

TRABALHO DE GRADUAÇÃO

**CONTROLE DE MOVIMENTO
DE UM ROBÔ MÓVEL NÃO-HOLONÔMICO
COM TRAÇÃO DIFERENCIAL**

Letícia Helena Silva Porto

Brasília, julho de 2017



**ENGENHARIA
MECATRÔNICA**
UNIVERSIDADE DE BRASÍLIA

UNIVERSIDADE DE BRASÍLIA
Faculdade de Tecnologia
Curso de Graduação em Engenharia de Controle e Automação

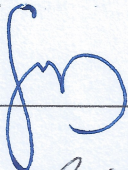
TRABALHO DE GRADUAÇÃO
CONTROLE DE MOVIMENTO
DE UM ROBÔ MÓVEL NÃO-HOLONÔMICO
COM TRAÇÃO DIFERENCIAL

Letícia Helena Silva Porto

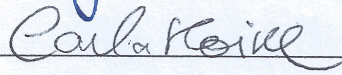
*Relatório submetido como requisito parcial de obtenção
de grau de Engenheiro de Controle e Automação*

Banca Examinadora

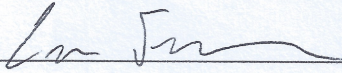
Professor Geovany Araújo Borges, ENE/UnB
Orientador



Professora Carla M. C. e C. Koike, CIC/UnB
Examinador interno



Prof. Luis Felipe Cruz Figueredo, ENE/UnB
Examinador interno



Brasília, julho de 2017

FICHA CATALOGRÁFICA

PORTO, LETÍCIA HELENA SILVA

Controle de Movimento de um Robô Móvel Não-Holonômico com Tração Diferencial,

[Distrito Federal] 2017.

xiii, 70p., 297 mm (FT/UnB, Engenheiro, Controle e Automação, 2017). Trabalho de Graduação – Universidade de Brasília. Faculdade de Tecnologia.

1. Controle de Movimento

2. Robótica

3. Robô móvel não-holonômico diferencial

I. Mecatrônica/FT/UnB

II. Título (Série)

REFERÊNCIA BIBLIOGRÁFICA

PORTO, LETÍCIA HELENA SILVA, (2017) Controle de Movimento de um Robô Móvel Não-Holonômico com Tração Diferencial. Trabalho de Graduação em Engenharia de Controle e Automação, Publicação FT. TG-*n*°008, Faculdade de Tecnologia, Universidade de Brasília, Brasília, DF, 70p.

CESSÃO DE DIREITOS

AUTOR: Leticia Helena Silva Porto

TÍTULO DO TRABALHO DE GRADUAÇÃO: Controle de Movimento de um Robô Móvel Não-Holonômico com Tração Diferencial.

GRAU: Engenheiro

ANO: 2017

É concedida à Universidade de Brasília permissão para reproduzir cópias deste Trabalho de Graduação e para emprestar ou vender tais cópias somente para propósitos acadêmicos e científicos. O autor reserva outros direitos de publicação e nenhuma parte desse Trabalho de Graduação pode ser reproduzida sem autorização por escrito do autor.

Leticia Helena Silva Porto

Campus Darcy Ribeiro, SG-11, Universidade de Brasília.

70919-970 Brasília – DF – Brasil.

Dedicatória

Aos meus pais, Francisco e Janete, e à equipe DROID

Leticia Helena Silva Porto

Agradecimentos

Primeiramente, gostaria de agradecer ao meu orientador, professor Geovany Borges, por toda a sua ajuda e conhecimento, que foram essenciais para o desenvolvimento deste trabalho. Gostaria de agradecer também a todos os amigos que fiz durante o curso, especialmente aos que conheci na equipe DROID, Artur, Aninha, Abdullah, Bamidele, Bira, Bruno, Daniel, Felipe, Gabriel, Giordano, Letícia Ploeg, Marcela, Natalia, Pedro, Rod, Sara, Schiavini, Teo, muito obrigada! Participar dessa equipe com vocês foi a melhor parte da minha graduação, muito obrigada por ter me permitido vivenciar todos esses momentos com vocês e por ter permitido que eu aprendesse tanto com excelentes pessoas. Um agradecimento especial ao Johnny e ao Crepe, que, além de fazerem parte dessa equipe maravilhosa, me acompanharam desde o meu ingresso na universidade e se tornaram grandes amigos. Agradeço aos meus amigos que contribuíram para o desenvolvimento deste projeto, muito obrigada Luan e Thiago, por toda a dedicação de vocês para a construção do robô e pela paciência com as crises de teimosia do Bruce. Agradeço especialmente aos amigos que trabalharam na arquitetura junto comigo: Camila, Ricardo e Rodrigo. Obrigada também a DROID, que foi a fonte de inspiração para a realização deste trabalho e pela oportunidade de ter participado da competição Robomagellan. Muito obrigada a todos vocês pelo apoio nos momentos de estresse, por compartilhar todo o frio que o Bruce nos fez passar e pelo excelente trabalho! Sem vocês e a DROID este trabalho não teria sido possível. Agradeço especialmente à Camila, pelo ótimo trabalho neste projeto, pela sua ajuda em todas as adversidades que passamos juntas durante o desenvolvimento do Bruce e por ter se tornado uma grande amiga. Muito obrigada! Rodrigo, muito obrigada por ter sido meu companheiro durante praticamente toda a minha graduação, tanto como amigo, quanto como namorado. Obrigada por sempre me motivar a seguir em frente, me apoiar nos meus momentos de crise, por ter vibrado sempre com as minhas conquistas e por ser a minha inspiração para tentar sempre ser uma pessoa melhor. Muito obrigada também por ter feito parte desse trabalho, por todo seu esforço e dedicação. Sem você, participar deste projeto e dos tantos outros que fizemos juntos não teria sido a mesma coisa. Te amo muito! Por fim, agradeço à minha família. Obrigada aos meus pais, Francisco e Janete, por todo o trabalho duro que vocês tiveram para que eu pudesse chegar até aqui, por terem sido sempre meu porto seguro e me dado apoio para vencer todas as dificuldades. Agradeço a minha irmã Ana Júlia, por ter sido sempre a minha fonte de inspiração de força e com quem eu sempre compartilhei as minhas vivências. Obrigada aos meus avós, tios e tias, por acreditarem em mim e por fazerem eu me sentir sempre capaz de seguir em frente. Não poderia deixar de agradecer especialmente a minha tia Aparecida, por ter me apoiado durante toda a minha graduação, sem você este trabalho não teria sido possível. Amo muito todos vocês.

Letícia Helena Silva Porto

RESUMO

Este trabalho apresenta o desenvolvimento de um controlador de movimento para um robô móvel não-holonômico com tração diferencial, construído com o objetivo de participar da competição *Robomagellan*. O controlador desenvolvido é composto por quatro malhas de controle, o controle e rastreamento de trajetórias, que é baseado no modelo cinemático do robô e é encarregado de fazer com que o robô gere e execute segmentos de caminho entre pontos determinados; o controlador de desvio de obstáculos, implementado com base na metodologia das zonas virtuais deformáveis para evitar colisões durante a movimentação do robô; o controle de aproximação, que torna o robô capaz de se aproximar de objetos de interesse a partir da identificação destes por técnicas de visão computacional; e o controlador de velocidades, que, a partir de um modelo de referência, adapta os seus parâmetros para manter os sistemas de propulsão nas velocidades desejadas. Os resultados obtidos demonstram que o sistema desenvolvido é capaz de controlar a movimentação do robô com êxito na execução de percursos similares aos da competição e na aproximação de cones de sinalização de trânsito laranjas.

Palavras Chave: robótica móvel, controlador adaptativo, controlador de velocidade, controlador de trajetória, controlador de desvio de obstáculos

ABSTRACT

This work presents the development of a motion controller of a non-holonomic mobile robot with differential traction. This robot was built with the objective of participating in the Robomagellan competition. The developed controller consists of four control loops. The path control is based on the kinematic model of the robot and is in charge of the generation and execution of path segments between determined points. The obstacles avoidance controller, that is implemented based on the methodology of the deformable virtual zones to avoid collisions during robot movement. The approach control, which makes the robot able to approach objects of interest, these objects are identified with computer vision techniques. And at lower level the speed controller, which, from a reference model, adapts its parameters to keep the propulsion systems at the desired speeds. The obtained results show that the developed system is able to control the movement of the robot with success in the execution of routes similar to the one of the competition and in the approximation of orange traffic cones.

Keywords: mobile robotic, adaptive control, speed control, path control, obstacle avoidance control

SUMÁRIO

1	INTRODUÇÃO	1
1.1	CONTEXTUALIZAÇÃO E MOTIVAÇÃO	1
1.2	DEFINIÇÃO DO PROBLEMA	3
1.2.1	ARQUITETURA DO ROBÔ	4
1.3	OBJETIVOS DO PROJETO	5
1.4	RESULTADOS PRINCIPAIS OBTIDOS	6
1.5	APRESENTAÇÃO DO MANUSCRITO	6
2	FUNDAMENTAÇÃO TEÓRICA	7
2.1	INTRODUÇÃO	7
2.2	REVISÃO BIBLIOGRÁFICA	7
2.3	CONTROLADORES DE VELOCIDADE	9
2.3.1	CONTROLADOR PROPORCIONAL INTEGRAL DERIVATIVO	9
2.3.2	CONTROLADOR ADAPTATIVO POR MODELO DE REFERÊNCIA	10
2.4	CONTROLE E RASTREAMENTO DE TRAJETÓRIA	13
2.4.1	SEGMENTOS DE CAMINHO	14
2.4.2	PLANEJADOR DE MOVIMENTO	14
2.4.3	CONTROLADOR DE TRAJETÓRIA POR MODELO DE REFERÊNCIA	16
2.5	DESVIO DE OBSTÁCULOS	19
2.5.1	MÉTODOS REATIVOS	20
2.5.2	MÉTODOS DELIBERATIVOS	20
2.5.3	MÉTODOS HÍBRIDOS	20
2.5.4	ZONA VIRTUAL DEFORMÁVEL	21
3	ESTRUTURA DO CONTROLADOR DE MOVIMENTO	24
3.1	INTRODUÇÃO	24
3.2	MODELAGEM	25
3.3	PROJETO DO CONTROLADOR DE VELOCIDADE	27
3.3.1	CÁLCULO DAS VELOCIDADES ANGULARES DAS RODAS	27
3.3.2	IDENTIFICAÇÃO DO SISTEMA	28
3.3.3	CONTROLADOR ADAPTATIVO POR MODELO DE REFERÊNCIA	29
3.4	PROJETO DO CONTROLADOR DE TRAJETÓRIA	30
3.4.1	PLANEJADOR DE MOVIMENTO	30

3.4.2	CONTROLADOR DE TRAJETÓRIA POR MODELO DE REFERÊNCIA	33
3.5	MODELO DA ZONA VIRTUAL DEFORMÁVEL (ZVD)	37
3.5.1	SENSORES ULTRASSÔNICOS	39
3.5.2	DEFORMAÇÃO DA ZONA VIRTUAL DEFORMÁVEL	40
3.5.3	AÇÃO DE CONTROLE PARA EVITAR O OBSTÁCULO	42
3.6	APROXIMAÇÃO DO PONTO DE INTERESSE.....	43
3.7	O CONTROLADOR DE MOVIMENTO	45
4	RESULTADOS.....	48
4.1	INTRODUÇÃO	48
4.2	RESULTADOS EM SIMULAÇÃO.....	48
4.2.1	O AMBIENTE DE SIMULAÇÃO.....	48
4.2.2	EXECUÇÃO DE UMA TRAJETÓRIA COM CORREÇÃO DA ORIENTAÇÃO FINAL	49
4.2.3	EXECUÇÃO DE UMA TRAJETÓRIA COM O PLANEJADOR DE ROTAS	51
4.3	RESULTADOS NO ROBÔ REAL.....	53
4.3.1	RESULTADOS DO CONTROLADOR DE VELOCIDADE MRAC.....	53
4.3.2	EXECUÇÃO DE UMA TRAJETÓRIA	56
4.3.3	EXECUÇÃO DE UMA TRAJETÓRIA COM DESVIO DE OBSTÁCULOS.....	61
4.3.4	TESTE DE APROXIMAÇÃO DO CONE.....	62
5	CONCLUSÕES	64
	REFERÊNCIAS BIBLIOGRÁFICAS	66
	ANEXOS.....	69
I	DESCRIÇÃO DO CONTEÚDO DO CD	70

LISTA DE FIGURAS

1.1	Robô Atlas durante a competição ¹	2
1.2	<i>Curiosity Rover</i> ²	3
1.3	Robô desenvolvido para o projeto ³	4
1.4	Diagrama da arquitetura do robô.....	6
2.1	Diagrama de blocos de um controlador PID.....	10
2.2	Diagrama de blocos de um controlador MRAC.....	11
2.3	Modelo de uma trajetória baseada em segmentos de caminho.....	15
2.4	Segmento de caminho em forma de segmento de reta.....	15
2.5	Segmento de caminho em forma de arco de círculo.....	15
2.6	Trajeto em forma de segmento de reta.....	16
2.7	Modelo de um robô diferencial no sistema de coordenadas inercial.....	18
2.8	Representação de uma deformação na ZVD.....	23
3.1	Diagrama do controlador de movimento.....	25
3.2	Modelo de um robô diferencial.....	26
3.3	Diagrama de blocos do controlador de velocidade MRAC.....	31
3.4	Diagrama do Controle e Rastreamento de Trajetória.....	32
3.5	Representação de um segmento de caminho.....	32
3.6	Modelo de referência e modelo do robô real.....	35
3.7	Segmento intermediário para a correção da orientação.....	38
3.8	Representação da Zona Virtual Deformável.....	38
3.9	Localização dos sensores ultrassônicos no robô.....	40
3.10	Sensor ultrassônico utilizado ⁴	40
3.11	Deteção de um obstáculo pelos sensores ultrassônicos.....	41
3.12	Modelo do robô real.....	41
3.13	Robô identificando o cone na fase de aproximação.....	44
3.14	Diagrama do controlador de aproximação.....	45
4.1	Robô utilizado para os testes em simulação.....	49
4.2	Trajetória executada pelo robô simulado em relação ao robô virtual com correção da orientação final.....	49
4.3	Erros de posição e orientação entre o robô simulado e o robô virtual.....	50
4.4	Orientação do robô simulado e do robô virtual.....	51

4.5	Ambiente simulado no Gazebo.	52
4.6	Mapa de custo do ambiente de teste em simulação.	52
4.7	Trajétórias executadas pelo robô simulado e pelo robô virtual a partir dos pontos do planejador de rotas.	53
4.8	Posição e orientação do robô simulado em relação ao robô virtual.	54
4.9	Erros de posição e orientação entre o robô simulado e o robô virtual.	55
4.10	Velocidades das rodas e as velocidades de referência.	56
4.11	Erros entre as velocidades das rodas e as velocidades de referência.	57
4.12	Mapa de custo do ambiente de teste com o robô real.	58
4.13	Trajétórias executadas pelo robô real e pelo robô virtual.	58
4.14	Erros de posição e orientação entre o robô real e o robô virtual.	59
4.15	Velocidades calculadas pelo controlador de trajetória e as velocidades atuais do robô.	60
4.16	Trajétória executada pelo robô real em relação a referência, com desvio de obstáculos.	61
4.17	Robô aproximando-se do cone.	62
4.18	Posição e orientação do cone em relação ao robô real.	63
4.19	Velocidades calculadas pelo controlador de aproximação.	63

LISTA DE SÍMBOLOS

Símbolos Latinos

U	Tensão do motor, no domínio de Laplace	[V]
u	Tensão do motor	[V]
e	Erro do controlador	
E	Erro do controlador, no domínio de Laplace	
K_p	Constante de ganho proporcional do controlador	
K_d	Constante de ganho derivativo do controlador	
K_i	Constante de ganho integral do controlador	
P	Pontos bidimensionais que indicam a rota a ser realizada	
x	Localização na abcissa do eixo de coordenadas inercial	
y	Localização na ordenada do eixo de coordenadas inercial	
r	Raio da roda	[m]
b	Medida da metade da largura do robô	[m]
X	Abcissa do eixo de coordenadas inercial	
Y	Ordenada do eixo de coordenadas inercial	
\mathbf{R}	Matriz de transformação de coordenadas	
\mathbf{p}	Vetor de posições do robô	
\mathbf{q}	Vetor de velocidades do robô	
\mathbf{s}	Vetor da Zona Virtual Deformável não deformada	
s_i	Componente da Zona Virtual Deformável	
\mathbf{s}_D	Vetor da Zona Virtual Deformável deformada	
\mathbf{i}	Vetor de intrusão da Zona Virtual Deformável	
v	Velocidade linear do robô	[m/s]
v_t	Velocidade linear do robô, calculada pelo controlador de trajetória	[m/s]
v_z	Velocidade linear do robô, calculada pelo controlador de desvio de obstáculos	[m/s]
v_c	Velocidade linear do robô, calculada pelo controlador de aproximação	[m/s]
d_c	Distância do cone em relação ao robô	[m]
\mathbf{d}	Vetor com a leitura dos sensores ultrassônicos em relação ao centro do robô	

Símbolos Gregos

ω	Velocidade angular do robô	[rad/s]
ω_e	Velocidade de rotação da roda esquerda	[rad/s]
ω_d	Velocidade de rotação da roda direita	[rad/s]
ω_t	Velocidade angular do robô, calculada pelo controlador de trajetória	[rad/s]
ω_z	Velocidade angular do robô, calculada pelo controlador de desvio de obstáculos	[rad/s]
ω_c	Velocidade angular do robô, calculada pelo controlador de aproximação	[rad/s]
Ω	Velocidade angular das rodas do robô, no domínio de Laplace	[rad/s]
γ	Ganho de adaptação do controlador MRAC	
Δ	Vetor de deformação da Zona Virtual Deformável	
δ	Constante positiva	
θ	Orientação do robô	[rad]
θ_c	Orientação do cone em relação ao robô	[rad]
θ_f	Orientação final desejada para o robô	[rad]
θ_1	Parâmetro de adaptação do controlador MRAC	
θ_2	Parâmetro de adaptação do controlador MRAC	
ρ	Vetor com as leituras dos sensores ultrassônicos	
λ	Segmento de caminho	
ϕ	Orientação do segmento de caminho	[°]
τ	Constante de tempo	[s]
φ	Ângulo de alcance de leitura dos sensores ultrassônicos	[°]
ψ	Ângulo entre os sensores ultrassônicos em relação ao arco	[°]
σ	Ângulo entre os sensores ultrassônicos em relação ao centro do robô	[°]

Grupos Adimensionais

i, j	Contador
--------	----------

Subscritos

r	referência
m	modelo de referência
e	esquerdo
d	direito

Sobrescritos

- Variação temporal
- * Entrada de referência para o controlador

Siglas

GPS	Sistema de Posicionamento Global - <i>Global Positioning System</i>
DROID	Divisão de Robótica Inteligente
NED	<i>North East Down</i>
IMU	Unidade de Medida Inercial - <i>Inertial Measurement Unit</i>
PID	Proporcional Integral Derivativo
ZVD	Zona Virtual Deformável
MRAC	Controlador Adaptativo por Modelo de Referência - <i>Model Reference Adaptativ Control</i>
PWM	Modulação por Largura de Pulso - <i>Pulse Width Modulation</i>
ROS	Sistema Operacional de Robôs - <i>Robot Operating System</i>

Capítulo 1

Introdução

1.1 Contextualização e motivação

A robótica é um ramo de estudo amplo que apresentou um grande salto em seu desenvolvimento nas últimas décadas. Uma de suas vertentes é a robótica móvel que, por definição, é a área da robótica que agrupa os robôs que não possuem base fixa, ou seja, que podem se locomover dentro de um espaço limitado ou não [1]. Essa locomoção pode ocorrer via terrestre, aérea ou aquática.

As aplicações da robótica móvel são muito variadas e, por vezes, relacionadas a algumas tarefas que podem ser nocivas à saúde humana, como, por exemplo, o resgate de vítimas de acidentes, em áreas como a agricultura, no transporte de cargas perigosas ou em tarefas de exploração solitárias ou cooperativas junto a outros veículos não tripulados. Um exemplo de aplicação de robôs móveis é exibido na Figura 1.1, que mostra o robô Atlas, desenvolvido pela empresa *Boston Dynamics*, sendo utilizado em uma competição, organizada pelo Departamento de Defesa dos Estados Unidos, com o intuito de identificar o humanoide mais apto a atuar no resgate de seres humanos.

Dessa forma, observa-se que uma característica desejada para esses robôs é que estes sejam autônomos, ou seja, capazes de navegar em um ambiente não-estruturado sem que haja uma intervenção humana [2]. Entretanto, os robôs podem ser autônomos de diferentes formas. Um alto nível de autonomia é desejado em campos como a exploração espacial, na qual a comunicação possui atrasos e as interrupções são inevitáveis. Um exemplo pode ser visualizado na Figura 1.2, que exhibe o robô *Curiosity*. Este robô foi construído pela NASA e possuía uma certa autonomia de navegação, pois era destinado a explorar a superfície de Marte¹.

Para o caso específico de robô móveis com rodas, deve-se considerar que estes podem apresentar restrições ao seu movimento, como aquelas devido ao escorregamento das rodas. As restrições que fazem com que a dimensão do espaço das velocidades seja menor que a dimensão do espaço de configuração do robô são chamadas de restrições não-holonômicas [3]. Os automóveis são um exemplo típico desse tipo de restrição, já que não são capazes de se movimentar lateralmente. Dessa forma, precisam fazer uma manobra para estacionar, pois seus movimentos são limitados a

¹Fonte: <https://www.jpl.nasa.gov/news/news.php?release=2013-259>

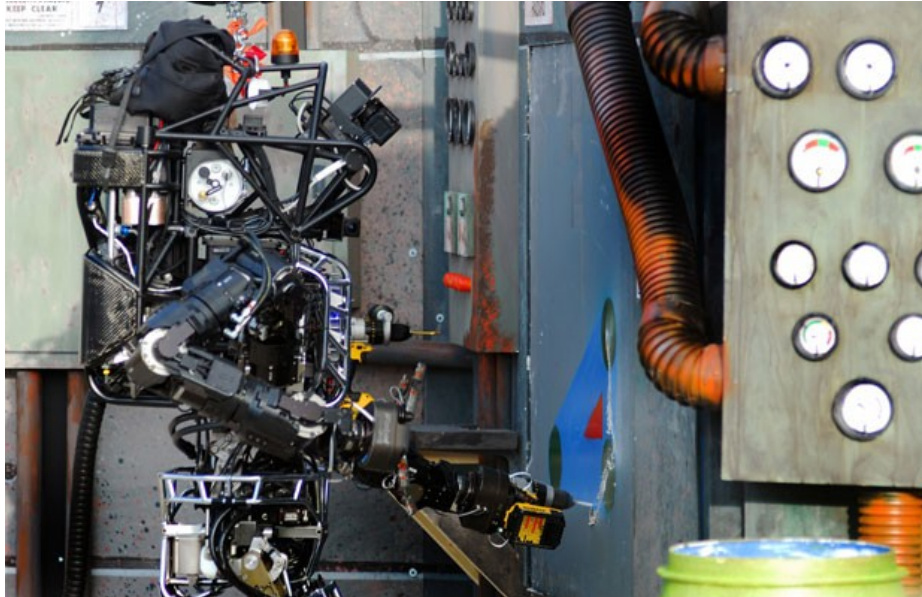


Figura 1.1: Robô Atlas durante a competição².

serem colineares com a direção das rodas.

O desejo de atribuir uma certa “inteligência” a esses robôs, para que estes resolvam situações adversas de forma autônoma, e as possíveis restrições existentes na movimentação destes tornam desafiador o desenvolvimento de estratégias de controle para a navegação autônoma de robôs móveis com rodas em ambientes não-estruturados ou controlados.

Este desafio da robótica móvel tem motivado estudos, conferências e torneios na área de navegação autônoma. Esses torneios, geralmente, visam reunir entusiastas da robótica com o objetivo de encontrar qual deles executa da melhor forma uma tarefa especificada. Dentre essas competições, destaca-se a *Robogames*, também conhecida como Olimpíada da Robótica, pois é o maior evento de competições de robótica do mundo e possui mais de 50 categorias, entre elas a *Robomagellan*.

O desafio proposto pela *Robomagellan* foca na navegação autônoma de robôs móveis terrestres em um ambiente não-estruturado. O robô deve ser capaz de navegar entre um ponto inicial até um ponto de destino, especificados por meio de coordenadas no Sistema de Posicionamento Global, do inglês *Global Positioning System* (GPS). Além disso, ele deve encontrar pontos específicos pelo caminho, demarcados por cones de sinalização de trânsito laranjas. Por ser um ambiente dinâmico, podem surgir obstáculos no percurso, como árvores, construções, placas de trânsito, entre outros³.

Um dos primeiros passos para a conclusão do desafio é tornar o robô capaz de se movimentar no ambiente entre os pontos desejados de forma eficiente e sem colidir com obstáculos. Com esse intuito, o presente trabalho propõe a elaboração de um sistema de controle de movimento de um robô móvel não-holonômico com tração diferencial, visando contribuir para a construção e aperfeiçoamento de um robô para participar da competição de navegação autônoma *Robomagellan*.

²Foto retirada de: <http://g1.globo.com/tecnologia/noticia/2013/12/startup-do-japao-vence-desafio-de-robos-criados-para-resgatar-humanos.html>.

³Fonte: <http://robogames.net/rules/magellan.php>.



Figura 1.2: *Curiosity Rover*⁴.

1.2 Definição do problema

A navegação autônoma de robôs em ambientes totalmente desconhecidos é um dos principais desafios da robótica móvel. Robôs devem ser capazes de se localizar no ambiente, tomar decisões, planejar e executar rotas, independente de intervenções humanas. Com o intuito de desenvolver essa área da robótica, surgem, cada vez mais, torneios que desafiam os participantes a construir robôs com essas habilidades, como a categoria *Robomagellan* da *Robogames*.

Neste trabalho é abordado o problema da movimentação autônoma de um robô móvel não-holonômico com tração diferencial, desenvolvido para o desafio proposto pela competição *Robomagellan*. Este problema envolve a geração e execução de uma trajetória entre os pontos de interesse, o desvio de obstáculos, a aproximação de objetos específicos e o controle das velocidades das rodas.

Para o rastreamento e controle de trajetória, optou-se por uma técnica que se baseia no modelo cinemático do robô e utiliza um modelo de referência para calcular a lei de controle, como a apresentada em [4]. E, para lidar com os possíveis obstáculos, foi adotada uma metodologia reativa, denominada zona virtual deformável, como a implementada em [5].

Portanto, é necessária a utilização de uma técnica de controle capaz de garantir que as velocidades do robô correspondam às calculadas como referência pela malha de controle de trajetória. Para tanto, foi escolhida a metodologia de controle adaptativo por modelo de referência, apresentada em [6] e [7].

Dispõe-se, para a implementação deste trabalho, do robô exibido na Figura 1.3. Este robô foi construído em conjunto com outros três trabalhos de graduação e uma breve descrição da sua arquitetura será realizada na Subseção 1.2.1.

⁴Foto retirada de: https://www.nasa.gov/mission_pages/msl/index.html.

