UNIVERSIDADE FEDERAL DO CEARÁ

CENTRO DE TECNOLOGIA

DEPARTAMENTO DE ENGENHARIA ELÉTRICA

CURSO DE GRADUAÇÃO EM ENGENHARIA ELÉTRICA

WILLIAM SANTOS FERREIRA

CONTROLE DE TRAJETÓRIA DE ROBÔ MÓVEL COM DESVIO DE OBSTÁCULOS

FORTALEZA

2023

WILLIAM SANTOS FERREIRA

CONTROLE DE TRAJETÓRIA DE ROBÔ MÓVEL COM DESVIO DE OBSTÁCULOS

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Engenharia Elétrica do Centro de Tecnologia da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Engenharia Elétrica.

Orientador: Prof. Dr. Fabrício Gonzalez Nogueira

FORTALEZA

2023

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Sistema de Bibliotecas
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

F444c Ferreira, William Santos.
Controle de Trajetória de Robô Móvel com Desvio de Obstáculos / William Santos Ferreira. – 2023.
44 f. : il. color.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Centro de Tecnologia, Curso de Engenharia Elétrica, Fortaleza, 2023.
Orientação: Prof. Dr. Fabrício Gonzalez Nogueira.
Coorientação: Prof. Dr. Bismark Claure Torrico.

1. Controle de Trajetória. 2. Desvio de Obstáculos. 3. Algoritmo A*. 4. Klancar. I. Título.
CDD 621.3

A minha família e amigos, que me acompanharam durante esta jornada.

AGRADECIMENTOS

A Deus, por toda a força durante esta trajetória.

A minha mãe, pelo apoio, dedicação e investimento.

A minha namorada Emilly, por todo o amor, apoio e confiança que ela me proporcionou ao longo desta jornada.

Aos amigos da Engenharia Elétrica, especialmente a Fernanda, Marina, Sergio e Barata, agradeço pelos momentos marcantes que compartilhamos durante nossa jornada acadêmica.

Ao meu orientador, o professor Dr. Fabrício Gonzalez Nogueira, agradeço por confiar em mim e me proporcionar a oportunidade de fazer parte do GPAR.

Ao professor Carlos Gustavo C. Branco, por toda a ajuda e aprendizado recebido.

Aos meus amigos, Emanuel, Eduardo e Dálete, que me deram suporte e estiveram ao meu lado durante todo o percurso.

Aos meus amigos do trabalho, Paulo, Idemberg e Ivan, pela ajuda e suporte durante esta fase final do curso.

RESUMO

O presente trabalho tem como objetivo estudar e desenvolver um sistema de controle de trajetória para um robô móvel capaz de desviar de obstáculos presentes em seu ambiente. O controle de trajetória é uma área fundamental na robótica móvel, permitindo que o robô se mova de forma autônoma e segura em ambientes complexos. Inicialmente, é apresentada a cinemática do robô móvel, descrevendo as equações de posição que governam o seu movimento. Em seguida, é abordado o problema do desvio de obstáculos, considerando a necessidade de um planejamento de trajetória eficiente. Para solucionar esse desafio, é adotado o uso do algoritmo A* (A-star) como método de planejamento de trajetória. O A* é um algoritmo amplamente utilizado na robótica móvel devido à sua eficiência e capacidade de encontrar caminhos em espaços discretizados. Além disso, é apresentado o controlador *Klancar*, um controlador simples e eficaz para o seguimento da trajetória planejada. Esse controlador utiliza uma matriz de ganhos para ajustar o erro de posição do robô e gerar comandos de velocidade linear e angular. O sistema proposto é avaliado em diferentes cenários, nos quais o robô móvel deve desviar de obstáculos e alcançar um destino pré-definido.

Palavras-chaves: Controle de Trajetória, Desvio de Obstáculos, *Klancar*, Algoritmo A*

ABSTRACT

The present work aims to study and develop a trajectory control system for a mobile robot capable of avoiding obstacles in its environment. Trajectory control is a fundamental area in mobile robotics, enabling the robot to move autonomously and safely in complex environments. Initially, the kinematics of the mobile robot are presented, describing the position equations that govern its motion. Next, the problem of obstacle avoidance is addressed, considering the need for efficient trajectory planning. To solve this challenge, the A* (A-star) algorithm is selected as a trajectory planning method. A* is a widely used algorithm in mobile robotics due to its efficiency and ability to find paths in discretized spaces. Additionally, the *Klancar* controller, a simple and effective controller for tracking the planned trajectory, is introduced. This controller uses a gain matrix to adjust the robot's position error and generate linear and angular velocity commands. The proposed system is evaluated in different scenarios, where the mobile robot must navigate around obstacles and reach a predefined destination.

