UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE TECNOLOGIA E GEOCIÊNCIAS
DEPARTAMENTO DE ENGENHARIA ELÉTRICA
CURSO DE GRADUAÇÃO EM ENGENHARIA DE CONTROLE E AUTOMAÇÃO

MÂNLIO GOMES FREIRE FILHO

MONTAGEM, IDENTIFICAÇÃO E CONTROLE DE UM ROBÔ SEGUIDOR DE LINHA DE COMPETIÇÃO

Recife
2024

MÂNLIO GOMES FREIRE FILHO

MONTAGEM, IDENTIFICAÇÃO E CONTROLE DE UM ROBÔ SEGUIDOR DE LINHA DE COMPETIÇÃO

Trabalho de Conclusão de Curso apresentado ao Departamento de Engenharia Elétrica da Universidade Federal de Pernambuco, como requisito parcial para obtenção do grau de Engenheiro de Controle e Automação

Orientador(a): Prof. Dr. Fabrício Bradaschia

Recife
2024

Ficha de identificação da obra elaborada pelo autor,
através do programa de geração automática do SIB/UFPE

Freire Filho, Mânlio Gomes.
Montagem, identificação e controle de um robô seguidor de linha de competição / Mânlio Gomes Freire Filho. - Recife, 2024.
99 p. : il., tab.

Orientador(a): Fabrício Bradaschia
Trabalho de Conclusão de Curso (Graduação) - Universidade Federal de Pernambuco, Centro de Tecnologia e Geociências, Engenharia de Controle e Automação - Bacharelado, 2024.
Inclui referências, apêndices, anexos.

1. Controle PID. 2. Identificação de sistemas. 3. MATLAB. 4. Robô seguidor de linha. 5. Robótica móvel. I. Bradaschia, Fabrício. (Orientação). II. Título.

600 CDD (22.ed.)

MÂNLIO GOMES FREIRE FILHO

MONTAGEM, IDENTIFICAÇÃO E CONTROLE DE UM ROBÔ SEGUIDOR DE LINHA DE COMPETIÇÃO

Trabalho de Conclusão de Curso apresentado ao Departamento de Engenharia Elétrica da Universidade Federal de Pernambuco, como requisito parcial para obtenção do grau de Engenheiro de Controle e Automação

Aprovado em: 27/08/2024

BANCA EXAMINADORA

_____

Prof. Dr. Fabrício Bradaschia (Orientador)
Universidade Federal de Pernambuco

_____

Prof. Dr. Márcio Evaristo Da Cruz Brito (Examinador Interno)
Universidade Federal de Pernambuco

_____

Eng. M.Sc. Valdemar Moreira C. Junior (Examinador Externo)
Universidade Federal de Pernambuco

AGRADECIMENTOS

Eu gostaria de expressar minha total gratidão a todas as pessoas que contribuíram de alguma forma na minha jornada acadêmica e para a realização deste trabalho.

Agradeço à minha família, especialmente aos meus pais, Adriana e Manlio, pelo amor incondicional, pelo apoio emocional e financeiro, e por acreditarem no meu potencial. Sem vocês, nada disso seria possível.

À minha estimada noiva e futura esposa, Suelen Gomes, por sua paciência, compreensão, amor e apoio inestimável. Sua presença e incentivo foram mais que fundamentais para a realização deste trabalho.

A todos meus amigos, por estarem ao meu lado, oferecendo palavras de encorajamento e por todos os momentos de descontração que tornaram essa caminhada mais leve.

Aos meus professores e orientadores, pela instrução, paciência e dedicação. Em especial, agradeço ao Professor Fabrício Bradaschia, por sua orientação valiosa, conselhos e incentivos durante todo o desenvolvimento deste trabalho, que se estendeu por mais tempo que o previsto.

Aos membros e amigos da equipe Maracatronics, pela colaboração, pelo aprendizado compartilhado e pelo espírito de equipe que sempre nos impulsionou a alcançar nossos objetivos, em especial à futura subequipe dos autonomos individuais, que devem seguir meu legado.

A todos que, direta ou indiretamente, contribuíram para a realização deste trabalho, meu muito obrigado.

RESUMO

Um robô seguidor de linha é uma máquina autônoma que pode reconhecer e rastrear uma linha contrastante predeterminada pintada no chão, que o robô segue para se mover. Essa é uma das modalidades incluídas em diversas competições de robótica, com o intuito de difundir conhecimento e aspirações tecnológicas, além de ter diversas aplicações industriais. Este trabalho apresenta o desenvolvimento de um robô seguidor de linha, denominado "Avexadinho", cujos componentes e estrutura mecânica foram fornecidos pela equipe de robótica Maracatronics. O processo consistiu em três etapas principais: montagem do robô, identificação do sistema e aplicação do controle. A montagem envolveu a integração dos sensores, motores, microcontrolador e outros componentes eletrônicos, resultando em um robô funcional e qualificado para competições na categoria de seguidor de linha Pro. A identificação do sistema foi realizada utilizando conjuntos de dados experimentais da resposta do sistema a diferentes degraus de entrada por ferramentas do *System Identification* do MATLAB, encontrando-se um modelo matemático de primeira ordem com atraso, tanto em tempo contínuo quanto discreto. O controlador foi projetado, em domínio discreto, usando o mapa do Lugar Geral das Raízes, proporcionando um tempo de assentamento rápido e minimizando oscilações, foi definido como tipo PI para evitar ampliações de ruídos. O robô demonstrou, em testes práticos, que mesmo em condições adversas, ele era capaz de seguir linhas de maneira precisa e estável. O estudo conclui que o projeto foi um sucesso dentro dos objetivos específicos e faz uma contribuição significativa para o campo de robótica móvel e controle de sistemas, podendo ser usado como um modelo base para diversos projetos futuros.

Palavras-chaves: Controle PID; Identificação de sistemas; MATLAB; Robô seguidor de linha; Robótica móvel.

ABSTRACT

A line follower robot is an autonomous machine capable of recognizing and tracking a predetermined contrasting line painted on the floor, which the robot follows to move. This is one of the modalities included in various robotics competitions, with the aim of spreading knowledge and technological aspirations, as well as having several industrial applications. This work presents the development of a line follower robot named "Avexadinho", whose components and mechanical structure were provided by the Maracatronics robotics team. The process consisted of three main stages: robot assembly, system identification, and control application. The assembly involved integrating sensors, motors, a microcontroller, and other electronic components, resulting in a functional robot qualified for competitions in the Pro line follower category. The system identification was performed using experimental data sets of the system's response to different input steps, employing MATLAB's System Identification tools, resulting in a first-order mathematical model with delay in both continuous and discrete time. The controller was designed in the discrete domain using the General Root Locus map, providing a fast settling time and minimizing oscillations. It was defined as a PI-type controller to avoid noise amplification. In practical tests, the robot demonstrated that even under adverse conditions, it was capable of following lines accurately and stably. The study concludes that the project was a success within its specific objectives and makes a significant contribution to the field of mobile robotics and system control, serving as a base model for various future projects.

Keywords: Line-following robot; MATLAB; Mobile robotics; PID control; System identification.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

A/D	Analógico para Digital
ARMAX	*AutoRegressive Moving Average with eXogenous inputs* - AutoRegressivo de Médias Móveis com Entradas Exógenas
ARX	*AutoRegressive with eXogenous inputs* - AutoRegressivo com Entradas Exógenas
AVG	*Automated Guided Vehicle* - Veículo Guiado Automatizado
c.c	Corrente Contínua
EDO	Equação Diferencial Ordinária
FTMF	Função de Transferência em Malha Fechada
IMC	*Internal Model Control* - Controle por Modelo Interno
IR	*Infrared* - Infravermelho
LiPo	Polímero de Lítio
LQG	Linear-Quadrático-Gaussiano
NRMSE	*Normalized Root Mean Squared Error* - Erro de raiz quadrática média normalizado
OBR	Olimpíada Brasileira de Robótica
PLA	Ácido Polilático
PWM	*Pulse Width Modulation* - Modulação por Largura de Pulso
rpm	Rotações por Minuto
RSL	Robô Seguidor de Linha
SLC	Seguidor de Linha de competição
STEM	*science, technology, engineering and mathematics* - ciência, tecnologia, engenharia e matemática
UFPE	Universidade Federal de Pernambuco

LISTA DE SÍMBOLOS

a_i	Coeficiente de Retroação
b_i	Coeficiente de Entrada
c	Centro da Curva
$C(s)$	Modelo Matemático do Controlador no Domínio de Laplace
$C(z)$	Modelo Matemático do Controlador no Domínio de Z
ΔV	Diferencial de Velocidade dos Motores, dado em rpm
ΔV_{pu}	Diferencial de Velocidade dos Motores Normalizado, dado em Porcentagem
d	Distância do eixo dos motores até a faixa dos sensores, dada em mm
ds	Distância da aosição do sensor até o centro da faixa de sensores, dada em mm
$e(k)$	Erro do sistema no domínio discreto
$e(k-1)$	Erro do sistema no domínio discreto para o tempo anterior
$E(s)$	Erro do sistema no domínio de Laplace, $R(s) - Y(s)$
$E(z)$	Erro do Sistema no Plano z
f	Proporção da Velocidade Máxima no Motor Frenado, dada em Porcentagem
$G(s)$	Modelo Matemático da Planta no Domínio de Laplace
$H(s)$	Modelo Matemático do Sensor no Domínio de Laplace
$j\omega$	Eixo da imaginário do plano s
K	Constante de Ganho
K_d	Constante de Ganho Derivativo
K_i	Constante de Ganho Integral
K_p	Constante de Ganho Proporcional
l	Distância entre os Centros das Rodas do Robô, dada em mm

ls	Distância entre Dois Sensores Seguidos, dada em mm
m	Atraso de Entrada para Regressão Linear
n	Atraso de Saída para Regressão Linear
np	Posição Linear do Sensor Avaliado
o	Centro do Eixo dos Motores
r	Raio da Curva, dado em mm
$r(k)$	Entrada de Referência do Sistema no Domínio Discreto
$R(s)$	Entrada de Referência do Sistema no Domínio de Laplace
s	Variável Complexa na Transformada de Laplace
$\phi(t)$	Vetor de Entradas e Saídas Passadas (Regressores)
ψ	Vetor de Coeficiente de Retroação da Regressão Linear
θ	Ângulo Médio Formado entre os Sensores Ativos e o Centro do Eixo dos Motores, dado em Graus
Θ	Posição Angular de Giro do Robô, dada em Graus
Θ_{pu}	Posição Angular de Giro do Robô Normalizada, dada em Porcentagem
T	Tempo
Ts	Tempo de amostragem, dado em ms
$u(k)$	Ação de Controle no domínio discreto
$u(k-1)$	Ação de Controle no domínio discreto no tempo anterior
$U(s)$	Ação de Controle no Domínio de Laplace
$U(z)$	Ação de Controle no Domínio Digital
v_1	Velocidade linear da roda esquerda
v_2	Velocidade Linear da Roda Direita
$x(t)$	Entrada no Domínio do Tempo
$y(t)$	Saída no Domínio do Tempo
$Y(s)$	Saída do Sistema no Domínio de Laplace
z	Variável Complexa na Transformada Z
ω	Velocidade Angular de Giro do Robô

SUMÁRIO

1 INTRODUÇÃO

Um Robô Seguidor de Linha (RSL) é um veículo guiado automatizado (AVG-*Automated Guided Vehicle*) que, com uma série de sensores, motores e outros engenhos mecânicos, eletrônicos ou magnéticos, é capaz de percorrer percursos predefinidos por uma linha sem a interferência humana. Esses dispositivos são objeto de estudo frequente no campo da indústria, onde otimizam processos de logística e automação, no campo de serviços, como em aplicações hospitalares e comerciais e, eventualmente, podem vir a se tornar um importante elemento de transporte urbano, devido à precisão e velocidade na execução de tarefas específicas (PAKDAMAN MEHRAN; SANAATIYAN, 2010).

Existem diversos modelos de RSL voltados à indústria, como o "Weasel" da SSI SCHAEFER Group, que pode ser visto na Figura 1 levando um compartimento de transporte, diferentes modelos de trator de reboque AVG da Asseco CEIT, qual um deles está apresentado na Figura 2 com um periférico de carga, entre outros modelos de grupos como INDEVA, MIR, E80 e ASTI, com aplicações em diversas indústrias, como a Tesla, como pode ser visto no seguinte *link*: https://www.youtube.com/watch?v=8_lfxPI5ObM. Esses robôs industriais dependem e fazem parte de um sistema de monitoramento e controle detalhado de fuções e trajetos definidos por um sistema supervisório integrado à planta industrial (SHETH., 2014).

Figura 1 – Robô seguidor de linha industrial, Weasel da SSI SCHAEFER.

Fonte: (SSI SCHAEFER, 2016).

Na indústria, um RSL pode ser utilizado no transporte de produtos pesados, frágeis ou de risco (como exemplo materiais radioativos, inflamáveis, explosivos, cortantes ou perfurantes) de forma eficiente e segura, aumentando significativamente os ganhos dos processos produtivos. Áreas comuns para o uso desse tipo de robô são da indústria automotiva, em transporte de containers, centros de distribuição, alimentos e bebidas, papel e impressão,

Figura 2 – Robô seguidor de linha industrial, *Tow Tractor* da CEIT.

Fonte: (ASSECO CEIT, A.S., 2014).

medicamentos e cosméticos, aço e outros produtos pesados, em centros de logística e de estoque (ASSECO CEIT, A.S., 2014). Além de aplicações fabris, esse tipo de robô também pode ser usado em aplicações hospitalares, por exemplo, fazendo o monitoramento de pacientes e no transporte de equipamentos e documentos, no ramo de restaurantes, onde pode atuar como uma automatização do garçom e ,em geral, esse tipo de robô pode ser configurado para exercer a função de transporte por um percurso determinado em praticamente qualquer área de atuação (SHETH., 2014).

Um dos grandes problemas do mundo da robótica é a estabilização do robô. Por isso, estudar como diminuir os impactos e oferecer estabilidade é significante e importante (CHAOQUAN LI; XUESHAN, 2011). Num mundo ideal, o RSL está sempre centralizado acima da linha a qual o mesmo está seguindo, mas na prática nem sempre isso ocorre. Problemas e limitações operacionais podem ser responsáveis pelo não cumprimento dos critérios, mas imprecisões causadas pelo algoritmo que o controla são os principais responsáveis pela diferença de desempenho e possível falta de estabilidade entre robôs de características semelhantes. Essa estabilidade é importante, principalmente se o robô encontrar uma mudança de trilha de uma reta para uma curva ou vice-e-versa (SANJAYA ALI ; MAWENGKANG, 2019).

Outra aplicação bem comum de RSLs, é em competições de robótica, que são ambientes mais acessíveis ao público em geral, onde são exibidas disputas entre equipes, promovendo conhecimentos de robótica de forma acessível e recreativa.

1.1 CONTEXTUALIZAÇÃO

Competições de robótica são eventos voltados para o desenvolvimento educacional e tecnológico, incentivando, principalmente em crianças e adolescentes, o interesse em ciência, tecnologia, engenharia e matemática (STEM - *Science, Technology, Engineering and Mathematics*) por meio de disputas entre equipes em categorias nas quais as habilidades e

características dos robôs podem ser testadas e avaliadas de forma prática e lúdica. Competições como a IRONcup, a RSM e a *Winter Challenge*, organizadas pela RoboCore e Inatel, reunem mais de 500 competidores de equipes de diversos países em cada edição (RoboCore, 2023). A Olimpíada Brasileira de Robótica (OBR) é o maior evento de robótica da América Latina, juntando participantes de 27 estados brasileiros, atinge um grande público todo ano. Em 2023 ela contou com a participação de mais de 200 mil estudantes, sem contar com espectadores e aspirantes que assistiram ao evento (ROBOCUP, 2024). Essas competições são compostas por diversas categorias em comum, como combate, futebol e sumô de robôs, além das categorias de seguidor de linha.

A categoria de Robôs Seguidores de Linha desafia os participantes a projetar máquinas capazes de seguir autonomamente trajetos delimitados por linhas contrastantes sem sair completamente dela no menor tempo possível. Um robô capaz de realizar esse feito, estando dentro das especificações definidas pela competição em questão, é chamado de Seguidor de Linha de Competição (SLC). Esta categoria não apenas testa a precisão dos sistemas de sensoriamento e controle, mas também demonstra a aplicação prática de conceitos teóricos em robótica. Um vídeo expositivo sobre os objetivos e características dessa categoria pode ser visto no seguinte *link*: $https://www.youtube.com/watch?v=ET\_KOCVGc98$.

Nesse contexto, a Equipe de Robótica Maracatronics foi criada em Junho de 2012 como um projeto de extensão do Departamento de Engenharia Mecânica da Universidade Federal de Pernambuco (UFPE) voltado para robótica e tencologia que, através de projetos multidisciplinares, desenvolve conhecimentos técnicos, noções na gestão de projetos, trabalho em equipe, gerenciamento de recursos, orçamentos e prazos (INOVA DAMEC, 2017). O Maracatronics tem como sua missão difundir a robótica e fomentar o conhecimento científico e, por isso, a equipe participa de competições reconhecidas nacionalmente, onde já ganhou alguns prêmios, além de participar de projetos externos como aulas de robótica em escolas da rede pública e organização de eventos como, por exemplo, a OBR. Uma foto do Maracatrônics na competição *Winter Challenge* de 2017 pode ser vista na Figura 3.

1.2 MOTIVAÇÃO

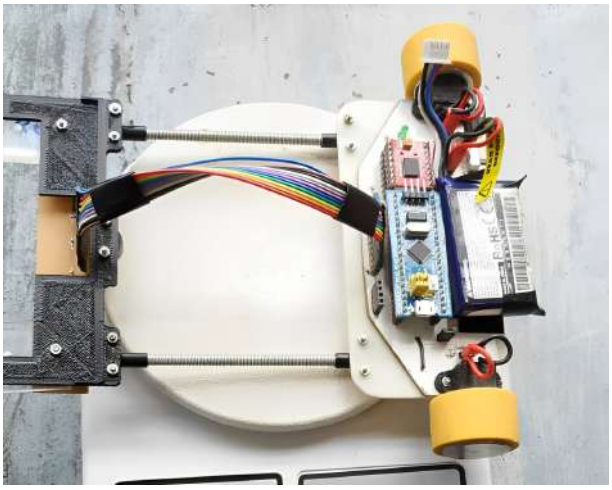
Para competições das categorias de Seguidor de Linha, o Maracatronics possuia um robô criado em 2018 nomeado "Avexado", apresentado na Figura 4, esse Robô SLC nunca chegou a competir, e foi desenvolvido como uma melhoria do SLC "Mói de Quento", representado na Figura 5, que foi capaz de competir na subcategoria de Seguidor de linha Pro na *Winter Challenge* nos anos de 2017 e 2018, como mostrado na Figura 6, mas, em ambas edições, não foi capaz de terminar o percurso. A equipe só voltou a competir nessa categoria em 2022, após a pandemia de Covid-19, depois de realizar algumas alterações improvisadas na configuração do "Avexado", resultando em um robô provisório chamado

"Mulambo", apresentado na Figura 7. Este robô foi capaz de vencer a competição realizada pelo *CESAR School* em uma categoria de seguidor de linha arduíno, uma categoria mais simples que a de Seguidor de Linha Pro, e a principal tomada de tempo realizada nessa competição pode ser vista no *link* https://www.youtube.com/watch?v=x25aRtu9ZZo, no tempo 2:55:05. Vídeos demonstrativos do funcionamento dos robôs SLC acima citados podem ser vistos nos *links* apresentados na Tabela 1.

Figura 3 – Equipe Maracatronics na *Winter Challenge* 2017.

Fonte: Autor.

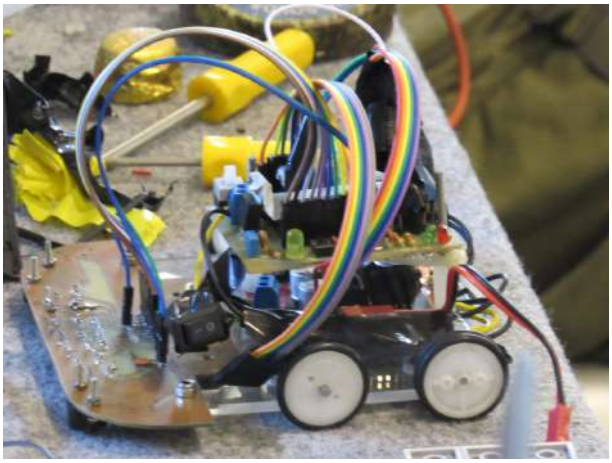
Figura 4 – Robô SLC "Avexado".



Fonte: Autor.

Entretanto esse modelo ainda apresentava limitações e imprecisões. Seu microcontrolador, um Arduino Pro Mini que substituiu o STM32 do "Avexado" devido exigência da competição, bem como sua mecânica e eletrônica, que estavam comprometidas, precisariam ser substituídos e reformulados, pois o controlador não tinha entradas suficientes para

Figura 5 – Robô SLC "Mói de Quento".

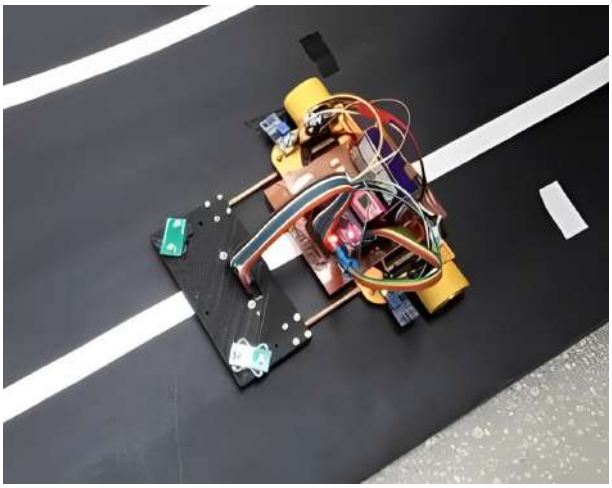


Fonte: Autor.

Figura 6 – Competição seguidor de linha Pro, *Winter Challenge* 2017.

Fonte: Autor.

Figura 7 – Robô SLC "Mulambo".



Fonte: Autor.

um projeto mais avançado e estava apresentando problemas na compilação dos códigos, além de que o projeto apresentou problemas de tensão em alguns pontos da eletrônica e o

Tabela 1 – Vídeos dos antigos robôs SLC da equipe Maracatronics.

Nome do robô SLC	***Link* para vídeo de funcionamento**
Mói de Quento	https://youtu.be/SCa2UQKGcUg
Avexado	https://youtube.com/shorts/OUbNKfUSjpk
Mulambo	https://youtube.com/shorts/4sTiIHDo-iQ

Fonte: Autor.

material das peças começou a demonstrar sinais de desgaste. Como as etapas da construção do "Mulambo" havia sido pouco documentadas e os critérios e objetivos se alinhavam mais com o do robô anterior, seria necessário usar o "Avexado" como base para desenvolver um novo robô SLC capaz de competir na categoria de Seguidor de linha Pro. Também faz-se necessário manter um registro formal e detalhado da montagem do robô, modelagem do sistema e aplicação de um controle que possa ser replicado para outros robôs e aplicações adequadas tanto do Maracatronics quanto de outras equipes de robótica, assim reforçando a missão da equipe.

O "Avexado" apresentava diversos problemas, principalmente em relação ao seu controlador, que foi herdado do "Mói de Quento". Este controlador era um sistema PID (Proporcional-Integral-Derivativo) analógico adaptado digitalmente para aplicação em robótica, com constantes definidas por métodos iterativos. Embora este método tenha apresentado uma melhora no desempenho em comparação ao modelo anterior, ele não conseguiu eliminar as instabilidades que impediram a realização das provas nas competições de 2017 e 2018. Além disso, a mecânica do Avexado podia apresentar variações durante a prova, agravando ainda mais a instabilidade. Outros problemas incluíam dificuldades para compilar os códigos, risco de curtos-circuitos, desconexões de componentes e fluxo de corrente dos motores para o driver, dificuldade na manutenção dos sensores e complicações com módulos externos de *bluetooth* e rádio.

Com o conhecimento dos problemas enfrentados pelos robôs anteriores, podemos abordar o processo de melhoria para o projeto de um novo robô, começando pela otimização do método de controle do "Avexado". A ideia de modelar um controlador clássico para um modelo identificado é vantajosa, pois visa melhorar o desempenho do robô SLC em termos de precisão e estabilidade. Levando em consideração a dinâmica do robô com a pista, desta forma, é possível reduzir instabilidades críticas e definir um comportamento desejado durante o seguimento do trajeto, através da definição correta dos parâmetros do controlador. Além disso, sendo um sistema digital, a aplicação de um controlador digital deve apresentar resultados superiores à digitalização de modelos analógicos utilizados nos modelos anteriores.

1.3 OBJETIVOS

1.3.1 Geral

Elaborar um robô SLC "Avexadinho", que supere em estabilidade e velocidade seu modelo base, o robô SLC "Avexado", usando peças fornecidas pela equipe, criando um guia acessível para futura consulta.

1.3.2 Específicos

Para atingir o objetivo geral, foram definidos os seguintes objetivos específicos, e suas metodologias envolvidas:

- Projetar um robô SLC para a equipe Maracatronics.
  - Integrar componentes fornecidos pela Equipe para construir um robô SLC apto a competir na categoria Seguidor de Linha Pro, testando e aprovando o desempenho deles.
  - Documentar detalhadamente todas as etapas do processo de desenvolvimento, incluindo a montagem, coleta de dados, identificação do sistema, projeto e implementação do controlador, além dos resultados dos testes.
- Definir um método experimental para identificação do sistema.
  - projetar e realizar ensaios para coletar dados da resposta do sistema a diferentes degraus de entrada, garantindo que os dados sejam adequadamente documentados e transportáveis entre diferentes ambientes de trabalho.
  - Utilizar ferramentas do *System Identification* do MATLAB para identificar um modelo matemático adequado para a planta do robô e validá-lo.
- Realizar o projeto do controlador.
  - Avaliar os métodos de controle baseados em PID e selecionar um controlador que possa atender aos requisitos de projeto de forma simples e eficaz.
  - Utilizar o Lugar Geométrico das Raizes e a resposta em frequência para projetar os parâmetros do controlador, buscando estabilidade, rapidez de resposta e minimização de oscilações.
  - Converter o controlador e aplicar ao sistema discreto do microcontrolador.
- Validar o controlador obtido.
  - Realizar testes práticos em diferentes cenários para garantir que o robô SLC siga as linhas de forma precisa e estável.

  - Comparar o desempenho do "Avexadinho" com os robôs anteriores, destacando as melhorias em termos de precisão e controle.

- Documentar Metodologia usada no projeto.
- Identificar limitações e propor melhorias.

1.4 ORGANIZAÇÃO TEXTUAL

Este trabalho de conclusão de curso é organizado da seguinte forma:

- **Capítulo 2: Fundamentação Teórica**: Neste capítulo, são apresentados os principais conceitos teóricos que fundamentam o trabalho, detalhando os componentes que compõem um Robô SLC, as regras de uma competição, e uma base acerca do controle e da identificação de sistema;
- **Capítulo 3: Montagem, Identificação e Controle do Robô SLC**: Este capítulo detalha como o estudo foi conduzido, descrevendo a metodologia utilizada, o desenvolvimento do projeto e os resultados obtidos;
- **Capítulo 4: Conclusão e Trabalhos Futuros**: Este capítulo sintetiza os resultados encontrados e traz uma discução sobre suas implicações. Também são apresentadas recomendações para estudos futuros, destacando limitações e desafios que necessitam ser superados e sugerindo novas áreas de pesquisa e desenvolvimento.

No final deste documento, são apresentadas as referências utilizadas para embasar teoricamente o trabalho, bem como apêndices contendo informações adicionais relevantes, como códigos implementados e um *link* para o *drive* com planilhas, arquivos e vídeos demonstrativos das etapas do projeto.

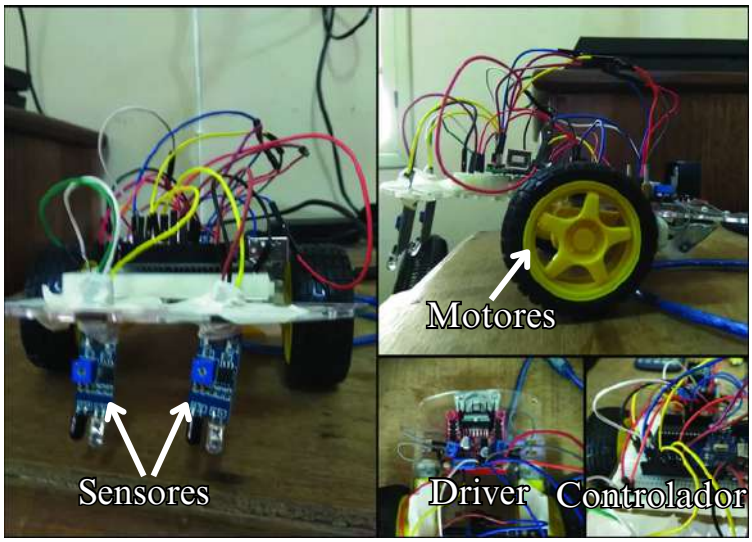
2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo, é exposto o embasamento teórico necessário para a construção deste trabalho. São abordadas características a respeito da montagem do robô e as regras relacionadas às competições da categoria Seguidor de Linha. Também são descritos detalhes sobre o uso de controladores PID e variantes, e métodos de identificação de sistemas, que são utilizados no desenvolvimento do trabalho.

2.1 CARACTERÍSTICAS DE UM ROBÔ SEGUIDOR DE LINHA

Um robô seguidor de linha é uma construção complexa, composta por diversos componentes interligados, cada um desempenhando um papel específico para seu funcionamento. Os principais elementos que constituem um robô seguidor de linha de competição são: sensores de linha, motores, bateria, *driver* para motores, estrutura corporal e controlador, sendo alguns desses apresentados na Figura 8.

Figura 8 – Principais componentes de um RSL.



Fonte: Adaptado de (CANDIDO, 2018).

Os sensores de linha, geralmente, são sensores de refletância, capazes de detectar a luz refletida na superfície próxima e, assim, determinar a posição da linha em relação à faixa de sensores. Esses sensores devem possuir uma alta resolução e robustez para garantir uma resposta rápida às variações de luminosidade e uma ampla faixa de atuação para distinguir com precisão os tons da linha (PAKDAMAN MEHRAN; SANAATIYAN, 2010).

