CENTRO FEDERAL DE EDUCAÇÃO TECNOLÓGICA DE MINAS GERAIS

CURSO DE ENGENHARIA DE CONTROLE E AUTOMAÇÃO

ELIANFRANCIS SILVEIRA DE SOUZA

SISTEMAS DE CONTROLE E NAVEGAÇÃO PARA ROBÔS DA CATEGORIA IEEE VSSS

Leopoldina

2026

ELIANFRANCIS SILVEIRA DE SOUZA

SISTEMAS DE CONTROLE E NAVEGAÇÃO PARA ROBÔS DA CATEGORIA IEEE VSSS

Trabalho de Conclusão de Curso apresentado ao Curso de Engenharia de Controle e Automação do Centro Federal de Educação Tecnológica de Minas Gerais, do Campus Leopoldina, como parte do requisito para obtenção do título de Bacharel em Engenharia de Controle e Automação.

Orientador: Accacio Ferreira dos Santos Neto

Coorientador: Vinícius Barbosa Schettino

Leopoldina

2026

Souza, Elianfrancis Silveira de.
S729s Sistemas de controle e navegação para robôs da categoria IEEE VSSS. / Elianfrancis Silveira de Souza. – 2027.
126 f. : il., gráfs.; figs.

Orientador: Accacio Ferreira dos Santos Neto.
Coorientador: Vinícius Barbosa Schettino.
Bibliografia: f. 85-86.

Trabalho de Conclusão de Curso (Graduação) - Centro Federal de Educação Tecnológica de Minas Gerais, *Campus* Leopoldina, apresentada a curso de Engenharia de Controle e Automação. Departamento de Eletroeletrônica, 2026.

1. Sistemas de controle. 2. Robótica. 3. Controladores. 4. Visão computacional . 5. Trajetória. 6. Previsão. I. Santos Neto, Accacio Ferreira dos. II. Schettino, Vinícius Barbosa. III. Título.

CDU: 681.5

Elaboração da ficha catalográfica pela bibliotecária Luzia Adriana Damasceno, CRB 6º - 2305 / CEFET-MG *campus* Leopoldina.

MINISTÉRIO DA EDUCAÇÃO
CENTRO FEDERAL DE EDUCAÇÃO TECNOLÓGICA DE MINAS GERAIS
COORDENAÇÃO DO CURSO DE ENGENHARIA DE CONTROLE E AUTOMAÇÃO - LP

FOLHA DE APROVAÇÃO DE TCC Nº 3/2026 - CECALP (11.51.20)

Nº do Protocolo: NÃO PROTOCOLADO

Leopoldina-MG, 16 de junho de 2026.

ELIANFRANCIS SILVEIRA DE SOUZA

SISTEMAS DE CONTROLE E NAVEGAÇÃO PARA ROBÔS DA CATEGORIA IEEE VSSS

Trabalho de Conclusão de Curso apresentado ao Curso de Engenharia de Controle e Automação do Centro Federal de Educação Tecnológica de Minas Gerais, do Campus Leopoldina, como parte do requisito para obtenção do grau de Bacharel em Engenharia de Controle e Automação.

Aprovado em 17 de junho de 2026.

Banca Examinadora:

Prof. D. Eng. Accacio Ferreira do Santos Neto (CEFET-MG) - Presidente
Prof. D. Sc. Vinícius Barbosa Schettino (CEFET-MG) - Coorientador
Prof. D. Eng. Rodolfo Lacerda Valle (CEFET-MG)
Prof. D. Sc. Matusalém Martins Lanes (CEFET-MG)

Leopoldina
2026

(Assinado digitalmente em 17/06/2026 18:20)
ACCACIO FERREIRA DOS SANTOS NETO
PROFESSOR ENS BASICO TECN TECNOLOGICO
DEELP (11.61.04)
Matrícula: ###317#7

(Assinado digitalmente em 18/06/2026 14:29)
MATUSALEM MARTINS LANES
PROFESSOR ENS BASICO TECN TECNOLOGICO
DEELP (11.61.04)
Matrícula: ###943#7

(Assinado digitalmente em 24/06/2026 14:27)
RODOLFO LACERDA VALLE
PROFESSOR ENS BASICO TECN TECNOLOGICO
DEELP (11.61.04)
Matrícula: ###471#2

(Assinado digitalmente em 23/06/2026 09:10)
VINICIUS BARBOSA SCHETTINO
PROFESSOR ENS BASICO TECN TECNOLOGICO
DEELP (11.61.04)
Matrícula: ###138#3

Processo Associado: 23062.030941/2026-96

Visualize o documento original em https://sig.cefetmg.br/public/documentos/index.jsp informando seu número: **3**, ano: **2026**, tipo: **FOLHA DE APROVAÇÃO DE TCC**, data de emissão: **16/06/2026** e o código de verificação: **2e3603aa77**

AGRADECIMENTOS

Quero iniciar meus agradecimentos ao meu pai, Francisco Oswaldo Lacerda de Souza, e à minha mãe, Eliana de Andrade Silveira Lacerda de Souza, pelo amor e apoio incondicional, e por sempre me guiarem com sabedoria, permitindo que eu seguisse firme nesta jornada.

Agradeço também ao meu irmão, Francislian Silveira de Souza, e à minha companheira, Ana Luisa Gonsalves Pires, por estarem presentes nos momentos mais desafiadores e por compartilharem comigo cada passo desta conquista.

Mesmo descrevendo com palavras, jamais esquecerei as entrelinhas da construção deste sonho, pois cada detalhe desta trajetória foi possível graças ao esforço coletivo de amigos, colegas e professores, lembrando que, assim como na engenharia, nada se constrói sozinho.

Gostaria de registrar, ainda, minha gratidão a todos que direta ou indiretamente participaram desta jornada, meu sincero obrigado.

RESUMO

Este trabalho descreve o desenvolvimento de um sistema integrado de navegação e controle para robôs móveis diferenciais da categoria *Very Small Size Soccer* (VSSS) da IEEE. O desenvolvimento dessa tecnologia constitui um desafio complexo, pois exige a integração de estratégias que lidem com percepção, restrições cinemáticas e a necessidade de adaptação rápida a trajetórias dinâmicas em um espaço de jogo limitado e sujeito a ruídos de medição. Para solucionar esse problema, propõe-se uma arquitetura baseada na abordagem de Campos Potenciais Artificiais (em inglês, *Artificial Potential Field*, APF). A estratégia combina o planejamento reativo para desvio de obstáculos em tempo real com leis de controle linear e angular acopladas. Isso garante que o veículo intercepte a bola e navegue de forma autônoma e suave, respeitando as restrições físicas dos motores e os limites do ambiente competitivo. A metodologia utiliza técnicas de visão computacional para rastreamento em tempo real do robô e da bola. O sistema de controle angular emprega um controlador Proporcional-Integral-Derivativo (PID) associado aos campos potenciais, integrando ainda algoritmos de predição de trajetória da bola e tomada de decisão para a condução ao gol adversário. Os experimentos práticos permitiram verificar o funcionamento integrado da visão computacional, da navegação e do controle em diferentes condições de posicionamento e movimentação, incluindo aproximação à bola, perseguição de uma referência móvel e reorientação diante de elevado erro angular.

Palavras-chave: robótica móvel; controle PID; visão computacional; campos potenciais; previsão de trajetória.

ABSTRACT

This work describes the development of an integrated navigation and control system for differential mobile robots in the IEEE Very Small Size Soccer Futebol de Robôs de Tamanho Muito Pequeno (em inglês, *Very Small Size Soccer*) (VSSS) category. Developing this technology is a complex challenge, as it requires the integration of strategies that address perception, kinematic constraints, and the need for rapid adaptation to dynamic trajectories in a limited playing space subject to measurement noise. To solve this problem, an architecture based on the Artificial Potential Fields (APF) approach is proposed. The strategy combines reactive planning for real-time obstacle avoidance with coupled linear and angular control laws. This ensures that the vehicle intercepts the ball and navigates autonomously and smoothly, respecting the physical constraints of the motors and the limits of the competitive environment. The methodology uses computer vision techniques for real-time tracking of the robot and the ball. The angular control system employs a PID structure associated with potential fields, also integrating ball trajectory prediction algorithms and decision-making for driving towards the opponent's goal. Practical experiments were used to verify the integrated operation of computer vision, navigation, and control under different positioning and motion conditions, including ball approach, tracking of a moving reference, and reorientation under a large angular error. Future work includes expanding the system to a multi-agent team, migrating the architecture to ROS 2, and incorporating artificial intelligence techniques for trajectory optimization and predicting opponent movements.

Keywords: Mobile Robotics. PID Control. Computer Vision. Potential Fields. Trajectory Prediction.

MINISTÉRIO DA EDUCAÇÃO
CENTRO FEDERAL DE EDUCAÇÃO TECNOLÓGICA DE MINAS GERAIS
COORDENAÇÃO DO CURSO DE ENGENHARIA DE CONTROLE E AUTOMAÇÃO - LP

TERMO DE RESPONSABILIDADE Nº 132/2026 - CECALP (11.51.20)

Nº do Protocolo: NÃO PROTOCOLADO

Leopoldina-MG, 16 de junho de 2026.

Eu, **ELIANFRANCIS SILVEIRA DE SOUZA**, matrícula **201723510297**, discente do Curso de Engenharia de Controle e Automação, declaro para os devidos fins, que o presente Trabalho de Conclusão de Curso, de título **SISTEMAS DE CONTROLE E NAVEGAÇÃO PARA ROBÔS DA CATEGORIA IEEE VSSS**, é de minha autoria e que estou ciente dos:

- Artigos 297 a 299 do Código Penal;
- Decreto-Lei n.º 2.848 de 7 de dezembro de 1940;
- Lei n.º 9.610, de 19 de fevereiro de 1998, sobre os Direitos Autorais;
- Regulamento Disciplinar Discente do CEFET-MG;
- E que plágio consiste na reprodução de obra alheia e submissão dela como trabalho próprio ou na inclusão, em trabalho próprio, de ideias, textos, tabelas ou ilustrações (quadros, figuras, gráficos, fotografias, retratos, lâminas, desenhos, organogramas, fluxogramas, plantas, mapas e outros) transcritos de obras de terceiros sem a devida e correta citação da referência.

Por ser verdade, e por ter ciência do referido artigo, firmo a presente declaração, isentando o (a) professor(a) **D. Sc. Accacio Ferreira do Santos Neto** (orientador), o (a) professor(a) **D. Sc. Vinícius Barbosa Schettino** (coorientador), os membros da banca examinadora e o Centro Federal de Educação Tecnológica de Minas Gerais, Campus Leopoldina, de qualquer responsabilidade.

(Assinado digitalmente em 23/06/2026 08:23)
Elianfrancis Silveira de Souza
DISCENTE
Matrícula: 2017#######7

Processo Associado: 23062.030941/2026-96

Visualize o documento original em https://sig.cefetmg.br/public/documentos/index.jsp informando seu número: **132**, ano: **2026**, tipo: **TERMO DE RESPONSABILIDADE**, data de emissão: **16/06/2026** e o código de verificação: **343b74a60d**

LISTA DE ILUSTRAÇÕES

Figura 1.1 – Aplicações do sistema de movimentação diferencial em diferentes plataformas móveis. 20
Figura 1.2 – Detecção de objetos utilizando visão computacional. 21
Figura 1.3 – Representação da categoria VSSS em ambiente de jogo. 22
Figura 1.4 – Representação da estratégia adotada para evitar colisões com os limites laterais do campo e conduzir a bola até o ponto objetivo. 25
Figura 2.1 – Diferentes vistas do robô desenvolvido: (a) vista superior e (b) vista em perspectiva. 27
Figura 2.2 – Dimensões do campo. 28
Figura 2.3 – Fluxo de aquisição e processamento de dados. 29
Figura 2.4 – Vistas do robô desenvolvido: (a) robô com uniforme de identificação; (b) placa de controle; (c) integração da eletrônica ao chassi com PCB e ESP32. 30
Figura 2.5 – Vista lateral do Chassi 31
Figura 2.6 – Esquemático do chassi e confecção via impressão 3D. 32
Figura 2.7 – Esquemático eletrônico da PCB. 33
Figura 2.8 – Roteamento das trilhas da PCB. 34
Figura 2.9 – Ferramentas utilizadas na preparação da usinagem da PCB: (a) FlatCAM e (b) bCNC. 35
Figura 2.10 – Etapas do processo de usinagem da PCB. 35
Figura 2.11 – Microcontrolador ESP32 DevKit V1 utilizado no robô. 36
Figura 2.12 – Módulo inercial GY-87 utilizado no robô. 37
Figura 2.13 – Mini-ponte H MX1616 utilizada no acionamento dos motores. 38
Figura 2.14 – Elementos utilizados no robô: (a) micromotor CC com encoder e (b) bateria de alimentação. 39
Figura 2.15 – Arquitetura de comunicação entre os principais módulos do sistema ROS. . . 41
Figura 2.16 – Etapas do processamento de imagens utilizado para aquisição, tratamento e reconhecimento de padrões. 43
Figura 2.17 – Etapas do processamento visual: (a) vista superior do campo processada e (b) calibração de cores para identificação dos elementos. 44
Figura 2.18 – Estrutura do campo experimental desenvolvido para os testes. 45

Figura 2.19–Conjunto de suporte da câmera aérea, composto por base de sustentação, tripé de microfone e câmera posicionada sobre o campo. 47
Figura 3.1 – Representação geométrica do problema de controle: a pose do robô é representada pelo centro dos círculos verde e amarelo; as coordenadas da bola são obtidas pelo centro do círculo laranja; o vetor alvo é indicado pela seta amarela; o vetor de orientação da trajetória é representado pela seta azul-clara; ; o vetor de repulsão é representado pela seta roxa e as restrições do campo VSSS são delimitadas pelo retângulo verde. 50
Figura 3.2 – Diagrama de blocos da estratégia de controle e navegação proposta. 52
Figura 3.3 – Representação espacial e postura do robô diferencial no plano. 53
Figura 3.4 – Restrições de movimento e velocidades nas rodas do robô diferencial. 54
Figura 3.5 – Representação das velocidades linear e angular no corpo do robô. 55
Figura 3.6 – Representação geométrica dos erros de posição e orientação do robô móvel em coordenadas polares. 56
Figura 3.7 – Diagrama de blocos do sistema de controle em malha fechada baseado em coordenadas polares. 57
Figura 3.8 – Resposta do controle angular para rotações em sentidos opostos. 60
Figura 3.9 – Variação do erro angular e correspondente resposta do controlador. 60
Figura 3.10–Resposta do controlador diante de variações abruptas do erro angular. 60
Figura 3.11–Comportamento da velocidade linear em função do erro angular e da distância normalizada ao alvo. 63
Figura 3.12–Representação das componentes atrativa e repulsiva utilizadas nos Campos Potenciais Artificiais. 65
Figura 3.13–Diagrama de blocos do sistema de controle em malha fechada ponto espacial corrigido 66
Figura 3.14–Interação das forças repulsivas e atrativas do campo potencial sobre o robô. . 67
Figura 3.15–Diferentes respostas da interação das forças repulsivas e atrativas do campo potencial sobre o robô. 68
Figura 4.1 – Configuração inicial do robô no primeiro cenário do Experimento 1. 71
Figura 4.2 – Trajetórias da bola (vermelho) e do robô (verde) no primeiro cenário do Experimento 1. 72
Figura 4.3 – Erro angular α (painel superior) e sinal de monitoramento do controlador (painel inferior) no Experimento 1. 73

Figura 4.4 – Referências de velocidade angular das rodas: `data[0]` em vermelho e `data[1]` em verde, no Experimento 1. 73
Figura 4.5 – Configuração inicial do segundo cenário do Experimento 1, com o robô alinhado e distante da bola. 74
Figura 4.6 – Trajetórias da bola e do robô no segundo cenário do Experimento 1. 75
Figura 4.7 – Erro angular (painel superior) e referências de velocidade das duas rodas (painel inferior) no segundo cenário do Experimento 1. 76
Figura 4.8 – Configuração inicial do Experimento 2, com a bola utilizada como referência móvel. 77
Figura 4.9 – Trajetórias da bola (vermelho) e do robô (verde) durante a perseguição circular no Experimento 2. 78
Figura 4.10 – Erro angular α (painel superior) e ganho da velocidade linear com ponto objetivo definido (painel inferior) durante o Experimento 2. 79
Figura 4.11 – Referências de velocidade angular das rodas durante o Experimento 2, limitadas ao intervalo de -20 a 20 rad/s. 80
Figura 4.12 – Configuração inicial do robô no Experimento 3, com elevado erro de orientação em relação ao alvo. 81
Figura 4.13 – Trajetória do robô durante a manobra de reorientação no Experimento 3. . . 82
Figura 4.14 – Erro angular (painel superior) e referências de velocidade angular das rodas (painel inferior) durante o Experimento 3. 83
Figura 4.15 – QR Code para acesso ao vídeo do teste de navegação utilizando campos potenciais. 83
Figura A.1 – Esquema do filtro de suavização e da previsão por curva de Bézier quadrática. 90
Figura A.2 – Exemplo de interpolação por curva de Bézier quadrática a partir de pontos sucessivos. 91
Figura B.1 – Representação das cores do robô e plano cartesiano, considerando seu deslocamento no plano 2D pelas coordenadas (X_r, Y_r) e sua rotação angular ω em torno do próprio eixo vertical Z_r, definido pelo centro entre as duas marcações de cores do robô. 92
Figura B.2 – Definição do tamanho total da área de detecção utilizada para preservar a orientação frontal do robô. 93
Figura C.1 – Exemplo de posições da bola e das componentes de velocidade estimadas por diferenças sucessivas. 95
Figura C.2 – Representação geométrica do ponto de condução posicionado atrás da bola em relação ao objetivo. 98
Figura C.3 – Comportamento do filtro exponencial aplicado ao erro angular. 100

Figura C.4–Ativação da correção lateral durante o domínio da bola. 101
Figura C.5–Resposta da correção lateral durante o domínio da bola. 101
Figura D.1–Sinal de controle convertido em PWM pelo ESP32. 105
Figura D.2–Erro angular define o Sinal de controle que são convertidos em PWM. 105
Figura F.1 – Arquitetura ROS 114
Figura G.1–Simulação do modelo cinemático direto de um robô diferencial no plano cartesiano utilizando diferentes velocidades angulares nas rodas. 116
Figura G.2–Simulação de navegação ponto a ponto de um robô diferencial utilizando controle proporcional de posição e orientação. 118
Figura G.3–Simulação de navegação utilizando campos potenciais artificiais para desvio de obstáculos de um robô diferencial. 120
Figura G.4–Simulação de navegação com ajuste dos parâmetros do campo potencial para realização de manobras mais complexas. 122
Figura H.1–QR Code para acesso ao vídeo do teste de navegação utilizando campos potenciais. 124
Figura H.2–QR Code para acesso ao vídeo explicação visual do sistema desenvolvido. . 124
Figura H.3–QR Code para acesso ao vídeo Calibração do sistema de visão computacional. 125
Figura H.4–QR Code para acesso ao vídeo Teste do controlador PID para perseguição da bola. 125
Figura H.5–Resultado do processamento de imagem realizado, com representação visual das informações extraídas para percepção e navegação do robô. 126

LISTA DE TABELAS

Tabela 2.1 – Lista de componentes utilizados no circuito eletrônico 36
Tabela 4.1 – Parâmetros de sintonia do sistema de controle e navegação do robô. 70
Tabela D.1 – Mapa de pinos do ESP32 para motores e sensores 105
Tabela E.1 – Funções do nó ROS do robô agrupadas por módulo. 107
Tabela E.2 – Parâmetros físicos do robô e seus significados. 108
Tabela E.3 – Parâmetros de controle do robô e seus significados. 108
Tabela E.4 – Parâmetros de filtragem e suavização do erro angular. 108
Tabela E.5 – Parâmetros de perseguição e condução do robô. 109
Tabela E.6 – Variáveis de estado do robô e seus significados. 109
Tabela E.7 – Parâmetros de campos potenciais do robô e seus significados. 110
Tabela E.8 – Principais tópicos ROS publicados pelo sistema do robô. 110
Tabela E.9 – Ferramentas e bibliotecas utilizadas no sistema 111
Tabela E.10–Recursos de controle e técnicas aplicadas na movimentação do robô. 112
Tabela E.11–Elementos visuais do OpenCV e suas funções no sistema. 113
Tabela E.12–Nós e módulos observados no sistema ROS e suas funções. 113

LISTA DE ABREVIATURAS E SIGLAS

L298N Driver de motor CC utilizado no robô

PCB Placa de Circuito Impresso (em inglês, *Printed Circuit Board*)

PID Controlador Proporcional-Integral-Derivativo

PWM Modulação por Largura de Pulso (em inglês, *Pulse Width Modulation*)

ROS Sistema Operacional de Robôs (em inglês, *Robot Operating System*)

VSSS Futebol de Robôs de Tamanho Muito Pequeno (em inglês, *Very Small Size Soccer*)

HSV Matiz, Saturação e Valor (em inglês, *Hue, Saturation and Value*)

LISTA DE SÍMBOLOS

α	Erro angular do robô em relação ao alvo
β	Erro de orientação final do robô
γ	Ângulo do vetor posição entre robô e alvo
Δx	Diferença de posição em x entre robô e objetivo
Δy	Diferença de posição em y entre robô e objetivo
ρ	Distância euclidiana entre robô e objetivo
v	Velocidade linear do robô
ω	Velocidade angular do robô
v_l	Velocidade da roda esquerda
v_r	Velocidade da roda direita
ω_l	Velocidade angular da roda esquerda
ω_r	Velocidade angular da roda direita
r	Raio nominal da roda
b	Distância entre as rodas (largura do eixo)
$\forall$	Para todo
$\in$	Pertence

SUMÁRIO

1	**INTRODUÇÃO**	**20**
1.1	Contextualização	22
1.2	Referencial Bibliográfico	23
1.3	Objetivos	24
1.4	Organização	26
2	**ROBÔS DA CATEGORIA VSSS**	**27**
2.1	Categoria IEEE VSSS	27
2.2	Robô desenvolvido	29
2.2.1	Estrutura mecânica	30
2.2.1.1	Chassi	30
2.2.2	Eletrônica	32
2.2.2.1	Placa de circuito impresso	32
2.2.2.2	ESP32	36
2.2.2.3	Módulo GY-87	37
2.2.2.4	Ponte H	38
2.2.2.5	Micromotor com Encoder e Bateria	39
2.3	Sistema de Comunicação e Visão Computacional	40
2.3.1	ROS	40
2.3.2	OpenCV	42
2.4	Campo desenvolvido da Categoria VSSS	44
2.4.1	Campo	44
2.4.2	Câmera e suporte de posicionamento	45
2.5	Considerações parciais	47
3	**SISTEMA DE CONTROLE E NAVEGAÇÃO**	**49**
3.1	Formalização do Problema de Controle	49
3.2	Estratégia Desenvolvida	51
3.2.1	Módulo de Cinemática Diferencial	52
3.2.2	Módulo de Controle Angular e de distância	56
3.2.2.1	Arquitetura da Malha de Controle	57

3.2.2.2 Implementação do Controlador Angular Controlador Proporcional-Integral-Derivativo (PID) 58
3.2.2.3 Controlador da Velocidade Linear 61
3.2.3 Módulo de Geração de Referência 64
4 RESULTADOS EXPERIMENTAIS 69
4.1 Configuração e Sintonia de Parâmetros 69
4.2 Experimento 1: Alinhamento Inicial ao Objetivo 70
4.3 Experimento 2: Perseguição de Trajetória Circular 76
4.4 Experimento 3: Reorientação sob Alta Variação Angular 80
5 CONCLUSÕES E TRABALHOS FUTUROS 84
5.1 Conclusões 84
5.2 Trabalhos futuros 84
REFERÊNCIAS 86
APÊNDICE A – Recepção das posições da bola 88
A.1 Nó de Detecção da Bola 88
A.1.1 Filtro de Suavização da Bola 88
APÊNDICE B – Nó de Detecção do Robô 92
APÊNDICE C – Implementação da estratégia e filtro do erro angular . . 94
C.1 Predição do ponto de interceptação 95
C.2 Ponto inicial de interceptação 95
C.3 Distância até o ponto previsto 95
C.4 Definição do *Lookahead* (horizonte de previsão) 96
C.5 Reposicionamento do alvo na direção prevista 96
C.6 Erro de distância e erro angular 97
C.7 Filtro Exponencial do Erro Angular 98
C.7.1 Equação do filtro 98
C.7.2 Como funciona 99
C.7.3 Finalidade 99
C.8 Correção Lateral Durante o Domínio da Bola 100
APÊNDICE D – Implementação do Controle no ESP32 102
D.1 Conexão Wi-Fi e Comunicação ROS 102
D.2 Recepção de sinais de controle 102
D.3 Geração do PWM 103

D.4 Publicação dos sensores 104
D.5 Resumo das cores e pinos 105
APÊNDICE E – Tabelas de Referenciação 106
E.1 Parâmetros físicos 108
E.2 Parâmetros de controle 108
E.3 Filtros e Suavizações 108
E.4 Perseguição e condução 109
E.5 Variáveis de estado 109
E.6 Campos potenciais 110
E.7 Tópicos ROS publicados 110
E.8 Ferramentas utilizadas no sistema 110
E.9 Recursos de controle utilizados na movimentação 112
E.10 Ferramentas visuais utilizadas com OpenCV 113
E.11 Nós e módulos observados no sistema ROS 113
APÊNDICE F – Arquitetura ROS rqt graph 114
APÊNDICE G – Testes de simulação do robô diferencial no MATLAB . 115
G.1 Teste 1: Validação do modelo cinemático direto 115
G.2 Teste 2: Navegação ponto a ponto utilizando controle proporcional 117
G.3 Teste 3: Navegação utilizando campos potenciais artificiais 118
G.3.1 Teste 3.1: Avaliação do comportamento em cenário com maior complexidade de navegação 120
APÊNDICE H – Vídeos complementares dos experimentos 123
H.1 Apresentação geral do trabalho 123
H.2 Explicação visual do sistema desenvolvido 124
H.3 Calibração do sistema de visão computacional 124
H.4 Teste do controlador PID para perseguição da bola 125
H.5 Considerações finais 126

1 INTRODUÇÃO

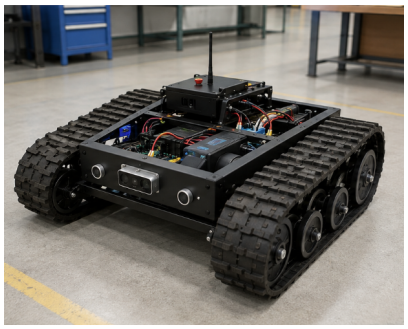
Os sistemas de movimentação diferencial estão presentes em diversas aplicações da engenharia, sendo amplamente utilizados em plataformas móveis que possuem como característica principal o controle independente dos elementos de propulsão. Esse modelo de movimentação apresenta grande relevância devido à sua simplicidade mecânica e capacidade de realizar manobras em espaços reduzidos, características que tornam essa arquitetura adequada para diferentes aplicações robóticas.

Plataformas baseadas nesse princípio podem ser encontradas em diversos contextos, como cadeiras de rodas motorizadas Figura 1.1a, veículos equipados com esteiras Figura 1.1b, embarcações com propulsão independente Figura 1.1c e robôs de inspeção industrial Figura 1.1d. Apesar das diferenças construtivas entre essas aplicações, o princípio de controle permanece semelhante: a variação das velocidades entre os lados esquerdo e direito permite modificar a trajetória, controlar a orientação e executar movimentos complexos (SICILIANO et al., 2010).

Figura 1.1 – Aplicações do sistema de movimentação diferencial em diferentes plataformas móveis.



(a) Cadeira de rodas robótica com sistema de movimentação diferencial.



(b) Plataforma móvel com sistema de movimentação diferencial por esteiras.



(c) Embarcação com propulsão independente utilizando movimentação diferencial.



(d) Robô de inspeção de silos de grãos com movimentação diferencial.

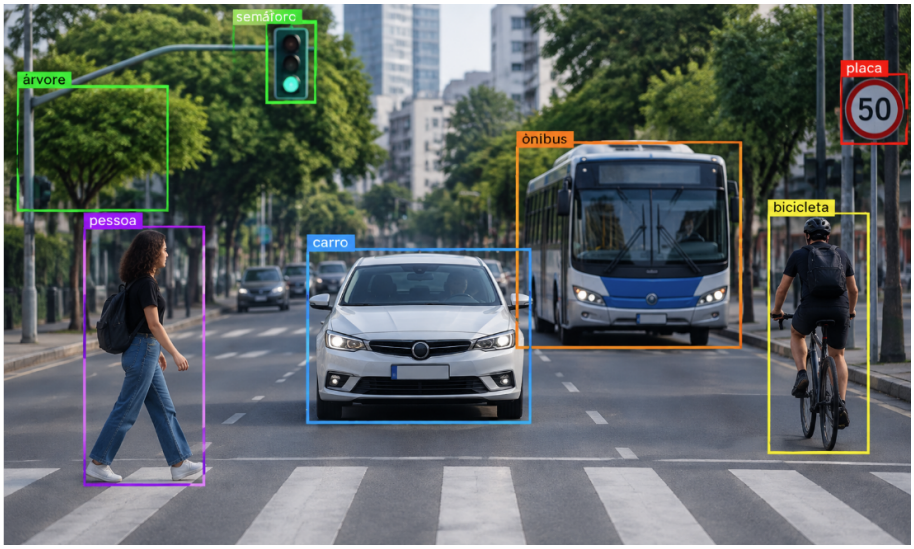
Fonte: Elaborado pelo autor, com aprimoramento gráfico realizado por ferramenta de inteligência artificial.

Dessa forma, técnicas desenvolvidas para robôs móveis diferenciais apresentam potencial

de aplicação em diferentes plataformas, uma vez que os modelos matemáticos, estratégias de controle e algoritmos de navegação podem ser adaptados conforme as características físicas e operacionais de cada sistema. Entretanto, para que esses veículos operem de maneira autônoma, torna-se necessário integrar mecanismos capazes de interpretar o ambiente, estimar sua posição e tomar decisões durante a movimentação.

Nesse contexto, a percepção do ambiente assume papel fundamental nos sistemas robóticos modernos. Técnicas de visão computacional permitem extrair informações a partir de imagens capturadas por câmeras, possibilitando a identificação de objetos, localização de referências, detecção de obstáculos e obtenção de informações utilizadas pelos algoritmos de navegação e controle Figura 1.2 (SZELISKI, 2010).

Figura 1.2 – Detecção de objetos utilizando visão computacional.



Fonte: Elaborado pelo autor, com aprimoramento gráfico realizado por ferramenta de inteligência artificial.

A integração entre percepção, planejamento e controle representa um dos principais desafios da robótica móvel, pois exige que o sistema seja capaz de transformar informações do ambiente em comandos de atuação precisos. Assim, estratégias desenvolvidas para robôs diferenciais podem contribuir para aplicações que demandam autonomia, como sistemas de inspeção, logística, assistência, exploração e veículos inteligentes.

Dentro desse cenário, os robôs móveis da categoria IEEE Very Small Size Soccer (VSSS) constituem uma plataforma experimental adequada para o desenvolvimento e validação dessas técnicas, ao apresentarem restrições semelhantes às encontradas em aplicações reais, como limitações mecânicas, necessidade de controle em tempo real, percepção do ambiente e tomada de decisão autônoma.

1.1 Contextualização

Os robôs da categoria VSSS Figura 1.3, têm conquistado notoriedade em ambientes educativos não apenas em função de suas dimensões reduzidas e custo acessível, mas, sobretudo, pelas oportunidades que proporcionam para experiências de aprendizagem (VÁZQUEZ-HURTADO; RODRIGUEZ-PADILLA, 2022).

Figura 1.3 – Representação da categoria VSSS em ambiente de jogo.



Fonte: Elaborado pelo autor, com aprimoramento gráfico realizado por ferramenta de inteligência artificial.

A adoção dessas plataformas permite integrar diversas disciplinas, abarcando conceitos de eletrônica, mecânica, programação e fundamentos da física (FERREIRA et al., 2023). É viável oferecer experiências educativas que abrangem programação e controle, sistemas embarcados, visão computacional, entre outros temas (FERREIRA et al., 2023).

No entanto, mesmo com sua adequação para propósitos educacionais, os robôs VSSS enfrentam diversos desafios de engenharia que precisam ser superados para possibilitar modelagem e controle precisos. Seu design compacto impõe limitações severas à integração de componentes, enquanto a percepção eficaz, a locomoção e a robustez são imprescindíveis para garantir um bom desempenho em tempo real (BASSANI et al., 2020). Essas restrições influenciam diretamente o comportamento dinâmico do sistema, resultando em fenômenos complexos como o deslizamento longitudinal, o escorregamento lateral e o atrito dinâmico das rodas que devem ser obrigatoriamente considerados no desenvolvimento de modelos confiáveis para controle e navegação.

No que tange à modelagem dinâmica, esses obstáculos decorrem da intrincada interação entre os subsistemas mecânicos e eletrônicos do veículo. Fatores como o desgaste natural de

componentes e as variações bruscas de iluminação no ambiente de jogo são aspectos difíceis de mensurar com exatidão. Além disso, a fusão de sensores, que abrange odometria por encoders, giroscópios e câmeras do sistema de visão global, eleva significativamente a complexidade da estimação de estados, tornando o sistema suscetível a ruídos e erros acumulados de medição (FAYYAD et al., 2020).

Diante deste cenário, este trabalho foi concebido com a principal motivação de propor uma estratégia de controle e navegação eficiente para robôs da categoria VSSS. A intenção é estruturar uma arquitetura que combine campos potenciais ao controle angular e linear, permitindo que o veículo desvie de obstáculos em tempo real e em conformidade com as regras da categoria. Os fundamentos teóricos e os trabalhos que servem de base para esta proposta são detalhados a seguir no Referencial Bibliográfico.

1.2 Referencial Bibliográfico

Para entender como os estudos recentes resolvem os problemas de controle e desvio de obstáculos, esta seção apresenta os principais trabalhos que servem de base para este projeto. O foco está em pesquisas sobre navegação de robôs diferenciais na categoria VSSS, que segue as regras da *RoboCup Small Size League (SSL)* (AKIN et al., 2013). O texto foi organizado da seguinte forma: primeiro, abordam-se os trabalhos sobre identificação de modelos dinâmicos; depois, os controladores voltados para estabilidade; e, por fim, as estratégias de campos potenciais para desviar de obstáculos em tempo real.

O trabalho de Ribeiro et al. (2025) introduz uma metodologia de identificação do modelo dinâmico de robôs da categoria VSSS baseada em manobras operacionais padrão e estimação paramétrica via método *interior-point*. A pesquisa desenvolve múltiplos modelos em diferentes cenários e demonstra que a abordagem obtém representações dinâmicas consistentes com o comportamento real do robô, oferecendo uma base sólida para avanços em estratégias de controle e autonomia de sistemas móveis.

Complementarmente, Kouvakas, Koumboulis e Sigalas (2024) apresentam um controlador não linear destinado a robôs diferenciais, que emprega um desacoplamento de entrada/saída em duas etapas e comunicação sem fio parcial. Essa metodologia evidencia como abordagens sofisticadas de controle podem aprimorar a estabilidade, o rastreamento de trajetória e a coordenação de sistemas multiagentes em robôs móveis.

Por fim, Samodro (2023) propõem um algoritmo para o planejamento de trajetória fundamentado em campos potenciais artificiais, aplicável a robôs diferenciais em ambientes dinâmicos. A pesquisa elucida a definição de campos atrativos e repulsivos, além de transformar essas forças em comandos de velocidade linear e angular, possibilitando a navegação autônoma e a prevenção de colisões, servindo como referência para sistemas de controle reativos em contextos com múltiplos obstáculos móveis.