Keywords: Trajectory Control, Obstacle Avoidance, Klancar, A* Algorithm

LISTA DE FIGURAS

SUMÁRIO

1 INTRODUÇÃO

A robótica móvel tem se destacado como uma área de pesquisa promissora, impulsionada pelos avanços tecnológicos e pela demanda por sistemas autônomos capazes de operar em ambientes complexos e dinâmicos. O controle preciso da trajetória de um robô móvel é um elemento-chave para garantir seu desempenho eficiente e seguro.

A trajetória é uma concepção mais abrangente do que simplesmente um caminho. Enquanto um caminho se refere ao conjunto de pontos no espaço que define uma rota a ser seguida, a trajetória incorpora também o elemento temporal, ou seja, o caminho percorrido ao longo do tempo.

A trajetória não se limita apenas à descrição do percurso físico de um objeto, mas também inclui a consideração do tempo como um fator essencial. Ela engloba a relação entre a posição do objeto e a sua evolução temporal, fornecendo um quadro completo do movimento e deslocamento ao longo de um determinado período (GASPARETTO *et al.*, 2012).

O controle de trajetória envolve a capacidade do robô em seguir uma rota desejada, enquanto evita obstáculos presentes em seu ambiente. Esse desafio é particularmente relevante em aplicações onde os robôs compartilham o espaço com seres humanos ou em ambientes não estruturados, que carecem de uma estrutura regular e previsível.

Dentre as abordagens existentes, o controlador proposto por (KLANCAR *et al.*, 2005) se destaca como uma solução simples e eficaz para o controle de trajetória de robôs móveis não holonômicos, que possuem alguma restrição de movimento. Esse controlador ajusta o erro de posição e orientação do robô, fornecendo comandos de velocidade linear e angular que permitem seu deslocamento suave ao longo da trajetória desejada.

Além disso, o planejamento de trajetória (GASPARETTO *et al.*, 2012) é um aspecto crucial para garantir a navegabilidade do robô. Diferentes algoritmos têm sido desenvolvidos para determinar o caminho mais adequado em um ambiente, considerando restrições e evitando colisões com obstáculos estáticos e móveis. Dentre esses algoritmos, destaca-se o A*, que tem sido amplamente utilizado e adaptado para diversas aplicações de robótica móvel.

1.1 Objetivos

1.1.1 Objetivo Geral

O objetivo do presente trabalho é implementar e avaliar um controlador em um robô móvel diferencial não holonômico, com o intuito de permitir que o robô siga trajetórias pré-definidas, desviando de obstáculos presentes no ambiente.

1.1.2 Objetivos Específicos

- Apresentar de forma detalhada a cinemática do robô móvel diferencial, abordando os conceitos e equações envolvidas na descrição do seu movimento;
- Descrever as ferramentas computacionais utilizadas no processo de aplicação do controlador e obtenção dos resultados;
- Realizar um estudo do algoritmo de planejamento de trajetória conhecido como *A** (*A-Star*), explorando suas principais características;
- Investigar e aplicar um controlador específico, baseado em (KLANCAR *et al.*, 2005), para o robô móvel diferencial, avaliando sua eficácia na execução de trajetórias e no desvio de obstáculos.
- Implementar o algoritmo de planejamento de trajetória e o controlador proposto, utilizando as ferramentas computacionais mencionadas anteriormente, e realizar simulações para obter resultados quantitativos e qualitativos do desempenho do robô móvel seguindo a trajetória desejada e evitando obstáculos.

Esses objetivos específicos têm como propósito orientar a organização e o desenvolvimento do trabalho, abrangendo desde a compreensão teórica dos conceitos e algoritmos envolvidos até a implementação prática e a análise dos resultados obtidos nas simulações do robô móvel diferencial.

1.2 Estado da Arte

Ao longo dos anos, o controle de trajetória em robótica móvel tem sido objeto de estudo e pesquisa contínuos (JIANGDAGGER; NIJMEIJER, 1997). Embora em muitas situações os robôs móveis sejam utilizados em espaços controlados e previsíveis, também existem aplicações em ambientes dinâmicos e imprevisíveis, nos quais a interação com pessoas

e objetos é constante (SVENSTRUP *et al.*, 2010). Nesse contexto, é essencial contar com um controlador que seja capaz de lidar com a complexidade desses ambientes e realizar desvios de obstáculos de forma eficiente.

Uma abordagem de controle de trajetória estudada é o controlador proposto por (KLANCAR *et al.*, 2005), que se destaca por sua simplicidade e efetividade. Esse controlador ajusta a velocidade linear e angular do robô móvel com base no erro de trajetória. Ele foi desenvolvido com o objetivo de permitir que o robô siga uma trajetória desejada, evitando obstáculos no caminho.