Os motores, geralmente de corrente contínua (c.c.), por sua vez, são responsáveis pela movimentação do robô e representam uma das partes mais desafiadoras do projeto, são a principal parte da estrutura mecânica, além de serem a maior fonte de consumo de energia da bateria, devendo responder de forma rápida e precisa às variações de tensão, com um

bom torque, para garantir um controle eficaz do robô. A escolha adequada dos motores é muito significativa, pois são partes que têm um custo financeiro elevado em relação aos demais componentes, além do fato do custo dos motores crescer exponencialmente em comparação com suas características de torque, potência, peso e volume. Motores fortes em velocidade e torque podem comprometer a leitura dos sensores e atrapalhar o curso, ao erguer a faixa de sensores em arrancadas e fazer o robô SLC derrapar em curvas, enquanto motores fracos podem resultar em desempenho abaixo do padrão esperado. Tais problemas exigem um investimento ainda maior para serem corrigidos, como compra de ventoinhas ou motores adicionais (PAKDAMAN MEHRAN; SANAATIYAN, 2010).

Os *drivers* dos motores são responsáveis por controlar o funcionamento dos motores de forma adequada, servindo como a interface do controlador com os atuadores. Geralmente, são utilizados *drivers* baseados em inversores ponte H (ponte completa) monofásicos, que permitem o controle da velocidade dos motores através da aplicação de tensão em c.c. de amplitude variável, obtida através de uma técnica de PWM (*Pulse Width Modulation* - Modulação por Largura de Pulso) para controle dos disparos dos semicondutores do inversor.

A bateria fornece energia para os componentes eletrônicos e mantém o robô funcionando de forma remota. É preciso que a bateria tenha capacidade de fornecimento de corrente suficiente para alimentar os motores em potência máxima durante todo o período da prova da competição.

A estrutura do robô refere-se ao arranjo mecânico das peças, onde são dispostos os componentes eletrônicos e mecânicos. Considerações como o centro de massa, disposição das peças e a massa total do robô devem ser levadas em conta, uma vez que podem afetar significativamente o desempenho do robô SLC, tanto nas manobras quanto na aceleração e desaceleração dos motores.

O controlador é o cérebro do robô, sendo responsável por interpretar os sinais dos sensores e enviar os comandos necessários para garantir um desempenho eficiente. Deve possuir capacidade de processamento suficiente para executar cálculos e comandos com precisão, além de contar com periféricos adequados e suficientes para interligar todos os componentes do robô (HASAN KAZI MAHMUD; NAHID, 2012).

Após a explicação de cada componente da montagem e seu funcionamento no robô, é necessário apresentar as regras específicas que devem ser seguidas para que o robô esteja apto a competir. Essas regras variam de acordo com cada competição e categoria e devem ser rigorosamente observadas durante o processo de construção e operação do robô.

2.2 REGRAS DE UMA COMPETIÇÃO DE SEGUIDOR DE LINHA

As competições de robôs seguidores de linha geralmente possuem um conjunto de regras específicas que regem o *design*, a construção e a operação dos robôs participantes. Estas regras são estabelecidas para garantir um ambiente justo e seguro, bem como

promover a inovação e criatividade dos participantes (ROBOCORE, 2023). Por existirem diversas competições de robótica em âmbito nacional e internacional de diferentes níveis e modalidades, não é raro que uma mesma categoria possua algumas diferenças em suas regras de acordo com o nível de complexidade, público-alvo e proposta de exibição da competição, apesar disso, muitas competições compartilham um conjunto comum de regras para categoria de seguidor de linha.

Para esse trabalho são usadas como referência as regras da RoboCore, versão atualizada com base nas regras e recomendações redigidas pela Comunidade dos Seguidores de Linha do Brasil datada de 30/04/2021 (ROBOCORE, 2023). Excluindo as regras de definições de equipe e fazendo um apanhado dos tópicos mais importantes, podemos classificar as regras como sendo de dois tipos: Especificações do Robô e Especificações da Competição (ROBOCORE, 2023).

2.2.1 Especificações do Robô

Para ser elegível para competir na categoria de Seguidor de Linha Pro, um robô deve atender a certas premissas básicas. Os robôs devem ser totalmente autônomos e com todos os componentes embarcados. Eles não podem ser controlados externamente, com exceção para serem iniciados ou para ajustes de parâmetros. Estes por sua vez só podem ser alterados entre duas tentativas, seja de maneira remota ou por meio de chaves ou botões, desde que não haja qualquer alteração no código do robô SLC. Também existem alguns limitadores importantes para a montagem: o robô não pode exceder 250 mm de comprimento, 250 mm de largura e 200 mm de altura, não podendo alterar suas dimensões durante a tomada de tempo (ROBOCORE, 2023).

Embora algumas competições permitam o uso de métodos de empuxo para aumentar a força normal em relação ao solo para melhoria do desempenho, como ventoinhas, turbinas ou helices, muitas competições regionais não permitem tais métodos, portanto a equipe Maracatronics evita usá-los. Outro ponto importante é que poucas são as competições que limitam o peso do robô para essa categoria, visto que velocidade e peso apresentam uma relação de crescimento inversamente proporcional nesse contexto. Também são poucas as competições que restringem os componentes, como tipo e quantidade de motores, sensores, placas controladoras e baterias. Por outro lado, algumas competições restringem usos de motores com desempenho muito superiores.

2.2.2 Especificações da Competição

Sintetizando o método de avaliação, que basicamente envolve a realização de três tentativas para completar um percurso recolhendo a melhor tomada de tempo, restam questões sobre o percurso, que via de regra é o trajeto definido por uma linha branca de aproximadamente 19 mm de largura que se estende a partir de uma marcação de partida até uma marcação de chegada em uma pista composta por uma ou mais mantas emborrachadas

de cor preta colocadas sobre uma superfície plana, podendo conter emendas e possíveis desníveis. A linha consiste em combinações de retas e arcos que pode cruzar sobre si com um ângulo de intersecção de 90° entre dois trechos de retas. Em algumas competições, a pista apresenta também marcações no lado direito da linha (em relação ao sentido do percurso) indicando os pontos de partida e de chegada e marcações no lado esquerdo no ponto em que houver alteração da curvatura do trajeto (ROBOCORE, 2023).

Outros aspectos importantes incluem os materiais utilizados, que podem variar em tonalidade e aderência, e o ambiente da competição, que pode apresentar variações na luminosidade, inclinação da superfície e outros eventos fortuitos.

Para que um robô atinja seu melhor desempenho durante a competição, é necessária uma programação precisa e um controle eficaz de suas funções. Para otimizar o desempenho do robô durante as tomadas de tempo, frequentemente são aplicadas diferentes formas de controle, como On/Off aplicado em (HASAN KAZI MAHMUD; NAHID, 2012) e (PAKDAMAN MEHRAN; SANAATIYAN, 2010), método *Fuzzy* aplicado em (SANJAYA ALI ; MAWENGKANG, 2019), controle baseado em PID, como aplicado em (SHETH., 2014), avanço e atraso de fase, controle preditivo, entre outros. Mesmo que alguns métodos de controle possam guiar um robô SLC melhor, a escolha de qual método usar é flexível, e pode ser adaptada seguindo critérios do programador ou da equipe.

2.3 O CONTROLE DO ROBÔ

O controle em sistemas robóticos envolve a manipulação das variáveis dinâmicas de um robô, principalmente por meio de técnicas de *feedbacks* em malhas fechadas, visando alcançar objetivos específicos de funcionamento, como estabilidade, precisão e resposta rápida a perturbações. Nesta seção, são explorados princípios fundamentais do controle de robôs, desde os conceitos básicos até técnicas mais avançadas.

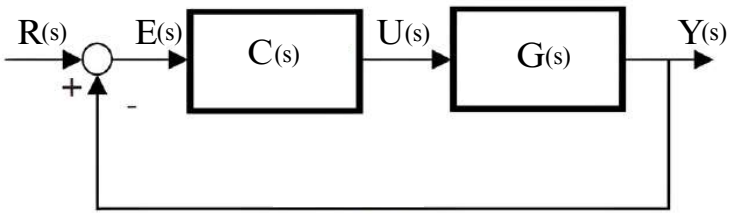
2.3.1 Conceitos Iniciais

O controle clássico se destaca como uma abordagem tradicional para projetar sistemas de controle, utilizando técnicas matemáticas e analíticas para garantir estabilidade e bom desempenho. Através da modelagem e análise do sistema, que capturam a física essencial do mesmo, o controle clássico garante que o sistema se comporte de forma previsível e atinja os objetivos desejados, como resposta rápida a mudanças e alta precisão (ÅSTRöM; MURRAY, 2006).

Sistemas de controle em malha fechada usam o *feedback* para comparar o desempenho real do sistema com o desempenho desejado e realizar as correções necessárias. Essa estrutura de controle inclui pelo menos dois sistemas dinâmicos: a planta, que representa o sistema físico a ser controlado, e o controlador, que calcula as ações corretivas com base no *feedback* recebido da planta (LANDAU; ZITO, 2006). A Figura 9 ilustra um exemplo

simples de sistema de controle em malha fechada, onde $G(s)$ representa a planta, $C(s)$ representa o controlador, $Y(s)$ representa a saída do sistema em malha fechada, $R(s)$ se trata da entrada do sistema, geralmente definida como um *setpoint* ou referência a ser seguida, $E(s)$, representa o erro, sendo a diferença entre a entrada e a saída retornada pelo *feedback*, $U(s)$ representa a ação de controle.

Figura 9 – Diagrama de blocos simplificado do robô SLC.



Fonte: Adaptado de (LANDAU; ZITO, 2006).

À luz desse trabalho, o controlador do sistema pode ser definido como o algorítimo interno ao microcontrolador e a planta pode ser definida como a montagem completa do robô SLC, tendo como os principais pontos os motores, servindo de atuador, e os sensores, como o responsável pelo *feedback*.

Para desenvolver as análises no sistema em malha fechada é preciso conhecer o conceito de função de transferência, que é uma descrição compacta da relação entrada/saída para um sistema linear. A combinação de funções de transferência com diagramas de blocos fornece um método poderoso para lidar com sistemas lineares complexos. A forma mais comum para tratar sistemas na teoria de controle clássico é a transformada de Laplace para domínio da frequência, que envolve a representação matemática dos sistemas e sinais em termos de suas componentes de frequência, facilitando a compreensão do comportamento do sistema em resposta a diferentes entradas e condições, permitindo a representação das funções de transferência e a resolução de equações diferenciais lineares (LANDAU; ZITO, 2006). Essa transformada resulta em uma equação de onde é possível se extrair de forma mais simples, além de outras características, a ordem do sistema.

A ordem do sistema refere-se à complexidade do comportamento dinâmico do sistema, determinada pelo número de integrações ou diferenciações nas equações que descrevem o sistema. Para sistemas lineares, a ordem é igual ao número de derivadas presentes nas equações diferenciais, no domínio da frequência, isso equivale à maior potência de s no denominador da função de transferência (OGATA, 2010).

2.3.2 Estabilidade

A estabilidade é uma propriedade fundamental de sistemas de controle, definindo se o sistema é capaz de manter seu comportamento projetado ao longo do tempo, mesmo na presença de perturbações ou variações nas condições de operação. Um sistema estável é

capaz de retornar a um estado de equilíbrio após ser perturbado, garantindo uma resposta de saída que tende a seguir a variável de referência.

Em relação à ordem, sistemas de ordem elevada tendem a ser mais complexos e são mais propensos a terem menor faixa de estabilidade para um dado controlador, também são mais sensíveis a perturbações e variações paramétricas quando comparadas a um sistema de menor ordem com o mesmo controlador (OGATA, 2010).

A estabilidade do sistema em malha fechada será determinada pelas partes reais das raízes (polos) do denominador da função de transferência em malha fechada FTMF(s) (LANDAU; ZITO, 2006). Um sistema estável não apresentará oscilações crescentes ou um comportamento divergente ao longo do tempo. Para ser estável, o sistema deve seguir por base dois critérios: quando o sistema é excitado por uma entrada limitada em amplitude (ou energia), a saída também deve ser limitada em amplitude (ou energia) e, na ausência da entrada, a saída deve tender a zero independentemente das condições iniciais. Se o sistema for linear, contínuo e invariante no tempo, a estabilidade pode ser definida pela tabulação de Routh-Hurwitz e pela análise da estabilidade relativa usando os conceitos de margens de fase e ganho na resposta em frequência (OGATA, 2010). Um sistema digital é considerado estável se todos os seus polos estiverem dentro do círculo unitário no plano complexo. Por outro lado, se algum polo estiver fora do círculo unitário, o sistema será instável e sua saída acabará divergindo para o infinito (LANDAU; ZITO, 2006).

A localização dos polos da função transferência afetam diretamente na estabilidade do sistema e pode ser avaliada utilizando um mapa de polos e zeros, que se trata de um sistema de coordenadas no plano s, onde o eixo das abscissas se refere à parte real e o eixo das ordenadas é a parte imaginária do polo. O plano s também é utilizado para realizar o projeto de controladores pelo método do lugar geométrico das raízes.

2.3.3 Controle Proporcional-Integral-Derivativo e suas variações

O controle é responsável por duas características muito importantes do sistema em operação, os regimes estacionário e transitório. O regime estacionário se refere ao estado que o sistema estará ao final da aplicação da resposta inicial, ou seja, é o estado em que o sistema se encontra depois de decorrido um tempo relativamente longo. Nele são analisados importantes aspectos sobre a estabilidade e ganhos do sistema de controle. O regime transitório se refere ao estado em que o sistema se situa desde quando é estimulado pelo sinal de entrada até entrar no regime estacionário. Os principais aspectos analisados nele são a ordem do sistema e o comportamento do sistema nesse período, pois existem algumas características nele que podem ser afetadas pelo controle a ser utilizado, podendo ser usadas como critérios do projeto do controlador, como tempo de atraso, tempo de subida, tempo de pico, máximo sobressinal e tempo de acomodação. A característica da resposta transitória de um sistema de malha fechada está diretamente relacionada à localização dos polos de malha fechada (OGATA, 2010).

O controle Proporcional-Integral-Derivativo (PID) é a forma mais comum de controlar malhas fechadas em sistemas dinâmicos lineares invariantes no tempo, devido à sua simplicidade e eficácia. Ele oferece um equilíbrio entre estabilidade, precisão e resposta dinâmica. Ele funciona com base em constantes de atuação de controle proporcional, integrativo e derivativo, sendo capaz de definir uma ação de controle com base no presente, no passado e na previsão de futuro dos erros do sistema de controle (LANDAU; ZITO, 2006). Cada componente pode ser descrito mais detalhadamente como sendo:

- Ação proporcional (P) é responsável por corrigir o erro presente no sistema de forma proporcional ao valor do erro. Isso significa que quanto maior o erro, maior será a correção aplicada. A ação proporcional é crucial para definir uma reação imediata de controle, ajudando a reduzir o erro estático do sistema de controle, porém é sensível a ruídos na medição do erro e é incapaz de eliminar o erro estático por si só;

- A ação integral (I) atua sobre o valor acumulado passado do erro do sistema de controle. A partir da evolução do erro acumulado, é possível encontrar uma ação de controle capaz de fazer a grandeza controlada atingir o valor de referência do sistema de controle com precisão. Isso ajuda a eliminar o erro de regime permanente, porém um ganho integral alto pode levar a um tempo de resposta lento e à saturação do atuador, especialmente em sistemas com grandes constantes de tempo, e induzir oscilações no sistema, especialmente em conjunto com a ação derivativa;

- A ação derivativa (D) é responsável por prever a trajetória futura do erro do sistema de controle, com base na taxa de mudança atual do erro. Isso permite definir uma ação de controle imediata que reaja à tendência do erro, ajudando a reduzir oscilações excessivas e melhorando a estabilidade do sistema, porém é altamente sensível a ruídos na medição do erro e intensifica a resposta do sistema a altas frequências, podendo levar a oscilações e instabilidade.

A ação de controle obtida na saída de um controlador PID pode ser representada pela seguinte equação no domínio do tempo:

$$u(t) = k_p \cdot e(t) + k_i \cdot \int_0^t e(\tau)\, d\tau + k_d \frac{de}{dt} \tag{2.1}$$

onde $u(t)$ é o sinal de controle, $e(t)$ é o erro, k_p é o ganho proporcional, k_i é o ganho integrativo e k_d é o ganho derivativo. No domínio da frequência, a ação pode ser representada pela equação:

$$U(s) = K_p \cdot E(s) + \frac{K_i}{s} \cdot E(s) + K_d \cdot s \cdot E(s) \tag{2.2}$$

Observando as equações anteriores, nota-se que trabalhar em domínio da frequência, por não depender de integrais e derivadas, pode facilitar na montagem do experimento e na transformação e discussão do sistema, que será posteriormente discretizado para uso no controlador do robô SLC (OGATA, 2010).

No entanto, o controlador PID pode apresentar alguns problemas de desempenho ou complexidades desnecessárias em alguns casos. Neste contexto, os controladores P, PI e PD são alternativas complementares que expandem as possibilidades de controle em diversos cenários ao abstrair de algum tipo de ganho, suas características estão simplificadas a seguir:

- Controlador Proporcional (P): Responde diretamente ao erro, ajustando a saída de forma proporcional à sua magnitude. Ele é indicado para sistemas simples, por ter uma baixa complexidade de implementação e uma rápida resposta a distúrbios, entretanto, é incapaz de gerar erro estacionário nulo e é sensível a ruídos.

- Controlador Proporcional Integral (PI): Elimina o erro em estado estacionário, assegurando que a saída do sistema converja para o valor de referência desejado. Essa característica o torna indicado para aplicações onde a precisão absoluta é essencial, como em sistemas de controle de temperatura ou velocidade, contudo, ele apresenta sensibilidade a ruídos de baixa frequência, tendência à oscilação em alguns sistemas e tem um tempo de assentamento mais lento.

- Controlador Proporcional Derivativo (PD): Proporciona uma resposta rápida e eficaz a distúrbios. É indicado para aplicações onde a resposta dinâmica é crítica, como em sistemas de controle de posição ou velocidade de motores, porém, é sensível a ruídos, mais complexo para implementar e suscetível a instabilidade em alguns sistemas.

A escolha do tipo de controlador depende das características específicas do sistema a ser controlado, dos requisitos de desempenho desejados e das condições de operação. Uma análise profunda do sistema e dos objetivos da aplicação é fundamental para a seleção do controle mais adequado (OGATA, 2010).

2.3.4 Métodos de Projeto de Controlador

Para encontrar as constantes do controlador, seja do tipo P, I, PI, PD ou PID, existem diversas formas de sintonização, das quais as mais comuns podemos citar a sintonização por Ziegler-Nichols, que consiste em estimar valores com base no comportamento da planta a uma entrada específica, a resposta em frequência, construindo o diagrama de bode do controlador com base na margens de ganho e de fase desejadas, na otimização computacional, que utiliza artifícios computacionais para encontrar as melhores constantes para os critérios desejados, e por alocação de polos e zeros para a melhoria das características de resposta, que consiste em criar um controlador que, em uma malha fechada, faça o sistema se comportar de acordo com uma função desejada, podendo até alterar a ordem do sistema por anulação de polos e zeros (OGATA, 2010).

Existem outros métodos capazes de moldar o comportamento de sistemas, garantindo estabilidade, precisão e eficiência, que recebem um maior foco nesse estudo.

O método do Lugar Geométrico das Raízes (LGR) é uma técnica gráfica eficaz para projetar e analisar sistemas de controle em malha fechada. Ele mapeia as raízes da equação característica no plano complexo s conforme o ganho do sistema varia. Esse método oferece uma visão clara do comportamento do sistema em relação às mudanças no ganho, facilitando o ajuste preciso do sistema para atender aos requisitos específicos de estabilidade e desempenho (OGATA, 2010).

Com base na função de transferência, é possível traçar o lugar das raízes no plano complexo s. Isso envolve a localização das raízes da equação característica do sistema para diferentes valores do ganho do controlador, assim, é possível determinar como as características de desempenho do sistema variam com as constantes do controle PID. Por exemplo, é possível identificar os valores das constantes que resultam em uma resposta rápida, estável e sem oscilações.

O projeto de controlador com resposta em frequência é uma abordagem que se baseia na análise das características de frequência do sistema, como ganho e fase, para projetar um controlador que atenda às especificações de desempenho desejadas. Isso envolve as etapas: Análise frequencial da planta, usando o o diagrama de Bode ou o diagrama de Nyquist; Definição de especificações de desempenho, incluindo requisitos de estabilidade, tempo de resposta e rejeição de perturbações; Projeto do controlador, que pode ser feito utilizando os conceitos de alocação dos zeros, polos e ganho do controlador no diagrama de Bode de forma a atingir as margens de fase e ganho e frequência de passagem desejadas; E validar o controlador, que por meio de simulações computacionais ou testes em planta real, permitindo verificar se o sistema atende às especificações de desempenho estabelecidas e identificar possíveis melhorias ou ajustes adicionais necessários. Essa metodologia permite o desenvolvimento de sistemas de controle precisos, estáveis e robustos para uma variedade de aplicações (OGATA, 2010).

O projeto de controladores utilizando os métodos mencionados anteriormente só é possível quando se conhece o modelo matemático do atuador, da planta e do sensor que estão presentes no sistema de controle. Alguns métodos de identificação de sistemas conseguem obter modelos matemáticos sem a necessidade de calcular formalmente as equações diferenciais ordinárias (EDOs) que regem o comportamento dinâmico dos sistemas, nem encontrar explicitamente os parâmetros dessas EDOs.

2.4 IDENTIFICAÇÃO DO ATUADOR, DO ROBÔ E DO SENSOR

A identificação matemática do sistema desempenha um papel crucial no seu projeto do sistema de controle. Este processo envolve a obtenção de um modelo matemático representativo do sistema real, o que pode ser alcançado por meio de métodos analíticos ou experimentais. Nesta seção, os princípios essenciais da identificação de sistemas são apre-

sentados, destacando as abordagens e os desafios associados (ISERMANN R. ; MÜNCHHOF, 2011).

2.4.1 Modelagem do sistema

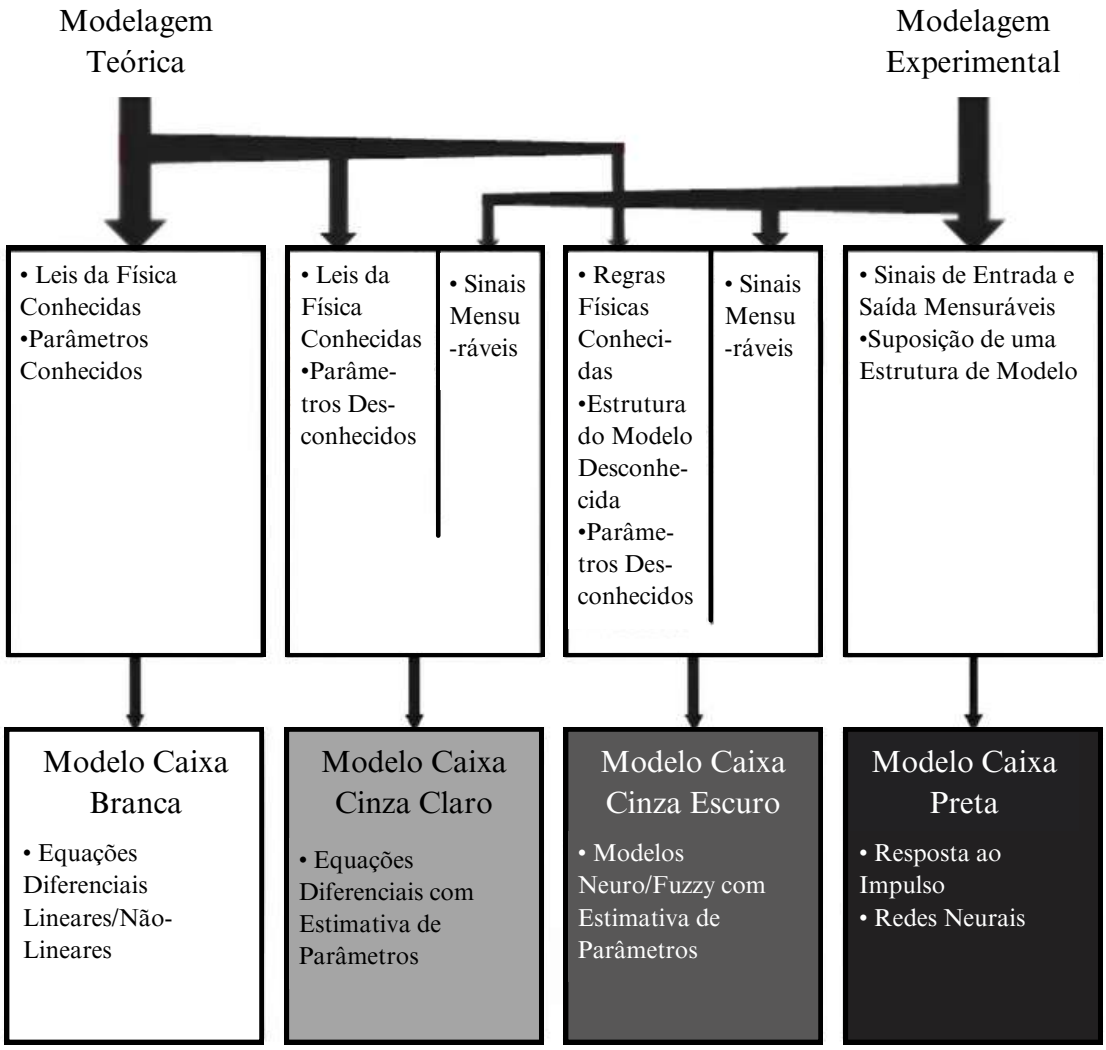
A modelagem de um sistema fornece uma representação matemática que descreve o seu comportamento dinâmico. Existem diferentes abordagens para modelagem, desde modelos simplificados baseados em princípios físicos até modelos mais complexos que capturam detalhes específicos do sistema. A escolha do modelo depende da aplicação e da disponibilidade de dados (KEESMAN, 2011).

Os métodos para obtenção dos modelos podem ser categorizados em três tipos principais: caixa branca, caixa preta e caixa cinza. O modelo de caixa branca é baseado no conhecimento dos princípios físicos e das equações diferenciais ordinárias que regem o comportamento dinâmico do sistema, sendo especialmente útil quando há um bom entendimento dos processos subjacentes ao sistema. O modelo de caixa preta, por outro lado, é uma representação empírica que descreve o sistema apenas com base em dados observados, sem levar em consideração os detalhes internos do sistema, sendo útil quando não se tem um entendimento completo dos processos físicos do sistema. Já o modelo de caixa cinza é uma combinação dos dois modelos supracitados, incorporando conhecimento físico e dados experimentais, resultando em modelos que respeitam o comportamento dinâmico ditado pelas EDOs, porém utilizando os dados empíricos para obter coeficientes precisos para tais EDOs (LJUNG, 1998). Essa distinção pode ser melhor observada na Figura 10.

Independente do método abordado, um modelo matemático é sempre uma aproximação do sistema real. Na prática, a complexidade do sistema, o conhecimento prévio limitado e a indisponibilidade de obtenção de dados completos impedem uma descrição matemática exata do sistema. Contudo, mesmo que haja pleno conhecimento do sistema e dados suficientes disponíveis, o modelo resultante poderia se tornar muito complexo e custoso para ser usado a depender da aplicação, então uma descrição exata muitas vezes não é necessária (KEESMAN, 2011). Consequentemente, a identificação do sistema é considerada uma modelagem aproximada para uma determinada aplicação com base em dados observados e conhecimento prévio do sistema. O procedimento de identificação descrito na Figura 11 delineia as etapas envolvidas no processo de obtenção de um modelo matemático de um sistema utilizando o método de modelagem por caixa-preta ou caixa-cinza.

Como mencionado anteriormente, o conhecimento prévio, os objetivos e os dados são os principais elementos no procedimento de identificação do sistema, devendo-se compreender que estas entidades não são independentes. Na maioria das vezes, os dados são coletados com base no conhecimento prévio do sistema e nos objetivos de modelagem, levando a um projeto de experimento apropriado. Ao mesmo tempo, os dados observados também podem levar a um ajuste do conhecimento prévio ou mesmo dos objetivos (KEESMAN, 2011).

Figura 10 – Diferentes tipos de modelagem de sistemas.



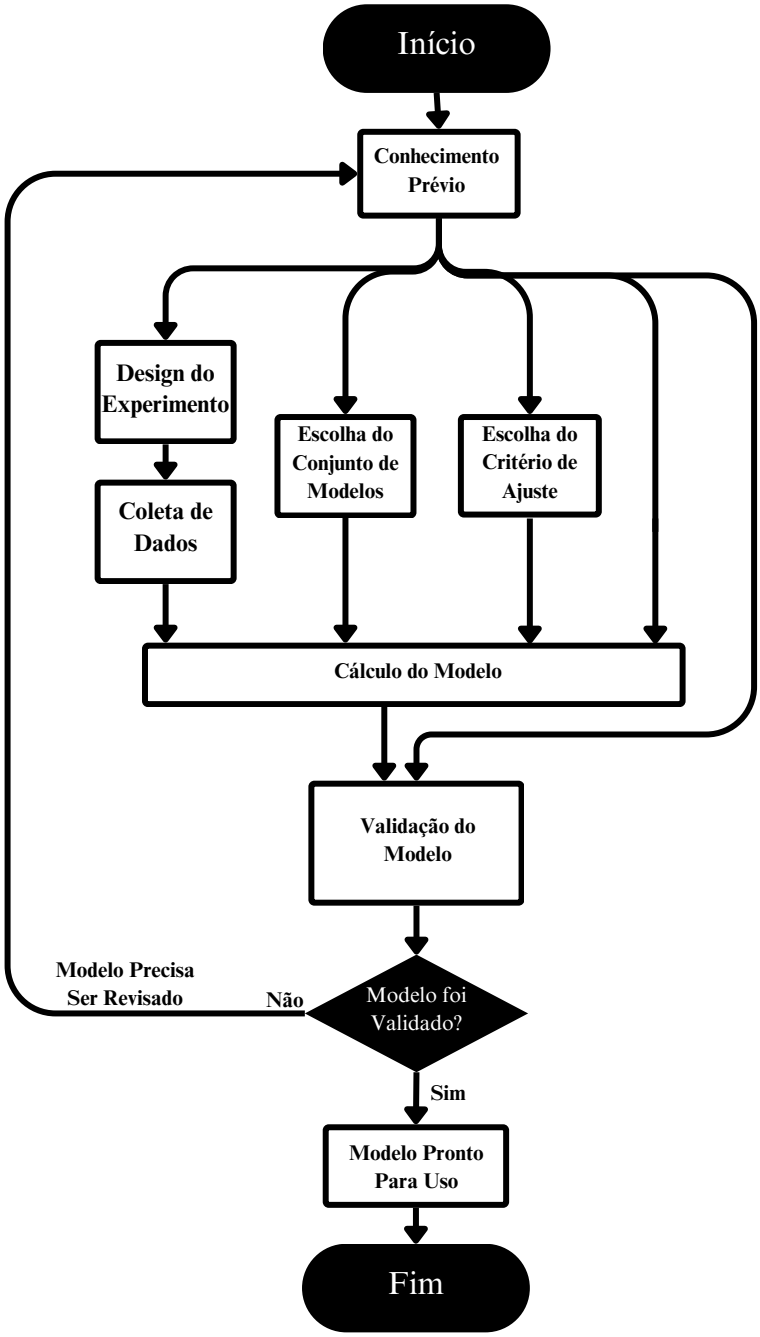
Fonte: Adaptado de (ISERMANN R. ; MÜNCHHOF, 2011).