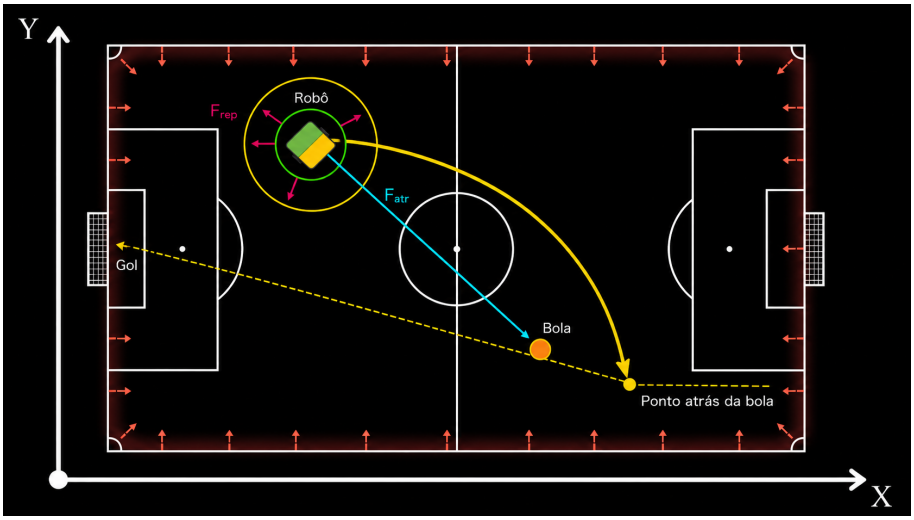
A partir dos trabalhos analisados, é possível ver que a pesquisa na área de robôs de futebol avançou bastante, mas de forma separada. Por exemplo, de Souza Sales Ribeiro et al. (2025) focaram em determinar o modelo matemático do robô VSSS; Kouvakas et al. (2024) criaram um controle focado em estabilidade; e Samodro et al. (2023) desenvolveram um desvio de obstáculos reativo rápido.

No entanto, os trabalhos analisados abordam partes distintas do problema. Ribeiro et al. (2025) concentram-se na identificação do modelo dinâmico; Kouvakas, Koumboulis e Sigalas (2024) priorizam a estabilidade e o rastreamento; e Samodro (2023) tratam do planejamento reativo por campos potenciais. Em uma partida de VSSS, essas funções precisam atuar de forma integrada: o robô deve ajustar sua trajetória diante de obstáculos e, simultaneamente, manter o controle de orientação e velocidade necessário para alcançar e conduzir a bola. Assim, este trabalho se justifica pela integração dessas etapas em uma estratégia reativa compatível com as limitações físicas do robô e com a necessidade de resposta em tempo real.

1.3 Objetivos

O objetivo geral deste trabalho consiste em desenvolver um sistema integrado de navegação e controle para robôs móveis diferenciais da categoria VSSS, fundamentado na abordagem de Campos Potenciais Artificiais (APF). A estratégia visa combinar o planejamento reativo de trajetórias para desvio de obstáculos em tempo real a leis de controle linear e angular acopladas, garantindo que o veículo intercepte a bola e navegue de forma autônoma e suave Figura 1.4, respeitando as restrições físicas dos motores e os limites operacionais do ambiente competitivo.

Figura 1.4 – Representação da estratégia adotada para evitar colisões com os limites laterais do campo e conduzir a bola até o ponto objetivo.



Fonte: Elaborado pelo autor, com aprimoramento gráfico realizado por ferramenta de inteligência artificial.

Para se atingir o objetivo geral deste trabalho delimitou-se os seguintes objetivos específicos:

1. **Desenvolver a plataforma física** (hardware) do robô móvel diferencial para a categoria VSSS, integrando o microcontrolador ESP32 aos atuadores e sistemas de alimentação;
2. **Implementar a arquitetura de comunicação** e processamento de dados integrando a visão computacional global ao robô por meio do Sistema Operacional de Robôs (em inglês, *Robot Operating System*) (ROS);
3. **Projetar as leis de controle linear e angular** em coordenadas polares, incorporando um controlador PID com ação *anti-windup* para garantir a estabilidade direcional do veículo;
4. **Formular a estratégia de navegação** baseada em Campos Potenciais Artificiais (APF) para atração ao alvo e desvio de obstáculos em tempo real;
5. **Desenvolver o algoritmo de tomada de decisão** para as fases de interceptação e condução da bola em direção ao gol adversário;
6. **Validar experimentalmente o sistema proposto** por meio de ensaios práticos progressivos.

1.4 Organização

A fim de melhor apresentar o trabalho desenvolvido, esta monografia está organizada em cinco capítulos, estruturados de forma sequencial para facilitar a compreensão do desenvolvimento do sistema robótico, desde a contextualização do problema até a análise dos resultados experimentais.

O Capítulo 2 descreve a categoria IEEE Very Small Size Soccer (VSSS) e a plataforma física desenvolvida, contemplando a estrutura mecânica, o sistema eletrônico, a comunicação, a visão computacional, o campo experimental e o suporte utilizado para posicionamento da câmera.

O Capítulo 3 aborda os sistemas de controle e navegação do robô, incluindo as representações espaciais utilizadas, a arquitetura geral de controle e as formulações matemáticas empregadas no processamento das posições, na estimativa da velocidade da bola, na predição do ponto de interceptação, na filtragem do erro angular e no controle PID. Também são apresentados o cálculo da velocidade linear, os campos potenciais, as forças repulsivas das bordas, a correção lateral, o modelo cinemático diferencial e os principais elementos da implementação, como funções, variáveis, tópicos ROS, ferramentas de controle e módulos de visão computacional.

O Capítulo 4 apresenta os experimentos realizados para avaliar a arquitetura proposta, considerando diferentes condições de posicionamento do robô e da bola, com análise da navegação, do controle angular, da interceptação e da condução da bola. Ao final, os resultados são apresentados, comparados e discutidos.

Por fim, o Capítulo 5 reúne as conclusões do trabalho, relacionando os resultados obtidos aos objetivos estabelecidos, além de apresentar as limitações identificadas e possibilidades de aprimoramento em trabalhos futuros.

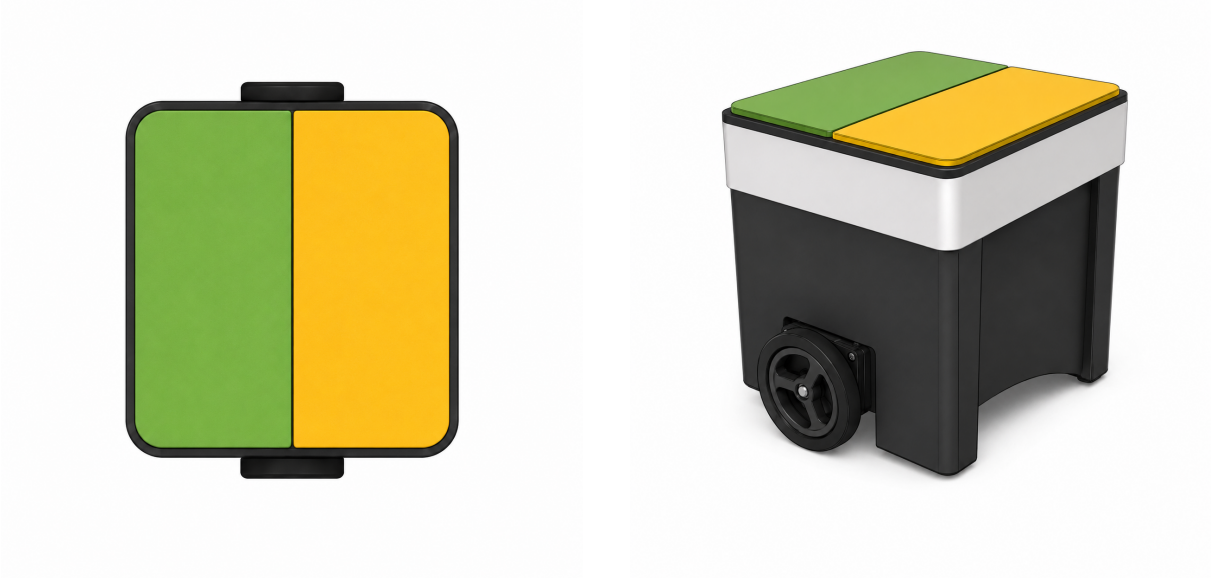
2 ROBÔS DA CATEGORIA VSSS

Este capítulo tem como objetivo apresentar o robô móvel da Categoria IEEE VSSS desenvolvido para pequenas competições de futebol de robô. Além disso, o intuito é apresentar alguns conceitos fundamentais sobre modelagem mecânica, reaproveitamento de recursos para novas finalidades, sistemas de comunicação e visão computacional.

2.1 Categoria IEEE VSSS

A categoria VSSS do IEEE consiste em uma modalidade de competição de robótica autônoma, cujo principal objetivo é incentivar o avanço de sistemas embarcados e estratégias de controle em robôs móveis de dimensões reduzidas. De acordo com as normas da categoria, o robô pode apresentar dimensões máximas de 7,5 × 7,5 × 7,5 cm, normalmente projeto em formato de cubo como ilustra a Figura 2.1.

Figura 2.1 – Diferentes vistas do robô desenvolvido: (a) vista superior e (b) vista em perspectiva.



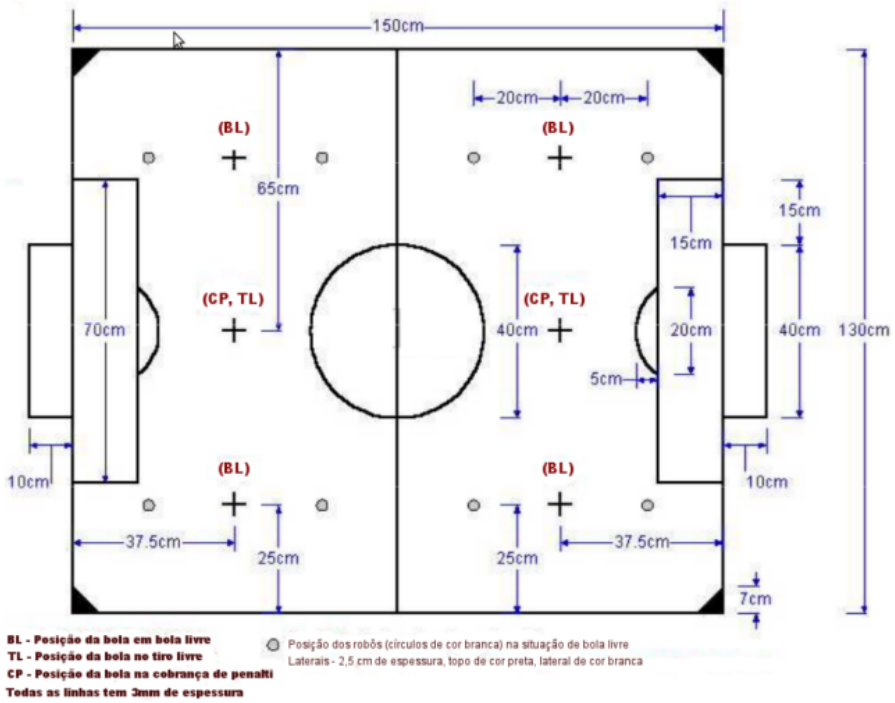
(a) Vista superior do robô. (b) Vista em perspectiva do robô.

Fonte: Elaborado pelo autor, com aprimoramento gráfico realizado por ferramenta de inteligência artificial.

Estes robôs competem numa arena que mede 150 cm × 130 cm, simulando um campo de futebol em miniatura, conforme ilustrado na Figura 2.2. Esta arena é monitorada por um sistema de visão aérea centralizado, posicionado acima do campo, responsável por identificar continuamente a bola e os robôs, ilustrados na Figura 2.3. Dessa forma, o computador central da

equipe processa os dados de posicionamento e envia comandos sem fio a cada robô (PINTO, 2021).

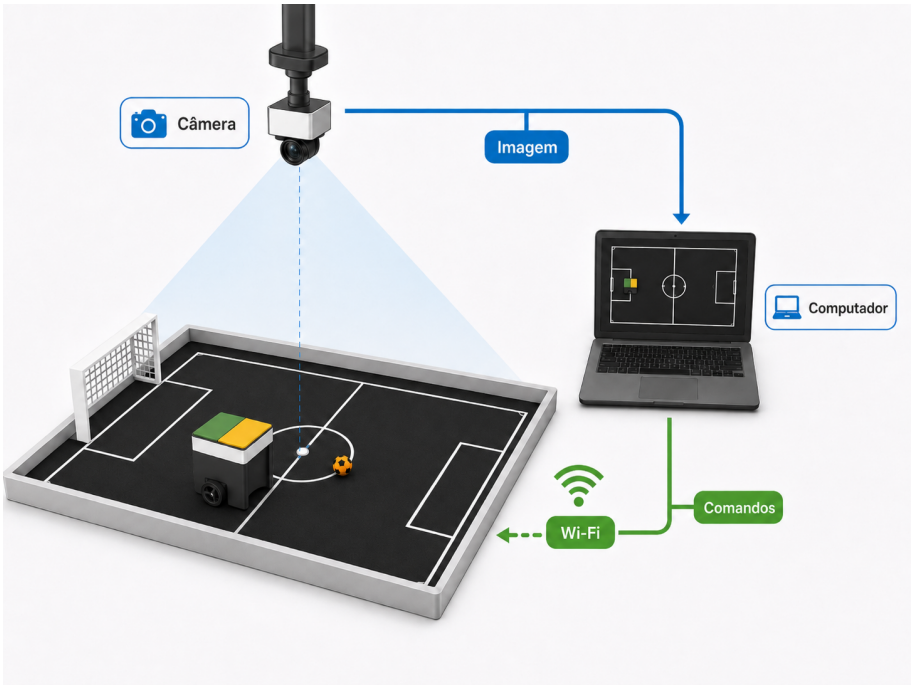
Figura 2.2 – Dimensões do campo.



Fonte: Adaptado de Pinto (2021).

A Figura 2.2 apresenta as dimensões de referência do campo utilizado como base para a montagem do ambiente experimental. A partir dessa geometria, o sistema de visão superior fornece ao computador as informações necessárias para localizar os elementos da partida. Esse fluxo de aquisição e envio de comandos é sintetizado na Figura 2.3.

Figura 2.3 – Fluxo de aquisição e processamento de dados.



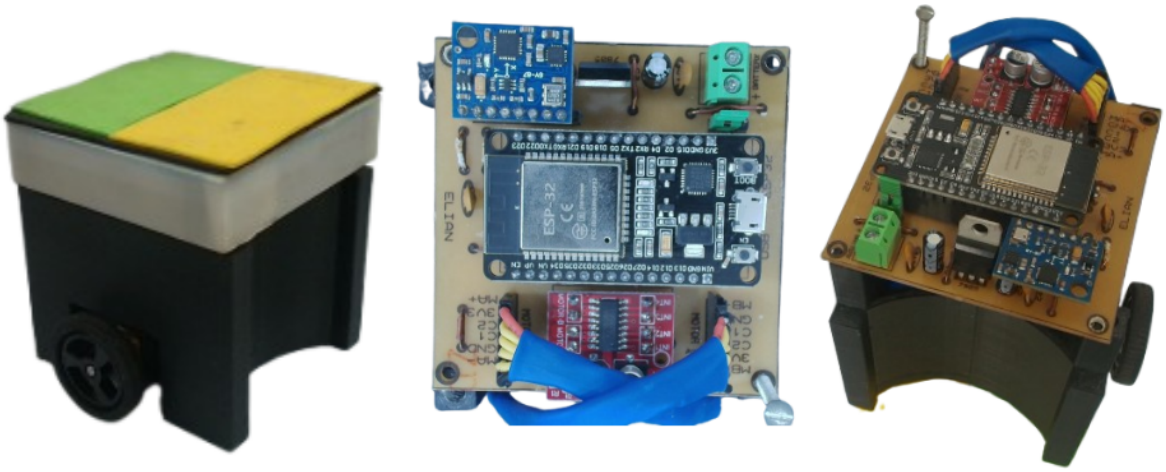
Fonte: Elaborado pelo autor, com aprimoramento gráfico realizado por ferramenta de inteligência artificial.

O fluxo apresentado evidencia a separação entre percepção, processamento e atuação: a câmera observa o campo, o computador processa as posições e gera as referências de movimento, e os comandos são transmitidos ao robô por comunicação sem fio. A seguir, apresenta-se a plataforma física utilizada para executar essa arquitetura.

2.2 Robô desenvolvido

De acordo com as normas estabelecidas para a categoria IEEE VSSS foi desenvolvido o robô mostrado na Figura 2.4. Esta seção apresenta a estrutura mecânica do robô, responsável por acomodar e sustentar os principais componentes do sistema, como a Placa de Circuito Impresso (em inglês, *Printed Circuit Board*) (PCB), os motores e a bateria.

Figura 2.4 – Vistas do robô desenvolvido: (a) robô com uniforme de identificação; (b) placa de controle; (c) integração da eletrônica ao chassi com PCB e ESP32.



(a) Robô com uniforme de identificação.

(b) Placa de controle.

(c) Integração da eletrônica ao chassi com PCB e ESP32.

Fonte: Autoria própria.

A Figura 2.4 fornece uma visão geral do conjunto construído. Nas subseções seguintes são detalhadas a estrutura mecânica e a eletrônica responsáveis pela montagem e pelo acionamento do robô.

2.2.1 Estrutura mecânica

A Figura 2.4 apresenta diferentes vistas do robô desenvolvido para a categoria IEEE VSSS. A estrutura mecânica foi projetada para acomodar e sustentar todos os componentes essenciais do sistema, incluindo a PCB[1], os motores, a bateria e o microcontrolador ESP32, garantindo rigidez estrutural, proteção mecânica dos componentes e estabilidade do conjunto durante a movimentação. A parte superior do robô, denominada camisa, apresenta cores de referência que permitem a identificação pelo sistema de visão computacional, facilitando o rastreamento e o controle da navegação do robô em tempo real

2.2.1.1 Chassi

O chassi do robô foi desenvolvido visando atender aos requisitos de mobilidade, estabilidade e compactação necessários para aplicações na categoria IEEE-VSSS. Sua estrutura foi projetada

[1] Ver Figura 2.8

de forma a acomodar os componentes eletrônicos, motores, sensores e sistema de alimentação de maneira organizada e funcional.

A construção do chassi foi realizada utilizando material leve e resistente, ABS, buscando reduzir o peso total do robô sem comprometer sua rigidez estrutural. O formato adotado foi planejado para facilitar a locomoção, o controle de movimento e a proteção dos componentes internos durante os testes e partidas.

O projeto do chassi também considerou a distribuição adequada do centro de massa, contribuindo para maior estabilidade nas mudanças rápidas de direção e durante a execução das estratégias de navegação e condução da Figura 2.5 e 2.6. Além disso, foram previstos espaços específicos para fixação da placa controladora, sensores, bateria e motores de acionamento

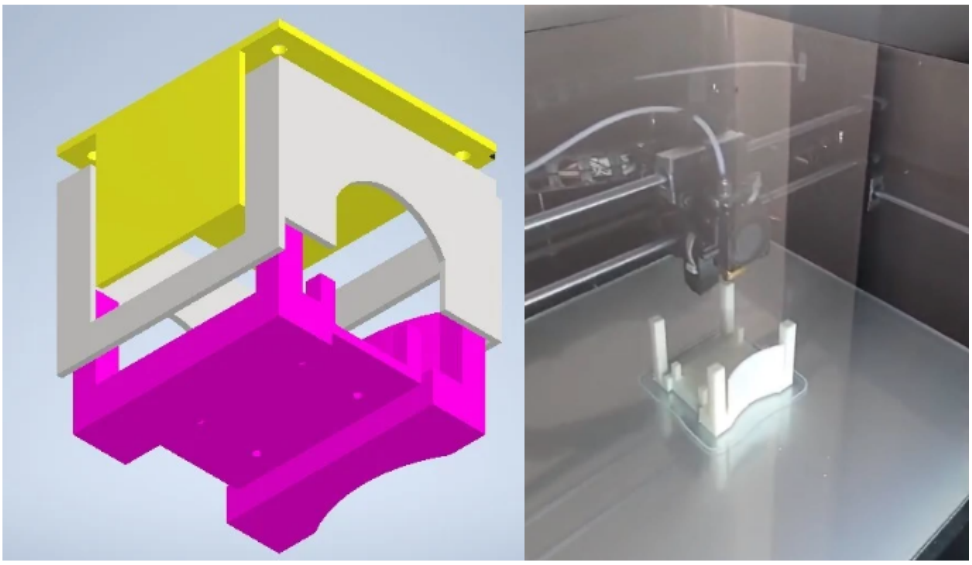
Embora o robô desenvolvido não siga integralmente todas as especificações oficiais da categoria IEEE-VSSS, sua estrutura apresenta características semelhantes às utilizadas em plataformas acadêmicas e experimentais de futebol de robôs, permitindo a realização de testes práticos e validação dos algoritmos de controle e visão computacional implementados no projeto.

Figura 2.5 – Vista lateral do Chassi



Fonte: Elaborado pelo autor, com aprimoramento gráfico realizado por ferramenta de inteligência artificial.

Figura 2.6 – Esquemático do chassi e confecção via impressão 3D.



(a) Esquemático do chassi do robô. (b) Confecção do chassi via impressão 3D.
Fonte: Autoria própria.

As Figuras 2.5 e 2.6 evidenciam a organização mecânica adotada e a fabricação do chassi por impressão 3D. Definida a estrutura física, a próxima etapa do projeto corresponde à integração dos circuitos responsáveis pelo processamento, alimentação e acionamento dos motores.

2.2.2 Eletrônica

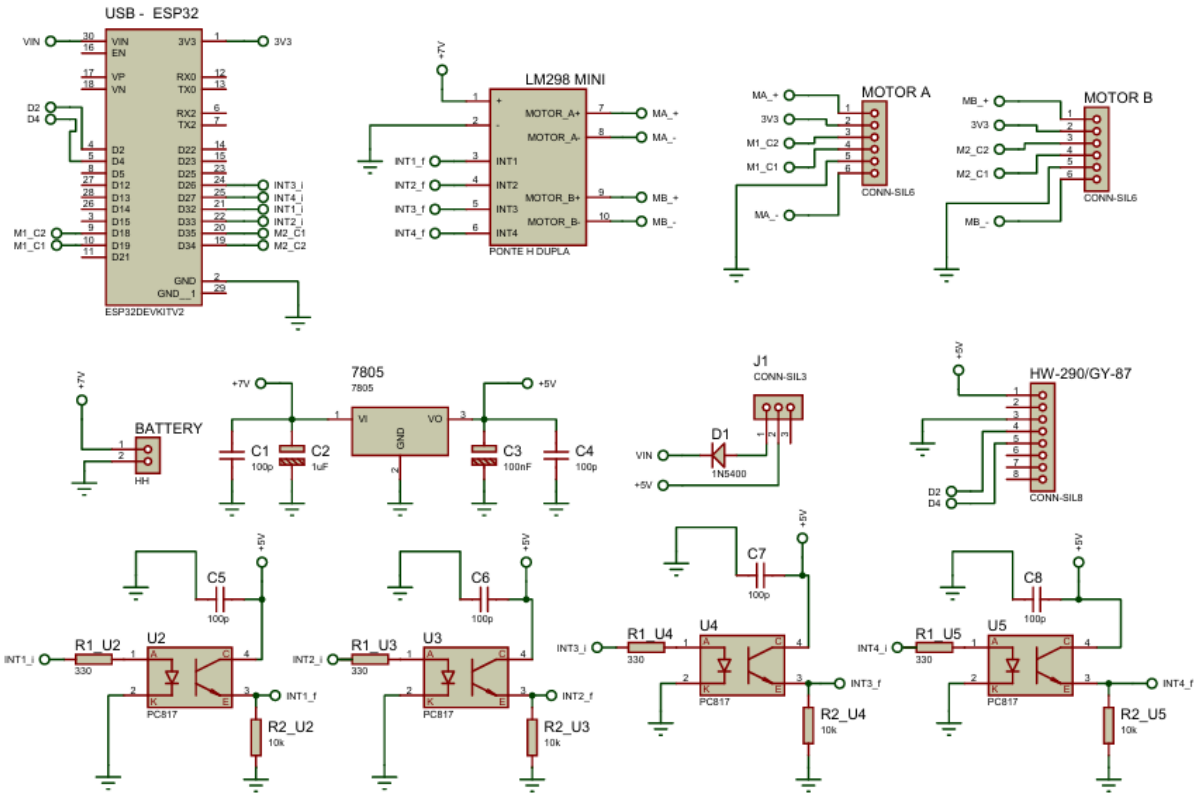
A eletrônica do robô é composta por uma PCB, responsável pela integração dos principais componentes eletrônicos utilizados no sistema. Nesta seção, serão apresentados os elementos que compõem a PCB, bem como as etapas relacionadas ao seu processo de fabricação, incluindo a usinagem da placa e a soldagem dos componentes.

2.2.2.1 Placa de circuito impresso

A PCB, apresentada na Figura 2.8, foi projetada para estabelecer as conexões elétricas e eletrônicas entre os componentes mostrados na Figura 2.7. O projeto do circuito eletrônico e o roteamento das trilhas foram realizados no software *Proteus*. O processo de fabricação envolveu a elaboração do esquemático, o roteamento das trilhas, a geração dos arquivos Gerber, a usinagem, a perfuração, a soldagem dos componentes e os testes funcionais. Os arquivos Gerber foram processados no *FlatCAM* para geração das trajetórias de usinagem e dos códigos G (*G-code*); posteriormente, o *bCNC* foi utilizado na interface com a máquina CNC durante a

fabricação física da placa. Após a produção, foram realizados ajustes mecânicos e elétricos para adequar a placa às dimensões do robô e verificar o funcionamento dos circuitos.

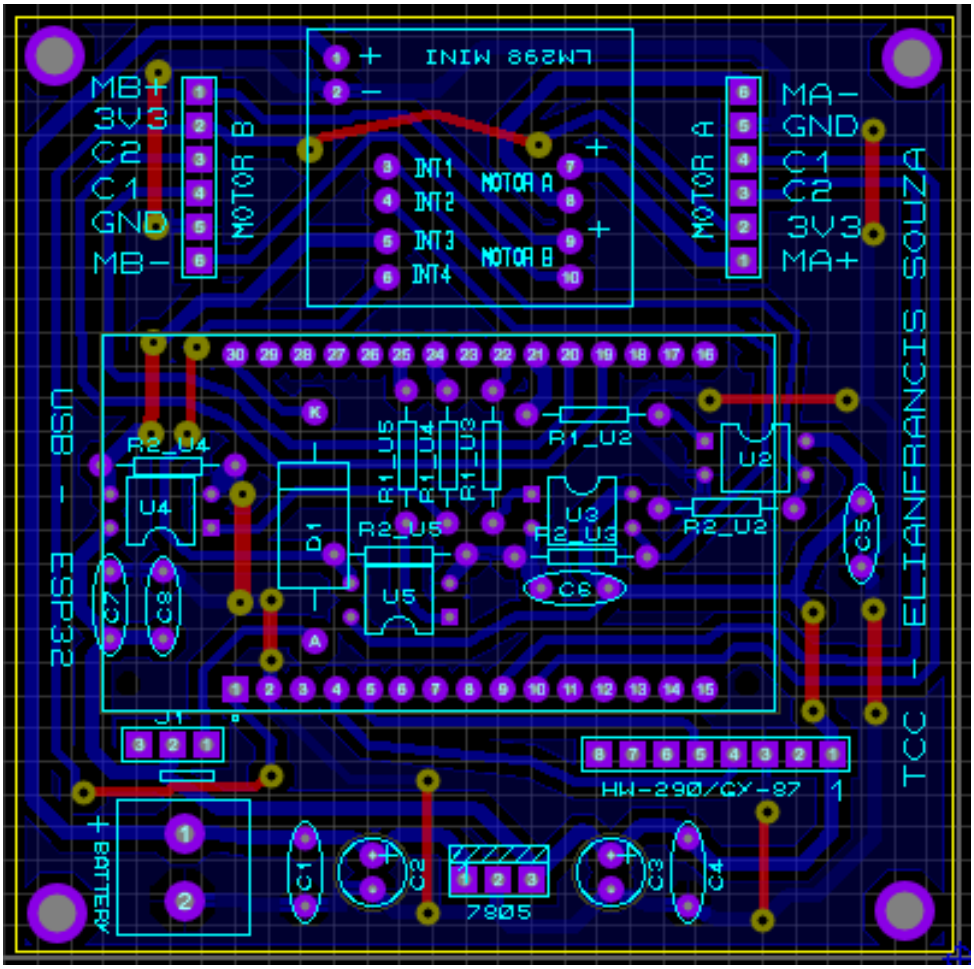
Figura 2.7 – Esquemático eletrônico da PCB.



Fonte: Autoria própria.

O esquemático da Figura 2.7 define as interligações entre os componentes do circuito. Com base nessas conexões, foi realizado o roteamento físico das trilhas, apresentado na Figura 2.8, utilizado posteriormente para gerar os arquivos de fabricação.

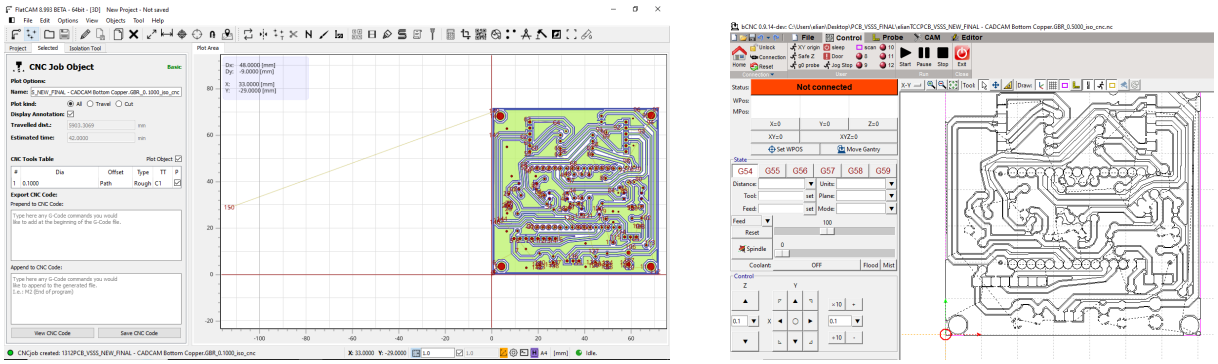
Figura 2.8 – Roteamento das trilhas da PCB.



Fonte: Autoria própria.

Após a definição do esquemático e do roteamento, os arquivos de fabricação foram preparados para a usinagem. A Figura 2.9 apresenta as duas ferramentas utilizadas nessa etapa: o *FlatCAM*, empregado na preparação das trajetórias a partir dos arquivos Gerber, e o *bCNC*, utilizado para envio e acompanhamento dos comandos da máquina CNC.

Figura 2.9 – Ferramentas utilizadas na preparação da usinagem da PCB: (a) FlatCAM e (b) bCNC.

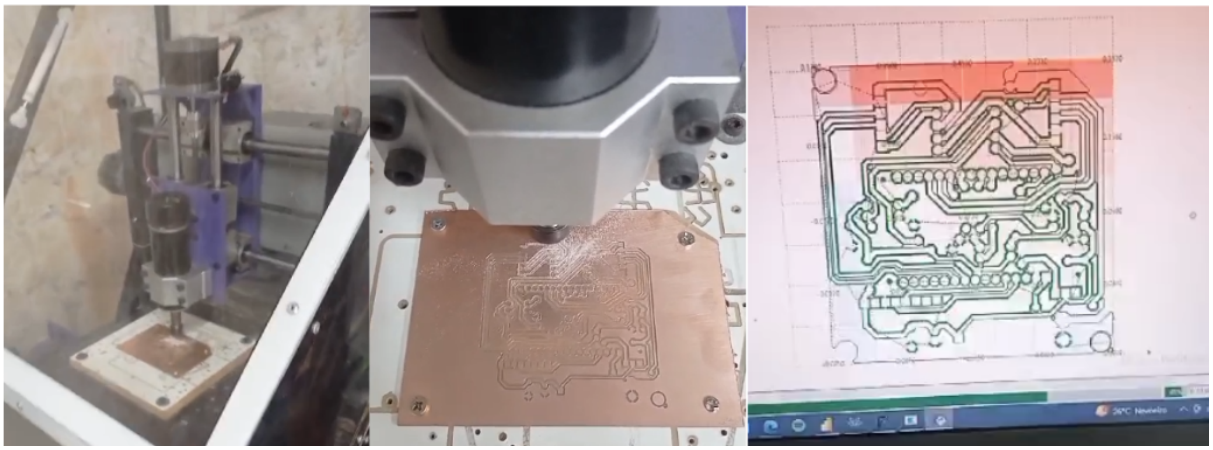


(a) FlatCAM (b) bCNC

Fonte: Autoria própria.

A sequência prática de fabricação é sintetizada na Figura 2.10, que reúne etapas do processo de usinagem e preparação física da placa. Após essa etapa, os componentes eletrônicos foram montados e soldados conforme o projeto.

Figura 2.10 – Etapas do processo de usinagem da PCB.



(a) Preparação da placa. (b) Processo de usinagem. (c) Acompanhamento dos comandos CNC.

Fonte: Autoria própria.

A Tabela 2.1 relaciona os principais componentes empregados na montagem do circuito, suas identificações no esquemático e as respectivas quantidades.

Tabela 2.1 – Lista de componentes utilizados no circuito eletrônico

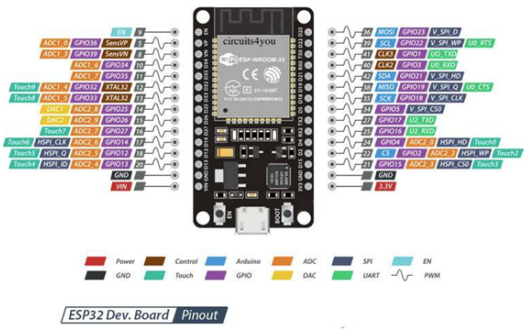
Componente	**Identificação no circuito**	**Qtd.**
ESP32 DevKit	ESP32	1
Driver de motor Driver de motor CC utilizado no robô (L298N) Mini	LM298 MINI	1
Motor CC com encoder	MOTOR A e MOTOR B	2
Módulo inercial GY-87	HW-290/GY-87	1
Regulador de tensão	7805	1
Bateria	BATTERY	1
Diodo retificador	D1 / 1N5400	1
Optoacoplador	U2, U3, U4 e U5 / PC817	4
Resistor 330 Ω	R1_U2 a R1_U5	4
Resistor 10 kΩ	R2_U2 a R2_U5	4
Capacitor 100 pF	C1, C4, C5, C6, C7 e C8	6
Capacitor 10 μF	C2	1
Capacitor 100 nF	C3	1
Conector de 6 pinos	CONN-SIL6	2
Conector de 3 pinos	J1 / CONN-SIL3	1
Conector de 2 pinos	BATTERY	1

Fonte: Autoria própria.

2.2.2.2 ESP32

No projeto, foi utilizado o microcontrolador ESP32 DevKit V1 como unidade principal de processamento embarcado do robô, apresentado na Figura 2.11. A escolha dessa plataforma ocorreu devido à sua capacidade de processamento, conectividade Wi-Fi integrada e compatibilidade com diferentes sensores e dispositivos utilizados no sistema.

Figura 2.11 – Microcontrolador ESP32 DevKit V1 utilizado no robô.