No campo do planejamento de trajetória em robótica móvel, a escolha do algoritmo adequado depende do ambiente de estudo e dos requisitos específicos do sistema. Diversos algoritmos têm sido propostos e utilizados com sucesso em diferentes cenários (KARUR *et al.*, 2021). Um dos algoritmos clássicos é o algoritmo de *Dijkstra*, que é amplamente aplicado para encontrar o caminho mais curto em um grafo ponderado. Esse algoritmo é eficiente para ambientes com baixa complexidade e sem considerar obstáculos móveis.

Outro algoritmo popular é o algoritmo A*, que combina a eficiência do algoritmo de *Dijkstra* com a heurística para melhorar o desempenho em termos de tempo computacional. O algoritmo A* é amplamente utilizado em robótica móvel devido à sua capacidade de encontrar soluções ótimas ou aproximadas em tempo real.

O algoritmo D* é uma extensão do A* que permite a atualização eficiente do caminho quando ocorrem mudanças no ambiente. Ele é especialmente útil em cenários onde os obstáculos são dinâmicos ou quando a localização do robô é incerta.

Outro algoritmo relevante é o RRT (SVENSTRUP *et al.*, 2010) (*Rapidly-Exploring Random Tree*) , que utiliza uma abordagem de amostragem aleatória para construir uma árvore de exploração do espaço de configuração. O RRT é eficaz em ambientes com obstáculos complexos e possui a capacidade de encontrar trajetórias viáveis em tempo real.

Além das abordagens mencionadas anteriormente, destaca-se o Campo de Potencial Artificial (APF - Artificial Potential Field) como um algoritmo amplamente utilizado na robótica móvel para a navegação e planejamento de trajetória, conforme abordado por (JR *et al.*, 2019). O APF baseia-se no conceito de campos de potencial para direcionar o robô em direção a um objetivo, ao mesmo tempo em que evita obstáculos presentes no ambiente.

1.3 Estrutura do Trabalho

O presente trabalho está organizado em quatro seções principais, conforme descrito a seguir:

- **Capítulo 1: Introdução**

  Apresenta uma visão geral do tema abordado, justificando a importância do controle de trajetória de robôs móveis com desvio de obstáculos e delineando os objetivos do trabalho.

- **Capítulo 2: Modelagem e Simulação da Cinemática de um Robô Móvel Diferencial**

  Neste capítulo, é realizada a modelagem e simulação da cinemática de um robô móvel diferencial. São apresentadas as equações de movimento que descrevem o comportamento do robô e são discutidos os principais conceitos envolvidos na sua cinemática. Além disso, é apresentado o software *MATLAB*, que será utilizado nas simulações.

- **Capítulo 3: Controle de Trajetória com Desvio de Obstáculos**

  Este capítulo aborda os conceitos fundamentais relacionados ao planejamento de trajetórias em ambientes com obstáculos. É discutido o conceito de *Occupancy Grid Map*, que permite a representação do ambiente em forma de grade ocupada por obstáculos. Em seguida, é apresentado o algoritmo A*, amplamente utilizado para o planejamento eficiente de trajetórias. Além disso, é abordado o controlador de seguimento de trajetória, que utiliza informações do ambiente e da trajetória planejada para guiar o robô móvel de forma segura e eficiente. São realizadas simulações utilizando os conceitos apresentados e os resultados obtidos são analisados e discutidos.

- **Capítulo 4: Conclusão**

  Apresenta um resumo dos principais resultados e contribuições do trabalho. São destacados os pontos fortes e limitações do sistema de controle de trajetória desenvolvido e são apontadas possíveis direções para trabalhos futuros, visando aprimorar o desempenho e a aplicabilidade do robô móvel diferencial no desvio de obstáculos.

2 MODELAGEM E SIMULAÇÃO DA CINEMÁTICA DE UM ROBÔ MÓVEL DIFERENCIAL

2.1 Introdução

O objetivo deste capítulo consiste em realizar a modelagem do robô móvel, que é o elemento central deste estudo. Para tanto, será apresentada a definição do robô móvel, juntamente com a obtenção das suas equações cinemáticas, conforme indicado em (SALEM, 2013). Essa modelagem é fundamental para compreender como ocorre o movimento do robô e quais parâmetros podem ser utilizados no controle da sua trajetória. Além disso, o software *MATLAB* será empregado para a verificação das equações obtidas, permitindo uma análise mais precisa e detalhada do comportamento do robô móvel.