2.4.2 Não Linearidades do Sistema

Os métodos de modelagem mais tradicionais, como funções de transferência, espaço de estados e a transformada de Laplace, necessitam que o sistema seja linear, ou seja, precisam que sejam satisfeitas as propriedades da homogeneidade, que diz que a saída do sistema varia proporcionalmente à uma variação escalar da entrada, e da superposição, que diz que se duas entradas $x_1(t)$ e $x_2(t)$ produzem as saídas $y_1(t)$ e $y_2(t)$, respectivamente, então a entrada $x_1(t) + x_2(t)$ produz a saída $y_1(t) + y_2(t)$ (OGATA, 2010). No entanto, na prática, muitos sistemas exibem comportamentos não lineares que podem impactar significativamente a precisão dos modelos e a eficiência dos controladores. Essas não linearidades devem ser identificadas e tratadas a fim de evitar problemas na identificação do sistema.

A saturação de controladores é uma das não linearidades mais comum em sistemas

Figura 11 – O ciclo da identificação de sistema.



Fonte: Adaptado de (KEESMAN, 2011).

de controle. Ela é um fenômeno que ocorre devido a restrições físicas nos atuadores e hardwares ou limitações do processo controlado, quando a entrada do atuador excede seus limites máximos ou mínimos. Matematicamente, a saturação pode ser representada por uma função não linear que limita a saída do controlador dentro de um intervalo predefinido (OGATA, 2010).

Outro ponto de não linearidade que pode afetar na identificação e no controle do

sistema é a zona morta, que se trata de uma região de inatividade do atuador onde, para um intervalo específico de valores de entrada, a saída do atuador permanece constante. Em termos matemáticos, a zona morta é definida como um intervalo de valores de entrada para o qual a função de transferência do atuador é zero (OGATA, 2010).

Embora não seja uma não-linearidade propriamente dita, a presença de atrasos pode resultar em comportamentos não lineares em sistemas de controle de velocidade ou posição. Os atrasos introduzem uma defasagem entre a aplicação da ação de controle e a observação de seus efeitos. Métodos de identificação incluem testes de resposta ao degrau e análise de frequência. Ao lidar com atrasos, é essencial considerar seu impacto no desempenho do sistema, o que pode exigir a implementação de estratégias de compensação adequadas. No entanto, em algumas situações, os atrasos podem ser pequenos o suficiente para serem negligenciados, simplificando o projeto do controlador (OGATA, 2010).

Compreender essas não linearidades é fundamental para o projeto e para implementações eficazes de sistemas de controle. Modelar adequadamente e desenvolver estratégias de compensação apropriadas garantem o desempenho desejado do sistema em várias condições operacionais (OGATA, 2010).

Dentro do robô SLC, os principais componentes suscetíveis a essas não linearidades são:

- Saturação: tanto os motores, que não podem exceder sua faixa de tensão de operação independentemente da capacidade da bateria e do *driver*, quanto os terminais dos circuitos integrados (módulo de sensores, *driver* dos motores e microcontrolador), que operam dentro de uma faixa limitada pelas suas tensões de alimentação;
- Zona Morta: manifesta-se nos motores durante a operação em baixas velocidades;
- Atraso: todos os componentes contribuem para um atraso geral, sendo os de dinâmica mais lenta, como os motores e *drivers*, os mais significativos nesse aspecto.

2.4.3 Coleta de Dados e Projeto dos Experimentos

A obtenção de dados experimentais é essencial para a identificação do sistema. Isso geralmente envolve o projeto e a realização de experimentos controlados, nos quais entradas conhecidas são aplicadas ao sistema e as saídas correspondentes são registradas. O projeto adequado dos experimentos é crucial para garantir a qualidade e a relevância dos dados coletados (LJUNG, 1998), devendo ser cuidadosamente planejado para garantir a qualidade e a relevância dos dados obtidos. Alguns pontos importantes a se considerar incluem:

- Seleção das entradas de teste: as entradas de teste devem ser escolhidas de forma a proporcionar informações úteis sobre o comportamento do sistema, devendo ser representativas das condições operacionais esperadas e variar ao longo de uma faixa

relevante de operação, levando em consideração o tipo de sistema, os objetivos da identificação e os recursos disponíveis. A frequência e a amplitude das entradas de teste devem ser escolhidas de acordo com as características do sistema, considerando as limitações dos atuadores e garantindo que as entradas estejam dentro da faixa operacional aceitável. Degraus, rampas, impulsos e sinais aleatórios são os tipos de entrada de testes mais comuns (LJUNG, 1998). Em um robô SLC pontos que precisam ser levados em consideração são as condições do ambiente e do robô, a faixa de atuação dos sensores e atuadores e a aplicação dos métodos para leitura e para controle. Os sinais de entrada mais comuns são do tipo degrau.

- Escolha do tempo de amostragem e duração do experimento: o tempo de amostragem deve ser escolhido cuidadosamente para garantir a precisão da identificação. O procedimento de amostra dos dados produzidos pelo sistema é inerente aos sistemas de aquisição de dados. É inevitável que a amostragem como tal conduza a perdas de informação, e é importante selecionar a frequência de amostragem para que essas perdas sejam insignificantes. A amostragem deve ser escolhida de forma que seja possível capturar com um bom nível de detalhes o comportamento dinâmico do sistema quando a entrada é aplicada ao mesmo. Além disso, a duração do experimento deve ser suficiente para capturar todas as dinâmicas do sistema, esperando o mesmo atingir o regime permanente. Tais práticas auxiliam na estimação precisa dos parâmetros (LJUNG, 1998). Para robôs SLC o melhor tempo de amostragem depende da dinâmica de seus motores e da capacidade do controlador, sendo este o maior tempo entre duas amostragens, com grandes diferenças de velocidade, sem que nenhuma informação dos sensores seja perdida.

- Replicação e aleatoriedade: replicação refere-se à repetição dos experimentos sob as mesmas condições para reduzir a variabilidade dos resultados e aumentar a confiabilidade das conclusões. Ao realizar múltiplas replicatas, é possível avaliar a consistência dos resultados e identificar possíveis fontes de erro experimental. Por outro lado, a aleatoriedade diz respeito à aleatorização na seleção da ordem dos experimentos e na distribuição das condições de teste. Isso ajuda a reduzir o viés experimental e a garantir que os efeitos de variáveis não controladas sejam distribuídos de forma uniforme entre as replicatas. A aleatoriedade também permite uma melhor generalização dos resultados para além das condições específicas do experimento (LJUNG, 1998). Em testes de prova do Robô SLC, a replicação e aleatoriedade, quanto mais exercidas, geram maiores quantidades de dados para análise, criando assim, dados mais confiáveis.

Em resumo, o projeto dos experimentos para coleta de dados deve ser guiado pelos objetivos da identificação do sistema e pelas características específicas do sistema sob investigação. Após a coleta dos dados, é essencial realizar uma análise cuidadosa para

extrair informações relevantes sobre o comportamento do sistema, podendo envolver técnicas estatísticas para identificar padrões ou tendências nos dados, estimando assim os parâmetros do sistema.

2.4.4 Métodos de Estimação de Parâmetros do Sistema

A aplicação dos dados no sistema de identificação pode ser descrita por meio de técnicas de regressão linear. A regressão linear é uma ferramenta estatística utilizada para modelar a relação entre uma variável dependente, $y(t)$, e uma ou mais variáveis independentes, $x(t)$ (LJUNG, 1998).

A abordagem básica da regressão linear consiste em encontrar os parâmetros do modelo que melhor se ajustam aos dados observados. Esses parâmetros são estimados de forma a minimizar a diferença entre as saídas do modelo e do sistema real obtida através de dados medidos, sendo geralmente estimada pelo cálculo do erro quadrático médio.

Matematicamente, a regressão linear pode ser expressa como:

$$y(t) + a_1 \cot y(t-1) + ... + a_n \cdot y(t-n) = b_1 \cdot x(t-1) + ... + b_m \cdot x(t-m) \tag{2.3}$$

Nesta equação, $y(t)$ representa a saída do sistema no tempo t, $x(t)$ é a entrada do sistema, a_i e b_i são os coeficientes de retroação e de entrada e n e m são os atrasos da saída e da entrada respectivamente (LJUNG, 1998).

Para simplificar a notação, podemos representar os coeficientes a_i e b_i em forma de vetor ψ, e as saídas e entradas passadas em forma de vetor $\phi(t)$. Assim, a equação de diferença linear pode ser reescrita como:

$$y(t) = \phi^T(t) \cdot \psi \tag{2.4}$$

Onde:

- $\psi = [a_1...a_n \quad b_1...b_m]^T$ é o vetor de parâmetros do modelo.
- $\phi(t) = [-y(t-1)... - y(t-n)u(t-1)...u(t-m)]^T$ é o vetor de regressores.

A estrutura do modelo que é linear em ψ é semelhante a um modelo de regressão linear. Os regressores são os componentes do vetor $\phi(t)$ e o objetivo é estimar os parâmetros ψ que melhor ajustam o modelo aos dados observados (LJUNG, 1998).

Para estimar os parâmetros do modelo, são utilizadas técnicas de regressão linear. Umas das técnicas mais comuns é a estimação por mínimos quadrados, amplamente utilizada devido à sua simplicidade computacional e propriedades estatísticas bem definidas. Nela, os parâmetros são estimados minimizando a diferença quadrada entre as saídas observadas e as saídas previstas pelo modelo. Outras técnicas que oferecem abordagens alternativas importantes que merecem serem citadas são:

- Estimação de Máxima Verossimilhança: neste método, os parâmetros do modelo são estimados maximizando a função de verossimilhança dos dados observados. Essa abordagem assume que os dados observados são amostrados de uma distribuição probabilística e busca os parâmetros que maximizam a probabilidade de observar os dados (ISERMANN R. ; MÜNCHHOF, 2011);

- Estimação por Mínimos Quadrados Ponderados: semelhante à estimação por mínimos quadrados, mas atribui pesos diferentes a diferentes observações. Esses pesos são determinados com base na confiabilidade das observações, dando mais importância às observações mais confiáveis e menos importância às observações menos confiáveis (ISERMANN R. ; MÜNCHHOF, 2011);

- Técnicas Baseadas em Algoritmos de Otimização: existem várias técnicas de otimização que podem ser aplicadas à regressão linear, como o método do gradiente descendente, o método de Newton-Raphson e algoritmos baseados em inteligência computacional. Esses métodos buscam os parâmetros do modelo que minimizam uma função objetivo, que pode ser uma função de erro ou uma função de verossimilhança (ISERMANN R. ; MÜNCHHOF, 2011).

Dessa forma, a aplicação dos dados no sistema de identificação envolve o ajuste de um modelo linear aos dados experimentais, com o objetivo de estimar os parâmetros do modelo que melhor descrevem o comportamento do sistema (LJUNG, 1998). É importante também mencionar a existência de ferramentas de identificação de sistemas no MATLAB, principalmente a *System Identification Toolbox*, que usa diversos métodos, como otimização computacional e mínimos quadrados, para estimar modelos por regressão linear em formato de funções transferências, espaço de estados, modelos polinomiais, entre outros (MATHWORKS INC., 2024).

2.4.5 Validação do Modelo

Após a estimação dos parâmetros, é necessário verificar se o modelo identificado é capaz de reproduzir com precisão o comportamento do sistema em condições diferentes das usadas para sua identificação. A validação envolve a comparação das saídas do modelo com os dados observados que não foram utilizados durante o processo de estimação. Isso permite verificar se o modelo é capaz de reproduzir, com precisão e um bom grau de generalidade, o comportamento do sistema em condições diferentes das usadas para a identificação (KEESMAN, 2011).

Existem várias técnicas de validação disponíveis, das quais destacam-se:

- Divisão em Conjuntos de Treinamento e Teste: os dados são divididos em dois conjuntos distintos: um conjunto de treinamento, utilizado para identificar o modelo, e um conjunto de teste, reservado para validar o modelo identificado. Essa abordagem

verifica a capacidade do modelo de generalizar a operação do sistema com dados não utilizados na identificação (ISERMANN R. ; MÜNCHHOF, 2011). É comumente aplicada em projetos com grande volume de dados, permitindo a separação de dados para treinamento e teste.

- Validação Cruzada: os dados são divididos em k partes, onde o modelo é treinado k vezes. Em cada iteração, uma das partes é utilizada como conjunto de teste e o restante como conjunto de treinamento, assim, seleciona o modelo melhor validado enquanto avalia possíveis problemas nos dados (LJUNG, 1998). É preferencial quando há limitação de dados, possibilitando múltiplas iterações de treinamento e teste para uma avaliação robusta do modelo.

- Validação por Subamostragem: uma fração dos dados é selecionada aleatoriamente como conjunto de teste, sendo o restante utilizado para treinamento. Esse processo é repetido várias vezes com diferentes divisões dos dados, e os resultados são agregados para uma estimativa mais robusta do desempenho do modelo (LJUNG, 1998). Similar à validação cruzada, permite a seleção do modelo melhor validado e a identificação de problemas nos dados.

- Validação por Replicação: utilizando o modelo identificado, experimentos são repetidos para coletar dados da resposta da planta a diversas entradas ou condições operacionais. Essa abordagem compara as respostas reais da planta com as previsões do modelo para quantificar a variabilidade experimental e avaliar a confiabilidade dos modelos identificados (KEESMAN, 2011).

- Validação por Comparação com Modelos Alternativos: compara o modelo identificado com modelos concorrentes utilizando critérios como erro médio quadrático ou coeficiente de determinação. Essa técnica determina se o modelo selecionado oferece uma descrição estatisticamente superior do sistema em relação aos modelos alternativos (ISERMANN R. ; MÜNCHHOF, 2011). É aplicada quando existem várias abordagens possíveis para modelagem do sistema.

Existem vantagens e desvantagens associadas a cada método, e a escolha da abordagem mais adequada depende da natureza do problema em questão, bem como dos recursos disponíveis. Para um robô SLC, mesmo que métodos como a validação cruzada e a divisão em conjuntos de treinamento e teste sejam plenamente aplicáveis, a natureza prática e experimental da validação por replicação a torna particularmente adequada para a modelagem de sistemas dinâmicos como um robô seguidor de linha. Além disso, algumas ferramentas de identificação, como o *System Identification Toolbox* do MATLAB, possuem métodos de validação integrados, que podem se mostrar simples e eficazes no auxílio à validação de sistemas.

Em resumo, a identificação do sistema é um processo complexo que envolve a modelagem, a coleta de dados, a estimação de parâmetros e a validação do modelo. Com uma compreensão sólida desses princípios e técnicas, é possível desenvolver modelos precisos e confiáveis para controle de sistemas dinâmicos em uma ampla variedade de aplicações (ISERMANN R. ; MÜNCHHOF, 2011). À luz do estudo em questão, a identificação pode ser auxiliada por ferramentas e técnicas que facilitem o processo de identificação, desde que todas as etapas sejam acompanhadas atentamente e que os metódos sejam usados adequadamente.

2.5 CONCLUSÕES PARCIAIS

Neste capítulo, foram abordados uma série de aspectos fundamentais que compõem a base teórica para o desenvolvimento do controle de um robô seguidor de linha. Ao longo das diferentes seções, foram exploradas as seguintes áreas: uma análise das características e componentes de um robô seguidor de linha, um panorama das regras em competições de robôs seguidores de linha, uma discussão sobre os princípios básicos do controle de sistemas dinâmicos e uma exploração dos princípios de identificação do sistema, abrangendo métodos de modelagem, coleta de dados, estimação de parâmetros e validação de modelos. Essas partes integradas da fundamentação teórica oferecem uma compreensão abrangente dos conceitos e princípios necessários para o desenvolvimento bem-sucedido do controle de um robô seguidor de linha. No próximo capítulo, esses conhecimentos são aplicados no processo de identificação do robô SLC e no projeto do seu sistema de controle.

3 MONTAGEM, IDENTIFICAÇÃO E CONTROLE DO ROBÔ SLC

Neste capítulo, é apresentada a metodologia adotada para a realização deste trabalho, fundamentada nos conceitos discutidos no capítulo anterior, o desenvolvimento das etapas abordadas na metodologia e a apresentação dos resultados, que são necessários par ao andamento do projeto. São explorados em detalhes a montagem do robô SLC, o processo de identificação do sistema dessa montagem e a aplicação do controle para este sistema. Esta seção é essencial para compreendermos como os princípios teóricos discutidos anteriormente são traduzidos na prática, permitindo assim, avançar na implementação e no desenvolvimento do projeto, além de analisar de forma técnica e comparativa a eficácia das metodologias aplicadas e o desempenho do robô.

3.1 MONTAGEM DO ROBÔ SLC

A montagem do Robô SLC é dividida em categorias que desenvolvem separadamente tanto a parte mecânica quanto a eletrônica e a programação do robô. A equipe Maracatronics divide o trabalho desses escopos entre os membros, garantindo uma comunicação constante entre as equipes de mecânica, eletrônica e programação para garantir a eficiência e qualidade do processo de montagem. O projeto do robô SLC foi idealizado para atender a diversas modalidades e categorias de competição de Robótica de Seguidores de Linha seguindo os critérios estabelecidos no capítulo anterior e, como apresentado na introdução do trabalho, é produzido com base em um modelo anterior de 2018, o robô "Avexado", demonstrado na Figura 4.

3.1.1 Parte Mecânica

Na parte mecânica, os focos principais foram na escolha dos motores e das rodas. Foi responsabilidade da equipe maracatrônics a modelagem e impressão das peças da estrutura do robô (mancais, suporte às esferas de rotação livre e suporte de ligação à placa eletrônica, vulgarmente chamado d epescoço), feitas em impressão 3D usando filamento PLA (Ácido Polilático) devido à sua resistência e densidade em comparação a outros métodos. A principal atualização em relação ao robô base nessa parte foi quanto ao comprimento do robô, que antes era variável por meio de hastes deslizáveis, atualmente foi definido como um valor fixo definido pela equipe com base em desempenhos do modelo anterior. A equipe mecânica também ficou responsável pela construção das pistas de teste e prova.

3.1.2 Parte Eletrônica

Na parte eletrônica, foram escolhidos e dispostos os componentes eletrônicos, considerando os objetivos fracionados especificos do projeto e as limitações físicas impostas pelas

especificações mecânicas, bem como o projeto e confecção da placa de circuitos desse robô. Em relação ao projeto de 2018, as principais atualizações foram a substituição do microcontrolador STM32 por um microcontrolador ESP32, devido à sua compilação mais prática e à presença integrada de módulos *bluetooth* e *Wi-Fi*, que são úteis na etapa da coleta de dados, eliminando a necessidade de um módulo externo (ESPRESSIF SYSTEMS CO. LTD., 2024).

Outra atualização foi que os sensores individuais do Avexado foram substituídos por uma faixa de sensores integrados, simplificando a confecção da placa do robô e possíveis manutenções futuras. Neste momento, é importante salientar apenas a disposição dos sensores nesse módulo integrado, sendo oito sensores paralelos com 9,5 mm de distância entre si, operando na configuração analógica.

Além dessas modificações, detalhes da montagem que são necessários para o desenvolvimento do trabalho são sobre o *driver* baseado em um inversor ponte H monofásico, que aciona os motores a partir de um par de sinais para indicar o sentido de rotação e um sinal do tipo PWM (Modulador por Largura de Pulso - *Pulse Width Modulation*) para definir a velocidade dos os motores. Para este trabalho foram disponibilizados dois pares de micro motores de corrente contínua do tipo N20, motores relativamente comuns e de baixo custo, um par com uma relação de redução ajustada para uma velocidade máxima de 300 rpm (rotações por minuto) e outro par com uma velocidade máxima de 100 rpm. Apesar da escolha parecer simples, ela é melhor discutida na seção 3.3.2.2. Esses motores estão equipados com *encoders* baseados em sensor Hall, permitindo a contagem precisa das rotações e fornecendo ao controlador uma forma de medir a velocidade dos motores (CHI HAI MOTOR CO. LTD., 2020).

3.1.3 Programação

Na programação, a ênfase recai sobre a criação dos códigos para implemetação no microcontrolador. Esses códigos são necessários para realizar rotinas importantes para o projeto do robô SLC, como teste de componentes, processos de funcionamento, coleta de dados para a identificação do modelo e aplicação do controle, dos quais os últimos dois temas são tradados com mais detalhes em seções posteriores.

Os códigos foram elaborados e compilados na IDE do Arduino com suporte a ESP para *Windows*, em linguagem C++ pela familiaridade com a linguagem e sua aplicação em programação orientada a objeto. Antes da placa ser prototipada, é necessário testar todos os componentes na sua atual disposição em protoboard, utilizando códigos apropriados para extrair e avaliar informações como velocidade e qualidade dos componentes. Para a comunicação entre computador e robô, são utilizados dois modos: comunicação via cabo para compilação de códigos e testes em bancada local e comunicação via *bluetooth* para coleta de dados remotos, evitando possíveis interferências do *Wi-Fi* em portas de leitura de alguns sensores.

Os principais códigos de teste utilizados nas etapas subsequentes incluem interrupções periódicas para a coleta de dados, comunicação sem fio, estimativa da velocidade dos motores através dos sensores de efeito Hall, e controle de chaveamento dos motores em operações específicas.

A conversão A/D (Analógico/Digital), a calibração dos sensores e a tratativa das velocidades dos motores também são pontos importantes sobre a programação, sendo explicados com mais detalhes na próxima seção.

3.1.4 Tratamento dos Sensores e Motores

Como os sensores e motores são a interface do microcontrolador com o ambiente externo, é preciso tratar esses componentes com cautela e avaliar a melhor forma de integrá-los à luz do contexto deste trabalho. Para isso, foram adotadas estratégias para o tratamento dos sensores e motores do robô seguidor de linha a fim de garantir um desempenho adequado e seguro do sistema. Foram realizadas ações específicas para padronizar a leitura dos sensores, incluindo a sua calibração e integração com o conversor A/D, e a resposta dos motores, especialmente quanto à conversão da velocidade linear em velocidade angular.

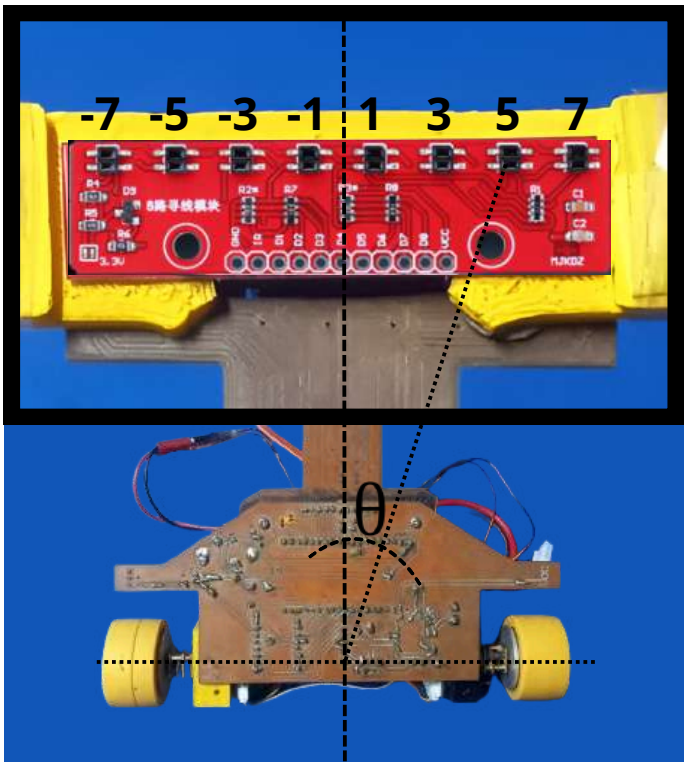
3.1.4.1 Sensores

Quanto à leitura dos sensores, o método de tratamento se dá em uma conversão dos valores de tensão equivalente à luminosidade refletida nos terminais de leitura do microcontrolador em valores discretos para a posição relativa do robô à linha, essa conversão A/D está intimamente relacionada à rotina de calibração, que consiste em um ciclo para excitar todos os sensores várias vezes ao passar pela linha e pela parte externa a ela, garantindo que todos os sensores tenham experimentado valores semelhantes aos máximos e mínimos de refletância encontrados durante todo o percurso. Com base nesses valores, calculam-se limiares para cada sensor usando estimativas para definir um valor seguro para a transição da reflexão do percurso. Com os limiares calibrados, o microcontrolador pode determinar quais sensores estão ativos sobre a linha e, com base nisso, definir um valor para a posição relativa entre o robô e a linha, dessa forma, convertendo o conjunto de valores referentes às luminosidades refletidas de cada sensor em um vetor digital de sensores ativos e, depois disso, em um valor digital que reflete a posição do robô em relação à linha.

O arranjo dos sensores na faixa frontal do robô é disposto de forma que apenas dois ou três sensores visualizem a linha ao mesmo tempo em partes normais do trajeto. Isso permite mensurar um valor de posição com base nos sensores ativos no momento após a conversão A/D, do qual são retirados os valores da variável de controle. Também é possível configurar o valor da posição levando em consideração o ângulo médio formado entre os sensores ativos e o centro do eixo dos motores, θ. A disposição dos sensores, seus

valores em relação ao centro do robô e detalhes visuais da configuração angular podem ser vistos na Figura 12. A partir da representação visual da Figura 12, é possível determinar o ângulo θ que cada sensor possui com relação ao eixo de simetria do robô, cujos valores em graus estão representados na Tabela 2.

Figura 12 – Definição das posições lineares e angulares a partir da combinação dos sensores ativos.



Fonte: Autor.

Os valores dos pares dos extremos da Tabela 2 tiveram seus valores calculados a parte, levando em consideração a espessura da linha e adotando critérios de pertencimento. Os mesmos critérios foram usados para definir a posição dos penultimos valores, visto que nessas condições apenas um único sensor estará sobre a linha, o que leva a uma variação de angulação mais brusca da penúltima posição para a antepenúltima, do que dela para a última.

3.1.4.2 Motores

A movimentação do robô é diretamente controlada pelos motores, sendo essenciais para seu deslocamento ao longo da linha de forma precisa e consistente. Como a variável de leitura dos sensores é uma grandeza angular, é necessário converter, de alguma forma, a velocidade linear das rodas do robô em uma velocidade angular de rotação do robô. No contexto do projeto, foi definido que, para um melhor desempenho, os motores devem ficar em suas velocidades máximas quando o robô já estiver centralizado com a faixa do percurso. A velocidade máxima, para o caso tratado, é quando o PWM do *driver* permite

Tabela 2 – Posições por combinação de Sensores.

Sensores Ativos	**Posição Linear**	**Posição Angular θ**
Nenhum, mas o último foi o -7	-8	-12,08°
-7	-7	-10,01°
-7 e -5	-6	-7,26°
-7, -5 e -3	-5	-6,06°
-5 e -3	-4	-4,85°
-5, -3 e -1	-3	-3,64°
-3 e -1	-2	-2,43°
-3, -1 e 1	-1	-1,22°
-1 e 1	0	0,00°
-1, 1 e 3	+1	+1,22°
1 e 3	+2	+2,43°
1, 3 e 5	+3	+3,64°
3 e 5	+4	+4,85°
3, 5 e 7	+5	+6,06°
5 e 7	+6	+7,26°
7	+7	+10,01°
Nenhum, mas o último foi o 7	+8	+12,08°

Fonte: Autor

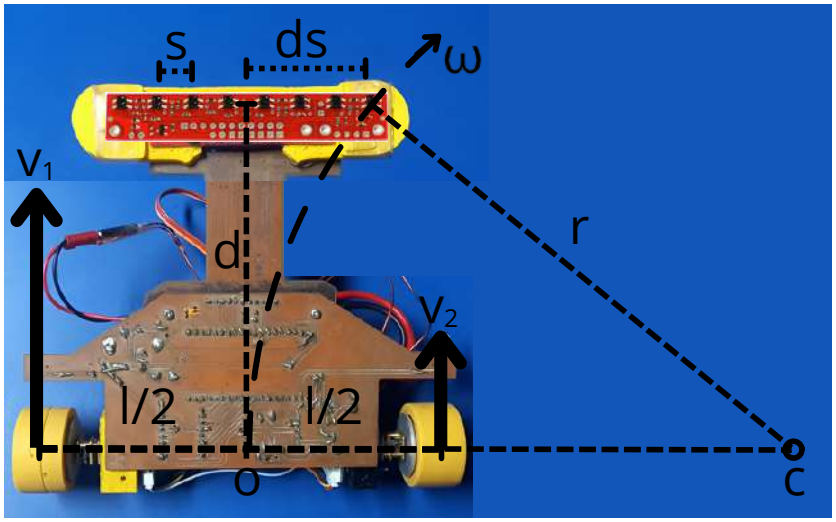
a passagem integral da tensão da bateria para o motor e é descrita em valor digital por 255. Para rotacionar o robô para um dos lados, é possível manter a velocidade máxima do motor que não realizará o pivô e frenar, em uma proporção controlada, o motor que irá realizar o movimento de pivô. Assim, a diferença de velocidades lineares entre as rodas (o motor que está na velocidade máxima e o que está com uma velocidade abaixo da máxima) indicará o quão inclinado o robô fará a curva. Os cálculos podem ser feitos com base nas dimensões do robô, representadas na Figura 13, e nas leis físicas conforme apresentado na equação a seguir:

$$\omega = \frac{v_1}{r + l/2} = \frac{v_2}{r - l/2} \tag{3.1}$$

Onde ω é a velocidade angular de giro do robô, v_1 e v_2 são as velocidades lineares das rodas mais afastada e mais próxima do centro da curva, respectivamente, l é a distância entre os centros das rodas do robô, r é o raio da curva, que é a distância do centro da curva e do centro do eixo dos motores, que na Figura 13 são representados por c e o, respectivamente. Isolando a variável r nas duas equações em (3.1) e subtraindo uma equação resultante da outra, obtém-se a seguinte equação:

$$\omega = \frac{v_1 - v_2}{l} \tag{3.2}$$

Figura 13 – Representação da rotação do robô em função da frenagem de um dos motores.



