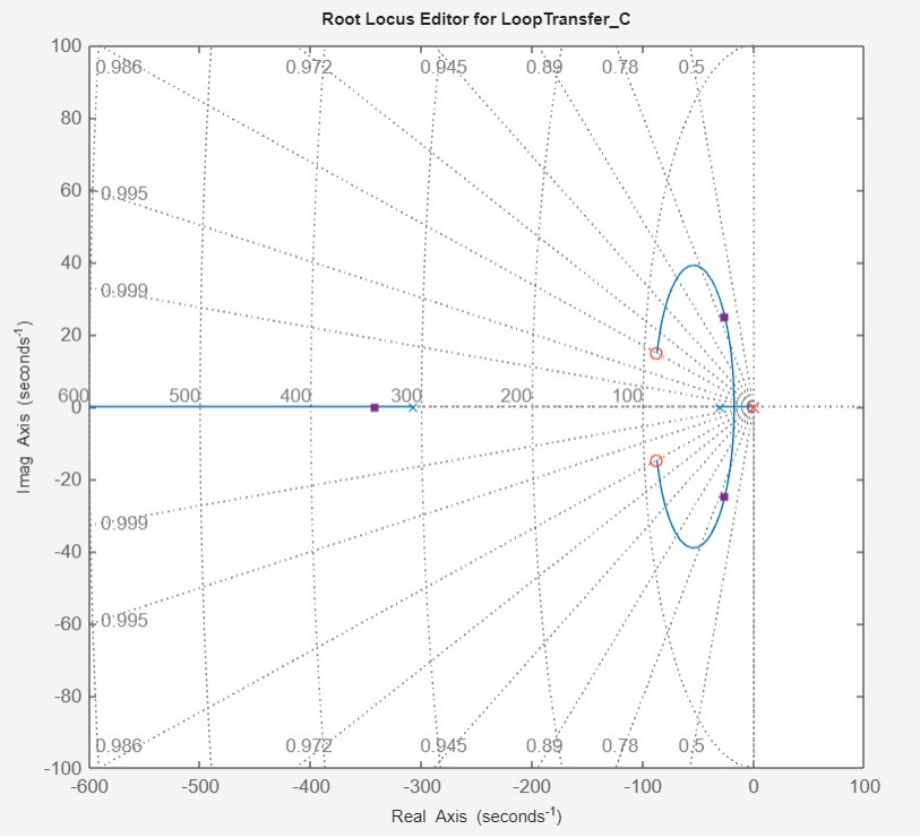
No Código 10 é quase o mesmo que o código do PI, o que diferencia é a parte do controlador, onde aparecem as variáveis `a0`, `a1` e `a2` que são extraídas da equação 64. Já a `u_menos2` é a entrada com o atraso igual a 2, sua condição inicial é 0. Após isso, é salvo o valor `u_atual` em `u[n]`. O resto do código parece com o Código 9.

```
for n in range(Ns):
    # Geração do sinal de referência
    r[n] = Amplitude * square(2 * np.pi * freq * n * Ts) + setpoint

    # Cálculo do erro
    e_atual = r[n] - y_menos1
    e[n] = e_atual

    # Ação de controle PI
    u_atual = u_menos2+(a0*e_atual)-(a1*e_menos1)+(a2*e_menos2)
    u_atual = np.clip(u_atual, 0, 4)
    u[n] = u_atual
```