2.2 Robô Móvel Diferencial

Inicialmente, é importante definir o tipo de robô que será utilizado ao longo dos capítulos. O robô escolhido é o robô móvel diferencial, que possui motores dedicados para cada roda (esquerda e direita), além de rodas auxiliares, como a roda castor. Um exemplo desse tipo de robô é ilustrado na Figura 1. No caso do robô "HULK", representado na figura, são utilizadas duas rodas auxiliares, juntamente com as rodas principais esquerda e direita, cada uma com seu próprio atuador. Para mover esse robô para frente, é necessário ativar os dois motores com velocidades iguais. Com uma rotação contrária, o robô começa a se mover na direção oposta. Quando velocidades diferentes são aplicadas, o veículo deixa de se mover em linha reta e passa a seguir um movimento mais curvilíneo. No entanto, essas características iniciais são refletidas nas equações de cinemática, que serão abordadas nos próximos tópicos.

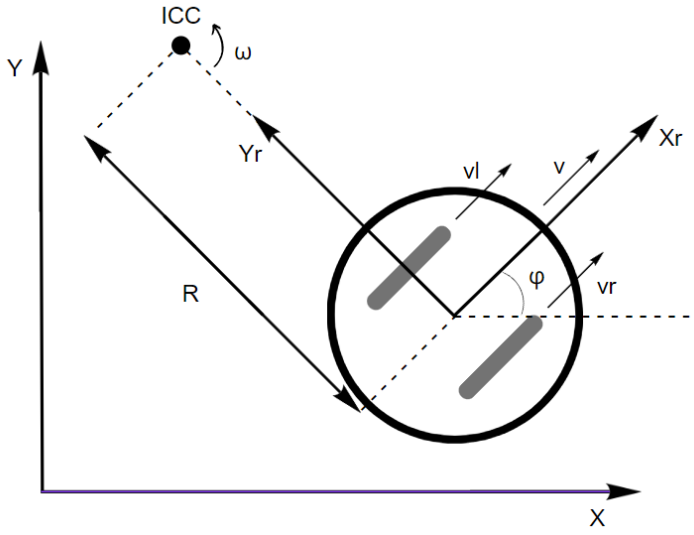
Figura 1 – Exemplo de robô móvel diferencial - Robô HULK, pertencente ao GPAR

Fonte: O próprio autor

2.2.1 Cinemática

Os parâmetros de interesse do robô móvel necessários para o controle são a velocidade linear (v), a velocidade angular (ω) e a orientação (θ). Essas velocidades podem ser subdivididas em velocidades das rodas esquerda e direita, tanto para o movimento linear quanto para o movimento angular. Essa distinção é crucial devido à natureza diferencial do robô, que permite o controle independente de cada motor das rodas. Para compreender a influência das velocidades das rodas, é possível representar o robô como ilustrado na Figura 2.

Figura 2 – Representação das velocidades linear e angular do robô móvel



Fonte: O próprio autor

Para o deslocamento linear, a distância total percorrida pelo robô móvel pode ser calculada pela Equação (2.1), em que Δx_l é a distância percorrida pela roda esquerda, Δx_r é a

distância percorrida pela roda direita e Δx é a distância total percorrida. A velocidade linear pode ser obtida dividindo Δx pelo intervalo de tempo Δt, conforme a Equação (2.2).

$$\Delta x = \frac{\Delta x_l + \Delta x_r}{2} \tag{2.1}$$

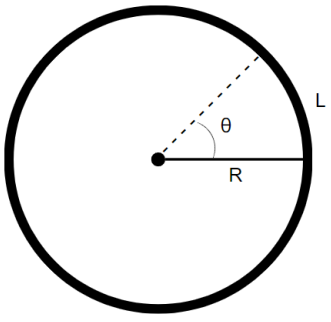
$$v = \frac{v_l + v_r}{2} \tag{2.2}$$

Considerando que o robô está percorrendo uma trajetória circular com raio R e um arco de circunferência L, como ilustrado na Figura 3, o ângulo θ do arco de circunferência é definido pela razão $\frac{L}{R}$ em radianos. A velocidade angular pode ser obtida relacionando a variação angular $\Delta\theta$ com o intervalo de tempo Δt, conforme a Equação (2.4).