Fonte: Autor.

A equação (3.2) indica que a velocidade de rotação de giro do robô é diretamente proporcional à diferença de velocidades lineares das duas rodas. Essa velocidade ω, com sentido convencionado em função da Figura 13, será positiva (rotação no sentido horário) se o motor 2 (motor direito) frenar, ou seja, se $v_2 < v_1$; e será negativa (rotação no sentido anti-horário) se o motor 1 (motor esquerdo) frenar, ou seja, $v_1 < v_2$. Desta forma, não apenas podemos transformar o diferencial de velocidade linear em uma componente angular, representada por ΔV, mas também é possivel definir uma velocidade de rotação com base em uma alteração percentual em apenas um dos motores, consideração essa que é explicada com maiores detalhes na próxima seção.

Para facilitar a visualização das grandezas quando colocadas lado a lado, ao custo de uma simples conversão proporcional, é possível normalizar tanto a posição θ, que é o valor de entrada do controlador, quanto a ação de velocidade diferencial dos motores ΔV, que é o valor da saída do controlador, em porcentagem de seus valores máximos, 12,08° para θ e 255 para ΔV, assim criando as variáveis respectivas Θ_{pu} e $DeltaV_{pu}$, atuando em uma faixa igual para ambos dados, de -100% a 100%, atentando-se que a saturação é necessária durante a aplicação.

3.2 RESULTADOS DA MONTAGEM

A montagem do robô SLC envolveu a integração de componentes mecânicos e eletrônicos, selecionados com base em critérios de robustez, precisão e disponibilidade, conforme descrito anteriormente. A Tabela 3 lista os principais componentes utilizados e suas principais características, e foi montada a partir do seguinte detalhamento dos componentes:

- Sensores de Linha: foram escolhidos sensores de refletância do tipo IR (infraver-

Tabela 3 – Componentes utilizados na montagem do robô SLC.

Componente	**Modelo**	**Especificações**	**Imagem**
Sensores de Linha	QTR-8A (POLOLU)	8 sensores IR, < 100 mA	Figuras 14 e 15
Motores	N20 com *encoder* Hall	100 ou 300 rpm, 2 kgf.cm, 70 mA, 15 g	Figura 16
Microcontrolador	ESP32 wroom 32d	10 ADC, 9 GPIO, 2 PWM	Figura 17
Driver de Motores	TB6612FNG	$\doteq$200 μs de atraso	Figura 18
Bateria	Turnigy nano-tech airsoft	LiPo 3S, 11.1V, 1000 mAh	Figura 19
Estrutura Mecânica	Impressa pela Maracatronics	173 mm x 215 mm, 97 g	Figura 20

Fonte: Autor

melho) do módulo QTR-8A da POLOLU, compostos por um par de um Led IR e um fototransistor em cada, eles são precisos e consomem pouca energia, visto que os oito sensores ativos em sua configuração paralelo demandam menos de 100 mA, ideal para o projeto do SLC. O módulo usado pode ser visto na na Figura 14 e um diagrama esquemático desses sensores podem ser vistos na Figura 15 (POLOLU, 2014).

Figura 14 – Módulo de sensores QTR-8A.

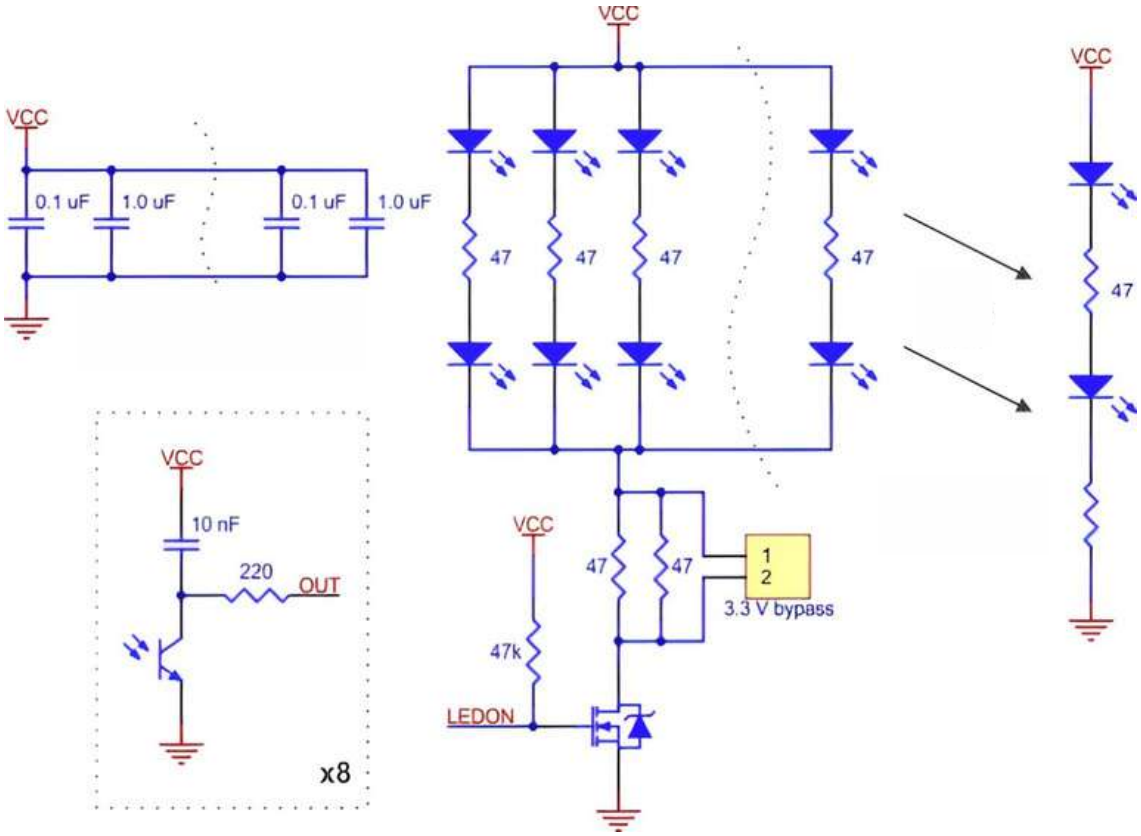
Fonte: (POLOLU, 2014).

- Motores: os motores disponibilizados foram do tipo N20, de 100 ou 300 rpm, com um *encoder* baseado em sensor Hall, apresenta um torque de 2 kgf.cm, pesa cerca de 15g e em condições máximas apresenta uma corrente de 70 mA com uma potência máxima de 0,9W. Um desses motores está representado na Figura 16 (CHI HAI MOTOR CO. LTD., 2020).

- Microcontrolador: o microcontrolador ESP32 wroom 32d foi escolhido devido ao seu desempenho, velocidade e número de portas que, para esse projeto, são usadas 10 portas analógicas e 9 portas digitais, sendo 2 dessas do tipo PWM. O modelo esquemático desse tipo de controlador está detalhado na Figura 17 para uma melhor compreensão (ESPRESSIF SYSTEMS CO. LTD., 2024).

- *Driver* de Motores: o módulo de ponte-H do tipo TB6612FNG foi escolhido pela sua disponibilidade e baixo custo, ele pode controlar até dois motores com base em entradas digitais que definem o sentido de rotação e a velocidade. Testes revelaram que o tempo de resposta desse *driver* é da ordem de 200 μs, tal *driver* pode ser visualizado na Figura 18 (TOSHIBA, 2007).

Figura 15 – Diagrama esquemático do módulo de sensores QTR-8A.



Fonte: Adaptado de (POLOLU, 2014).

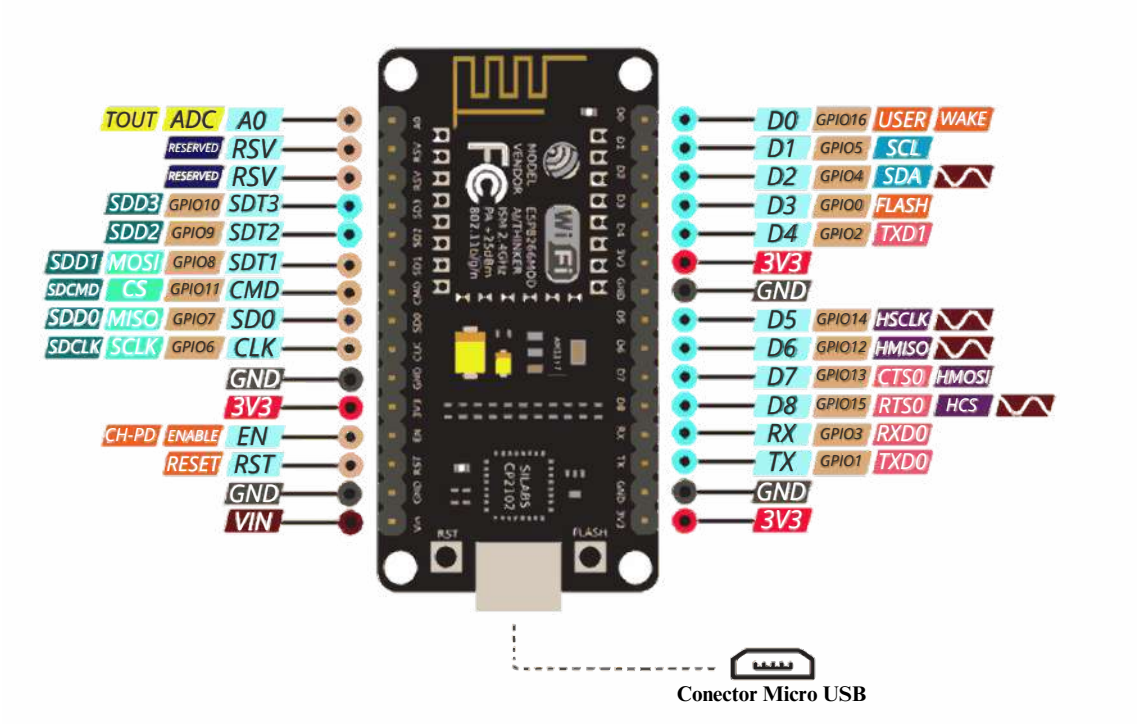
Figura 16 – Motor N20 equipado com sensor Hall.

Fonte: Adaptado de (CHI HAI MOTOR CO. LTD., 2020).

- Bateria: a energia é fornecida por uma bateria LiPo (Polímero de Lítio) de 3 células, 11,1V da turnigy que fornece até 1000 mAh, com uma alta velocidade de descarga, fazendo os motores trabalharem em seu potencial máximo para as velocidades definidas. Ela está representada na Figura 19 (TURNIGY POWER SYSTEMS., 2024).

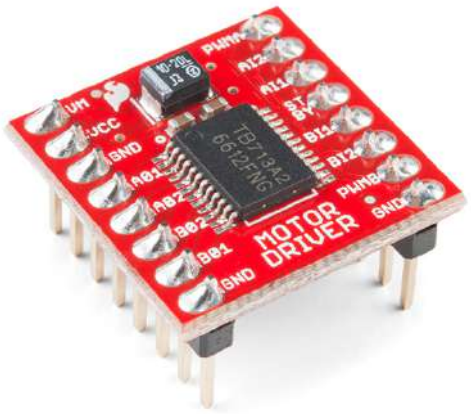
- Estrutura Mecânica: uma estrutura leve e resistente feita de impressão em filamento PLA e placa de fenolite proporcionam estabilidade e segurança ao robô, funcionando como chassi para os componentes anteriormente citados.

Figura 17 – Esquemático do microcontrolador ESP32.



Fonte: Adaptado de (ESPRESSIF SYSTEMS CO. LTD., 2024).

Figura 18 – Módulo do *driver* de ponte-H, TB6612FNG.



Fonte: (TOSHIBA, 2007).

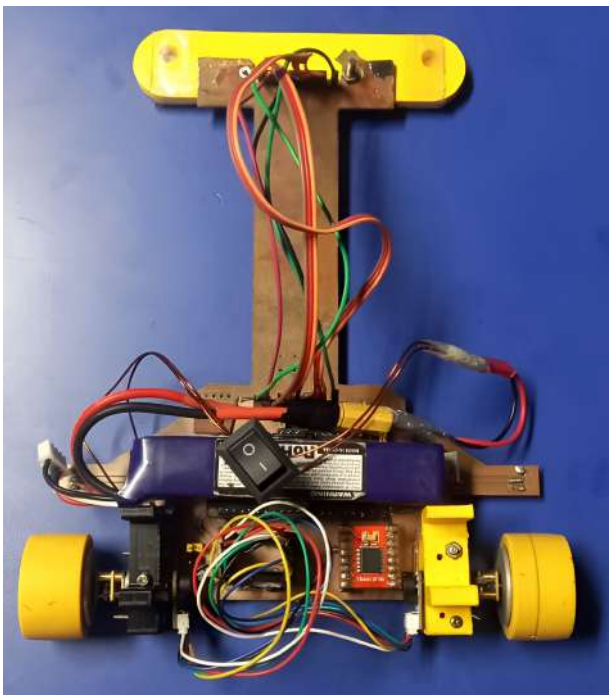
Além dos componentes acima listados, reguladores de tensão, resistores, chaves, bornes e esferas de rotação livre completam a montagem do robô. A montagem completa do robô pode ser visualizada na Figura 20 e o projeto eletrônico, na Figura 21.

Após a montagem, foram realizados testes iniciais para verificar a operação básica dos componentes eletrônicos do robô, obtendo resultados de que tais peças estavam respondendo adequadamente aos comandos e apresentando os comportamentos esperados.

Figura 19 – Bateria Turnigy nano-tech airsoft.

Fonte: (TURNIGY POWER SYSTEMS., 2024).

Figura 20 – Montagem do "Avexadinho".

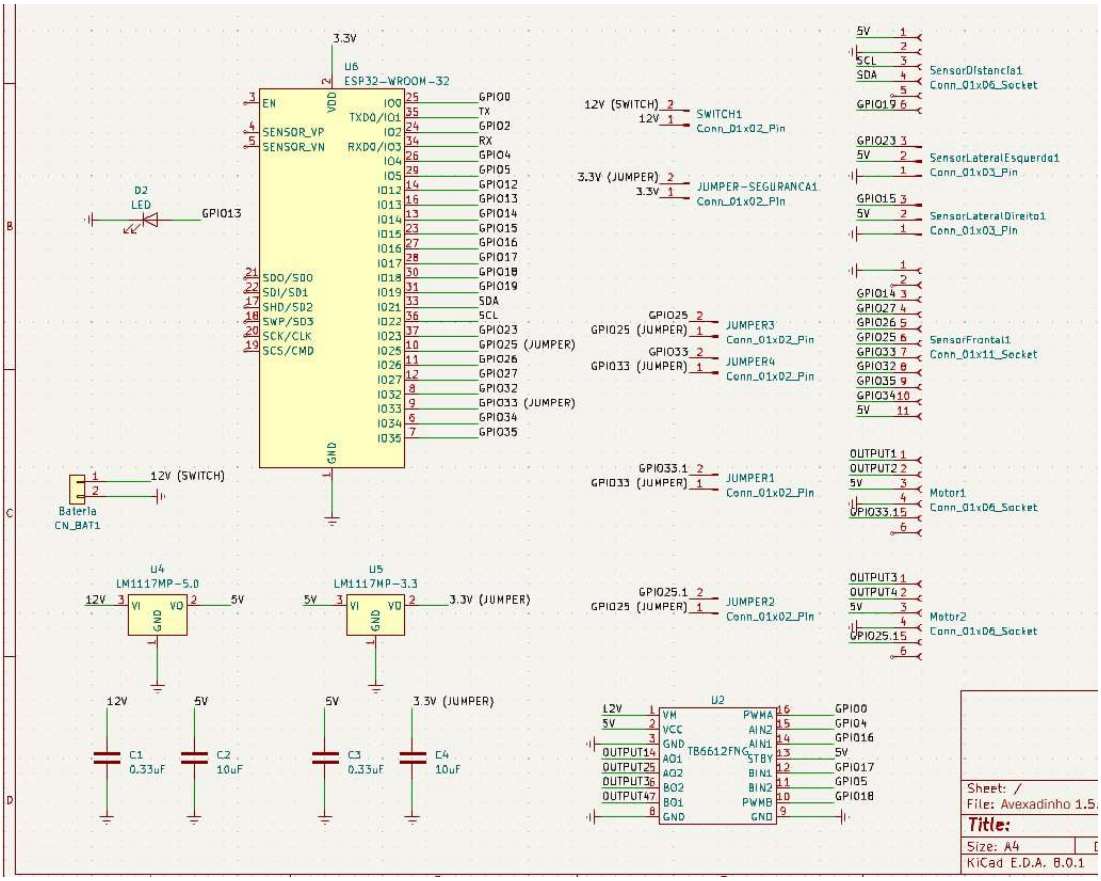


Fonte: Autor.

O principal problema encontrado em relação ao projeto inicial da montagem foi quanto à impressão de parte da estrutura mecânica, que não pôde ser impressa a tempo dos testes e elaboração do trabalho e, por questão de compatibilidade, foi substituída por uma placa de fenolite sem aplicação eletrônica.

Os sensores testados apresentaram uma faixa de atuação com valores digitais no controlador variando entre 200 e 1500 ao refletirem o caminho branco e um valor de 4095 ao refletirem o caminho preto. Esses valores foram inicialmente utilizados para calibrar os sensores durante os testes. Contudo, para os resultados finais, foi incluída uma rotina de calibração que ocorre antes do início da cronometragem do trajeto pelo robô. Essa

Figura 21 – Projeto eletrônico do "Avexadinho".



Fonte: Autor.

rotina permite um ajuste mais preciso para contornar possíveis problemas de desgaste ou variações na luminosidade.

A montagem dos componentes, mecanicamente resultou em uma estrutura sólida, resultando em um robô SLC com 173 mm de largura, 215 mm de comprimento e 280 g de massa, apresentou uma potência máxima próxima a 1,8 W, seu código ficou com um tempo médio de execução de 216 μs, assim estando apto para particiar de competições nessa categoria, com características de montagem adequadas. Para concluir a montagem, a programação foi quase inteiramente reescrita. Um fluxograma correspondente pode ser visualizado na Figura 22, onde cada bloco está resumido a seguir:

- Rotina de calibração: é a etapa inicial do robô, antes de qualquer tomada de tempo, um rotina de calibração automática é realizada para que cada sensor seja calibrado individualmente para as condições daquele percurso.

- Leitura dos sensores: é o início do ciclo de repetição, aqui os sensores são lidos e, com base na calibração, um vetor binário referente ao pertencimento dos sensores sobre a linha é criado.

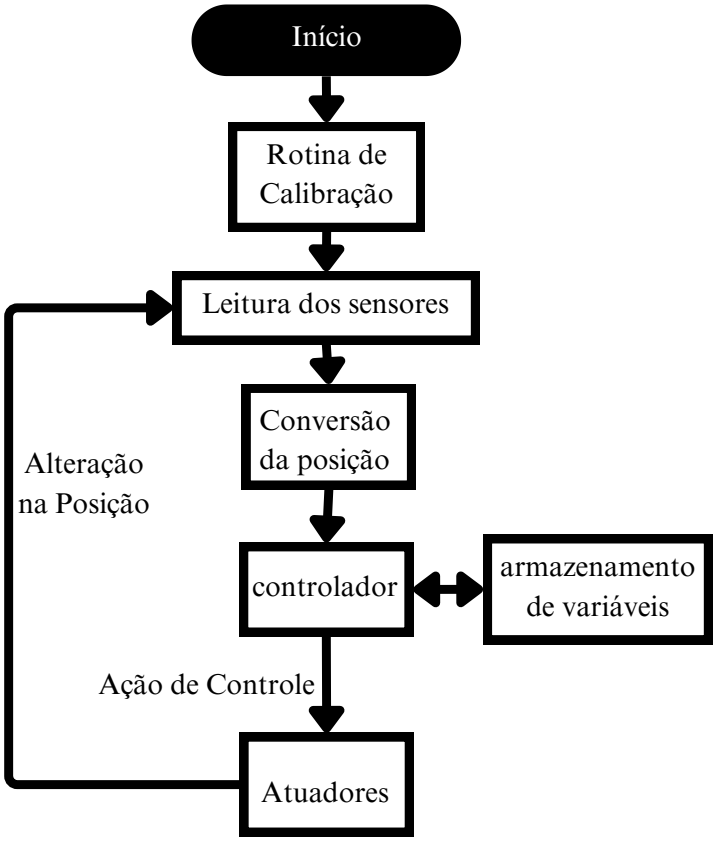
- conversão da posição: com base na configuração do vetor criado na etapa anterior,

ocorre uma conversão dos sinais em um valor de posição com relação a linha, que varia de -100 a +100 em 17 estados quantizados.

- Controlador: com base na posição atual e variáveis anteriores (posições e ações de controle), o controlador calcula uma nova ação de controle a ser acionada nos atuadores e armazena as variáveis atuais.
- Armazenamento de variáveis: é responsável por entregar ao controlador as variáveis anteriores e armazenar as variáveis atuais para serem utilizadas no próximo ciclo.
- Atuadores: Por fim o controlador deve acionar o *driver* dos motores conforme ação definida pelo controlador a fim de causar uma alteração na posição pelo movimento dos motores.

Esse código em si podem ser localizados no Apêndice A deste trabalho, porém até esse ponto, as constantes do controlador, apresentado nas linhas 111 e 117, não haviam sido definidas, assim como o tempo de aplicação do controle digital, representado pelo intervalo da interrupção na linha 128.

Figura 22 – Fluxograma da programação do "Avexadinho".



Fonte: Autor.

Com os componentes eletrônicos e mecânicos definidos e integrados, a programação base modelada e as considerações fundamentais realizadas para o tratamento dos sensores

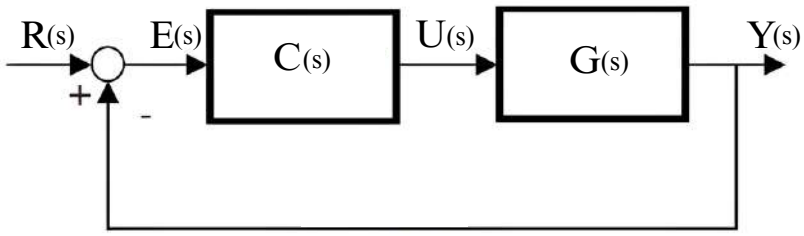
e motores, a atenção do projeto agora se volta para a próxima etapa: a identificação do sistema.

3.3 IDENTIFICAÇÃO DO SISTEMA DO ROBÔ SLC

Nesta seção, é apresentado um guia detalhado para a identificação do sistema de um robô seguidor de linha, visando a obtenção de um modelo matemático preciso que represente seu comportamento dinâmico. O modelo é utilizado posteriormente para o projeto e ajuste do controlador, garantindo um desempenho otimizado do robô SLC. O sistema a ser modelado pode ser representado pelo diagrama de blocos na Figura 23. Neste diagrama, o modelo matemático da dinâmica do robô é descrito por $G(s)$, $U(s)$ representa a ação de controle, correspondente ao diferencial de velocidades angulares dos motores, ΔV. A saída do sistema, $Y(s)$, saída do sistema, refere-se à posição angular de giro do robô, Θ. $R(s)$ é a referência a ser seguida e $E(s)$ denota o erro de posição. O controlador é representado por $C(s)$ e será projetado nas seções subsequentes. Todas estas variáveis pertencem ao domínio de Laplace do plano s.

Geralmente, em diagramas de blocos desse tipo, é comum incluir um bloco adicional no caminho do *feedback* para representar a dinâmica do sensor do sistema, representado por $H(s)$. No entanto, de acordo com as considerações feitas na seção 3.1.4.1, é possível considerar que esta dinâmica está incorporada na planta $G(s)$, assumindo que o sensor $H(s)$ tem valor unitário.

Figura 23 – Diagrama de blocos simplificado do robô SLC.



Fonte: Adaptado de (LANDAU; ZITO, 2006).

3.3.1 Modelagem do Sistema

A primeira etapa da metodologia na identificação do sistema é a definição do modelo a ser adotado, e pela dificuldade da representação física do robô com todas leis físicas que o englobam, como aerodinâmica, variação de momento angular e linear, variação do coeficiente de atrito das rodas com a pista, reflatância da luz, entre outras, o uso da abordagem baseada em caixa preta ou cinza é favorecido. Por definição do trabalho, o modelo a ser adotado é o de caixa preta. Este modelo representa a relação entre a entrada e a saída do sistema sem revelar sua estrutura interna e, dentro da abordagem de

caixa preta, um modelo de regressão linear é utilizado para representar a relação entre o diferencial de velocidade linear das rodas (entrada da planta), ΔV, e a variação da posição angular de giro do robô (saída da planta), Θ. Este modelo é simples de implementar e interpretar, e se mostrou adequado em estudos de sistemas semelhantes (KEESMAN, 2011).

3.3.2 Coleta de Dados

A coleta de dados é uma etapa fundamental que envolve a realização de experimentos controlados nos quais entradas conhecidas são aplicadas ao sistema e as saídas correspondentes são registradas. Nesta seção, é descrito o procedimento de coleta de dados, detalhando a definição dos parâmetros de entrada, a descrição dos experimentos e o armazenamento dos dados obtidos.

3.3.2.1 Parametrização da Entrada

A parametrização adequada da entrada dos degraus de teste é importante para garantir a obtenção de dados significativos e representativos do comportamento do robô seguidor de linha. As entradas de teste devem ser cuidadosamente escolhidas para abranger uma variedade de condições operacionais e desafios encontrados pelo robô durante o percurso da linha.

Considerando a natureza do sistema e os objetivos da identificação, optou-se por utilizar degraus de teste como entradas de referência para o robô. Um degrau de teste consiste em uma mudança súbita e definida na velocidade linear de um dos motores, que gerará uma diferença de velocidade angular do robô (ω), que é aplicada em um determinado instante de tempo durante o percurso da linha.

É importante considerar a amplitude e a duração dos degraus de teste para garantir uma cobertura adequada do espaço de entrada e capturar as dinâmicas do sistema. Além disso, a frequência de aplicação dos degraus de teste deve ser suficiente para permitir a estimativa precisa dos parâmetros do modelo.

Para definir valores para os degraus de teste, é necessário considerar o tratamento dado aos motores quanto à conversão da variável de controle. Para isso, é necessário voltar à seção 3.1.4.2, que descreve equações que convertem a entrada linear relativa entre as rodas em uma velocidade angular do robô. Aqui é necessário fazer o processo inverso, com base em um conjunto de velocidades angulares, definir um conjunto de velocidades lineares para uma roda. Esse conjunto de velocidades são os valores a serem avaliados para uso nos degraus de entrada.

A partir da equação (3.1), pode-se extrair a seguinte equação para calcular o raio da curva com base nas velocidades dos motores:

$$r = \frac{(v_1 + v_2) \cdot l}{(v_1 - v_2) \cdot 2} \tag{3.3}$$

Levando em consideração a Figura 13, e usando de trigonometria para calcular as distâncias dos sensores para o centro do eixo dos motores e para o centro do raio da curva, é possível calcular uma velocidade angular imediatamente necessária para cada sensor, com base no raio da curva ideal, para o centro dos motores corresponder a posição atual do sensor, sendo representado pelas seguintes equações:

$$r^2 = d^2 + (r - ds)^2 \tag{3.4}$$

$$r = \frac{ds^2 + d^2}{2 \cdot ds} \tag{3.5}$$

Onde d é a distância do eixo dos motores até a faixa dos sensores, ds é a distância da posição do sensor avaliado até o centro da faixa de sensores, onde o referencial da posição é 0. Unindo as Equações (3.5) e (3.3) e considerando que uma velocidade sempre está no máximo e a outra é uma proporção f desse valor máximo, onde f está entre 0 e 1, e que ds pode ser representado como $np \cdot ls/2$, onde np é a posição linear avaliada e ls é a distância entre dois sensores seguidos (FITZGERALD; AL, 1975). É possível chegar às seguintes equações:

$$\frac{ds^2 + d^2}{ds} = \frac{(1 + f) \cdot l}{(1 - f)} \tag{3.6}$$

$$f = \frac{ds^2 + d^2 - ds \cdot d}{ds^2 + d^2 + ds \cdot d} \tag{3.7}$$

$$f = \frac{(np \cdot ls/2)^2 + d^2 - np \cdot ls \cdot d/2}{(np \cdot ls/2)^2 + d^2 + np \cdot ls \cdot d/2} \tag{3.8}$$

Substituindo o valor de np em (3.8) por cada uma das posições, é possivel encontrar a relação $f = v_2/v_1$, para v_1 em 100% de velocidade, e v_2 sendo um valor decrescido dessa razão, desde que entre 0% e 100%. Para as posições positivas (+), os resultados mantêm v_2 em 100%, bastando replicar os valores decrescidos para v_1, de forma que o resultado f fica invertido, resultando assim em possíveis parâmetros de entradas para testes. Usando essa fórmula é possivel criar um conjunto de posições relativas às posições dos sensores sobre a linha com as velocidades instantâneas percentuais para cada motor conforme a Tabela 4, onde as velocidades estão dispostas em termos de frenagens proporcionais à velocidade máxima.

Com os valores de velocidade parametrizados para as possíveis entradas, torna-se necessário definir os detalhes adicionais para a realização dos experimentos de coleta de dados, conforme é apresentado na seção subsequente.

3.3.2.2 Experimentos Para Coleta de Dados

O procedimento de coleta de dados consiste na execução dos experimentos planejados, nos quais os degraus de teste são aplicados ao robô seguidor de linha em uma pista com as mesmas propriedades da pista final e as saídas correspondentes a uma resposta da planta em malha aberta são registradas. Abaixo, são descritas as etapas envolvidas no procedimento de coleta de dados:

Tabela 4 – Percentual de frenagem dos motores em função das posições lineares do conjunto de sensores.