Fonte: Elaborado pelo autor, com aprimoramento visual realizado com auxílio de ferramenta de inteligência artificial.

O ESP32 foi responsável pelo controle dos motores, aquisição de dados dos sensores e comunicação com o sistema principal desenvolvido para o futebol de robôs. Sua arquitetura

permitiu a execução das rotinas de controle em tempo real, contribuindo para o desempenho do robô durante os testes de navegação e movimentação em campo.

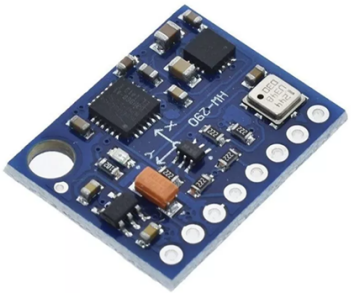
A placa também possibilitou integração com o ROS por meio de comunicação sem fio, permitindo o envio e recebimento de informações entre o robô e o computador responsável pelo processamento da visão computacional e pelas estratégias de jogo.

Além disso, o uso do ESP32 DevKit V1 proporcionou maior flexibilidade no desenvolvimento do projeto, devido à ampla disponibilidade de bibliotecas, suporte da comunidade e facilidade de programação em C/C++ no ambiente Arduino IDE. Dessa forma, a plataforma atendeu de maneira satisfatória às necessidades acadêmicas e experimentais do sistema desenvolvido.

2.2.2.3 Módulo GY-87

O módulo sensor GY-87, apresentado na Figura 2.12, foi utilizado com o objetivo de auxiliar na obtenção de informações relacionadas à orientação e movimentação do robô. O módulo integra diferentes sensores em uma única placa, incluindo acelerômetro, giroscópio, magnetômetro e sensor de pressão atmosférica, possibilitando a coleta de dados importantes para aplicações em robótica móvel.

Figura 2.12 – Módulo inercial GY-87 utilizado no robô.



Fonte: Elaborado pelo autor, com aprimoramento visual realizado com auxílio de ferramenta de inteligência artificial.

Embora o sensor não tenha sido utilizado diretamente na malha de controle do robô durante esta etapa do projeto, sua implementação teve como finalidade a realização de estudos e testes experimentais, visando futuras comparações entre os dados obtidos pelos sensores embarcados e as informações provenientes do sistema de visão computacional.

O módulo foi empregado principalmente na estimativa de orientação do robô em campo, permitindo identificar sua direção de deslocamento e monitorar parâmetros relacionados ao movimento. A utilização do giroscópio e acelerômetro possibilitou a obtenção de dados de

velocidade angular e aceleração, enquanto o magnetômetro forneceu referências de orientação espacial.

A comunicação entre o módulo GY-87 e o microcontrolador foi realizada por meio do protocolo I2C, proporcionando transmissão de dados de forma simples. Os dados coletados foram processados pelo sistema embarcado e utilizados para análise do comportamento do robô durante os testes realizados no campo experimental.

Além disso, o módulo GY-87 apresentou dimensões compatíveis com a plataforma e reuniu, em uma única placa, recursos adequados para experimentos envolvendo navegação, monitoramento de movimento e futuras implementações de fusão sensorial em sistemas de futebol de robôs.

2.2.2.4 Ponte H

A mini-ponte H MX1616, apresentada na Figura 2.13, foi utilizada para o acionamento dos motores responsáveis pela locomoção do robô. O módulo é destinado ao controle de motores de corrente contínua (CC), permitindo acionamento bidirecional e controle de velocidade por meio de sinais Modulação por Largura de Pulso (em inglês, *Pulse Width Modulation*) (PWM).

Figura 2.13 – Mini-ponte H MX1616 utilizada no acionamento dos motores.



Fonte: Elaborado pelo autor, com aprimoramento visual realizado com auxílio de ferramenta de inteligência artificial.

A ponte H MX1616 foi integrada ao sistema embarcado do robô, sendo controlada pelo microcontrolador responsável pelo processamento dos comandos de movimentação. Sua utilização possibilitou o controle individual dos motores, permitindo executar movimentos como avanço, ré, curvas e rotações no próprio eixo, características essenciais para aplicações na categoria IEEE-VSSS.

O módulo apresenta dimensões compactas, tornando-se adequado para sistemas embarcados com restrições de espaço físico. Sua faixa de operação foi utilizada de forma compatível com os micromotores empregados no robô desenvolvido.

A comunicação entre o microcontrolador e a ponte H foi realizada por meio de sinais digitais e PWM, permitindo ajustar a velocidade e o sentido de rotação no acionamento dos motores. Dessa forma, a MX1616 integrou o sistema de tração do robô e permitiu o acionamento compatível com os comandos gerados pelo controlador.

2.2.2.5 Micromotor com Encoder e Bateria

Foram utilizados micromotores CC com encoder integrado, apresentados juntamente com a bateria na Figura 2.14. Esses motores são responsáveis pela locomoção do robô e pela execução dos movimentos necessários durante as partidas e testes experimentais. Os motores foram selecionados devido às suas dimensões reduzidas, baixo consumo de energia e compatibilidade com a estrutura compacta exigida pela categoria IEEE-VSSS.

Figura 2.14 – Elementos utilizados no robô: (a) micromotor CC com encoder e (b) bateria de alimentação.



(a) Micromotor CC com encoder.

(b) Bateria utilizada na alimentação.

Fonte: Elaborado pelo autor, com aprimoramento visual realizado com auxílio de ferramenta de inteligência artificial.

Os encoders acoplados aos motores possibilitam a obtenção de informações relacionadas à rotação das rodas, permitindo estimar parâmetros como velocidade angular, deslocamento e distância percorrida. Entretanto, nesta etapa do projeto, os sinais dos encoders não foram utilizados diretamente na malha de controle do robô, empregados principalmente para aquisição de dados e estudos experimentais voltados a futuras implementações.

A utilização dos encoders tem como objetivo permitir, em trabalhos futuros, a realização de comparações entre técnicas de odometria e os dados obtidos pelo sistema de visão computacional, contribuindo para análises de posicionamento, estimativa de trajetória e fusão sensorial em aplicações de futebol de robôs.

Para alimentação do sistema, foi utilizada uma bateria recarregável compatível com os requisitos elétricos do robô, fornecendo energia para os motores, sensores e demais componentes

eletrônicos embarcados. A bateria foi escolhida considerando fatores como peso, dimensões compactas e capacidade de alimentação adequada ao tempo de operação desejado.

A integração entre motores, encoders e sistema de alimentação permitiu o funcionamento do robô de forma estável durante os testes realizados, oferecendo uma plataforma adequada para experimentos envolvendo controle, navegação e futuras implementações de odometria em robótica móvel.

2.3 Sistema de Comunicação e Visão Computacional

O sistema de comunicação e visão computacional foi estruturado para realizar a integração entre a percepção do ambiente, o processamento das informações e o controle do robô. A arquitetura proposta utiliza o ROS Subseção 2.3.1 como plataforma de comunicação entre os diferentes módulos do sistema e a biblioteca OpenCV Subseção 2.3.2 para o tratamento das imagens capturadas pela câmera posicionada sobre o campo. Dessa forma, as informações referentes à posição e à orientação do robô, bem como à localização da bola, são processadas e transmitidas aos módulos de controle, que geram os comandos de movimentação enviados ao robô por meio de uma rede Wi-Fi.

2.3.1 ROS

O ROS é um *framework* de código aberto amplamente utilizado em robótica para facilitar o desenvolvimento de sistemas complexos e distribuídos. Ele fornece ferramentas e bibliotecas que permitem a comunicação eficiente entre diferentes módulos de um robô, abstraindo detalhes de hardware e permitindo integração entre sensores, atuadores e algoritmos de controle (QUIGLEY et al., 2009).

Este *framework* foi utilizado como a estrutura principal de comunicação do sistema, por meio da biblioteca `rospy`. O ROS permitiu organizar a arquitetura em nós independentes, cada um responsável por executar tarefas específicas, como aquisição de imagens, detecção da bola, identificação do robô, suavização de posições, controle de movimento, estimativa de velocidades e comunicação com o hardware.

A troca de informações entre os módulos ocorreu por meio de tópicos ROS, possibilitando a publicação e assinatura de dados em tempo real. O sistema compartilha informações sobre velocidades das rodas, erro angular, trajetórias do robô e da bola, pontos de interceptação, forças repulsivas e comandos de velocidade linear e angular.

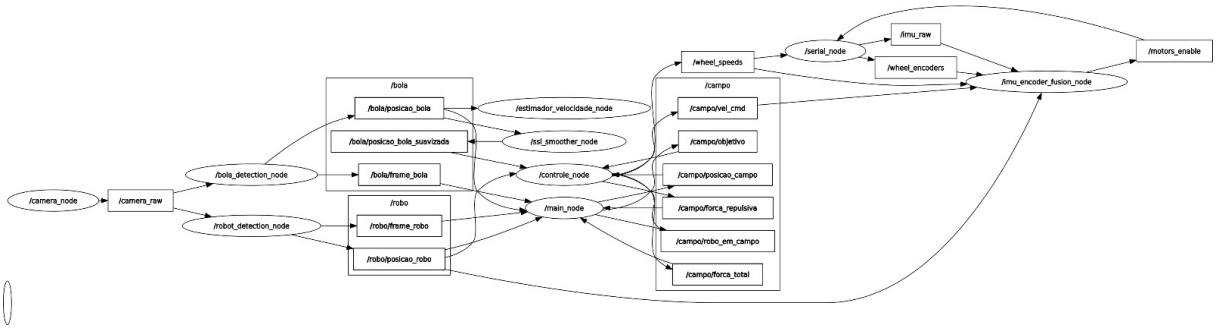
A arquitetura distribuída proporcionou maior modularidade, facilitou a manutenção do sistema e permitiu a integração coordenada entre os módulos de visão computacional, controle, navegação, estimativa de estados e atuação, contribuindo para o desempenho do robô na categoria IEEE VSSS.

A transmissão de dados entre o ROS e o ESP32 DevKit V1 foi realizada utilizando o protocolo TCP, permitindo o envio contínuo de comandos e o monitoramento das informações do robô durante os testes.

A implementação embarcada dessa comunicação, incluindo a conexão Wi-Fi, a recepção das referências de velocidade, a geração dos sinais PWM e a publicação dos dados dos sensores, é detalhada no Apêndice .

A organização dos principais módulos de comunicação é apresentada na Figura 2.15. Nessa arquitetura, o nó de aquisição disponibiliza as imagens da câmera; os nós de visão computacional extraem as posições da bola e do robô; os módulos de navegação e controle processam essas informações e calculam as referências de movimento; por fim, os comandos são transmitidos ao ESP32 para acionamento dos motores. A separação em nós permite que percepção, processamento e atuação troquem informações por tópicos ROS sem que todas as funções sejam executadas em um único processo.

Figura 2.15 – Arquitetura de comunicação entre os principais módulos do sistema ROS.



Fonte: Autoria própria.

Uma síntese dos principais nós, tópicos, parâmetros e ferramentas empregados na arquitetura é reunida no Apêndice . Para o leitor que desejar inspecionar a conectividade efetivamente observada durante a execução, o grafo completo gerado pela ferramenta `rqt_graph`[2].

[2] O `rqt_graph` Figura 2.15 é uma ferramenta gráfica do ROS utilizada para visualizar os nós em execução e as conexões estabelecidas entre eles por meio dos tópicos, é apresentado de forma ampliada no Apêndice F

2.3.2 OpenCV

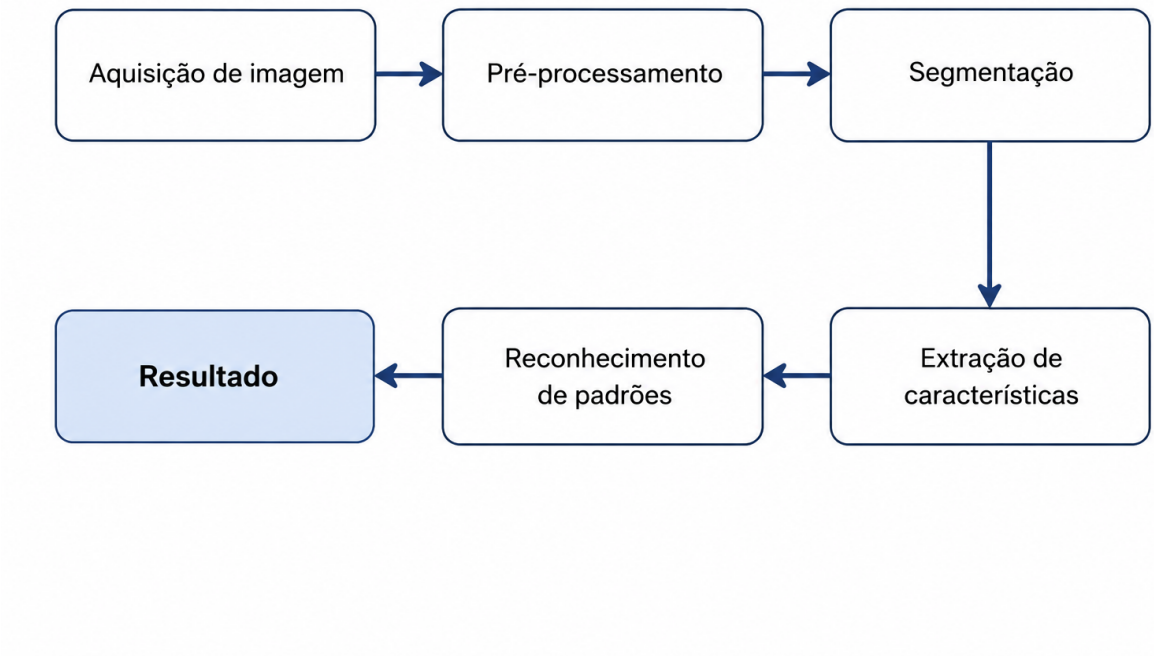
A biblioteca OpenCV é amplamente utilizada para processamento de imagens e visão computacional, oferecendo funções que facilitam a análise, manipulação e visualização de imagens digitais (SIGUT et al., 2020).

O sistema foi desenvolvido utilizando a linguagem Python, uma linguagem de programação de alto nível conhecida por sua simplicidade, legibilidade e ampla gama de bibliotecas científicas e de engenharia. Foram empregados recursos do Python, como NumPy, `math`, `time`, `deepcopy` e `collections.deque`, que auxiliaram nos cálculos numéricos, no tratamento de posições, na estimativa de velocidade, no controle PID e no armazenamento de históricos utilizados pelo sistema.

Dentro desse contexto, a biblioteca OpenCV, por meio do módulo `cv2`, foi aplicada para processar as imagens capturadas pela câmera do sistema de visão computacional do robô. O OpenCV permitiu manipular essas imagens e representar visualmente informações relevantes para a análise do comportamento do robô, incluindo campo, bola, objetivo, orientação, vetores, forças e regiões de interesse, como mostrado na Figura 2.17a. A biblioteca também forneceu recursos para exibir imagens processadas, criar janelas gráficas, permitir interações com o mouse, desenhar elementos geométricos e inserir textos auxiliares na interface.

O processamento das imagens envolveu etapas Figura 2.16, como conversão para o espaço de cores Matiz, Saturação e Valor (em inglês, *Hue, Saturation and Value*) (HSV) Essa representação foi utilizada para facilitar a segmentação dos elementos do campo com base nas cores de identificação, aplicação de filtros, limiarização e detecção de objetos, possibilitando extrair informações essenciais para o sistema de navegação e controle, conforme exemplificado na Figura 2.17b. Essas informações foram posteriormente enviadas aos módulos de controle implementados no ROS, contribuindo para a tomada de decisão e movimentação do robô em campo.

Figura 2.16 – Etapas do processamento de imagens utilizado para aquisição, tratamento e reconhecimento de padrões.

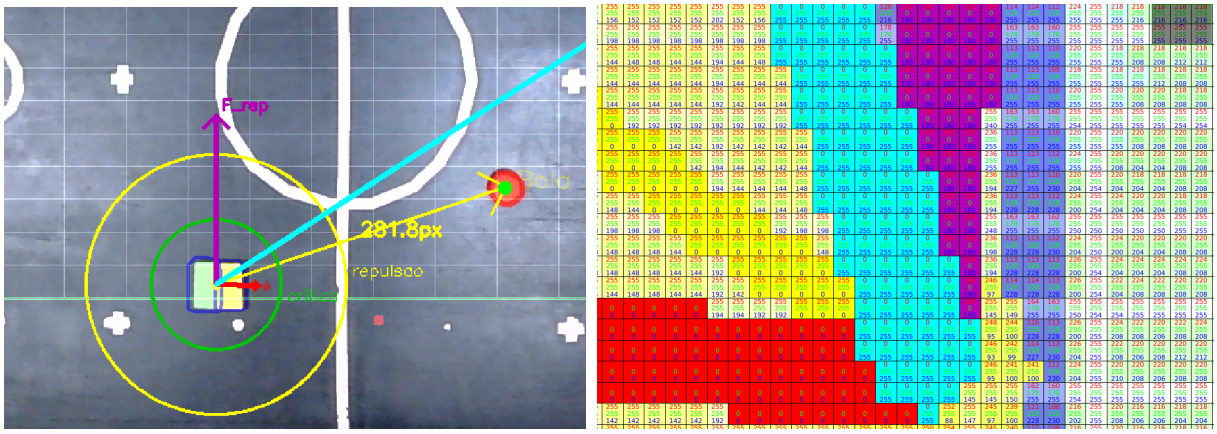


Fonte: Autoria própria.

Dessa forma, a biblioteca OpenCV forneceu as ferramentas necessárias aos experimentos envolvendo rastreamento de objetos, navegação baseada em visão computacional e sistemas de futebol de robôs autônomos.

A Figura 2.17 sintetiza duas etapas do sistema visual. Na Figura 2.17a, observa-se a imagem superior do campo com as marcações inseridas pelo processamento, utilizadas para acompanhar a posição e a orientação dos elementos relevantes. Já a Figura 2.17b mostra a etapa de calibração e segmentação por cores, empregada para separar visualmente as regiões associadas ao robô e à bola antes do cálculo de suas coordenadas.

Figura 2.17 – Etapas do processamento visual: (a) vista superior do campo processada e (b) calibração de cores para identificação dos elementos.



(a) Vista superior do campo com captura e processamento de imagem para controle do robô.

(b) Calibração de cores utilizada para identificar e rastrear os elementos no campo experimental.

Fonte: Autoria própria.

O procedimento completo de detecção da bola, cálculo do centróide, publicação das coordenadas e suavização da trajetória é apresentado no Apêndice[3]. De forma complementar, o cálculo da posição e da orientação do robô a partir das marcações frontal e traseira é detalhado no Apêndice[4]. Esses apêndices permitem aprofundar a implementação da etapa de percepção sem sobrecarregar a descrição conceitual deste capítulo.

Com a etapa de percepção definida, o sistema passa a dispor das coordenadas necessárias para o controle e a navegação. A próxima seção descreve o ambiente físico utilizado para a realização dos experimentos e para o posicionamento da câmera de visão global.

2.4 Campo desenvolvido da Categoria VSSS

2.4.1 Campo

O campo utilizado no desenvolvimento deste projeto foi construído com base nas características da categoria IEEE-VSSS (CBR, 2021), adotando as dimensões principais de 1,50 m por 1,30 m e elementos visuais compatíveis com o ambiente de jogo empregado nos experimentos. O objetivo da construção foi disponibilizar um ambiente experimental controlado para os testes do sistema de visão, navegação e controle.

A estrutura do campo, apresentada na Figura 2.18, possui aproximadamente 1,50 m de comprimento por 1,30 m de largura, confeccionada sobre uma superfície plana de MDF,

[3] A implementação completa do nó de recepção das posições da bola está apresentada no Apêndice A.
[4] A implementação completa do nó de detecção do robô está apresentada no Apêndice B.

garantindo estabilidade mecânica e uniformidade durante a movimentação dos robôs. A pintura superior foi realizada utilizando tinta da cor preta, favorecendo a detecção visual pelos algoritmos de visão computacional.

Figura 2.18 – Estrutura do campo experimental desenvolvido para os testes.



Fonte: Autoria própria.

As marcações do campo foram realizadas com linhas brancas, delimitando regiões como laterais, meio de campo e áreas dos gols. As bordas laterais também podem ser observadas na Figura 2.18 e foram adicionadas para impedir a saída da bola e dos robôs da área útil, auxiliando na continuidade das partidas e experimentos.

Os gols foram construídos proporcionalmente ao tamanho do campo, permitindo simular situações reais de jogo da categoria IEEE-VSSS. Toda a estrutura foi planejada visando compatibilidade com um sistema de visão superior, utilizado para rastreamento da bola e do robô em tempo real.

Dessa forma, conforme ilustrado na Figura 2.18, o ambiente desenvolvido reproduz as características necessárias aos ensaios realizados neste trabalho, permitindo testar o rastreamento visual, as estratégias de navegação e os algoritmos de controle aplicados ao futebol de robôs.

2.4.2 Câmera e suporte de posicionamento

Para a aquisição das imagens do campo foi utilizada uma câmera USB 2.0, com resolução HD 1080p e especificação comercial de 30 Mpx. A câmera foi instalada acima do campo, com orientação aproximadamente perpendicular ao plano de jogo, de modo a fornecer uma visão

global da arena. As imagens capturadas são disponibilizadas ao sistema ROS pelo tópico `/camera_raw` e processadas pelos nós de visão computacional para identificação da bola e do robô.

O suporte utilizado para posicionamento da câmera foi desenvolvido com foco na obtenção de uma visão superior do campo, necessária para o funcionamento do sistema de visão computacional empregado na categoria IEEE-VSSS. Como o projeto possui caráter acadêmico e experimental, foram utilizados recursos alternativos de baixo custo para a montagem da estrutura de suporte.

Para elevar e posicionar a câmera adequadamente, foi utilizada uma estrutura adaptada composta por um banco como base de sustentação e um tripé de microfone para fixação e ajuste de altura, conforme ilustrado na Figura 2.19. Essa solução permitiu obter estabilidade mecânica e flexibilidade no posicionamento da câmera sobre o campo, garantindo o enquadramento adequado da área de jogo durante os testes e experimentos realizados.

A utilização desses recursos alternativos apresentou resultados satisfatórios para a captura das imagens em tempo real, possibilitando a detecção dos robôs e da bola pelo sistema de visão computacional. Além disso, a estrutura desenvolvida demonstrou ser uma solução acessível e funcional para ambientes acadêmicos, reduzindo custos de implementação sem comprometer significativamente o desempenho do sistema de monitoramento visual.

Figura 2.19 – Conjunto de suporte da câmera aérea, composto por base de sustentação, tripé de microfone e câmera posicionada sobre o campo.



Fonte: Autoria própria.

2.5 Considerações parciais

O desenvolvimento deste projeto envolveu a integração entre as áreas de mecânica, eletrônica e programação, possibilitando a construção de uma plataforma funcional aplicada à categoria IEEE-VSSS de futebol de robôs. A etapa mecânica permitiu a confecção da estrutura física do robô e do campo experimental, considerando fatores como dimensões reduzidas, mobilidade e adaptação aos requisitos da categoria.

Na parte eletrônica, foram integrados componentes responsáveis pelo acionamento, alimentação e monitoramento do sistema, incluindo microcontrolador, ponte H, sensores e módulos de comunicação. A utilização de componentes de baixo custo e fácil integração contribuiu para o desenvolvimento de uma plataforma acessível para pesquisas e experimentos em robótica móvel.

Já na área de programação, foram desenvolvidos algoritmos responsáveis pelo processamento das informações obtidas pelo sistema de visão computacional, comunicação entre módulos e controle de movimentação do robô. A utilização do ROS 1, associada à biblioteca OpenCV e à comunicação via Wi-Fi utilizando protocolo TCP, possibilitou a implementação de uma arquitetura modular e distribuída, adequada para aplicações em tempo real.

Além disso, sensores como o GY-87 e os encoders dos motores foram incorporados ao sistema com foco em futuras implementações e comparações experimentais, como técnicas de odometria e fusão sensorial associadas à visão computacional.

Dessa forma, a integração entre mecânica, eletrônica e software permitiu o desenvolvimento de uma base sólida para estudos e aplicações em futebol de robôs, oferecendo uma plataforma capaz de ser expandida futuramente com novos algoritmos de controle, navegação autônoma e inteligência artificial.

3 SISTEMA DE CONTROLE E NAVEGAÇÃO

Este capítulo trata do controle e da navegação do robô, implementados por meio da integração entre a percepção do ambiente, a definição dinâmica do alvo, campos potenciais, o controle PID angular e a conversão cinemática para um robô diferencial. O sistema utiliza informações da postura do robô, da posição e velocidade da bola, dos limites do campo e do objetivo final definido pelo usuário.

3.1 Formalização do Problema de Controle

O problema central deste trabalho consiste em controlar a postura de um robô móvel com cinemática diferencial pertencente à categoria VSSS, de modo que ele seja capaz de navegar de forma autônoma, interceptar a bola no espaço de jogo e conduzi-la eficientemente até o gol adversário (alvo), evitando colisões com obstáculos e paredes.

Para modelar formalmente essa tarefa, define-se um referencial global bidimensional cartesiano fixo no plano do campo, conforme ilustrado na Figura 3.1. A postura do robô em qualquer instante de tempo é representada por:

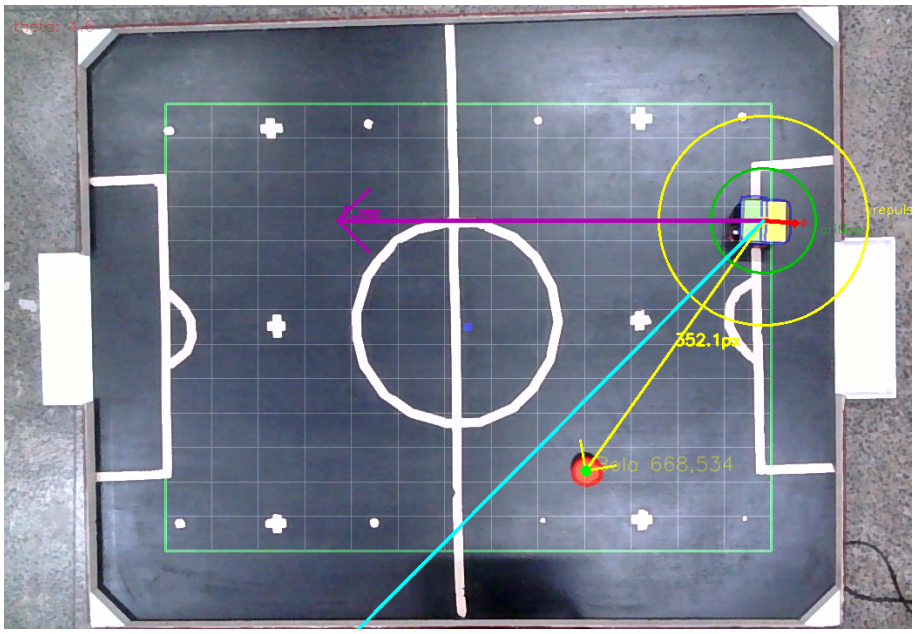
$$q(t) = \begin{bmatrix} x_r(t) & y_r(t) & \theta_r(t) \end{bmatrix}^T, \tag{3.1}$$

em que (x_r, y_r) denota a posição do centro geométrico do robô e θ_r representa sua orientação angular em relação ao eixo horizontal global. Analogamente, as posições da bola e do centro do gol alvo são definidas por:

$$P_b(t) = \begin{bmatrix} x_b(t) & y_b(t) \end{bmatrix}^T, \tag{3.2}$$

$$P_a = \begin{bmatrix} x_a & y_a \end{bmatrix}^T. \tag{3.3}$$

Figura 3.1 – Representação geométrica do problema de controle: a pose do robô é representada pelo centro dos círculos verde e amarelo; as coordenadas da bola são obtidas pelo centro do círculo laranja; o vetor alvo é indicado pela seta amarela; o vetor de orientação da trajetória é representado pela seta azul-clara; ; o vetor de repulsão é representado pela seta roxa e as restrições do campo VSSS são delimitadas pelo retângulo verde.



Fonte: Autoria própria.

O objetivo do controle pode ser dividido em duas fases sequenciais e complementares:

1. **Fase de Interceptação:** minimizar o erro de posição entre o robô e a bola,

$$e_p(t) = \left\| P_b(t) - \begin{bmatrix} x_r(t) \\ y_r(t) \end{bmatrix} \right\| \to 0, \tag{3.4}$$

garantindo que o robô se aproxime por meio de um ponto de condução ideal localizado imediatamente atrás da bola e alinhado ao gol.

2. **Fase de Condução:** minimizar o erro de posição da bola em relação ao alvo,

$$\|P_a - P_b(t)\| \to 0, \tag{3.5}$$

juntamente com o erro de orientação lateral do robô, mantendo o alinhamento necessário para o empurrão ou chute final sem perder o controle da bola.

Matematicamente, o sistema de controle deve computar as ações de velocidade linear $v(t)$ e angular $\omega(t)$ enviadas ao robô para mitigar esses erros, respeitando o modelo cinemático do robô diferencial e as limitações físicas do ambiente competitivo.

A escolha da arquitetura de controle proposta neste trabalho se fundamenta justamente na necessidade de integrar percepção, planejamento e atuação de forma coordenada e em tempo real. Cada módulo da estratégia foi selecionado visando atender aos desafios específicos da categoria VSSS, garantindo precisão na interceptação, segurança na navegação autônoma e manutenção das restrições cinemáticas do robô.

Para lidar com a presença de oponentes e barreiras físicas durante o cumprimento das metas de posicionamento, adota-se uma abordagem de campos potenciais artificiais, onde forças atrativas guiam o robô ao objetivo (P_b ou P_a) e forças repulsivas evitam colisões. Essa modulação de velocidades, combinada a um controle PID angular para correção de desvios e derrapagens, garante que as trajetórias geradas respeitem a capacidade motora real das rodas, resultando em um sistema integrado, fluido e confiável para a competição.

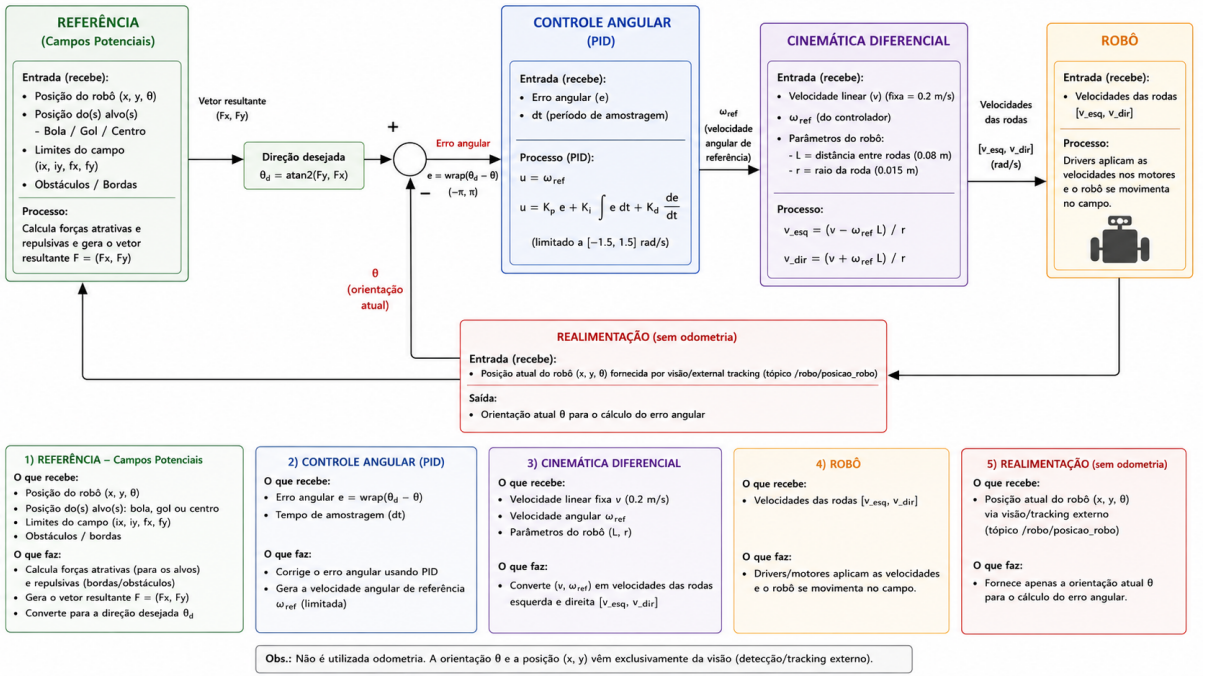
3.2 Estratégia Desenvolvida

Para solucionar o problema de navegação e desvio de obstáculos proposto, foi desenvolvida uma estratégia fundamentada na abordagem de Campos Potenciais Artificiais integrada a uma malha de correção do controle angular/linear do robô.

A escolha da presente arquitetura de controle para o robô na categoria VSSS se fundamenta na necessidade de integrar percepção, planejamento e atuação de forma coordenada e em tempo real. Cada módulo foi selecionado visando atender aos desafios específicos da categoria, garantindo precisão na interceptação da bola, segurança na navegação autônoma e manutenção das restrições físicas e mecânicas do robô.

A arquitetura geral do sistema é apresentada na Figura 3.2. A estrutura adota uma topologia em malha fechada, constituída por três módulos principais: (i) Módulo de Cinemática Diferencial; (ii) Módulo de Controle Angular e de Distância; e (iii) Módulo de Geração de Referência. O funcionamento detalhado de cada um desses blocos é descrito nas subseções a seguir.

Figura 3.2 – Diagrama de blocos da estratégia de controle e navegação proposta.



Fonte: Elaborado pelo autor, com aprimoramento gráfico realizado por ferramenta de inteligência artificial.

A Figura 3.2 evidencia o encadeamento entre a geração da referência, o cálculo dos erros de controle e a conversão das velocidades do corpo para as rodas. A partir dessa organização, cada módulo pode ser analisado separadamente sem perder a relação de realimentação existente entre percepção, controle e atuação.

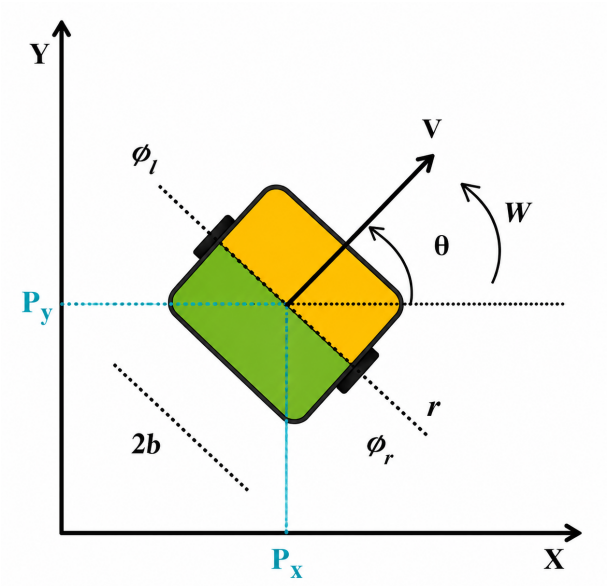
A implementação computacional das etapas de estimativa da velocidade da bola, predição do ponto de interceptação, definição do *lookahead*, cálculo do erro angular, filtragem e correção lateral durante o domínio da bola é apresentada no e detalha no Apêndice[1]. Assim, o presente capítulo mantém o foco na formulação da estratégia, enquanto o apêndice reúne os detalhes necessários para reproduzir sua implementação.