Listing 10 – Código do Controlador PID

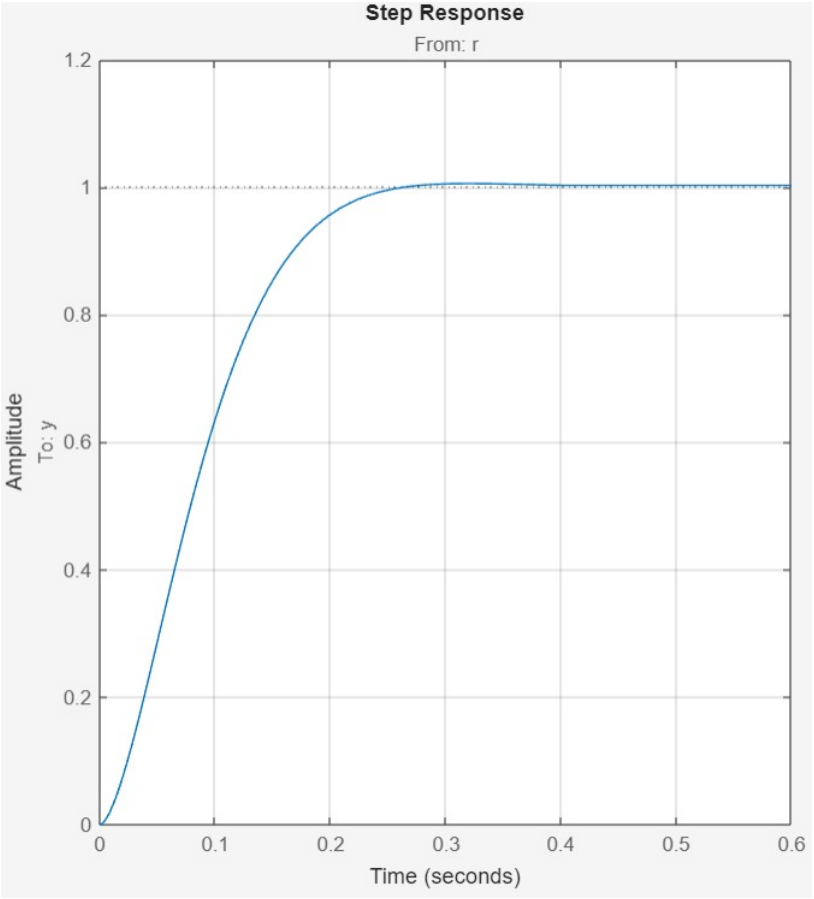
Na Figura 23 mostra o LGR do sistema na ferramenta rtool do MATLAB, onde pode-se adicionar polos e zeros para observar o comportamento desejado na saída. Como o controlador é PID, é adicionado um polo na origem e dois zeros, como mostra a Equação 58. Um polo e um zero são da parte PI, que garantem o erro nulo em regime permanente. Já o outro zero é da parte derivativa, que possibilita melhorar a resposta transitória do sistema, diminuindo o overshoot e o tempo de acomodamento, além de aumentar a velocidade do sistema (NISE, 2012). O resultado da resposta ao degrau do controlador PID aplicado ao sistema é apresentado na Figura 24.

Figura 23 – LGR do controlador PID



Fonte: Elaboração própria

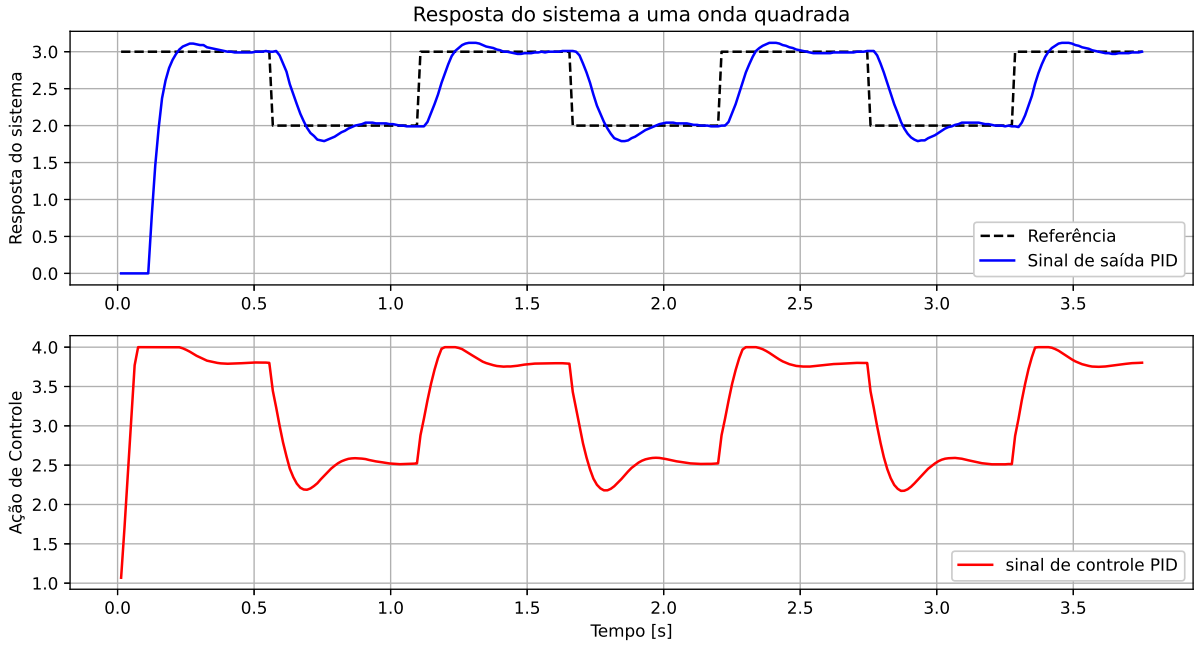
Figura 24 – Resposta ao degrau do controlador PID



Fonte: Elaboração própria

Após encontrar os parâmetros utilizando o LGR e implementar o controlador PID no Python, gerando o gráfico representado pela Figura 25.