1.2.1 Arquitetura do robô

A Figura 1.3 exibe a plataforma em que foi implementado o presente trabalho. Esse robô foi construído com o intuito de participar da competição *Robomagellan*. Por ser um desafio com um grau de complexidade considerável, esse projeto foi realizado paralelamente com outros três trabalhos de graduação e com o auxílio da equipe de competição Divisão de Robótica Inteligente (DROID) da Universidade de Brasília. A Figura 1.4 mostra o diagrama da arquitetura geral do robô, com todos os módulos desenvolvidos.

Como pode ser visualizado na imagem, a arquitetura do robô foi estruturada em módulos e a sua implementação foi dividida em três plataformas embarcadas no veículo, dois Arduinos (ATMega 2560) e um *Raspberry Pi 3*. Para a comunicação de dados entre esses sistemas, foi desenvolvido, em um trabalho complementar a este, um sistema executivo que gerencia as interfaces de comunicação. Esse sistema de mais alto nível, o executivo do robô, é denominado controlador de navegação. Ele também coordena a execução e a interação dos outros módulos, determinando as ações do robô, como, por exemplo, se ele deve calcular ou recalculer uma rota, executar a rota calculada, permanecer parado, se aproximar do cone, entre outras.

O sistema de localização é encarregado de fornecer a localização e a orientação atual do robô, no sistema de coordenadas NED, do inglês *North East Down*. Para tanto, o robô possui um sensor que recebe a sua posição no sistema de posicionamento global, do inglês *Global Positioning System* (GPS), e uma unidade de medida inercial, do inglês *Inertial Measurement Unit* (IMU).

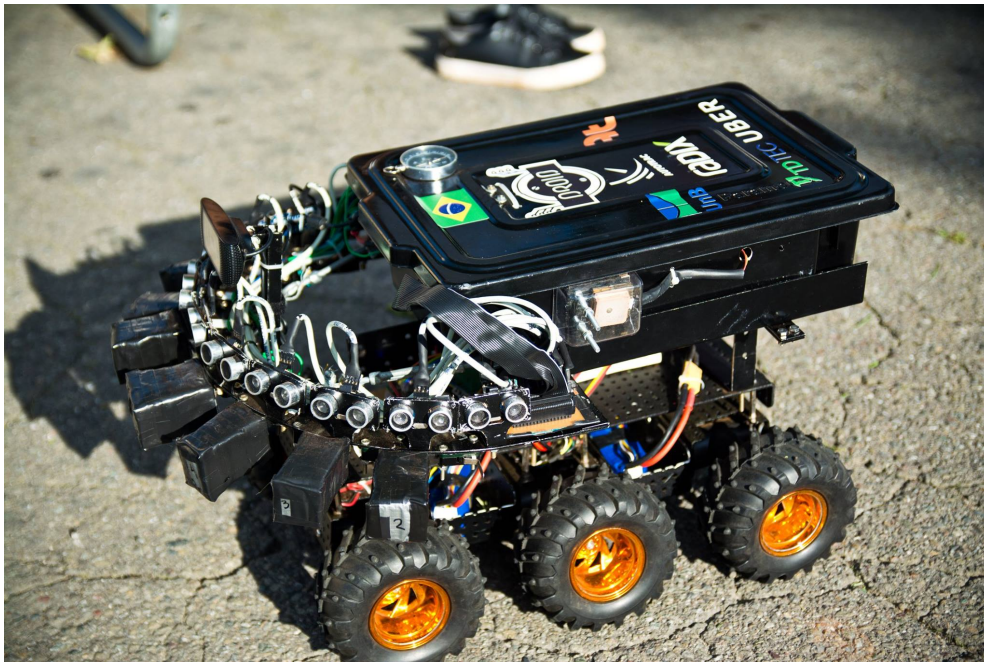


Figura 1.3: Robô desenvolvido para o projeto⁵.

O módulo do planejador de rotas possui um conhecimento prévio do ambiente da competição, no formato de um mapa, que possui informações limitadas dos lugares nos quais o robô pode se

⁵Fotografia por João Bosco Gouvêa Ramos.

movimentar, como ruas e calçadas; e locais caracterizados como obstáculos, como, por exemplo, prédios, escadas, cercas, entre outros. Esse planejador é responsável por, a partir da informação dos locais de interesse em coordenadas GPS, calcular uma rota entre a localização atual do robô e o próximo destino desejado. Essa rota é caracterizada por um conjunto de pontos que conectam as duas localizações informadas, de forma que esses pontos não coincidam com os obstáculos já conhecidos.

O módulo de visão computacional é o responsável por identificar os marcos, representados por cones de sinalização de trânsito laranjas. Esse módulo fornece a distância e a orientação que o cone está em relação à parte frontal do robô. Para essa finalidade, o robô foi equipado com uma câmera *webcam*.

O controlador de movimento, que é a parte da arquitetura desenvolvida neste trabalho, tem a sua execução gerenciada pelo controlador de navegação. Esse módulo recebe os dados fornecidos pelos outros sistemas para garantir que o robô calcule e execute as trajetórias entre os pontos fornecidos pelo planejador de rotas, recebendo como realimentação a posição atual do robô, proveniente do sistema de localização.

Além disso, esse sistema é responsável pelo desvio de obstáculos, a partir das leituras dos sensores ultrassônicos fornecidas pelo módulo executivo; e pela aproximação do cone, utilizando a distância e a orientação fornecidas pelo módulo de visão computacional. Ele também deve garantir que o robô permaneça na velocidade especificada, para tanto, este contém dois codificadores incrementais, que permitem o cálculo das velocidades das rodas da direita e da esquerda.

1.3 Objetivos do projeto

O objetivo principal deste trabalho é implementar um controlador de movimento em um robô móvel com tração diferencial, que irá participar do desafio proposto pela categoria *Robomagellan* da *Robogames*. Esse robô deve ser capaz de rastrear e executar trajetórias entre pontos especificados por um planejador de rotas, em um ambiente dinâmico, sem colidir com obstáculos. Esses obstáculos podem ser, por exemplo, um pedestre, um carro, uma lixeira, entre outros. Logo, é necessário um sistema que garanta a integridade física do robô. Além disso, para concluir o desafio, o robô deve ter a habilidade de se aproximar de objetos de interesse com acurácia. E, para que o robô seja capaz de executar todas essas condições, é fundamental que seja desenvolvido também um bom controlador de velocidade.

Assim, este trabalho tem como finalidade desenvolver um controlador formado a partir da interação de malhas de controle de trajetória, desvio de obstáculos, aproximação de objetos e velocidade. De modo que esse sistema coordene a movimentação do robô para a execução do desafio proposto pela competição *Robomagellan*. Espera-se que os resultados obtidos com este trabalho possam ser utilizados para pesquisas futuras na área da robótica móvel autônoma.

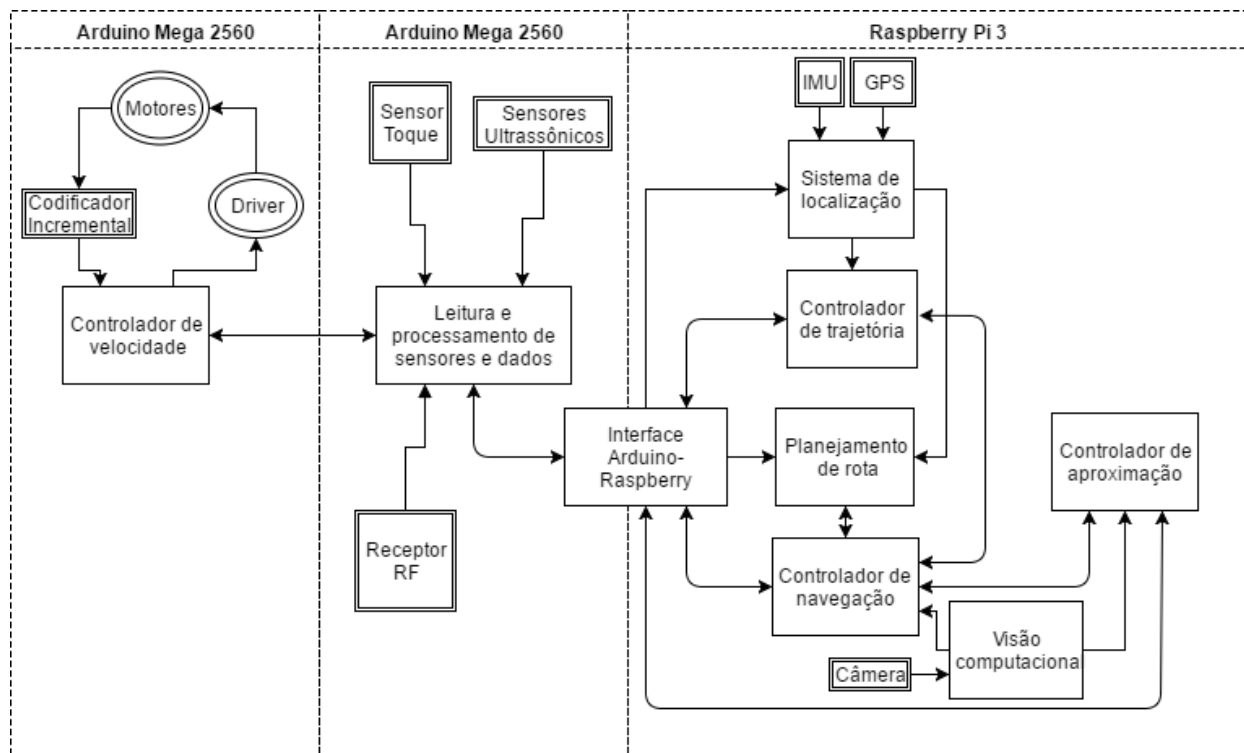


Figura 1.4: Diagrama da arquitetura do robô.

1.4 Resultados principais obtidos

Com o trabalho desenvolvido, foi possível controlar a movimentação de um robô móvel não-holonômico com tração diferencial. Com o controlador implementado, o robô foi capaz de gerar e executar uma trajetória entre pontos determinados, sem colidir com obstáculos; se aproximar de objetos de interesse; e aplicar as velocidades desejadas nos sistemas de propulsão.

Esse controlador foi desenvolvido para ser parte da arquitetura de um robô construído para executar o desafio da competição *Robomagellan*, porém, ele poderia ser implementado para controlar a movimentação de um robô em arquiteturas com outros objetivos.

1.5 Apresentação do manuscrito

Este trabalho está organizado em cinco capítulos. No Capítulo 2 é apresentada a revisão da literatura existente sobre as metodologias de controle utilizadas, assim como o embasamento teórico utilizado para o desenvolvimento deste projeto. No Capítulo 3 é apresentada a modelagem matemática desenvolvida durante o trabalho, descrevem-se as etapas e a maneira como o controlador de movimento foi implementado. No Capítulo 4 estão os resultados obtidos em testes realizados em ambiente de simulação e com o robô real, para a validação da metodologia proposta. O Capítulo 5 contém as conclusões deste trabalho e propostas para trabalhos futuros utilizando o controlador de movimento aqui apresentado. Em anexo, encontra-se uma descrição do CD que contém a versão digital do trabalho.

Capítulo 2

Fundamentação Teórica

2.1 Introdução

Este capítulo irá, primeiramente, fazer uma breve revisão bibliográfica da literatura já existente sobre o tema de controladores de movimento e sua aplicação direcionada à robótica móvel. Serão apresentados trabalhos relevantes acerca do assunto, bem como alguns artigos que foram fundamentais no desenvolvimento deste trabalho.

Além disso, este capítulo possui ênfase nos principais conceitos teóricos utilizados para o desenvolvimento do controlador de movimento proposto, que tem como objetivo fazer com que o robô execute uma determinada trajetória para alcançar os pontos de interesse com acurácia e sem que haja colisões com obstáculos. Para tanto, definiu-se quatro malhas de controle, que terão suas bases teóricas explicitadas. Essas malhas gerenciam a execução da trajetória, o desvio de obstáculos, a aproximação de objetos e a velocidade do robô.

2.2 Revisão bibliográfica

Esta seção tem como objetivo introduzir e discutir as abordagens encontradas na literatura acerca da navegação autônoma de robôs móveis em ambientes parcialmente desconhecidos.

Um dos objetivos da robótica móvel é o desenvolvimento de robôs autônomos, que tem como um dos desafios a locomoção destes em ambientes dinâmicos. Já que o robô deve ser capaz de navegar até um destino especificado, sem interferência humana, evitando possíveis obstáculos no seu caminho [2].

Como já explicado anteriormente, o objetivo deste trabalho é implementar um controlador de movimento de um robô autônomo não-holonômico, que permita a navegação em um ambiente aberto ou *outdoor*. Para tanto, foi necessário integrar técnicas de controle de velocidade, trajetória e desvio de obstáculos para que o robô pudesse realizar tarefas específicas, como navegar entre pontos determinados da forma mais precisa possível, evitando colisões.

Os controladores de velocidade são responsáveis por atuarem nos sistemas de propulsão do robô para que este se mova na velocidade desejada. Entre as diferentes técnicas para implementar um controle de velocidade, destacam-se o Controlador Proporcional Integral Derivativo (PID) [8]; e o Controlador Adaptativo por Modelo de Referência, que pode modificar o seu comportamento, adaptando-se ao sistema a ser controlado, como o implementado em [6].

Um bom controlador de velocidade é fundamental para sistemas com uma estrutura em cascata, como a desenvolvida pelo autor em [6], que utiliza a malha de controle de velocidade para manter o robô na velocidade de referência calculada pelo controlador de trajetória.

Os controladores de trajetória são encarregados de fazer com que o veículo siga uma trajetória de referência. Essa trajetória pode ser representada por curvas livres, como em [9], que gera trajetórias para um robô autônomo baseadas nas curvas de Bézier; ou por forma de estruturas geométricas, como arcos e segmentos de reta, que são mais eficientemente tratáveis do ponto de vista computacional [10], além de serem de mais fácil execução em robôs não-holonômicos.

Em relação aos controladores de trajetória, a maioria destes são baseados no modelo cinemático, como em [11], que desenvolve, a partir do modelo cinemático, um controlador de seguimento de trajetória baseado em Lógica Fuzzy. Entretanto, também foram propostos métodos baseados no modelo dinâmico do robô. Em [4], foi desenvolvido um controlador de trajetória adaptativo para a dinâmica de um robô móvel não-holonômico com parâmetros desconhecidos.

Os controladores baseados no modelo cinemático possuem a velocidade como saída, desse modo, eles exigem bons controladores de velocidade. E, por usar uma estrutura em cascata, a dinâmica do erro de velocidade tem que ser bem mais rápida que a do erro de trajetória. Estes controladores baseados no modelo cinemático foram considerados mais adequados para o desenvolvimento deste trabalho, devido ao *hardware* disponível para implementação. Desse modo, foi escolhido um controlador de seguimentos de trajetória baseado em um robô de referência, como o modelado em [4].

Além de minimizar o erro em relação à trajetória de referência, desviar de obstáculos durante a navegação autônoma também é uma tarefa primordial, já que torna o robô capaz de chegar ao seu destino sem colisões. Dessa forma, várias soluções têm sido propostas para o problema de detecção de obstáculos e como o robô deve agir para evitá-los. Esses métodos podem ser classificados como: deliberativos, reativos ou híbridos [2].

A Zona Virtual Deformável (ZVD) é um método de controle reativo para o desvio de obstáculos e foi utilizada em diversos trabalhos. Em [12], um conjunto de robôs móveis utilizam esse método para, de forma cooperativa, modificar o processo de geração de trajetórias a partir das informações fornecidas por todos os robôs. Já em [5], esse método é integrado a um filtro de partículas visando fazer com que um robô, que possui apenas um sensor *laser* a frente, possa navegar sem colidir com obstáculos. A partir dessa análise, nota-se que métodos reativos, como a ZVD, são mais eficientes em ambientes dinâmicos e possuem custo computacional mais baixo que os métodos deliberativos. Logo, percebe-se que a ZVD é uma escolha coerente para o problema analisado neste trabalho.

Analisando a literatura existente que engloba o controle de movimentação de robôs móveis

autônomos nos âmbitos da velocidade, trajetória e desvio de obstáculos, percebe-se que ainda não é um problema completamente resolvido. As diversas técnicas existentes estão sendo constantemente aprimoradas. Ainda assim, em muito casos, principalmente em casos de trabalhos sobre controladores de trajetória, o sistema é executado em ambientes estáticos.

Através do estudo da literatura, foram escolhidos os métodos e técnicas mais adequados para implementar o controlador de movimento, levando em consideração o *hardware* disponível.

2.3 Controladores de velocidade

O controlador de velocidade é encarregado de garantir que os motores permaneçam na velocidade desejada, para que se possa atingir a posição de referência ou desviar de um obstáculo.

Para a malha de controle da velocidade podem ser aplicadas diversas metodologias de controle, entre elas, a do controle Proporcional Integral Derivativo (PID) e a do controlador adaptativo por modelo de referência, do inglês *Model Reference Adaptive Control* (MRAC). A metodologia escolhida para o controle das velocidades do robô, neste trabalho, foi a do MRAC. Pois esse controlador garante que os sistemas de propulsão do robô possuam um comportamento predefinido mesmo se estes apresentarem mudanças no decorrer da execução da rotina do robô.

2.3.1 Controlador proporcional integral derivativo

O controlador Proporcional Integral Derivativo (PID) é uma técnica de controle de processos, que atua sobre uma variável a ser manipulada a partir de três ações de controle: ação de controle proporcional, ação de controle integral e ação de controle derivativo [8]. O diagrama de blocos do controlador PID é exibido na Figura 2.1.

O projeto do controlador pode ser baseado na representação em segunda ordem dos sistemas de propulsão, apresentada na equação

$$G(s) = \frac{\Omega(s)}{U(s)} = \frac{b_1s + b_0}{s^2 + a_1s + a_0}, \quad (2.1)$$

que relaciona, no domínio de Laplace, a velocidade de rotação $\Omega(s)$ com a tensão de entrada do motor $U(s)$.

O controlador PID combina as três ações de controle do erro entre a referência de velocidade de rotação $\omega^*(s)$ e a saída $\omega(s)$ aplicadas na entrada de excitação do sistema. Dessa forma, a lei de controle, no domínio do tempo, é dada por

$$u(t) = u(0) + K_p e(t) + K_i \int_0^t e(\tau) d\tau + K_d \frac{de(t)}{dt}, \quad (2.2)$$

com $e(t) = \omega^*(t) - \omega(t)$ e em que K_p é o ganho proporcional, K_i o ganho integral e K_d o ganho derivativo.

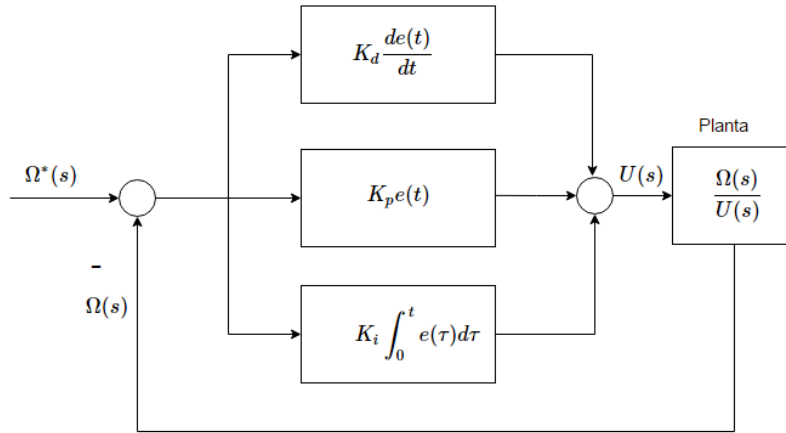


Figura 2.1: Diagrama de blocos de um controlador PID.

Aplicando a transformada de Laplace, considerando $u(0) = 0$, a lei de controle é dada pela equação

$$\frac{U(s)}{E(s)} = K_p + \frac{K_i}{s} + K_d s. \quad (2.3)$$

Para a aplicação desse controlador no veículo, os parâmetros K_p , K_i e K_d podem ser obtidos empiricamente através da realização de testes.

2.3.2 Controlador adaptativo por modelo de referência

A ideia geral do Controle Adaptativo por Modelo de Referência, do inglês *Model Reference Adaptive Control* (MRAC), é criar um controlador em malha fechada com parâmetros que são atualizados para alterar a resposta do sistema. Nesse controlador, a saída do sistema é comparada com uma saída desejada de um modelo de referência e um sinal de erro é gerado.

Assim, os parâmetros do controlador são atualizados com base neste erro através de regras de adaptação baseadas na teoria de estabilidade de Lyapunov. O objetivo é fazer com que os parâmetros converjam para valores que façam a resposta do sistema coincidir com a resposta do modelo de referência [7]. O diagrama de blocos do controlador MRAC pode ser visualizado na Figura 2.2.

Para o projeto de um controlador MRAC pode ser considerado um sistema de primeira ordem dado por uma função de transferência de acordo com

$$G(s) = \frac{B(s)}{A(s)} = \frac{b_0}{a_1 s + a_0} = \frac{\Omega(s)}{U(s)}, \quad (2.4)$$

que corresponde à equação diferencial

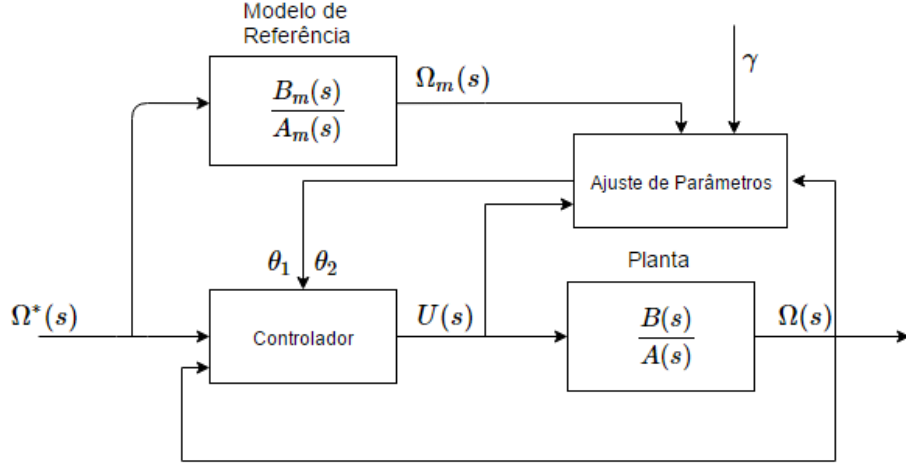


Figura 2.2: Diagrama de blocos de um controlador MRAC.

$$\dot{\omega}(t) = -a\omega(t) + bu(t). \quad (2.5)$$

Considerando um modelo de referência em malha fechada, que apresenta a forma descrita na equação

$$G_m(s) = \frac{b_m}{s + a_m} = \frac{\Omega(s)}{\Omega^*(s)}. \quad (2.6)$$

A partir de $G_m(s)$, pode-se obter a equação diferencial

$$\dot{\omega}_m(t) = -a_m\omega_m(t) + b_m\omega^*(t), \quad (2.7)$$

que descreve o modelo de referência no domínio do tempo.

O projeto do MRAC é baseado na combinação de uma lei de controle com uma lei de adaptação, que gera estimativas em tempo real dos parâmetros do controlador. O sinal de controle será definido por meio de uma combinação linear do sinal de referência $\Omega^*(s)$, do sinal de saída da planta $\Omega(s)$ e dos parâmetros do controlador θ_1 e θ_2 [6]. Dessa forma, é esperado que o sistema siga o modelo de referência com a lei de controle

$$U(s) = \theta_1\Omega^*(s) - \theta_2\Omega(s). \quad (2.8)$$

Os parâmetros do controlador, que irão levar o sistema em malha fechada a apresentar um comportamento conforme o modelo de referência, são definidos como

$$\theta_1 = \frac{b_m}{b}, \quad (2.9)$$

$$\theta_2 = \frac{a_m - a}{b}. \quad (2.10)$$

Os parâmetros (θ_i) são ajustados em função do erro entre a saída da planta e a saída do modelo de referência. Neste controlador, se o modelo da planta variar ou se o modelo inicial não é exato, ele ainda deve ser capaz de fazer o sistema se comportar de acordo com o modelo de referência. O projeto desse controlador apresentado foi retirado de [6, 13].

O erro $e(t)$ é definido pela diferença entre o sinal de saída gerado pela planta e o sinal de saída do modelo de referência, dessa forma, $e(t)$ é dado por

$$e(t) = \omega(t) - \omega_m(t). \quad (2.11)$$

O controlador será projetado buscando leis de adaptação que já incorporam, a priori, garantia de estabilidade, por pertencerem a uma classe que satisfaz algum critério do tipo Lyapunov [13].

Introduzindo-se a função candidata de Lyapunov, dada pela equação

$$V(e, \theta_1, \theta_2) = \frac{1}{2} \left(e^2 + \frac{1}{b\gamma} (b\theta_2 + a - a_m)^2 + \frac{1}{b\gamma} (b\theta_1 - b_m)^2 \right), \quad (2.12)$$

sendo $V(e, \theta_1, \theta_2)$ uma função real positiva definida e igual a zero se o erro $e(t)$ for nulo e se os parâmetros do controlador forem iguais às equações (2.9) e (2.10). A derivada dessa função é dada por

$$\frac{dV}{dt} = e(t) \frac{de(t)}{dt} + \frac{1}{\gamma} (b\theta_2 + a - a_m) \frac{d\theta_2}{dt} + \frac{1}{\gamma} (b\theta_1 - b_m) \frac{d\theta_1}{dt}. \quad (2.13)$$

Tal que para $V(e, \theta_1, \theta_2)$ ser qualificada como uma função de Lyapunov, deve-se ter $dV/dt < 0$ [6, 13]. Sendo

$$\frac{de(t)}{dt} = -a_m e(t) - (b\theta_2 + a - a_m)\omega(t) + (b\theta_1 - b_m)\omega^*(t), \quad (2.14)$$

obtem-se

$$\frac{dV}{dt} = -a_m e^2(t) + \frac{1}{\gamma} (b\theta_2 + a - a_m) \left(\frac{d\theta_2}{dt} - \gamma \omega(t) e(t) \right) + \frac{1}{\gamma} (b\theta_1 - b_m) \left(\frac{d\theta_1}{dt} + \gamma \omega^*(t) e(t) \right), \quad (2.15)$$

na qual γ é o ganho de adaptação dos parâmetros dos controladores.

Desse modo, com as regras de adaptação dos parâmetros dadas por

$$\frac{d\theta_1}{dt} = -\gamma\omega^*(t)e(t), \quad (2.16)$$

$$\frac{d\theta_2}{dt} = \gamma\omega(t)e(t), \quad (2.17)$$

obtém-se

$$\frac{dV}{dt} = -a_m e^2(t) < 0. \quad (2.18)$$

Isto implica que, sendo V uma função sempre real positiva, $V(t) < V(0)$ para $t > 0$. Mas não implica que haja convergência dos parâmetros do controlador para os seus valores corretos. Segundo [6], a convergência de parâmetros só é garantida com excitação persistente.

2.4 Controle e rastreamento de trajetória

Para resolver o problema do rastreamento de trajetórias aplicado em robôs móveis autônomos, as abordagens existentes estão divididas em dois paradigmas, o paradigma do controle sequencial de postura e o paradigma do segmento de caminho.

O controle sequencial de posturas consiste em, a cada passo de amostragem, a postura desejada para o robô é definida como um ponto sobre a trajetória de referência, que é parametrizada pelo tempo. Nesse caso, o robô estará seguindo a trajetória predeterminada, amostrada sob forma de sequência de posturas.