3.2.1 Módulo de Cinemática Diferencial

Para fins de análise e controle do veículo na categoria VSSS, a modelagem geométrica e a conversão de velocidades são tratadas de forma analítica neste módulo. Conforme ilustrado na Figura 3.3, o estado de um robô móvel diferencial no plano cartesiano bidimensional exige a determinação concomitante de sua posição e de sua orientação em relação a um referencial global fixo no campo.

[1] A implementação completa da estratégia e filtro do erro angular está apresentada no Apêndice C.

Figura 3.3 – Representação espacial e postura do robô diferencial no plano.



Fonte: Elaborado pelo autor, com aprimoramento gráfico realizado por ferramenta de inteligência artificial.

A postura (ou vetor de estado) do veículo no referencial global é formalizada por (CORKE, 2023):

$$q = \begin{bmatrix} x_r & y_r & \theta_r \end{bmatrix}^T . \tag{3.6}$$

As coordenadas cartesianas x_r e y_r determinam a posição do centro geométrico do robô em relação aos eixos fixos X e Y do campo. A orientação θ_r é medida, no sentido anti-horário, a partir do semieixo positivo de X até o eixo longitudinal do robô, sendo normalizada no intervalo $[-\pi, \pi]$ radianos. Na Figura 3.3, as indicações P_x, P_y e θ correspondem, respectivamente, a x_r, y_r e θ_r. O referencial global permanece fixo ao campo, enquanto o eixo longitudinal local do robô acompanha sua orientação instantânea.

Devido ao alinhamento fixo das rodas e à impossibilidade de esterçamento mecânico, o veículo está sujeito a uma restrição não holonômica: ele não pode se mover lateralmente de forma instantânea. O movimento de translação ocorre exclusivamente ao longo do eixo longitudinal local, enquanto o movimento de rotação se dá em torno do eixo vertical Z do chassi, acionado pela diferença de velocidades entre os atuadores (CORKE, 2023).

As velocidades lineares desenvolvidas pelas rodas esquerda (v_l) e direita (v_r) dependem diretamente do raio nominal da roda (r) e das respectivas velocidades angulares de rotação dos motores, denotadas por ω_l e ω_r (CORKE, 2023):

$$v_l = r \cdot \omega_l \tag{3.7}$$

$$v_r = r \cdot \omega_r \tag{3.8}$$

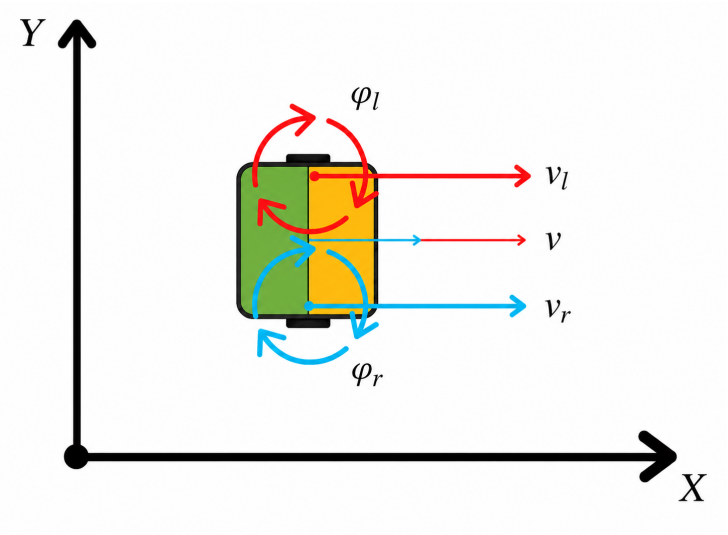
A partir dessas velocidades independentes, e considerando a simetria geométrica em que as rodas encontram-se dispostas de forma equidistante a uma distância b (comprimento do eixo entre as rodas), estabelece-se a relação com as velocidades linear resultante (v) e angular (ω) do corpo do robô, expressas por (CORKE, 2023):

$$v = \frac{v_r + v_l}{2} = \frac{r}{2}(\omega_r + \omega_l) \tag{3.9}$$

$$\omega = \frac{v_r - v_l}{b} = \frac{r}{b}(\omega_r - \omega_l) \tag{3.10}$$

A Figura 3.4 sintetiza a geometria do robô diferencial, indicando as velocidades das rodas e a restrição de movimento lateral associada à configuração não holonômica.

Figura 3.4 – Restrições de movimento e velocidades nas rodas do robô diferencial.



Fonte: Elaborado pelo autor, com aprimoramento gráfico realizado por ferramenta de inteligência artificial.

Para mapear o deslocamento do veículo em relação ao referencial inercial do campo, realiza-se a decomposição trigonométrica com base na orientação atual θ_r. Esse mapeamento constitui o modelo cinemático direto, formalizado matricialmente (CORKE, 2023):

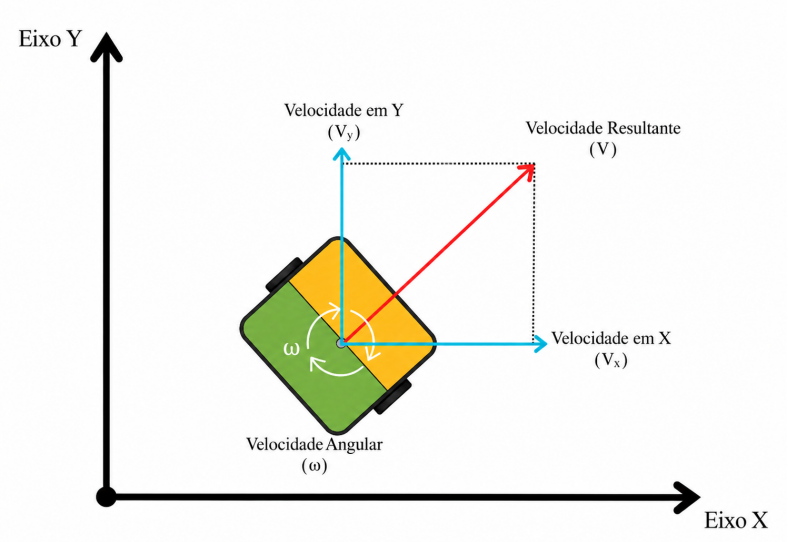
$$\dot{q} = \begin{bmatrix} \dot{x}_r \\ \dot{y}_r \\ \dot{\theta}_r \end{bmatrix} = \begin{bmatrix} \cos(\theta_r) & 0 \\ \sin(\theta_r) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} \tag{3.11}$$

A estrutura dessa matriz evidencia o **desacoplamento cinemático** do sistema: a velocidade linear v atua nas taxas de variação posicional $(\dot{x}_r, \dot{y}_r)$ dependendo da orientação instantânea,

enquanto a velocidade angular ω afeta de forma isolada a taxa de orientação ($\dot{\theta}_r = \omega$). Esse fator justifica a arquitetura adotada, permitindo que a malha de controle PID angular calcule e atue diretamente sobre a variável ω.

A relação entre a velocidade linear v, a velocidade angular ω e o referencial local do robô é ilustrada na Figura 3.5.

Figura 3.5 – Representação das velocidades linear e angular no corpo do robô.



Fonte: Elaborado pelo autor, com aprimoramento gráfico realizado por ferramenta de inteligência artificial.

Por fim, para que os comandos gerados pelos blocos anteriores (Campos Potenciais e Controle PID) se transformem em ações físicas, este módulo executa a cinemática inversa. Isolando-se as velocidades dos motores a partir das equações cinemáticas do corpo, obtêm-se as referências de velocidade angular necessárias para cada roda (CORKE, 2023):

$$\omega_r = \frac{2v + \omega \cdot b}{2r} \tag{3.12}$$

$$\omega_l = \frac{2v - \omega \cdot b}{2r} \tag{3.13}$$

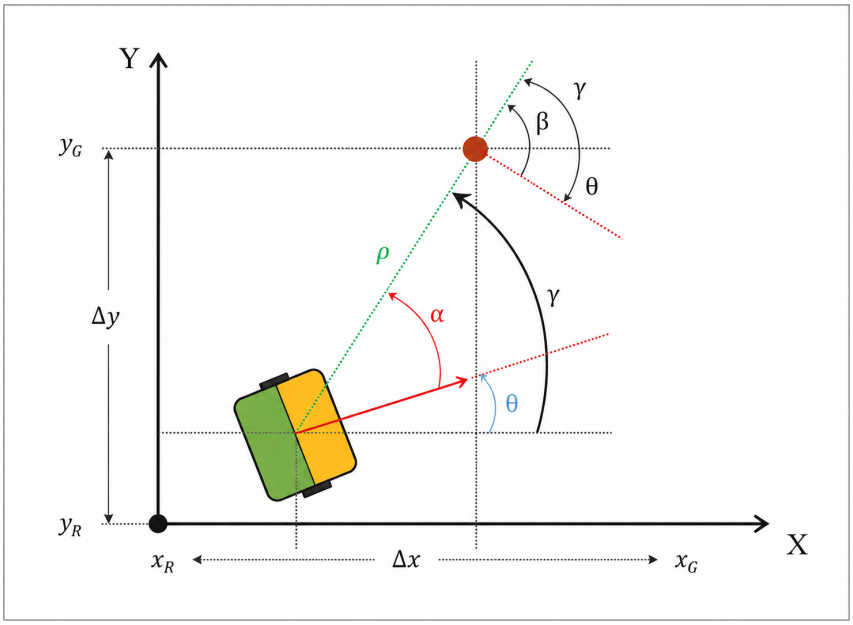
Essas velocidades angulares de referência são subsequentemente convertidas em sinais de comando PWM, fechando o ciclo de atuação e garantindo que o robô execute com fidelidade física a trajetória planejada.

O procedimento implementado no ESP32 para receber essas referências, limitar os comandos e convertê-los em sinais PWM aplicados aos motores é descrito no Apêndice .

3.2.2 Módulo de Controle Angular e de distância

Para o desenvolvimento da malha de controle do robô diferencial, optou-se pela transformação do modelo cinemático cartesiano para o sistema de coordenadas polares. Essa abordagem é amplamente consolidada no controle de robôs diferenciais, pois converte o problema de acoplamento não linear original em um sistema linearizado em relação ao erro de distância e de orientação direcionada ao ponto objetivo, conforme ilustrado na Figura 3.6.

Figura 3.6 – Representação geométrica dos erros de posição e orientação do robô móvel em coordenadas polares.



Fonte: Elaborado pelo autor, com aprimoramento gráfico realizado por ferramenta de inteligência artificial.

A transição do espaço cartesiano global para o espaço polar é fundamentada pelas variáveis ρ_t (distância euclidiana ao alvo), α_t (erro de direcionamento angular) e β_t (erro em relação à orientação desejada no objetivo). Inicialmente, define-se o ângulo do vetor posição entre o robô e o alvo em relação ao eixo horizontal global, denotado por γ_t (SIEGWART; NOURBAKHSH; SCARAMUZZA, 2011):

$$\gamma_t = \text{atan2}(\Delta y_t, \Delta x_t). \tag{3.14}$$

A utilização do operador $\text{atan2}(\cdot)$ permite determinar corretamente o quadrante do vetor alvo, mapeando o ângulo no intervalo $[-\pi, \pi]$ radianos (SIEGWART; NOURBAKHSH; SCARAMUZZA, 2011).

A partir de γ_t, a distância euclidiana e os erros angulares associados à postura do robô

são formalizados por:

$$\rho_t = \sqrt{(\Delta x_t)^2 + (\Delta y_t)^2}, \tag{3.15}$$

$$\alpha_t = \gamma_t - \theta_{R,t}, \tag{3.16}$$

$$\beta_t = \theta_{G,t} - \gamma_t. \tag{3.17}$$

Nessas expressões, $\Delta x_t = x_G - x_R$ e $\Delta y_t = y_G - y_R$ representam as diferenças de posição entre o robô e a referência corrente; $\theta_{R,t}$ é a orientação instantânea do robô. A orientação desejada $\theta_{G,t}$ não é fixa: durante a condução da bola ela é atualizada a partir da direção bola–gol,

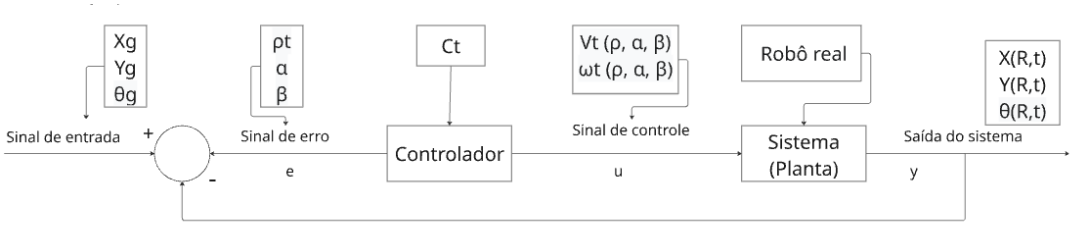
$$\theta_{G,t} = \operatorname{atan2}(y_a - y_b(t),\, x_a - x_b(t))\,, \tag{3.18}$$

de modo que a referência angular acompanha a geometria instantânea entre a bola e o gol. Na implementação do controlador angular, o erro utilizado diretamente para gerar ω é α_t; β_t descreve a orientação final desejada e auxilia na interpretação geométrica da estratégia.

3.2.2.1 Arquitetura da Malha de Controle

Como o objetivo estratégico do sistema baseia-se no seguimento de caminhos reativos em um ambiente altamente dinâmico, a malha fechada opera focada na mitigação contínua e simultânea dos erros polares acoplados. A arquitetura estrutural que rege esse comportamento é detalhada no diagrama de blocos da Figura 3.7.

Figura 3.7 – Diagrama de blocos do sistema de controle em malha fechada baseado em coordenadas polares.



Fonte: Autoria própria.

O fluxo de sinais estruturado na malha fechada explicita a integração entre os módulos de planejamento e atuação. Inicialmente, o sistema de visão computacional (locado acima do campo) captura as posições globais do robô, da bola e dos obstáculos em tempo real. Esses dados são processados pelo Módulo de Geração de Referência que, através dos campos potenciais, calcula as coordenadas espaciais corrigidas $(x_{\mathrm{g}}, y_{\mathrm{g}}, \theta_{\mathrm{g}})$.

Na sequência, a postura atual do robô $q_t = [x_r, y_r, \theta_r]^T$ é comparada a essa referência para realizar a transformação de coordenadas cartesianas para polares, gerando instantaneamente os erros de distância (ρ_t) e direcionamento (α_t). O erro angular alimenta diretamente o bloco do controlador, que computa o esforço necessário para o alinhamento e posicionamento via velocidade angular e linear (v_t, ω_t).

Por fim, esse par de velocidades desejadas (v_t, ω_t) é encaminhado ao bloco de cinemática inversa, que traduz os comandos de corpo em velocidades individuais para as rodas esquerda e direita, fechando o ciclo de controle de forma contínua a cada iteração do consultar no Apêndice.[2]

3.2.2.2 Implementação do Controlador Angular PID

Para o controle da orientação do robô em direção aos alvos móveis ou estáticos do campo, implementou-se um controlador PID. A função principal desta malha é computar a velocidade angular de referência ω a partir do erro de direcionamento instantâneo α, mitigando desvios provocados por perturbações ou deslizamentos.

A lei de controle clássica do PID discreto aplicada ao sistema combina as ações das parcelas proporcional (P), integral (I) e derivativa (D), sendo formalizada por (ÅSTRÖM; HÄGGLUND, 1995):

$$\omega(k) = k_p\,\alpha(k) + k_i\,I_\alpha(k) + k_d\,\dot{\alpha}(k) \tag{3.19}$$

onde k_p, k_i e k_d representam os ganhos sintonizados do controlador. Uma síntese dos parâmetros de controle e de seus significados é apresentada na Tabela E.3, reunida com as demais tabelas de consulta no Apêndice[3].

Cada componente atua sobre uma perspectiva temporal do erro angular (ÅSTRÖM; HÄGGLUND, 1995):

- **Ação Proporcional ($P = k_p\,\alpha$):** Determina a intensidade da correção imediata baseada na magnitude do erro atual $\alpha(k)$;
- **Ação Derivativa ($D = k_d\,\dot{\alpha}$):** Avalia a taxa de variação do erro através da aproximação discreta $\dot{\alpha} = \frac{\alpha(k)-\alpha(k-1)}{\Delta t}$, atuando como um amortecedor para suavizar os movimentos e reduzir oscilações abruptas;

[2] A implementação dos testes de simulação do robô diferencial mo Matlab está apresentada no Apêndice G.

[3] A implementação completa das tabelas reúnem variáveis, parâmetros, tópicos e recursos utilizados na implementação. Apêndice E.

- **Ação Integral ($I = k_i\, I_\alpha$):** Acumula o erro ao longo do tempo por meio da soma discreta $I_\alpha(k) = I_\alpha(k-1) + \alpha(k)\Delta t$, com o objetivo de eliminar o erro de regime permanente (residual).

Em sistemas robóticos sujeitos a saturações físicas nos atuadores, erros grandes ou persistentes podem fazer com que o termo acumulador da integral cresça excessivamente. Esse fenômeno, denominado *windup*, provoca elevados sobressinais (*overshoot*), instabilidade e atrasos severos na resposta de alinhamento do veículo. Para mitigar esse efeito, incorporou-se uma estratégia de *anti-windup* por saturação fixa, limitando a contribuição acumulada do integrador ao intervalo:

$$-0{,}4 \leq I_\alpha \leq 0{,}4. \tag{3.20}$$

Os limites $\pm 0{,}4$ foram adotados na etapa de sintonia experimental do controlador, em conjunto com o ajuste dos ganhos k_p, k_i e k_d. O critério utilizado foi impedir que a parcela integral continuasse crescendo durante erros angulares prolongados, sem eliminar sua contribuição para a correção do erro residual. Os valores utilizados nos ensaios são apresentados posteriormente na Tabela 4.1.

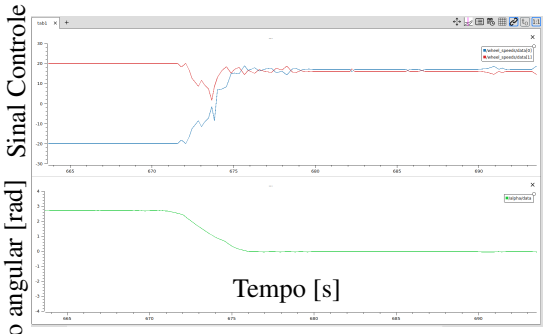
O algoritmo do PID opera sob um tempo de amostragem de $\Delta t = 0{,}1$ s, correspondente a uma atualização de 10 Hz da malha de controle. Essa frequência se refere especificamente ao cálculo do controlador e não deve ser confundida com a taxa de aquisição de imagens da câmera ou com as taxas de publicação dos demais tópicos ROS, que são parâmetros independentes do sistema. Nos ensaios, a atualização de 10 Hz foi mantida constante para permitir comparação entre as respostas do controlador.

As Figuras 3.8, 3.9 e 3.10 ilustram qualitativamente a relação entre o erro angular e a ação corretiva gerada pelo controlador. Na Figura 3.8, a inversão do sentido de rotação modifica o sinal do comando angular, permitindo ao robô corrigir a orientação tanto para a direita quanto para a esquerda antes do deslocamento retilíneo.

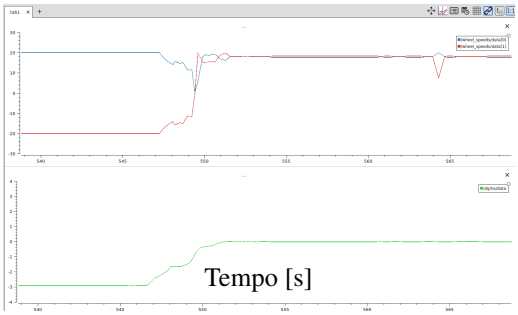
A Figura 3.9 apresenta duas respostas do controlador para variações do erro angular. O comportamento observado evidencia que a ação de controle acompanha a mudança de sinal e de magnitude do erro, reduzindo a correção à medida que o alinhamento é recuperado.

Por fim, a Figura 3.10 mostra situações em que o erro angular sofre mudanças mais acentuadas. Nesses casos, a parcela derivativa contribui para reagir à variação do erro, enquanto a limitação da integral reduz o risco de acúmulo excessivo da ação de controle.

Figura 3.8 – Resposta do controle angular para rotações em sentidos opostos.



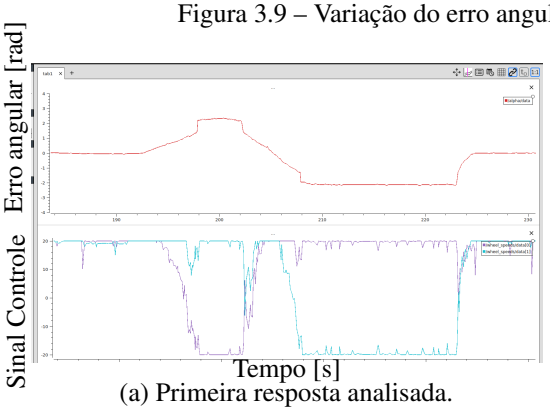
(a) Resposta durante giro à direita.



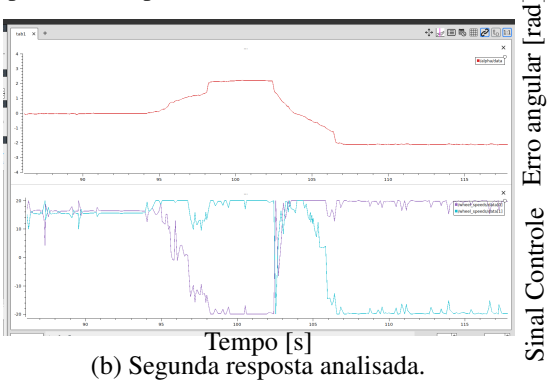
(b) Resposta durante giro à esquerda.

Fonte: Autoria própria.

Figura 3.9 – Variação do erro angular e correspondente resposta do controlador.



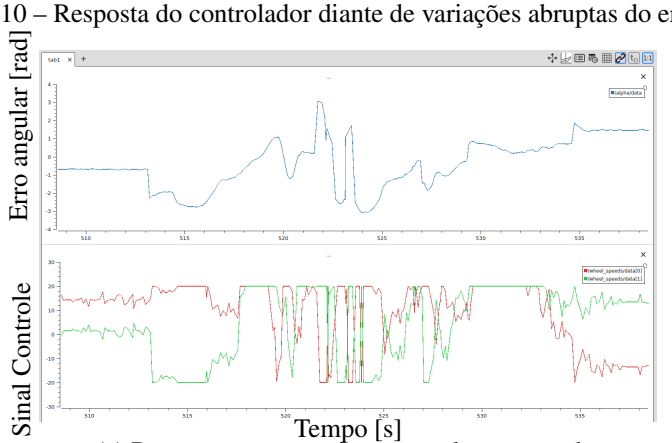
(a) Primeira resposta analisada.



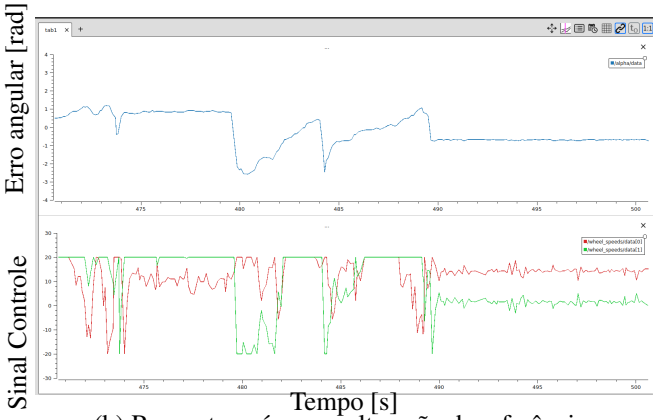
(b) Segunda resposta analisada.

Fonte: Autoria própria.

Figura 3.10 – Resposta do controlador diante de variações abruptas do erro angular.



(a) Resposta a uma variação angular acentuada.



(b) Resposta após nova alteração da referência.

Os ensaios qualitativos anteriores mostram a ação da malha angular diante de mudanças no erro de orientação. A partir desse comportamento, a estratégia também precisa regular a velocidade de avanço para evitar deslocamentos elevados quando o robô estiver desalinhado ou próximo do objetivo.

3.2.2.3 Controlador da Velocidade Linear

A estratégia para a determinação da velocidade linear v baseia-se na distância euclidiana até o objetivo e no desalinhamento angular do veículo. O propósito dessa abordagem é fazer com que o robô se desloque em velocidade máxima em trechos retilíneos e distantes, reduzindo progressivamente a translação à medida que se aproxima do alvo ou quando necessita realizar manobras de reorientação acentuadas (SIEGWART; NOURBAKHSH; SCARAMUZZA, 2011).

Para mitigar o risco de comandos de velocidade excessivos que possam saturar os atuadores ou causar perda de aderência, a distância atual ρ entre o robô e o alvo é primeiramente submetida a uma operação de normalização com saturação fixa, expressa por:

$$\rho_n = \min\left(\frac{\rho}{\rho_{\max}}, 1\right) \tag{3.21}$$

onde $\rho_{\max} = 1500$ representa a distância limite de normalização (em milímetros ou pixels, a depender do referencial do sistema de visão) e $\rho_n \in [0, 1]$ denota a distância adimensional normalizada.

A partir desse fator de atenuação, adotou-se neste trabalho uma lei não linear para a velocidade linear, construída a partir da combinação entre a distância normalizada e o desalinhamento angular:

$$v = v_{\max}\, \rho_n^{0{,}6} \cos(\alpha). \tag{3.22}$$

Essa expressão constitui uma escolha de projeto da estratégia implementada, e não a transcrição direta de uma lei apresentada na referência anterior. O termo $\cos(\alpha)$ reduz o avanço quando o robô está desalinhado, enquanto o expoente $0{,}6$ modifica o perfil de aproximação em função da distância. O parâmetro $v_{\max}$ representa a velocidade máxima de translação permitida ao veículo.

A dinâmica desse perfil de velocidade é regida por dois fatores matemáticos complementares:

- **Atenuação por Desalinhamento ($\cos(\alpha)$):** Atua como um modulador direcional. Quando o robô está perfeitamente alinhado ao alvo ($\alpha = 0$), o termo cossenoidal é unitário, permitindo aceleração total; caso o veículo esteja desalinhado (próximo de $\pm\pi/2$), a

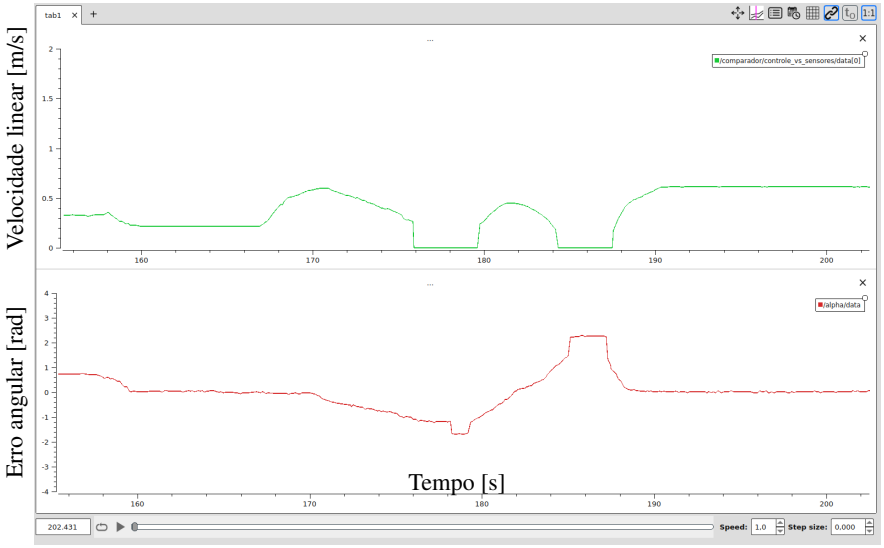
velocidade linear é severamente reduzida, priorizando a correção da orientação pelo controle angular antes do avanço transacional. Estes efeitos são ilustrados na Figura 3.11a;

- **Suavização por Perfil Exponencial ($\rho_n^{0,6}$):** O expoente fracionário de $0,6$ altera o perfil de desaceleração linear. Ele permite que o robô mantenha velocidades mais elevadas durante a maior parte da aproximação, decaindo de forma acentuada e suave apenas nas proximidades imediatas do ponto de parada, otimizando o tempo de resposta sem causar impactos mecânicos bruscos. Estes efeitos são ilustrados na Figura 3.11b

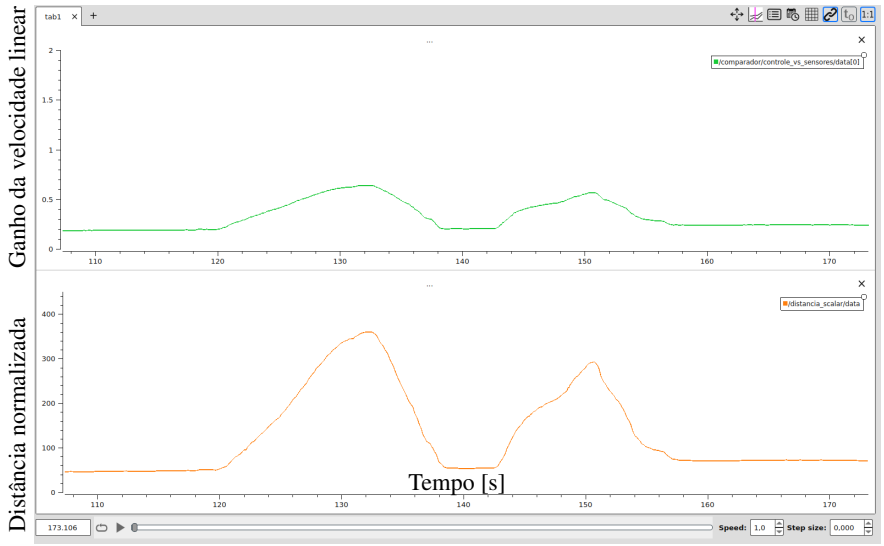
Essa formulação garante um comportamento dinâmico integrado entre as malhas de translação e rotação, promovendo trajetórias fluidas e seguras de acordo com os requisitos de desempenho exigidos na categoria VSSS.

A Figura 3.11 reúne os dois efeitos utilizados na lei de velocidade linear: a atenuação associada ao erro angular e a variação do ganho em função da distância normalizada ao alvo.

Figura 3.11 – Comportamento da velocidade linear em função do erro angular e da distância normalizada ao alvo.



(a) Evolução temporal do erro angular e da velocidade linear correspondente.



(b) Evolução temporal da distância normalizada ao alvo e do ganho aplicado à velocidade linear.

Fonte: Autoria própria.

Na Figura 3.11a, observa-se que o termo cossenoidal reduz progressivamente a velocidade linear à medida que o módulo do erro angular aumenta, priorizando a reorientação do robô. Já a Figura 3.11b evidencia o efeito de $\rho_n^{0,6}$: para distâncias maiores, a referência de velocidade permanece próxima de $v_{\max}$; nas proximidades do alvo, a velocidade é reduzida para tornar a aproximação mais gradual.

3.2.3 Módulo de Geração de Referência

O planejamento de trajetórias reativas e o desvio de obstáculos em tempo real são gerenciados por este módulo através da abordagem de Campos Potenciais Artificiais (APF) (PARK; JEON; LEE, 2001; REN; MCISAAC; PATEL, 2006). O princípio fundamental dessa técnica consiste em preencher o ambiente competitivo por um campo de forças virtuais, onde o robô é guiado pela superposição de forças atrativas, que o puxam em direção ao objetivo (como a bola ou o gol), e forças repulsivas, que o afastam de colisões contra as paredes do campo ou robôs adversários.

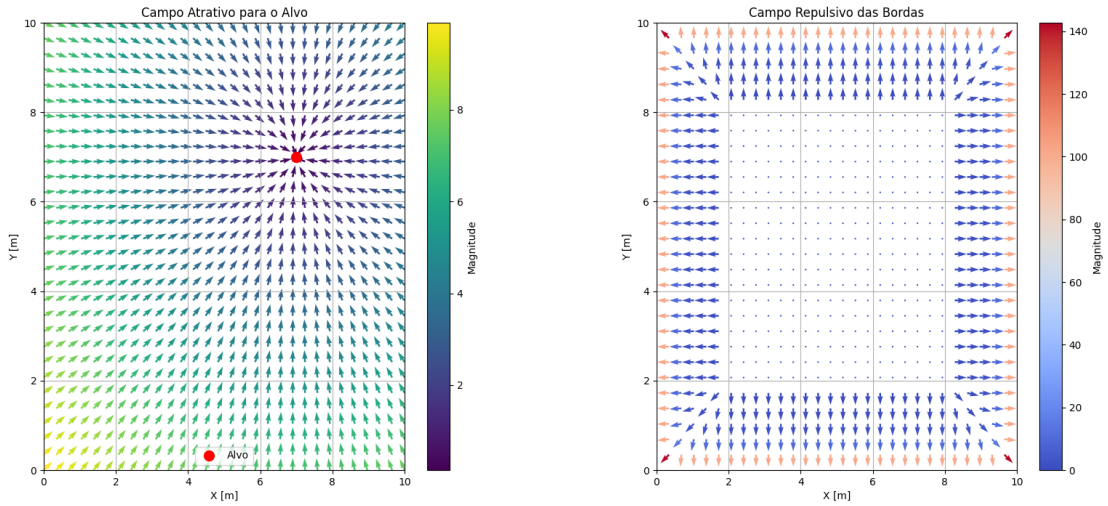
A força atrativa é definida a partir da diferença entre a posição atual do robô e a referência corrente. Desse modo, suas componentes cartesianas, $F_{x,\text{att}}$ e $F_{y,\text{att}}$, são formalizadas por:

$$F_{x,\text{att}} = x_{\text{alvo}} - x_r, \tag{3.23}$$

$$F_{y,\text{att}} = y_{\text{alvo}} - y_r. \tag{3.24}$$

O ponto $(x_{\text{alvo}}, y_{\text{alvo}})$ varia de acordo com a etapa da estratégia. Na fase de interceptação, ele corresponde ao ponto de perseguição calculado a partir da posição prevista da bola; na fase de condução, a referência é reorganizada para manter o robô atrás da bola e orientado na direção do gol adversário. Assim, a atração em direção ao gol ocorre indiretamente pela atualização do ponto de condução, e não pela aplicação permanente de uma força diretamente no centro do gol. A posição instantânea do robô é representada por (x_r, y_r). A Figura 3.12 ilustra as componentes atrativa e repulsiva empregadas pela estratégia.

Figura 3.12 – Representação das componentes atrativa e repulsiva utilizadas nos Campos Potenciais Artificiais.



(a) Representação da força atrativa em direção à referência corrente.

(b) Representação da força repulsiva próxima a uma região de restrição.

Fonte: Elaborado pelo autor, com aprimoramento gráfico realizado por ferramenta de inteligência artificial.

Para representar a contribuição das regiões de restrição, define-se o vetor repulsivo como a soma das contribuições ativas associadas às bordas ou obstáculos considerados pelo algoritmo:

$$\mathbf{F}_{\text{rep}} = \sum_{j \in \mathcal{R}} k_j \, \phi(d_j) \, \hat{\mathbf{n}}_j = \begin{bmatrix} F_{x,\text{rep}} \\ F_{y,\text{rep}} \end{bmatrix}, \tag{3.25}$$

em que d_j é a distância do robô à região de restrição j, $\hat{\mathbf{n}}_j$ é o vetor unitário orientado para longe dessa região, k_j é o ganho de repulsão correspondente e $\phi(d_j)$ representa a ponderação da intensidade em função da distância utilizada pelo algoritmo. Fora do raio de influência, a contribuição repulsiva é anulada. Nos ensaios foram empregados dois níveis de ganho, um para a região de repulsão e outro para a região crítica, cujos valores são apresentados na Tabela 4.1.