Posição Linear	**Frenagem percentual do motor 1**	**Frenagem percentual do motor 2**
-8	0,0%	23,7%
-7	0,0%	21,2%
-6	0,0%	18,5%
-5	0,0%	15,8%
-4	0,0%	12,9%
-3	0,0%	9,8%
-2	0,0%	6,7%
-1	0,0%	3,4%
0	0,0%	0,0%
+1	3,4%	0,0%
+2	6,7%	0,0%
+3	9,8%	0,0%
+4	12,9%	0,0%
+5	15,8%	0,0%
+6	18,5%	0,0%
+7	21,2%	0,0%
+8	23,7%	0,0%

Fonte: Autor

- **Preparação do Robô:** o robô seguidor de linha é devidamente configurado e preparado para a realização dos experimentos. Isso inclui a calibração dos sensores de linha, a verificação da integridade dos componentes mecânicos e eletrônicos, e a configuração dos parâmetros de controle.

- **Execução dos Experimentos:** os experimentos são realizados em um ambiente controlado, onde o robô é colocado em uma pista de teste contendo uma linha branca sobre um fundo preto. Um degrau de velocidade máxima é aplicado a ambos motores de modo que a componente angular do RSL seja nula, ΔV=0, então, decorrido um tempo suficiente para se atingir a velocidade máxima em regime permanente, um degrau de teste é aplicado em um dos motores e as saídas do sistema são registradas ao longo do tempo. Outro ponto importante é quanto à quantidade de ensaios a serem realizados, visto que, uma quantidade pequena de testes apresenta um grande risco quanto à confiabilidade e viés do experimento, enquanto que uma quantidade elevada pode significar trabalho redundante.

- **Registro dos Dados:** durante a execução dos experimentos, os dados correspondentes à entrada (ação da velocidade linar relativa, que gerará uma velocidade angular de referência, ΔV_{pu}) e à saída (variação de posição angular do robô conforme detectado pelos sensores, Θ_{pu}) são registrados continuamente utilizando um tempo de amostragem adequado, definido com base na dinâmica dos motores. Os dados

são enviados via *bluetooth* e armazenados em um formato adequado para análise posterior.

A pista utilizada nos testes está representada na Figura 24.

Figura 24 – Pista usada para a coleta de dados.

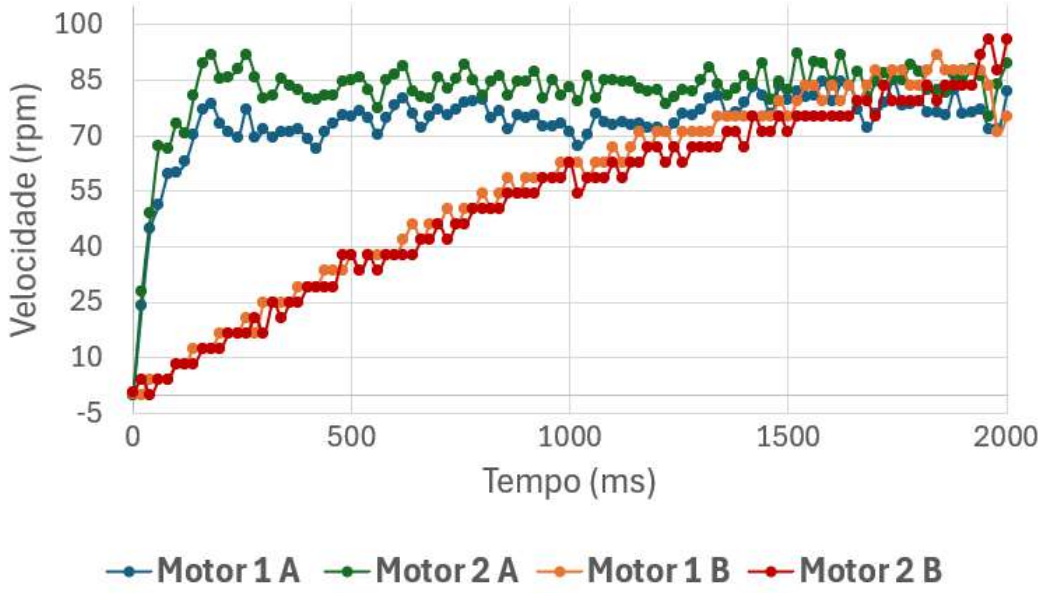
Fonte: Autor.

Antes de começar a coleta definitiva dos dados, três decisões importantes devem ser tomadas: quantas e quais provas precisam ser feitas, qual tempo de amostragem usar e qual dos motores disponibilizados apresenta características melhores para o controlador. Os motores que foram disponibilizados, conforme introduzidos na seção 3.3.2.2, são um par de motores de aproximadamente 100 rpm (motor A) e outro de aproximadamente 300 rpm (motor B), ambos com a mesma potência nominal. Usando os *encoders* dos motores para obter dados que demonstrassem a dinâmica de sua resposta a um degrau de amplitude máxima, as escolhas de motor e tempo de amostragem podem ser tomadas com base em um teste único.

Após realizado o teste mencionado para ambos os motores, foram obtidos os gráficos da Figura 25. Observa-se que a pista de prova usada não foi grande o suficiente para que o motor B atingisse sua velocidade nominal, devido ao seu baixo torque. Por outro lado, o motor A, por ter um torque mais elevado, conseguiu atingir o regime permanente mais rapidamente. Como o robô SLC operará em curvas rápidas e trechos curtos, o motor que possui maior torque e menor rotação nominal (motor A) é o mais adequado para esta função, atingindo o regime permanente após 400 ms.

A partir do tempo de acomodação de 400ms, é possível definir que a constante de tempo da dinâmica dominante do sistema é próxima aos 80ms (um quinto do tempo de acomodação). Assim, pelo teorema de Nyquist, é importante que o tempo de amostragem não seja superior a 40ms (metade da constante de tempo). Testes posteriores mostraram que um tempo de amostragem de 5 ms é mais adequado, pois permite a captação das

Figura 25 – Gráficos da dinâmica da aceleração do robô para dois tipos de motor.



Fonte: Autor.

grandezas durante variações bruscas que ocorrem durante curvas, manobras e possíveis perturbações da pista.

Sobre a quantidade e seleção de ensaios a serem feitos, os valores escolhidos para teste seriam referentes às posições de cada sensor centralizado na faixa branca da pista, resultando em oito ensaios. Porém, como os valores de frenagem demonstrados na Tabela 4 estão muito próximos, se limitar a esses dados pode apresentar um risco de enviesar o sistema a atuar em uma faixa menor que a de saturação, além de que apresentaria resultados muito próximos uns aos outros, onde um pequeno disturbio ou ruído de medição poderia comprometer a integridade do modelo identificado. Então, os ensaios foram escolhidos também com base na faixa de saturação do motor, resumindo-se em quatro situações de cada lado: frenagens referentes às posições extremas da Tabela 4, que são valores teóricos ideais para alinhamento dos casos mais internos e externos da faixa de sensores, mas como foram calculados com uma ótica espacial e imediata, desprezam conceitos físicos básicos como aceleração e conservação de movimento, para garantir um cenário mais realista, levando em conta as característica dos componentes, outras duas opções foram escolhidas para avaliar uma atuação maior e ter um menor enviesamento do robô SLC nos ensaios. Essas opções adicionais foram referentes à velocidades de frenagem total e de 50%. Um apanhado total dos parâmetros de entrada para os ensaios estão apresentados na Tabela 5. Nota-se que as frenagens percentuais foram alteradas em relação a seus valores de ensaio, isso se deve à quantização do controlador, que não consegue expressar os valores em uma faixa analógica. O Código usado para a coleta de dados está no apêndice B.

Após coletados, os dados precisam ser armazenados corretamente para a posterior identificação do sistema.

Tabela 5 – Ajustes do ensaios.

Entrada do ensaio (ΔV_{pu})	**Frenagem percentual do motor 1**	**Frenagem percentual do motor 2**
-100%	0,0%	100%
-50%	0,0%	50,2%
-23,7%	0,0%	23,9%
-3,4%	0,0%	5,1%
+3,4%	5,1%	0,0%
+23,7%	23,9%	0,0%
+50%	50,2%	0,0%
+100%	100%	0,0%

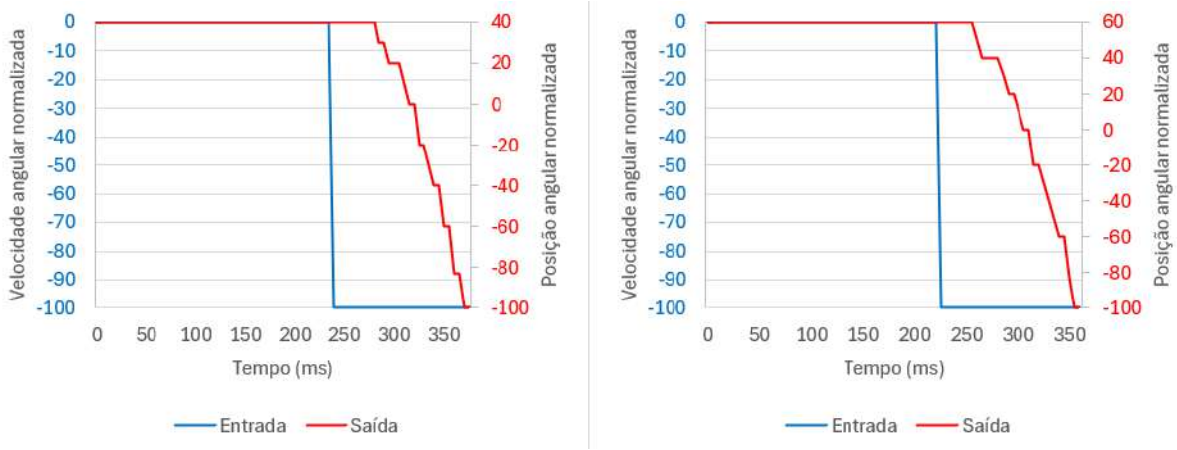
Fonte: Autor

3.3.2.3 Armazenamento dos Dados

Os dados coletados durante os experimentos são armazenados de forma organizada e segura para facilitar a análise e o processamento posterior. Eles são armazenados em formato digital, utilizando planilhas do Excel para facilitar a aplicação dos dados no *softaware* MATLAB, e ajudando na elaboração de gráficos. Cada conjunto de dados deve ser devidamente rotulado e documentado, incluindo informações sobre as condições experimentais, os parâmetros de entrada utilizados e quaisquer observações relevantes sobre o comportamento do sistema.

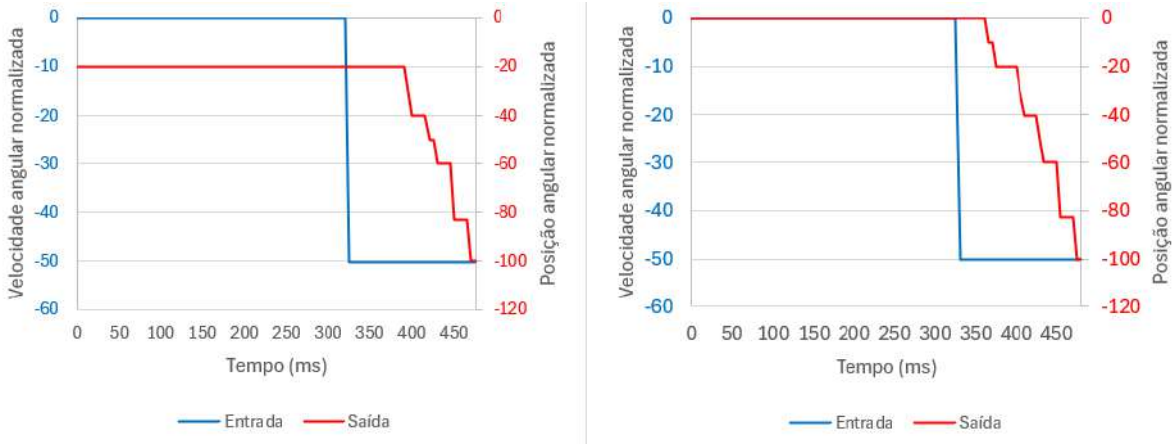
Os testes dos oito ensaios descritos no final da seção anterior foram realizados e para cada degrau de velocidade, com os dados coletados, foram obtidos os gráficos da diferença de velocidade angular pela posição angular, representados nas Figuras de 26 a 33, e gravados vídeos de cada prova deste experimento. Os vídeos de exemplos para cada teste podem ser encontrados no *Youtube*, como visto na Tabela 6. O apêndice C contém as planilhas completas dos ensaios.

Figura 26 – Gráficos da velocidade diferencial, ΔV_{pu} (entrada), e a posição angular do robô, Θ_{pu} (saída), para um degrau de ΔV_{pu} = -100%: Ensaio 1 (gráfico da esquerda) e ensaio 2 (gráfico da direita).



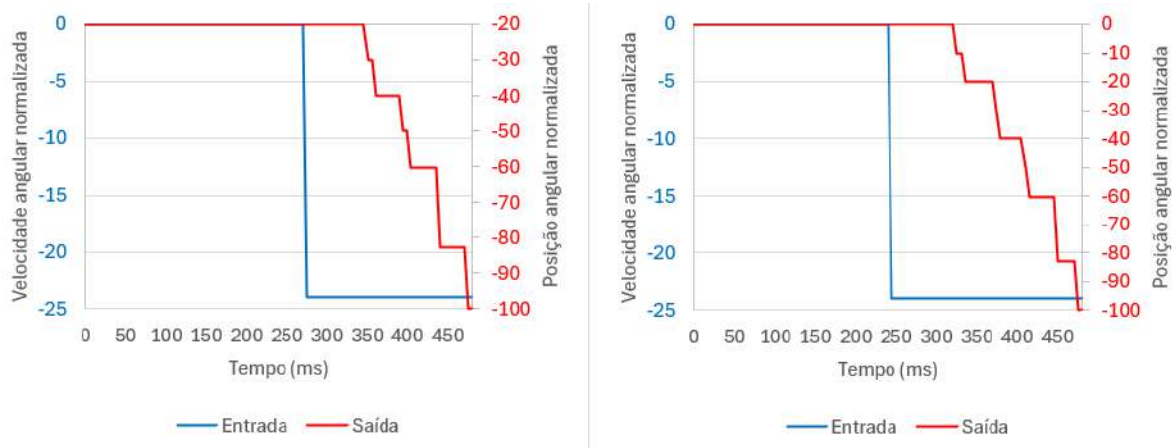
Fonte: Autor.

Figura 27 – Gráficos da velocidade diferencial, ΔV_{pu} (entrada), e a posição angular do robô, Θ_{pu} (saída), para um degrau de ΔV_{pu} = -50%: Ensaio 1 (gráfico da esquerda) e ensaio 2 (gráfico da direita).



Fonte: Autor.

Figura 28 – Gráficos da velocidade diferencial, ΔV_{pu} (entrada), e a posição angular do robô, Θ_{pu} (saída), para um degrau de ΔV_{pu} = -24%: Ensaio 1 (gráfico da esquerda) e ensaio 2 (gráfico da direita).



Fonte: Autor.

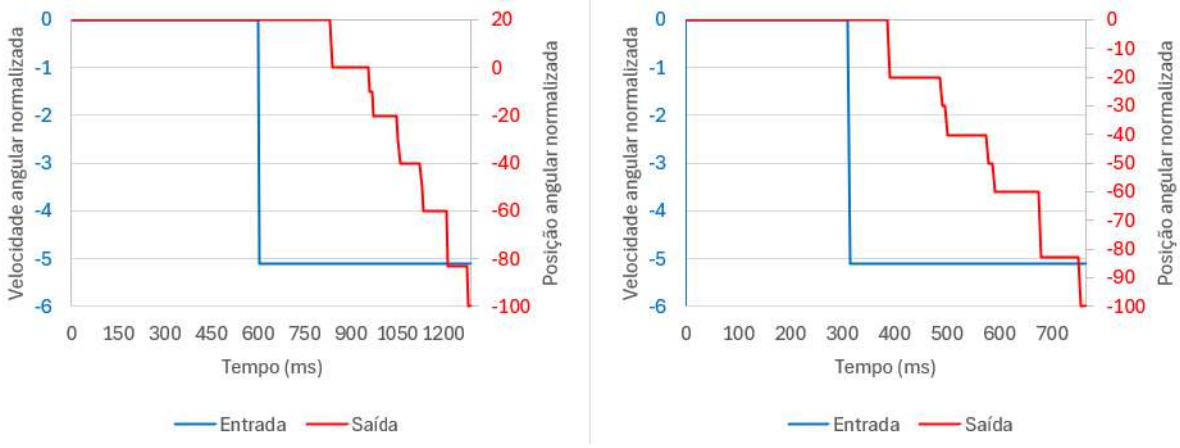
Com os dados coletados e devidamente armazenados, é possível iniciar o processo de obtenção do modelo matemático mais aderente ao sistema, usando as ferramentas adequadas.

3.3.3 Estimação de Parâmetros do Sistema

Nesta etapa, para estimar os parâmetros que compõem o modelo matemático do robô, é necessário usar uma técnica de regressão linear, conforme explicada na fundamentação teórica. Isso é feito para melhor adequação às características dos dados medidos.

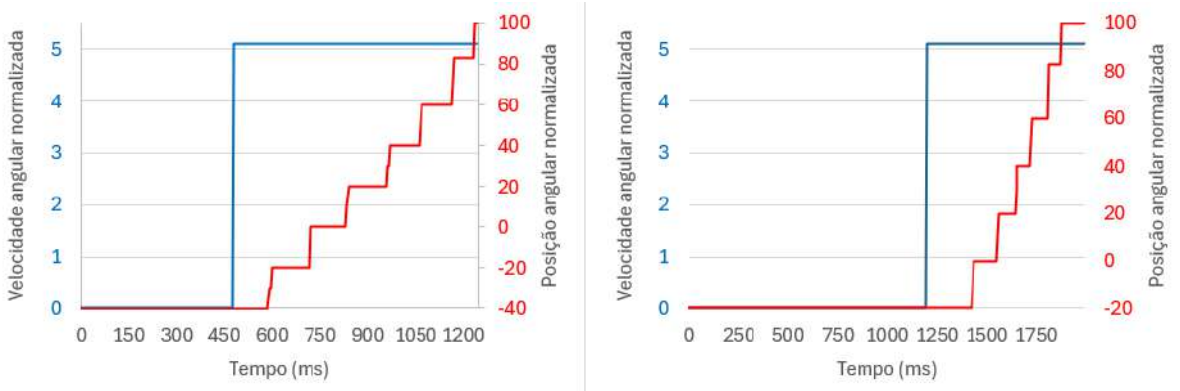
A ferramenta *"System Identification Toolbox"* do *MATLAB* é utilizada para estimar os parâmetros do sistema do robô seguidor de linha com base nos dados coletados em parte dos experimentos anteriores. A *System Identification Toolbox* fornece uma série de

Figura 29 – Gráficos da velocidade diferencial, ΔV_{pu} (entrada), e a posição angular do robô, Θ_{pu} (saída), para um degrau de ΔV_{pu} = -5%: Ensaio 1 (gráfico da esquerda) e ensaio 2 (gráfico da direita).



Fonte: Autor.

Figura 30 – Gráficos da velocidade diferencial, ΔV_{pu} (entrada), e a posição angular do robô, Θ_{pu} (saída), para um degrau de ΔV_{pu} = +5%: Ensaio 1 (gráfico da esquerda) e ensaio 2 (gráfico da direita).



Fonte: Autor.

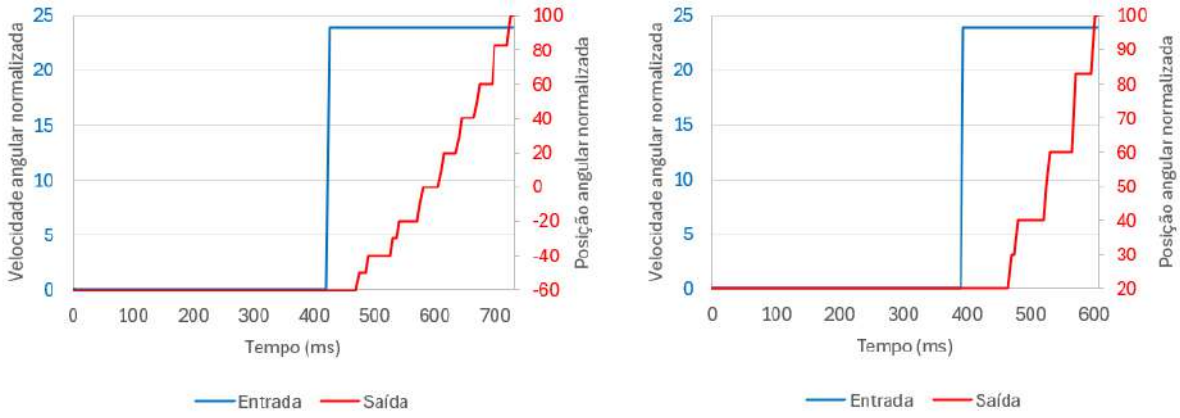
métodos e algoritmos para identificar modelos de sistemas dinâmicos a partir de dados de entrada e saída (MATHWORKS INC., 2024). Uma representação da ferramenta pode ser vista na Figura 34.

3.3.3.1 Apresentação da Ferramenta

A *System Identification Toolbox* é uma ferramenta poderosa que oferece uma variedade de técnicas para estimar modelos de sistemas lineares e não lineares a partir de dados experimentais (MATHWORKS INC., 2024). Alguns dos métodos disponíveis incluem:

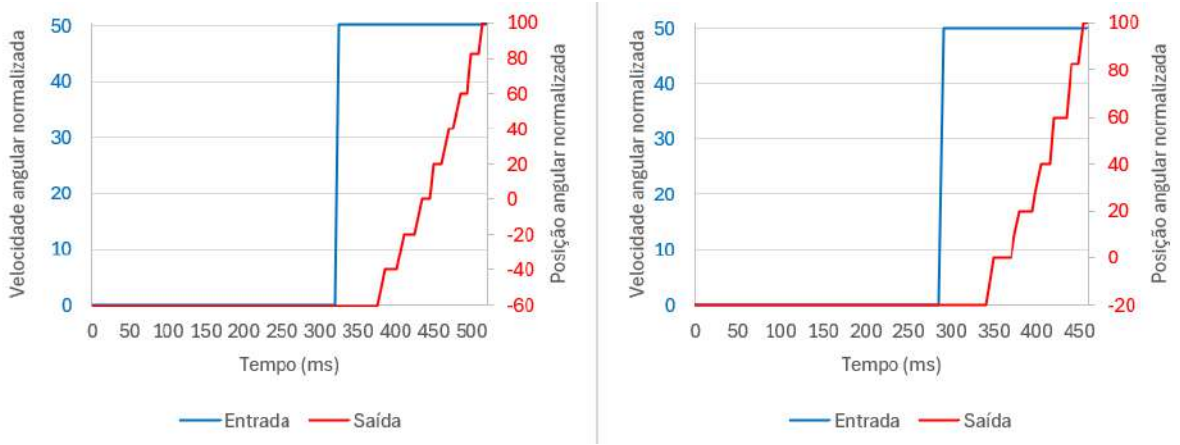
- Estimação de modelos ARX (*AutoRegressive with eXogenous inputs* - AutoRegressivo com Entradas Exógenas): modela a relação entre as saídas passadas do sistema e as entradas passadas, incluindo entradas exógenas, usando um modelo autoregressivo.

Figura 31 – Gráficos da velocidade diferencial, ΔV_{pu} (entrada), e a posição angular do robô, Θ_{pu} (saída), para um degrau de $\Delta V_{pu} = +24\%$: Ensaio 1 (gráfico da esquerda) e ensaio 2 (gráfico da direita).



Fonte: Autor.

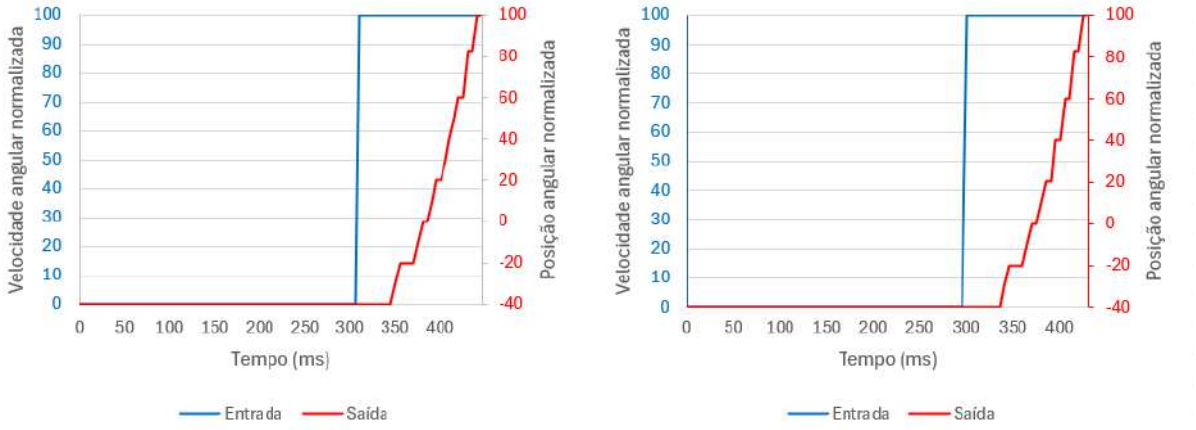
Figura 32 – Gráficos da velocidade diferencial, ΔV_{pu} (entrada), e a posição angular do robô, Θ_{pu} (saída), para um degrau de $\Delta V_{pu} = +50\%$: Ensaio 1 (gráfico da esquerda) e ensaio 2 (gráfico da direita).



Fonte: Autor.

- Estimação de modelos ARMAX (*AutoRegressive Moving Average with eXogenous inputs* - AutoRegressivo de Médias Móveis com Entradas Exógenas): similar ao ARX, mas também inclui termos de média móvel para lidar com comportamento transiente.
- Estimação de modelos de espaço de estados: modela o sistema em termos de equações de espaço de estados, descrevendo a evolução do estado do sistema ao longo do tempo.
- Estimação de modelos de séries temporais: modela o sistema como uma série temporal, explorando padrões de correlação e dependência temporal nos dados.
- Identificação de modelos polinomiais: estima modelos polinomiais que descrevem a relação entre entradas e saídas do sistema.

Figura 33 – Gráficos da velocidade diferencial, ΔV_{pu} (entrada), e a posição angular do robô, Θ_{pu} (saída), para um degrau de $\Delta V_{pu} = +100\%$: Ensaio 1 (gráfico da esquerda) e ensaio 2 (gráfico da direita).



Fonte: Autor.

Tabela 6 – Parâmetros dos ensaios e os links dos vídeos correspondentes.

Degrau de velocidade angular, ΔV_{pu}			***Link* do Youtube**
Degrau	Frenagem do motor 1	Frenagem do motor 2	
-100%	100%	0%	https://youtu.be/EojXJGbdMro
-50%	50,2%	0%	https://youtu.be/GtoweGippYg
-24%	23,9%	0%	https://youtu.be/Y-5TbTYW9RY
-5%	5,1%	0%	https://youtu.be/xZNyIZT9tdY
+5%	0%	5,1%	https://youtu.be/-5oaeaeCT2c
+24%	0%	23,9%	https://youtu.be/2gIEjzzSbw0
+50%	0%	50,2%	https://youtu.be/qqgkaJBxyKY
+100%	0%	100%	https://youtu.be/A8lv9MiVp6c

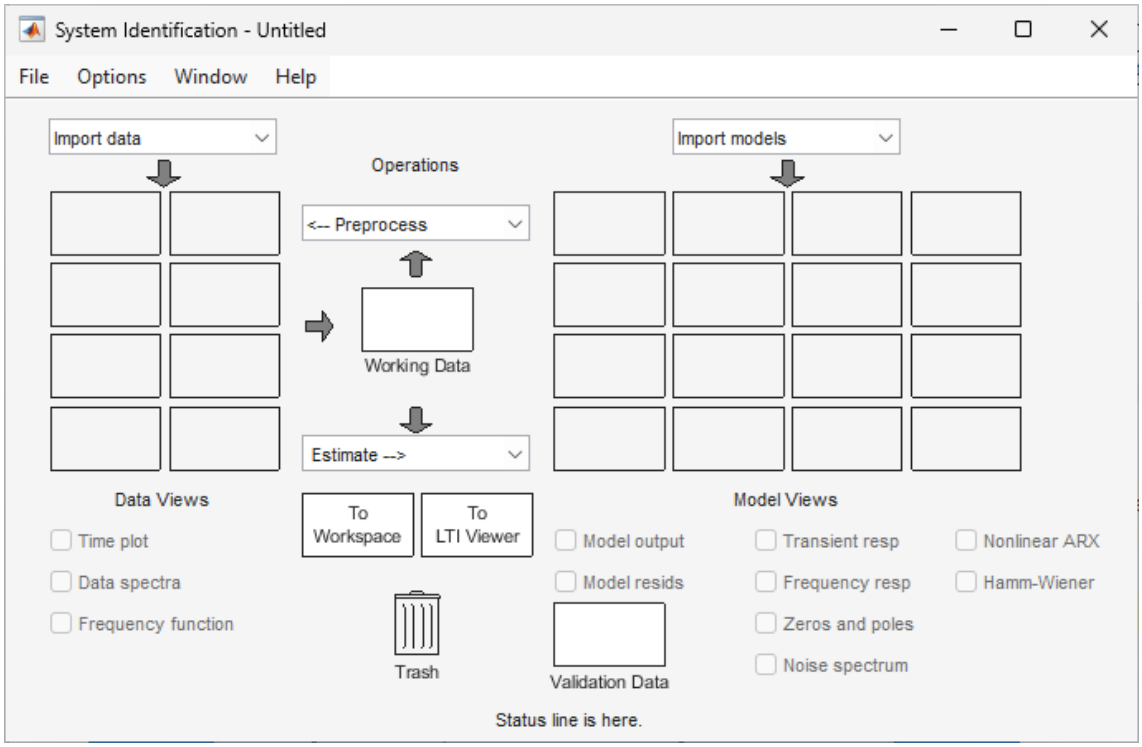
Fonte: Autor

- Estimação de modelos não lineares: identifica modelos que capturam relações não lineares entre entradas e saídas.