No segmento de caminho, não existe controle sobre a posição do robô, mas sobre variáveis que definem o erro de trajetória, que são em função da posição atual e da estrutura geométrica sendo rastreada [10].

Já o controlador de trajetória tem como objetivo principal fazer com que o robô consiga seguir a trajetória determinada minimizando o erro entre a trajetória de referência e a executada pelo robô. Sendo assim, as estratégias de controle mais adotadas utilizam a realimentação ou *feedback* para mensurar o erro no sistema controlado.

Neste trabalho, é apresentado um planejador de movimento com o paradigma do segmento de caminho e um controlador de trajetória, que recebe como entrada pontos desejados e é encarregado de garantir que o robô permaneça no caminho almejado, não colida com obstáculos e chegue no destino final na orientação correta.

Dessa forma, foi implementado um controlador baseado no modelo cinemático do robô que gera como saída as velocidades de referência para os motores. Além disso, esse controlador está integrado a um controle reativo de desvio de obstáculos, a Zona Virtual Deformável (ZVD).

2.4.1 Segmentos de caminho

Quando utilizado o paradigma de segmentos de caminho para o rastreamento de trajetórias, é especificada uma trajetória paramétrica que o robô deverá seguir de maneira satisfatória¹. A Figura 2.3 ilustra um exemplo de uma trajetória paramétrica, na qual os trechos (λ_i) representam os segmentos de reta ou os arcos que conectam os pontos da rota (P_j).

Os segmentos de reta da trajetória, como o exibido na Figura 2.4, podem ser caracterizados pelos pontos inicial $P_a = (x_a, y_a)$ e final $P_b = (x_b, y_b)$, pelo seu ângulo de inclinação α e pelo vetor de direção \vec{d} , calculado como

$$\vec{d} = \frac{P_b - P_a}{\|P_b - P_a\|}. \quad (2.19)$$

Já as trajetórias em arco de círculo, como a exibida na Figura 2.5, podem ser caracterizadas pelo centro do arco (x_0, y_0) , pelo raio r , pelo ângulo inicial α_0 e pelo ângulo final α_1 . Nesse tipo de trajetória deve ser válida a relação

$$\alpha_1 - \alpha_0 \leq \pi. \quad (2.20)$$

Um arco que não obedeça essa relação e possua uma angulação maior deve ser subdividido em segmentos menores.

2.4.2 Planejador de movimento

O planejador de movimento é encarregado de determinar qual segmento de trajetória deve ser rastreado a cada instante. Desse modo, é fundamental que ele identifique quando o segmento atual foi devidamente seguido.

Uma das abordagens possíveis para reconhecer se um segmento ainda está sendo seguido, é determinar uma zona na qual é considerado que o robô está seguindo a trajetória λ_i e implementar uma função de teste $f(t) \in \lambda_i$, que identifique se o robô se encontra dentro dessa zona ou não.

No caso de um caminho representado pelo segmento de reta exibido na Figura 2.6, a região na qual pode ser considerado que o robô está rastreando o segmento λ_i pode ser especificada por todo ponto $(x, y) \in X \times Y$ que, quando transformado para o sistema de coordenadas $x^b x^b y^b$, possuir a componente do eixo x^b negativa.

Sendo o ponto $\mathbf{p} = (x, y)^T \in \lambda_i$, sua transformação para o eixo de coordenadas $x^b x^b y^b$ é feita por:

$$\mathbf{R}^T(\alpha) * \left\{ \begin{pmatrix} x \\ y \end{pmatrix} - \begin{pmatrix} x_b \\ y_b \end{pmatrix} \right\}, \quad (2.21)$$

sendo $\mathbf{R}^T(\alpha)$ a matriz de rotação no plano, dada por

¹Retirado dos materiais de aula do professor Geovany Borges.

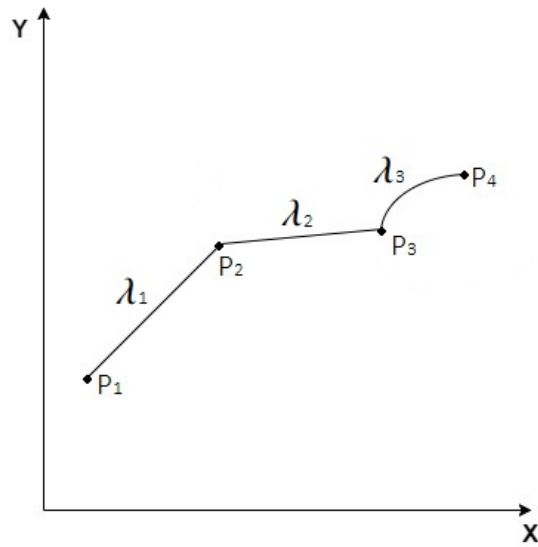


Figura 2.3: Modelo de uma trajetória baseada em segmentos de caminho.

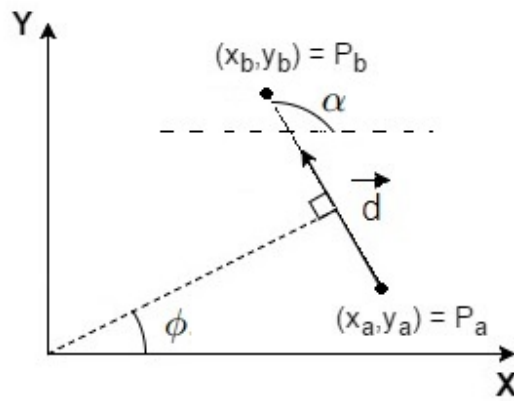


Figura 2.4: Segmento de caminho em forma de segmento de reta.

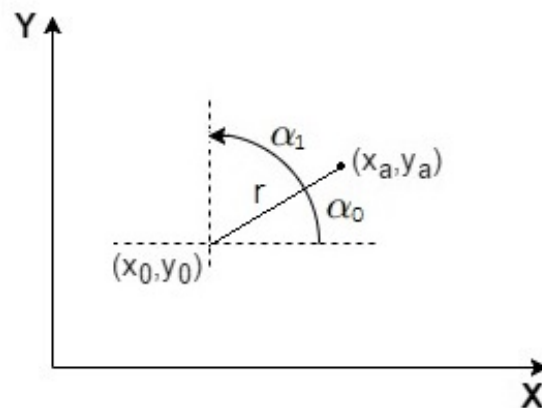


Figura 2.5: Segmento de caminho em forma de arco de círculo.

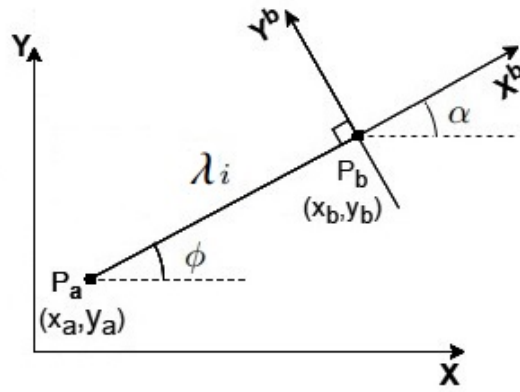


Figura 2.6: Trajeto em forma de segmento de reta.

$$\mathbf{R}^T(\alpha) = \begin{pmatrix} \cos(\alpha) & \sin(\alpha) \\ -\sin(\alpha) & \cos(\alpha) \end{pmatrix}, \quad (2.22)$$

e o ângulo α o ângulo de rotação do eixo $x^b y^b$ em relação ao eixo XxY , obtido por

$$\alpha = \arctan 2(y_b - y_a, x_b - x_a). \quad (2.23)$$

Desse modo, extraíndo-se a componente em x^b , o ponto $p = (x, y)^T$ pertencerá ao segmento de caminho λ_i , se

$$\cos(\alpha) * (x - x_b) + \sin(\alpha) * (y - y_b) < 0. \quad (2.24)$$

Assim, se um robô iniciou seguindo a trajetória dada pelo segmento λ_i , o planejador de movimento considerará o segmento completamente seguido e mudará para o próximo trecho do trajeto quando a condição dada pela equação (2.24) falhar, sendo $\mathbf{p} = (x, y)^T$ a posição atual do robô.

2.4.3 Controlador de trajetória por modelo de referência

Considerando um robô com tração diferencial, como o exibido na Figura 2.7. Seja $\mathbf{p} = [x, y, \theta]^T$ o vetor que representa a postura do robô, no qual $(x, y) \in \mathbb{R}^2$ denota a posição do centro de massa do robô e $\theta \in [-\pi, \pi]$ é o ângulo entre o vetor velocidade linear e o eixo horizontal. Para esse caso, têm-se o modelo cinemático [4]

$$\frac{d}{dt} \begin{bmatrix} x \\ y \\ \theta \\ \theta_d \\ \theta_e \end{bmatrix} = \begin{bmatrix} \frac{r}{2} \cos \theta & \frac{r}{2} \cos \theta \\ \frac{r}{2} \sin \theta & \frac{r}{2} \sin \theta \\ r & -r \\ \frac{r}{2b} & -\frac{r}{2b} \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \omega_d \\ \omega_e \end{bmatrix}, \quad (2.25)$$

no qual θ_d e θ_e são, respectivamente, os ângulos das rodas direita e esquerda, ω_d e ω_e são, respectivamente, as velocidades angulares das rodas direita e esquerda, r é o raio das rodas e b é a metade da largura do robô, como exibido na Figura 2.7.

Esse modelo pode ser simplificado considerando apenas a relação entre as velocidades angulares das rodas e a velocidade linear v e angular ω do robô, de modo que

$$\begin{bmatrix} \omega_d \\ \omega_e \end{bmatrix} = \begin{bmatrix} \frac{1}{r} & \frac{b}{r} \\ \frac{1}{r} & -\frac{b}{r} \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix}. \quad (2.26)$$

Combinando-se as equações (2.25) e (2.26), obtêm-se o modelo cinemático do robô com tração diferencial, dado por

$$\frac{d}{dt} \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} = \begin{bmatrix} \cos \theta & 0 \\ \sin \theta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix}. \quad (2.27)$$

Esse método de rastreamento de trajetória realiza o controle do movimento do robô utilizando um modelo de um robô, chamado de robô virtual, como referência, que executa o caminho planejado de maneira ideal. Desse modo, o robô real se mantém na trajetória desejada ao se movimentar “seguindo” o robô virtual.

O modelo cinemático do robô de referência é dado por

$$\frac{d}{dt} \begin{bmatrix} x_r \\ y_r \\ \theta_r \end{bmatrix} = \begin{bmatrix} \cos \theta_r & 0 \\ \sin \theta_r & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_r \\ \omega_r \end{bmatrix}, \quad (2.28)$$

no qual $\mathbf{p}_r = [x_r, y_r, \theta_r]^T$ é a configuração de posição do robô de referência e o vetor $\mathbf{q}_r = [v_r, \omega_r]^T$ fornece as suas velocidades linear v_r e angular ω_r .

Os erros e_1 , e_2 e e_3 descrevem a diferença de posição e orientação entre o robô real e o modelo de referência e são definidos como

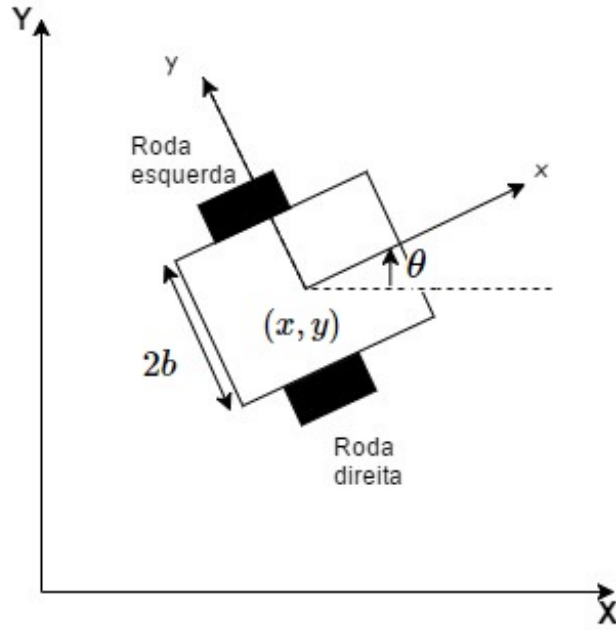


Figura 2.7: Modelo de um robô diferencial no sistema de coordenadas inercial.

$$\begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix} = \begin{bmatrix} \cos \theta & \sin \theta & 0 \\ -\sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_r - x \\ y_r - y \\ \theta_r - \theta \end{bmatrix}. \quad (2.29)$$

As leis de controle, apresentadas nas equações

$$v_t = v_r \cos e_3 + K_1 e_1, \quad (2.30)$$

$$\omega_t = \omega_r + v_r K_2 e_2 + K_3 \sin e_3, \quad (2.31)$$

retornam as velocidades linear v_t e angular ω_t que o robô real deve ter para que os erros e_1 , e_2 e e_3 convirjam para zero [4]. Sendo K_1 , K_2 e K_3 constantes positivas.

Os erros e_1 , e_2 e e_3 satisfazem a relação [4]

$$\frac{d}{dt} \begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix} = v \begin{bmatrix} -1 \\ 0 \\ 0 \end{bmatrix} + \omega \begin{bmatrix} e_2 \\ -e_1 \\ -1 \end{bmatrix} + \begin{bmatrix} v_r \cos e_3 \\ v_r \sin e_3 \\ \omega_r \end{bmatrix}. \quad (2.32)$$

A estabilidade desse controlador pode ser verificada através de uma análise baseada nos critérios de Lyapunov [13]. Define-se a função candidata de Lyapunov V_0 , dada pela equação (2.33), sendo que, para V_0 ser classificada como uma função de Lyapunov deve-se ter $dV_0/dt < 0$.

$$V_0 = \frac{1}{2}(e_1^2 + e_2^2) + \frac{1 - \cos e_3}{K_2}, \quad (2.33)$$

de modo que V_0 é uma função sempre real positiva e é igual a zero, caso os erros e_1 , e_2 e e_3 sejam nulos. Além disso, pode-se verificar que a sua derivada satisfaz a inequação [4]

$$\frac{d}{dt}V_0 = e_1 \frac{de_1}{dt} + e_2 \frac{de_2}{dt} + e_3 \frac{de_3 \sin e_3}{K_2} = -K_1 e_1^2 - \frac{K_3 \sin^2 e_3}{K_2} \leq 0. \quad (2.34)$$

Isso implica que a função pode ser classificada como uma função de Lyapunov. Em [4] ainda é mostrado que $\lim_{t \rightarrow \infty} (\mathbf{p}(t) - \mathbf{p}_r(t)) = 0$, sendo $\mathbf{p}_r(t)$ a posição do robô de referência e $\mathbf{p}(t)$ a posição do robô real.

Substituindo na equação (2.32) as velocidades linear v e angular ω do robô pelas velocidades de rotação das rodas ω_d e ω_e , obtêm-se

$$\frac{d}{dt} \begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix} = \omega_d \begin{bmatrix} -\frac{r}{2} + \frac{r}{2b} e_2 \\ -\frac{r}{2b} e_1 \\ \frac{r}{2b} \end{bmatrix} + \omega_e \begin{bmatrix} -\frac{r}{2} - \frac{r}{2b} e_2 \\ \frac{r}{2b} e_1 \\ \frac{r}{2b} \end{bmatrix} + \begin{bmatrix} v_r \cos e_3 \\ v_r \sin e_3 \\ \omega_r \end{bmatrix}. \quad (2.35)$$

Definindo os parâmetros a_1 e a_2 como

$$a_1 = \frac{1}{r}, \quad (2.36)$$

$$a_2 = \frac{b}{r}, \quad (2.37)$$

e assumindo que haja uma constante positiva δ , tal que $a_2 \geq \delta$, sendo $a_2 > 0$.

As velocidades de rotação ω_d e ω_e das rodas são calculadas como

$$\begin{bmatrix} \omega_d \\ \omega_e \end{bmatrix} = \begin{bmatrix} a_1 & a_2 \\ a_1 & -a_2 \end{bmatrix} \begin{bmatrix} v_t \\ \omega_t \end{bmatrix}. \quad (2.38)$$

2.5 Desvio de obstáculos

A robótica móvel enfatiza os problemas relacionados à locomoção dos robôs em ambientes complexos, que se modificam dinamicamente, e um dos objetivos da robótica móvel é a criação de robôs autônomos. Estes devem ser capazes de navegar nesses ambientes sem que haja a intervenção humana, ou seja, devem adquirir e utilizar o conhecimento sobre o ambiente, ter a habilidade de detectar obstáculos e agir de forma que possam concluir a tarefa que lhe foi designada [2]. Essa interação que o robô deve ter com o ambiente de trabalho é conseguida através dos vários sensores instalados a bordo do veículo [14, 15].

Sempre que um robô navega em um ambiente não-estruturado, é necessário assegurar que ele não irá colidir com obstáculos. Logo, na maioria das aplicações em robótica móvel é necessário um

método eficiente de desvio de obstáculos quando o robô é posto para navegar. Esses métodos, de acordo com a abordagem adotada, podem ser classificados de forma genérica como: deliberativos, reativos ou híbridos [16]. Dentre eles podem ser citados o Método de Detecção de Bordas [17], Grade de Certeza, do inglês *Certainty Grid* [18], Campos Potenciais [19], Campo de Forças Virtual, do inglês *Virtual Force Field* [20], Histograma do Campo Vetorial, do inglês *Virtual Force Histogram* [21] e a Zona Virtual Deformável (ZVD) [5].

2.5.1 Métodos reativos

Os métodos reativos não requerem nenhum conhecimento prévio do local no qual o robô irá navegar, o que pode impossibilitar o planejamento prévio de uma trajetória, pois não é necessário que exista um mapa do ambiente, a não ser alguma informação geral como o conhecimento que o recinto é plano e fechado. A navegação baseada nessa abordagem tem como principal característica o fato de que todas as ações são realizadas por meio do mapeamento de informações, obtidas com os sensores do robô, para um padrão de ações nos atuadores [2].

Nesse método o robô reage ao que percebe, o que permite que ele responda mais rapidamente a mudanças em seu entorno. Essa característica permite que um robô navegue em ambientes dinâmicos, onde os obstáculos podem mudar de posição. Como resultado, obtêm-se simplicidade e menor custo computacional, dois aspectos relevantes em robótica móvel.

Os métodos de Campos Potenciais, Zona Virtual Deformável (ZVD) e o da Detecção de Bordas são classificados como reativos.

2.5.2 Métodos deliberativos

Os métodos deliberativos são aqueles cuja navegação é baseada em um modelo do mundo ou mapa do ambiente no qual o robô está inserido. Nessa abordagem, inclui-se uma etapa de pré-planejamento da trajetória que será realizada a partir de um mapa previamente conhecido e das informações obtidas através dos sensores. O sucesso desse método baseia-se na fidelidade do modelo do mundo e da precisão dos sensores [5].

Como nesta abordagem é realizado um mapeamento do ambiente, é possível que o robô execute uma trajetória sem colidir com obstáculos até o ponto de destino, desde que esse caminho exista. Entretanto, os métodos que utilizam essa abordagem têm um elevado custo computacional e não são eficientes em ambientes dinâmicos.

São classificados como deliberativos os métodos da Grade de Certeza, do Histograma de Campo Vetorial e do Campo de Forças Vetoriais.

2.5.3 Métodos híbridos

No método híbrido as duas abordagens, deliberativa e reativa, são utilizadas em conjunto para realizar o planejamento das tarefas a serem realizadas e para lidar com situações de reação imediata,

como o surgimento de obstáculos dinâmicos no ambiente [2].

O algoritmo de desvio de obstáculos implementado neste trabalho foi baseado na arquitetura de controle híbrida. O robô possuía conhecimento prévio do ambiente, porém, por se tratar de um ambiente dinâmico, ao ar livre no meio de uma cidade, ele também precisava reagir ao ambiente de forma que pudesse chegar ao destino desejado sem colisões com obstáculos, como pedestres, animais, árvores ou carros.

Desse modo, foi implementado um algoritmo reativo Zona Virtual Deformável (ZVD) em conjunto com um método deliberativo, o planejador de rotas explicitado anteriormente.

2.5.4 Zona virtual deformável

O método das zonas virtuais deformáveis é um método de controle reativo desenvolvido originalmente para o desvio de obstáculos em robótica móvel [22, 23]. O método consiste no modelamento de uma zona protetora imaginária ao redor do robô, cujos parâmetros dependem do estado deste.

Assume-se que o robô não possui um conhecimento prévio do ambiente e a aproximação de objetos é detectada por sensores de proximidade. Quando algum objeto penetra nesta zona, causa uma deformação. E o sinal de controle é calculado de forma que o robô reaja no sentido que minimizará esta deformação [24].

De acordo com o princípio geral desse método, para controlar o comportamento reativo de um robô móvel, que se move no espaço \mathbb{R}^2 , é necessário modelar uma ZVD que envolva o robô. Modela-se a ZVD em geral como uma elipse no caso bidimensional, ou um elipsoide para o caso tridimensional, mas sua geometria poderá ser alterada em função da cinemática do robô e dos obstáculos detectados [25].

Os parâmetros desta elipse dependem do estado do robô. Em [25], podem ser vistas algumas definições que descrevem a abordagem da ZVD e a sua parametrização.

A Zona Virtual Deformável (ZVD) pode ser representada por um vetor \mathbf{s} , em que cada componente s_i corresponde a distância fornecida por cada sensor ultrassônico, que representa o comprimento desde o limite da ZVD até o centro de massa do robô. Assim, esse vetor será dado por

$$\mathbf{s} = [s_0, s_1 \dots s_n]^T. \quad (2.39)$$

O vetor \mathbf{q} caracteriza o movimento do robô e, quando a ZVD sofrer alguma deformação, o controle estará encarregado de fazer com que o robô reaja, no intuito de minimizar a deformação da ZVD. Esse vetor é composto pela velocidade linear v e velocidade angular ω , de modo que

$$\mathbf{q} = \begin{bmatrix} v \\ \omega \end{bmatrix}. \quad (2.40)$$

Dessa forma, a zona protetora imaginária depende unicamente do vetor que caracteriza o movimento do robô. Quando este se mover pelo ambiente, pode se aproximar de algum objeto, que entraria na zona e causaria uma deformação na ZVD. Essa deformação é caracterizada pelo vetor \mathbf{s}_D , apresentado na equação

$$\mathbf{s}_D = [s_{D0}, s_{D1} \dots s_{Dn}]^T, \quad (2.41)$$

na qual cada componente s_{D_i} depende da variação da medida de distância ρ_i fornecida pelo sensor ultrassônico correspondente.

O vetor de intrusão \mathbf{i} , dado por

$$\mathbf{i} = [i_0, i_1 \dots i_n]^T, \quad (2.42)$$

representa a profundidade da intrusão gerada pelo obstáculo na ZVD para cada s_i . Suas componentes são calculadas como: $i_i = d_{imax} - \rho_i$. Na ausência de obstáculos, o sensor fornece uma medida $\rho_i = d_{imax}$.

O vetor de deformação Δ , que representa a profundidade da deformação da zona protetora, é caracterizado por

$$\Delta = [\Delta_0, \Delta_1 \dots \Delta_n]^T. \quad (2.43)$$

A deformação Δ_i depende da intrusão de informação de proximidade e da configuração não-deformada da zona protetora, sendo calculada como

$$\Delta_i = \begin{cases} 0, & \text{se } \rho_i \geq s_i \\ s_i - \rho_i, & \text{se } \rho_i < s_i. \end{cases} \quad (2.44)$$

Esta deformação pode ser minimizada ou eliminada pelo robô através de ações de evasão do obstáculo, como uma mudança na direção do robô, para evitar uma colisão com o obstáculo encontrado [5]. Neste trabalho, será calculado um vetor de velocidades $\mathbf{q}_z = [v_z, \omega_z]^T$ a serem aplicadas no robô, de modo que ele reaja aos obstáculos e minimize as deformações da zona de proteção.

A Figura 2.8 exhibe uma situação na qual o robô se aproxima de um obstáculo, que gera uma deformação na ZVD.

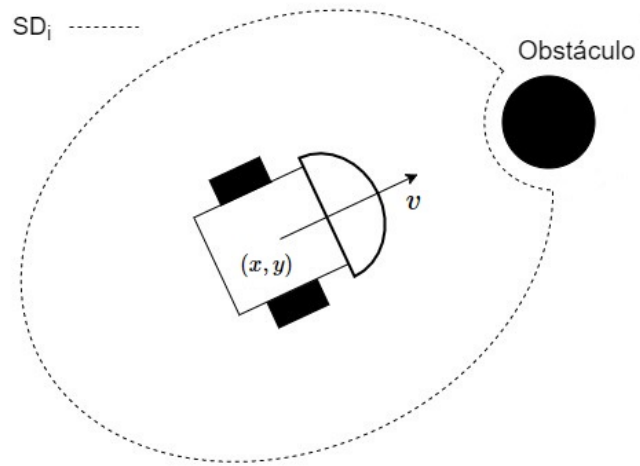


Figura 2.8: Representação de uma deformação na ZVD.

Capítulo 3

Estrutura do Controlador de Movimento

3.1 Introdução

Este capítulo tem como objetivo apresentar a metodologia implementada no presente trabalho para o desenvolvimento de um controlador de movimento para um robô móvel não-holonômico com tração diferencial.

Serão explicitados os módulos que compõem esse controlador, assim como a integração destes, em uma configuração de baixo para cima. O primeiro módulo do controlador de movimento desenvolvido foi o controlador de velocidade, utilizando a metodologia do controle adaptativo por modelo de referência, dado que os outros sistemas de controle dependem diretamente do bom funcionamento deste.

O segundo módulo implementado foi a malha de controle e rastreamento de trajetória, com o intuito de garantir que o robô execute de maneira satisfatória a rota calculada pelo planejador de rotas. Após a sua implementação, foi constatada a necessidade de um sistema que tratasse o surgimento de obstáculos durante a execução do controlador de trajetória, portanto, foi desenvolvida uma malha de controle reativo para o desvio de obstáculos, com a metodologia das Zonas Virtuais Deformáveis.

Além disso, depois de executada a rota especificada, era necessário que o robô fosse capaz de realizar uma aproximação fina do alvo, representado por um cone de sinalização. Assim, foi elaborada uma malha de controle de aproximação para lidar com este problema.

O diagrama do controlador de movimento é exibido na Figura 3.1, na qual a integração entre os sistemas descritos pode ser observada.

O desenvolvimento desta proposta consistiu nas etapas de projeto, implementação e testes em simulação e no robô real. Foram considerados aspectos teóricos e práticos das áreas da robótica móvel, como metodologias de controle de velocidade, posição e desvio de obstáculos.