Os ganhos de repulsão atuam diretamente como pesos dessa contribuição vetorial: para a mesma distância e direção, um valor maior de k_j aumenta a magnitude da força repulsiva e, consequentemente, aumenta sua influência sobre a direção resultante. Por esse motivo, o ganho utilizado na região crítica é superior ao ganho empregado na região de repulsão.

A força total resulta da soma vetorial das parcelas atrativa e repulsiva:

$$F_{x,\text{total}} = F_{x,\text{att}} + F_{x,\text{rep}}, \tag{3.26}$$

$$F_{y,\text{total}} = F_{y,\text{att}} + F_{y,\text{rep}}. \tag{3.27}$$

Na ausência de regiões de restrição dentro do raio de influência, $\mathbf{F}_{\text{rep}} = \mathbf{0}$ e a direção resultante coincide com a direção atrativa. Quando uma contribuição repulsiva é ativada, a direção de referência é modificada pela soma vetorial.

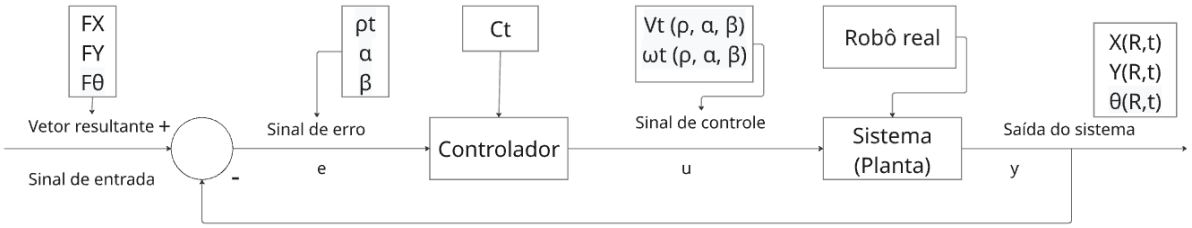
Por fim, o vetor de força total resultante (F_{total}) é mapeado de volta para o espaço de coordenadas do plano de jogo para estabelecer uma meta intermediária de navegação de curto prazo. Essa correção geométrica gera as coordenadas corrigidas do alvo, $(x_{\text{corr}}, y_{\text{corr}})$, definidas por:

$$x_{\text{corr}} = x_r + F_{x,\text{total}} \tag{3.28}$$

$$y_{\text{corr}} = y_r + F_{y,\text{total}} \tag{3.29}$$

Esse novo ponto espacial corrigido Figura 3.13 passa a atuar como a referência de entrada para as malhas subsequentes de controle cinemático. É a partir desse ponto de destino corrigido que a função $\mathrm{atan2}$ extrairá o erro de direcionamento angular (α) exigido pelo controlador PID, permitindo que o robô contorne oponentes de forma fluida e suave. A interação e a resultante final dos campos potenciais atuando em um cenário competitivo real são demonstradas na Figura 3.14a. Onde a interação do robô com a borda inferior gera uma força de repulsão que modifica a trajetória do robô, criando uma força resultante que o afasta da borda, evitando que fique preso e garantindo maior fluidez na locomoção.

Figura 3.13 – Diagrama de blocos do sistema de controle em malha fechada ponto espacial corrigido .



Fonte: Autoria própria.

Já na Figura 3.14, o robô sofre a ação simultânea de forças de repulsão provenientes do canto superior e do lado direito, resultando em um vetor combinado que direciona o robô para longe dessas regiões de restrição, mantendo-o dentro da área de jogo.

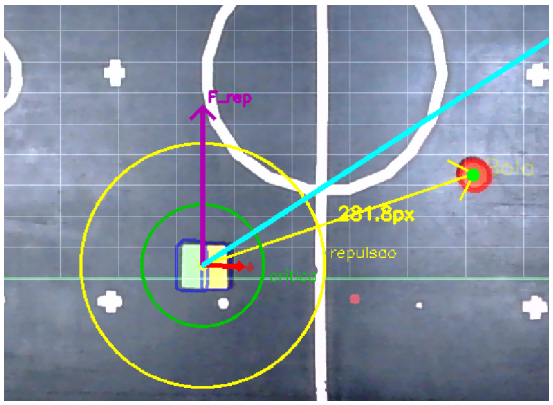
Na Figura 3.15a, observa-se a interação complementar entre as distâncias superiores e inferiores, que, em conjunto com as forças laterais, repelem o robô das bordas. Essa combinação de forças repulsivas Figura 3.15b cria um comportamento de “centro de trajetória”, onde o robô

é constantemente mantido afastado dos limites do campo, conforme evidenciado pelas forças resultantes ilustradas na figura.

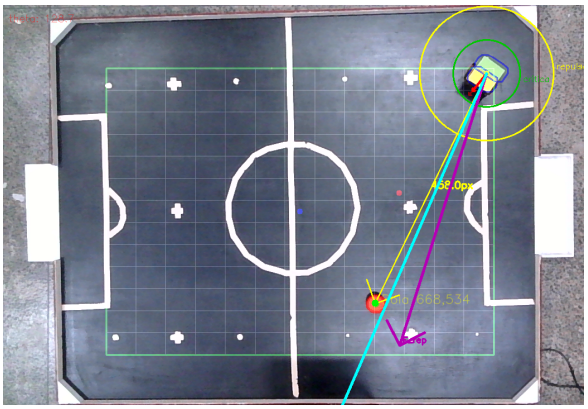
Essa dinâmica evidencia como a aplicação de campos potenciais permite ao robô ajustar continuamente sua trajetória, conciliando o deslocamento em direção ao objetivo com a evitação de obstáculos e bordas do campo, garantindo segurança e eficiência na navegação.

Os valores dos parâmetros físicos, ganhos de controle, limites de filtragem, parâmetros de perseguição, tópicos ROS e recursos utilizados na implementação são consolidados no Apêndice[4], que funciona como material de consulta para as formulações apresentadas neste capítulo.

Figura 3.14 – Interação das forças repulsivas e atrativas do campo potencial sobre o robô.



(a) Força repulsiva causada pela borda inferior do campo.

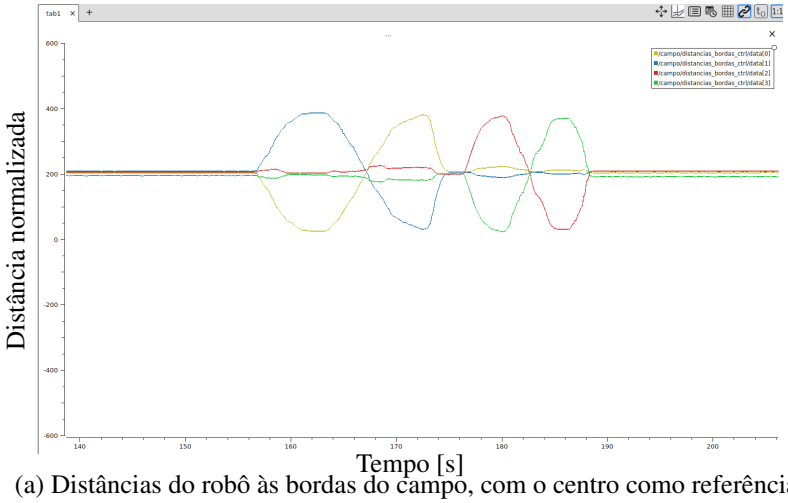


(b) Resultante das forças repulsivas da borda superior e lateral.

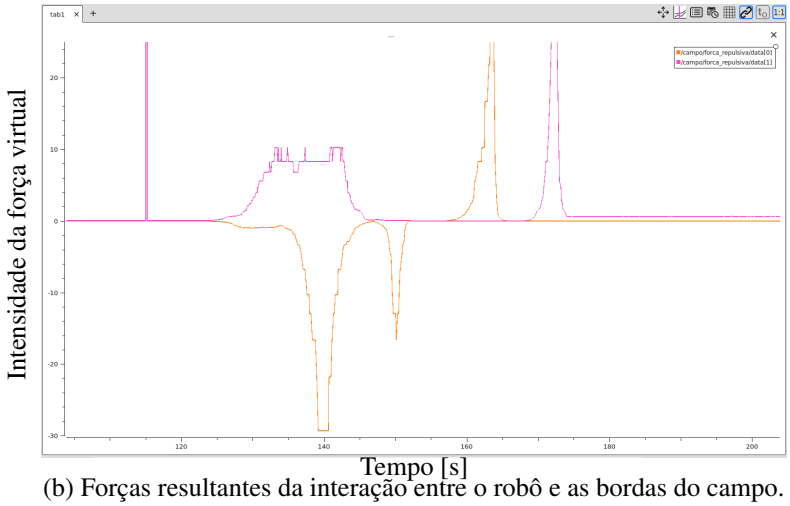
Fonte: Autoria própria.

[4] A implementação completa dos parâmetros utilizados na implementação. Apêndice E.

Figura 3.15 – Diferentes respostas da interação das forças repulsivas e atrativas do campo potencial sobre o robô.



(a) Distâncias do robô às bordas do campo, com o centro como referência.



(b) Forças resultantes da interação entre o robô e as bordas do campo.

Fonte: Autoria própria.

4 RESULTADOS EXPERIMENTAIS

Este capítulo é dedicado à apresentação e análise dos resultados experimentais obtidos para a validação do sistema de controle e navegação desenvolvido. Com o intuito de avaliar o desempenho do robô sob diferentes condições de operação, foram estruturados três experimentos distintos, progressivos em complexidade:

- **Primeiro experimento:** Dedicado à validação do comportamento do sistema de controle em uma condição inicial favorável, considerando o robô já alinhado ao objetivo;
- **Segundo experimento:** Destinado a testar o controle em um cenário dinâmico e de maior complexidade, no qual a bola descreveu um movimento circular, exigindo que o robô rastreasse e perseguisse essa trajetória continuamente;
- **Terceiro experimento:** Projetado para submeter o sistema a uma condição severa de perturbação, posicionando o robô de costas para o objetivo, expondo o algoritmo de controle a altas variações angulares e testando sua capacidade de reorientação rápida.

Antes da exposição dos ensaios, apresenta-se a parametrização adotada para os algoritmos. Nas seções subsequentes, são detalhados a metodologia de cada cenário, os gráficos coletados e a discussão das grandezas utilizadas na avaliação: trajetória do robô e da bola, erro angular e referências de velocidade das rodas.

4.1 Configuração e Sintonia de Parâmetros

Para garantir a repetibilidade dos experimentos e o desempenho dinâmico ideal do robô na categoria VSSS, realizou-se uma etapa prévia de calibração e sintonia das malhas de controle e navegação. Os coeficientes do controlador PID angular foram obtidos experimentalmente através do método de tentativa e erro, buscando um compromisso entre rapidez de resposta no alinhamento e mitigação de oscilações severas (*overshoot*).

Paralelamente, as constantes de ganho dos Campos Potenciais Artificiais (APF) foram ajustadas para ponderar o comportamento reativo do robô. A força de atração foi calibrada para garantir o avanço contínuo em direção ao alvo, enquanto o raio de influência e a magnitude da força repulsiva foram sintonizados para desviar de obstáculos de forma antecipada, sem causar travamentos locais ou desvios exagerados que comprometessem o tempo de jogada.

A Tabela 4.1 consolida os parâmetros numéricos fixados e utilizados de maneira idêntica ao longo dos três experimentos deste capítulo. Além dos ganhos já definidos nas seções anteriores, d_{bola} representa a distância limite para considerar que o robô está em condição de domínio da bola; $d_{\text{atrás}}$ define o afastamento utilizado para construir o ponto de condução atrás da bola; d_{offset} representa o deslocamento lateral aplicado nessa construção; α_{filtro} e α_{deadzone} são, respectivamente, o coeficiente do filtro do erro angular e a zona morta; $\text{lookahead}_{\min}$ e $\text{lookahead}_{\max}$ limitam o horizonte espacial de perseguição; e R_{crit} e R_{rep} delimitam as regiões crítica e de repulsão consideradas pela estratégia de Campos Potenciais.

Tabela 4.1 – Parâmetros de sintonia do sistema de controle e navegação do robô.

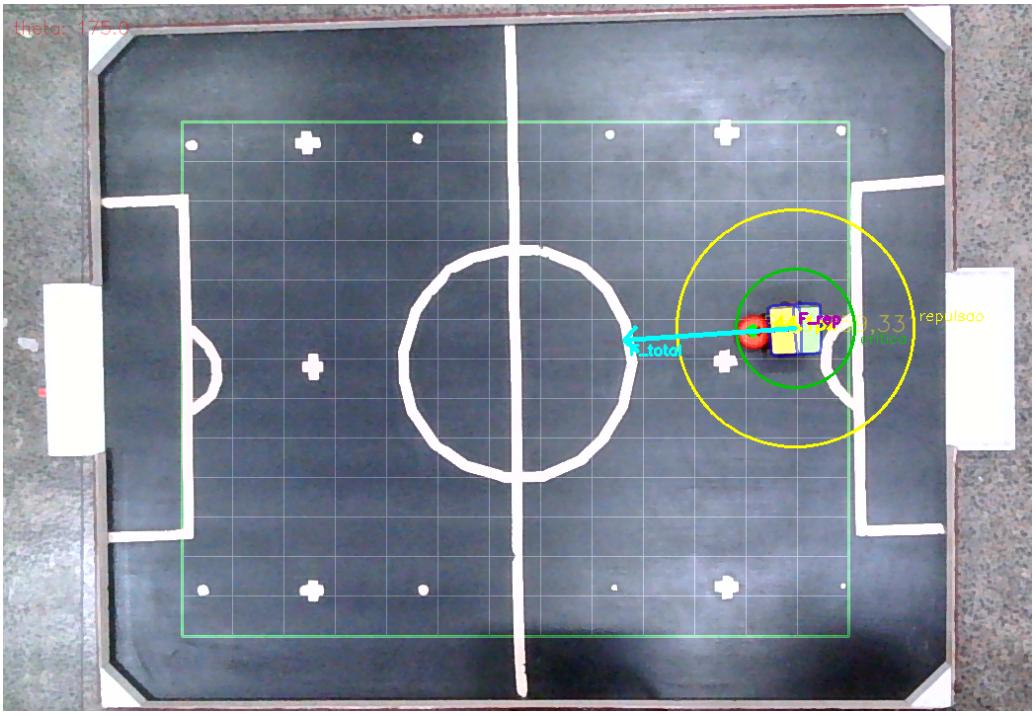
Módulo	**Parâmetro / Ganho**	**Símbolo**	**Valor adotado**
Controle Angular PID	Ganho proporcional	k_p	3,6
	Ganho integral	k_i	0,002
	Ganho derivativo	k_d	2,2
Anti-windup	Limite superior da integral	$I_{\max}$	0,4
	Limite inferior da integral	$I_{\min}$	$-0{,}4$
Velocidade Linear	Velocidade máxima de translação	$v_{\max}$	1,5 m/s
	Distância de saturação linear	$\rho_{\max}$	1500 pixels
Campos Potenciais	Ganho de atração do alvo	ζ	1,0
	Ganho de repulsão	$k_{\text{repulsão}}$	18000,0
	Ganho de repulsão crítica	$k_{\text{repulsão crítica}}$	32000,0
	Raio de influência	d_0	120 pixels
Geometria e Atuação	Raio da roda	r	0,0175 m
	Distância entre rodas	b	0,09 m
	Limite de velocidade angular das rodas	$\omega_{\text{roda,max}}$	20 rad/s
Controle de Bola	Distância para dominar a bola	d_{bola}	42 pixels
	Distância atrás da bola	$d_{\text{atrás}}$	100 pixels
	Offset lateral	d_{offset}	50 pixels
Filtros e Suavização	Fator de filtro do ângulo	α_{filtro}	0,7
	Zona morta do ângulo	α_{deadzone}	0,03 rad
	Lookahead mínimo	$\text{lookahead}_{\min}$	80 pixels
	Lookahead máximo	$\text{lookahead}_{\max}$	250 pixels
Áreas de Repulsão e Crítica	Raio da área crítica	R_{crit}	60 pixels
	Raio da área de repulsão	R_{rep}	120 pixels

Fonte: Autoria própria.

4.2 Experimento 1: Alinhamento Inicial ao Objetivo

O primeiro experimento avalia o comportamento do sistema em uma condição inicial favorável, na qual o robô já se encontra aproximadamente alinhado com a direção de aproximação da bola e do objetivo. A configuração inicial é apresentada na Figura 4.1.

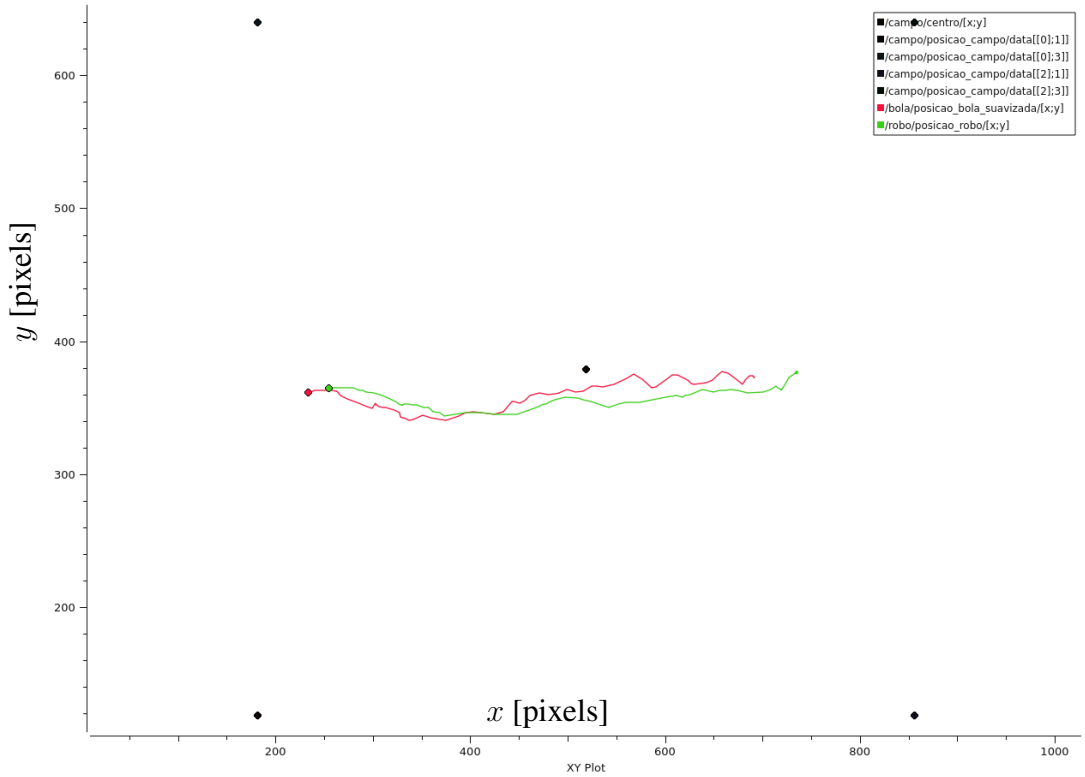
Figura 4.1 – Configuração inicial do robô no primeiro cenário do Experimento 1.



Fonte: Autoria própria.

A trajetória medida é mostrada na Figura 4.2. Nessa representação, a curva vermelha corresponde à posição suavizada da bola e a curva verde corresponde à posição do robô ao longo do ensaio; os marcadores pretos representam pontos de referência utilizados para caracterizar o campo. A proximidade entre as duas curvas durante a condução permite visualizar o acompanhamento da bola pelo robô.

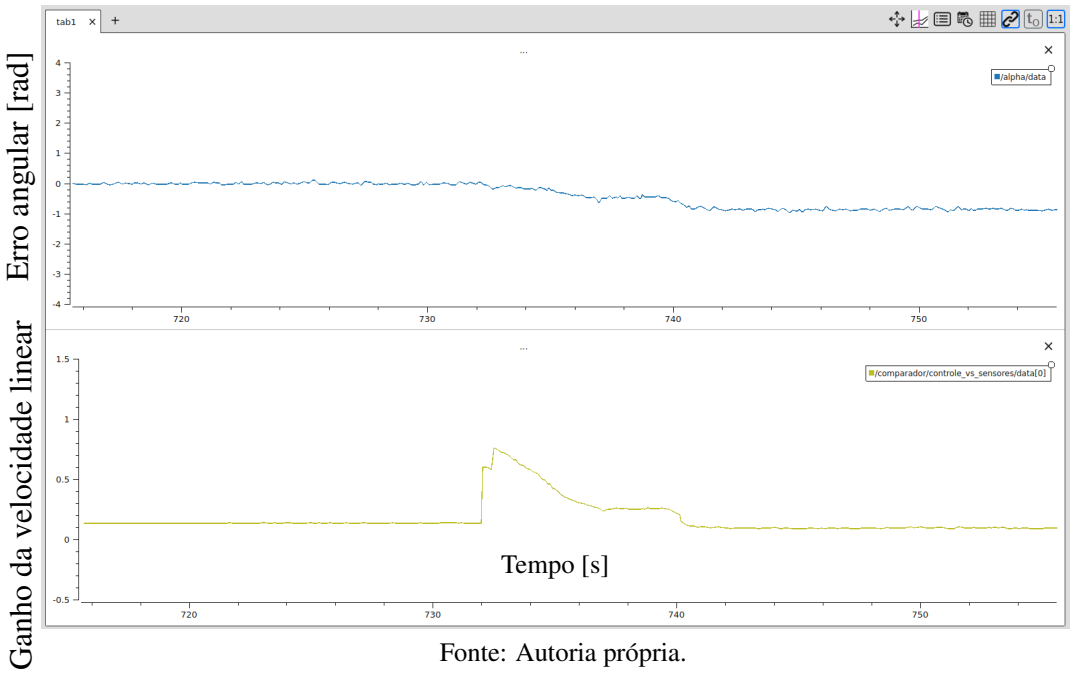
Figura 4.2 – Trajetórias da bola (vermelho) e do robô (verde) no primeiro cenário do Experimento 1.



Fonte: Autoria própria.

A Figura 4.3 reúne sinais utilizados para analisar o comportamento do controlador. O painel superior apresenta o erro angular α publicado pelo sistema. O painel inferior apresenta a primeira componente do tópico de monitoramento `/comparador/controle_vel_sensores`. A separação dos sinais permite observar a evolução do desalinhamento e a resposta do sistema ao longo do ensaio.

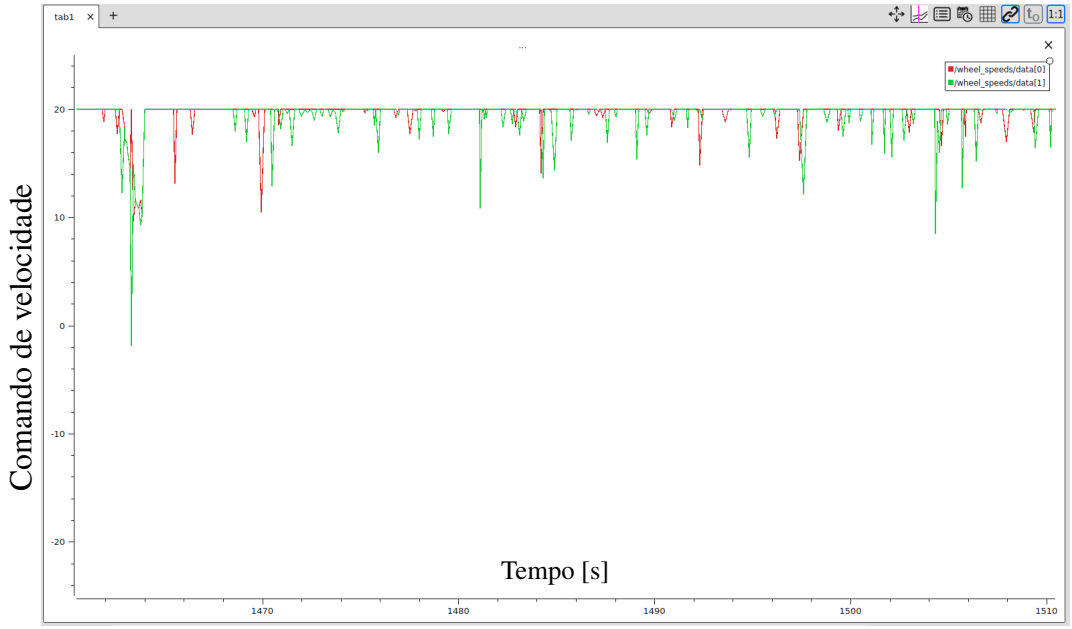
Figura 4.3 – Erro angular α (painel superior) e sinal de monitoramento do controlador (painel inferior) no Experimento 1.



Fonte: Autoria própria.

As referências de velocidade angular enviadas às duas rodas são apresentadas na Figura 4.4. A curva vermelha representa `/wheel_speeds/data[0]` e a curva verde representa `/wheel_speeds/data[1]`. Esses sinais são posteriormente convertidos em PWM pelo ESP32, conforme descrito no Apêndice correspondente.

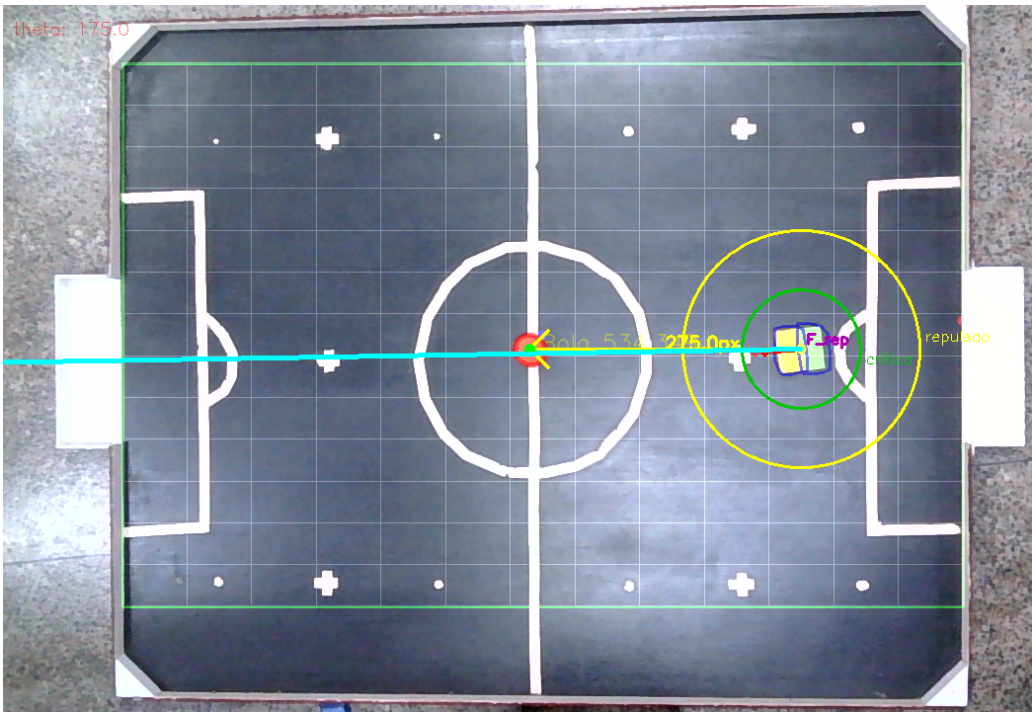
Figura 4.4 – Referências de velocidade angular das rodas: `data[0]` em vermelho e `data[1]` em verde, no Experimento 1.



Fonte: Autoria própria.

Em um segundo cenário do mesmo experimento, o robô também inicia alinhado em direção ao alvo, porém com maior distância em relação à bola. A configuração utilizada é apresentada na Figura 4.5.

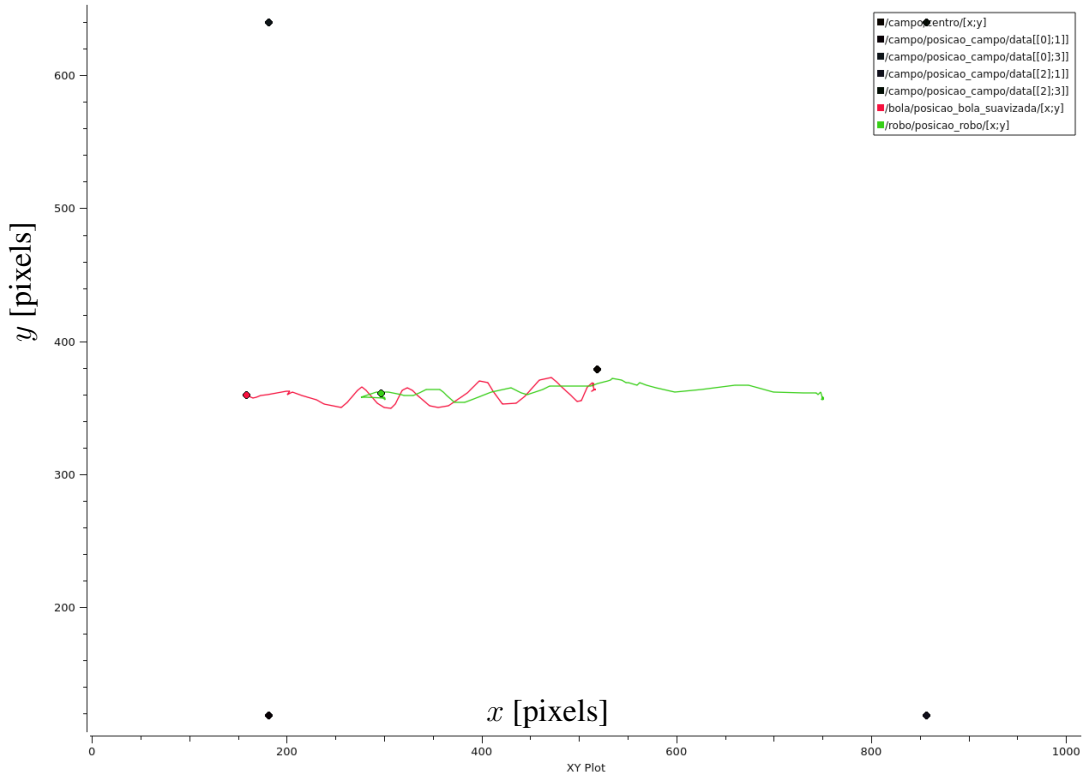
Figura 4.5 – Configuração inicial do segundo cenário do Experimento 1, com o robô alinhado e distante da bola.



Fonte: Autoria própria.

A Figura 4.6 apresenta novamente as trajetórias medidas da bola e do robô, permitindo observar a aproximação até o início da condução.

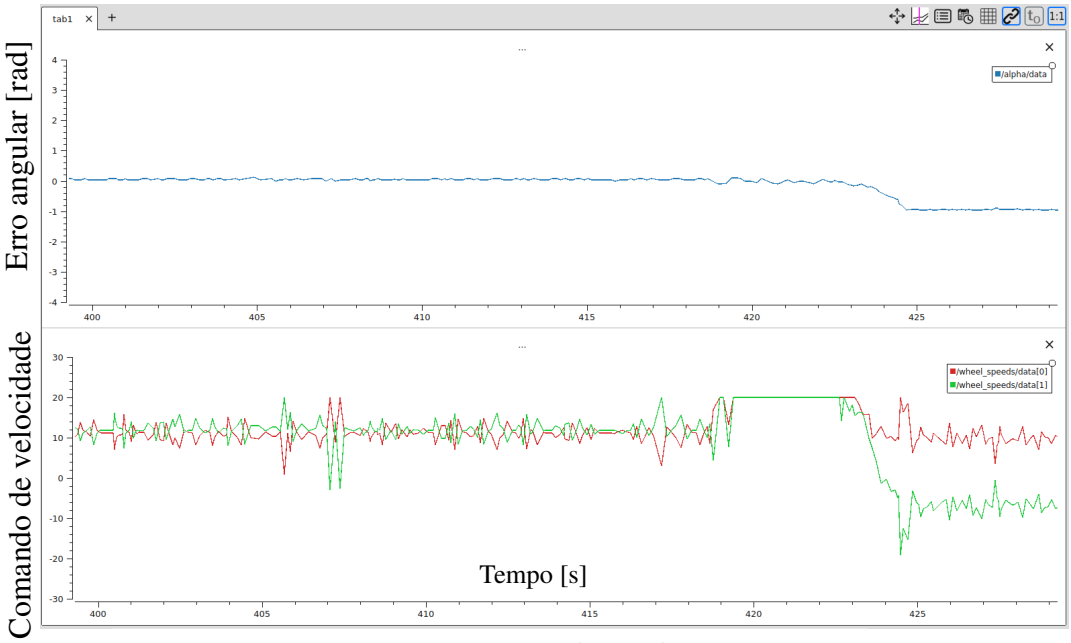
Figura 4.6 – Trajetórias da bola e do robô no segundo cenário do Experimento 1.



Fonte: Autoria própria.

Por fim, a Figura 4.7 apresenta, no painel superior, o erro angular α e, no painel inferior, as referências de velocidade angular das duas rodas. A curva vermelha do painel inferior corresponde a `/wheel_speeds/data[0]` e a curva verde a `/wheel_speeds/data[1]`. A mudança de sinal entre as referências evidencia a ação diferencial necessária para corrigir a orientação do robô.

Figura 4.7 – Erro angular (painel superior) e referências de velocidade das duas rodas (painel inferior) no segundo cenário do Experimento 1.

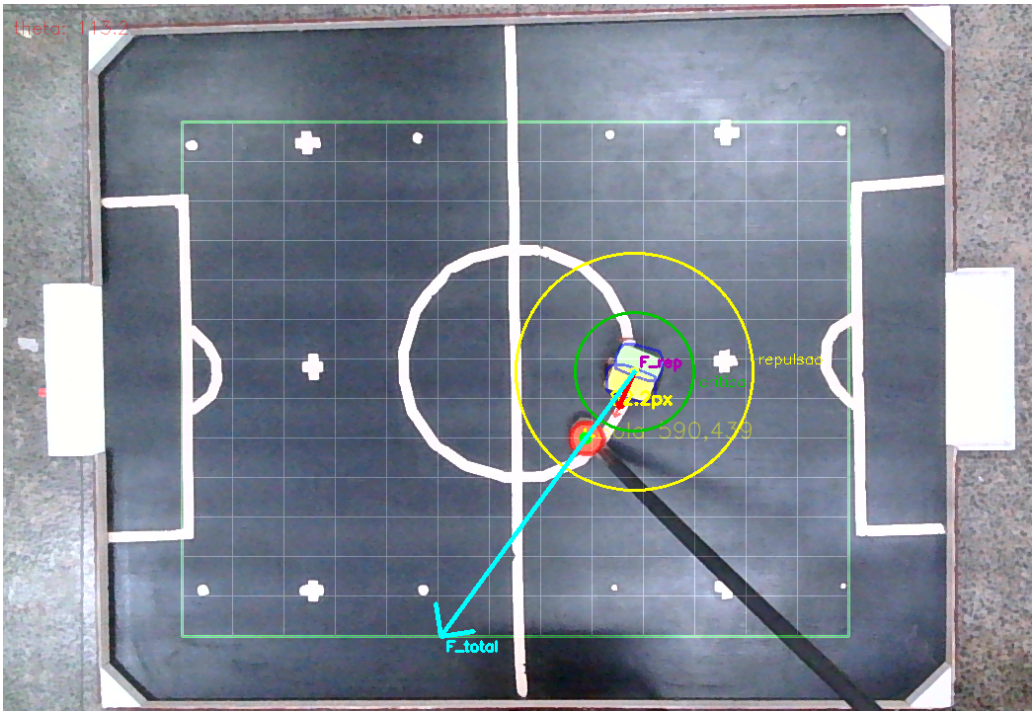


Fonte: Autoria própria.