$$\frac{\Delta\theta}{\Delta t} = \frac{L}{\Delta t} \cdot \frac{1}{R} \tag{2.3}$$

$$\omega = \frac{v}{R} \tag{2.4}$$

Figura 3 – Arco de circunferência de raio R



Fonte: O próprio autor

Esta relação do raio R pode ser verificada na Figura 2, em que o raio representa a distância do centro da circunferência até o centro do robô. Substituindo a Equação (2.2) na Equação (2.4), obtemos:

$$\omega = \frac{v_l + v_r}{2R} \tag{2.5}$$

A distância do centro da circunferência até a roda esquerda é dada por $(R-d)$, enquanto a distância até a roda direita é $(R+d)$, que são os novos raios. Utilizando a Equação (2.4) com as velocidades das rodas, podemos obter as relações (2.6) e (2.7):

$$v_l = \omega \cdot (R-d) \tag{2.6}$$

$$v_r = \omega \cdot (R+d) \tag{2.7}$$

Isolando o raio R nas Equações (2.6) e (2.7), que é um elemento desconhecido, temos (2.8) e (2.9):

$$R = \frac{v_l + \omega \cdot d}{\omega} \tag{2.8}$$

$$R = \frac{v_r - \omega \cdot d}{\omega} \tag{2.9}$$

Igualando as Equações (2.8) e (2.9):

$$\frac{v_l + \omega \cdot d}{\omega} = \frac{v_r - \omega \cdot d}{\omega} \tag{2.10}$$

$$v_r - v_l = 2\omega \cdot d \tag{2.11}$$

Com isso, podemos obter a relação da velocidade angular com base nas velocidades das rodas, dada pela Equação (2.12):

$$\omega = \frac{v_r - v_l}{2d} \tag{2.12}$$

2.2.2 Representação Matricial

Com os dados de velocidade de cada roda, é possível obter a velocidade linear e angular, como indicado anteriormente. Esta representação pode ser realizada de forma matricial:

$$\begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{-1}{2d} & \frac{1}{2d} \end{bmatrix} \cdot \begin{bmatrix} v_l \\ v_r \end{bmatrix} \tag{2.13}$$

A velocidade linear pode ser decomposta em suas componentes v_x e v_y ao longo dos eixos x e y, respectivamente. Ambas as componentes podem ser representadas em função de v:

$$v_x = v \cdot cos\theta \tag{2.14}$$

$$v_y = v \cdot sen\theta \tag{2.15}$$

Sendo assim podemos representar a derivada das posições em função das velocidades, como indicado em (2.16).

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} cos\theta & 0 \\ sen\theta & 0 \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} v \\ \omega \end{bmatrix} \tag{2.16}$$

Podemos discretizar essa representação considerando um tempo de amostragem T_s e um índice k para cada amostra. Durante os testes, é adotado um $T_s = 10ms$, que é um tempo adequado para os sensores normalmente utilizados nestas aplicações. Usando o método de Euler, podemos aproximar a localização e orientação do robô:

$$\begin{aligned} x(k+1) &= x(k) + v(k) \cdot cos\theta \cdot T_S \\ y(k+1) &= y(k) + v(k) \cdot sen\theta \cdot T_s \\ \theta(k+1) &= \theta(k) + \omega \cdot T_s \end{aligned} \tag{2.17}$$

2.3 Ferramenta de Simulação

Para realizar as simulações da cinemática do robô móvel e implementar o controlador adotado, é necessário utilizar uma plataforma adequada. De acordo com as referências bibliográficas (GALLI R. BARBER; MORENO, 2017) e (GONZALEZ; KLOETZER, 2015), o MATLAB®é um software amplamente utilizado para realizar essas simulações. Uma forma

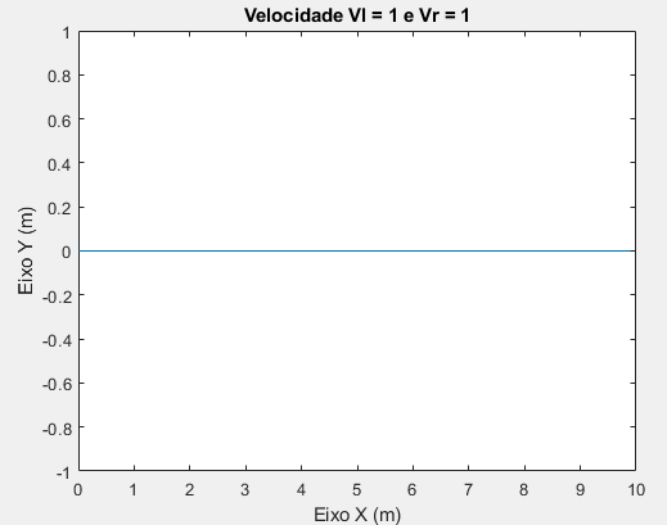
inicial de avaliar a utilidade dessa plataforma é através da representação gráfica das equações de cinemática do robô móvel. Essa representação gráfica permite visualizar de forma clara o movimento do robô e auxilia na compreensão do seu comportamento durante a execução das tarefas. Ao utilizar o MATLAB para representar graficamente as equações de cinemática, é possível obter informações valiosas sobre o desempenho do robô móvel e analisar seu deslocamento, orientação e trajetória em diferentes cenários simulados.