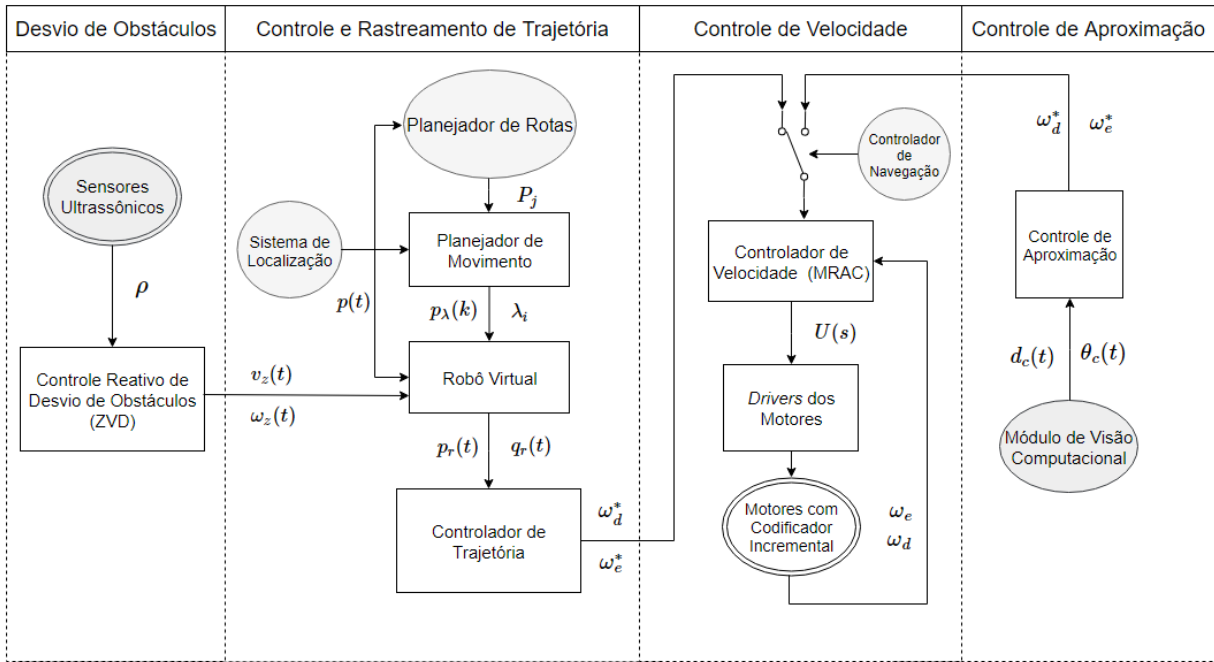


Figura 3.1: Diagrama do controlador de movimento.

3.2 Modelagem

Este trabalho aborda o desenvolvimento de um controlador de movimento de um robô não holonômico com tração diferencial em um ambiente aberto ou *outdoor*. O projeto desse controlador foi baseado no modelo cinemático de um robô diferencial. Sendo assim, essa seção irá descrever a modelagem cinemática de um robô diferencial, como o exibido na Figura 3.2.

O robô no qual foi implementado este controlador pode ser visualizado na Figura 1.3. Apesar do veículo real de testes possuir seis rodas, a modelagem do *hardware* utilizado foi simplificada para a de um robô diferencial com duas rodas, como o exibido na Figura 3.2, pois o robô utilizado continha contadores incrementais apenas nas rodas do meio de cada lado, ou seja, só era possível medir a velocidade de rotação de duas das seis rodas.

A posição do robô em relação aos eixos do sistema de coordenadas inercial (denominado neste trabalho como mundo), a cada instante de tempo t , pode ser descrita por um vetor

$$\mathbf{p}(t) = [x(t), y(t), \theta(t)]^T, \quad (3.1)$$

em que o ponto $(x(t), y(t)) \in \mathbb{R}^2$ é a posição do centro de massa do robô no instante t em relação ao mundo e o ângulo $\theta(t) \in [-\pi, \pi]$ é o ângulo entre o vetor velocidade linear e o eixo horizontal do sistema de coordenadas inercial.

As velocidades do robô em relação aos eixos do sistema de coordenadas inercial serão dadas pelo vetor $\dot{\mathbf{p}}(t)$, definido como

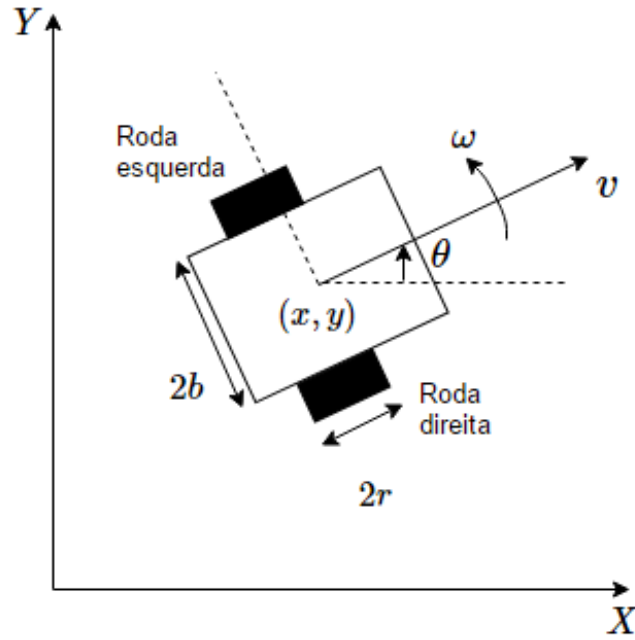


Figura 3.2: Modelo de um robô diferencial.

$$\dot{\mathbf{p}}(t) = [\dot{x}(t), \dot{y}(t), \dot{\theta}(t)]^T. \quad (3.2)$$

O vetor $\mathbf{q}(t) = [v(t), \omega(t)]^T$ que caracteriza as velocidades linear $v(t)$ e a angular $\omega(t)$ do robô se relaciona com a posição do robô $\mathbf{p}(t)$ de acordo com

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos \theta & 0 \\ \sin \theta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix}. \quad (3.3)$$

E as velocidades rotacionais das rodas direita ω_d e esquerda ω_e , podem ser obtidas com a relação

$$\begin{bmatrix} \omega_d \\ \omega_e \end{bmatrix} = \begin{bmatrix} \frac{1}{r} & \frac{b}{r} \\ \frac{1}{r} & -\frac{b}{r} \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix}, \quad (3.4)$$

em que os parâmetros r e b representam, respectivamente, o raio da roda e a metade da largura do robô.

3.3 Projeto do controlador de velocidade

Esta seção irá descrever o desenvolvimento da malha de controle de velocidade, que é encarregada de fazer com que o robô permaneça na velocidade de referência, possibilitando uma boa execução da trajetória. Este controlador, implementado em um Arduino embarcado no robô, recebe velocidades de referência para as rodas direita e esquerda e é realimentado a partir das leituras dos codificadores incrementais conectados ao Arduino, as quais são utilizadas para o cálculo das velocidades do robô.

Para essa malha, foi escolhido o método de controle adaptativo por modelo de referência. A ideia básica é escolher um modelo de referência e fazer com que o sistema se comporte como este modelo, em malha fechada. Controladores baseados nessa estratégia costumam apresentar um bom comportamento em regime permanente, com um sinal de controle suave. Porém, costumam apresentar transitório lento e oscilatório [7].

3.3.1 Cálculo das velocidades angulares das rodas

O acionamento dos motores se dá a partir de um atuador, que recebe sinais de comando do controlador de dois tipos: um sinal modulado em largura de pulso, do inglês *Pulse Width Modulation* (PWM), e sinais de polaridade. O *driver* trata esses sinais e os aplica aos motores, que são de corrente contínua. De modo que, a tensão média que será enviada aos motores depende em módulo dos sinais PWM; e os sinais de polaridade determinam o sentido em que os motores irão girar.

Os sinais PWM recebidos pelo circuito de acionamento variam entre 0 e 255, sendo que, 0 indica que 0% da tensão da bateria será transmitida aos terminais dos motores e 255 indica que 100% da tensão fornecida pela bateria será transmitida aos terminais dos motores. O movimento dos motores é transferido para as rodas por meio de caixas de redução, com uma razão 34,014:1.

No entanto, o circuito de acionamento dos motores apresenta uma característica não-linear, que pode ser constatada na tensão média aplicada, que deveria ser proporcional ao sinal PWM recebido do controlador. Porém, devido a quedas de tensão nos dispositivos semicondutores dos circuitos de acionamento, observa-se uma tensão média nos terminais dos motores diferente de zero apenas para um sinal PWM maior que uma determinada zona morta ou *dead zone*.

O robô utilizado para a implementação deste trabalho possui 1 motor esquerdo e 1 motor direito com codificadores incrementais. Estes permitem que a velocidade rotacional ω da roda seja calculada. Esse cálculo é realizado pelas equações

$$q = \frac{p}{1632,672}, \quad (3.5)$$

$$\omega(t) = \frac{\Delta q}{\Delta t} (rps), \quad (3.6)$$

em que p representa o número de pulsos contados pelo codificador incremental em um intervalo de

tempo Δt , dado em segundos; e q indica o número de rotações que a roda executou nesse intervalo de tempo.

Para a realização dos projetos dos controladores, as referências de velocidade de rotação no domínio de Laplace são representadas por $\Omega^*(s) = \mathcal{L}(\omega^*(t))$, as velocidades calculadas das rodas são indicadas por $\Omega(s) = \mathcal{L}(\omega(t))$ e as entradas de tensão dos motores são representadas por $U(s) = \mathcal{L}(u(t))$. Além disso, os índices e e d subscritos nas variáveis estão relacionados, respectivamente, com os motores da esquerda e da direita.

3.3.2 Identificação do sistema

Para realizar a identificação dos sistemas de propulsão, estes foram aproximados por modelos lineares. Entretanto, sabe-se que os semicondutores presentes nos circuitos de acionamento e o alto coeficiente de atrito estático das caixas de redução tornam esses sistemas não-lineares.

Um experimento foi realizado para a obtenção das curvas de velocidade de rotação $\Omega_d(s)$ e $\Omega_e(s)$, das rodas direita e esquerda, em função da tensão de entrada do motor, $U(s)$, fornecida pelo circuito de acionamento.

O experimento consistiu em enviar um sinal PWM constante igual a 60 e obter suas saídas correspondentes $\omega_d(t)$ e $\omega_e(t)$, calculadas como exposto na Subseção 3.3.1. Para obter a estimação do modelo de primeira ordem, com uma função de transferência de acordo com

$$\Omega(s) = \frac{K_m}{\tau s + 1} U(s), \quad (3.7)$$

na qual, os parâmetros de ganho, K_m , e constante de tempo, τ , foram calculados com o auxílio de um programa feito em MATLAB. De modo que τ foi definida como o tempo que o sistema levava para atingir 63,21% do valor em regime permanente. E o ganho K_m foi determinado como a relação entre a tensão de entrada e a velocidade da roda em regime permanente.

Os modelos contínuos estimados de primeira ordem $G_e(s)$ e $G_d(s)$, para os motores da direita e da esquerda, em relação ao sinal PWM enviado para o circuito de acionamento, são

$$G_e(s) = \frac{\Omega_e(s)}{U_e(s)} = \frac{14,36}{s + 2,865}, \quad (3.8)$$

$$G_d(s) = \frac{\Omega_d(s)}{U_d(s)} = \frac{14,94}{s + 2,865}. \quad (3.9)$$

Após realizada a discretização dos modelos estimados, feita segundo a equação [26]

$$G(z) = (1 - z^{-1}) \mathcal{Z} \left(\frac{G(s)}{s} \right), \quad (3.10)$$

obteve-se:

$$G_e(z) = \frac{\Omega_e(z)}{U_e(z)} = \frac{0,4118}{z - 0,91764}, \quad (3.11)$$

$$G_d(z) = \frac{\Omega_d(z)}{U_d(z)} = \frac{0,42975}{z - 0,91764}, \quad (3.12)$$

com o tempo de amostragem $T = 0,03$ segundos.

3.3.3 Controlador adaptativo por modelo de referência

Para o projeto dos controladores de velocidade adaptativo por modelo de referência, ou MRAC, foi utilizado como o modelo de referência o sistema descrito por

$$G_m(s) = \frac{\Omega_m(s)}{\Omega^*(s)} = \frac{200}{s + 200}, \quad (3.13)$$

ou seja, com $a_m = b_m = -200$. Esses parâmetros foram obtidos após alguns experimentos com o robô, por apresentarem o melhor comportamento de transitório para o MRAC.

Após discretizar esse modelo de referência, através da equação (3.10), obteve-se

$$G_m(z) = \frac{\Omega_m(z)}{\Omega^*(z)} = \frac{1 - e^{-200T}}{z - e^{-200T}}. \quad (3.14)$$

Considerando que o tempo de discretização utilizado foi de $T = 30ms$, de acordo com o implementado em *software*, obteve-se a equação de diferenças para o modelo de referência, dada por

$$\omega_m(k) = 0,00248\omega_m(k + 1) + 0,99752\omega^*(k - 1), \quad (3.15)$$

com o parâmetro ω^* podendo ser substituído por ω_e^* ou ω_d^* quando conveniente.

As regras de adaptação dos parâmetros dos controladores (θ_i) dadas pelas equações (2.16) e (2.17) foram discretizadas por meio da aproximação de Euler de primeira ordem para derivadas. Assim, para os parâmetros de adaptação, obteve-se as equações de diferenças

$$\theta_1(k) = \theta_1(k - 1) - T_v \gamma \omega^*(k) (\omega(k) - \omega_m(k)), \quad (3.16)$$

$$\theta_2(k) = \theta_2(k - 1) - T_v \gamma \omega(k) (\omega(k) - \omega_m(k)), \quad (3.17)$$

com as variáveis podendo ser devidamente substituídas de acordo com o motor a ser controlado.

Os ganhos de adaptação utilizados obedecem as relações

$$\gamma_e T_v = 0,08, \quad (3.18)$$

$$\gamma_d T_v = 0,07, \quad (3.19)$$

em que o ganho de adaptação γ foi escolhido através de análise experimental.

As leis de controle utilizadas são dadas pela equação (2.8). Desse modo, as leis de controle para os controladores de velocidade dos motores da esquerda e da direita são, respectivamente

$$u_e(k) = \theta_{1e}(k)\omega_e^*(k) - \theta_{2e}(k)\omega_e(k), \quad (3.20)$$

$$u_d(k) = \theta_{1d}(k)\omega_d^*(k) - \theta_{2d}(k)\omega_d(k), \quad (3.21)$$

com $\theta_{1e}(0) = 0,50$, $\theta_{2e}(0) = 0,25$, $\theta_{1d}(0) = 0,40$ e $\theta_{2d}(0) = 0,15$.

A Figura 3.3 exibe o diagrama de blocos do controlador de velocidade adaptativo por modelo de referência desenvolvido neste trabalho, que representa tanto a estrutura do controlador do motor direto quanto a do motor esquerdo.

3.4 Projeto do controlador de trajetória

A Figura 3.4 exibe, de forma simplificada, a arquitetura de controle e rastreamento de trajetória adotada neste trabalho. O Sistema de Localização, o Planejador de Rotas, o Desvio de Obstáculos e o Controlador de Velocidade, que também podem ser visualizados na imagem, são sistemas complementares ao controle de trajetória. Já que, para que o veículo execute a trajetória conforme o planejado é necessário, além de um controlador de trajetória, um bom sistema de localização, um bom controle de velocidade, um sistema para desviar de obstáculos e um planejador de rotas, que irá determinar os pontos de passagem P_j para que o robô chegue ao destino de interesse.

Para a elaboração do controle de movimento, optou-se por um rastreamento de trajetórias baseado na abordagem de segmentos de caminho. Desse modo, os caminhos a serem seguidos são de mais fácil execução para um robô não-holonômico e o custo computacional da implementação não é excessivo.

Em relação ao controlador proposto, este se baseia no modelo cinemático de um robô diferencial, como o descrito na Seção 3.2. Para tanto, é fundamental que o controlador de velocidade implementado funcione de maneira adequada. Dado que, o controlador de trajetória calcula velocidades de referência para as rodas direita ω_d e esquerda ω_e , com as quais o robô irá se manter no caminho desejado.

3.4.1 Planejador de movimento

Na arquitetura proposta para o robô, que pode ser visualizada na Figura 1.4, o planejador de rotas calcula os pontos da rota P_j a partir da localização atual do robô até o destino de interesse.

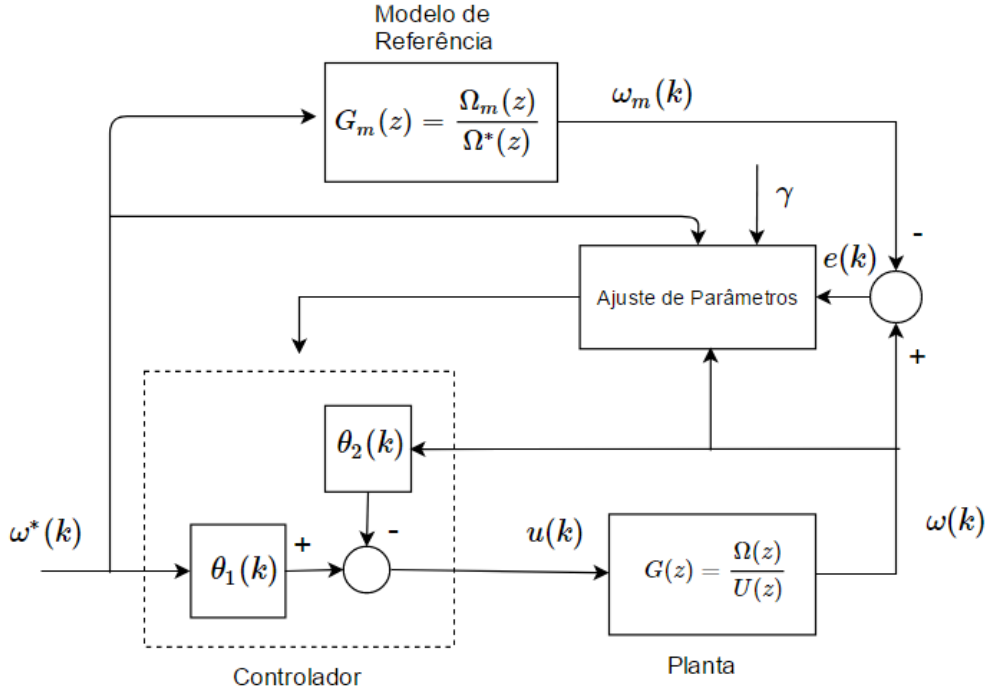


Figura 3.3: Diagrama de blocos do controlador de velocidade MRAC.

O planejador de movimento desenvolvido neste trabalho recebe esse conjunto de pontos e calcula os segmentos de caminho da trajetória, na forma de segmentos de reta.

A Figura 3.5 ilustra a situação em que o robô se encontra na posição $\mathbf{p} = [x_0, y_0, \theta_0]^T$, sendo $P_0 = (x_0, y_0)$, e tem como próximo destino o ponto $P_f = (x_f, y_f)$. Nesse caso, o planejador de movimento calcula a parametrização do segmento reta que liga P_0 à P_f .

Inicialmente, é obtida a inclinação ϕ do segmento de reta λ_1 , através da equação (3.24), com $\phi \in [-\pi, \pi]$. Em seguida, a partir da equação (3.25), é calculado o $\Delta\theta \in [-\pi, \pi]$, que determina o deslocamento angular que o robô deverá fazer para que ele fique paralelo ao segmento de reta λ_1 . São calculados também Δx e Δy , que representam, respectivamente, os deslocamentos no eixo horizontal e vertical que o robô deverá fazer para chegar ao ponto P_f , conforme abaixo:

$$\Delta x = x_f - x_0 \quad (3.22)$$

$$\Delta y = y_f - y_0 \quad (3.23)$$

$$\phi = \arctan 2(\Delta y, \Delta x) \quad (3.24)$$

$$\Delta\theta = \phi - \theta_0 \quad (3.25)$$

A parametrização do segmento de reta λ_1 é obtida através do cálculo das amostras do vetor de posições $\mathbf{p}_\lambda(k) = [x(k), y(k), \theta(k)]^T$. Ela é realizada de modo que entre duas posições consecutivas, $\mathbf{p}_\lambda(k)$ e $\mathbf{p}_\lambda(k+1)$, o intervalo de tempo seja constante e igual ao tempo de amostragem, $T = 0.1s$. E os deslocamentos em x , y e θ também sejam constantes e iguais a: x_i , y_i e θ_i .

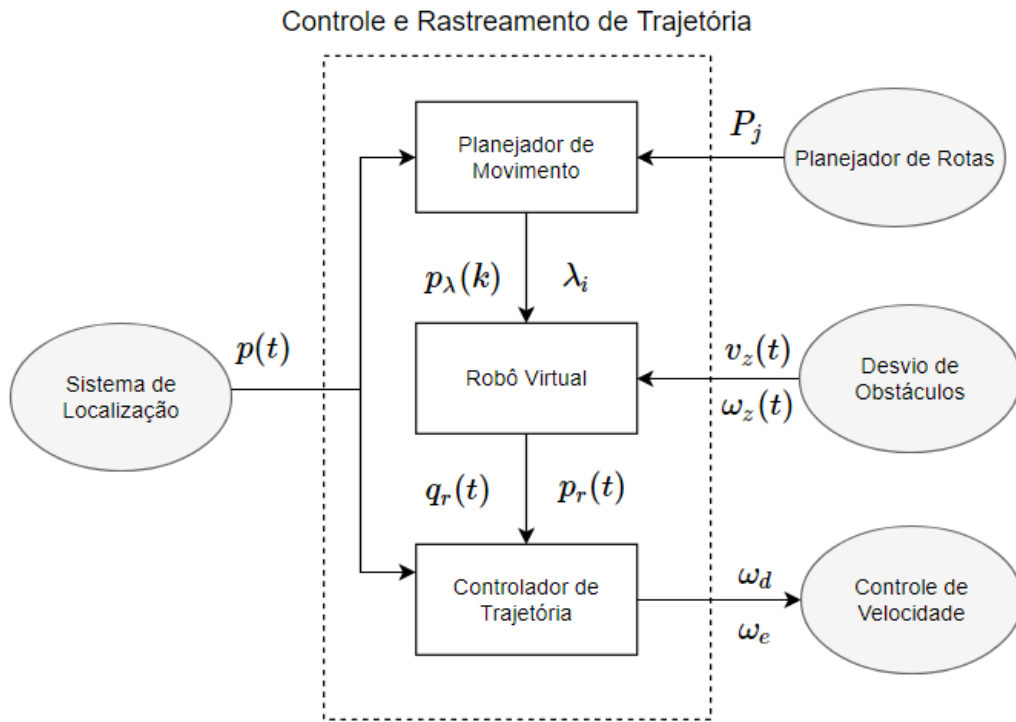


Figura 3.4: Diagrama do Controle e Rastreamento de Trajetória.

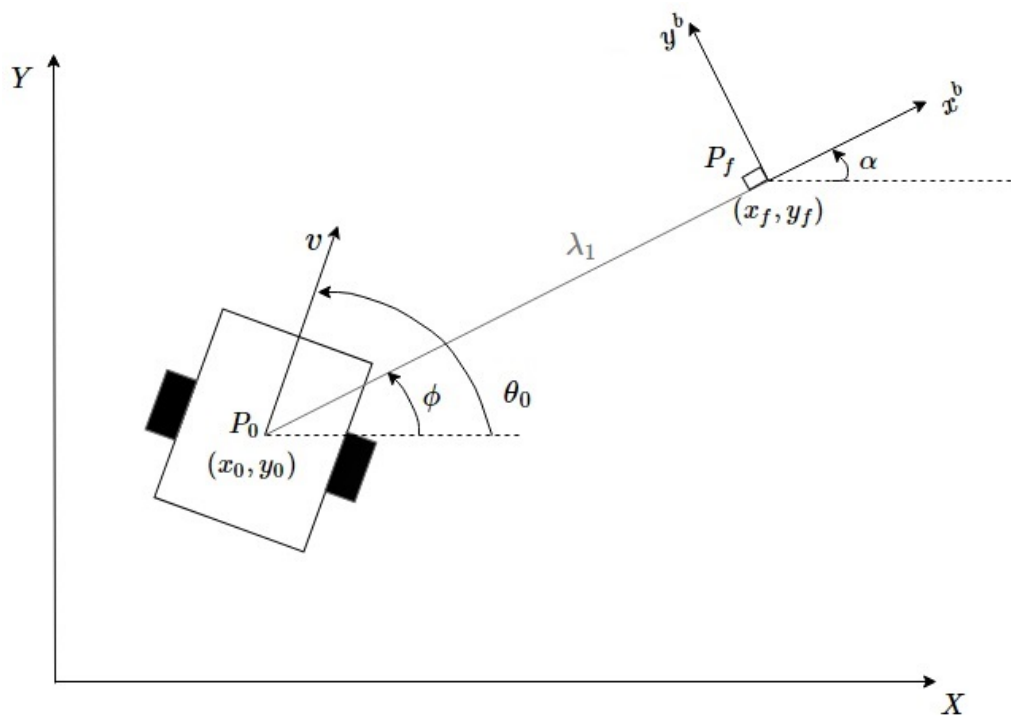


Figura 3.5: Representação de um segmento de caminho.

O número de amostras n de $\mathbf{p}_\lambda(k)$ é calculado de acordo com o intervalo de tempo Δt determinado para a execução do segmento de trajetória λ_1 , com a equação

$$n = \frac{\Delta t}{T}. \quad (3.26)$$

Os incrementos x_i , y_i e θ_i adicionados a cada amostra de $\mathbf{p}_\lambda(k)$, são calculados com as equações

$$x_i = \frac{\Delta x}{n}, \quad (3.27)$$

$$y_i = \frac{\Delta y}{n}, \quad (3.28)$$

$$\theta_i = \frac{\Delta \theta}{n}. \quad (3.29)$$

Assim, a parametrização de um segmento de caminho λ_i em relação a x , y e θ fica

$$\mathbf{p}_\lambda(k) = \begin{cases} x_k = x_0 + (kT x_i) \\ y_k = y_0 + (kT y_i) \\ \theta_k = \theta_0 + (kT \theta_i), \end{cases} \quad (3.30)$$

com $k \in [0, 1, \dots, n]$.

Além de obter a parametrização dada pela equação (3.30), o planejador de movimento determina se o robô ainda está seguindo o segmento λ_1 ou não, através da adaptação das equações (2.23) (2.24) para o sistema analisado, obtendo-se

$$\alpha = \arctan 2(y_b - y_0, x_b - x_0), \quad (3.31)$$

$$\cos(\alpha)(x - x_f) + \sin(\alpha)(y - y_f) < 0. \quad (3.32)$$

Assim, o robô estará seguindo o segmento de trajetória λ_1 , caso a condição determinada em (3.32) seja válida, sendo (x, y) a posição atual do centro do robô. Pois, nesse caso, a posição do robô transformada para o eixo de coordenadas $x^b y^b$ será positiva em relação ao eixo x^b . Logo, é considerado que o robô já seguiu o segmento λ_1 e poderá começar a seguir o próximo segmento de reta.

3.4.2 Controlador de trajetória por modelo de referência

O controlador de trajetória proposto neste trabalho é baseado no modelo cinemático de um robô com tração diferencial, como o descrito na Seção 3.2. O controlador sugerido utiliza um modelo de referência, ou robô virtual, que executa a trajetória de maneira ideal, para assegurar que o robô real prossiga na trajetória ao se movimentar seguindo o robô virtual.

O modelo de referência é um robô virtual, com o mesmo modelo cinemático do robô real, como o ilustrado na Figura 3.6. Esse modelo é caracterizado pelo vetor $\mathbf{p}_r = [x_r, y_r, \theta_r]^T$, no qual o ponto $(x_r, y_r) \in \mathbb{R}^2$ é a posição do robô virtual no sistema de coordenadas inercial, ou mundo, e $\theta_r \in [-\pi, \pi]$ é a orientação do modelo de referência em relação ao mundo. Além disso, o vetor $\mathbf{q}_r(t) = [v_r(t), \omega_r(t)]^T$ contém as velocidades linear $v_r(t)$ e angular $\omega_r(t)$ do modelo de referência.