4.3 Experimento 2: Perseguição de Trajetória Circular

No segundo experimento, a bola atua como uma referência móvel cuja posição varia ao longo de uma trajetória aproximadamente circular. O objetivo do ensaio é verificar se o sistema de visão, a predição do ponto de interceptação e o controlador conseguem atualizar continuamente a referência do robô diante dessa movimentação. Portanto, o movimento circular pertence à trajetória da bola durante o ensaio; o robô reage às posições medidas e procura persegui-la. A configuração inicial é apresentada na Figura 4.8.

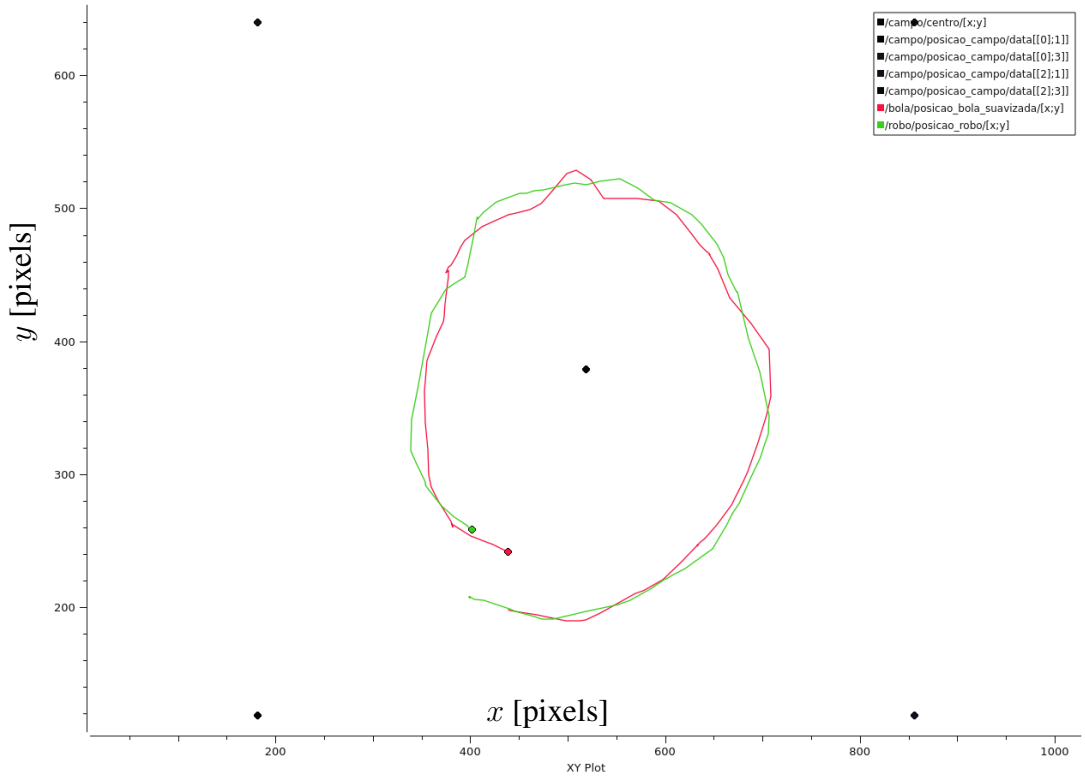
Figura 4.8 – Configuração inicial do Experimento 2, com a bola utilizada como referência móvel.



Fonte: Autoria própria.

A Figura 4.9 mostra as trajetórias registradas durante a perseguição. A curva vermelha representa a bola e a curva verde representa o robô. A semelhança entre os percursos evidencia que o robô atualiza sua direção à medida que a referência se desloca, embora permaneça um afastamento espacial associado à dinâmica de perseguição e ao ponto de interceptação empregado.

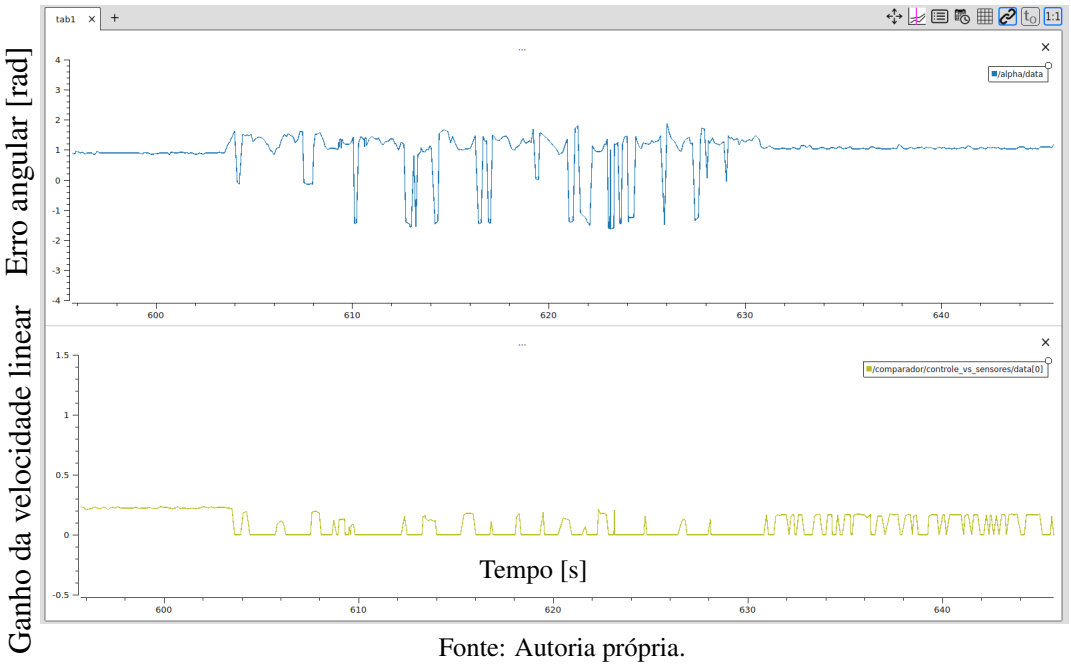
Figura 4.9 – Trajetórias da bola (vermelho) e do robô (verde) durante a perseguição circular no Experimento 2.



Fonte: Autoria própria.

A Figura 4.10 apresenta, no painel superior, o erro angular α durante a perseguição e, no painel inferior, a primeira componente do tópico de monitoramento `/comparador/controle_vel_ser`. As oscilações mais frequentes do erro angular são compatíveis com a necessidade de correções sucessivas enquanto a direção da referência varia continuamente.

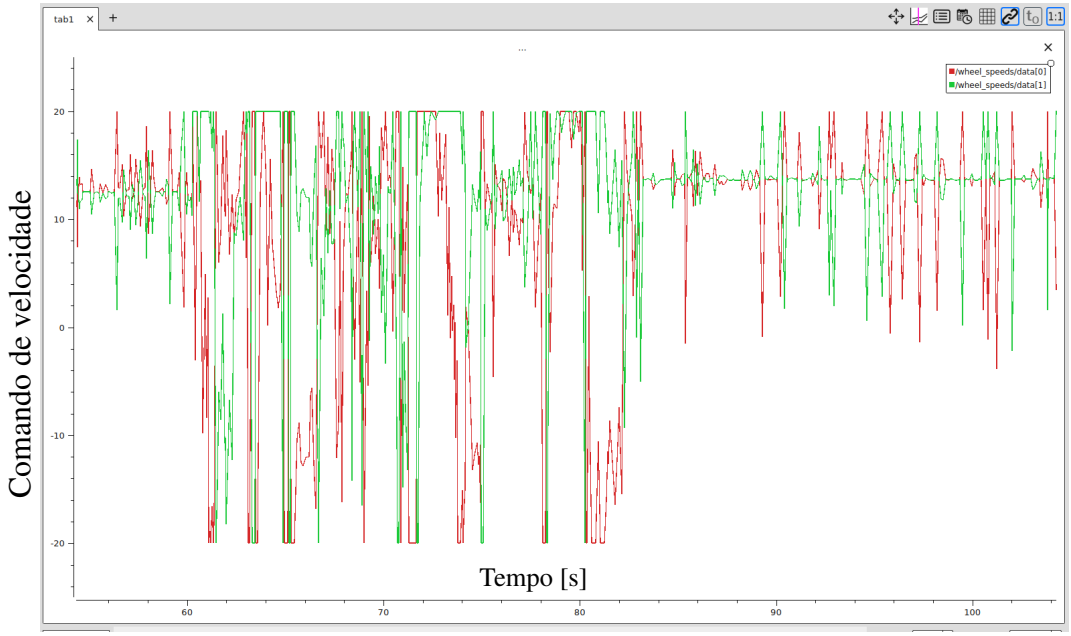
Figura 4.10 – Erro angular α (painel superior) e ganho da velocidade linear com ponto objetivo definido (painel inferior) durante o Experimento 2.



Fonte: Autoria própria.

As referências de velocidade angular das rodas são mostradas na Figura 4.11. A curva vermelha representa `/wheel_speeds/data[0]` e a curva verde representa `/wheel_speeds/data[1`. Em diversos instantes, as referências atingem os limites de ± 20 rad/s. Esse comportamento corresponde à saturação imposta às referências de velocidade das rodas antes da conversão para PWM no ESP32; portanto, os patamares em ± 20 rad/s não representam diretamente a saturação do ciclo de trabalho do PWM.

Figura 4.11 – Referências de velocidade angular das rodas durante o Experimento 2, limitadas ao intervalo de -20 a 20 rad/s.

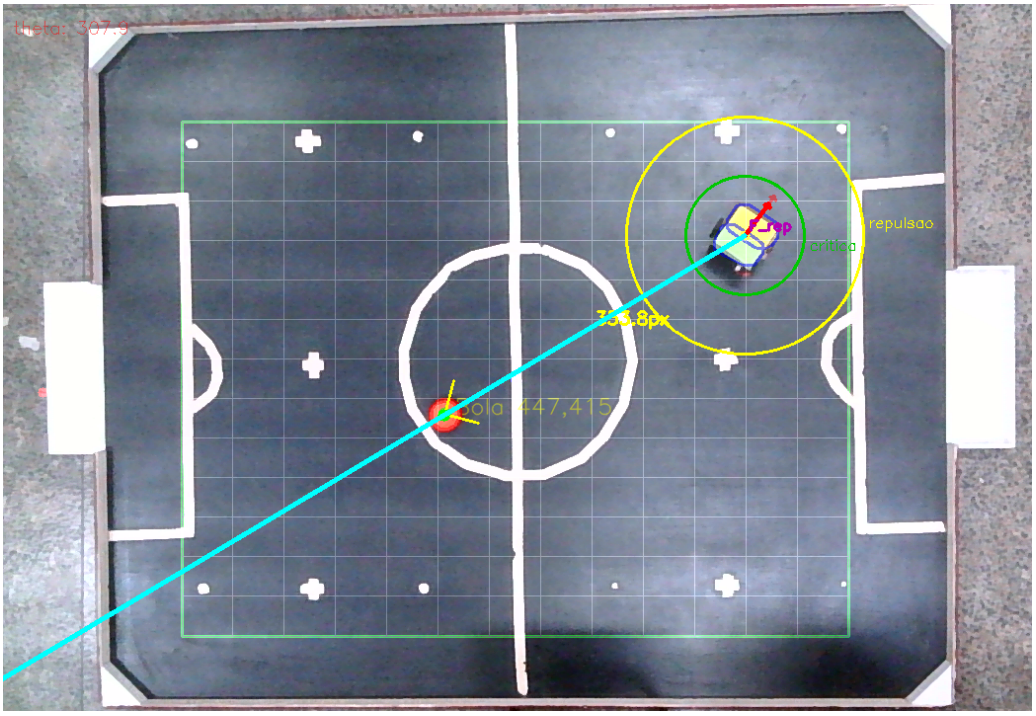


Fonte: Autoria própria.

4.4 Experimento 3: Reorientação sob Alta Variação Angular

O terceiro experimento submete o sistema a uma condição inicial de grande desalinhamento, com o robô orientado de costas para o alvo. Nessa situação, o controlador deve priorizar a correção angular antes do avanço, exigindo diferença significativa entre as referências das duas rodas. A configuração inicial é apresentada na Figura 4.12.

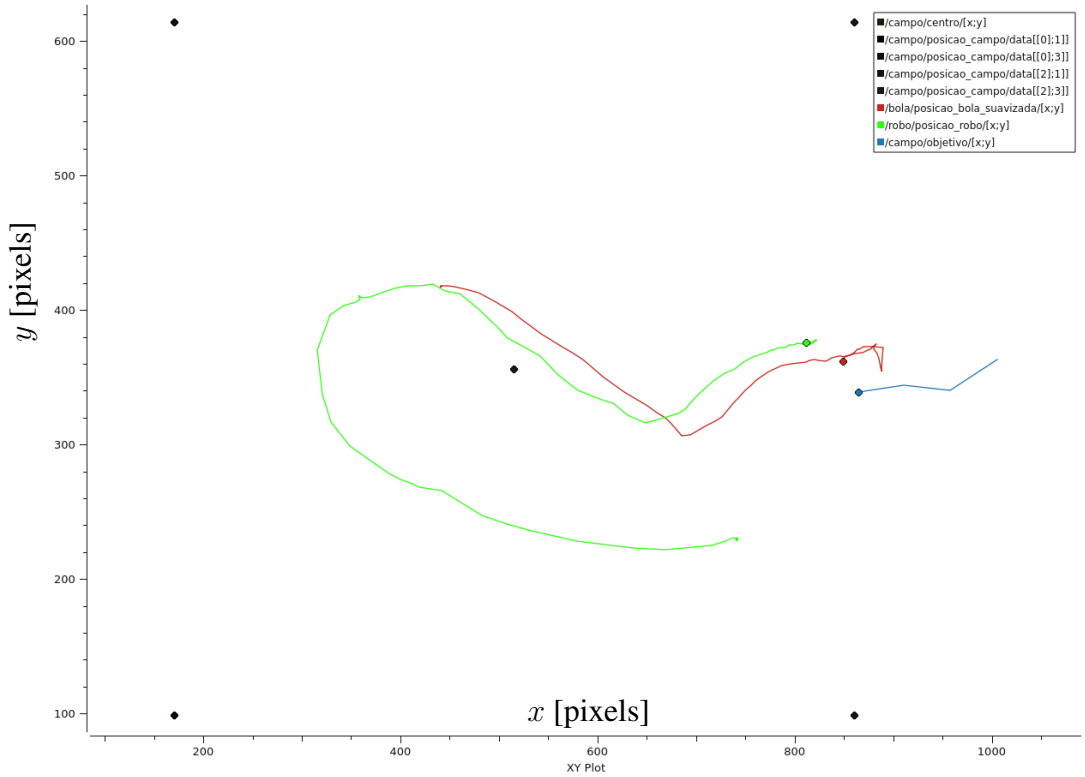
Figura 4.12 – Configuração inicial do robô no Experimento 3, com elevado erro de orientação em relação ao alvo.



Fonte: Autoria própria.

A trajetória resultante é apresentada na Figura 4.13. A curva verde representa o deslocamento do robô e a curva vermelha acompanha a posição da bola; o marcador/trecho azul identifica a referência de objetivo exibida no gráfico. O formato inicial da trajetória evidencia a manobra de reorientação realizada antes da aproximação ao alvo.

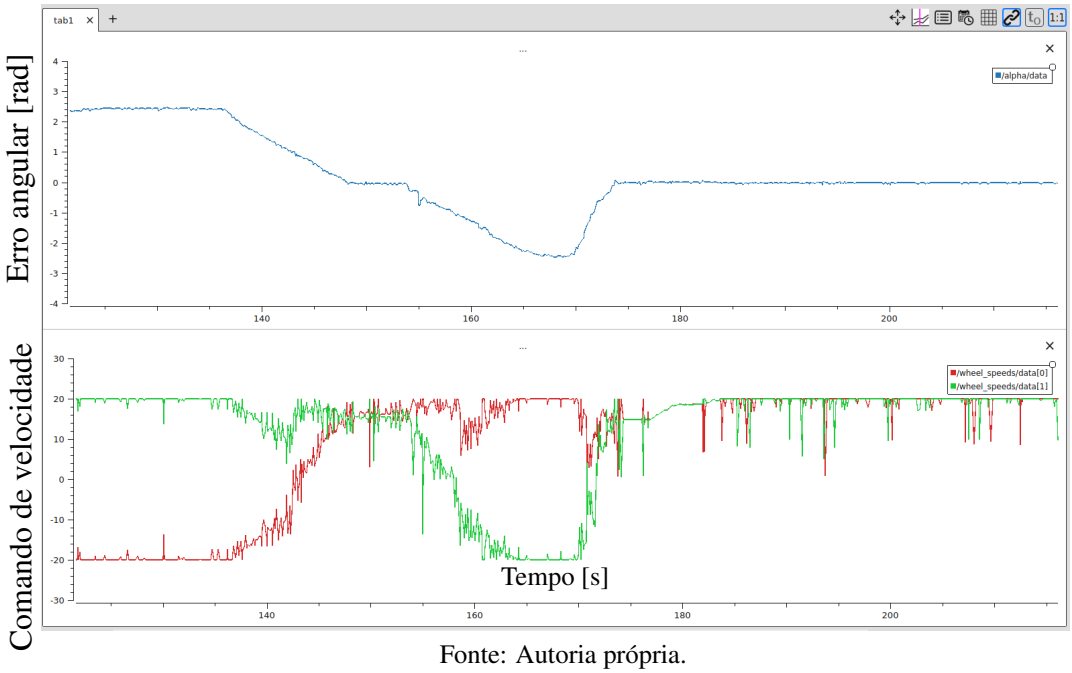
Figura 4.13 – Trajetória do robô durante a manobra de reorientação no Experimento 3.



Fonte: Autoria própria.

Na Figura 4.14, o painel superior mostra a evolução do erro angular α. O erro inicia com magnitude elevada e diminui à medida que o robô corrige sua orientação. No painel inferior são apresentadas as referências de velocidade angular das duas rodas: vermelho para `/wheel_speeds/data[0]` e verde para `/wheel_speeds/data[1]`. Durante a reorientação, as referências assumem sinais opostos e atingem o limite de ± 20 rad/s, comportamento coerente com a rotação do robô em torno do próprio eixo. Após o alinhamento, as referências retornam ao mesmo sentido para favorecer o deslocamento de aproximação.

Figura 4.14 – Erro angular (painel superior) e referências de velocidade angular das rodas (painel inferior) durante o Experimento 3.



Fonte: Autoria própria.

O registro em vídeo deste experimento, denominado “Sistemas de Controle e Navegação para Robôs da Categoria IEEE VSSS”, está disponível no Apêndice[1]

Figura 4.15 – QR Code para acesso ao vídeo do teste de navegação utilizando campos potenciais.

Fonte: Autoria própria.

[1] “Sistemas de Controle e Navegação para Robôs da Categoria IEEE VSSS”, está disponível no Apêndice H, Seção H.1.

5 CONCLUSÕES E TRABALHOS FUTUROS

5.1 Conclusões

O desenvolvimento deste projeto proporcionou uma experiência abrangente e enriquecedora, permitindo a aplicação prática de diversas ferramentas adquiridas ao longo do curso de Engenharia de Controle e Automação. A calibração minuciosa de parâmetros do sistema de controle, como os ganhos do PID angular, limites do Anti-Windup, velocidades máximas, campos potenciais e filtros de suavização, foi crucial para garantir a eficiência do sistema de navegação do robô.

A implementação dessas configurações possibilitou que o robô operasse de maneira estável, atingindo os resultados esperados em termos de precisão, tempo de resposta e condução da bola, demonstrando a eficácia do controle integrado entre software e hardware. Além disso, o trabalho com processamento de imagem e calibração de cores mostrou-se fundamental para o rastreamento confiável dos elementos no campo, integrando a percepção visual ao controle de movimento de forma coordenada.

Essa experiência permitiu não apenas validar conceitos teóricos estudados em sala de aula, mas também desenvolver habilidades práticas essenciais na resolução de problemas complexos de sistemas autônomos. O resultado final evidencia que o planejamento cuidadoso e a parametrização adequada de cada módulo do robô foram determinantes para o sucesso do projeto, consolidando um aprendizado completo sobre controle, automação e robótica móvel aplicada.

5.2 Trabalhos futuros

Para expandir e complementar o sistema desenvolvido, é desejável a construção de mais dois robôs, permitindo a formação de um time completo para simulações e partidas da categoria IEEE VSSS. Além disso, a migração do ambiente ROS 1 para o ROS 2 apresenta grande potencial, pois o ROS 2 oferece suporte nativo à comunicação multiagente, eliminando a necessidade de ferramentas externas para integração e coordenação entre múltiplos robôs.

Outro avanço possível seria a incorporação de técnicas de Inteligência Artificial, como redes neurais ou algoritmos de aprendizado de máquina, trabalhando em conjunto com os campos potenciais. Essa abordagem permitiria a predição de movimentos de outros robôs adversários e a otimização das trajetórias do time, aumentando a eficiência do controle, a

tomada de decisão em tempo real e a capacidade estratégica da equipe em cenários competitivos, os códigos-fonte desenvolvidos estão disponibilizados publicamente no repositório <https: //github.com/Elian-Souza>..

REFERÊNCIAS

AKIN, H. L. et al. Robocup rescue robot and simulation leagues. **AI magazine**, v. 34, n. 1, p. 78–78, 2013.

ÅSTRÖM, K. J.; HÄGGLUND, T. **PID Controllers: Theory, Design, and Tuning**. 2nd. ed. Research Triangle Park, North Carolina: International Society of Automation (ISA), 1995. ISBN 978-1556175169.

BASSANI, H. F. et al. A framework for studying reinforcement learning and sim-to-real in robot soccer. **arXiv preprint arXiv:2008.12624**, 2020.

CBR. **Regras Oficiais da Categoria Very Small Size Soccer (VSSS 3x3)**. [S.l.], 2021. Acessado em: 11 jun. 2026. Disponível em: <https://cbr.robocup.org.br/wp-content/uploads/2021/05/vssRules3x321.pdf>.

CORKE, P. **Robotics, Vision and Control: Fundamental Algorithms in Python**. 3rd. ed. Cham, Switzerland: Springer, 2023. ISBN 978-3031064685.

FAYYAD, J. et al. Deep learning sensor fusion for autonomous vehicle perception and localization: A review. **Sensors**, MDPI, v. 20, n. 15, p. 4220, 2020.

FERREIRA, R. H. X. et al. Initial development of ieee very small size soccer robots: Electromechanics, computer vision system and navigation strategy. **The Journal of Engineering and Exact Sciences**, v. 9, n. 10, p. 18148–18148, 2023.

KOUVAKAS, N. D.; KOUMBOULIS, F. N.; SIGALAS, J. A two stage nonlinear i/o decoupling and partially wireless controller for differential drive mobile robots. **Robotics**, MDPI, v. 13, n. 2, p. 26, 2024.

PARK, M. G.; JEON, J. H.; LEE, M. C. Obstacle avoidance for mobile robots using artificial potential field approach with simulated annealing. In: IEEE. **ISIE 2001. 2001 IEEE International Symposium on Industrial Electronics Proceedings (Cat. No. 01TH8570)**. [S.l.], 2001. v. 3, p. 1530–1535.

PINTO, A. H. M. **Regras IEEE Very Small Size Soccer (VSSS) – Série B**. 2021. Available upon request or internal distribution.

QUIGLEY, M. et al. Ros: an open-source robot operating system. In: KOBE. **ICRA workshop on open source software**. [S.l.], 2009. v. 3, n. 3.2, p. 5.

REN, J.; MCISAAC, K. A.; PATEL, R. V. Modified newton's method applied to potential field-based navigation for mobile robots. **IEEE transactions on robotics**, IEEE, v. 22, n. 2, p. 384–391, 2006.

RIBEIRO, D. d. S. S. et al. Ieee vsss dynamic model identification using standard maneuvers. In: IEEE. **2025 Brazilian Symposium on Robotics (SBR) and 2025 Workshop on Robotics in Education (WRE)**. [S.l.], 2025. p. 194–199.

SAMODRO, M. M. J. Artificial potential field path planning algorithm in differential drive mobile robot platform for dynamic environment. **International Journal of Robotics and Control Systems**, v. 3, n. 2, 2023.

SICILIANO, B. et al. **Robotics: Modelling, Planning and Control**. [S.l.]: Springer, 2010.

SIEGWART, R.; NOURBAKHSH, I. R.; SCARAMUZZA, D. **Introduction to Autonomous Mobile Robots**. 2nd. ed. Cambridge, Massachusetts: MIT Press, 2011. ISBN 978-0262015356.

SIGUT, J. et al. Opencv basics: A mobile application to support the teaching of computer vision concepts. **IEEE Transactions on Education**, IEEE, v. 63, n. 4, p. 328–335, 2020.

SZELISKI, R. **Computer Vision: Algorithms and Applications**. [S.l.]: Springer, 2010.

VÁZQUEZ-HURTADO, C.; RODRIGUEZ-PADILLA, C. Adapting a very small size soccer (vsss) competition for learning robotics in virtual teaching. In: IEEE. **2022 IEEE Global Engineering Education Conference (EDUCON)**. [S.l.], 2022. p. 553–558.

APÊNDICE A – RECEPÇÃO DAS POSIÇÕES DA BOLA

Em ambientes estruturados, como os utilizados na categoria IEEE VSSS, a percepção em duas dimensões é suficiente para que um robô terrestre realize a navegação e interação com o campo de jogo.

Para isso, foram desenvolvidos os nós `detector_bola` e `detector_robo`, responsáveis pela identificação dos elementos principais da partida: a bola e o robô. Ambos os nós recebem imagens do tópico `/camera_raw` e processam os dados em tempo real utilizando técnicas de visão computacional implementadas com a biblioteca OpenCV.

O processamento das imagens segue etapas semelhantes para a bola e para o robô. Inicialmente, as imagens são convertidas para o espaço de cores HSV, que proporciona maior robustez para segmentação baseada em cores. Em seguida, são aplicadas máscaras específicas: a bola é detectada pela cor laranja, enquanto o robô é identificado por marcações nas cores verde e amarela. A partir das regiões segmentadas, são extraídas as informações de posição (coordenadas x e y) e, no caso do robô, também a orientação no campo.

As informações obtidas são publicadas em tópicos ROS e integradas aos módulos de navegação, planejamento e controle, permitindo que o sistema acompanhe continuamente o estado do jogo e execute as ações necessárias de forma eficiente.

A.1 Nó de Detecção da Bola

A posição da bola é determinada pelo cálculo do centróide do contorno segmentado, resultando em coordenadas em pixels nos eixos horizontal (x) e vertical (y).

Essas coordenadas são publicadas no tópico `/bola/posicao_bola` utilizando mensagens do tipo `Point`, tornando os dados disponíveis para os demais módulos do sistema. Adicionalmente, uma imagem processada com marcações visuais da bola é publicada no tópico `/bola/frame_bola`, facilitando a validação e testes do algoritmo.

A.1.1 Filtro de Suavização da Bola

Para melhorar a estabilidade da posição da bola detectada pelo sistema de visão, foi implementado um nó ROS denominado `ssl_smoother_node`, responsável por suavizar os dados recebidos do tópico `/bola/posicao_bola`. A saída suavizada é publicada no tópico

`/bola/posicao_bola_suavizada`, permitindo que os módulos de navegação e controle utilizem informações mais consistentes.

O filtro combina duas técnicas: um suavizador exponencial e uma previsão utilizando uma curva de Bézier de grau 2. Seja $p_k = [x_k, y_k]^T$ a posição bruta da bola recebida na iteração k, s_{k-1} a posição suavizada calculada na iteração anterior e $s_k = [x_k^s, y_k^s]^T$ a nova posição suavizada. Na implementação deste trabalho, o suavizador exponencial é calculado de forma recursiva por:

$$s_k = \alpha\, p_k + (1 - \alpha)\, s_{k-1}. \tag{A.1}$$

Nessa expressão, $\alpha \in [0, 1]$ é exclusivamente o coeficiente de suavização da posição da bola e não deve ser confundido com o erro angular α utilizado no controlador. Valores menores de α atribuem maior peso ao histórico s_{k-1} e produzem maior suavização; valores próximos de 1 atribuem maior peso à medição atual p_k.

Para antecipar movimentos leves da bola, é aplicada uma previsão através de uma curva de Bézier de grau 2 utilizando os três últimos pontos históricos p_{k-2}, p_{k-1} e p_k. A posição prevista s_k^{bezier} é dada por:

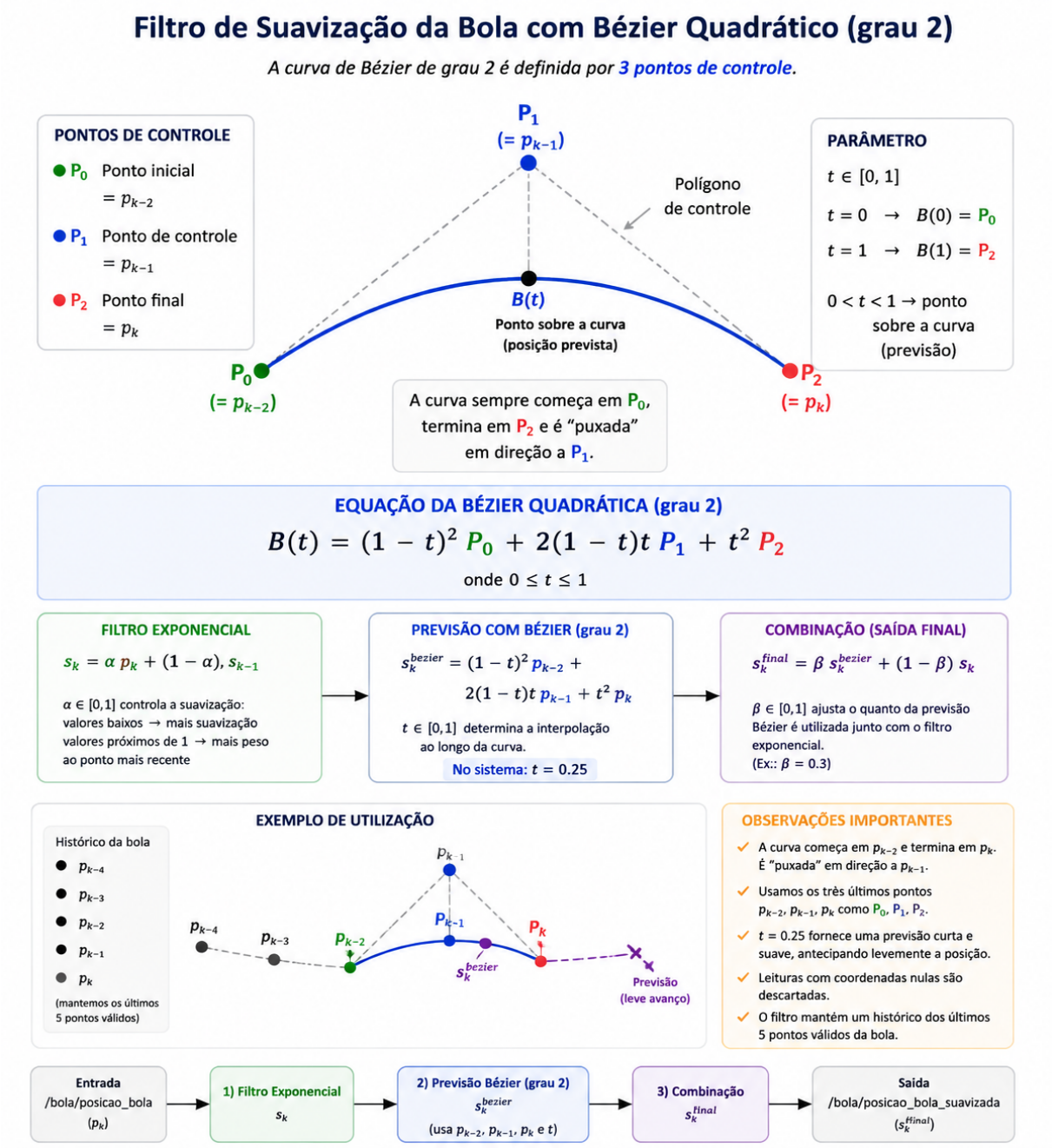
$$s_k^{\text{bezier}} = (1 - t)^2 p_{k-2} + 2(1 - t)t\, p_{k-1} + t^2 p_k, \tag{A.2}$$

onde $t \in [0, 1]$ determina a interpolação ao longo da curva. No sistema implementado, utilizou-se $t = 0{,}25$ para fornecer um leve avanço, antecipando a posição da bola de forma controlada.

O filtro mantém um histórico dos últimos cinco pontos válidos da bola para aplicação da curva de Bézier, descartando leituras com coordenadas nulas. A combinação do filtro exponencial com a previsão da curva permite reduzir ruídos provenientes da detecção visual e melhorar a consistência dos dados utilizados pelo robô na tomada de decisão durante a partida.

A Figura A.1 apresenta de forma esquemática o funcionamento da previsão quadrática. Os três pontos de controle correspondem às posições recentes da bola, e o parâmetro t determina a posição prevista sobre a curva. O diagrama também evidencia a sequência entre entrada da posição, suavização exponencial, previsão por Bézier e publicação da posição suavizada.

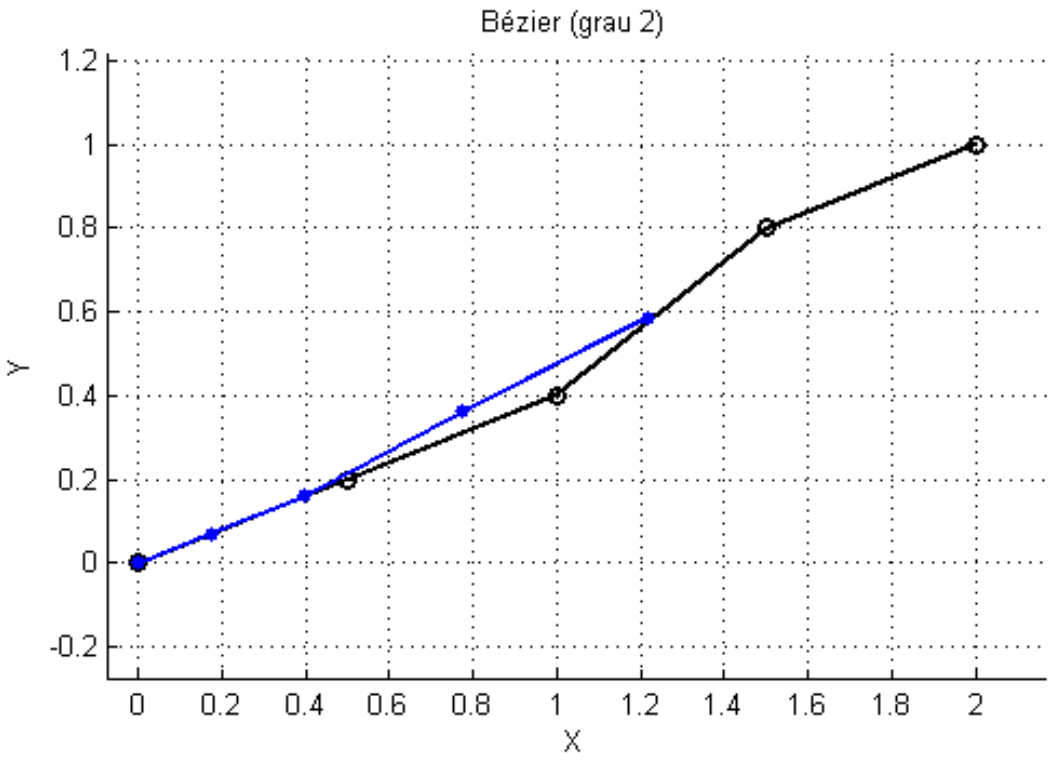
Figura A.1 – Esquema do filtro de suavização e da previsão por curva de Bézier quadrática.