Esses métodos permitem a identificação de modelos que capturam diferentes aspectos do comportamento dinâmico do sistema, desde sistemas lineares de ordem baixa até sistemas não lineares complexos (MATHWORKS INC., 2024).

Para este trabalho, inicialmente é utilizado o método de estimação de modelos polinomiais para avaliar possíveis ordens para o sistema, resultando em uma melhor aproximação de um modelo de dois polos e dois zeros, mas exibindo boas aproximações para outras ordens, que serão verificadas mais a frente usando uma estimação de modelos ARX, pela simplicidade de aplicação, para encontrar uma função transferência do sistema.

Figura 34 – *System Identification Toolbox*.



Fonte: (MATHWORKS INC., 2024).

3.3.3.2 Configuração da Ferramenta

Para realizar a estimação dos parâmetros do sistema, primeiro se configura a ferramenta de acordo com as características dos dados experimentais e as especificações do modelo desejado. As etapas para configurar a ferramenta incluem (MATHWORKS INC., 2024):

- Importação dos Dados: os dados experimentais coletados são importados para o MATLAB e organizados em formato adequado para análise, seja em pares individuais ou em pacotes com vários pares de dados agrupados.

- Escolha do Modelo: com base na natureza do sistema e nos objetivos da identificação, seleciona-se um modelo adequado para representar a dinâmica do robô seguidor de linha. Considerando a abordagem de caixa preta adotada, opta-se por um modelo de regressão linear pelo método da estimação de modelos ARX por conveniência e simplicidade.

- Definição da Estrutura do Modelo: A estrutura do modelo é definida especificando a ordem do modelo, ou seja, o número de polos e zeros a serem incluídos no modelo. A escolha da ordem do modelo pode ser baseada em análises prévias dos dados, em conhecimento prévio do sistema ou por métodos iterativos, podendo até se usar de técnicas de validação de modelo para definir a melhor configuração de polos e zeros.

- Estimação dos Parâmetros: utilizam-se os dados experimentais e a estrutura do modelo definida para estimar os parâmetros do sistema. A ferramenta realizará a estimação dos parâmetros utilizando métodos de otimização adequados, como mínimos quadrados ou máxima verossimilhança.

- Avaliação do Modelo: após a estimação dos parâmetros, precisa-se avaliar a qualidade do modelo identificado utilizando técnicas de validação cruzada e análise de resíduos. Isso nos permitirá verificar se o modelo é capaz de reproduzir com precisão o comportamento do sistema em condições diferentes das usadas para a identificação.

Depois de identificar o sistema, é possível gerar um modelo matemático e exportá-lo para a área de trabalho do MATLAB para uso em outras ferramentas. Definindo as configurações adequadas, basta seguir para a análise dos dados utilizando a ferramenta.

3.3.3.3 Utilização da Ferramenta e Validação

As funções de análise e estimativa no *System Identification Toolbox* permitem trabalhar com vários pacotes de dados. Essencialmente, se foram executados vários experimentos e registrados vários conjuntos de dados de entrada-saída, pode-se agrupá-los em um único objeto IDDATA e usá-los com qualquer rotina de estimativa. Em alguns casos, é possível dividir esse único conjunto de dados de medição para remover partes onde a qualidade dos dados não é boa. Por exemplo, parte dos dados pode ficar inutilizável devido a distúrbios externos ou falha do sensor. Nesses casos, cada boa parte dos dados pode ser separada e então combinada em um único objeto IDDATA multi-experimental (MATHWORKS INC., 2024).

Os dados coletados nos experimentos, devidamente tratados na área de trabalho do MATLAB para a melhor estimativa, são inseridos na ferramenta com as configurações de período e offsets conforme estimado nos experimentos para a identificação do sistema. Como foram feitos testes para 4 degraus em cada direção, é possível arranjar os dados de forma que parte dos conjuntos de dados sirvam para estimação do modelo enquanto outra parte seja usados para validação do modelo.

Ainda sobre a configuração da ferramenta, há três questões que são importantes de serem definidas: a) necessidade ou não de modelar cada motor individualmente; b) modelar o sistema diretamente em tempo discreto ou discretizá-lo apenas após o projeto do controlador; e c) como escolher a melhor estrutura de polos, zeros e ganho para o modelo da planta.

O método escolhido foi a criação de apenas um modelo digital para ambos motores. Com relação ao atraso, visto nas Figuras de 26 a 33, ele foi determinado no processo de tratamento dos dados e removido de cada resultado experimental obtido, de forma que a ferramenta de identificação ficou somente responsável por obter o modelo do robô sem o

atraso. Após a obtenção do modelo, o atraso determinado na etapa de tratamento de dados foi inserido manualmente. Os dados sem os atrasos são limitados pela saturação da posição Θ nos testes, variando até uma das extremidades deixar a linha, e se mantendo constante pelo restante do ensaio, por esse motivo, na ferramenta de identificação foi incluído apenas a resposta transitória da análise dos dados. Então, é possível criar diferentes saídas em formato digital assumindo diferentes combinações de polos e zeros e usar as amostras separadas para validar o modelo que melhor se adequa.

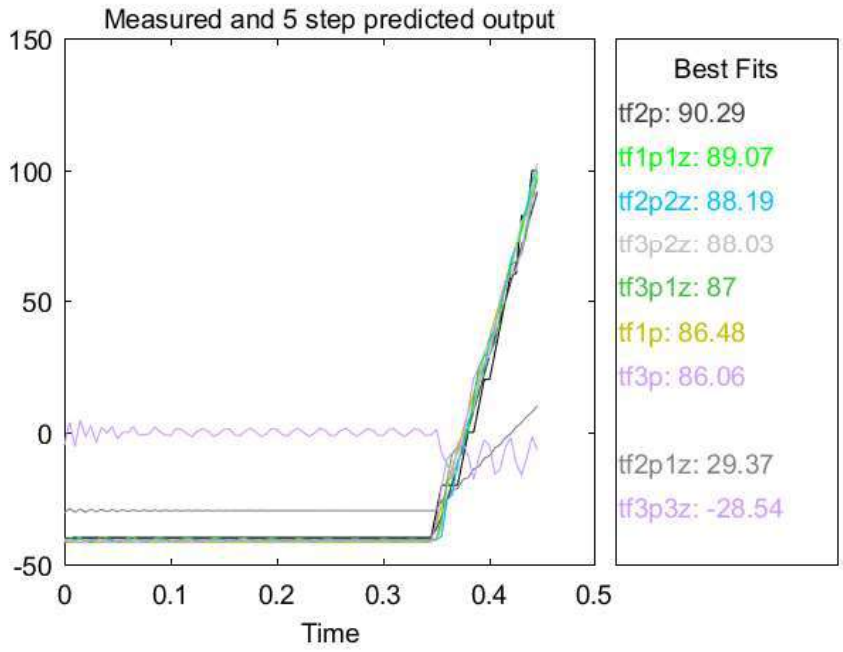
Seguindo os passos apresentados na subsessão anterior, utilizando dois conjuntos de dados de cada ensaio e usando um método iterativo, é possível gerar diversos modelos para avaliar qual modelo gera os melhores resultados.

A primeira iteração da ferramenta foi utilizada para definir a ordem do modelo a ser adotado para a planta, entre os diversos modelos testados. Assim, o modelo de primeira ordem, com um polo e sem zero, com um atraso de oito constantes de tempo (inseridas manualmente), se aproximou com uma boa precisão, conforme visto na Figura 35. Nela pode ser observado uma comparação de doze modelos, onde os quatro primeiros se destacam com uma mesma precisão de 81,55%, calculada pelo MATLAB por um método de NRMSE (*Normalized Root Mean Squared Error* - Erro de raiz quadrática média normalizado), onde tf1pd7 é um modelo de apenas um polo, mas que usou os dados antes da retirada manual do atraso, encontrando um atraso de sete constantes de tempo, tf1p1z é um modelo com um polo e um zero, tf1p1zd7 é o mesmo modelo, mas com um atraso de sete constantes de tempo, e tf1p é um modelo com apenas um polo e por ser o mais simples com o melhor resultado, foi o escolhido. Outros modelos que obtiveram uma precisão aceitável foram os referentes a: 3 polos e 2 zeros; 3 polos e 3 zeros; 3 polos e 1 zero; e 2 polos e 1 zero com atraso de sete constantes. Todavia, esses modelos apresentariam complexidades desnecessárias caso fossem escolhidos. Vale salientar que os resultados mostrados na Figura 35 se referem à identificação do modelo usando apenas os dados da frenagem do motor 1 a 100%, usando os mesmos dados da identificação na validação.

A segunda iteração foi utilizada para definir qual conjunto de dados geraria o melhor modelo, levando em consideração a validação par a par, aplicando uma validação cruzada. Parte dessa iteração pode ser vista na Figura 36, onde é possível verificar os dados dos ensaios como entrada do lado esquerdo, onde os de cima serão usados para identificação enquanto os de baixo para validação, os modelos ARX de diferentes configurações, estimados pela ferramenta, dispostos do lado superior direito, e uma tela no quadrante inferior direito, apresentando a saída comparativa entre cinco modelos gerados com base nas frenagens de 100% do motor 1, 100% do motor 2, 50% do motor 1, 24% do motor 1 e 5% do motor 1, respectivamente. Também é necessário avaliar outros modelos, gerados por conjuntos de dados empacotados, inclusive um formado por todos os dados de identificação.

Após essas iterações, é possível obter um modelo matemático, calculado a partir de

Figura 35 – Representação da saída real do sistema e a obtida por diversos modelos testados.



Fonte: Autor.

um único conjunto de dados de teste, referente ao da frenagem de 100% do motor 1, tendo resultado superior até sobre o modelo identificado usando todos os dados. Ele foi validado com um método de mínimos quadrados e teve uma adequação mínima de 59,7% e uma adequação média de 70,2% dos dados de teste usados para validação, enquanto o modelo identificado com todos os dados teve uma adequação média de 68,4%. É representado em tempo discreto por:

$$FT(z) = z^{-8} \cdot \frac{0.07067}{1 - z^{-1}} \tag{3.9}$$

Aplicando os mesmos dados e os mesmos métodos de forma sequencial, considerando agora sistemas de tempo contínuo (usando o plano s de Laplace), foi possível obter um sistema, contendo os seguintes parâmetros:

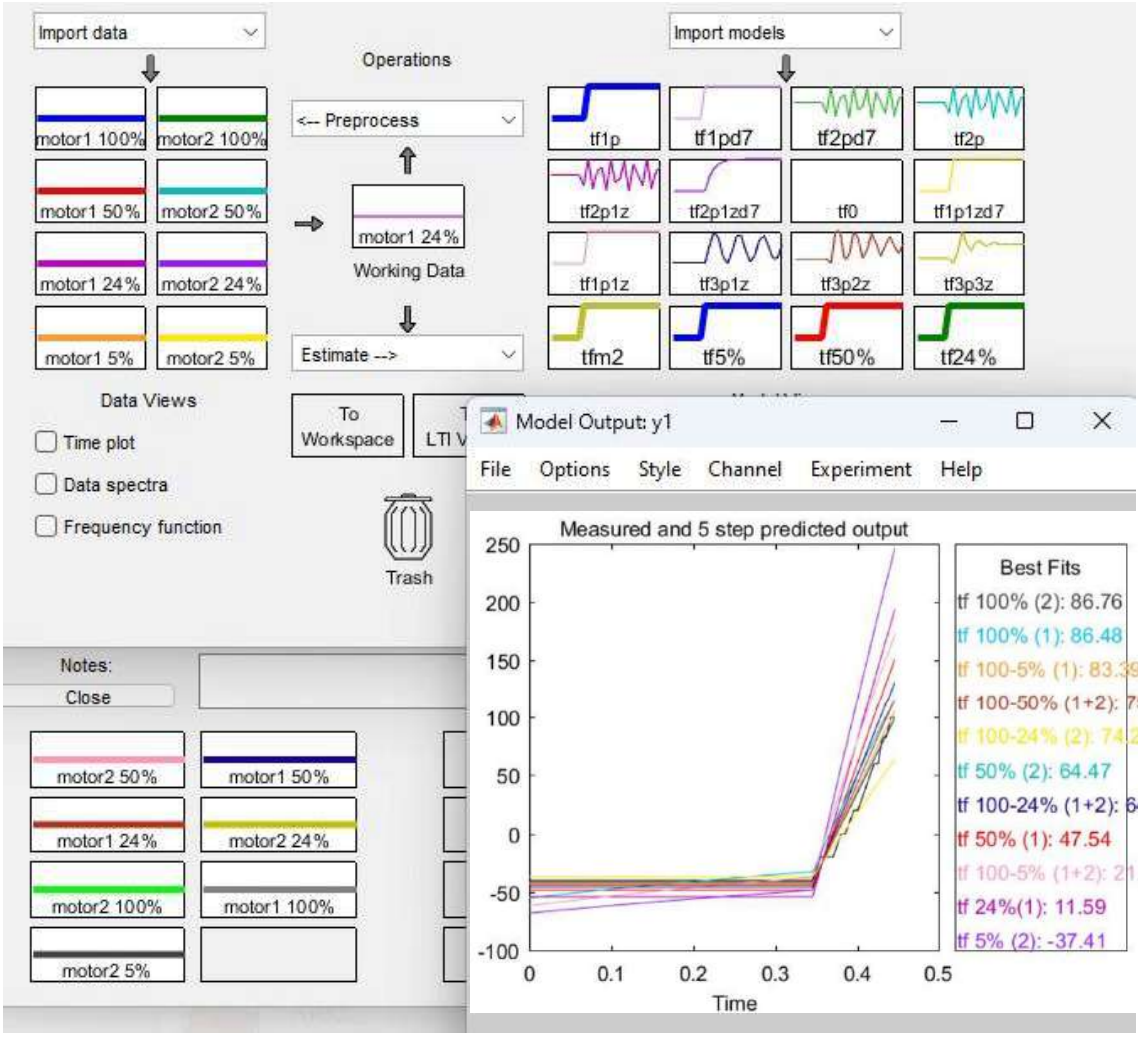
$$FT(s) = e^{-0,04s} \cdot \frac{14.21}{s + 5.565 \cdot 10^{-9}} \tag{3.10}$$

A conversão direta desse modelo usando a função $c2d()$ do MATLAB, com tempo de amostragem de 5 ms, resultou em um modelo descrito pela seguinte equação:

$$FT(z) = z^{-8} \cdot \frac{0.07105}{1 - z^{-1}} \tag{3.11}$$

Comparando os modelos demonstrados nas equações (3.9) e (3.11), é possível calcular uma diferença próxima a 0,5%, sendo assim considerados sistemas equivalentes.

Figura 36 – Iteração para identificar o sistema.



Fonte: Autor.

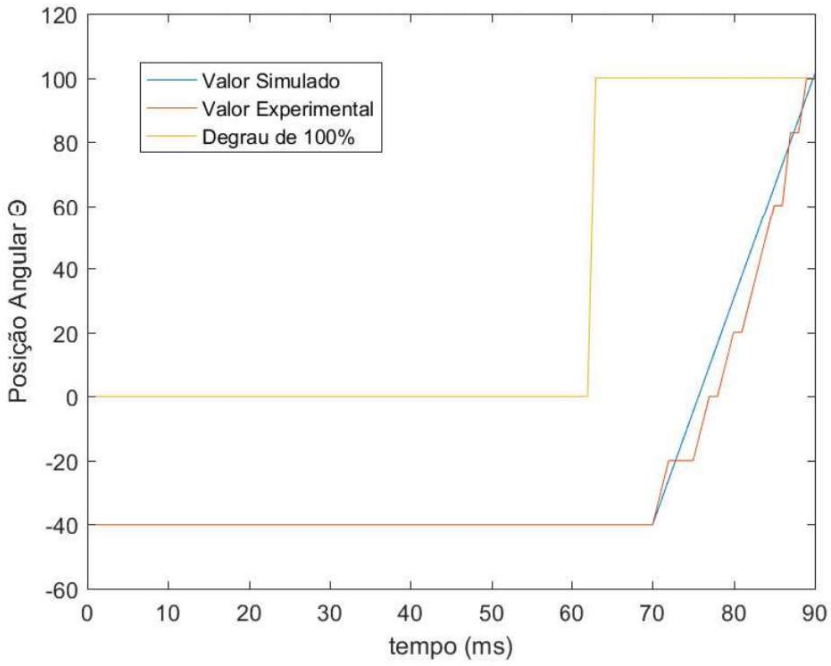
Finalmente, para validar os modelos identificados pelo *System Identification Toolbox*, parte dos dados de validação foi utilizada para comparar o comportamento esperado com o comportamento real. Foram selecionados arbitrariamente três ensaios, permitindo a geração dos gráficos apresentados nas Figuras 37, 38 e 39, os quais demonstram uma aproximação visual adequada para a planta identificada.

A identificação do sistema do robô seguidor de linha foi concluída, o que levou à criação de dois modelos matemáticos que descrevem a dinâmica do sistema com precisão. O próximo passo do projeto envolve escolher e projetar um controlador, utilizando estes modelos encontrados, para garantir que o robô tenha o melhor desempenho dinâmico possível na competição.

3.4 ESCOLHA E PROJETO DO CONTROLADOR DO ROBÔ SLC

Após a identificação do modelo do sistema, o próximo passo é o projeto do controlador. Isso garante que o robô seguidor de linha funcione de forma eficaz e precisa. A modelagem

Figura 37 – Comparação do comportamento real com o modelo identificado para degrau de $\Delta V_{pu} = +100\%$.



Fonte: Autor.

do controlador PID e suas variantes é abordada nesta seção, incluindo métodos para ajustar e otimizar os parâmetros do controlador para obter o desempenho desejado. Também são expostos os resultados quantitativos do controle escolhido.

3.4.1 Considerações Prévias

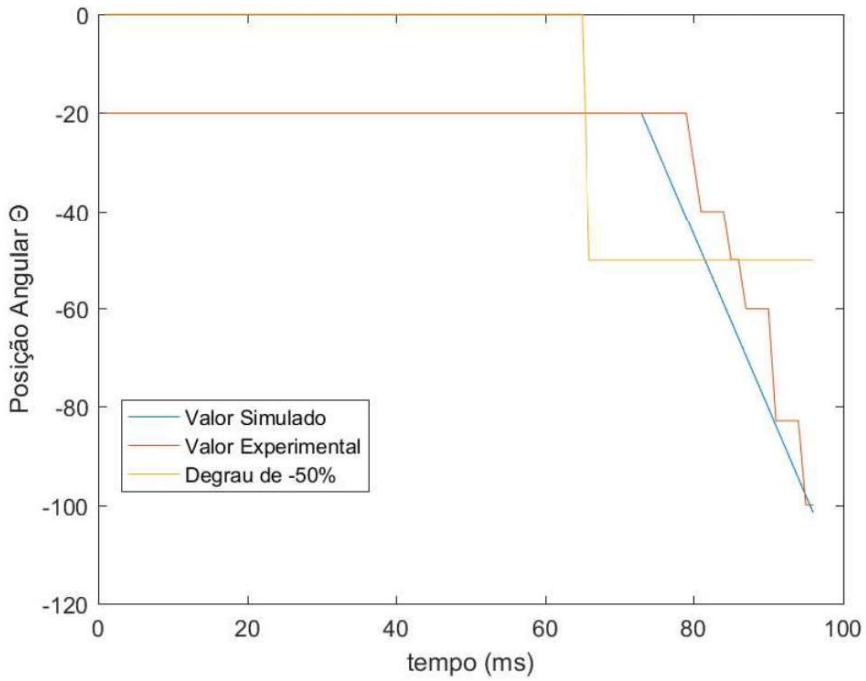
Nesta seção, são abordados os elementos básicos prévios ao projeto do controle do robô SLC. Isso inclui atrasos, saturação, ordem do sistema, conversão de entrada/saída, realimentação, discretização e a explicação do uso da ferramenta SISOTOOL do MATLAB.

3.4.1.1 Atraso, Saturação e Zona Morta

No controle de sistemas reais, é essencial considerar o atraso de transporte, a saturação e zona morta dos atuadores. O atraso pode causar instabilidade e deteriorar o desempenho do controlador, enquanto a saturação e a zona morta, limitam a resposta do sistema aos sinais de controle, especialmente em condições de operação extremas.

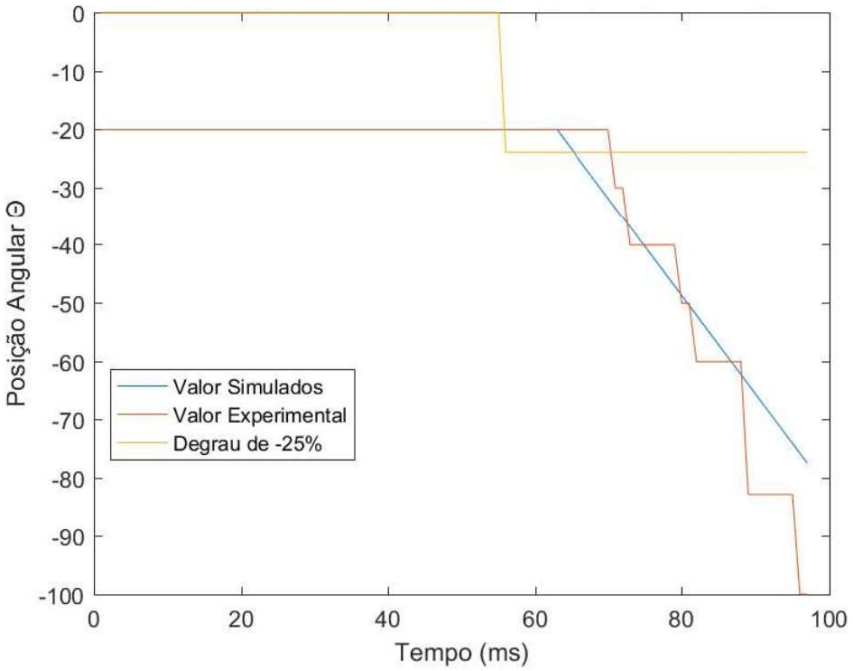
O atraso encontrado no modelo identificado apresenta um tempo aproximado de 40ms ou de 8 amostras de tempo (considerando um tempo de amostragem de 5ms). Esse tempo deve ser levado em consideração na elaboração do controlador para que a planta possa ser melhor controlada. No caso do modelo em domínio de s, o atraso é representado pela aproximação de Padé, que foi escolhido como 1ª ordem. Essa aproximação de Padé

Figura 38 – Comparação do comportamento real com o modelo identificado para degrau de ΔV_{pu} = -50%



Fonte: Autor.

Figura 39 – Comparação do comportamento real com o modelo identificado para degrau de ΔV_{pu} = -24%



Fonte: Autor.

pode ser utilizada se a frequência de passagem por 0dB for inferior a 1/10 da frequência equivalente do atraso, ou seja, inferior a 1/10 de 25 Hz (157 rad/s).

O sistema é saturado em seus limites mecânicos, onde um diferencial de velocidade angular passando de 75% começa a apresentar irregularidades em alguns trechos da pista por tentar fazer um dos motores girar muito próximo à sua zona morta. Assim, definiu-se os 75% de frenagem como sendo a saturação superior e uma variação de 0% como saturação inferior, já que 0% de frenagem se refere à velocidade e potência máximas.

3.4.1.2 Conversão de Entrada/Saída e Realimentação

Os sinais de entrada e saída devem ser convertidos para que o controle possa usá-los. Além disso, a realimentação do sistema deve ser implementada para que os sensores do robô forneçam ao controlador informações precisas e em tempo real. É importante salientar que, para os testes executados na coleta de dados, o microcontrolador entrava com a velocidade angular e media a posição angular, já que estava ensaiando somente o robô. Quando o controlador for inserido ao sistema, definindo a ação de controle para a planta, a entrada e saída são trocadas, sendo a saída do microcontrolador a diferença na velocidade angular calculada a partir da sua entrada, que é o erro da posição angular do robô em relação à posição da linha a ser seguida.

3.4.1.3 Discretização

Embora o controle possa ser projetado a partir de um modelo contínuo (analógico), o controlador deve ser configurado em domínio discreto antes de usá-lo no robô. Para garantir que o desempenho do controlador analógico seja mantido no domínio digital, a discretização emprega técnicas como a transformação bilinear (Tustin), que pode ser executada pelo MATLAB pela função $c2d()$, informando o tempo de amostragem $Ts = 5\text{ms}$.

3.4.1.4 Ferramenta SISOTOOL

O *Control System Designer App* ou SISOTOOL do MATLAB é uma ferramenta interativa que ajuda a projetar e ajustar controladores. Ela permite ajustes em tempo real para os parâmetros do controlador, análise de diagramas de LGR e respostas no domínio do tempo e da frequência em diferentes tipos de arquiteturas de sistemas. Ela apresenta diversas formas de projeto de controlador, desde métodos automátizados como sintonização de PID, de LQG (Linear-Quadrático-Gaussiano), de IMC (*Internal Model Control* - Controle por Modelo Interno) e sintonias baseadas em otimização, usando a ferramenta para ajudar na execução dos métodos, como editores gráficos interativos de diagramas de Bode e Nichols, Mapa do lugar geométrico das raízes, e levando em conta a possibilidade de adicionar, remover e deslocar polos, zeros e ganho do controlador em tempo real (MATHWORKS INC., 2024).

3.4.1.5 Geração dos LGRs

O LGR no domínio contínuo (s) e discreto (z) são gerados usando a ferramenta SISOTOOL. A ferramenta permite visualizar como os polos do sistema se movem no plano complexo à medida que o ganho do controlador é ajustado. Eles possuem algumas diferenças fundamentais devido à natureza distinta dos domínios.

Para o plano s (tempo contínuo), temos:

- Eixo Real: representa a estabilidade do sistema, onde a estabilidade é garantida se todas as raízes estão no semiplano esquerdo (partes reais negativas).
- Eixos Imaginários: representam as frequências naturais do sistema.
- Padrão de Trajetórias: as trajetórias das raízes no plano s são contínuas e podem ou não atravessar o eixo imaginário.

Para o plano Z (tempo discreto), temos:

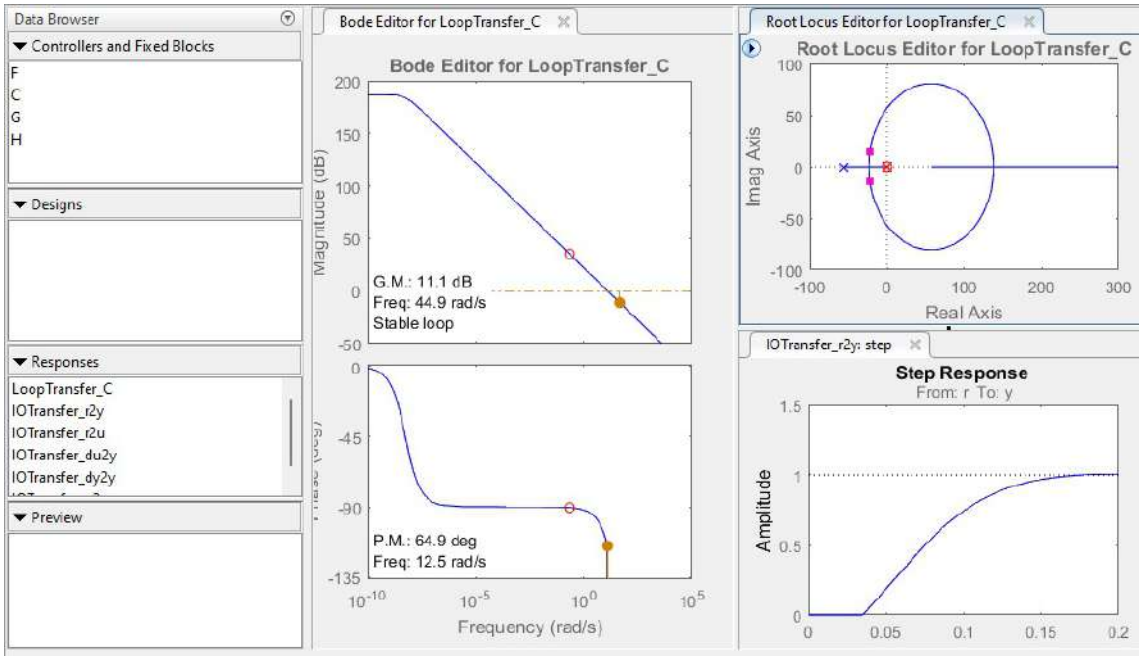
- Círculo Unitário: a estabilidade do sistema é garantida se todas as raízes estiverem dentro do círculo unitário (módulo menor que 1).
- Eixos Real e Imaginário: no plano z, o eixo real vai de -1 a 1 e o eixo imaginário corresponde ao círculo unitário.
- Padrão de Trajetórias: as trajetórias das raízes no plano z formam curvas fechadas e podem ou não sair do círculo unitário.

Como os domínios contínuo e discreto diferem em termos de critérios de estabilidade e representações de dinâmica do sistema, essas diferenças são importantes considerações para o projeto de controladores (OGATA, 2010).

Para um estudo mais completo, são importantes na modelagem do controlador, a apresentação dos conceitos de margem de ganho e margem de fase, que são medidas no domínio da frequência, extraídas de um diagrama de Bode, usadas para avaliar a estabilidade de um sistema. A margem de ganho determina como o ganho do sistema pode ser aumentado antes que se torne instável. Ela é medida no ponto onde a fase da função de transferência de malha aberta é -180°, e uma margem de ganho positiva mostra que o sistema é capaz de suportar um aumento no ganho sem perder a estabilidade. A margem de fase indica a quantidade de atraso de fase acima de -180° no ponto do diagrama de amplitude que cruza 0 dB (ganho de frequência cruzada). Uma margem de fase positiva indica que o sistema pode tolerar um certo atraso de fase sem perder a estabilidade (OGATA, 2010). No LGR, a margem de ganho se refere à análise do movimento dos polos de malha fechada à medida que o ganho varia, enquanto a margem de fase se relaciona à proximidade desses polos ao eixo imaginário.