O controlador recebe a posição e as velocidades do robô virtual, e, a partir do erro entre a posição do robô real $\mathbf{p}(\mathbf{t})$ e a posição do robô virtual $\mathbf{p}_r(t)$, fornece o vetor velocidade $\mathbf{q}_t(t) = [v_t(t), \omega_t(t)]^T$ para que o robô real alcance o modelo de referência. Logo, o objetivo do controlador de trajetória é que

$$\lim_{t \rightarrow \infty} (\mathbf{p}(t) - \mathbf{p}_r(t)) = 0. \quad (3.33)$$

Para tanto, é necessário garantir que o modelo de referência irá seguir os segmentos de caminho calculados pelo planejador de movimento, como descrito na Subseção 3.4.1. Assim, a posição e as velocidades do robô virtual são obtidos a partir da parametrização do segmento de caminho. A posição do modelo de referência a cada intervalo de amostragem é dada pela equação (3.30). Desse modo, \mathbf{p}_r é atualizado a cada instante de amostragem, com o intervalo de $T = 0,1s$.

Para a obtenção de \mathbf{q}_r , inicialmente é calculado o intervalo de tempo Δt necessário para a execução de um segmento de trajetória λ_i . Esse intervalo de tempo é calculado considerando os limites de velocidade do robô: Vx_{max} , Vy_{max} e $\dot{\theta}_{max}$.

Durante a execução de um segmento de caminho λ_i pelo robô virtual, ocorrem os deslocamentos em x , y e θ entre a posição inicial e o ponto final do segmento, representados por: Δx , Δy e $\Delta \theta$. O menor intervalo de tempo para a execução de cada um desses deslocamentos é calculado como

$$\Delta t_x = \frac{\Delta x}{Vx_{max}}, \quad (3.34)$$

$$\Delta t_y = \frac{\Delta y}{Vy_{max}}, \quad (3.35)$$

$$\Delta t_\theta = \frac{\Delta \theta}{\dot{\theta}_{max}}, \quad (3.36)$$

sendo Δt_x , Δt_y e Δt_θ , respectivamente, os menores intervalos de tempo para a execução do segmento de trajetória em x , y e θ . O maior intervalo de tempo obtido entre eles é considerado como o tempo máximo para a execução do segmento de trajetória, representado por Δt .

Em seguida, são calculadas as velocidades \dot{x}_r , \dot{y}_r e $\dot{\theta}_r$, através das equações

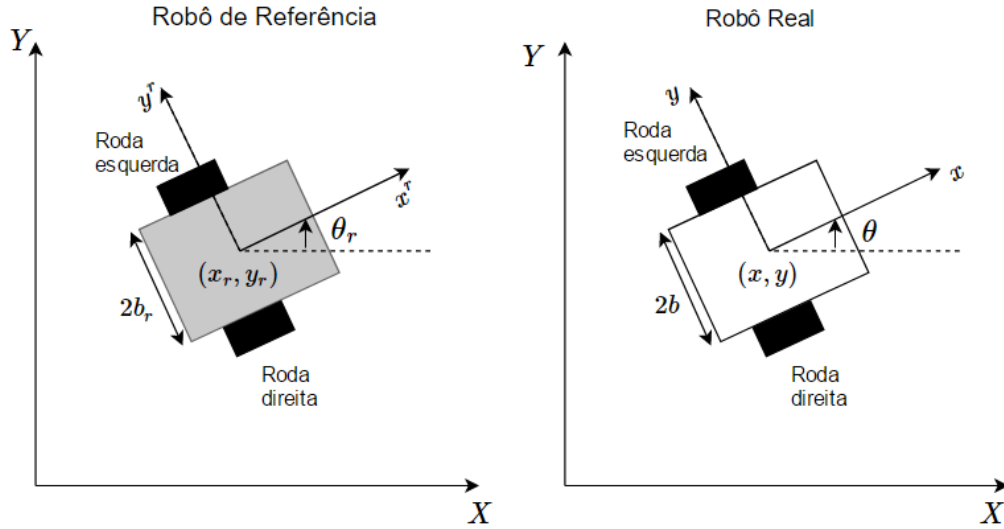


Figura 3.6: Modelo de referência e modelo do robô real.

$$\dot{x}_r = \frac{\Delta x}{\Delta t} (m/s), \quad (3.37)$$

$$\dot{y}_r = \frac{\Delta y}{\Delta t} (m/s), \quad (3.38)$$

$$\dot{\theta}_r = \frac{\Delta \theta}{\Delta t} (rad/s). \quad (3.39)$$

Essas velocidades do robô virtual são mantidas constantes durante toda a execução do segmento de trajetória λ_i , caso não sejam detectados obstáculos no caminho.

O vetor de velocidades $\mathbf{q}_r = [v_r, \omega_r]^T$ também será constante durante a execução de um segmento de trajetória λ_i , na ausência de obstáculos, e é obtido através de

$$v_r = \cos(\theta_r)\dot{x}_r + \sin(\theta_r)\dot{y}_r, \quad (3.40)$$

$$\omega_r = \dot{\theta}_r. \quad (3.41)$$

A diferença de posição entre o robô real e o modelo de referência é obtida pela equação (2.29), que gera os valores dos erros e_1 , e_2 e e_3 . Estes representam, respectivamente, a diferença de posição no eixo horizontal, no eixo vertical e a diferença na orientação. Os erros são calculados pelo controlador a um tempo de amostragem de $T = 0,05s$, como

$$e_1[h] = \cos(\theta[h])(x_r[h] - x[h]) + \sin(\theta[h])(y_r[h] - y[h]), \quad (3.42)$$

$$e_2[h] = -\sin(\theta[h])(x_r[h] - x[h]) + \cos(\theta[h])(y_r[h] - y[h]), \quad (3.43)$$

$$e_3[h] = \theta_r[h] - \theta[h]. \quad (3.44)$$

As leis de controle dadas pelas equações (2.30) e (2.31) fornecem as velocidades linear v_t e angular ω_t para que os erros sejam minimizados durante a execução da trajetória. O controlador calcula as leis de controle a um tempo de amostragem de $T = 0,05s$, que, na ausência de obstáculos, são dadas por

$$v_t[h] = v_r[h] \cos(e_3[h]) + 1,2e_1[h], \quad (3.45)$$

$$w_t[h] = w_r[h] + 1,3v_r[h]e_2[h] + 1,2 \sin(e_3[h]). \quad (3.46)$$

com $K_1 = 1,2$, $K_2 = 1,3$ e $K_3 = 1,2$. Deve-se destacar que estes parâmetros foram escolhidos através de análises experimentais com o robô real.

As velocidades de rotação para o motor direito ω_d e para o motor esquerdo ω_e são calculadas através da equação (2.38). Os parâmetros a_1 e a_2 para o robô real são

$$a_1 = \frac{1}{r} = \frac{1}{0,06}, \quad (3.47)$$

$$a_2 = \frac{b}{r} = \frac{0,075}{0,06}, \quad (3.48)$$

com $r = 0,06m$, $b = 0,075m$ e $a_2 \geq 1,25$.

Logo, as velocidades de rotação das rodas em (rad/s), e positivas para no sentido anti-horário, são dadas por

$$\omega_d[h] = a_1v_t[h] + a_2\omega_t[h], \quad (3.49)$$

$$\omega_e[h] = a_1v_t[h] - a_2\omega_t[h]. \quad (3.50)$$

Esse sistema de controle irá garantir que o robô chegue no ponto de destino. Porém, a orientação que o robô estaria ao chegar a esse ponto, seria a inclinação do último segmento de caminho seguido.

Para resolver esse problema e permitir que a orientação final do robô seja escolhida previamente, o controlador calcula um ponto intermediário $P_{int} = (x_{int}, y_{int})$, fazendo com que a inclinação do último segmento a ser seguido seja a mesma desejada para o robô.

Essa situação é ilustrada na Figura 3.7, na qual a orientação final de referência é representada por θ_f . E λ_f indica o último segmento de caminho gerado através dos pontos dados pelo planejador. Para que o robô chegue ao ponto P_f com uma orientação θ_f , é inserido o ponto $P_{int} = (x_{int}, y_{int})$, que é calculado como

$$x_{int} = x_f - \|\lambda_{int}\| \cos(\theta_f), \quad (3.51)$$

$$y_{int} = y_f - \|\lambda_{int}\| \sin(\theta_f), \quad (3.52)$$

com $\|\lambda_{int}\| = 1m$.

Assim, ao final da trajetória, em vez de o robô seguir o segmento de caminho λ_f , ele irá calcular um segmento até o ponto P_{int} e chegará ao ponto de destino final P_f executando o segmento λ_{int} .

3.5 Modelo da zona virtual deformável (ZVD)

O propósito deste trabalho é fazer com que um robô móvel navegue de forma autônoma e segura em um ambiente dinâmico, parcialmente desconhecido.

O sistema de controle proposto utiliza duas malhas de controle principais. Um controlador de trajetória, que é encarregado de fazer com que o robô siga a trajetória especificada pelo planejador de rotas. E um controlador de velocidade, que é responsável por manter os sistemas de propulsão na velocidade calculada pelo controlador de trajetória. Mesmo assim, ainda é necessário garantir que o robô não colida com obstáculos, que podem surgir durante a execução da trajetória.

Para lidar com os possíveis obstáculos, é proposto um sistema de controle que se baseia na definição de uma zona de proteção deformável ao redor do robô. Essa zona de proteção, chamada Zona Virtual Deformável, ou ZVD, baseia-se na interação robô-ambiente sem nenhum modelo prévio dos obstáculos. O método utilizado consiste na obtenção das deformações na ZVD, realizada através de sensores ultrassônicos devido à presença de obstáculos, e na ação de controle que tenta minimizá-los.

A ZVD pode ser de diferentes formas geométricas, mas a forma elíptica é geralmente a utilizada, sendo a forma escolhida para este trabalho. O centro de referência polar está localizado no centro do robô, como exibido na Figura 3.8. Considerando que o robô possui sensores ultrassônicos apenas na parte frontal, dispostos em forma de arco, a ZVD proposta é dividida em duas partes: a porção em branco na imagem, coberta pelos sensores; e a em azul, fora do alcance dos sensores.

Como descrito na Subseção 2.5.4, a Zona Virtual Deformável sem deformações é representada pelo vetor

$$\mathbf{s} = [s_0, s_1, \dots, s_n]^T, \quad (3.53)$$

no qual cada componente s_i representa a distância entre o limite da ZVD e o centro de referência, localizado na origem do centro de coordenadas fixo ao corpo do robô.

Quando o robô se aproxima de um obstáculo e este entra no alcance do sensor, as leituras dos sensores ultrassônicos interferem na ZVD na forma de variações nos componentes s_i e geram uma deformação. O vetor que representa a ZVD deformada é dado por

$$\mathbf{s}_D = [s_{D0}, s_{D1}, \dots, s_{Dn}]^T, \quad (3.54)$$

no qual cada componente s_{Di} indica a variação da medida fornecida pelos sensores na ZVD.

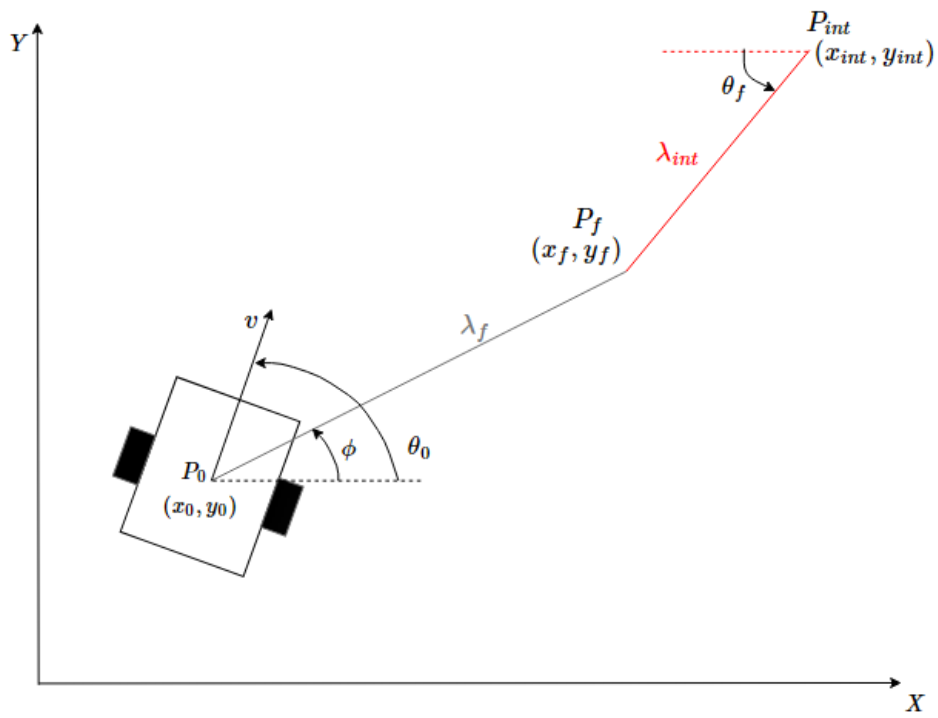


Figura 3.7: Segmento intermediário para a correção da orientação.

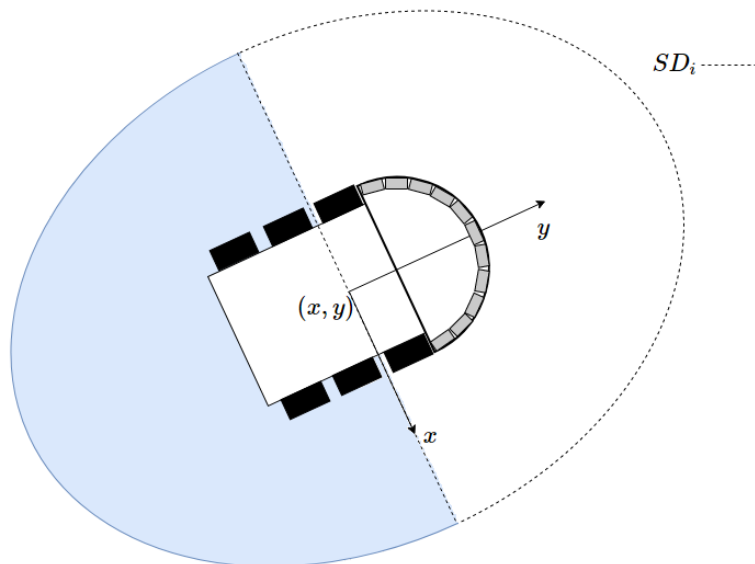


Figura 3.8: Representação da Zona Virtual Deformável.

3.5.1 Sensores ultrassônicos

A Figura 3.9 ilustra a disposição dos sensores ultrassônicos existentes no robô. O campo de visão dos sensores ultrassônicos é dividido em 11 setores ou zonas, sendo todos os 11 sensores utilizados do mesmo modelo, módulo sensor ultrassônico HC-SR04, exibido na Figura 3.10. Eles são caracterizados por possuírem um ângulo de leitura φ de até 15 graus e identificarem distâncias entre 2 e 500 cm¹. Para uma melhor visualização do problema, a Figura 3.12 exhibe o desenho do robô real, onde pode ser observado o arco que sustenta os suportes dos sensores ultrassônicos.

A leitura dos sensores ultrassônicos é fornecida pelo módulo executivo do robô, que calcula o vetor $\rho = [\rho_0, \rho_1, \rho_2, \dots, \rho_{10}]^T$ em relação ao centro do arco no qual os sensores estão fixados. E cada componente ρ_i representa a leitura de distância do sensor i . É considerado que essas leituras são realizadas no centro de cada sensor e igualmente espaçadas em um ângulo ψ , como exibido na Figura 3.9.

O controlador recebe as leituras e as transforma para o vetor $\mathbf{d} = [d_0, d_1, d_2, \dots, d_{10}]^T$. Cada componente d_i representa um valor medido pelo sensor ultrassônico i em relação à origem do eixo de coordenadas fixo ao robô xy . E para cada d_i está associado o ângulo σ_i onde está posicionado o centro do sensor em relação ao eixo de coordenadas xy .

Para o cálculo do vetor \mathbf{d} , foram utilizados os parâmetros: $l = 17,5$ cm, que representa o raio do arco, $h = 21$ cm, que indica a metade do comprimento do robô e $\psi = 17,4^\circ$, que é o ângulo entre dois sensores, em relação ao centro do arco. A partir de cada ρ_i é gerado um valor d_{xi} , que é a distância lida pelo sensor em relação ao eixo x do robô, e um valor d_{yi} , que representa a medida em relação ao eixo y fixado no centro do robô, como pode ser visualizado na Figura 3.11. Esses valores são obtidos com as equações (3.56) e (3.57), na qual a distância entre o centro do arco e a origem do sistema de coordenadas do robô é indicada por $D = 5$ cm.

$$d_{xi} = \rho_i \cos(\beta) \quad (3.55)$$

$$d_{yi} = \rho_i \sin(\beta) + D \quad (3.56)$$

$$d_i = \sqrt{d_{xi}^2 + d_{yi}^2} \quad (3.57)$$

Deve-se destacar que as informações obtidas pelos sensores ultrassônicos podem corresponder a medidas ruins, dependendo das circunstâncias, tais como portas abertas, quinas, ou mesmo quando o robô possui uma inclinação relativamente grande em relação ao obstáculo. Este último é um dos grandes problemas dos sonares, pois se a superfície encontrada pelo sinal ultrassônico tiver um posicionamento maior que o semi-ângulo do lóbulo de emissão do sensor, o sinal será refletido para longe do sonar e o obstáculo não será detectado. Isto se deve ao fato de que quanto mais inclinado estiver o sensor com relação ao obstáculo, menor é a amplitude do eco recebido pelo transdutor, ocasionando, assim, um maior erro na medida efetuada [27, 28].

¹Fonte: <http://www.huinfinito.com.br/sensores/469-modulo-sensor-ultrassonico-hc-sr04.html>

²Foto retirada de: <http://www.huinfinito.com.br/sensores/469-modulo-sensor-ultrassonico-hc-sr04.html>.

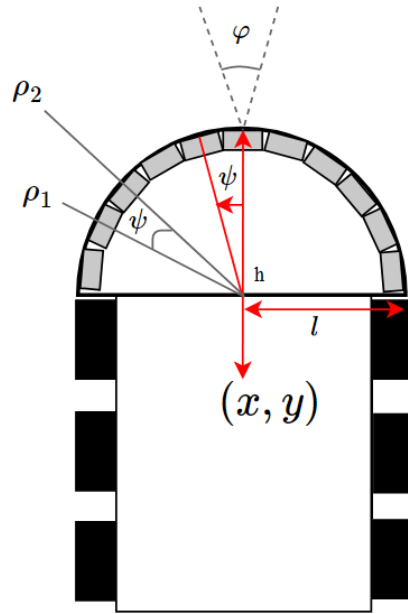


Figura 3.9: Localização dos sensores ultrassônicos no robô.



Figura 3.10: Sensor ultrassônico utilizado².

3.5.2 Deformação da zona virtual deformável

Após o cálculo do vetor \mathbf{d} com as leituras dos sensores ultrassônicos em relação ao centro do robô, este é comparado com o limite da zona virtual deformável sem deformações, ou seja, o vetor s definido na equação (3.53).

A ZVD desenvolvida neste trabalho possui 11 elementos, cada um deles é composto por um comprimento s_i localizado em um ângulo σ_i . Além disso, cada s_i possui uma componente s_{xi} , em relação ao eixo x , e uma componente s_{yi} , em relação ao eixo y , do sistema de coordenadas xy , com a origem fixada no centro do robô. Cada s_i será comparado a uma leitura d_i , de mesma orientação σ_i , obtida pelo sensor ultrassônico i , com $i \in [0, 1, ..10]$. Assim, é obtido o vetor de deformação Δ e as suas componentes no eixo x e no eixo y , através da adaptação da equação (2.44):

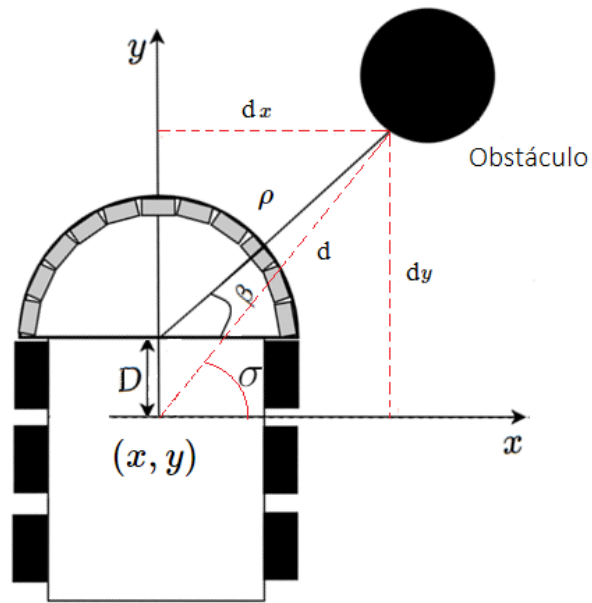


Figura 3.11: Detecção de um obstáculo pelos sensores ultrassônicos.

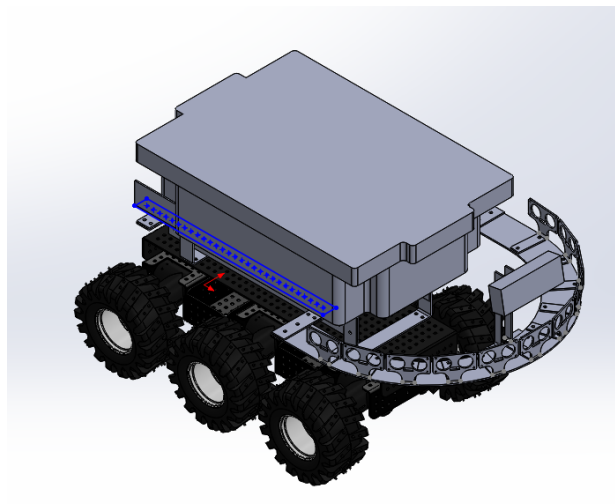


Figura 3.12: Modelo do robô real.

$$\Delta_i = \begin{cases} 0, & \text{se } d_i \geq s_i \\ s_i - d_i, & \text{se } d_i < s_i, \end{cases} \quad (3.58)$$

$$\Delta_{xi} = \begin{cases} 0, & \text{se } d_{xi} \geq s_{xi} \\ s_{xi} - d_{xi}, & \text{se } d_{xi} < s_{xi}, \end{cases} \quad (3.59)$$

$$\Delta_{yi} = \begin{cases} 0, & \text{se } d_{yi} \geq s_{yi} \\ s_{yi} - d_{yi}, & \text{se } d_{yi} < s_{yi}. \end{cases} \quad (3.60)$$

Esse vetor representa a profundidade da deformação da ZVD. O somatório dos valores Δ_i representa a deformação total Δ_T , calculada através da equação (3.61). O somatório das deformações em relação ao eixo x e ao eixo y são representados, respectivamente, por Δ_{Tx} e Δ_{Ty} .

$$\Delta_T = \sum_{i=0}^n \Delta_i \quad (3.61)$$

$$\Delta_{Tx} = \sum_{i=0}^n \Delta_{xi} \quad (3.62)$$

$$\Delta_{Ty} = \sum_{i=0}^n \Delta_{yi} \quad (3.63)$$

3.5.3 Ação de controle para evitar o obstáculo

A ação de controle gerada a partir do método da zona virtual deformável tem o intuito de evitar que o robô móvel colida com obstáculos, que alterem a forma da ZVD e possam representar um perigo para a integridade física do robô. Entretanto, a ZVD proposta neste trabalho atua apenas na parte frontal do robô, devido a limitações de *hardware*, pois o robô possui sensores ultrassônicos apenas nessa região.

Na zona de atuação da ZVD é criado um vetor $\mathbf{f} = [f_0, f_2, \dots, f_{10}]$. No qual cada f_i está atrelado ao sensor ultrassônico i e à orientação σ_i correspondente, podendo ser dividido em duas componentes: \mathbf{f}_{xi} , em relação ao eixo x do robô, e \mathbf{f}_{yi} , em relação ao eixo y do robô. As componentes $f_i[h]$ são calculadas pelo controlador a cada intervalo de amostragem, com um período $T = 0,05s$, como

$$f_i[h] = \begin{cases} 0, & \text{se } \Delta_i[h] - \Delta_i[h-1] < 0 \\ \Delta_i[h] - \Delta_i[h-1], & \text{se } \Delta_i[h] - \Delta_i[h-1] \geq 0, \end{cases} \quad (3.64)$$

$$f_{xi}[h] = \begin{cases} 0, & \text{se } f_i[h] = 0 \\ \Delta_{xi}[h] - \Delta_{xi}[h-1], & \text{se } f_i[h] > 0, \end{cases} \quad (3.65)$$

$$f_{yi}[h] = \begin{cases} 0, & \text{se } f_i[h] = 0 \\ \Delta_{yi}[h] - \Delta_{yi}[h-1], & \text{se } f_i[h] > 0. \end{cases} \quad (3.66)$$

É definido $f_T[h]$, que recebe o somatório de todas as componentes $f_i[h]$:

$$f_T[h] = \sum_{i=0}^{10} f_i[h]. \quad (3.67)$$

Tal que o ângulo $\sigma_T[h]$, representa a orientação resultante das deformações da ZVD. Ele é necessário para determinar a orientação em que o obstáculo se encontra em relação ao robô. E é calculado através de

$$f_{Tx}[h] = \sum_{i=0}^{10} f_{ix}[h], \quad (3.68)$$

$$f_{Ty}[h] = \sum_{i=0}^{10} f_{iy}[h], \quad (3.69)$$

$$\sigma_T[h] = \arctan 2(f_{Ty}[h], f_{Tx}[h]), \quad (3.70)$$

onde $f_{Tx}[h]$ e $f_{Ty}[h]$ representam, respectivamente, as componentes de $f_T[h]$ no eixo x e no eixo y do robô.

De acordo com [29], o vetor de velocidades $\mathbf{q}_z[h] = [v_z[h], \omega_z[h]]^T$ pode ser determinado pelas equações

$$v_z[h] = v_z[h-1] + K_t f_T[h] \cos(\sigma_T[h]) \quad (3.71)$$

$$\omega_z[h] = \omega_z[h-1] + K_r \sin(\sigma_T[h]), \quad (3.72)$$

com $K_t = -1, 2$ e $K_r = 1, 2$ constantes definidas através de análises experimentais.

Essa ação de controle tem o intuito de evitar uma colisão com o objeto que causou uma deformação na ZVD. Desse modo, ela é executada até que a deformação total, conforme calculada na equação (3.61), seja nula, ou seja, $\Delta_T = 0$.

3.6 Aproximação do ponto de interesse

Além de conseguir seguir a trajetória especificada pelo planejador sem colidir com obstáculos, para cumprir o percurso da competição *Robomagellan*, o robô deve ser capaz de, ao chegar nos pontos determinados, conseguir identificar e encostar no marco especificado, sinalizado por um cone de sinalização de trânsito laranja. O método de aproximação descrito nesta seção desconsidera o sistema de localização do robô, pois a imprecisão da posição no mundo era significativa para uma aproximação precisa do marco.