Fonte: Autoria própria.

A Figura A.2 mostra um exemplo geométrico da curva gerada a partir de pontos sucessivos. A curva de previsão suaviza a transição entre as amostras e fornece um ponto intermediário que pode ser utilizado pelo módulo de perseguição sem substituir o histórico de posições medido.

Figura A.2 – Exemplo de interpolação por curva de Bézier quadrática a partir de pontos sucessivos.



Fonte: Autoria própria.

Observações importantes:

- O histórico é armazenado em uma fila de tamanho máximo cinco, garantindo que apenas os pontos recentes influenciem a previsão.
- A saída suavizada é publicada como mensagem `geometry_msgs/Point`, com $z = 0$.
- Este método é particularmente útil em ambientes com ruído de detecção ou variações rápidas da posição da bola.

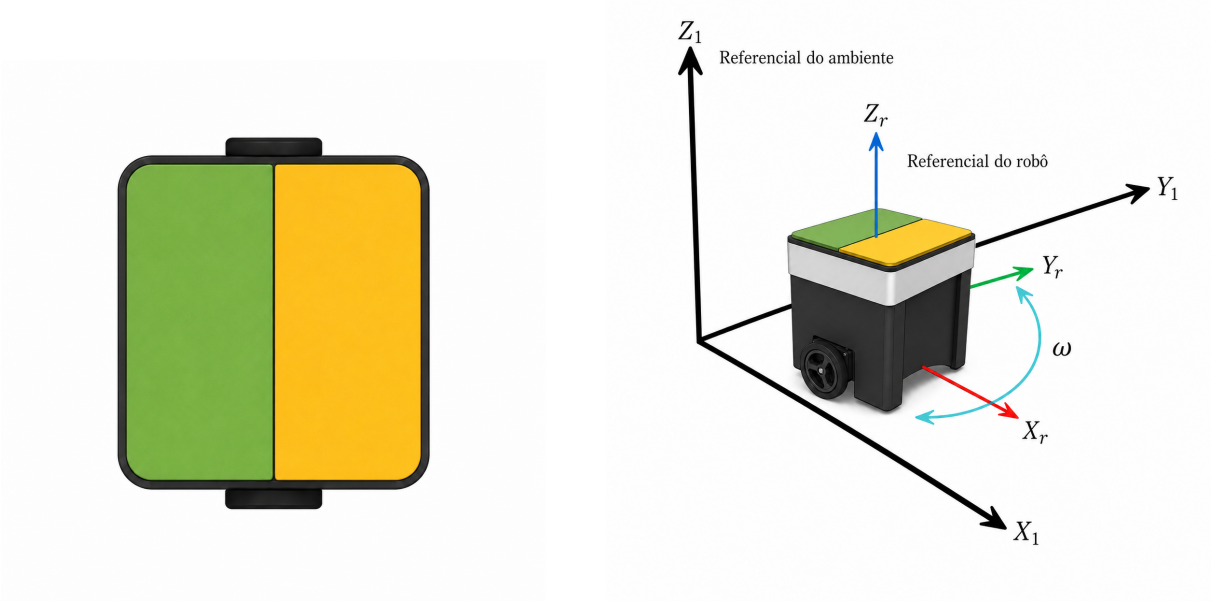
APÊNDICE B – NÓ DE DETECÇÃO DO ROBÔ

A detecção do robô segue um processo similar, aplicando máscaras para as cores verde (marca traseira) e amarela (marca frontal). Após a segmentação, são aplicados filtros morfológicos e técnicas de suavização para reduzir ruídos e aprimorar a precisão da detecção.

A partir dos contornos identificados, os centróides das marcas frontal e traseira são calculados. O ponto médio entre essas marcas define a posição do robô, enquanto a orientação é obtida pelo vetor que conecta as marcas, com o ângulo calculado utilizando a função `atan2()`.

Os dados de posição e orientação são publicados no tópico `/robo/posicao_robo`, onde as coordenadas `x` e `y` representam a posição do robô e a coordenada `z` armazena o ângulo de orientação em radianos. Similarmente à bola, uma imagem processada contendo as marcações e o vetor de orientação é publicada no tópico `/robo/frame_robo`, auxiliando na verificação e calibração do sistema.

Figura B.1 – Representação das cores do robô e plano cartesiano, considerando seu deslocamento no plano 2D pelas coordenadas (X_r, Y_r) e sua rotação angular ω em torno do próprio eixo vertical Z_r, definido pelo centro entre as duas marcações de cores do robô.



(a) Representação vista superior.

(b) Representação plano cartesiano.

Fonte: Autoria própria.

Figura B.2 – Definição do tamanho total da área de detecção utilizada para preservar a orientação frontal do robô.



(a) Representação de área frontal obstruída.



(b) Representação da expansão da área frontal de detecção.

Fonte: Autoria própria.

APÊNDICE C – IMPLEMENTAÇÃO DA ESTRATÉGIA E FILTRO DO ERRO ANGULAR

A velocidade da bola é estimada a partir da diferença entre duas posições consecutivas:

$$v_{x,\text{bola}} = \frac{x_{\text{bola}}(k) - x_{\text{bola}}(k-1)}{\Delta t} \tag{C.1}$$

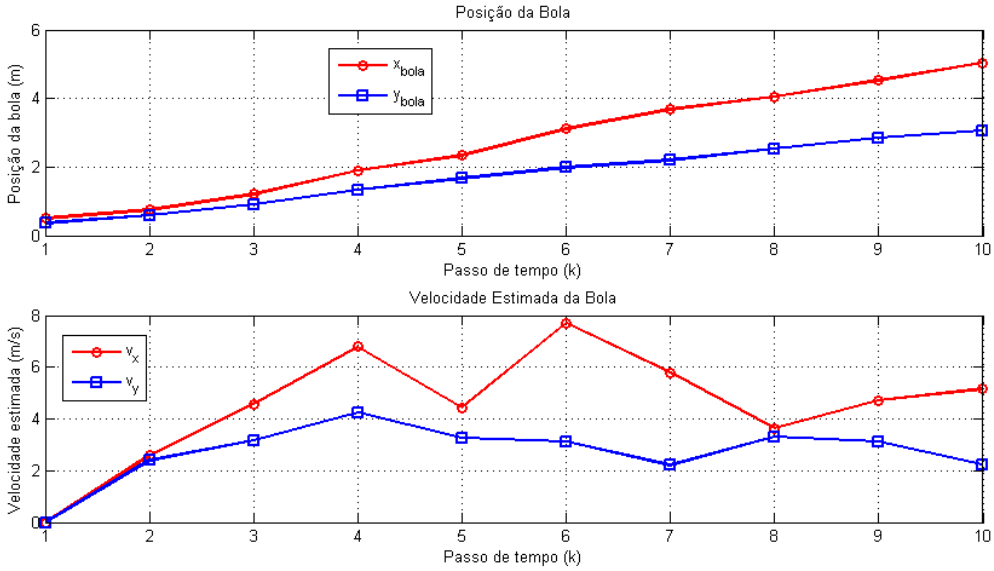
$$v_{y,\text{bola}} = \frac{y_{\text{bola}}(k) - y_{\text{bola}}(k-1)}{\Delta t} \tag{C.2}$$

Em que:

- $x_{\text{bola}}(k)$ e $y_{\text{bola}}(k)$ são as coordenadas atuais da bola;
- $x_{\text{bola}}(k-1)$ e $y_{\text{bola}}(k-1)$ são as coordenadas anteriores;
- Δt é o intervalo de tempo entre duas leituras consecutivas.

A Figura C.1 exemplifica o cálculo a partir de uma sequência de posições. No painel superior são apresentadas as coordenadas da bola ao longo das amostras; no painel inferior aparecem as componentes de velocidade estimadas pelas diferenças sucessivas. Dessa forma, variações maiores entre duas posições consecutivas produzem maiores valores de velocidade estimada.

Figura C.1 – Exemplo de posições da bola e das componentes de velocidade estimadas por diferenças sucessivas.



Fonte: Autoria própria.

C.1 Predição do ponto de interceptação

Para antecipar a posição da bola, o robô calcula um ponto de interceptação futuro com base na posição atual e na velocidade estimada da bola.

C.2 Ponto inicial de interceptação

$$x_{\text{alvo}} = x_{\text{bola}} + v_{x,\text{bola}} \cdot 0{,}25 \tag{C.3}$$

$$y_{\text{alvo}} = y_{\text{bola}} + v_{y,\text{bola}} \cdot 0{,}25 \tag{C.4}$$

- $(x_{\text{bola}}, y_{\text{bola}})$: posição atual da bola.
- $(v_{x,\text{bola}}, v_{y,\text{bola}})$: velocidade estimada da bola nos eixos X e Y.
- $0{,}25$ s : horizonte inicial de previsão (tempo à frente que o robô estima a posição da bola).

C.3 Distância até o ponto previsto

$$d = \sqrt{(x_{\text{alvo}} - x_r)^2 + (y_{\text{alvo}} - y_r)^2} \tag{C.5}$$

- (x_r, y_r) : posição atual do robô.
- d : distância entre o robô e o ponto de interceptação.

C.4 Definição do *Lookahead* (horizonte de previsão)

O *lookahead* define a distância, medida a partir da posição atual do robô, até uma referência intermediária posicionada na direção do ponto previsto da bola. Em vez de comandar o robô diretamente para uma referência distante, utiliza-se um ponto à frente limitado por valores mínimo e máximo. Esse procedimento evita referências muito próximas, que poderiam provocar correções excessivamente frequentes, e também evita referências muito distantes, que reduziriam a capacidade de reação às mudanças de trajetória.

$$L = \min\left(\max\left(0{,}5\,d,\ L_{\min}\right),\ L_{\max}\right). \tag{C.6}$$

- L: distância utilizada para posicionar a referência intermediária;
- $L_{\min}$: menor valor permitido para o *lookahead*;
- $L_{\max}$: maior valor permitido para o *lookahead*;
- $0{,}5d$: valor nominal adotado antes da aplicação dos limites mínimo e máximo.

C.5 Reposicionamento do alvo na direção prevista

$$x_{\text{alvo}} = x_r + \frac{x_{\text{alvo}} - x_r}{d}\,L \tag{C.7}$$

$$y_{\text{alvo}} = y_r + \frac{y_{\text{alvo}} - y_r}{d}\,L \tag{C.8}$$

- Essa etapa ajusta o alvo do robô na direção do ponto previsto da bola, mantendo a distância definida pelo lookahead L.
- Garante que o robô se mova de forma antecipada e suave, sem precisar ir exatamente até a posição instantânea da bola.

Observação: Essa estratégia melhora a interceptação da bola, permitindo que o robô se antecipe ao movimento da bola e reduza erros de atraso na ação.

C.6 Erro de distância e erro angular

A distância até o alvo é calculada por:

$$\rho = \sqrt{(x_{\text{alvo}} - x_r)^2 + (y_{\text{alvo}} - y_r)^2} \tag{C.9}$$

O ângulo do alvo em relação ao robô é:

$$\gamma = \arctan 2(y_{\text{alvo}} - y_r,\ x_{\text{alvo}} - x_r) \tag{C.10}$$

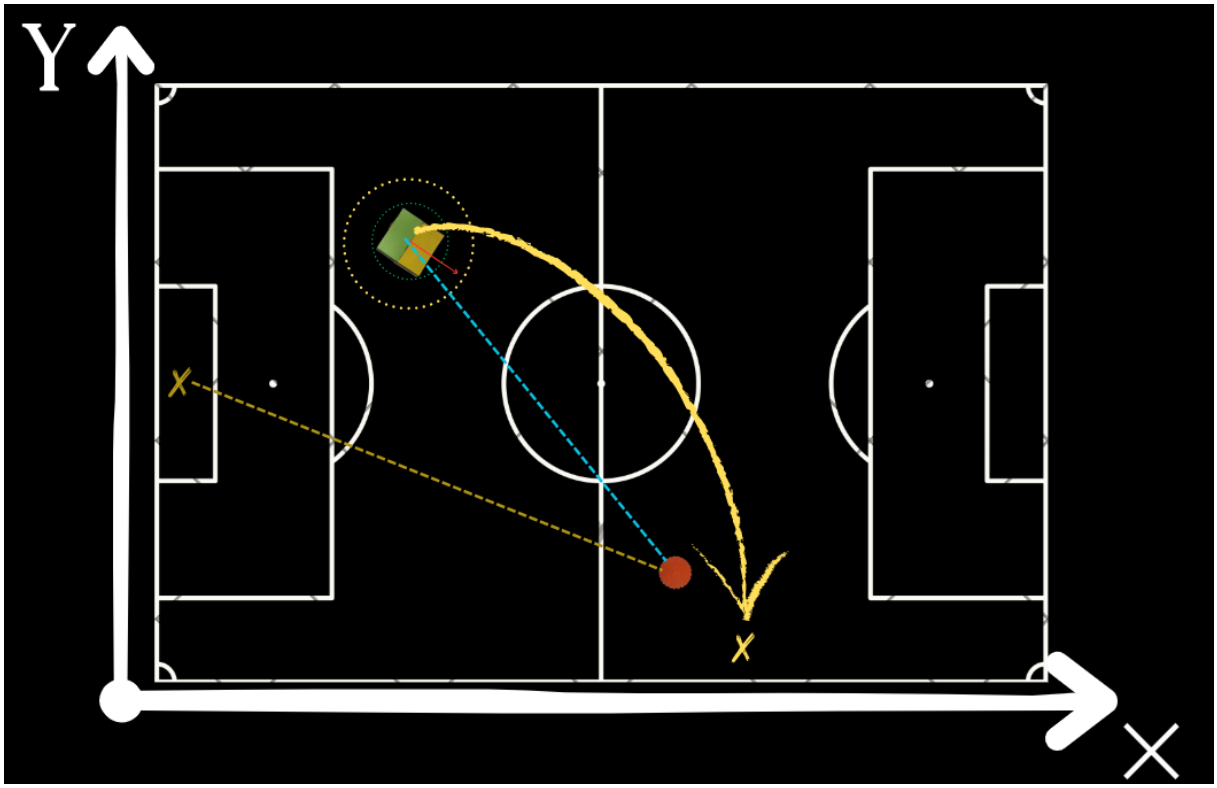
O erro angular é dado por:

$$\alpha = \gamma - \Theta_r \tag{C.11}$$

sendo posteriormente ajustado ao intervalo $[-\pi, \pi]$.

A Figura C.2 representa geometricamente o ponto de condução utilizado para posicionar o robô atrás da bola em relação ao objetivo. A linha entre a bola e o objetivo define a direção desejada de condução, e a referência do robô é deslocada para o lado oposto dessa direção. Com isso, o robô procura se aproximar da bola sem bloquear a direção de deslocamento pretendida em direção ao gol.

Figura C.2 – Representação geométrica do ponto de condução posicionado atrás da bola em relação ao objetivo.



Fonte: Autoria própria.

C.7 Filtro Exponencial do Erro Angular

O filtro exponencial é utilizado para suavizar o erro angular α, reduzindo oscilações rápidas ou ruídos provenientes de sensores, garantindo um controle mais estável do robô. A Tabela E.4, apresentada no Apêndice , reúne os parâmetros utilizados para filtragem e suavização do erro angular.

C.7.1 Equação do filtro

$$\alpha_f(k) = \beta\,\alpha_f(k-1) + (1-\beta)\,\alpha(k) \tag{C.12}$$

- $\alpha_f(k)$: erro angular filtrado no instante atual.
- $\alpha_f(k-1)$: erro angular filtrado no instante anterior.
- $\alpha(k)$: erro angular atual medido ou calculado.
- β : parâmetro do filtro, $0 \leq \beta \leq 1$, que define o peso do histórico versus a medida atual.

C.7.2 Como funciona

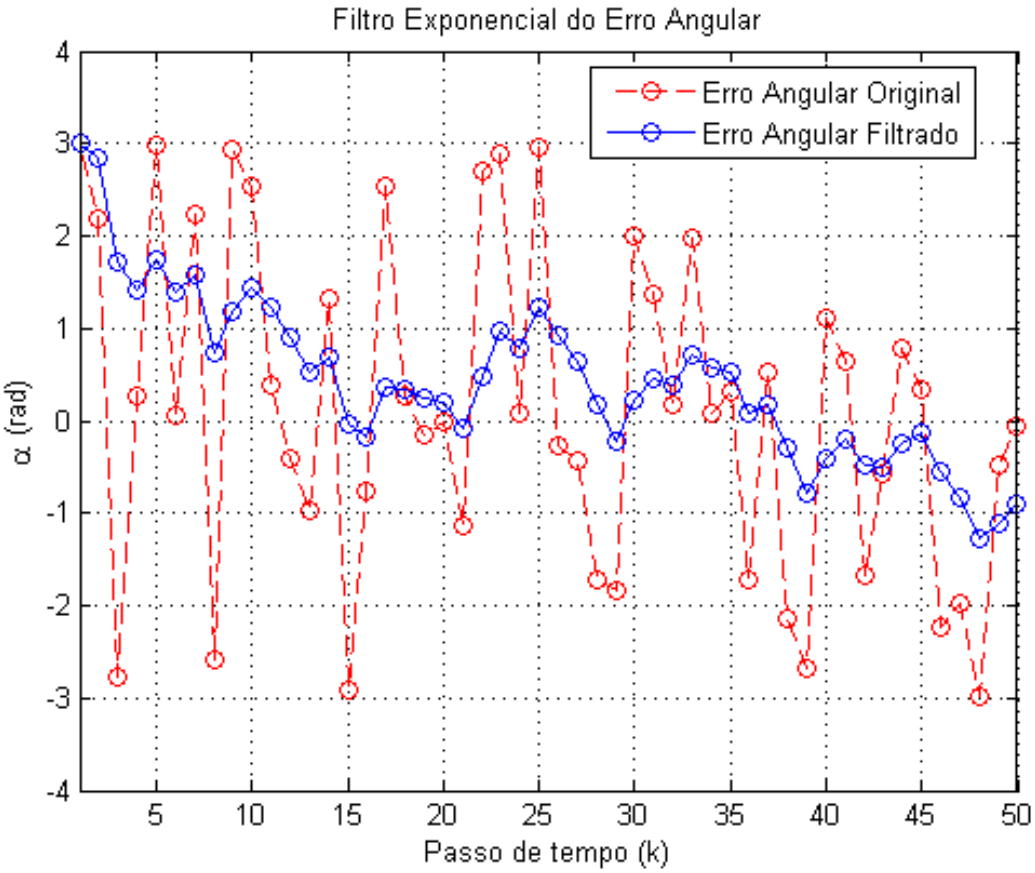
- O filtro combina o erro atual com o histórico do erro anterior.
- Quanto maior β, maior influência do histórico, resultando em um erro suavizado e resposta mais lenta.
- Quanto menor β, maior influência do erro atual, tornando o filtro mais responsivo, porém menos suave.

C.7.3 Finalidade

- Reduz ruído proveniente de medições imprecisas do ângulo do robô.
- Evita movimentos bruscos ou instáveis no controle angular.
- Melhora a precisão do PID angular, tornando a trajetória mais estável e previsível.

Observação: Esse tipo de filtro é simples, eficiente e muito usado em sistemas de controle onde é necessário balancear suavidade e resposta rápida.

Figura C.3 – Comportamento do filtro exponencial aplicado ao erro angular.



Fonte: Autoria própria.

C.8 Correção Lateral Durante o Domínio da Bola

Quando o robô está controlando a bola, é necessário ajustar lateralmente a posição para mantê-la centralizada em relação ao robô. A correção lateral é calculada por:

$$e_{\text{lat}} = -\sin(\alpha)\, d_{\text{bola}} \tag{C.13}$$

- e_{lat} : erro lateral, indicando quanto o robô deve ajustar sua orientação para centralizar a bola.
- α : ângulo entre a direção do robô e a linha que liga o robô à bola.
- d_{bola} : distância entre o robô e a bola.
- $-\sin(\alpha)$: transforma a diferença angular em uma correção lateral proporcional.

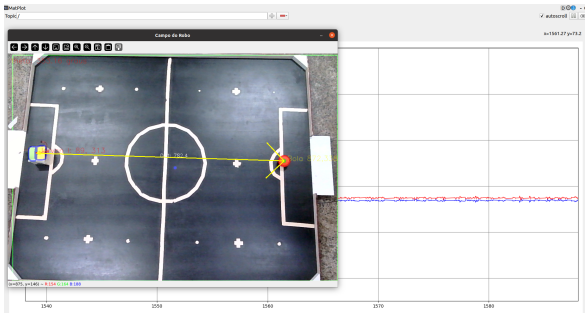
Essa correção é incorporada à velocidade angular do robô:

$$w = w + k_{\text{lat}}\, e_{\text{lat}} \tag{C.14}$$

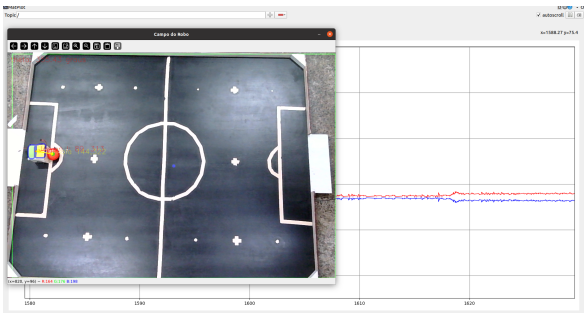
- w : velocidade angular atual do robô (rad/s).
- k_{lat} : ganho da correção lateral, controla a intensidade da resposta do robô.
- e_{lat} : erro lateral calculado anteriormente.

Observação: Esse ajuste garante que o robô mantenha a bola centralizada enquanto se move, corrigindo desvios laterais e melhorando o controle durante o domínio.

Figura C.4 – Ativação da correção lateral durante o domínio da bola.



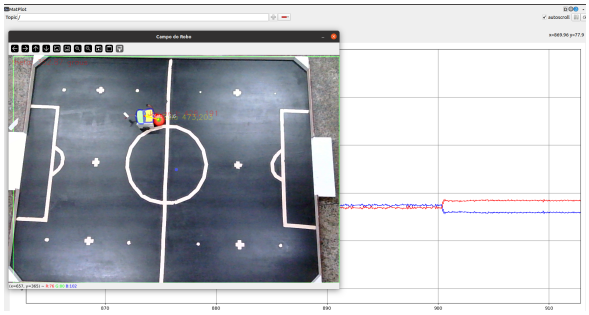
(a) Robô distante da bola, sem ativação da correção lateral.



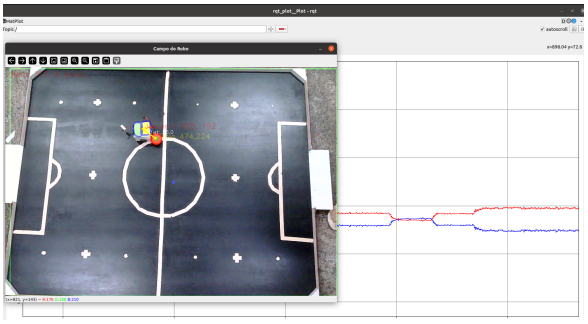
(b) Robô próximo da bola, com ativação da correção lateral.

Fonte: Autoria própria.

Figura C.5 – Resposta da correção lateral durante o domínio da bola.



(a) Aumento da ação de controle com o crescimento do erro de correção.



(b) Aumento da ação de controle para reduzir a perda de domínio da bola.

Fonte: Autoria própria.

APÊNDICE D – IMPLEMENTAÇÃO DO CONTROLE NO ESP32

O robô móvel utiliza um microcontrolador ESP32 como núcleo do sistema de controle, comunicação e aquisição de dados. Nesta seção são detalhadas as principais funcionalidades implementadas no firmware, incluindo conexão Wi-Fi, recepção de comandos, geração de sinais PWM para motores e publicação de dados dos sensores (encoders e IMU) via ROS.

Este apêndice documenta a implementação de controle e comunicação do robô no ESP32, servindo como referência para manutenção e desenvolvimento de novos módulos do sistema.

D.1 Conexão Wi-Fi e Comunicação ROS

O ESP32 estabelece conexão com a rede Wi-Fi do laboratório utilizando o SSID e senha definidos:

```
const char* ssid="TP-Link_ROS";
const char* password="sirolab2024";
```

Após a conexão, o microcontrolador conecta-se ao *ROS Master* em um endereço IP e porta específicos, permitindo troca de mensagens via tópicos:

- `wheel_speeds`: recebe velocidades alvo das rodas.
- `motors_enable`: habilita ou desabilita os motores.
- `wheel_encoders`: publica dados dos encoders das rodas.
- `imu_raw`: publica leituras do acelerômetro e giroscópio (MPU6050).

D.2 Recepção de sinais de controle

Os comandos de velocidade das rodas são recebidos em um *subscriber*:

```
void cmdCallback(const std_msgs::Float32MultiArray &msg){
    if(msg.data_length>0) target1=msg.data[0];
    if(msg.data_length>1) target2=msg.data[1];
}
```

Um segundo *subscriber* controla o estado dos motores:

```
void enableCallback(const std_msgs::Bool &msg){
    motors_enable=msg.data;
}
```

D.3 Geração do PWM

O controle de velocidade dos motores é realizado por PWM utilizando o módulo `ledc` do ESP32. A função `mapPWM` ajusta o valor recebido de velocidade para o intervalo de PWM definido:

```
int mapPWM(float s){
    if(s>20)s=20;
    if(s<-20)s=-20;

    if(abs(s)<0.001)return 0;

    float f=abs(s)/20.0;
    int pwm=MIN_PWM+(MAX_PWM-MIN_PWM)*f;

    if(s<0)pwm=-pwm;
    return pwm;
}
```

O sinal PWM é aplicado aos pinos dos motores com polaridade adequada:

```
void motor(int pwm,int a,int b){
    if(pwm>0){
        ledcWrite(a,pwm);
        ledcWrite(b,0);
    }
    else if(pwm<0){
        ledcWrite(a,0);
```

```
        ledcWrite(b,-pwm);
    }
    else{
        ledcWrite(a,0);
        ledcWrite(b,0);
    }
}
```

O *task* de controle executa a atualização do PWM a 500 Hz, garantindo resposta rápida aos comandos.

D.4 Publicação dos sensores

O ESP32 lê os encoders das rodas via interrupções:

```
void IRAM_ATTR isr1(){
  if(digitalRead(ENC1A)==digitalRead(ENC1B)) enc1++;
  else enc1--;
}
```

E a IMU é acessada via I2C (MPU6050), convertendo valores brutos em aceleração (g) e velocidade angular (°/s). Os dados são publicados a 30 Hz no ROS:

```
enc_pub.publish(&enc_msg);
imu_pub.publish(&imu_msg);
```

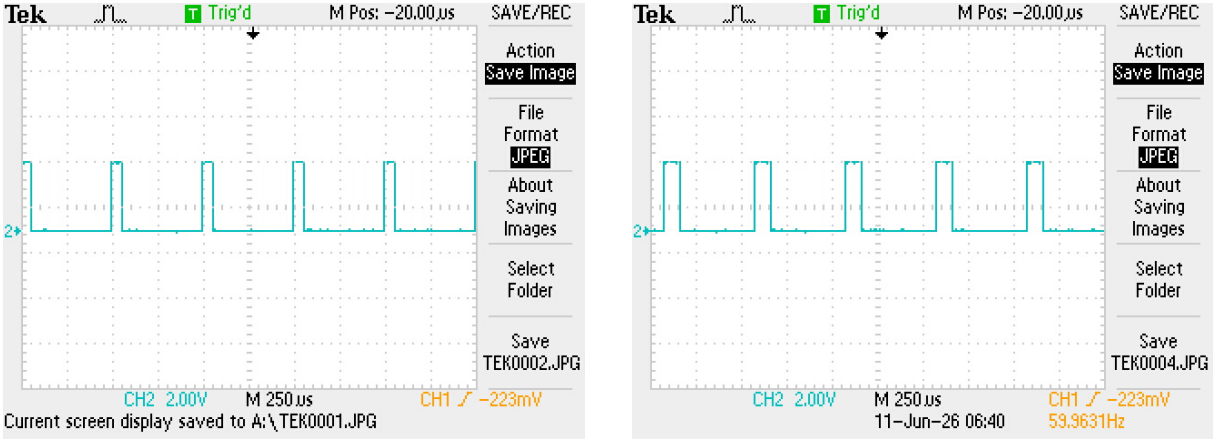
D.5 Resumo das cores e pinos

Tabela D.1 – Mapa de pinos do ESP32 para motores e sensores

Componente	Pinos ESP32	Função
Motor 1 IN1	26	PWM
Motor 1 IN2	27	PWM
Motor 2 IN1	32	PWM
Motor 2 IN2	33	PWM
Encoder 1A	35	Entrada digital + ISR
Encoder 1B	34	Entrada digital
Encoder 2A	18	Entrada digital + ISR
Encoder 2B	19	Entrada digital
MPU6050 SDA	4	I2C
MPU6050 SCL	2	I2C

Fonte: Autoria própria.

Figura D.1 – Sinal de controle convertido em PWM pelo ESP32.

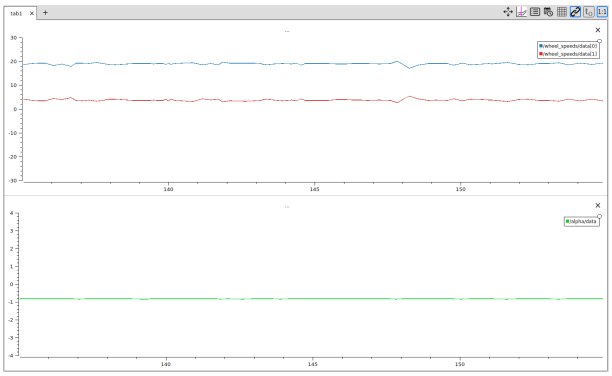


(a) Sinal roda direita.

(b) Sinal roda esquerda.

Fonte: Autoria própria.

Figura D.2 – Erro angular define o Sinal de controle que são convertidos em PWM.



Fonte: Autoria própria.

APÊNDICE E – TABELAS DE REFERENCIAÇÃO

As tabelas deste apêndice reúnem variáveis, parâmetros, tópicos e recursos utilizados na implementação, funcionando como material de consulta para a arquitetura descrita ao longo do trabalho.

Tabela E.1 – Funções do nó ROS do robô agrupadas por módulo.

Módulo / Função	**Descrição**
Inicialização	`__init__(self)`: Inicializa o nó ROS, define parâmetros físicos e de controle, cria variáveis internas, publishers e subscribers.
Callbacks	`callback_robo(self, msg)`: Recebe a pose do robô e atualiza `self.posicao_robo`. `callback_bola(self, msg)`: Recebe a posição da bola, atualiza `self.bola_pose` e estima a velocidade da bola. `callback_objetivo(self, msg)`: Armazena o objetivo final definido pelo usuário. `callback_campo(self, msg)`: Recebe os limites do campo desenhado e atualiza `self.campo_info`. `callback_robo_em_campo(self, msg)`: Atualiza o estado lógico que informa se o robô está dentro do campo.
Cálculos e Controle	`ajustar_angulo(self, ang)`: Ajusta qualquer ângulo para o intervalo $[-\pi, \pi]$. `calcular_ponto_interceptacao(self)`: Calcula o ponto de perseguição da bola usando predição e lookahead. `calcular_ponto_conducao(self)`: Calcula um ponto atrás da bola para condução em direção ao objetivo. `calcular_rho_alpha(self, alvo)`: Calcula a distância ρ e o erro angular α até o alvo. `controlador(self, rho, alpha)`: Aplica filtro no erro angular, PID angular e calcula as velocidades v e w. `diferencial(self, v, w)`: Converte v e w nas velocidades angulares da roda esquerda e da roda direita.
Gestão de Alvo	`limitar_alvo_dentro_campo(self, alvo)`: Garante que o alvo permaneça dentro do campo com margem de segurança. `calcular_repulsao_bordas(self)`: Calcula a força repulsiva gerada pelas bordas do campo. `aplicar_campos_potenciais(self, alvo, usar_objetivo_final=False)`: Combina força atrativa e repulsiva, gerando o alvo corrigido.
Diagnóstico	`publicar_diagnostico(...)`: Publica variáveis auxiliares para diagnóstico e visualização no ROS.
Estratégia e Execução	`estrategia(self)`: Implementa a lógica central de decisão, controle e atuação. `run(self)`: Executa continuamente a estratégia de controle no laço principal.

Fonte: Autoria própria.

E.1 Parâmetros físicos

Tabela E.2 – Parâmetros físicos do robô e seus significados.

Variável	Significado
`self.raio_roda`	Raio de cada roda do robô.
`self.distancia_rodas`	Distância entre as rodas do robô diferencial.
`self.rpm_motor`	Rotação nominal do motor em rpm.
`self.velocidade_maxima_motor`	Velocidade linear máxima teórica obtida a partir do raio da roda e da rotação do motor.
`self.max_wheel_speed`	Limite máximo de velocidade angular das rodas.

Fonte: Autoria própria.

E.2 Parâmetros de controle

Tabela E.3 – Parâmetros de controle do robô e seus significados.

Variável	Significado
`self.vel_linear_max`	Velocidade linear máxima permitida para o robô.
`self.omega_max`	Velocidade angular máxima permitida para o robô.
`self.kp`	Ganho proporcional do controlador PID angular.
`self.ki`	Ganho integral do controlador PID angular.
`self.kd`	Ganho derivativo do controlador PID angular.
`self.erro_integral`	Acúmulo temporal do erro angular.
`self.integral_max`	Limite máximo da parcela integral.
`self.alpha_old`	Valor anterior do erro angular.
`self.tempo_pid_old`	Instante anterior usado no cálculo da derivada do erro.

Fonte: Autoria própria.

E.3 Filtros e Suavizações

Tabela E.4 – Parâmetros de filtragem e suavização do erro angular.

Variável	Significado
`self.alpha_filtrado`	Erro angular após filtragem exponencial.
`self.filtro_alpha`	Coeficiente do filtro exponencial aplicado ao erro angular.
`self.deadzone_alpha`	Zona morta aplicada ao erro angular.

Fonte: Autoria própria.

E.4 Perseguição e condução

Tabela E.5 – Parâmetros de perseguição e condução do robô.

Variável	Significado
`self.lookahead_min`	Valor mínimo do lookahead.
`self.lookahead_max`	Valor máximo do lookahead.
`self.distancia_dominar_bola`	Distância máxima entre robô e bola para considerar domínio da bola.
`self.ganho_correcao_lateral`	Ganho aplicado à correção lateral quando o robô domina a bola.
`self.distancia_atras_bola`	Distância usada para posicionar o robô atrás da bola.
`self.offset_contorno`	Deslocamento lateral usado no ponto de condução.

Fonte: Autoria própria.

E.5 Variáveis de estado

Tabela E.6 – Variáveis de estado do robô e seus significados.

Variável	Significado
`self.posicao_robo`	Pose atual do robô recebida do ROS.
`self.bola_pose`	Posição atual da bola.
`self.bola_pose_old`	Posição anterior da bola.
`self.tempo_bola_old`	Instante anterior de leitura da posição da bola.
`self.vx_bola`	Velocidade estimada da bola no eixo x.
`self.vy_bola`	Velocidade estimada da bola no eixo y.
`self.objetivo`	Objetivo final definido pelo usuário.
`self.campo_info`	Vetor contendo informações do campo, como limites e centro.
`self.robo_em_campo`	Estado lógico indicando se o robô está dentro da área do campo.