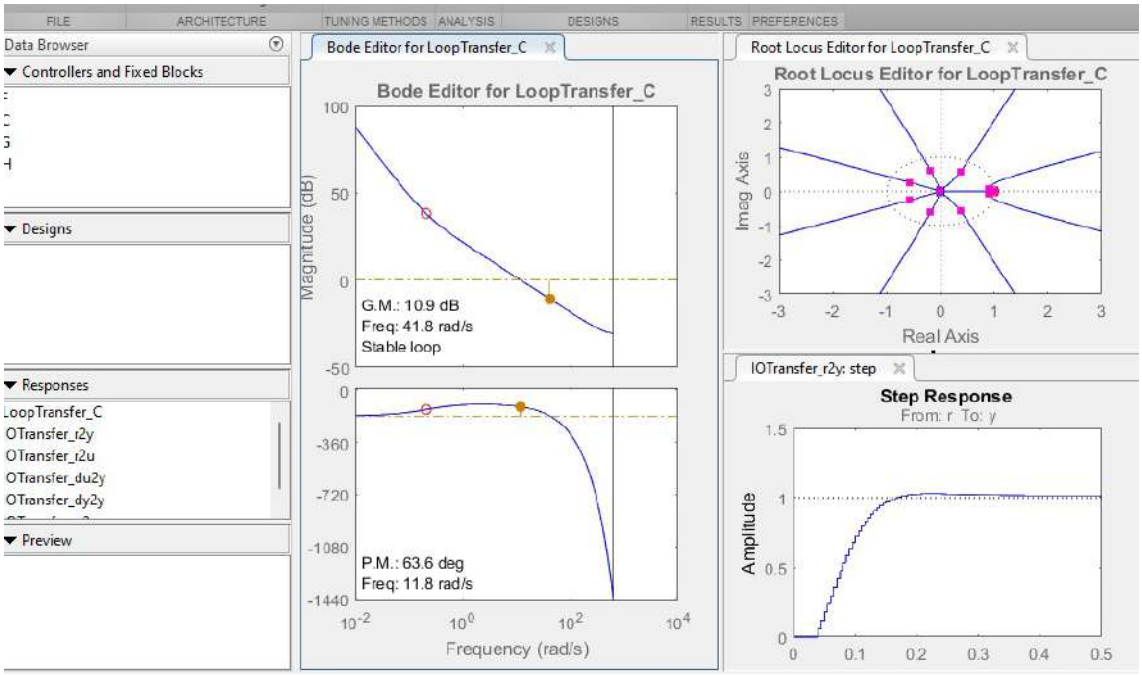
Os LGRs gerados com base nos sistemas identificados podem ser vistos nas Figuras 40 e 41.

Figura 40 – Exemplo do uso do SISOTOOL para uma planta contínua.



Fonte: Autor.

Figura 41 – Exemplo do uso do SISOTOOL para uma planta discreta.



Fonte: Autor.

3.4.2 Projeto do Controlador Baseado no LGR e na Resposta em Frequência

É apresentado, nesta seção, o uso da ferramenta SISOTOOL para gerar o LGR e o diagrama de Bode, necessários para projetar os ganhos do controlador. A análise dos LGRs é essencial para entender a dinâmica do sistema e ajustar os parâmetros do controlador adequadamente. Por fim, resultando nos controladores no domínio discreto e como aplicá-

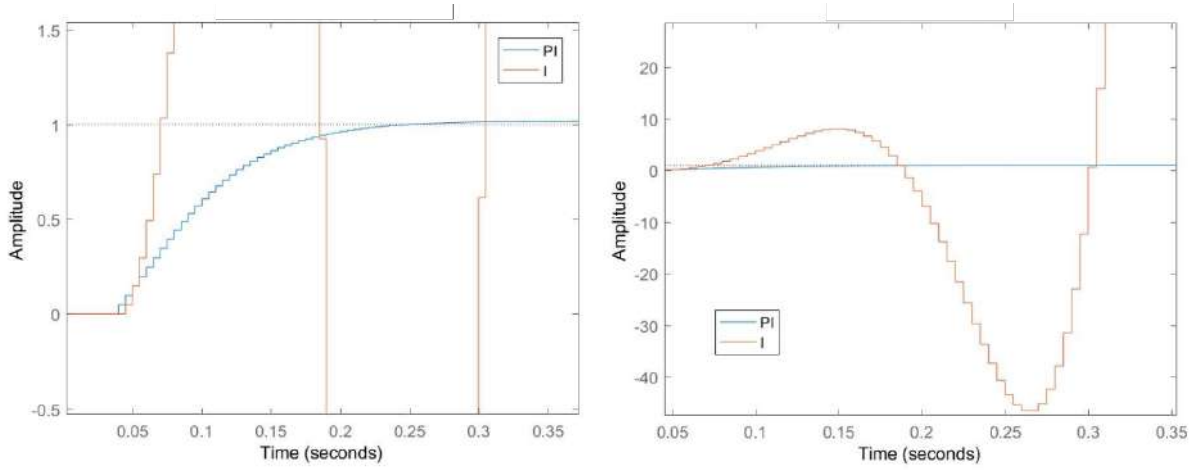
los em robôs seguidores de linha.

3.4.2.1 Ajuste do Controlador

Inicialmente, foram definidos os requisitos de projeto para o controlador do robô SLC: estabilidade do sistema, ausência de sobressinal ou qualquer tipo de oscilação, erro estacionário nulo para entradas do tipo degrau e menor tempo de acomodação possível. Tais requisitos foram escolhidos para garantir que o robô siga a linha sem trepidações, mantendo-se no percurso de forma precisa e eficaz.

Para atender a esses requisitos, optou-se por selecionar o controlador mais simples capaz de satisfazê-los. Controladores P e PD foram descartados, pois não atendem ao requisito de erro estacionário nulo. Restaram, então, as opções I, PI e PID. Porém, o controlador I, não foi capaz de gerar um controlador estável, a tentativa de criar um controle com os mesmos parâmetros do PI resultou em um crescimento oscilatório tendendo ao infinito, demonstrado no gráfico da Figura 42, demonstrando que esse controlador seria inadequado.

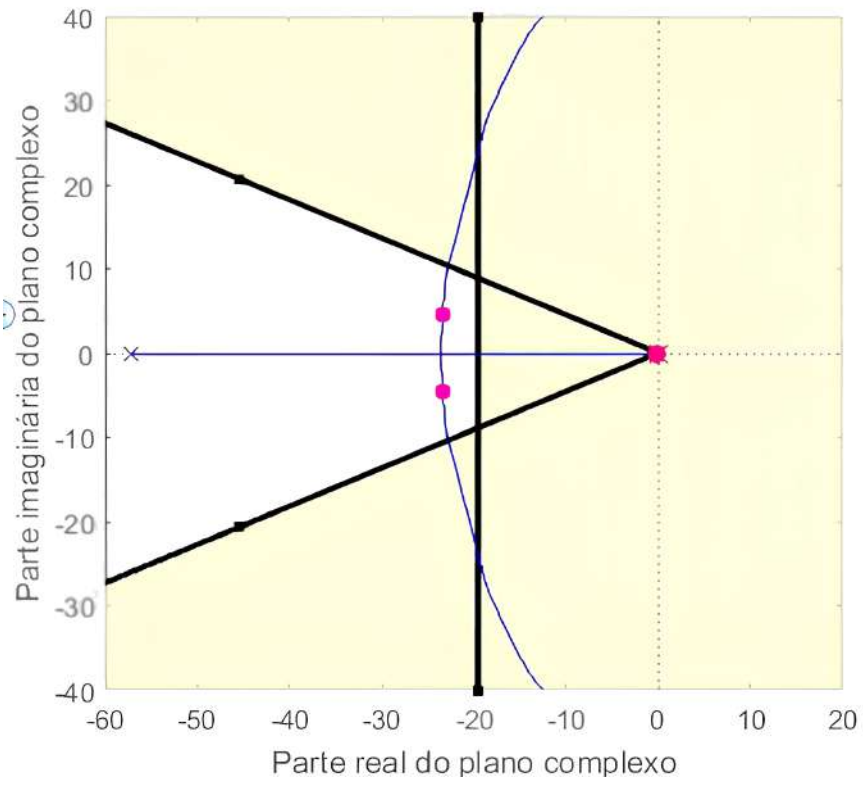
Figura 42 – Comparação dos controladores I e PI para uma mesma entrada: Escala em relação ao PI (gráfico à esquerda) e em relação ao I (gráfico à direita).



Fonte: Autor.

O controlador PI foi projetado usando os requisitos de projeto para restringir as regiões do LGR, para os critérios de tempo de acomodação de 0,2 s e máximo sobressinal de 1%, criando zonas de projetos das raízes conforme demonstrado nas Figuras 43 e 44, para mapas no plano s e z, respectivamente. Definiu-se o zero e o ganho do PI de modo que todos os pólos em malha fechada ficassem nesse intervalo, sendo reais, com baixa oscilação, com os pólos dominantes distantes do eixo $j\omega$. O uso de um controlador PID foi considerado desnecessário, pois exigiria a inclusão de um pólo extra para tornar o PID causal, sem uma melhoria significativa na dinâmica de controle que justificasse a complexidade adicional.

Figura 43 – Mapa do LGR em s com zonas de atendimento dos critérios do prejeto.



Fonte: Autor.

O controlador $C(s)$ em Laplace foi então transformado para o domínio discreto utilizando a transformação bilinear via a função $c2d()$ do MATLAB. Assim, obteve-se o controlador $C(z)$ mostrado na Equação 3.12. Aplicou-se a mesma metodologia para a planta discreta, resultando no controlador PI discreto da Equação 3.13. Ambas as metodologias, contínua e discreta, levaram a controladores muito próximos.

$$C(z) = 0,786 \cdot \frac{z - 0,999}{z - 1} \tag{3.12}$$

Para o sistema identificado discreto.

$$C(z) = 0,782 \cdot \frac{z - 0,999}{z - 1} \tag{3.13}$$

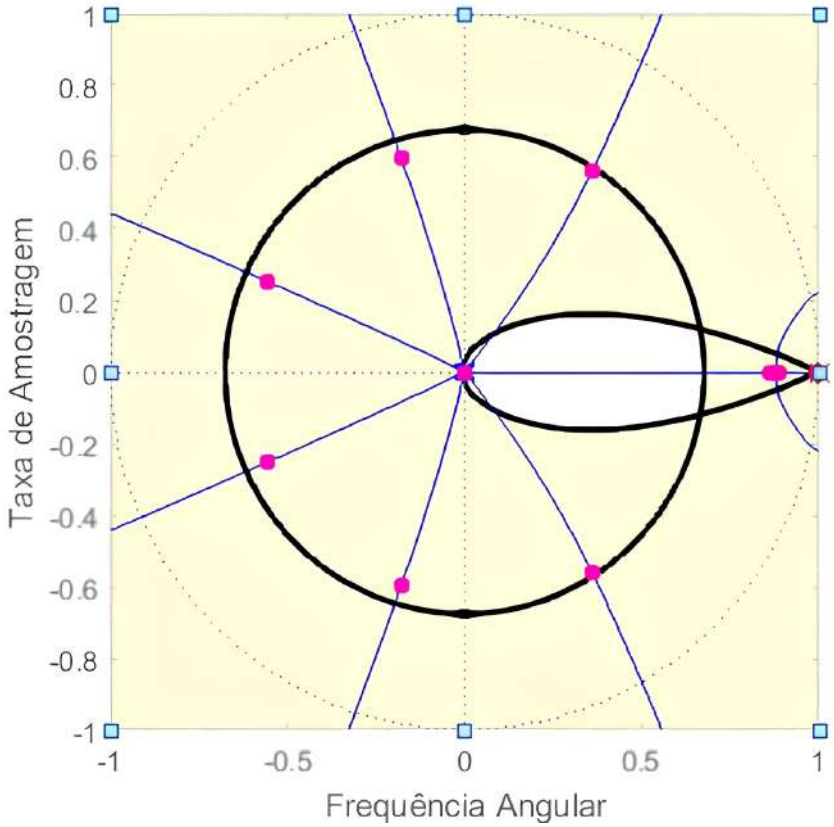
Para o sistema identificado analógico e discretizado depois.

Os resultados mostraram que o controlador discretizado apresenta um ganho bem próximo ao controlador discreto, indicando uma boa aproximação e andamento satisfatório do projeto até o momento.

3.4.2.2 Aplicação do Controlador

Para implementar o controlador no microcontrolador, é necessário transformar a função de transferência $C(z)$ para uma equação de diferença em função das entradas e saídas

Figura 44 – Mapa do LGR em z com zonas de atendimento dos critérios do prejeto.



Fonte: Autor.

atuais e passadas.

A função de transferência $C(z)$ encontrada é:

$$C(z) = \frac{U(z)}{E(z)} = K \cdot \frac{z - 0,999}{z - 1} \tag{3.14}$$

Manipulando-a para ficar em função das entradas e saídas atuais e passadas, temos:

$$C(z) = K \cdot \frac{(z - 0,999) \cdot z^{-1}}{(z - 1) \cdot z^{-1}} \tag{3.15}$$

Resultando em:

$$\frac{Y(z)}{U(z)} = K \cdot \frac{1 - 0,999 \cdot z^{-1}}{1 - z^{-1}} \tag{3.16}$$

Que pode ser escrita na forma:

$$Y(z) \cdot (1 - z^{-1}) = K \cdot U(z) \cdot (1 - 0,999 \cdot z^{-1}) \tag{3.17}$$

De onde é possível extrair:

$$Y(z) = K \cdot U(z) - 0,999 \cdot K \cdot U(z) \cdot z^{-1} + Y(z) \cdot z^{-1} \tag{3.18}$$

No tempo discreto k, a equação torna-se:

$$y(k) = K \cdot u(k) - 0,999 \cdot K \cdot u(k-1) + y(k-1) \tag{3.19}$$

Essa equação de diferença é programada no microcontrolador, onde $e(k) = r(k) - y(k)$ é o erro entre a referência $r(k)$, que é assumida como zero (alinhada com a fita), e a saída $y(k)$, que é a saída do sensor. A grandeza $u(k)$ representa o diferencial da velocidade angular do robô, assumindo valores entre -100% e +100%. K é a constante de ganho, que vale 0,786 para o controlador encontrado pela planta digital e, alternativamente, vale 0,782 para o controlador em s após discretização, apresentando um desvio relativo de 0,5%. Essas fórmulas são as aplicações diretas dos controladores que podem ser programadas no microcontrolador em função dos valores de entrada atual ($e(k)$), entrada anterior ($e(k-1)$) e saída anterior ($u(k-1)$). Assim, é possível aplicar o controlador da Equação 3.18 no robô SLC usando o valor de K obtido do controlador encontrado pela planta digital, que, a princípio, apresentou um desempenho satisfatório. Essa aplicação do controle no robô pode ser observada no apêndice A, como as linhas 111 e 117.

Para finalizar a aplicação do controle, vale ressaltar que o *wind-up* é um fenômeno a se atentar em controladores PI, ele ocorre quando o atuador está saturado, e o integrador no controlador continua a acumular erro, levando a um desempenho degradado e possíveis oscilações indesejadas. Para evitar esses problemas, são aplicadas técnicas de *anti-wind-up*. Neste trabalho, utilizou-se uma técnica não convencional que satura as constantes antes do integrador, sendo simples de aplicar, mas que pode não ser a mais adequada. Devido à complexidade do tema, uma análise mais aprofundada será deixada para trabalhos futuros.

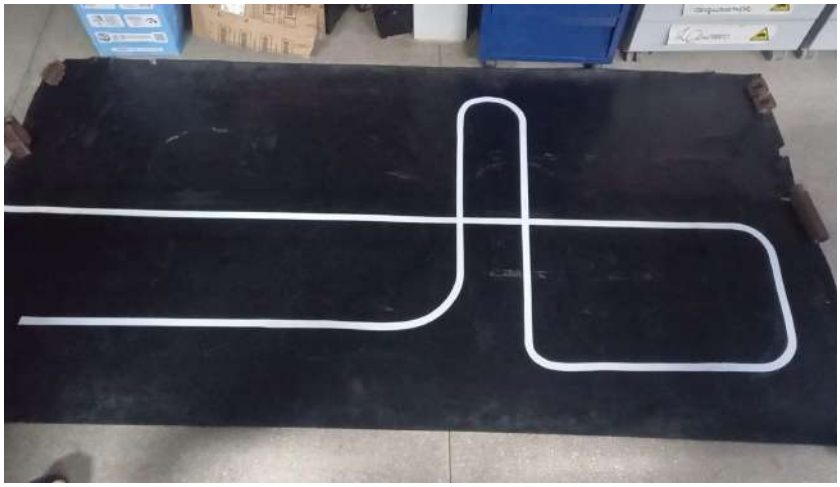
3.4.2.3 Análise dos Resultados

Para validar o controlador, foram feitos alguns testes práticos em uma pista mais elaborada contendo cenários com trechos retos, curvas suaves e curvas acentuadas para avaliar o tempo de acomodação e o desvio prático em cada situação. A pista montada para avaliação do controle pode ser vista na Figura 45, sendo também usada para os testes adicionais em condições adversas. O robô demonstrou capacidade de seguir todas as linhas sem desvio significativo, como pode ser visto no vídeo do *link* https://youtu.be/m8mqVxYbLJc, os resultados estão expressos na Tabela 7.

Valendo salientar que o desvio apresentado nessa tabela não se trata de um sobressinal, e sim, o quanto o robô se distancia da referência antes que a ação de controle possa corrigí-lo, é proporcional ao erro máximo em cada cenário. O tempo de alinhamento equivale ao tempo decorrido desde o desvio máximo até o alinhamento, não nessessáriamente é o tempo de acomodação do controle diretamente.

De forma a comparar o desempenho do controle no robô SLC real na pista prática com o modelado simulado, dados experimentais de ensaio foram inseridos no sistema para avaliar respostas esperadas a partir de um estado inicial. Usando como base um estado

Figura 45 – Pista para avaliação do controle projetado.



Fonte: Autor.

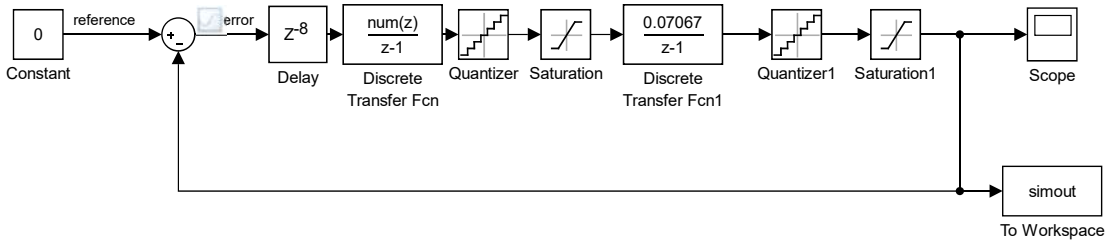
Tabela 7 – Resultado dos testes práticos.

Cenário	Tempo de Alinhamento	Desvio Máximo
Linha Reta	215 ms	8 mm
Curva Suave	465 ms	24 mm
Curva Acentuada	930 ms	55 mm

Fonte: Autor

inicial de Θ_{pu} máximo (100%), o diagrama de blocos do sistema foi construído no *Simulink* do MATLAB como visto na Figura 46. Seu resultado foi exportado para a área de trabalho do MATLAB para poder gerar outras análises e discussões.

Figura 46 – Sistema montado no *Simulink* para validação do modelo e do controle.



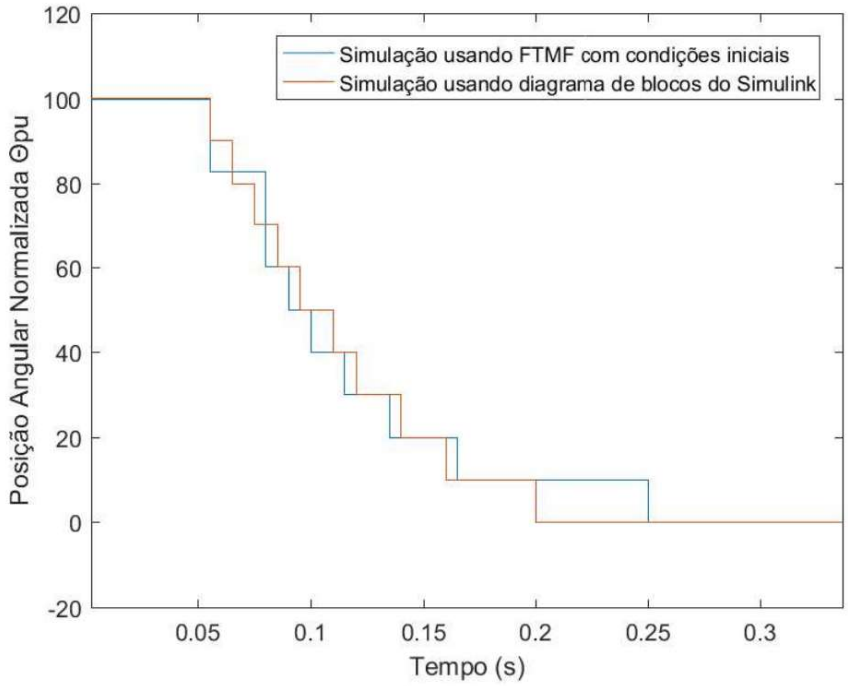
Fonte: Autor.

Alternativamente, é possível modelar a função transferência em malha fechada do sistema em tempo discreto obtido pela função *feedback()* do MATLAB, para esse caso, foram usados os modelos digitalizados do domínio de Laplace, isto é, tanto a planta identificada quanto controlador, são os encontrados em s, após a aplicação da função $c2d()$ usando Ts=5ms. A equação da FTMF pode ser descrita simplificadamente por:

$$\frac{z^{-8} \cdot (0,00963 - 0,00962z^{-1})}{1 - 1,4z^{-1} - 0,2z^{-2} + 0,4z^{-3} + 0,4z^{-4} + 0,1z^{-5} - 0,1z^{-6} - 0,1z^{-7}} \tag{3.20}$$

Tal modelo de malha fechada pôde ser usado para simular resultados de reação a estados iniciais não-nulos usando espaço de estados, então, repetindo o teste anterior, com um estado inicial de Θ_{pu} máximo (100%), é possível obter os resultados em forma gáfica e compará-los com resultados obtidos da simulação anterior. Essa comparação está apresentada na Figura 47, corroborando que o projeto pode ser elaborado por qualquer um dos dois métodos, tanto em domínio de z, quanto em domínio de s.

Figura 47 – Comparação dos sistemas simulados (diagrama de blocos em z e espaços de estados $c2d(s)$).

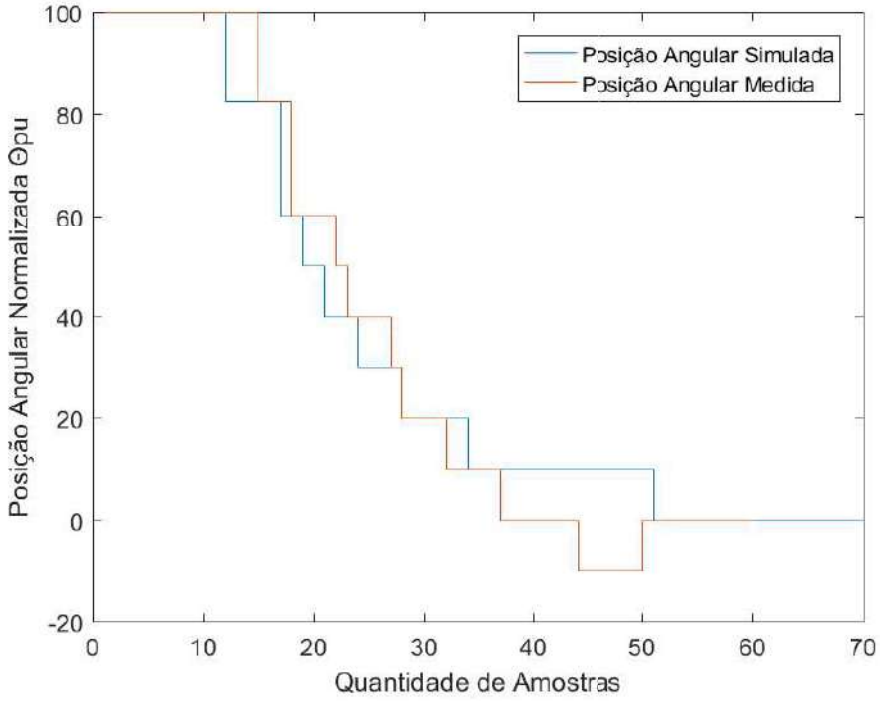


Fonte: Autor.

Concluindo essa análise, é possível ainda comparar qualquer um dos resultados anteriores com o teste prático do modelo real nas mesmas condições, resultando no gráfico da Figura 48 (onde foi usada a simulação da FTMF).

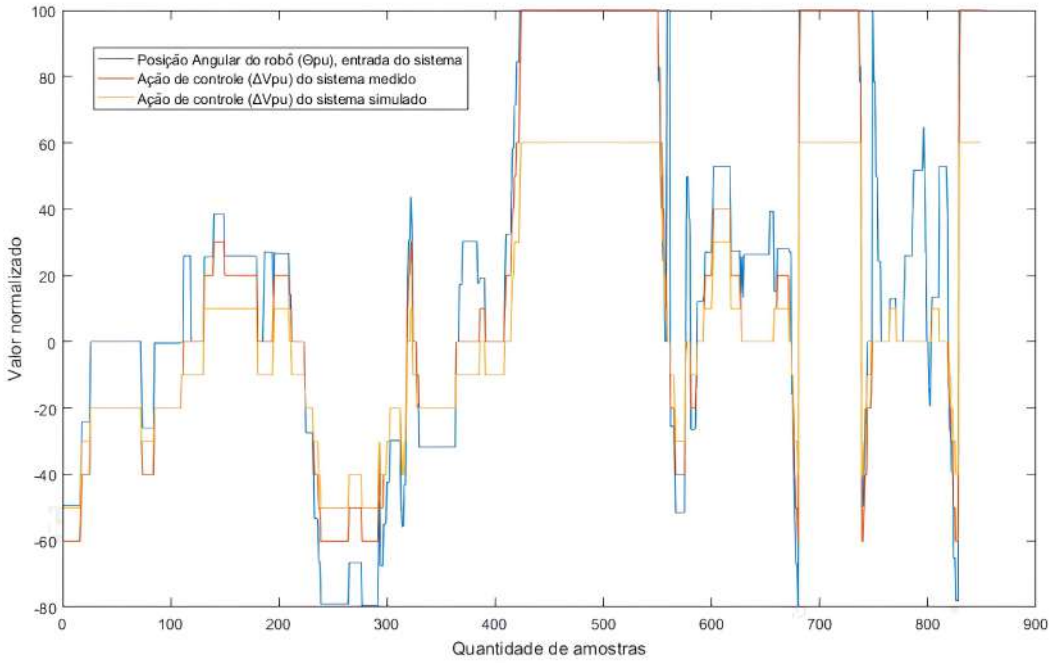
Para finalizar a análise do controlador modelado, é realizado um teste mais abrangente no qual o robô controlado deve seguir um percurso mais complexo. Durante esse teste, os valores de entrada ($-\Theta_{pu}$) e saída (ΔV_{pu}) foram armazenados. Ao aplicar esses valores de entrada em uma malha aberta com o controlador modelado, espera-se que as ações de controle, ΔV_{pu}, sejam equivalentes às do sistema real. Utilizou-se a função $lsim()$ do MATLAB para gerar o gráfico correspondente, resultando em valores relativamente próximos. No entanto, foi observada uma diferença notável quando comparada ao sistema real, embora essa diferença não comprometa o objetivo geral do controle. O gráfico comparativo dessa análise está representado na Figura 49.

Figura 48 – Comparação do sistema simulado em espaço de estados com modelo Real.



Fonte: Autor.

Figura 49 – Comparação da simulação com Resultados Reais.



Fonte: Autor.

3.4.2.4 Testes Adicionais

Foram realizados testes adicionais para avaliar o desempenho do controle do Robô SLC em condições adversas e possíveis influências do ambiente.

Influência da Luminosidade:

Os testes de luminosidade foram realizados para verificar a atuação dos sensores de linha em diferentes condições de iluminação. Em todos os casos a linha foi seguida adequadamente, como pode ser visto no vídeo de *link* https://youtu.be/XR8CXeN7kgg, restando considerações a serem feitas em relação ao desempenho do controle, que apresentou uma pequena variação para esses testes. Essa variação foi calculada usando fórmulas do Excel para comparar os dados nessas condições com os dados em condições normais, avaliando o desvio padrão dos valores. Os resultados apresentados na Tabela 8 estão em relação aos resultados obtidos de testes em condições normais.

Tabela 8 – Resultados dos testes de luminosidade.

Condição de iluminação	**Semelhança da detecção**
Luz Normal	Referência
Luz Alta	99,19%
Luz Baixa	95,86%

Fonte: Autor

Influência da Inclinação:

Testes adicionais foram realizados inclinando o plano da pista e avaliando a influência no controle como feito nos testes de iluminação, visto que o robô também foi capaz de seguir a linha adequadamente nesses testes, como pode ser visto no vídeo do *link* https://youtu.be/1WlM-4yKrIc. Analisando da mesma forma que o teste anterior, os resultados em relação aos testes em condições normais estão representados na Tabela 9.

Influência da Bateria:

Testes foram realizados para verificar o desempenho do robô com diferentes níveis de carga da bateria. Os resultados apresentados na Tabela 10 foram obtidos comparando os tempos de execução do percurso e as velocidades de cruzeiro, que em condições de bateria carregada são de aproximadamente 9,1 s e 80 cm/s, respectivamente.

Com a bateria abaixo de 40%, os motores apresentaram dificuldades com a zona morta, afetando o controle e desempenho do robô, abaixo de 30% o motor não fornece corrente para mover os motores.

Tabela 9 – Resultados dos testes de inclinação.

Inclinação do plano	**Semelhança da detecção**
0°	Referência
5°	94,8%
10°	87,6%

Fonte: Autor

Tabela 10 – Resultados dos testes com diferentes níveis de carga de bateria.