O sistema de identificação dos marcos, que funciona através de algoritmos de visão computacional, foi implementado em um trabalho complementar a este. O módulo de visão computacional fornece a distância $d_c(t)$ e a orientação $\theta_c(t) \in [-\pi/6, \pi/6]$, positiva no sentido horário, que o cone está da parte frontal do robô, em um instante de tempo t . A Figura 3.13 ilustra a situação na qual um cone é identificado.

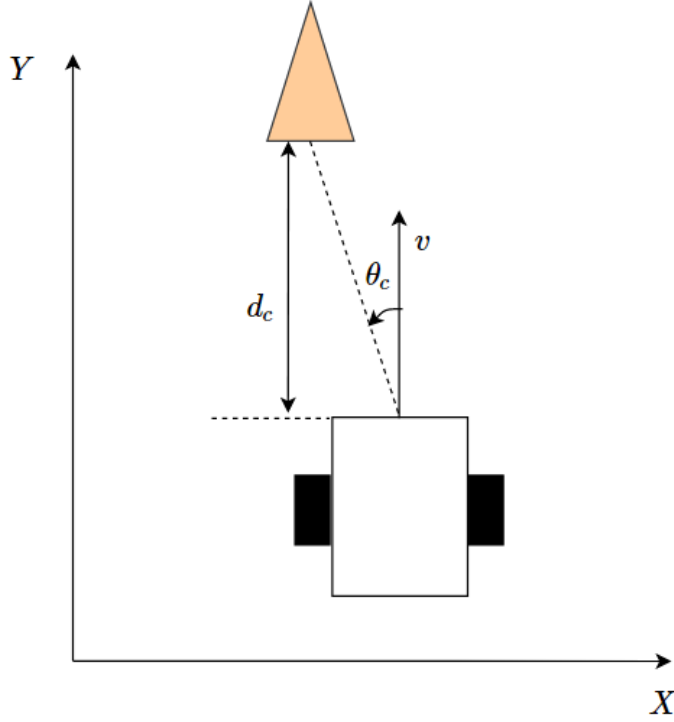


Figura 3.13: Robô identificando o cone na fase de aproximação.

A malha de controle de aproximação pode ser visualizada na Figura 3.14, o desenvolvimento dessa malha foi feito a partir da metodologia do controlador proporcional integral derivativo, utilizando apenas a ação de controle proporcional. O controlador recebe uma distância e uma orientação desejada em relação ao cone, $d_c^*(t) = 0$ e $\theta_c^*(t) = 0$. Posteriormente, calcula os erros $e_d(t) = d_c(t) - d_c^*(t)$ e $e_\theta(t) = \theta_c(t) - \theta_c^*(t)$ para obter as velocidades linear $v_c(t)$ e angular $\omega_c(t)$, que irão minimizar os erros. Esse cálculo é realizado com um período de amostragem de $T = 0,01s$ com as equações

$$v_c(t) = K_d e_d(t), \quad (3.73)$$

$$\omega_c(t) = K_\theta e_\theta(t), \quad (3.74)$$

em que $K_d = 0,7$ e $K_\theta = -15 \text{ rad}$ são constantes definidas através de avaliações experimentais com o robô. K_d foi escolhida para que o robô desacelerasse a medida que se aproximasse do cone, permitindo que o veículo tocasse no alvo de maneira suave. E K_θ foi definida de forma que o robô não alterasse abruptamente a velocidade angular e comprometesse a identificação visual do alvo.

A partir dessas velocidades são calculadas as velocidades de rotação para o motor direito $\omega_d^*(t)$ e para o motor esquerdo $\omega_e^*(t)$, através de

$$\omega_d^*(t) = a_1 v_c(t) + a_2 \omega_c(t), \quad (3.75)$$

$$\omega_e^*(t) = a_1 v_c(t) - a_2 \omega_c(t), \quad (3.76)$$

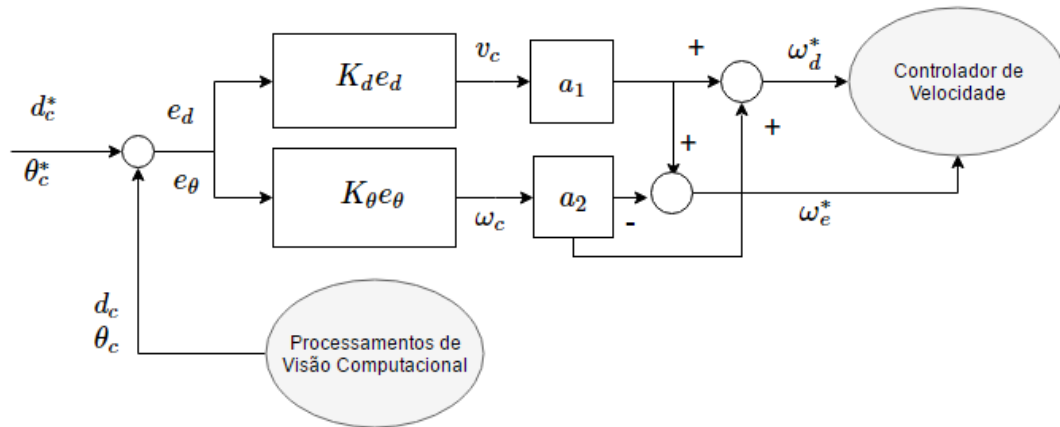


Figura 3.14: Diagrama do controlador de aproximação.

em que a_1 e a_2 são os parâmetros definidos, respectivamente, em (3.47) e (3.48). Estas velocidades de rotação serão passadas como referência para o módulo que realiza o controle de velocidade.

O objetivo é que os erros convirjam para zero e, conseqüentemente, o robô encoste no cone.

3.7 O controlador de movimento

Nas seções anteriores deste capítulo foram apresentados os módulos que coordenam a movimentação do robô, os sistemas de controle de velocidade, trajetória, desvio de obstáculos e de aproximação de um ponto de interesse. Estes interagem entre si conforme o diagrama exibido na Figura 3.1. Essa seção irá explicar como ocorre a integração desses módulos e o funcionamento geral do controlador de movimento proposto.

Deve-se destacar que, no problema abordado neste trabalho, a movimentação do robô é dividida em duas etapas principais. Inicialmente, têm-se a etapa de navegação, em que o robô deve ser capaz de navegar em um ambiente dinâmico, entre dois pontos especificados por coordenadas GPS. Durante essa etapa, para garantir a integridade física do veículo, é utilizada a malha de controle de desvios de obstáculos.

Em seguida, têm-se a etapa de aproximação fina, na qual o robô se movimenta visando encostar no cone laranja. Nesta etapa, são obtidas informações da posição e orientação do cone em relação ao robô. A partir dessas informações, o controlador de aproximação calcula as velocidades a serem aplicadas nos sistemas de propulsão do robô. Durante a fase de aproximação, não é utilizada a malha de controle de desvio de obstáculos, pois não é considerada a informação da posição do robô, já que esta não era precisa o suficiente.

O sistema que gerencia a navegação do robô, descrito na Subseção 1.2.1, determina quando a trajetória de referência deve ser seguida e quando o robô deve se guiar pelo controle de aproximação. Para a execução do percurso da competição *Robomagellan*, inicialmente, o planejador calcula os

pontos da rota até o primeiro ponto de interesse, onde é colocado um cone de sinalização de trânsito laranja. Esses pontos são passados para o módulo que realiza o controle e rastreamento de trajetória.

Após receber os pontos de interesse P_j , o planejador de movimento calcula o primeiro segmento de caminho λ_i . Este segmento conecta a localização atual do robô $\mathbf{p}(t)$ ao primeiro ponto P_j . Para que o robô execute este caminho de maneira satisfatória, o controlador de trajetória utiliza um modelo de referência, o robô virtual. Este modelo executa o segmento de caminho de forma ideal, com uma configuração de posição representada pelo vetor $\mathbf{p}_r(t)$ e uma de velocidade dada por $\mathbf{q}_r(t)$, em um instante de tempo t .

O controlador de trajetória recebe $\mathbf{p}_r(t)$ e $\mathbf{q}_r(t)$, discretizados como $\mathbf{p}_r[h]$ e $\mathbf{q}_r[h]$, com um intervalo de amostragem $T = 0,05s$. A partir desses dados, ele determina o vetor de velocidades $\mathbf{q}_t[h] = [v_t[h], \omega_t[h]]^T$, que será enviado ao controlador de velocidade, caso não sejam detectados obstáculos. Esse vetor é calculado com as leis de controle definidas na Subseção 3.4.2, como

$$v_t[h] = v_r[h] \cos(e_3[h]) + 1, 2e_1[h], \quad (3.77)$$

$$w_t[h] = w_r[h] + 1, 3v_r[h]e_2[h] + 1, 2 \sin(e_3[h]). \quad (3.78)$$

Paralelamente ao controle e rastreamento de trajetória, ocorre a execução do controle reativo de desvio de obstáculos, utilizando o método das Zonas Virtuais Deformáveis, ou ZVD. Esse método é caracterizado por uma zona protetora ao redor do robô que sofre deformações a partir da leitura dos sensores ultrassônicos, quando algum obstáculo ultrapassa a fronteira de proteção definida. Ao sofrer essas deformações, é calculado o vetor de velocidades $\mathbf{q}_z[h] = [v_z[h], \omega_z[h]]^T$, com um intervalo de amostragem de $T = 0,05s$, com o intuito de evitar a colisão com o obstáculo. Essas velocidades são calculadas a partir das leis de controle obtidas na Subseção 3.5.3

$$v_z[h] = v_z[h - 1] + K_t f[h] \cos(\sigma_T[h]), \quad (3.79)$$

$$\omega_z[h] = \omega_z[h - 1] + K_r \sin(\sigma_T[h]). \quad (3.80)$$

Se $v_z[h]$ ou $\omega_z[h]$ não for igual a zero, significa que há uma deformação na ZVD e o robô precisa reagir para não chocar com o obstáculo. Nesse caso, o controlador de movimento considera os resultados das duas leis de controle, de modo que, a lei de controle para evitar a colisão com obstáculos é aplicada ao movimento do robô virtual do controlador de trajetória, já que o robô real tende a “seguir” o modelo de referência dessa malha de controle.

Assim, o vetor de velocidades do robô virtual \mathbf{q}_r será recalculado com o intuito de evitar a colisão, de acordo com as equações

$$v_r[h] = v_r[h - 1] + K_t f[h] \cos(\sigma_T[h]), \quad (3.81)$$

$$\omega_r[h] = \omega_r[h - 1] + K_r \sin(\sigma_T[h]). \quad (3.82)$$

Em virtude da mudança no vetor de velocidades, a configuração de posição do modelo de referência \mathbf{q}_r passa a ser calculada como

$$x_r[h] = x_r[h - 1] + (v_r \cos(\theta_r[h - 1])T), \quad (3.83)$$

$$y_r[h] = y_r[h - 1] + (v_r \sin(\theta_r[h - 1])T), \quad (3.84)$$

$$\theta_r[h] = \theta_r[h - 1] + (\omega_r T), \quad (3.85)$$

$$(3.86)$$

em que $T = 0,05s$ representa o tempo de amostragem.

Nesse caso, o cálculo da configuração de velocidade $\mathbf{q}[h]$ enviada para o controle de velocidade, na ocorrência de obstáculos, fica de acordo com as equações (3.77) e (3.78).

A partir de $\mathbf{q}[h]$, são obtidas as velocidades de rotação para o motor direito ω_d^* e para o motor esquerdo ω_e^* , que serão dadas como referência para o controlador de velocidades MRAC, através da equação (3.4).

Porém, quando ocorre essa reação ao obstáculo, significa que no segmento de caminho existe algo que impossibilita a execução da trajetória até o ponto de interesse, seja um obstáculo fixo que não estava previamente incluído no mapa, como uma árvore ou uma lixeira; ou um obstáculo dinâmico, como um pedestre ou um carro. Em todo caso, o planejador de rotas irá definir uma nova rota até o destino de interesse, a partir da localização atual do robô, evitando incluir a região com o obstáculo identificado pelo ZVD.

O controle e rastreamento de trajetória recebe esses novos pontos P'_j e recalcula um próximo segmento a ser executado. Se o planejador não conseguir processar uma nova rota a tempo, quando a ZVD estiver livre de deformações, o controlador de movimento faz com que o robô siga para o próximo ponto, P_{j+1} , da rota calculada inicialmente. Como o intuito é chegar ao cone que se encontra no último ponto, pontos intermediários da rota podem ser desconsiderados sem prejuízos para o resultado final. Nesse caso, assim que o planejador de rotas processar a nova rota a ser seguida, o planejador de movimento utiliza os novos pontos para determinar os segmentos de caminho a serem executados.

Caso seja considerado que o robô já se encontra em uma região em que é possível rastrear o cone através do processamento das imagens obtidas pela câmera, o controle de navegação passa para o controlador de velocidades MRAC as velocidades de referência calculadas pelo controle de aproximação. Nesse caso, o módulo que realiza o controle e rastreamento de trajetórias não é executado e a configuração de velocidade fica $\mathbf{q} = \mathbf{q}_c = [v_c, \omega_c]$, com as velocidades v_c e ω_c calculadas, respectivamente, de acordo com as equações (3.73) e (3.74).

Capítulo 4

Resultados

4.1 Introdução

Este capítulo tem o intuito de apresentar os principais resultados obtidos através de experimentos realizados com o controlador de movimento. Será feita uma análise desses resultados com o objetivo de verificar o desempenho do sistema desenvolvido, em situações similares ao desafio proposto na competição *Robomagellan*.

Inicialmente, serão descritos os resultados obtidos através de testes executados em ambiente de simulação. Em seguida, serão apresentados os resultados dos experimentos realizados com o controlador de movimento implementado no robô desenvolvido paralelamente a este trabalho, exibido na Figura 1.3.

4.2 Resultados em simulação

4.2.1 O ambiente de simulação

A arquitetura do robô foi implementada utilizando a ferramenta ROS, do inglês *Robot Operating System*, ou Sistema Operacional de Robôs, que é um sistema que disponibiliza bibliotecas e ferramentas para auxiliar desenvolvedores de *software* a criar aplicações robóticas. Ele foi desenvolvido para facilitar a integração das várias funcionalidades do robô e disponibiliza e reutiliza bibliotecas de simuladores, como o *Stage* e o *Gazebo*¹.

Em virtude disso, o *Gazebo* foi o simulador escolhido para a execução dos testes em ambiente simulado, já que sua integração com o ROS é feita a partir de *plugins*, que facilitam o uso do *Gazebo* para representar uma situação com o robô real². Além disso, também foi utilizado o *Rviz*, do inglês ROS *visualization*, que é uma ferramenta de visualização para dados de sensores e informações de estado do ROS³.

¹Fonte: http://wiki.ros.org/pt_BR.

³Fonte: <http://gazebosim.org/>.

³Fonte: <http://sdk.rethinkrobotics.com/wiki/Rviz>.

4.2.2 Execução de uma trajetória com correção da orientação final

O primeiro teste realizado para verificar a execução do planejador de movimento ocorreu no ambiente de simulação. Este teste tem como objetivo validar a etapa de controle e rastreamento de trajetória, corrigindo a orientação final do robô através do cálculo do ponto intermediário, como descrito na Seção 3.4.

O modelo do robô utilizado no ambiente de simulação pode ser visualizado na Figura 4.1. Ele simula um robô Pioneer e foi escolhido por representar um robô diferencial e por ser de fácil acesso para a utilização no simulador Gazebo. Para a obtenção dos dados do sistema de localização, o simulador fornece a posição e a orientação atual do robô no ambiente de simulação e para adquirir os pontos da rota foi feita uma lista de pontos (x, y) , que foram fornecidos para o controle e rastreamento de trajetórias.

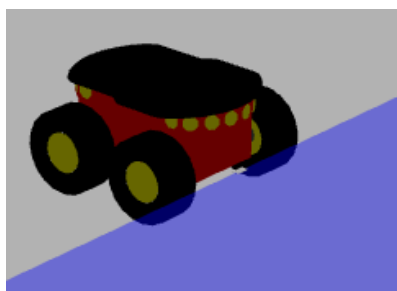


Figura 4.1: Robô utilizado para os testes em simulação.

As trajetórias executadas pelo robô controlado e pelo robô virtual são exibidas no gráfico da Figura 4.2. Nesse gráfico, os segmentos de reta em vermelho representam os segmentos de caminho calculados pelo planejador de movimento e executados pelo robô virtual, já os segmentos em azul indicam a trajetória executada pelo robô simulado; os pontos vermelhos do gráfico indicam os pontos P_j da rota e o ponto em verde representa a posição inicial do robô no ambiente; e o quadrado laranja indica o ponto intermediário calculado para corrigir a orientação final.

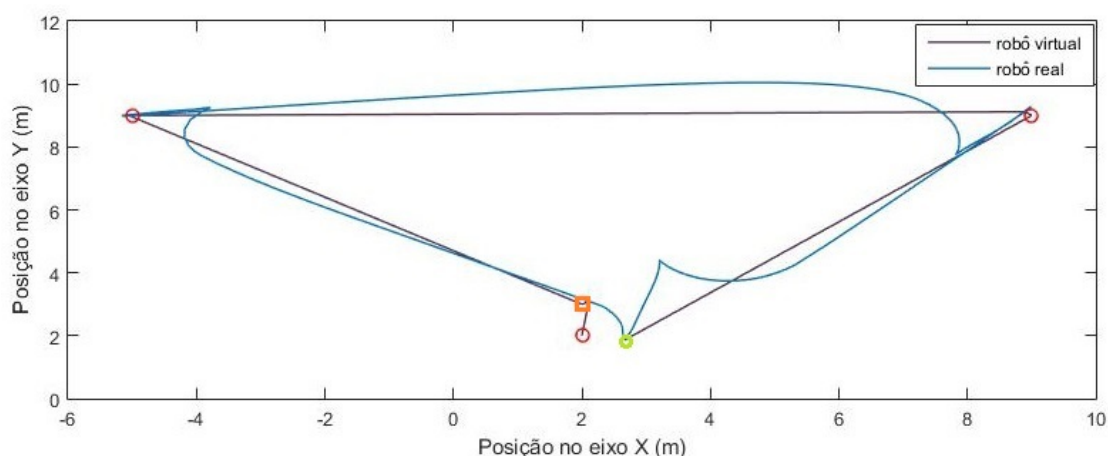
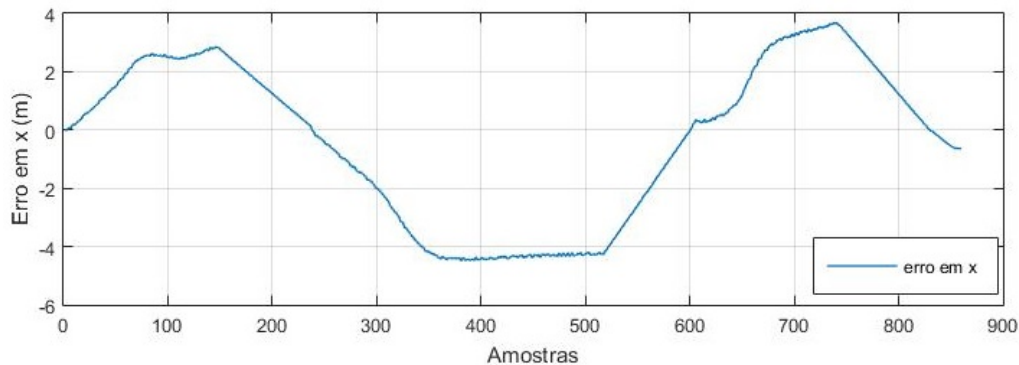


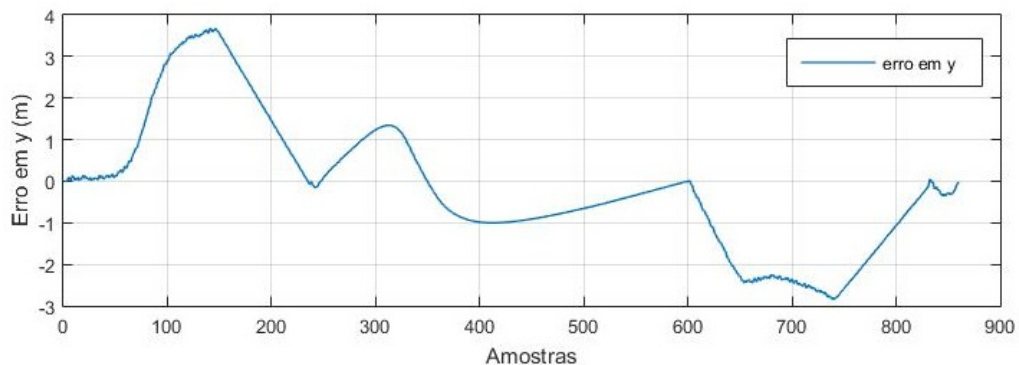
Figura 4.2: Trajetória executada pelo robô simulado em relação ao robô virtual com correção da orientação final.

Os erros de posição e orientação entre o robô controlado e a referência, a cada intervalo de amostragem, podem ser visualizados nos gráficos da Figura 4.3. No gráfico que representa o erro de posição no eixo X , observa-se que o robô chegou ao destino com um erro diferente de zero, porém, o erro de posição em relação ao eixo Y e o erro de orientação não foram significativos.

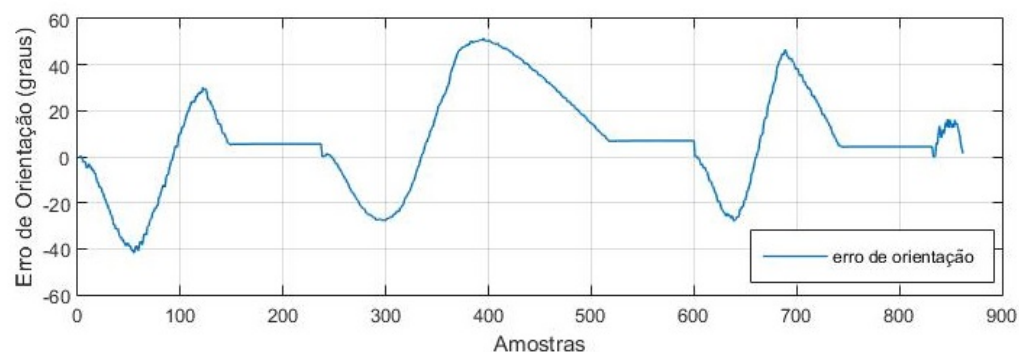
Em relação a etapa de correção da orientação final, a orientação de referência fornecida para o controlador de trajetória foi $\theta_f = -90^\circ$. O robô controlado chegou ao destino final com uma orientação $\theta = -94,25^\circ$. Logo, o erro na orientação final foi $\theta - \theta_f = -4,25^\circ$. As orientações de referência e a do robô controlado, a cada intervalo de amostragem, podem ser visualizadas no gráfico exibido na Figura 4.4, sendo $\theta \in [-\pi, \pi]$.



(a) Erro de posição no eixo X.



(b) Erro de posição no eixo Y.



(c) Erro de orientação entre o robô simulado e a referência.

Figura 4.3: Erros de posição e orientação entre o robô simulado e o robô virtual.

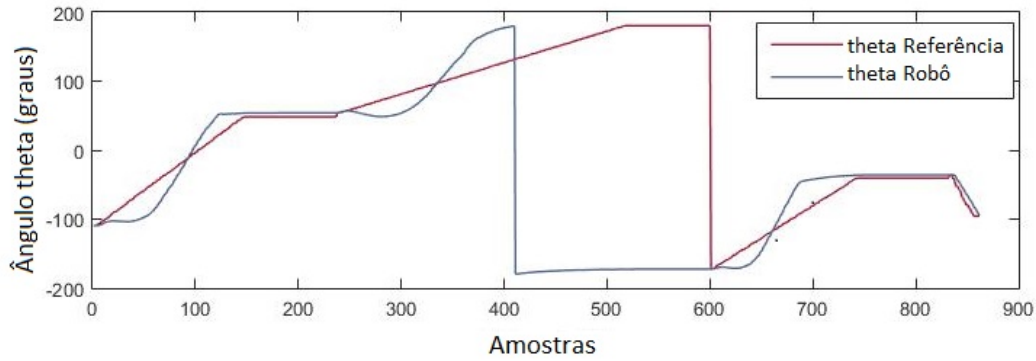


Figura 4.4: Orientação do robô simulado e do robô virtual.

Ao analisar as trajetórias executadas nesse experimento, percebe-se que o robô controlado conseguiu chegar nos pontos determinados pelo planejador de rotas, porém, com pequenos desvios dos segmentos de caminho calculados inicialmente. Esses desvios podem ser explicados devido ao fato do robô não ser holonômico, assim, ele não é capaz de executar as curvas formadas pela conexão de dois segmentos consecutivos, causando um desvio da referência. Porém, mesmo com essa limitação, o robô executou a rota de maneira satisfatória.

Com o teste realizado conclui-se que as etapas do controle e rastreamento de trajetória, planejador de movimento, robô virtual e o controle por modelo de referência com correção da orientação final, foram executadas conforme o esperado no desenvolvimento realizado na Seção 3.4.

4.2.3 Execução de uma trajetória com o planejador de rotas

Esta subseção apresenta os resultados do teste realizado no ambiente de simulação com o controle e rastreamento de trajetória executado juntamente com o módulo do planejador de rotas, descrito na Subseção 1.2.1. Um dos intuitos deste experimento foi avaliar a integração entre essas duas partes da arquitetura do robô. Além disso, para este trabalho, foi avaliado o desempenho do controle e rastreamento de trajetórias ao receber os pontos da rota P_j do planejador e tendo a sua execução coordenada pelo controle de navegação.

Neste experimento o robô real foi simulado no Gazebo conforme exibido pela Figura 4.1. O ambiente onde executou-se o teste foi construído no simulador de forma similar ao local da competição *Robomagellan* e pode ser visualizado na Figura 4.5. Para a realização desse ensaio, o planejador de rotas possuía um conhecimento prévio do local no qual o robô simulado iria executar a trajetória, na forma de um mapa de custo. A visualização desse mapa no Rviz é apresentada na Figura 4.6, na qual a região em preto representa os obstáculos fixos, como, por exemplo prédios, lixeiras e canteiros; e a área em claro representa a região em que o robô pode se movimentar.

A Figura 4.6 apresenta também a rota calculada pelo planejador de rotas, representada pelos segmentos de reta em rosa. Essa rota é fornecida ao controlador de movimento na forma de pontos de passagem P_j , sendo os pontos dos extremos dos segmentos.

Após receber a rota calculada e a localização atual do robô, o planejador de movimento calculou

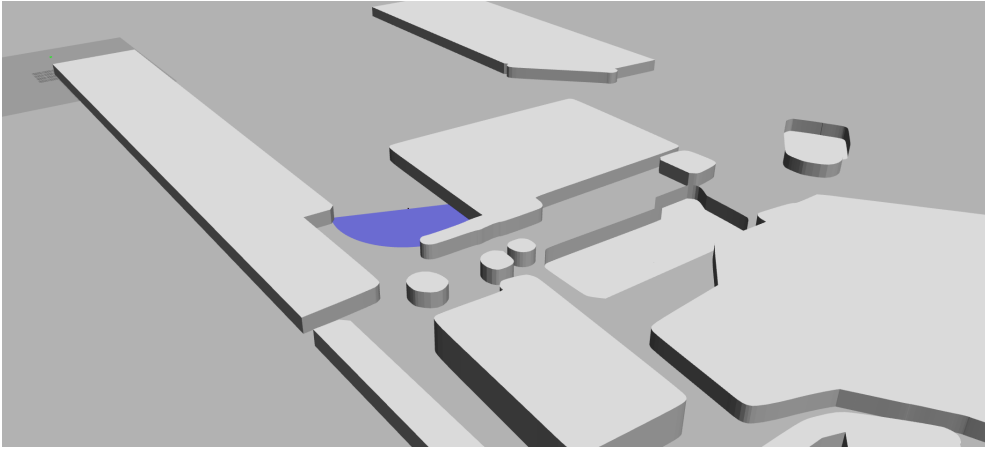


Figura 4.5: Ambiente simulado no Gazebo.