Fonte: Autoria própria.

E.6 Campos potenciais

Tabela E.7 – Parâmetros de campos potenciais do robô e seus significados.

Variável	Significado
`self.raio_area_critica`	Raio da região crítica próxima às bordas.
`self.raio_area_repulsao`	Raio da região em que a repulsão das bordas começa a atuar.
`self.k_repulsao`	Ganho da força repulsiva na região de repulsão.
`self.k_repulsao_critica`	Ganho da força repulsiva na região crítica.
`self.epsilon_borda`	Valor mínimo usado para evitar divisão por zero no cálculo da repulsão.

Fonte: Autoria própria.

E.7 Tópicos ROS publicados

Tabela E.8 – Principais tópicos ROS publicados pelo sistema do robô.

Tópico	Conteúdo publicado
`/wheel_speeds`	Velocidades das rodas esquerda e direita.
`/alpha`	Erro angular utilizado no controlador.
`/trajetoria_robo`	Pose do robô para visualização da trajetória.
`/trajetoria_bola`	Posição da bola para visualização da trajetória.
`/ponto_interceptacao`	Alvo corrente utilizado na navegação.
`/campo/distancias_bordas_ctrl`	Distâncias entre o robô e as bordas do campo.
`/campo/forca_repulsiva`	Componentes x e y da força repulsiva.
`/campo/forca_total`	Componentes x e y da força resultante.
`/campo/em_repulsao_ctrl`	Estado lógico indicando entrada na área de repulsão.
`/campo/em_critica_ctrl`	Estado lógico indicando entrada na área crítica.
`/campo/vel_cmd`	Velocidade linear e angular calculadas pelo controlador.

Fonte: Autoria própria.

E.8 Ferramentas utilizadas no sistema

A Tabela E.9 apresenta as principais ferramentas e bibliotecas utilizadas no sistema desenvolvido.

Tabela E.9 – Ferramentas e bibliotecas utilizadas no sistema

Ferramenta / Biblioteca	**Onde aparece**	**Função no sistema**
ROS (`rospy`)	`import rospy`	Estrutura principal do sistema, responsável pela comunicação entre nós por meio de tópicos ROS.
OpenCV (`cv2`)	`import cv2`	Processamento visual e interface gráfica; desenha campo, círculos, vetores, textos, objetivo e forças na imagem.
NumPy (`numpy`)	`import numpy as np`	Operações numéricas, cálculo de distâncias, filtragem de posições e criação de estruturas matriciais.
CV Bridge	`from cv_bridge import CvBridge`	Conversão entre mensagens de imagem do ROS e imagens compatíveis com OpenCV.
`geometry_msgs/Point`	Mensagens ROS	Representação de posições do robô, bola, objetivo e ponto de interceptação.
`geometry_msgs/Vector3`	Mensagens ROS	Representação de vetores, como velocidade da bola e vetor distância.
`std_msgs/Bool`	Mensagens ROS	Representação de estados lógicos, como entrada em repulsão ou área crítica.
`std_msgs/Float32`	Mensagens ROS	Publicação de grandezas escalares, como erro angular e distância.
`std_msgs/Float32MultiArray`	Mensagens ROS	Publicação de vetores numéricos, como velocidades das rodas, forças e distâncias às bordas.
`sensor_msgs/Image`	Mensagens ROS	Recebimento das imagens da câmera.
Biblioteca `math`	`import math`	Funções trigonométricas e matemáticas para ângulos, cinemática, distâncias e controle.
`time`	`import time`	Medição de intervalos de tempo para estimativas de velocidade e derivada do PID.
`deepcopy`	`from copy import deepcopy`	Cópia segura da posição anterior da bola.
`collections.deque`	`import collections`	Armazenamento de históricos da bola, robô e distâncias.

Fonte: Autoria própria.

E.9 Recursos de controle utilizados na movimentação

Tabela E.10 – Recursos de controle e técnicas aplicadas na movimentação do robô.

Recurso / Técnica	Onde aparece	Função no sistema
PID angular	`controlador(self, rho, alpha)`	Controla a velocidade angular do robô a partir do erro angular.
Campos potenciais	`aplicar_campos_potenciais(...)`	Combina força atrativa e força repulsiva para gerar a direção resultante de movimento.
Força atrativa	`fx_att`, `fy_att`	Direciona o robô para o alvo corrente.
Força repulsiva	`calcular_repulsao_bordas()`	Afasta o robô das regiões de restrição consideradas pelo algoritmo.
Força resultante	`fx_total`, `fy_total`	Define a direção final de deslocamento do robô.
Lookahead	`calcular_ponto_interceptacao()`	Cria uma referência intermediária à frente na direção prevista da bola.
Ponto de condução	`calcular_ponto_conducao()`	Posiciona a referência do robô atrás da bola para conduzi-la até o objetivo.
Filtro exponencial	`self.alpha_filtrado`	Suaviza o erro angular, reduzindo ruídos e oscilações.
Zona morta angular	`self.deadzone_alpha`	Ignora erros angulares muito pequenos para evitar microcorreções.
Anti-windup	`self.integral_max`	Limita a parcela integral do PID para reduzir acúmulo excessivo.
Correção lateral	`erro_lateral`	Melhora a condução da bola quando o robô já está próximo dela.
Bloqueio de avanço	`if abs(alpha) > 1.05: v = 0`	Faz o robô girar antes de avançar quando está muito desalinhado.
Redução de velocidade por risco	`if em_repulsao`, `if em_critica`	Diminui a velocidade em regiões próximas às bordas.
Limitação do alvo no campo	`limitar_alvo_dentro_campo()`	Garante que o alvo de navegação permaneça dentro da área válida.
Cinemática diferencial	`diferencial(self, v, w)`	Converte velocidade linear e angular em velocidades das rodas.

Fonte: Autoria própria.

E.10 Ferramentas visuais utilizadas com OpenCV

Tabela E.11 – Elementos visuais do OpenCV e suas funções no sistema.

Elemento visual / Função	Função no sistema
`cv2.imshow`	Exibe a janela com a imagem processada.
`cv2.namedWindow`	Cria a janela gráfica.
`cv2.setMouseCallback`	Permite interação com o mouse para desenhar o campo e definir o objetivo.
`cv2.rectangle`	Desenha o campo e a grade.
`cv2.circle`	Desenha centro do campo, objetivo, previsão da bola e círculos de repulsão/crítica.
`cv2.arrowedLine`	Desenha orientação do robô, vetor distância, velocidade da bola e forças.
`cv2.putText`	Escreve textos e legendas na imagem.
`cv2.addWeighted`	Faz a fusão visual entre camadas de imagem.

Fonte: Autoria própria.

E.11 Nós e módulos observados no sistema ROS

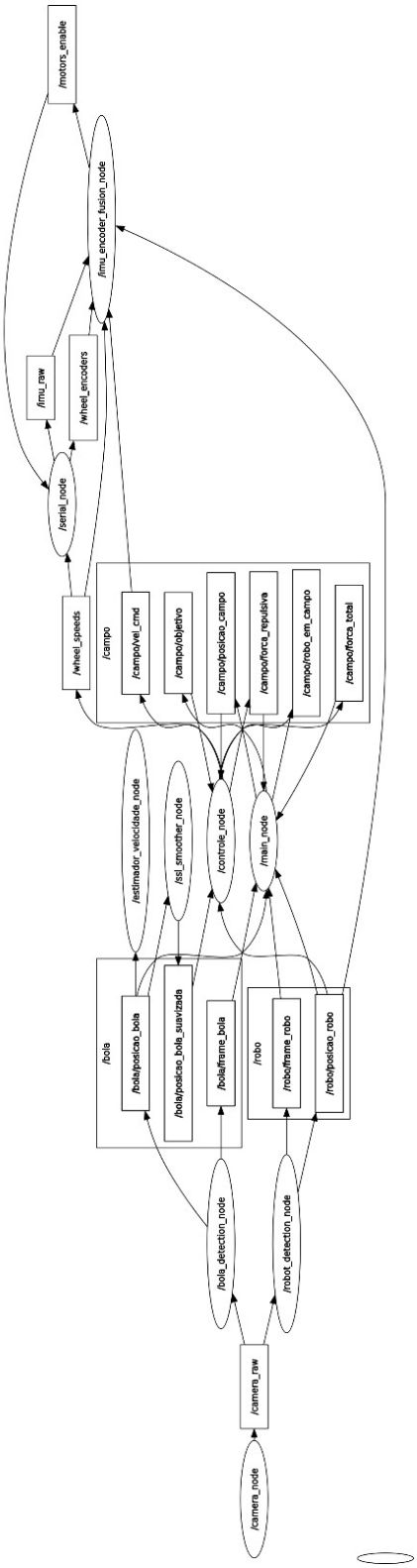
Tabela E.12 – Nós e módulos observados no sistema ROS e suas funções.

Módulo / Nó	Função no sistema
`/camera_node`	Aquisição da imagem da câmera.
`/bola_detection_node`	Detecção da bola.
`/robot_detection_node`	Detecção do robô.
`/ssl_smoother_node`	Suavização da posição da bola.
`/main_node`	Fusão das informações visuais e interação com o campo.
`/controle_node`	Controle principal da movimentação.
`/estimador_velocidade_node`	Estimativa de velocidade e/ou odometria.
`/serial_node`	Comunicação serial com o hardware.
`/comparador_sensores_unificado`	Comparação ou fusão entre sensores.
`/imu_raw`	Dados crus da IMU.
`/wheel_encoders`	Leitura dos encoders das rodas.
`/estimador/odometria`	Estimativa da odometria do robô.
`/estimador/comparacao`	Comparação entre estimativas e sensores.
`/motors_enable`	Sinal de habilitação dos motores.

Fonte: Autoria própria.

APÊNDICE F – ARQUITETURA ROS RQT GRAPH

Figura F.1 – Arquitetura ROS .



Fonte: Autoria própria.

APÊNDICE G – TESTES DE SIMULAÇÃO DO ROBÔ DIFERENCIAL NO MATLAB

Neste apêndice são apresentados os testes realizados em ambiente MATLAB para validação do modelo cinemático e das estratégias de movimentação aplicadas ao robô diferencial. As simulações foram desenvolvidas como uma etapa preliminar para verificar o comportamento matemático do robô móvel antes da implementação no sistema físico.

O modelo considerado representa uma plataforma diferencial composta por duas rodas motrizes independentes, cuja movimentação é determinada pela diferença entre as velocidades angulares das rodas esquerda e direita. A relação entre as velocidades das rodas e a velocidade do robô é dada por:

$$v = \frac{r}{2}(\omega_r + \omega_l) \tag{D.1}$$

$$\omega = \frac{r}{b}(\omega_r - \omega_l) \tag{D.2}$$

onde v representa a velocidade linear do robô, ω representa a velocidade angular, r corresponde ao raio das rodas, ω_r e ω_l são as velocidades angulares das rodas direita e esquerda, respectivamente, e b representa a distância entre as rodas.

A pose do robô é definida pelo vetor de estado:

$$P = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} \tag{D.3}$$

onde x e y representam a posição do robô no plano cartesiano e θ corresponde à orientação angular em relação ao sistema de referência global.

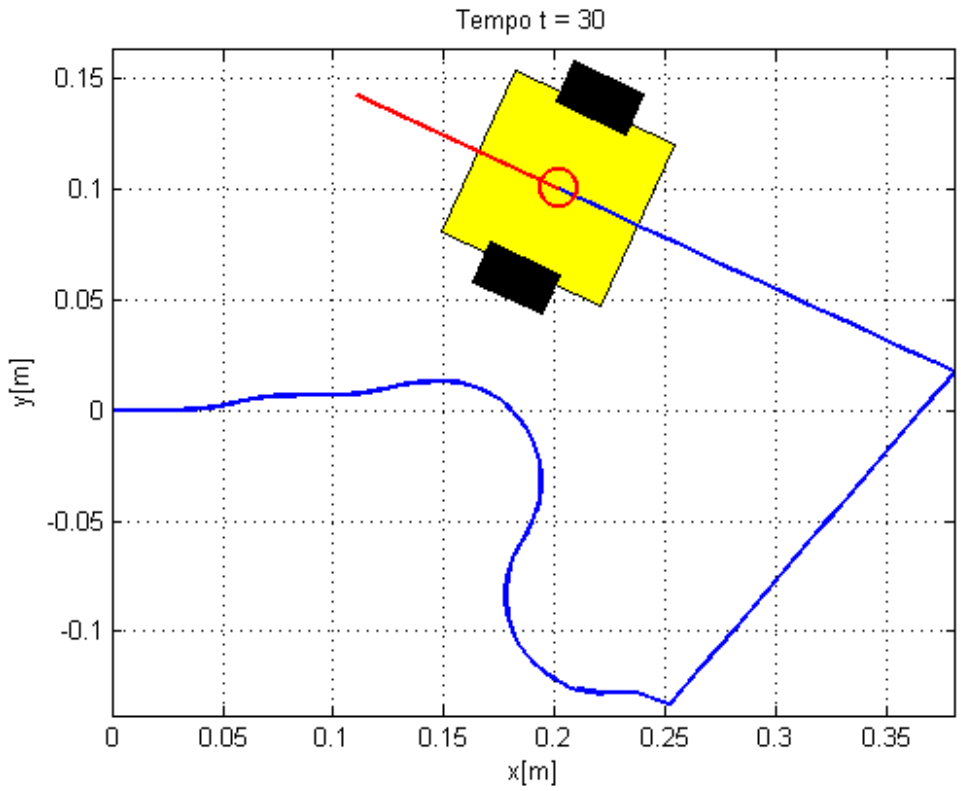
G.1 Teste 1: Validação do modelo cinemático direto

O primeiro teste teve como objetivo verificar a capacidade do modelo cinemático em reproduzir os movimentos característicos de um robô diferencial. Para isso, foram aplicadas diferentes combinações de velocidades angulares nas rodas esquerda e direita, permitindo observar deslocamentos em linha reta, curvas e rotações em torno do próprio eixo.

A atualização da pose foi realizada a partir das velocidades linear e angular obtidas pelo modelo cinemático, permitindo armazenar o histórico da trajetória percorrida pelo robô durante a simulação.

A Figura G.1 apresenta o resultado obtido. O corpo do robô é representado pelo elemento geométrico amarelo, as rodas motrizes pelas regiões pretas, o rastro azul indica a trajetória percorrida e o vetor vermelho representa a orientação instantânea do robô.

Figura G.1 – Simulação do modelo cinemático direto de um robô diferencial no plano cartesiano utilizando diferentes velocidades angulares nas rodas.



Fonte: Autoria própria.

A análise da trajetória obtida demonstra o comportamento esperado para uma plataforma diferencial. Quando as velocidades das rodas possuem o mesmo valor, o robô realiza deslocamento aproximadamente retilíneo. Já a diferença entre as velocidades dos atuadores provoca alteração da orientação, permitindo a execução de curvas. Além disso, velocidades de sentidos opostos possibilitam a rotação do robô em torno do próprio centro geométrico.

Esse teste permitiu validar a implementação da cinemática direta do robô diferencial, confirmando que o modelo matemático representa adequadamente os movimentos possíveis da plataforma móvel.

G.2 Teste 2: Navegação ponto a ponto utilizando controle proporcional

O segundo teste teve como objetivo avaliar uma estratégia simples de navegação ponto a ponto, utilizando como referência posições desejadas no plano cartesiano. Diferentemente do primeiro teste, no qual as velocidades das rodas eram definidas diretamente, neste experimento as velocidades linear e angular foram determinadas a partir dos erros de posição e orientação em relação ao ponto alvo.

O erro de posição foi calculado pela diferença entre a posição atual do robô e a posição desejada:

$$e_p = \sqrt{(x_d - x)^2 + (y_d - y)^2} \tag{D.4}$$

onde x_d e y_d representam as coordenadas do ponto desejado.

O erro angular foi obtido pela diferença entre a orientação desejada para o alvo e a orientação atual do robô:

$$e_\theta = \theta_d - \theta \tag{D.5}$$

A partir desses erros, foram aplicados controladores proporcionais para definir as velocidades:

$$v = K_v e_p \tag{D.6}$$

$$\omega = K_\omega e_\theta \tag{D.7}$$

Posteriormente, as velocidades linear e angular foram convertidas em velocidades angulares das rodas utilizando o modelo cinemático inverso:

$$\omega_r = \frac{2v + \omega b}{2r} \tag{D.8}$$

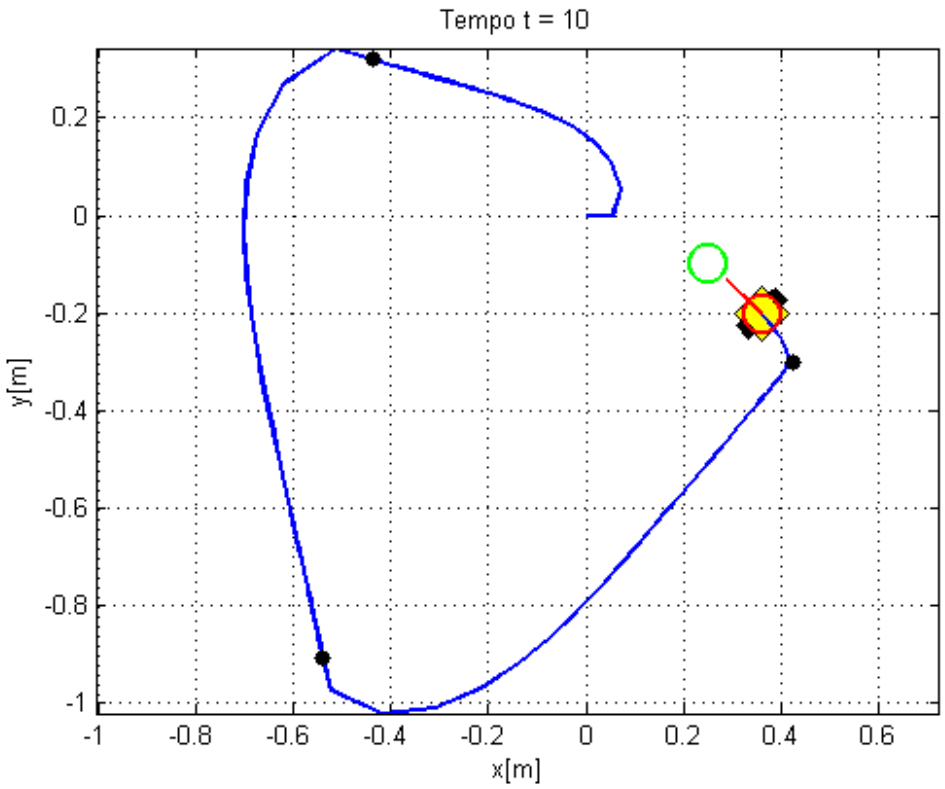
$$\omega_l = \frac{2v - \omega b}{2r} \tag{D.9}$$

Durante a simulação foram gerados diferentes pontos de destino no plano cartesiano, representando posições que poderiam ser ocupadas por um objetivo móvel. O robô calculou

continuamente o erro em relação ao ponto atual e realizou a movimentação até atingir uma região de aproximação definida.

A Figura G.2 apresenta o resultado da simulação de navegação ponto a ponto. O rastro azul representa a trajetória realizada pelo robô, o marcador vermelho indica a posição atual e os pontos verdes representam os objetivos gerados durante a simulação.

Figura G.2 – Simulação de navegação ponto a ponto de um robô diferencial utilizando controle proporcional de posição e orientação.



Fonte: Autoria própria.

O resultado obtido demonstra que o robô é capaz de ajustar sua orientação e deslocamento em função do erro existente entre sua posição atual e o ponto desejado. Dessa forma, a simulação validou a integração entre o modelo cinemático diferencial e uma estratégia básica de controle de movimentação.

G.3 Teste 3: Navegação utilizando campos potenciais artificiais

O terceiro teste teve como objetivo avaliar uma estratégia de navegação com desvio de obstáculos utilizando o método de campos potenciais artificiais. Diferentemente do teste anterior, no qual o robô realizava o deslocamento diretamente em direção aos pontos de destino,

nesta etapa foram inseridas restrições no ambiente para avaliar a capacidade de desvio durante a navegação.

O método de campos potenciais considera o ambiente como um campo de forças, no qual o objetivo exerce uma influência atrativa sobre o robô, enquanto os obstáculos geram forças repulsivas. A combinação dessas forças determina a direção de movimento do robô no plano cartesiano.

A força atrativa é calculada por:

$$F_{att} = K_{att}(G - P) \tag{D.10}$$

onde K_{att} representa a constante atrativa, G corresponde à posição do objetivo e P representa a posição atual do robô.

A força repulsiva é determinada pela distância entre o robô e os obstáculos presentes no ambiente:

$$F_{rep} = K_{rep}\left(\frac{1}{\epsilon}\left(\frac{1}{\epsilon_0} - \frac{1}{\epsilon}\right)\right) \tag{D.11}$$

onde K_{rep} representa a constante repulsiva, ϵ corresponde à distância entre o robô e o obstáculo e ϵ_0 representa a distância limite da região de influência do obstáculo.

A força resultante utilizada para movimentação do robô é obtida pela soma das componentes:

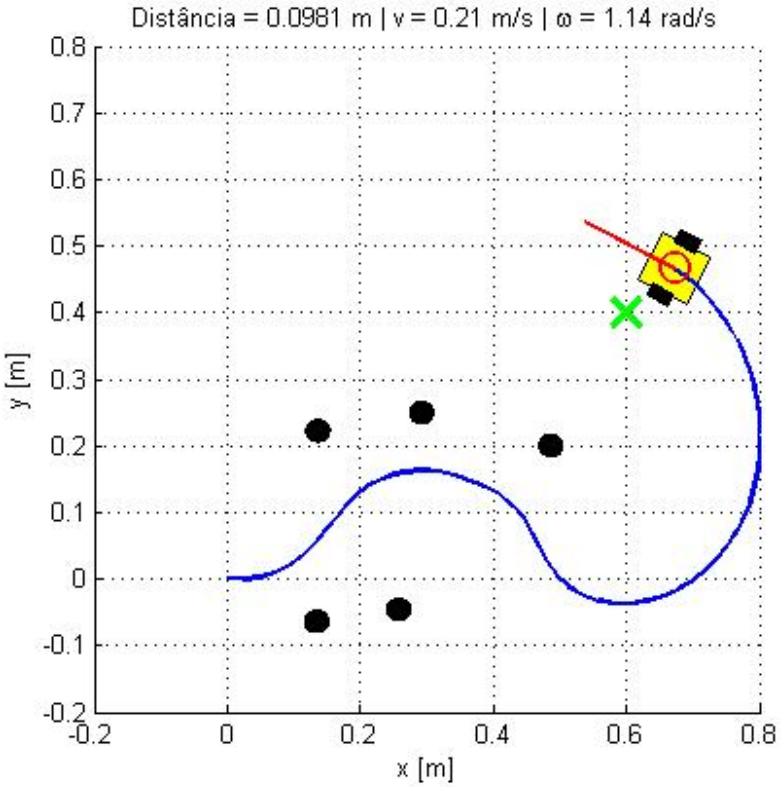
$$F_{tot} = F_{att} + F_{rep} \tag{D.12}$$

A direção da força resultante determina a orientação desejada do robô, enquanto o módulo da força é utilizado para definir a velocidade de deslocamento no modelo cinemático diferencial.

Na simulação foram inseridos obstáculos circulares representando regiões proibidas no ambiente de navegação. O robô deve alcançar o objetivo evitando colisões, realizando alterações na trajetória conforme a influência das forças repulsivas.

A Figura G.3 apresenta o resultado obtido. A trajetória percorrida pelo robô é representada pelo rastro azul, os obstáculos são indicados pelos círculos pretos e o objetivo final é representado pela marcação verde.

Figura G.3 – Simulação de navegação utilizando campos potenciais artificiais para desvio de obstáculos de um robô diferencial.



Fonte: Autoria própria.

O resultado demonstra que o robô é capaz de modificar sua trajetória conforme as restrições do ambiente, mantendo a tendência de deslocamento em direção ao objetivo devido à componente atrativa e evitando regiões próximas aos obstáculos devido à componente repulsiva.

Esse teste permitiu validar uma estratégia de navegação mais próxima das condições encontradas em aplicações reais, nas quais o robô deve realizar deslocamentos considerando obstáculos presentes no ambiente de operação.

G.3.1 Teste 3.1: Avaliação do comportamento em cenário com maior complexidade de navegação

Após a validação inicial da estratégia de campos potenciais artificiais, foi realizado um experimento adicional utilizando a mesma abordagem de navegação, porém com ajustes nos parâmetros do sistema e alteração da disposição dos obstáculos no ambiente.

O objetivo deste teste foi avaliar o comportamento do robô diferencial em uma situação de maior complexidade, exigindo a realização de manobras mais acentuadas e alterações de

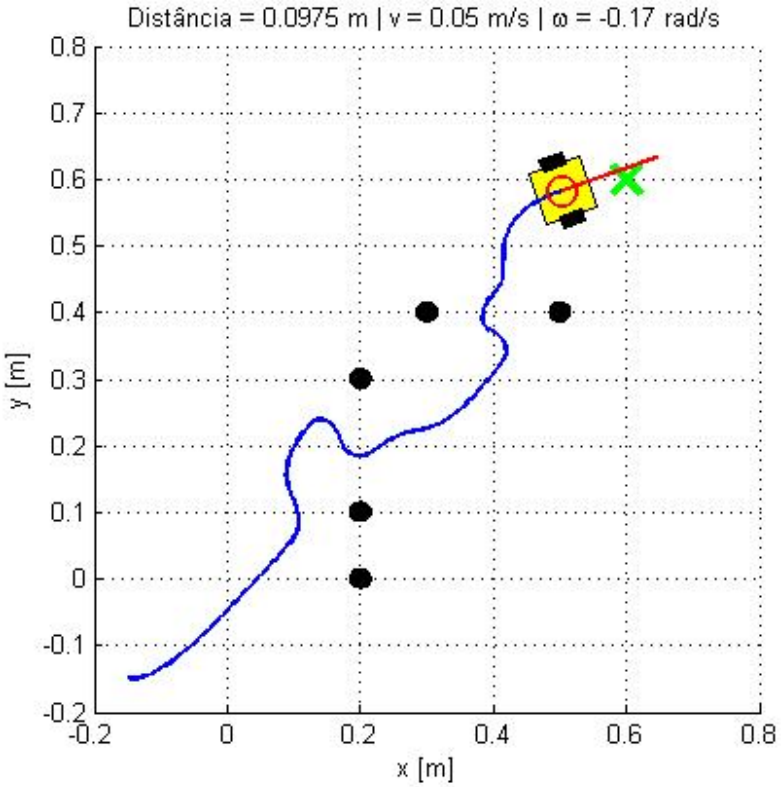
trajetória durante o deslocamento até o objetivo. Para isso, foram modificados alguns parâmetros responsáveis pela dinâmica da navegação, incluindo a velocidade máxima de deslocamento, a intensidade das forças atrativa e repulsiva e a região de influência dos obstáculos.

A redução da velocidade máxima permitiu uma movimentação mais controlada do robô durante as mudanças de direção, enquanto o ajuste da força atrativa modificou a tendência de aproximação ao objetivo. Já a alteração da força repulsiva possibilitou controlar a influência dos obstáculos sobre a trajetória, permitindo diferentes níveis de afastamento durante a navegação.

Além dos ajustes dos parâmetros do controlador, os obstáculos foram posicionados manualmente em uma configuração mais restritiva, criando uma trajetória com maior necessidade de desvio e interação entre as forças presentes no campo potencial. Dessa forma, foi possível observar o comportamento do algoritmo em uma situação mais próxima de um ambiente real de operação.

A Figura G.4 apresenta o resultado da simulação após o ajuste dos parâmetros. Observa-se que o robô realiza uma trajetória com mudanças de direção mais complexas, desviando dos obstáculos devido à atuação das forças repulsivas e mantendo a tendência de deslocamento em direção ao objetivo devido à componente atrativa.

Figura G.4 – Simulação de navegação com ajuste dos parâmetros do campo potencial para realização de manobras mais complexas.



Fonte: Autoria própria.

O resultado obtido demonstra que o ajuste dos parâmetros do campo potencial influencia diretamente o comportamento da navegação. A alteração do equilíbrio entre as forças atrativa e repulsiva permitiu modificar a resposta do robô diante das restrições do ambiente, possibilitando a realização de manobras mais complexas sem comprometer a convergência para o objetivo final.

Esse experimento complementar reforça a capacidade de adaptação da estratégia de campos potenciais para diferentes configurações de ambiente, demonstrando sua aplicação como uma alternativa para navegação de robôs diferenciais em cenários com obstáculos.

Os testes apresentados neste apêndice forneceram uma validação progressiva dos modelos matemáticos e estratégias de controle empregados, iniciando pela validação da cinemática diferencial, passando pelo controle de posição e finalizando com uma estratégia de navegação com desvio de obstáculos baseada em campos potenciais artificiais.

APÊNDICE H – VÍDEOS COMPLEMENTARES DOS EXPERIMENTOS

Este apêndice apresenta os registros em vídeo dos experimentos realizados durante o desenvolvimento do robô da categoria IEEE VSSS. Os vídeos têm como objetivo complementar os resultados apresentados ao longo deste trabalho, permitindo visualizar o comportamento do sistema desenvolvido durante as etapas de validação dos módulos de controle, navegação e visão computacional.

Os registros audiovisuais apresentam desde a demonstração geral do projeto até testes específicos dos algoritmos desenvolvidos, como calibração do sistema de visão baseado em cores HSV e validação do controlador PID aplicado à movimentação do robô.

H.1 Apresentação geral do trabalho

O primeiro vídeo apresenta uma visão geral do trabalho desenvolvido, denominado "Sistemas de Controle e Navegação para Robôs da Categoria IEEE VSSS". O registro demonstra o funcionamento do robô durante a operação e permite observar a integração entre os módulos desenvolvidos, incluindo controle, navegação e percepção visual.

Durante este experimento, o robô foi confinado em uma região reduzida do campo de testes, sem utilizar as limitações físicas das bordas do campo como referência para sua movimentação. Essa abordagem foi adotada para validar o comportamento da estratégia de navegação baseada em campos potenciais, permitindo avaliar a atuação das forças atrativas em direção ao objetivo e das forças repulsivas utilizadas para evitar aproximações indesejadas durante o deslocamento.

Dessa forma, o experimento demonstra que a movimentação do robô é determinada pelos campos de influência calculados pelo algoritmo de navegação, e não por restrições físicas impostas pelo ambiente.

O vídeo referente a esta demonstração está disponível no endereço:

<https://www.youtube.com/watch?v=r1D36Togc30>



Figura H.1 – QR Code para acesso ao vídeo do teste de navegação utilizando campos potenciais.

Fonte: Autoria própria.

H.2 Explicação visual do sistema desenvolvido

O segundo vídeo apresenta uma explicação visual dos principais elementos que compõem o trabalho. São apresentados aspectos relacionados à estrutura do robô, funcionamento dos módulos desenvolvidos e organização geral do sistema proposto.

O vídeo referente a esta explicação está disponível no endereço:

<https://www.youtube.com/watch?v=U7yODDCIOZI>



Figura H.2 – QR Code para acesso ao vídeo explicação visual do sistema desenvolvido.

Fonte: Autoria própria.

H.3 Calibração do sistema de visão computacional

O terceiro vídeo apresenta o processo de calibração das cores utilizadas pelo sistema de visão computacional. Durante o procedimento, são ajustados os parâmetros do espaço de cores HSV, permitindo a identificação adequada dos elementos utilizados pelo sistema de percepção.

Essa etapa é fundamental para o funcionamento dos módulos de detecção, pois possibilita a segmentação dos objetos a partir das características de cor presentes na imagem capturada.

O vídeo referente à calibração está disponível no endereço:

<https://www.youtube.com/watch?v=an875IqZnXw>

Figura H.3 – QR Code para acesso ao vídeo Calibração do sistema de visão computacional.

Fonte: Autoria própria.

H.4 Teste do controlador PID para perseguição da bola

O quarto vídeo apresenta um teste experimental utilizando somente o controlador PID para movimentação do robô em direção à bola. Neste ensaio, não é utilizada a estratégia baseada em campos potenciais, permitindo observar isoladamente o comportamento do controlador responsável pelo ajuste da movimentação em função do erro existente entre o robô e o alvo.

O registro demonstra a resposta do sistema de controle durante a tarefa de aproximação da bola.

O vídeo referente a este teste está disponível no endereço:

<https://www.youtube.com/shorts/s4E9pZ5ZEbY>

Figura H.4 – QR Code para acesso ao vídeo Teste do controlador PID para perseguição da bola.

Fonte: Autoria própria.

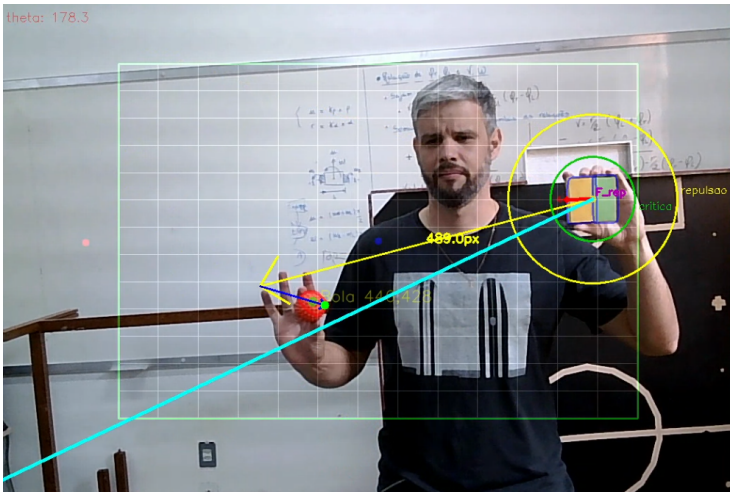
O software desenvolvido durante este trabalho encontra-se disponível em repositório público[1].

[1] GitHub <https://github.com/Elian-Souza>

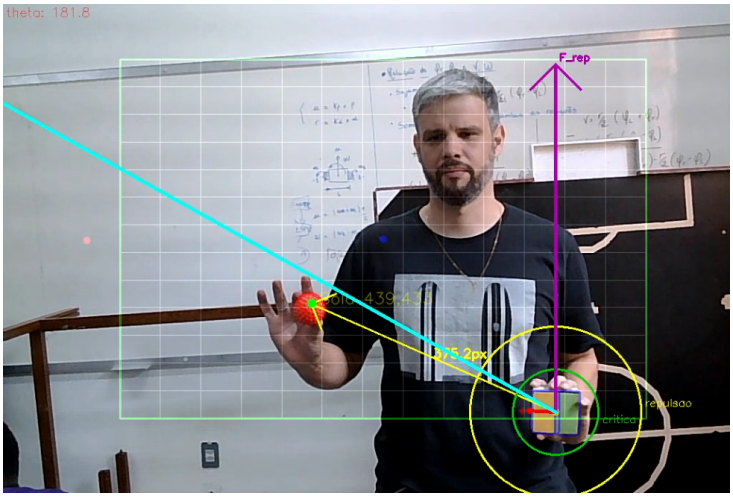
H.5 Considerações finais

Os vídeos apresentados neste apêndice complementam os resultados obtidos nos experimentos práticos, permitindo observar visualmente a integração entre os módulos de visão computacional, controle e navegação desenvolvidos. Dessa forma, os registros visuais Figura H.5 auxiliam na validação experimental do sistema proposto e na análise do comportamento real do robô diferencial em ambiente de testes.

Figura H.5 – Resultado do processamento de imagem realizado, com representação visual das informações extraídas para percepção e navegação do robô.



(a) Mapeamento do movimento da bola.



(b) Representação da força repulciva.

Fonte: Autoria própria.