Nível de Carga	**Tempo de Percurso**	**Velocidade de Cruzeiro**
100%	9,1 s	80 cm/s
Cerca de 70%	11,0 s	65 cm/s
Cerca de 40%	19,6 s	35 cm/s

Fonte: Autor

3.4.2.5 Falhas em componentes

Durante o funcionamento de um robô SLC, diversos problemas operacionais podem aparecer, comprometendo seu desempenho. Um dos problemas comuns é a desconexão de componentes, como sensores, devido a perda de conexão ou mau contato nos *jumpers* do robô. Para avaliar o impacto dessas falhas no desempenho do robô SLC, foram realizados testes adicionais onde, em cada teste, um dos jumpers dos sensores foi desconectado e o robô foi posto para seguir o mesmo trajeto dos testes anteriores. Observou-se que a desconexão de qualquer sensor resulta em uma queda significativa no rendimento do robô. Especificamente:

- Sensores Centrais Desconectados: quando um dos sensores centrais (-1 e 1) foi desconectado, o alinhamento em retas foi o principal impactado. O robô não conseguiu manter uma trajetória reta, oscilando frequentemente na pista.
- Sensores Laterais Desconectados: a desconexão de um dos sensores laterais (-5, -3, 3 e 5) gerou dificuldades significativas na realização de curvas, comprometendo a navegação em trajetórias sinuosas.
- Sensores Externos Desconectados: a desconexão dos dois sensores mais externos (-7 e 7) representou o pior cenário. Quando o robô saía da pista pelo lado do sensor desconectado, ele não conseguia retornar, resultando em descarte da tomada de tempo e possível desqualificação do robô em competição.

Além dos sensores, outros componentes do robô também estão sujeitos a problemas de conexão devido a falhas nos *jumpers*, acarretando em mau funcionamento geral. Portanto, a confiabilidade das conexões é crucial para a operação adequada do robô SLC.

Os testes em condições adversas e os testes de conexão demonstram a importância de considerar influências ambientais e operacionais na operação do robô SLC. É essencial manter uma manutenção preventiva meticulosa do robô, garantindo que a bateria esteja devidamente carregada e que todos os componentes estejam corretamente conectados e funcionais para assegurar um melhor desempenho e evitar falhas durante as competições.

3.5 DISCUSSÃO DOS RESULTADOS

Os resultados apresentados neste capítulo, demonstram que o robô SLC "Avexadinho" atendeu aos requisitos projetados em termos de montagem, identificação do sistema, aplicação do controle e desempenho independente dos métodos alternativos usados, sendo capaz de operar de maneira eficiente em ambientes controlados e adversos, desde que mantenha um nível aceitável de bateria e não perca a conexão com os sensores. A operação do controle PI aplicado em condições adversas e a adequação dos modelos matemáticos, apoiado pelos resultados dos experimentos realizados e comparados com as simulações, indicam que o sistema está bem projetado para aplicações práticas. O principal problema encontrado no projeto final é na realização de percursos onde curvas fechadas tenham um raio menor que a distância do centro de uma roda até seu sensor mais próximo, impossibilitando o controle de atuar por limitações construtivas, mas do modo como o controlador está integrado, esse problema é amenizado ao travar uma das rodas quando o robô sai completamente da linha, fazendo uma correção de percurso sem necessidade de uma ação de controle. O robô pode ser aplicado em um percurso completo, seguindo a linha conforme vídeo reproduzido no *link* https://youtube.com/shorts/lfERTCETmms.

3.6 CONCLUSÕES PARCIAIS

Neste capítulo, foram abordadas as etapas do desenvolvimento e da validação do robô SLC "Avexadinho": montagem, identificação do sistema e projeto do controlador. A montagem do robô destacou a importância de uma seleção cuidadosa de componentes mecânicos e eletrônicos para garantir a robustez e precisão na operação. A identificação do sistema, realizada com a *System Identification Toolbox* do MATLAB, resultou em um modelo ARX com 70% de precisão, adequado para os propósitos de controle. A aplicação do controle utilizou um controlador PI, ajustado para otimizar o tempo de acomodação e evitar oscilações. A transformação das equações de controle garantiu a operação eficiente do robô, lidando com atrasos e saturações.

Os testes realizados validaram a eficácia do robô em termos de identificação e controle do sistema, confirmando a adequação dos componentes e técnicas utilizadas. No próximo capítulo, serão apresentadas as conclusões do projeto, destacando as principais realizações, desafios enfrentados e sugestões para futuros trabalhos.

4 CONCLUSÕES

O desenvolvimento do robô seguidor de linha envolveu diversas etapas que agregaram à montagem do robô, à identificação do sistema e à aplicação do controle, que foram abordadas de forma que fosse possível compreender cada passo do processo. Das etapas globais, os principais resultados qualitativos foram:

- Montagem do Robô: a estrutura do robô foi construída utilizando componentes disponibilizados pela equipe Maracatronics. A integração dos sensores, motores, microcontrolador e outros componentes eletrônicos foi realizada com sucesso, resultando na construção de um robô funcional dentro dos critérios para ser apto a competir na maioria das competições na categoria de Seguidor de linha Pro.

- Identificação do Sistema: realizando a coleta de dados para um conjunto de testes de prova e utilizando ferramentas do *System Identification* do MATLAB, foi possível identificar que um modelo matemático de primeira ordem com atraso se destacou entre as demais possibilidades estudadas para representar o sistema. O modelo pôde também ser identificado em tempo contínuo, para que fosse feita uma análise sobre a conversão e como alternativa caso fosse necessário modelar um controlador mais complexo, visto que alterações no plano contínuo são mais intuitivas e visuais quando comparadas às alterações no plano discreto.

- Aplicação do Controle: com os modelos identificados, foi possível projetar um controlador do tipo PI eficaz na estabilização do robô, proporcionando um tempo de assentamento rápido e minimizando o sobressinal. Um controlador bem semelhante foi encontrado para os dois casos estudados, modelo digital e modelo contínuo em s posteriormente digitalizado, representando assim uma boa conversão e aproximação. Um controlador do tipo PID seria uma complexidade desnecessária, pois a possibilidade de um comportamento mais controlável viria ao custo de um modelo de planta fechada mais complexo, com uma maior chance de perder estabilidade devido a ruídos de medição. Testes práticos em diferentes cenários demonstraram que o robô com o controlador encontrado pode seguir linhas de forma precisa e estável dentro do tempo esperado, mantendo um desempenho satisfatório mesmo em condições adversas.

A partir dos três pontos descritos acima, pode-se afirmar que os objetivos específicos traçados na Introdução foram alcançados:

- A montagem do robô SLC foi feita de forma adequada e os componentes de sua montagem foram avaliados e aprovados em desempenho individual e coletivo.

- Os experimentos de coleta de dados foram executados com sucesso, de forma que fosse possível avaliar o comportamento do sistema em condições semelhantes à competitiva.
- Quanto à Coleta de Dados, o processo foi elaborado e documentado adequadamente, facilitando o transporte dos dados entre planilhas e ambientes de trabalho.
- Foi possível identificar um modelo que representasse o comportamento da planta com uma ordem e atraso compatíveis ao comportamento dos testes executados.
- Também foi possível validar esse modelo identificado usando métodos de validação cruzada e iteração.
- Usando o mapa do LGR para projetar o controlador, foi encontrado um controlador do tipo PI que atendia a critérios de estabilidade, velocidade de resposta e oscilação.
- A validação do controle foi feita na teoria, quando o resultado apresentou comportamentos compatíveis com as simulações, e na prática, quando o robô foi capaz de seguir um trajeto mais completo da forma esperada, feito que pode ser visto com mais detalhes no vídeo do *link* https://youtube.com/shorts/lfERTCETmms.
- Por fim, o sistema final também foi avaliado em condições adversas, apresentando resultados bastante positivos, demonstrando que o sistema encontrado é robusto e capaz de operar de forma eficaz desde que não tenha seu funcionamento comprometido.

Também é possível dizer que o trabalho pode ser usado como um guia que contribui para o campo de robótica móvel e controle de sistemas, e usado como um modelo de referência para futuros projetos de robôs seguidores de linha e sistemas semelhantes, tanto em termos de *hardware* quanto de *software*. A metodologia utilizada para a identificação do sistema e o projeto do controlador PI pode ser aplicada a outros sistemas robóticos, contribuindo para a literatura existente e oferecendo uma abordagem prática para engenheiros e pesquisadores, não se limitando a uma única equipe de robótica, dessa forma corroborando com a missão da equipe Maracatronics de difundir a robótica e fomentar o conhecimento científico.

Durante a elaboração dos pontos discutidos no trabalho, também foram encontrados pontos de discussão e análise que vão além do escopo inicial do trabalho, como o caso da identificação de um modelo para a planta usando o *systemIdentification* do MATLAB, que obteve um resultado mais satisfatório usando os dados de um único ensaio, do que usando os dados empacotados de todos os ensaios, o que pode representar uma limitação da ferramenta quando trabalhando com muitos dados de entrada, impactando na sua capacidade de generalização ao usar o algoritmo de otimização.

Entretanto, mesmo com os resultados positivos apresentados, os objetivos gerais não foram inteiramente atendidos, pois, mesmo que em relação ao modelo base do "Avexado", o robô SLC projetado tenha um controle mais preciso, com melhor desempenho nas curvas e centralização nas retas, a escolha de usar o motor com menor rotação, apresentado na seção 3.3.2.2, acabou por trocar velocidade por precisão. Ele foi capaz de operar com uma velocidade de cruzeiro de 80 cm/s, superando seus antecessores "Mói de Quento" (com uma velocidade estimada de 30 cm/s) e "Avexado" (com uma velocidade estimada de 50 cm/s), mas não sendo capaz de superar o robô SLC anterior "Mulambo" (com uma velocidade estimada de 120 cm/s). Como consequência, em uma pista maior, acaba sendo lento para fins de competição.

4.1 DESAFIOS E LIMITAÇÕES

Apesar dos sucessos alcançados no desenvolvimento do robô SLC "Avexadinho", houve alguns desafios e limitações.

Problemas mecânicos foram o primeiro limitador, pois, devido a restrições de tempo e demandas de outros projetos da equipe, parte da estrutura mecânica não pôde ser impressa conforme planejado, sendo substituída por uma placa de fenolite sem aplicação eletrônica; as rodas em silicone, que seriam mais largas para uma melhor aderência, não puderam ser refeitas por falta de material, assim, sendo necessário usar os modelos menores do projeto antecessor; o pouco espaço e falta de material para melhorar a pista restringiram bastante na realização dos experimentos e nas validações práticas dos modelos. Tais alterações acabam por impactar significativamente o desempenho do robô.

Na parte eletrônica, o maior desafio foi o investimento de tempo e dinheiro que foi necessário para adquirir e prototipar a placa de circuito impresso e os módulos anexos, além de corrigir os erros. Apesar de não afetar o controle do robô de forma direta, esses contratempos tomaram recursos de pessoal e financeiros que poderiam ser direcionados para a pesquisa e melhoria de componentes.

Por fim, o maior limitador foram os motores, que, ademais da baixa potência que levou ao dilema entre torque e velocidade, também foram os responsáveis pelo atraso de 40ms do robô, visto que os demais componentes somados têm um atraso da ordem de 1% do atraso total do robô, logo o motor é responsável por 99% do atraso. Portanto, a adotação de motores com maior potência, resposta mais rápida e menor atraso poderia ser o primeiro passo para melhorar o desempenho do robô. Outros componentes que poderiam ser alterados são:

- Bateria: a bateria poderia ser substituída por um modelo mais leve de mesma potência, mesmo que com uma menor capacidade, dado que as três tomadas de tempo levam, no máximo, dois minutos.

- Comprimento do Robô: um comprimento menor possibilitaria o robô a fazer curvas de menor raio sem desalinhar por condições mecânicas, e como o controle aplicado foi capaz de gerar um bom alinhamento nas retas, não é necessário um suporte tão grande, reduzindo a massa e apresentando ângulos mais expressivos para o controle.

- *Jumpers*: Em alguns casos, o *jumper* pode sofrer mau contato e desconexões em oscilações durante execução ou no manuseio do robô, comprometendo a integridade do controle e colocando em risco a prova, principalmente se acontecer com os sensores críticos. Usar circuitos impressos e módulos integrados pode evitar que esses problemas ocorram.

4.2 TRABALHOS FUTUROS

Para trabalhos futuros focados em melhorar o desempenho e a robustez do robô seguidor de linha "Avexadinho", vários caminhos podem ser tomados com base nos problemas e limitações encontrados:

- Motores: para melhorar a resposta do sistema, é interessante avaliar motores com menor atraso e maior potência. A análise e pesquisa de outros tipos de motores, como *brushless*, pode melhorar o desempenho significativamente.

- Melhoria da Estrutura Mecânica: pesquisar novos materiais e métodos de fabricação para a estrutura do robô, a fim de encontrar uma maior estabilidade e robustez. A impressão de peças personalizadas e o uso de rodas mais largas para maior aderência também podem ser pontos de foco.

- Gerenciamento de Energia e Bateria: um caminho seria criar um sistema de gerenciamento de energia para maximizar o uso da bateria e prolongar a duração do robô, como também avaliar o uso baterias de menor peso, mesmo que de menor capacidade.

- Controle Adaptativo: implementar técnicas usuais de controle adaptativo que ajustem automaticamente os parâmetros do controlador com base nas condições operacionais em tempo real pode melhorar a performance em diferentes tipos de pistas e condições do ambiente. Por exemplo, o desempenho diminui com a redução do SoC da bateria por conta, provavelmente, da redução da tensão que é aplicada aos motores. É possível ajustar (aumentar) os valores saturados de tensão aplicados aos motores conforme o SoC da bateria reduz.

- Simulação e Modelagem Avançada: utilizar técnicas de modelagem e simulações mais avançadas para prever o comportamento do robô em cenários mais complexos, como ferramentas de simulações 3D e realidade virtual, que podem ser exploradas

para avaliar o robô em diferentes ambientes. Uma alternativa mais acessível envolve usar uma pista real e definir que tipo de sinais os sensores produzem para diferentes velocidades de cruzeiro e pensar num controle mais complexo capaz de reagir a diversos cenários mais rapidamente.

- Redução do Tamanho: diminuir o comprimento e o peso do robô, mantendo o centro de massa baixo, devem melhorar sua velocidade de cruzeiro e sua capacidade de manobrar em curvas bruscas, aumentando a competitividade em provas de maior complexidade.

- Aprimoramento da Eletrônica: otimizar o design da placa de circuitos para reduzir o tamanho e melhorar a eficiência. Integrar melhor os componentes eletrônicos para reduzir falhas de conexão, como problemas com jumpers, substituindo o encaixe da faixa de sensores por um acoplamento direto.

- Uso de sensores laterais: como algumas competições apresentam marcações ao lado da pista em inícios e finais de curvas é possível criar um controle para curvas e outro para retas e usar sensores laterais para identificar essas marcações, chaveando os controladores nesses casos.

- Estratégia de *anti-windup*: mesmo que no projeto do controlador, foi obtido um sistema sem sobressinal, na prática o robô passou um pouco da referência, isso pode ser um efeito do *wind-up*, que ocorre no integrador discreto devido à saturação na ação de controle. É necessário avaliar se o sistema de controle está saturando e por quanto tempo, e se a implementação de uma estratégia de *anti-windup* deve ser considerada nesse caso.

- Variabilidade de Métodos: Refazer esse trabalho usando métodos adequados diferentes dos usados nesse trabalho, como a modelagem em caixa-cinza, uso de outros sensores como câmera, a identificação pela resposta em frequência, outro *design* de experimentos para coleta de dados, algum método de sintonização diferente ou a implementação de um controlador com termo derivativo ou de outro tipo. Todos são formas alternativas de trilhar os passos da elaboração desse trabalho, porém de formas distintas, que podem resultar em controles até mais eficientes que o resultado desse trabalho.

Além de trabalhos focados em melhorar o desempenho de um robô seguidor de linha ou dispositivo controlado, outras linhas de estudo podem surgir, como o estudo da ferramenta *systemIdentification* ou outras ferramentas do MATLAB, quando tratando um número elevado de dados para alcançarem uma correta identificação.

REFERÊNCIAS

ASSECO CEIT, A.S. *Tow Tractor AGVs*. 2014. Disponível em: <https://www.asseco-ceit.com/en/agv-systems/agv-truck/>. Acesso em: 10/07/2024.

CANDIDO, G. *ROBÔ SEGUIDOR DE LINHA*. 2018. Disponível em: <https://portal.vidadesilicio.com.br/robo-seguidor-de-linha/>. Acesso em: 10/07/2024.

CHAOQUAN LI; XUESHAN, G. K.-L. Smooth Control the Coaxial Self-Balance Robot Under Impact Disturbances. *International Journal of Advanced Robotic Systems*, n. 1, p. 1–9, Jan 2011.

CHI HAI MOTOR CO. LTD. *DC Core Driving Gear Motor datasheet*. 2020. Disponível em: <https://cdn-shop.adafruit.com/product-files/4640/n20+motors_C15011+6V.pdf>. Acesso em: 10/07/2024.

ESPRESSIF SYSTEMS CO. LTD. *ESP32 Series datasheet*. 2024. Disponível em: <https://www.espressif.com/en/support/documents/technical-documents>. Acesso em: 10/07/2024.

FITZGERALD, A. E.; AL et. *Electric Machinery*. [S.l.]: McGraw-Hill, 1975.

HASAN KAZI MAHMUD; NAHID, A.-A. M. A. A. Implementation Of Autonomous Line Follower Robot. *International Conference on Informatics, Electronics Vision (ICIEV)*, p. 865–869, 2012.

INOVA DAMEC. *Maracatronics*. 2017. Disponível em: <https://damecanicaufpe.wixsite.com/damec/maracatronics>. Acesso em: 10/07/2024.

ISERMANN R. ; MÜNCHHOF, M. *Identification of Dynamic Systems: An Introduction with Applications*. [S.l.]: Springer, 2011.

KEESMAN, K. J. *System Identification: An Introduction*. [S.l.]: Springer, 2011.

LANDAU, I. D.; ZITO, G. *Digital Control Systems: Design, Identification and Implementation*. [S.l.]: Springer, 2006.

LJUNG, L. *System Identification: Theory for the User*. [S.l.]: Pearson Education, 1998.

MATHWORKS INC. *MATLAB Documentation R2024a*. [S.l.], 2024. Disponível em: <https://www.mathworks.com>. Acesso em: 10/07/2024.

OGATA, K. *Engenharia de Controle Moderno (5ª ed.)*. [S.l.]: Pearson Universidades, 2010.

PAKDAMAN MEHRAN; SANAATIYAN, M. M. R. M. A line follower robot from design to implementation: Technical issues and problems. In: *The 2nd International Conference on Computer and Automation Engineering (ICCAE)*. [S.l.: s.n.], 2010.

POLOLU. *QTR-8A Reflectance Sensor Array*. 2014. Disponível em: <https://www.pololu.com/docs/pdf/0J12/QTR-8x.pdf>. Acesso em: 10/07/2024.

ROBOCORE. *Regras Seguidor de Linha*. 2023. Disponível em: <https://robocore-eventos.s3.sa-east-1.amazonaws.com/public/Regras+-+Seguidor+de+Linha.pdf>. Acesso em: 10/07/2024.

RoboCore. *RSM Challenge 2023*. 2023. Disponível em: <https://events.robocore.net/rsm-2023/>. Acesso em: 29/08/2024.

ROBOCUP. *Vai começar a OBR 2024!* 2024. Disponível em: <https://obr.robocup.org.br/2024/03/25/vai-comecar-a-obr-2024/>. Acesso em: 10/07/2024.

SANJAYA ALI ; MAWENGKANG, H. . E. S. . Z. M. Stability Of Line Follower Robots With Fuzzy Logic and Kalman Filter Methods. *Journal of Physics: Conference Series*, 2019.

SHETH., P. A. P. H. B. S. S. A MECHATRONICS DESIGN OF A LINE TRACKER ROBOT USING ZIEGLER NICHOLS CONTROL TECHNIQUE FOR P, PI AND PID CONTROLLERS. *International Mechanical Engineering Congress (IMEC - 2014)*, n. 1, p. 529–532, Jun 2014.

SSI SCHAEFER. *AGV Weasel*. 2016. Disponível em: <https://www.ssi-schaefer.com/pt-br/produtos/transporte/veiculos-automaticamente-guiados/automated-guided-vehicle-weasel--1407364>. Acesso em: 10/07/2024.

TOSHIBA. *TB6612FNG datasheet*. 2007. Disponível em: <https://www.sparkfun.com/datasheets/Robotics/TB6612FNG.pdf>. Acesso em: 10/07/2024.

TURNIGY POWER SYSTEMS. *Turnigy nano-tecnologia 1000mAh 3S 20-40C Lipo AIRSOFT pacote*. 2024. Disponível em: <https://hobbyking.com/pt_pt/turnigy-nano-tech-1000mah-3s-20-40c-lipo-airsoft-pack.html>. Acesso em: 10/07/2024.

ÅSTRöM, K. J.; MURRAY, R. *Feedback Systems: An Introduction for Scientists and Engineers*. [S.l.: s.n.], 2006.

APÊNDICE A – CÓDIGO DO SEGUIDOR DE LINHA

```
#define _TIMERINTERRUPT_LOGLEVEL_     0
#define TIMER_BASE_CLK 1000000*TIMER_DIVIDER
#include <ESP32TimerInterrupt.h>
#include <SparkFun_TB6612.h>

#define AI1N1 16 //portas relacionadas ao motor 1
#define AI1N2 4
#define PWM1A 0
#define BI1N1 17 //portas relacionadas ao motor 2
#define BI1N2 5
#define PWM1B 18
#define STBY 13
const int offsetA = 1;
const int offsetB = 1;

Motor motor1 = Motor(AI1N1, AI1N2, PWM1A, offsetA, STBY);
Motor motor2 = Motor(BI1N2, BI1N1, PWM1B, offsetB, STBY);

#define D1 14          //pinagem dos sensores
#define D2 27
#define D3 26
#define D4 25
#define D5 33
#define D6 32
#define D7 35
#define D8 34
#define SD 21
#define SE 15

int stt1,stt2,stt3,cont;

int s[8];//vetores de sensores usados na calibração e leitura
int lim[8] = {3500,3500,3500,3500,3500,3500,3500,3500};
int maxi[8] = {0,0,0,0,0,0,0,0};
int mini[8] = {4095,4095,4095,4095,4095,4095,4095,4095};

float pos,pos1,vel,vel1;  // variáveis do controle
int quan;
int sai = 100;

```

```
//-------------------------FUNÇÕES--------------------------
//---------Função de leitura dos sensores e conversão A/D---
void leitura(){
      s[0] = analogRead(D1);
      s[1] = analogRead(D2);
      s[2] = analogRead(D3);
      s[3] = analogRead(D4);
      s[4] = analogRead(D5);
      s[5] = analogRead(D6);
      s[6] = analogRead(D7);
      s[7] = analogRead(D8);

      for(int i = 0;i<8;i++){
        if(s[i]<lim[i]){
          pos+=-2*i+7;
          quan++;
        }
      }
      if(s[0]<lim[0])sai=100;
      if(s[5]<lim[5])sai=-100;
      if(quan>0){
        pos=(pos/quan);
        if(abs(pos)>6) pos=8.27;
        pos=pos*10.02;
      }else{
        pos=sai;
      }
      quan=0;
}
void calibração(){//----- Rotina de calibração--------------
      for(int i=0;i<1000;i++){
        leitura();
        motor1.drive(131+sai*1.24);
        motor2.drive(131-sai*1.24);
        for(int j=0;j<8;j++){
          if(s[j]<mini[j]) mini[j]=s[j];
          if(s[j]>maxi[j]) maxi[j]=s[j];
        }
        delay(5);
      }
      for(int j=0;j<8;j++){
        if (maxi[j]-mini[j]>=1000) lim[j]=(mini[j]+3*maxi[j])/4;
        if (maxi[j]-mini[j]<1000) lim[j]=100;
      }
}
```

```
//------------ Rotina de detecção do fim do percurso--------
void percurso(){
      stt1 = digitalRead(SE)+2*digitalRead(SD);
      if(stt1!=stt2){
        if(stt3==3 && stt2==2 && stt1==3){
          cont++;
          if(cont==2){
            delay(20);
            motor1.drive(0);
            motor2.drive(0);
            while(1);
        }}
        stt3=stt2;
        stt2=stt1;
      }
 }

//------------------- Rotina do controle-------------------
bool IRAM_ATTR TimerH(void * timerNo){
    leitura();
    if(abs(pos)==100){
      motor1.drive(131+pos*1.24);
      motor2.drive(131-pos*1.24);
    }else if(pos>=0){
      motor1.drive(255,0);
      vel=255-(vel1+(0.78627)*(pos-0.999*abs(pos1)))*2.55;//PI Digital
      if (vel<35) vel=0;
      if (vel>255) vel=255;
      motor2.drive(vel,0);
    }else{
      motor2.drive(255,0);
      vel=255-(vel1+(0.78627)*(abs(pos)-0.999*abs(pos1)))*2.55;//PI Digital
      if (vel<35) vel=0;
      if (vel>255) vel=255;
      motor1.drive(vel,0);
    }
    vel1=vel;
    pos1=pos;
    pos=0;
}

```

```
127 //-----------SETUP E LOOP-----------------
128
129 #define Interv  5
130
131 ESP32Timer ITimer0(0);
132
133 void setup() {
134   pinMode(PWM1A, OUTPUT);
135   pinMode(AI1N1, OUTPUT);
136   pinMode(AI1N2, OUTPUT);
137   pinMode(PWM1B, OUTPUT);
138   pinMode(BI1N1, OUTPUT);
139   pinMode(BI1N2, OUTPUT);
140
141   calibração();
142
143   if (ITimer0.attachInterruptInterval(Interv * 1000, TimerH)){
144       Serial.print(F("Starting  ITimer0 OK"));
145   }else Serial.println(F("Can't be set. Select another timer"));
146   forward(motor1,motor2,255);
147 }
148
149 void loop() {
150       percurso();
151       delay(10);
152     }
```

APÊNDICE B – CÓDIGO DA COLETA DE DADOS

```
#include <SparkFun_TB6612.h>
#include "BluetoothSerial.h"

#define AI1N1 16
#define AI1N2 4
#define PWM1A 0
#define STBY 13

#define BI1N1 17
#define BI1N2 5
#define PWM1B 18

#define freio 100  //---proporcional de freio do motor(0-100)%
#define vel int(255-2.55*freio)

const int offsetA = 1;
const int offsetB = 1;

float tempo[2100],entrada[2100],saida[2100];
int sm[2100];

Motor motor1 = Motor(AI1N1, AI1N2, PWM1A, offsetA, STBY);
Motor motor2 = Motor(BI1N2, BI1N1, PWM1B, offsetB, STBY);

int indx;

#define D1 14
#define D2 27
#define D3 26
#define D4 25
#define D5 33
#define D6 32
#define D7 35
#define D8 34

int s[8];
int volta = 2;
int lim[8] = {3500,3500,3500,3500,3500,3500,3500,3500};
float pos,temp;
int quan,sai;
const char *pin = "1234"; // Change this to more secure PIN.

```

```
String device_name = "ESP32-BT";

#if !defined(CONFIG_BT_ENABLED)||!defined(CONFIG_BLUEDROID_ENABLED)
#error Bluetooth isnt enabled! Please run `make menuconfig` to enable it
#endif

#if !defined(CONFIG_BT_SPP_ENABLED)
#error Serial Bluetooth not available.Its only available for ESP32 chip.
#endif

BluetoothSerial SerialBT;

void leitura(){
      s[0] = analogRead(D1);
      s[1] = analogRead(D2);
      s[2] = analogRead(D3);
      s[3] = analogRead(D4);
      s[4] = analogRead(D5);
      s[5] = analogRead(D6);
      s[6] = analogRead(D7);
      s[7] = analogRead(D8);

      for(int i = 0;i<8;i++){
        if(s[i]<lim[i]){
          pos+=-2*i+7;
          quan++;
        }
      }
      if(s[0]<lim[0])sai=100;
      if(s[7]<lim[5])sai=-100;
      if(quan>0){
        pos=(pos/quan);
        if(abs(pos)>6)pos=8.27*pos/abs(pos);
        pos=pos*10.02;
      }else{
        pos=sai;
      }
      quan=0;
}

void setup() {

  Serial.begin(9600); //115200
  SerialBT.begin(device_name); //Bluetooth device name
  Serial.printf("The device \"%s\" is started.", device_name.c_str());
  #ifdef USE_PIN
  SerialBT.setPin(pin);
  Serial.println("Using PIN");
```

```
  #endif

  pinMode(PWM1A, OUTPUT);
  pinMode(AI1N1, OUTPUT);
  pinMode(AI1N2, OUTPUT);

  pinMode(PWM1B, OUTPUT);
  pinMode(BI1N1, OUTPUT);
  pinMode(BI1N2, OUTPUT);
  temp=millis();
}

void loop() {
  if(SerialBT.read()=='s'){
    forward(motor1,motor2,255);
    indx = 0;
    temp=millis();
    for(int n = temp;n<(temp+1200);){
      leitura();
      tempo[indx]=millis();
      entrada[indx]=0;
      saida[indx]=pos;
      sm[indx]=(4-s[0]/1000)*100000000+(4-s[1]/1000)*10000000+
+(4-s[2]/1000)*1000000+(4-s[3]/1000)*100000+(4-s[4]/1000)*10000+
+(4-s[5]/1000)*1000+(4-s[6]/1000)*100+(4-s[7]/1000)*10+(4-s[8]/1000);
      pos=0;
      delay(5);
      indx++;
      n=millis();
    }
    //motor1.drive(vel,0);//descomentado para testes para o outro lado
    motor2.drive(vel,0);//comentado para testes para o outro lado
    temp = millis();
    for(int n=temp;n<(temp+800);){
      leitura();
      tempo[indx]=millis();
      entrada[indx]=float(255-vel)/2.55;//degrau
      saida[indx]=pos;
      sm[indx]=(4-s[0]/1000)*100000000+(4-s[1]/1000)*10000000+
+(4-s[2]/1000)*1000000+(4-s[3]/1000)*100000+(4-s[4]/1000)*10000+
+(4-s[5]/1000)*1000+(4-s[6]/1000)*100+(4-s[7]/1000)*10+(4-s[8]/1000);
      pos=0;
      delay(5);
      indx++;
      n=millis();
    }
```

```
    motor2.drive(0,0);
    motor1.drive(0,0);
    for(int e=0;e<indx;e++){
      SerialBT.print(tempo[e]);
      SerialBT.print('/');
      SerialBT.print(entrada[e]);
      SerialBT.print('/');
      SerialBT.print(saida[e]);
      SerialBT.print('/');
      SerialBT.println(sm[e]);
    }
  }
  SerialBT.println("\n");
  SerialBT.println(volta);
  volta++;
  delay(200);
}
```

APÊNDICE C – LINKS EXTERNOS

Drive do material do trabalho:
drive.google.com/drive/folders/1DYkIp1gq6hbXuxb_2r1R1bmvQMyZ3Z30?usp=sharing