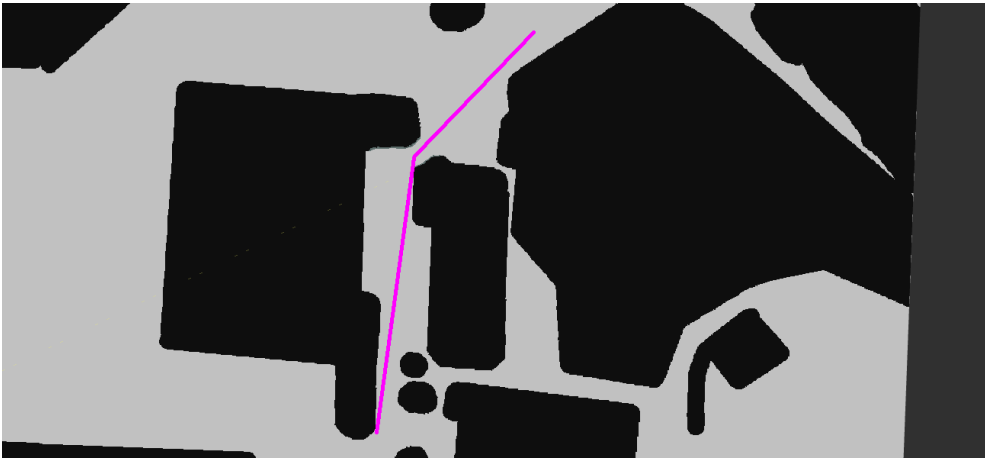


Figura 4.6: Mapa de custo do ambiente de teste em simulação.

os segmentos de caminho λ_i por onde o robô deveria se mover. A execução desses segmentos pelo modelo de referência e pelo robô controlado pode ser observada no gráfico da Figura 4.7. No gráfico, os marcadores em formato circular indicam os pontos P_j recebidos, os segmentos em vermelho representam o trajeto executado pelo robô virtual e os segmentos em azul correspondem ao trajeto realizado pelo robô simulado.

A Figura 4.8 apresenta os gráficos de posição e orientação do robô virtual, \mathbf{p}_r , e do robô simulado, \mathbf{p} , em cada intervalo de amostragem. Os erros de posição e orientação entre o robô e a referência são exibidos nos gráficos da Figura 4.9, nos quais nota-se que os erros no final da execução da trajetória convergiram para zero.

Analisando-se os resultados obtidos com o experimento, conclui-se que o planejador de movimento teve um desempenho satisfatório, pois o robô controlado atingiu os pontos especificados pelo planejador de rotas com os erros convergindo para zero ao final da execução. Entretanto, durante a execução do primeiro segmento de caminho, o robô controlado se distanciou da trajetória de referência, pois ele mudou de orientação.

Além disso, pode-se verificar que a interação entre o módulo planejador de rotas e o controlador

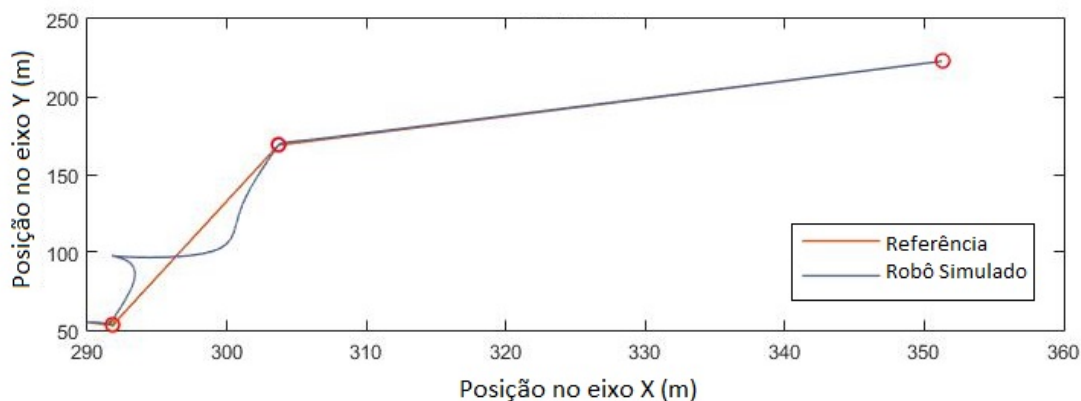


Figura 4.7: Trajetórias executadas pelo robô simulado e pelo robô virtual a partir dos pontos do planejador de rotas.

de movimento ocorreu como o esperado no planejamento da arquitetura do robô, como descrito na seção 1.2.1.

4.3 Resultados no robô real

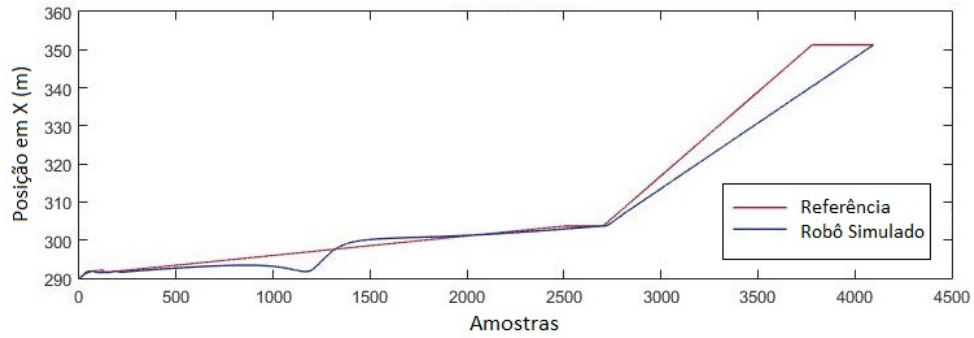
Nesta seção serão apresentados e analisados os resultados obtidos a partir dos experimentos realizados com o controlador de movimento implementado na plataforma real, exibida na Figura 1.3. Inicialmente, foram realizados experimentos com o controlador de velocidade proposto, em seguida, foram realizados testes com os outros módulos do controlador de movimento. Porém, não foi possível coletar dados executando todos os módulos do controlador de movimento em um só ensaio, devido às limitações físicas do robô. Assim, foram realizados testes separadamente para a etapa de aproximação do objeto de interesse e para a etapa de execução de uma trajetória.

4.3.1 Resultados do controlador de velocidade MRAC

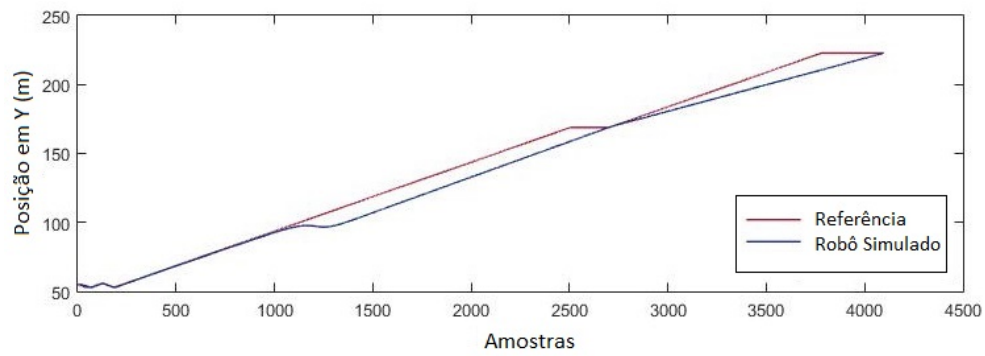
O primeiro ensaio realizado com o robô real teve como objetivo a validação do controlador de velocidade proposto. O experimento consistiu em observar o comportamento dos sistemas de propulsão ao serem fornecidas velocidades de rotação de referência para as rodas direita ω_d^* e esquerda ω_e^* . Para a realização deste primeiro teste, o robô foi mantido em uma plataforma de apoio, de forma que as rodas não tiveram contato com o solo.

A Figura 4.10 exibe os gráficos com as velocidades calculadas das rodas e a velocidade de referência, em relação ao tempo. Nos gráficos, a curva em azul representa as velocidades calculadas a partir das leituras dos codificadores incrementais e a curva em vermelho indica a referência. Os gráficos para o sistema de propulsão direito e esquerdo, exibidos, respectivamente, pelas Figuras 4.10a e 4.10b, apresentaram um comportamento semelhante. Assim, a análise do controlador MRAC será realizada de forma geral.

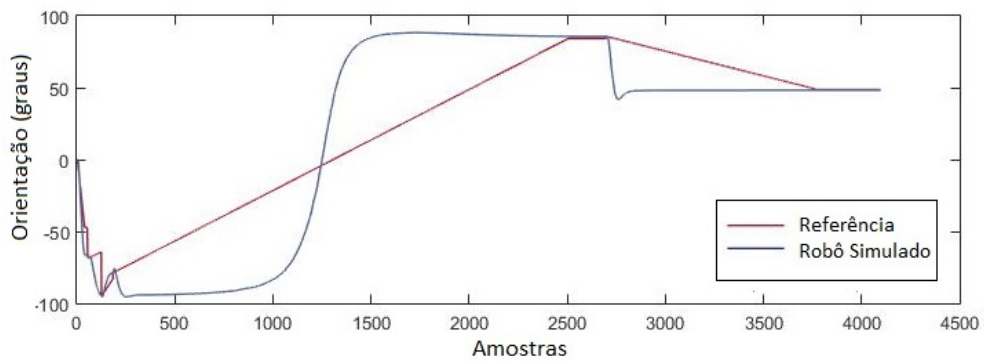
A partir dos dados obtidos, percebe-se que o controlador de velocidade MRAC apresentou, em



(a) Posição do robô em relação a referência, no eixo X .

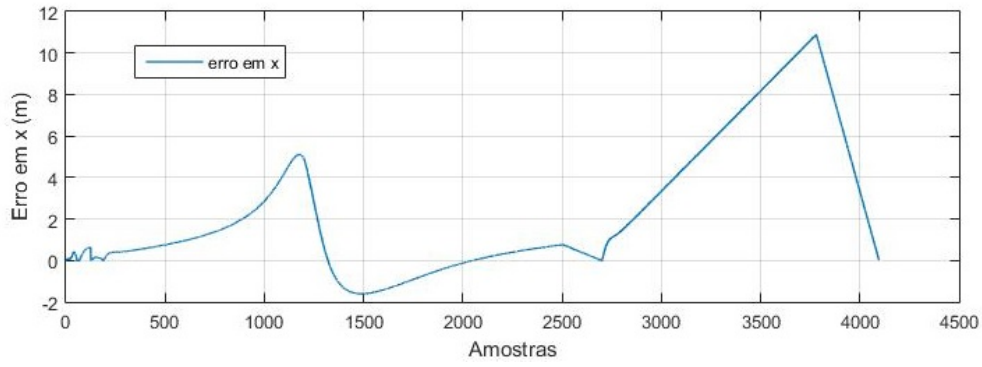


(b) Posição do robô em relação a referência, no eixo Y .

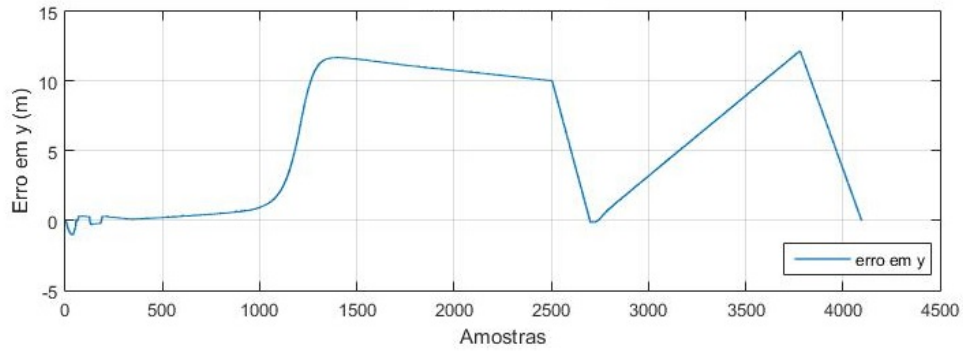


(c) Orientação do real e a orientação de referência.

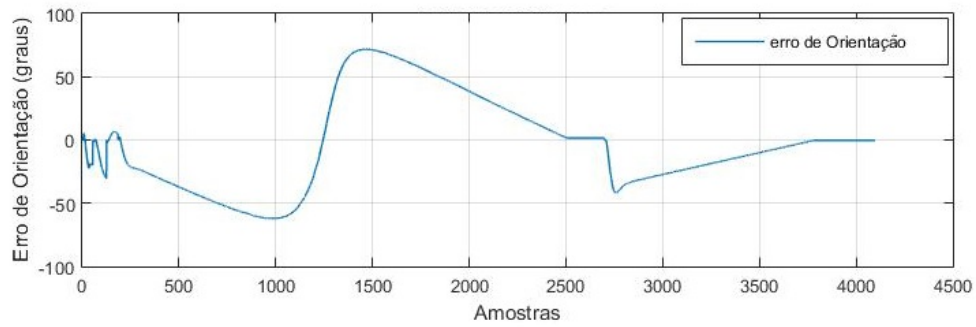
Figura 4.8: Posição e orientação do robô simulado em relação ao robô virtual.



(a) Erro de posição no eixo X.

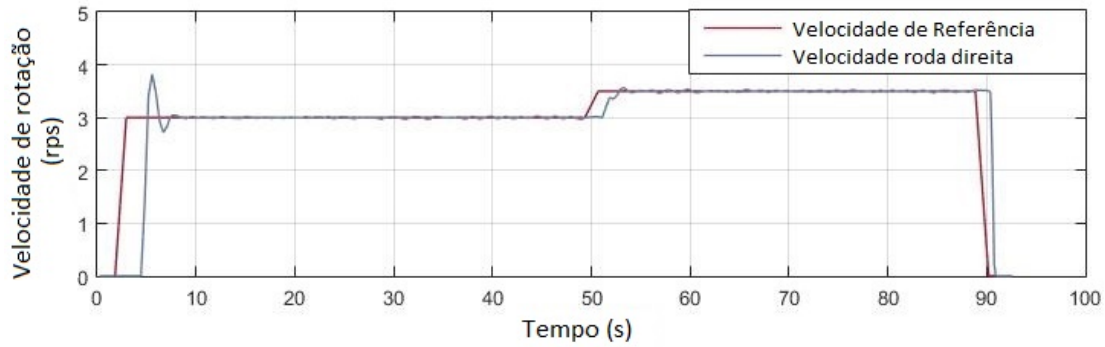


(b) Erro de posição no eixo Y.

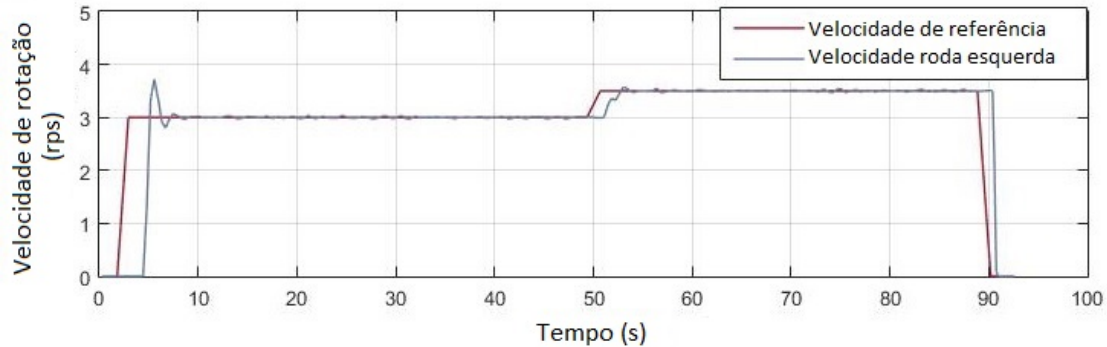


(c) Erro de orientação entre o robô e a referência.

Figura 4.9: Erros de posição e orientação entre o robô simulado e o robô virtual.



(a) Velocidade de rotação da roda direita $\omega_d(t)$ em relação a referência $\omega_d^*(t)$.



(b) Velocidade de rotação da roda esquerda $\omega_e(t)$ em relação a referência $\omega_e^*(t)$.

Figura 4.10: Velocidades das rodas e as velocidades de referência.

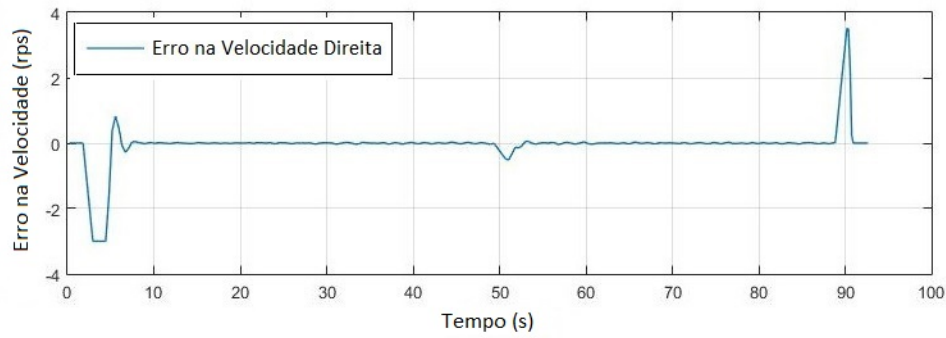
regime permanente, erro nulo. Entretanto, o transitório foi lento e levemente oscilatório, como o esperado para a metodologia de controle escolhida. Os erros entre as velocidades das rodas direita e esquerda e as respectivas referências podem ser visualizados nos gráficos contidos nas Figuras 4.11a e 4.11b. Analisando esses gráficos, constata-se que os erros foram significativos somente durante as transições nos valores de referência de velocidade.

Assim, conclui-se que o controlador de velocidade MRAC apresentou um desempenho satisfatório no sistema analisado, visto que o robô manteve-se na velocidade de referência durante o regime permanente. Logo, este controlador é considerado adequado para a execução juntamente com o controlador de trajetória e com o controle de aproximação.

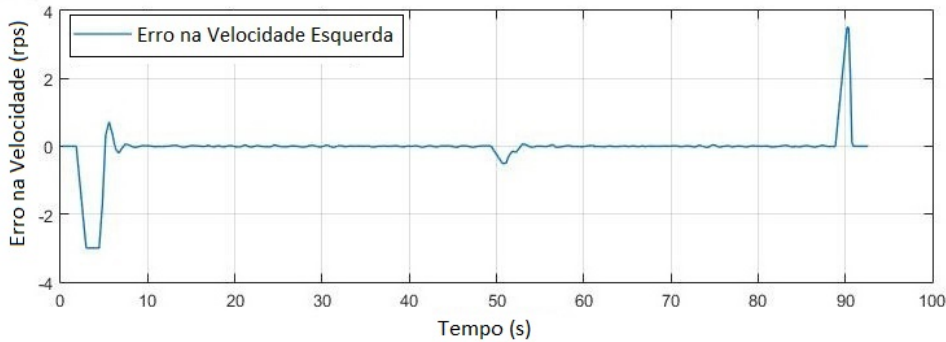
4.3.2 Execução de uma trajetória

O segundo experimento tem como finalidade fazer com que o robô real execute uma trajetória, a partir dos pontos da rota P_j calculados pelo planejador de rotas e da posição e orientação do veículo, $\mathbf{q}(t)$, dadas pelo sistema de localização. Assim, esse experimento pretende avaliar a interação dos módulos da arquitetura proposta e o desempenho do controlador de movimento aplicados ao robô real.

Entretanto, devido a limitações físicas do veículo e falhas nos sensores que forneciam a coordenada GPS e a orientação do robô, o experimento foi realizado aplicando-se as velocidades



(a) Erro entre a velocidade da roda direita ω_d e a referência ω_d^* .



(b) Erro entre a velocidade da roda esquerda ω_e e a referência ω_e^* .

Figura 4.11: Erros entre as velocidades das rodas e as velocidades de referência.

calculadas pelo controlador de trajetória nas rodas, mas sem estas estarem em contato direto com o solo.

Devido a restrições do robô, inicialmente, foram implementados somente o controlador de velocidade e o controle e rastreamento de trajetórias sem a correção da orientação final, para a execução de uma trajetória. Em seguida, foi realizado um segundo ensaio, no qual o controle de desvio de obstáculos também foi implementado na execução de uma trajetória pelo robô.

A localização e orientação do robô foram obtidas a partir da odometria e a visualização do “movimento” do veículo ao executar a trajetória foi possível a partir do deslocamento do robô no mapa do Rviz, exibido pela Figura 4.12. Esse mapa representa uma quadra de Brasília, onde foi escolhido realizar o ensaio. Nele, os locais em preto indicam os obstáculos fixos e a região mais clara o espaço no qual o robô poderia se deslocar.

Além disso, a Figura 4.12 exibe a trajetória calculada pelo planejador de rotas, indicada pelos segmentos de reta em rosa. Ademais, o quadrado verde representa o robô e o arco branco simboliza as leituras dos 11 sensores ultrassônicos. A trajetória executada pelo veículo e a realizada pelo modelo de referência são exibidas no gráfico da Figura 4.13. De modo que, a curva em azul indica a trajetória executada pelo modelo de referência e a em vermelho a realizada pelo robô real; e os círculos em vermelho e preto representam os pontos da rota P_j .

Analisando-se o gráfico obtido, percebe-se que o veículo controlado conseguiu executar a trajetória conforme o esperado, destacando-se que, por se tratar de um robô não-holonômico, posto que,

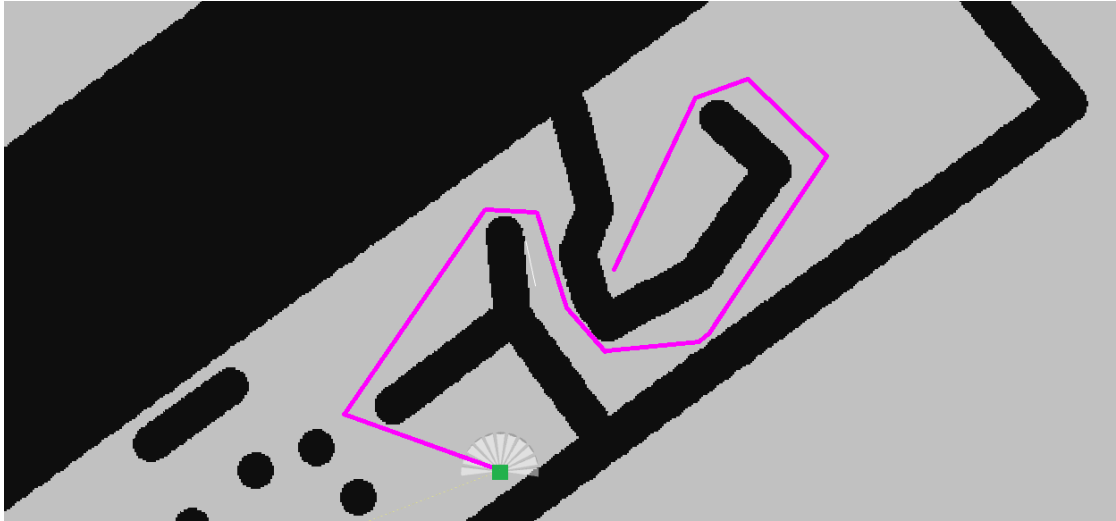


Figura 4.12: Mapa de custo do ambiente de teste com o robô real.

não é capaz de se mover lateralmente, ele não conseguiu se locomover exatamente nos segmentos de caminho calculados pelo planejador de movimento, ocorrendo desvios da trajetória de referência.

Quando esses desvios são significativos, o planejador de rotas calcula novos pontos P_j para o destino desejado, a partir da localização atual do robô. Nesse ensaio, o veículo se distanciou consideravelmente da rota, fazendo com que o planejador recalculasse o percurso de referência, os novos pontos P_j recebidos estão representados no gráfico pelos círculos em preto.

Os erros de posição e orientação do veículo real em relação ao modelo de referência são apresentados nos gráficos da Figura 4.14. Apesar do distanciamento considerável do robô de um dos segmentos de caminho, ocasionando erros de posição e orientação consideráveis, observa-se que a trajetória foi realizada com êxito.

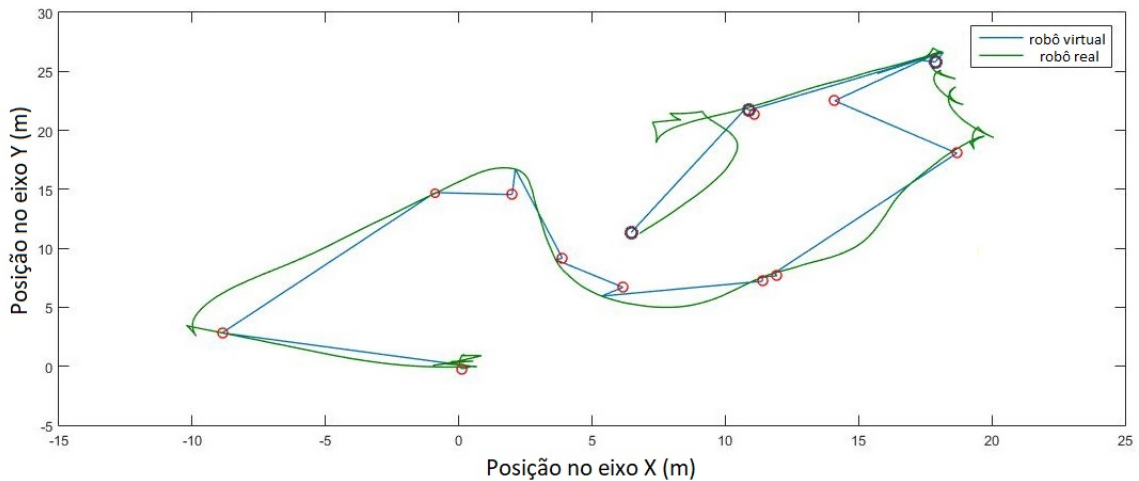
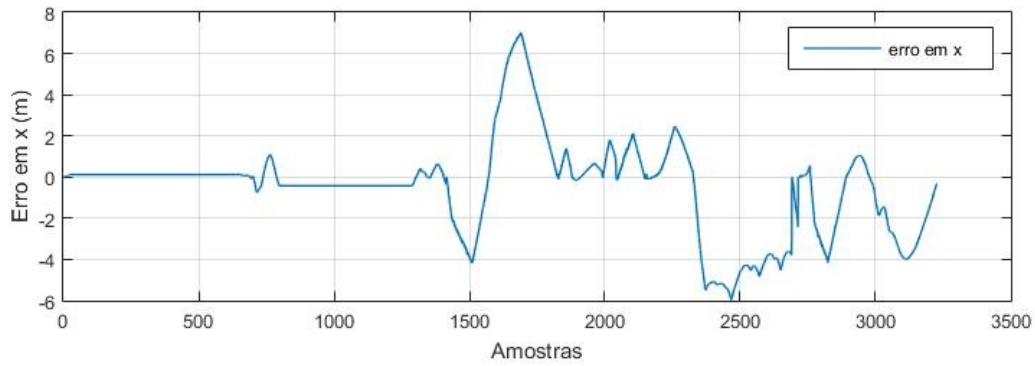
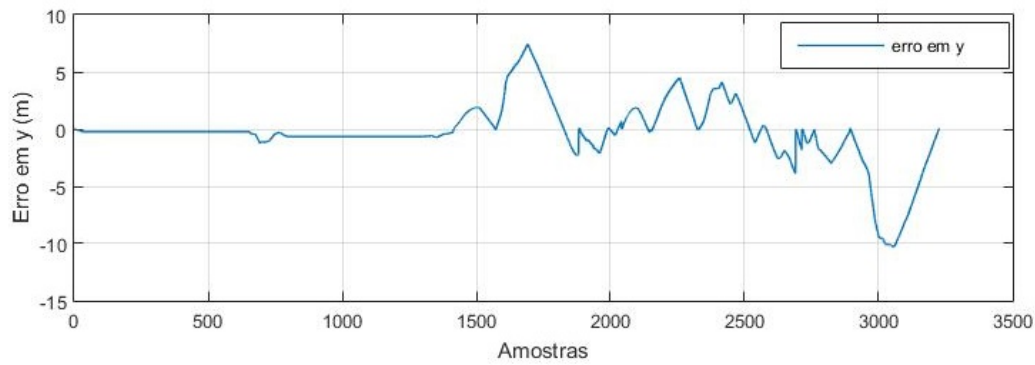


Figura 4.13: Trajetórias executadas pelo robô real e pelo robô virtual.