2.3.1 Utilização do MATLAB® para simulação de movimento

Ao discretizar as equações de localização e orientação, é possível obter informações sobre o movimento de um robô móvel diferencial, com base nas velocidades linear e angular fornecidas. A matriz de velocidades (2.13) relaciona as velocidades das rodas esquerda e direita com a velocidade linear e angular do robô. Ao analisar as equações (2.2) e (2.4), observamos que quando $v_l = v_r$, o movimento é puramente linear, pois a velocidade angular ω é igual a zero. Por outro lado, quando $v_l = -v_r$, o movimento é puramente angular, com velocidade linear v igual a zero. Além disso, quando $v_r > v_l$, o robô realiza uma curva para a esquerda, enquanto que quando $v_l > v_r$, ocorre uma curva para a direita. Essas diferentes configurações de velocidades resultam em movimentos distintos do robô.

- $v_r = v_l$: Movimento puramente linear como indica a Figura 4, o movimento é representado pela linha azul;

Figura 4 – Velocidade das rodas iguais

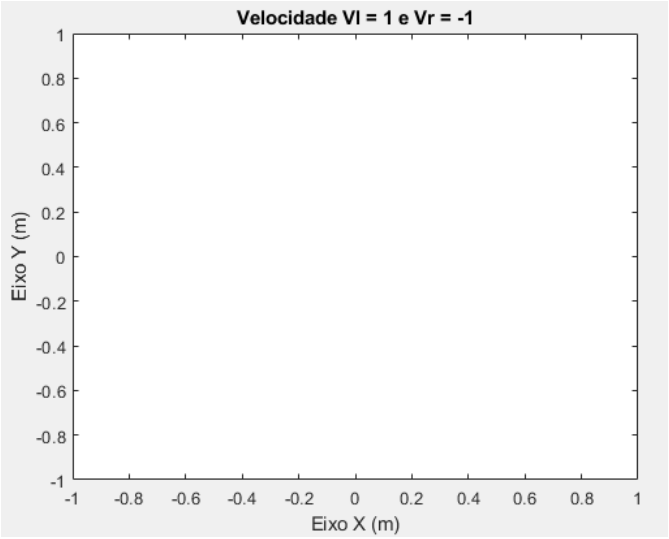


Fonte: O próprio autor

- $v_l = -v_r$: Movimento puramente angular como indica a Figura 5, perceba que não há

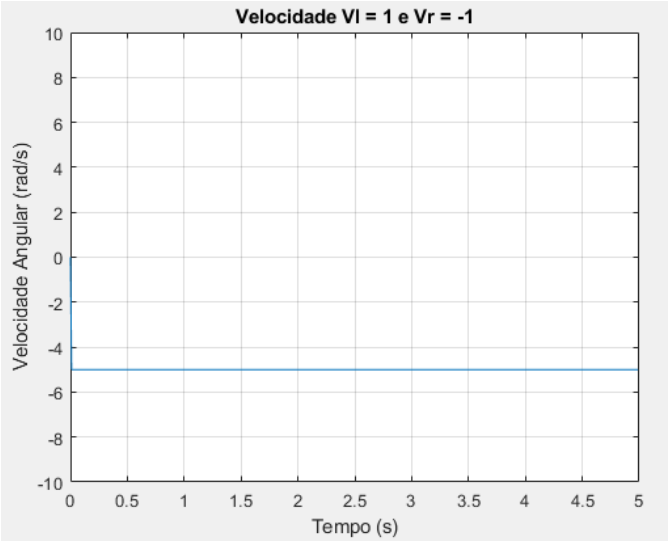
movimento em x ou em y, já que o robô está circulando sobre o eixo z. Na Figura 6 é possível notar que a velocidade angular não é nula, mas se mantém constante ao longo do tempo.