Figura 25 – Sinal de saída usando o controlador PID



Fonte: Elaboração própria

6 Conclusão

6.1 Conclusão

O ensino e aprendizagem em engenharia necessita do contato dos alunos com ferramentas práticas, que consolidem conhecimentos teóricos fundamentais. Este trabalho insere-se neste contexto ao oferecer como produto um laboratório de bolso ou Pocket-Lab baseado no filtro de Sallen-Key. O sistema atende às funcionalidades planejadas a priori, ao apresentar resultados que demonstraram boa eficiência em sua construção, mas também nas competências de identificação e controle apresentadas na subseção 6.1.1. Além disso, o Filtro-SK demonstrou ser eficaz por sua simples implementação e replicação, atendendo à portabilidade e ao uso em ambientes fora dos laboratórios convencionais de engenharia de controle, tais como a sala de aula e o ambiente não presencial. Isto abre possibilidades interessantes tanto para a demonstração de experimentos reais no escopo das disciplinas teóricas em sala de aula, quanto para a proposição de atividades de ensino baseadas em experimentos reais para os alunos, que passam a depender apenas de um computador e do Pocket-Lab.

O projeto foi desenvolvido na Universidade Federal do Pará pela graduação em engenharia elétrica e tende a ser aplicado em turmas de graduação e pós-graduação nos próximos anos, sendo aplicadas em áreas voltadas para análise de sistemas lineares e invariantes no tempo (LTI) e controle de sistemas dinâmicos através de inúmeras abordagens que abrangem estes campos de estudo. Por ter uma topologia simples e de baixo custo, é possível ser utilizado em trabalhos e pesquisas independentes.

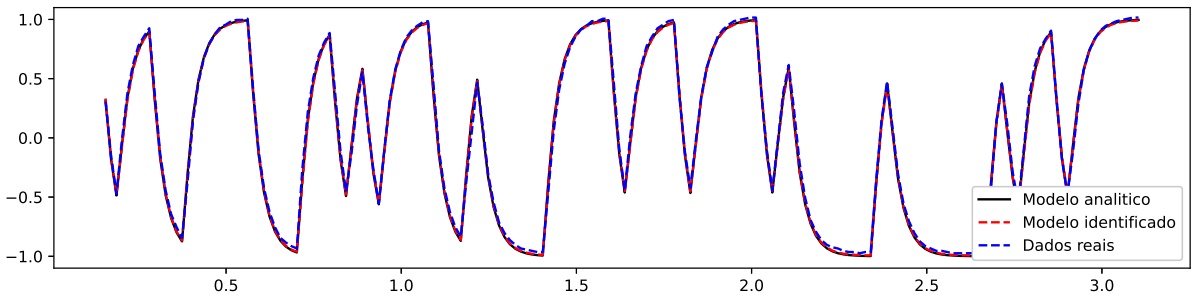
Uma das dificuldades foi projetar os controladores e também por ser a primeira vez utilizando o Arduino e o Python.

Em geral, a aplicação de Pocket-Lab vem ganhando força nos últimos anos, se mostrando uma referência útil para o ensino prático de engenharia. Deste modo, por apresentar uma boa receptividade no âmbito acadêmico e visando a sua maior usabilidade ao longo dos próximos anos, nas etapas futuras a serem desenvolvidas, é fazer a substituição de alguns componentes como o amp-op LM358P que tem suas limitações, como o fato de não ser *rail-to-rail* que pode comprometer os resultados como gerar distorções caso a frequência ou amplitudes mais altas. O apropriado é trocar por um que tenha a propriedade *rail-to-rail* como o MCP6002. Propõe-se também a substituição do Arduino pelo microcontrolador ATtiny85, visando acoplar o dispositivo diretamente à porta USB de um computador, aumentando assim a portabilidade e usabilidade do Pocket-Lab.