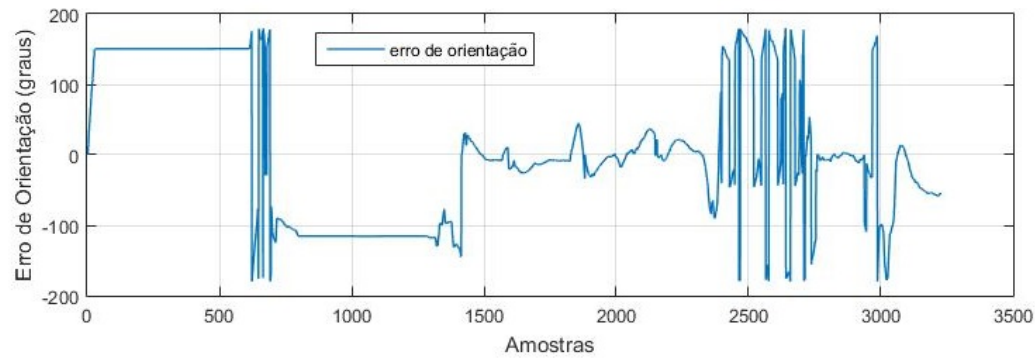
Uma vez que, o controlador de trajetória implementado foi executado conforme o esperado. Já que, o planejador de movimento funcionou como esperado, calculando os segmentos de caminho λ_i



(a) Erro de posição no eixo X , obtido no experimento com o robô real.



(b) Erro de posição no eixo Y , obtido no experimento com o robô real.



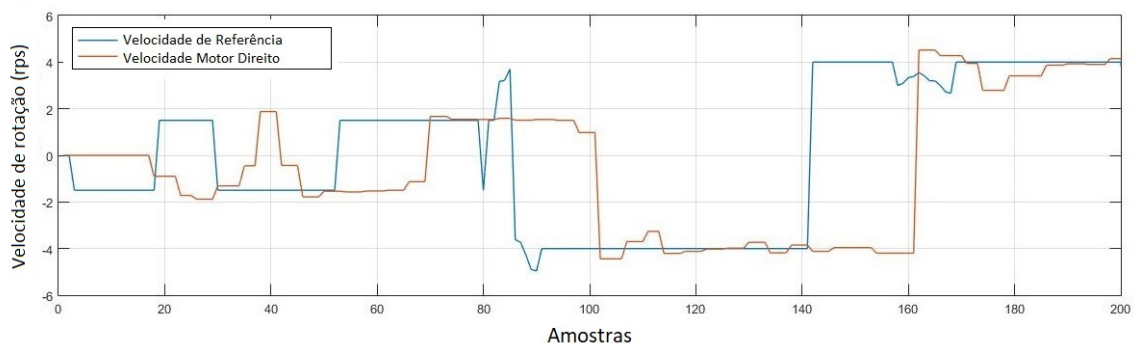
(c) Erro de orientação entre o robô real e a referência, obtido no experimento com o robô real.

Figura 4.14: Erros de posição e orientação entre o robô real e o robô virtual.

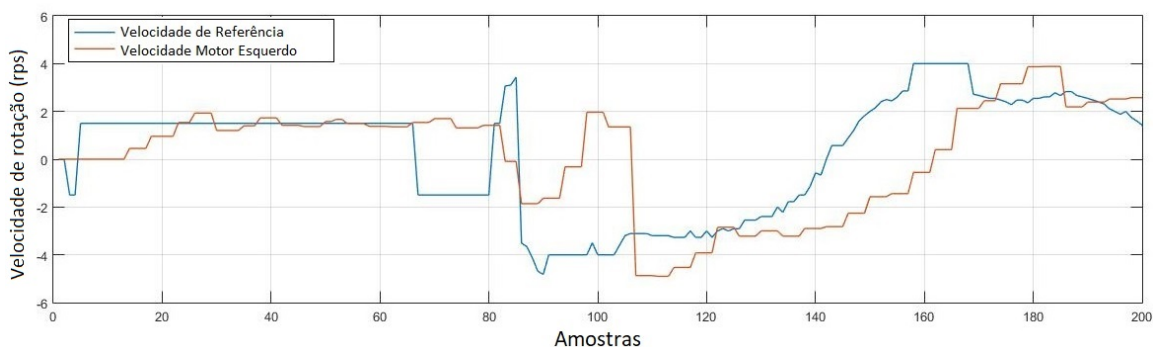
entre os pontos da rota P_j e definindo quando o veículo deveria passar para o próximo segmento. E o controlador de trajetória foi capaz de manter o robô “seguindo” o movimento do modelo de referência na maior parte do caminho executado e corrigindo o movimento do robô na ocorrência do desvio da trajetória.

Além disso, a interação com o controlador de velocidade MRAC sucedeu como esperado, pois as velocidades de rotação calculadas pelo controlador de trajetória foram repassadas para os sistemas de propulsão de forma satisfatória, como pode ser observado nos gráficos da Figura 4.15. Nesses gráficos, a curva em vermelho indica a velocidade de rotação das rodas e a a curva em azul representa a velocidade calculada pelo controlador de trajetória, a cada intervalo de amostragem. Observa-se nessas curvas que o controlador de velocidade conseguiu repassar para os sistemas de propulsão a velocidade desejada.

A partir da análise dos dados obtidos nesse primeiro ensaio de execução de uma trajetória com o robô real, conclui-se que a interação entre os módulos da arquitetura transcorreu com sucesso, permitindo simular uma situação parecida com os percursos do desafio proposto pela competição *Robomagellan*. Porém, não foi possível executar todos os módulos previstos na arquitetura, devido às limitações físicas do robô. Além disso, verificou-se com o experimento realizado, que o controlador de movimento funcionou conforme o esperado, pois o veículo foi capaz de executar a rota calculada com êxito.



(a) Velocidade de rotação da roda direita em relação a referência.



(b) Velocidade de rotação da roda esquerda em relação a referência.

Figura 4.15: Velocidades calculadas pelo controlador de trajetória e as velocidades atuais do robô.

4.3.3 Execução de uma trajetória com desvio de obstáculos

O segundo teste realizado para a execução de uma trajetória com o robô real tem como objetivo fazer com que o robô percorra a trajetória desviando dos obstáculos que poderiam aparecer no seu caminho. Para efetuar esse experimento, o robô permaneceu suspenso em uma plataforma, deixando as rodas sem contato com o solo.

Logo, para a implementação do desvio de obstáculos, foram simulados obstáculos durante a execução do trajeto, colocando-se temporariamente objetos no alcance dos sensores ultrassônicos durante a realização do percurso. O mapa do local de testes permaneceu conforme a Figura 4.12.

A trajetória executada pelo robô e os segmentos de caminho de referência são apresentados no gráfico da Figura 4.16. Nesse gráfico, os segmentos de reta em azul e em rosa representam o movimento do robô virtual e o segmento em vermelho o percurso feito pelo robô real. Além disso, os círculos em vermelho indicam os pontos da rota P_j , fornecidos pelo planejador de rotas.

Observando-se o trajeto que o robô real realizou, em duas situações este não seguiu o segmento em azul, que representa o movimento do robô virtual, pois havia obstáculos no seu caminho. Assim que o robô desviou desses obstáculos, o planejador de movimento calculou novos segmentos de caminho para o destino desejado a partir da posição em que o robô real estava, esses segmentos recalculados estão desenhados em rosa no gráfico. E fazendo com que o robô virtual também passasse a seguir o novo segmento calculado, a mudança de posição do robô virtual é indicada pelos segmentos tracejados em azul.

A partir da realização desse experimento, conclui-se que a interação do controlador de desvio de obstáculos com o controle e rastreamento de trajetórias ocorreu como esperado. Observa-se também que o método proposto para o desvio de obstáculos, a Zona Virtual Deformável, impediria que o robô colidisse com algum objeto. Porém, como não foi possível a realização de testes com o robô se locomovendo de fato, não foram coletados dados da reação do robô ao desviar de obstáculos em uma situação mais real.

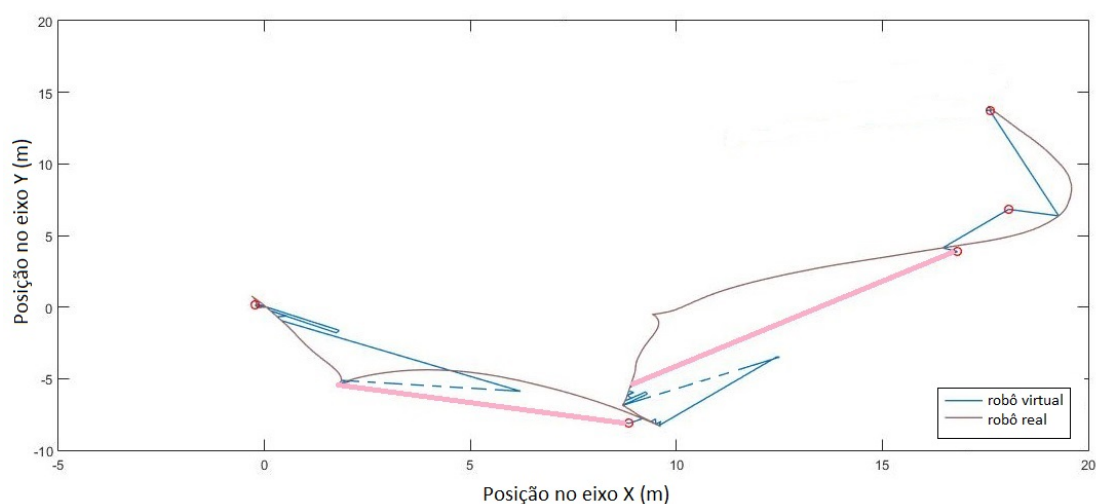


Figura 4.16: Trajetória executada pelo robô real em relação a referência, com desvio de obstáculos.

4.3.4 Teste de aproximação do cone

O último teste realizado com o robô real tem como objetivo validar o método proposto para o controle de aproximação do objeto de interesse. Neste experimento somente serão implementados o controlador de velocidade e o controlador de aproximação, dos módulos desenvolvidos neste trabalho.

Para a execução deste teste, o controle de aproximação recebe como parâmetros a distância d_c e a orientação θ_c que o cone se encontra da parte frontal do robô, que são dados fornecidos pelo módulo da arquitetura que faz o processamento de imagens. Assim, neste experimento também será avaliada a interação do controlador de movimento com outros módulos da arquitetura do robô.

Para que o robô fizesse o percurso até o cone, ele foi colocado inicialmente em uma posição em que o cone já era identificado, através das imagens da câmera. A Figura 4.17 exibe o robô finalizando a execução do controlador de aproximação.

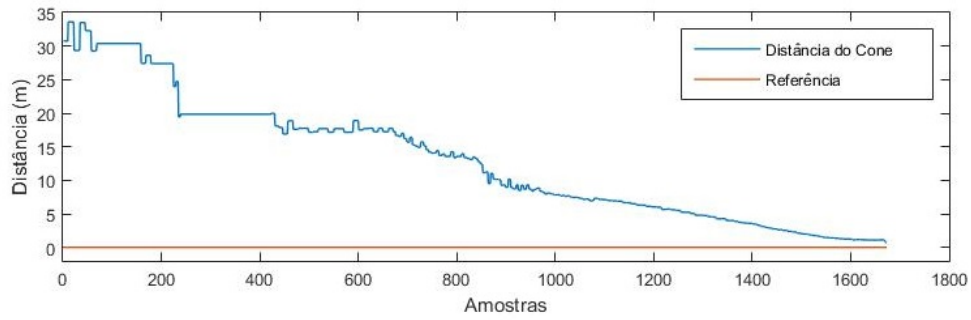
As Figuras 4.18a e 4.18b apresentam, respectivamente, os gráficos da distância d_c e da orientação θ_c , a cada intervalo de amostragem. Percebe-se que o controlador de aproximação executou como esperado, já que o robô foi capaz de encostar no cone, assim como requisitado no desafio da *Robomagellan*.

As Figuras 4.19a e 4.19b exibem, respectivamente, os gráficos das velocidades linear v_c e angular ω_c calculadas pelo controlador, a cada intervalo de amostragem. Observa-se que, conforme o veículo se aproximava do cone, as velocidades foram reduzidas, assim, o robô foi capaz de chegar de maneira suave no alvo. Nota-se também, que inicialmente a velocidade linear calculada permaneceu em $v_c = 1.1$ rot/s, isso ocorreu, pois foi definido esse valor de segurança para quando o robô estivesse a uma distância superior a 5 metros do cone.

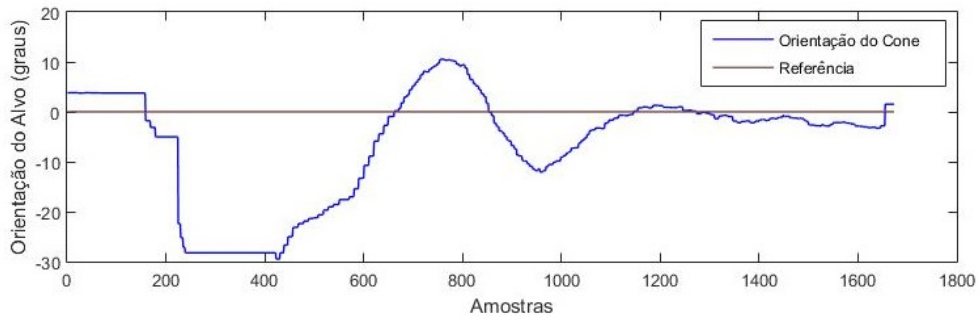
A partir deste teste realizado, conclui-se que o controlador de aproximação proposto interagiu conforme o esperado com os outros módulos da arquitetura do robô. Além disso, percebe-se que o controlador foi executado como esperado, possibilitando que o robô encostasse no alvo.



Figura 4.17: Robô aproximando-se do cone.

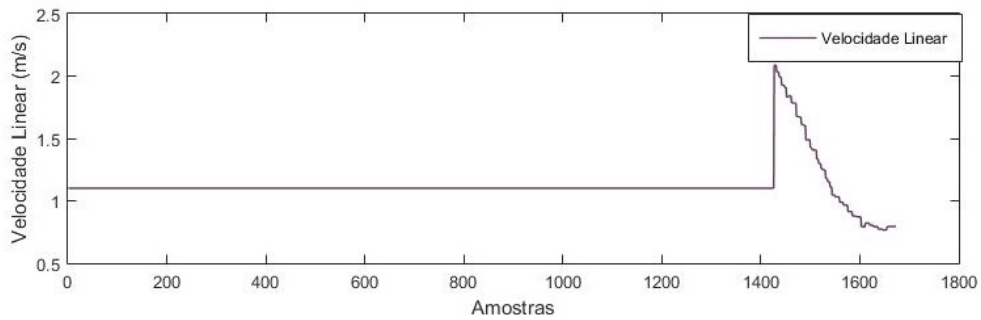


(a) Posição do cone em relação ao robô, d_c .

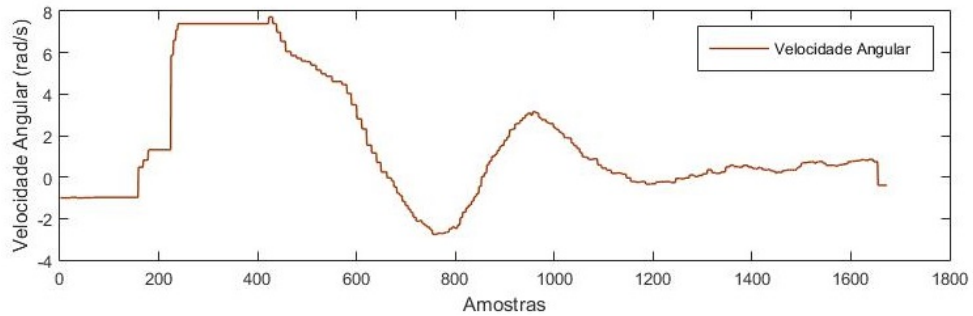


(b) Orientação do cone em relação ao robô, ω_c .

Figura 4.18: Posição e orientação do cone em relação ao robô real.



(a) Velocidade linear calculada pelo controlador de aproximação.



(b) Velocidade angular calculada pelo controlador de aproximação.

Figura 4.19: Velocidades calculadas pelo controlador de aproximação.

Capítulo 5

Conclusões

Esse trabalho apresentou o desenvolvimento de um controlador de movimento para um robô móvel não-holonômico com tração diferencial, com o intuito de contribuir para a construção de um robô para participar da competição *Robomagellan*. Esse robô foi desenvolvido em paralelo com outros trabalhos de graduação e com o auxílio da equipe Divisão de Robótica Inteligente (DROID) da Universidade de Brasília.

Inicialmente, apresentou-se uma revisão bibliográfica da literatura já existente sobre o tema abordado, mencionando os trabalhos que foram fundamentais para a realização desse projeto e artigos relevantes acerca do assunto tratado. Foram descritos, inclusive, os principais conceitos teóricos utilizados para a elaboração desse projeto.

O controlador proposto foi elaborado considerando aspectos teóricos e práticos das áreas da robótica móvel, como metodologias de controle de velocidade, posição e desvio de obstáculos. Já que ele é formado a partir da interação de quatro malhas de controle, o controle e rastreamento de trajetórias, este recebe os pontos da rota e é encarregado de fazer com que o robô se mantenha nos caminhos planejados; o controlador de desvio de obstáculos, que forma uma zona protetora imaginária ao redor do robô com o intuito de evitar colisões com obstáculos; o controlador de aproximação, o qual é responsável por fazer o robô encostar em um alvo determinado; e o controlador de velocidade, que garante que o veículo se mantenha na velocidade desejada.

O desenvolvimento do controle de movimento ocorreu em conjunto com outros trabalhos, sendo dependente destes para o seu funcionamento no robô real. Entretanto, o controlador de movimento poderia ser implementado em outras arquiteturas, necessitando apenas fornecer os dados de entrada essenciais para o seu funcionamento.

Os resultados obtidos através dos experimentos realizados forneceram informações qualitativas e quantitativas a respeito do desempenho do controlador de movimento. Assim, foi possível verificar que o controlador atingiu os objetivos estabelecidos no início do projeto, ou seja, ele coordenou com êxito a movimentação do robô para a execução de percursos similares aos do desafio proposto pela *Robomagellan*. Fazendo com que o robô fosse capaz de rastrear e executar trajetórias entre pontos determinados, desviar de possíveis obstáculos, se aproximar de pontos de interesse e repassar para os sistemas de propulsão as velocidades desejadas. Entretanto, não foi possível realizar

experimentos que simulassem precisamente o percurso da competição, pois o robô teve que ser mantido suspenso na maioria dos testes realizados.

Para trabalhos futuros, é proposto que seja finalizada e melhorada a implementação do controlador de movimento em um robô real, de modo que todos os seus módulos possam ser executados para o cumprimento do desafio proposto pela *Robomagellan*. Como melhorias no controlador, propõe-se que seja implementado um sistema de controle de desvio de obstáculos durante a fase de aproximação do cone e um planejador de movimento que leve em consideração as características não-holonômicas do robô.

Além disso, como o intuito da competição é simular uma situação na qual os robôs sejam autônomos e capazes de se deslocar em uma cidade sem a intervenção humana, sugere-se que o sistema desenvolvido nesse trabalho seja implementado em outras arquiteturas e veículos de teste, sendo utilizado para realizar percursos mais complexos do que os executados nesse trabalho e para se aproximar de outros objetos de interesse.

REFERÊNCIAS BIBLIOGRÁFICAS

- [1] NASCIMENTO, T. P. do. *Controle de trajetória de robôs móveis omni-direcionais: uma abordagem multivariável*. Dissertação (Mestrado) — Universidade Federal do Rio Grande da Bahia, 2009.
- [2] MURPHY, R. R. *Introduction To AI Robotics*. Cambridge, Massachusetts: MIT, 2000.
- [3] KÜHNE, F. *Controle Preditivo de robôs móveis não holonômicos*. Dissertação (Mestrado) — Universidade Federal do Rio Grande do Sul, 2005.
- [4] FUKAO, T.; NAKAGAWA, H.; ADACHI, N. Adaptive tracking control of a nonholonomic mobile robot. *IEEE Transactions on Robotics and Automation*, v. 16, n. 5, oct 2000.
- [5] CHAVEZ, J. R. M. *Zona Virtual Deformável com Filtro de Partículas no Rastreamento de Obstáculos em Robótica Móvel*. Dissertação (Mestrado em Sistemas Mecatrônicos) — Universidade de Brasília, 2014.
- [6] BORGES, G. A. *Um Sistema Óptico de Reconhecimento de Trajetórias para Veículos Automáticos*. Dissertação (Mestrado) — Universidade Federal da Paraíba, 1998.
- [7] JACOME, I. C. *Controle Adaptativo por Modelo de Referência e Estrutura Variável Discreto no Tempo*. Dissertação (Mestrado) — Universidade Federal do Rio Grande do Norte, 2013.
- [8] NISE, N. S. *Engenharia de Sistemas de Controle*. sixth. [S.l.]: LTC, 2012.
- [9] KAWABATA, K.; MA, L.; XUE, J.; ZHU, C.; ZHENG, N. A path generation for automated vehicle based on bezier curve and via-points. *Robotics and Autonomous Systems*, Elsevier, v. 74, p. 243–252, 2015.
- [10] BORGES, G. A.; DEEP, G. S.; LIMA, A. M. N. Controladores cinemáticos de trajetória para robôs móveis com tração diferencial. *VI Simpósio Brasileiro de Automação Inteligente*, 2003.
- [11] RESENDE, C. Z.; ESPINOSA, F.; SARCINELLI-FILHO, M.; BASTOS-FILHO, T. F. Controlador de seguimento de trajetória para robôs móveis com ganhos dinâmicos. *Simpósio Brasileiro de Automação Inteligente - SBAI*, X, p. 983–988, 2011.
- [12] PINTO, A. G.; FRAISSE, P.; ZAPATA, R. A decentralized adaptive trajectory planning approach for a group of mobile robots. *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference*, 2006.

- [13] FALEIROS, A. C.; YONEYAMA, T. *Teoria Matemática de Sistemas*. [S.l.]: Editora ITA, 2002.
- [14] HEINEN, F. J. *Sistema de controle híbrido para robôs móveis autônomos*. Dissertação (Mestrado) — Universidade do Vale do Rio do Sinos, 2002.
- [15] PEREIRA, F. G. *Navegação e Desvio de Obstáculos Usando um Robô Móvel Dotado de Sensor de Varredura Laser*. Dissertação (Mestrado) — Universidade Federal do Espírito Santo, 2006.
- [16] JUNG, C. R.; OSÓRIO, F. S.; KELBER, C. R.; HEINEN, F. J. Computação embarcada: Projeto e implementação de veículos autônomos inteligentes. *Congresso da Sociedade Brasileira de Computação*, 2005.
- [17] KUC, R.; BARSHAN, B. Navigating vehicles through an unstructured environment with sonar. *IEEE International Conference on Robotics and Automation*, v. 3, p. 1422–1426, 1989.
- [18] ELFES, A. Sonar-based real-world mapping and navigation. *IEEE Journal of Robotic and Automation*, p. 249–265, 1987.
- [19] KHATIB, O. Real-time obstacle avoidance for manipulators and mobile robots. *The International Journal of Robotics Research*, v. 5, p. 90–98, 1986.
- [20] BORENSTEIN, J.; KOREN, Y. Real-time obstacle avoidance for fast mobile robots. *IEEE Transactions on Systems, Man, and Cybernetics*, v. 19, n. 5, p. 1179–1187, 1989.
- [21] BORENSTEIN, J.; KOREN, Y. The vector field histogram - fast obstacle avoidance for mobile robots. *IEEE Journal of Robotics and Automation*, v. 7, n. 3, p. 278–288, jun 1991.
- [22] PAIM, P. K.; JOUVENCEL, B.; BORGES, G. A. Controle reativo para o robô submarino taipan. *VII SBAI/ II IEEE LARS*, sep 2005.
- [23] ZAPATA, R. *Quelques aspects topologiques de la planification de mouvements et des actions réflexes en robotique mobile*. Tese (Doutorado) — Université de Montpellier II, 1991.
- [24] SANCHEZ, A.; CUAUTLE, R.; OSORIO, M. A.; ZAPATA, R. Reactive motion planning for mobile robots. *Mobile Robots Motion Planning, New Challenges*, p. 469–486, 2008.
- [25] A.CACITTI; R.ZAPATA. Reactive behaviours of mobile manipulators based on the dvz approach. *IEEE International Conference on Robotics and Automation (ICRA 2001)*, p. 469–486, 2001.
- [26] OGATA, K. *Discrete-Time Control Systems*. second. Englewood Cliffs, New Jersey: Prentice-Hall International, Inc, 1995.
- [27] FREIRE, E. O. *Desenvolvimento de um Sistema de Sensoriamento Ultra-Sônico para Robô Móvel Orientado a Agentes*. Dissertação (Mestrado) — UFES, 1997.
- [28] BASTOS-FILHO, T. F. *Seguimiento y Análisis de Entornos de Soldadura por Arco Automatizada Mediante Ultrasonidos*. Tese (Doutorado) — Universidad Complutense de Madrid, 1995.

- [29] LAPIERRE, L.; ZAPATA, R.; LEPINAY, P. Simultaneous path following and obstacle avoidance control of a unicycle-type robot. *IEEE International Conference on Robotics and Automation (ICRA 2007)*, v. 1, p. 2617–2622, 2007.

ANEXOS

I. DESCRIÇÃO DO CONTEÚDO DO CD

O CD entregue com este trabalho contém uma cópia digital do relatório, assim como uma pasta com todos os arquivos utilizados para a elaboração do mesmo. Além disso, o arquivo README possui instruções de como executar os códigos desenvolvidos neste trabalho.