Figura 5 – Velocidade das rodas opostas



Fonte: O próprio autor

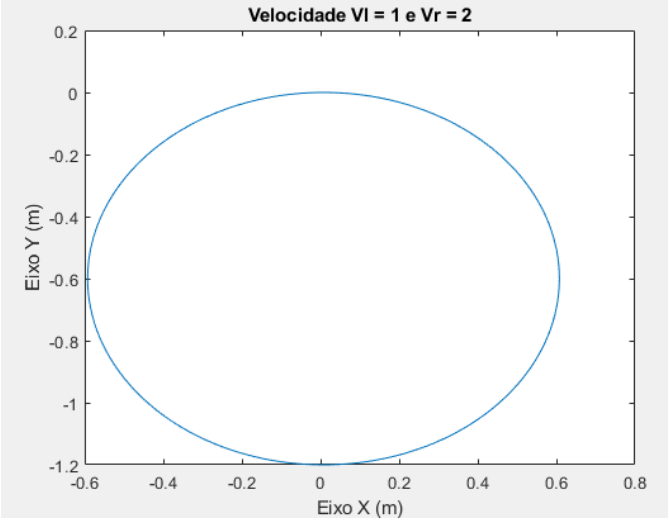
Figura 6 – Velocidade Angular



Fonte: O próprio autor

- $v_r > v_l$: Curva para esquerda como indica a Figura 7. Essa curva, ao longo do tempo, cria uma circunferência, de raio R. Esse raio pode ser determinado pela relação das velocidadades, linear e angular. No caso indicado, $v = 1,5m/s$ e $\omega = 2,5rad/s$, utilizando as equações 2.2 e 2.4. O raio da curvatura é de $R = 0,6m$, como mostra a Figura 7.

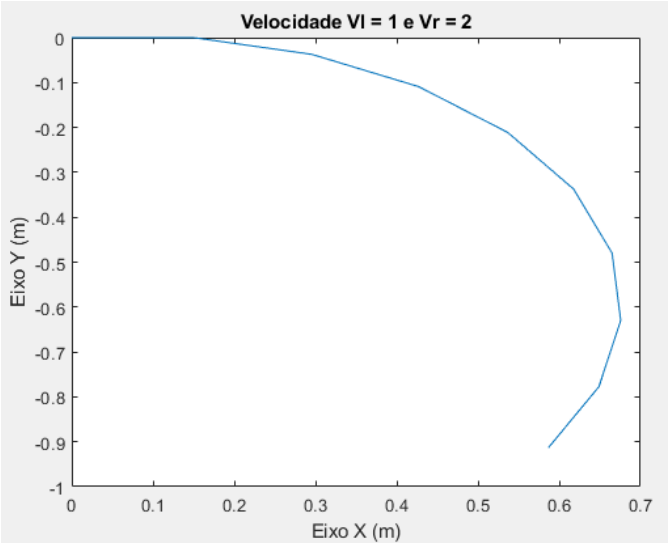
Figura 7 – Velocidade da roda direita maior que da roda esquerda



Fonte: O próprio autor

- $v_l > v_r$: Curva para direita como indica a Figura 8.

Figura 8 – Velocidade da roda esquerda maior que da roda esquerda



Fonte: O próprio autor

2.3.2 Conclusão do Capítulo

No presente capítulo, foi apresentada a cinemática do robô móvel diferencial, fornecendo uma descrição matemática do movimento e orientação do robô com base nas velocidades de suas rodas. Foram apresentadas as equações que estabelecem a relação entre a velocidade linear e angular do robô e as velocidades individuais das rodas, proporcionando uma base teórica sólida para compreender o comportamento cinemático do robô.

Adicionalmente, destacou-se a importância do MATLAB® como uma ferramenta

computacional poderosa para simular as equações de cinemática do robô móvel. O MATLAB® oferece recursos avançados e uma interface amigável, o que facilita a implementação e visualização dessas equações. Essa plataforma permite uma análise detalhada do movimento do robô em diversas situações, contribuindo para um estudo aprofundado do seu comportamento cinemático.

3 CONTROLE DE TRAJETÓRIA COM DESVIO DE OBSTÁCULOS

3.1 Introdução

Este capítulo apresenta o conceito do *Occupancy Grid Map*, e como utilizar ele para criar um algoritmo de planejamento de trajetória com desvio de obstáculos. Além disso, é apresentado o controlador a ser utilizado para fazer o robô móvel seguir a trajetória criada, utilizando o software *MATLAB* para fazer simulações tanto do algoritmo quanto do controlador.

3.2 *Occupancy Grid Map*

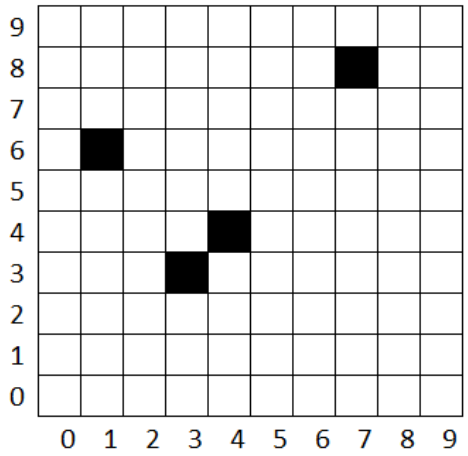
O ambiente no qual o robô se desloca pode apresentar obstáculos ou ser um ambiente livre de obstruções. A representação desse espaço é obtida a partir dos dados dos sensores do robô e pode ser tratada de duas formas: mapa *offline* e mapa *online*.