6.1.1 Resultados

O principal resultado e produto deste trabalho é o projeto e a implementação do filtro de Sallen-Key no contexto de Pocket-Lab. A placa implementada possui dimensões compactas, medindo 3,5 cm x 4,5 cm, com conexões para alimentação e sinais de entrada e saída para a placa Arduino. Assim, pode-se considerá-la um equipamento real e portátil de fácil instalação e configuração pelos usuários. Com a implementação prática do Pocket-Lab, foi possível realizar a identificação do sistema utilizando o método de mínimos quadrados. Para isso, aplicou-se na entrada PRBS, amostragem de 15 ms, que foi coletada juntamente com a saída do sistema. Com estes sinais foi identificado um modelo LTI discreto de 2ł ordem por função de transferência, com estrutura e parâmetros altamente similares ao modelo nominal. O ajuste verificado entre os dados simulados pelo modelo ultrapassou 99% quando comparado aos dados reais, medido pelo coeficiente de determinação $R2$, o que é altamente apropriado em se tratando de um modelo linear. A Figura 26 evidencia ainda mais a proximidade entre os modelos e os dados reais.

Figura 26 – Gráfico dos modelos analítico, identificado e os dados reais



Fonte: Elaboração própria

Na Tabela 6 apresentam-se os ajustes dos modelos com os dados reais; por sua proximidade nos resultados, mostram uma confiança e precisão dos modelos.

Tabela 6 – Resultado do ajuste dos modelos

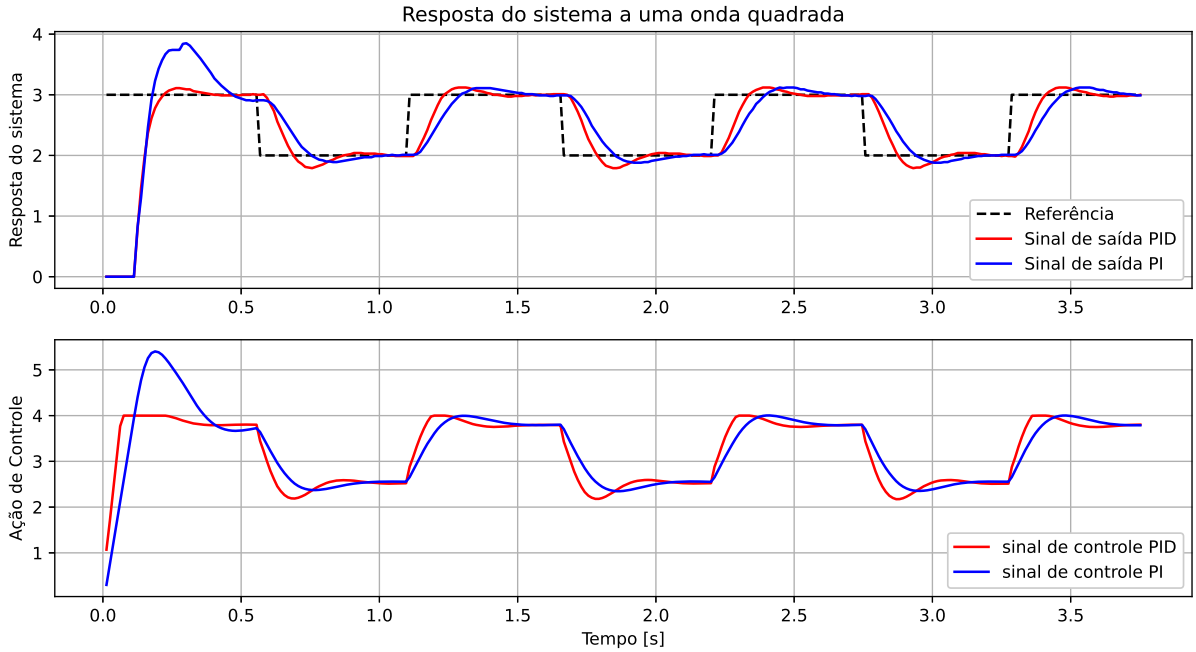
Ajuste		
Modelo Analítico	Modelo Identificado	$99,9\%$
Modelo Analítico	Dados reais	$99,4\%$
Modelo Identificado	Dados reais	$99,1\%$

Fonte: Elaboração própria

Além da identificação do modelo, foi realizado o projeto de controladores PI e PID utilizando o LGR, onde se pode observar o gráfico ao adicionar polos e zeros. Em ambos os controladores foi aplicada na entrada uma onda quadrada. Fazendo uma comparação dos dois controladores, o de melhor resultado é o controlador PID, pois além de corrigir

erro em regime permanente, melhorou a resposta transitória, diminuindo o overshoot e o tempo de acomodamento, como mostra a Figura 27.

Figura 27 – Sinal de saída PI e PID



Fonte: Elaboração própria

Desta forma, o Pocket-Lab se mostrou eficiente com todas essas funcionalidades planejadas, na execução de atividades envolvendo a identificação, análise e projeto de sistemas dinâmicos lineares como: (i) análise de resposta a sinais fundamentais; (ii) análise de resposta em frequência; (iii) identificação e (iv) projeto e implementação de controladores PI e PID. E todos os assuntos de interesse didático na área de engenharia de controle. Comparativamente aos seus equivalentes de mercado ou presentes na literatura, o Pocket-Lab tem como vantagem o baixo custo de implementação e replicação, além da sua portabilidade, que se destaca neste processo pela ausência de fonte de alimentação externa, sendo substituída pela placa Arduino.

Referências

AGUIAR, B. F. M. *Tools and Control Experiences Using TCLab Arduino Kit*. Dissertação (Mestrado) — Universidade do Porto (Portugal), 2020. Citado na página 12.

AGUIRRE, L. A. *Introdução à Identificação de Sistemas–Técnicas Lineares e Não-Lineares Aplicadas a Sistemas Reais*. [S.l.]: Editora UFMG, 2007. Citado 2 vezes nas páginas 41 e 43.

BAYAR, M. F.; KURT, U. Effects of mobile learning science course on students academic achievement and their opinions about the course. *Science Education International*, v. 32, n. 3, p. 254–263, 2021. Citado na página 11.

DANTAS, T. et al. Determinação da taxa de amostragem e analise do efeito do falseamento dos componentes de frequencia de um sinal de identificação. *Blucher Chemical Engineering Proceedings*, Blucher Proceedings, v. 1, n. 2, p. 12440–12447, 2015. Citado na página 27.

INSTRUMENTS, T. Industry-standard dual operational amplifiers. *SLOS068AA. LM358 Datasheet*, 1976. Citado na página 31.

JR, A. P. *Amplificadores Operacionais e Filtros Ativos-8*. [S.l.]: Bookman Editora, 2015. Citado na página 15.

JUNG, W. *Op Amp applications handbook*. [S.l.]: Newnes, 2005. Citado na página 15.

KETSMAN, O.; SANTANA, J. A. C. Engaging students with pocketlab interactive mobile technology in a science classroom: A mixed methods study. *Mid-Western Educational Researcher*, v. 36, n. 1, p. 8, 2024. Citado na página 11.

LATHI, B. P. Linear systems and signals. *ISBN13: 9780195158335*, 2006. Citado 5 vezes nas páginas 16, 19, 23, 25 e 29.

MORAIS, E. V. de et al. Evolução dos laboratórios experimentais de engenharia elétrica: Premissas para o ensino à distância e pesquisa coorporativa. *Revista Visão Universitária*, v. 1, n. 1, 2014. Citado na página 11.

NISE, N. S. Engenharia de sistemas de controle, 6ºed. *LTC, São Paulo*, 2012. Citado 6 vezes nas páginas 17, 21, 23, 24, 53 e 57.

OGATA, K. Modern control engineering. 2020. Citado na página 26.

RODRIGUES, M. d. S. P.; KURASHIMA, C. S.; LORDELO, A. D. S. Design of pid controllers based on root locus for the position control of dc motors. *Journal of Production and Automation (JPAUT) ISSN 2595-9573*, v. 6, n. 1, p. 2–10, 2023. Citado 2 vezes nas páginas 49 e 55.

SANDIM, C. C. Estudo e implemetação de filtros ativos analógicos. 2023. Citado na página 15.

SOUZA, K. C. d. et al. Identificação não-linear no espaço de estados por regressão esparsa de bancada motor-gerador. 2023. Citado na página 41.

TAKAHASHI, H. Early history of laboratory lessons for engineering education. *Journal of JSEE*, Japanese Society for Engineering Education, v. 59, n. 1, p. 1_5–1_10, 2011. Citado na página 11.

TEIXEIRA, R.; SILVA, C.; PARÁ, P. Identificaçao nao linear no espaço de estados de motor-gerador via regressao esparsa. Citado na página 40.