No caso do mapa *offline*, este é construído a partir de informações prévias e não sofre atualizações durante o movimento do robô. Por outro lado, o mapa *online* é atualizado conforme o robô se move no ambiente, incorporando informações sensoriais em tempo real.

Para que o robô possa distinguir entre objetos e caminhos livres, é necessário definir critérios de classificação. Uma abordagem comum é representar o espaço do ambiente por meio de uma grade, onde cada célula é identificada pelas coordenadas X e Y. Essa representação tabular permite indicar quais células são ocupadas por objetos e quais estão livres para o deslocamento do robô.

Essa estrutura de grade facilita a análise e o planejamento de trajetórias, permitindo que o robô tome decisões com base nas informações do ambiente representadas na tabela. Dessa forma, o mapa do ambiente se torna uma importante ferramenta para a navegação e a interação segura do robô móvel em seu entorno.

Figura 9 – Representação de um mapa de grade de células



Fonte: O próprio autor

A Figura 9 apresenta a representação de um mapa em grade conhecido como *Grid Map*. Nessa representação, é possível identificar as células ocupadas e as células livres por meio de suas coordenadas. Por exemplo, na coordenada (1,6), observamos uma célula preenchida em preto, que indica a presença de um obstáculo. Já na coordenada (2,8), temos uma célula em branco, representando um espaço livre.

Essa representação segue o conceito de *Occupancy Grid Map*, no qual cada célula é atribuída a um valor que indica se está ocupada ou livre, conforme mencionado em (ELFES, 1989). Na Figura 10, as células livres são representadas pelo valor 1, enquanto as células ocupadas têm o valor 0. É importante ressaltar que a precisão do mapa criado aumenta à medida que a dimensão das células diminui.

Figura 10 – Exemplo de *Occupancy Grid Map*

9	1	1	1	1	1	1	1	1	1	1
8	1	1	1	1	1	1	1	0	1	1
7	1	1	1	1	1	1	1	1	1	1
6	1	0	1	1	1	1	1	1	1	1
5	1	1	1	1	1	1	1	1	1	1
4	1	1	1	1	0	1	1	1	1	1
3	1	1	1	0	1	1	1	1	1	1
2	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
0	1	1	1	1	1	1	1	1	1	1
	0	1	2	3	4	5	6	7	8	9

Fonte: O próprio autor

3.3 Planejamento de Trajetória

O planejamento de trajetória desempenha um papel fundamental na criação de um caminho viável do ponto A ao ponto B, levando em consideração o mapa previamente construído e armazenado no robô. É importante destacar que nem sempre é possível escolher qualquer caminho para que o robô alcance seu destino, uma vez que o ambiente deve ser considerado.

Existem diferentes algoritmos e métodos para o planejamento de caminhos, como mencionado em (KLANCAR, 2017). Um desses algoritmos amplamente utilizado é o conhecido como algoritmo A*. O algoritmo A* combina a busca em largura com uma heurística que estima o custo restante para atingir o objetivo, permitindo encontrar um caminho ótimo ou próximo do ótimo em termos de custo.

*3.3.1 Algoritmo A**

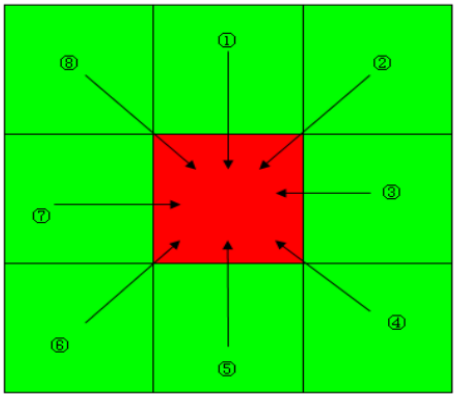
Ao considerar um mapa, como o ilustrado na Figura (10), o robô precisa navegar pelas células até chegar ao destino desejado. Em cada movimento, existem oito células vizinhas que podem ser escolhidas, conforme apresentado na Figura (11). No entanto, é essencial atribuir um peso a cada célula para garantir que a trajetória definida seja eficiente. Para isso, o algoritmo A* calcula o custo de cada posição em relação ao destino.

Uma abordagem comum é utilizar a distância euclidiana entre a célula atual e o destino como medida de custo, conforme demonstrado na Equação (3.1). Esse cálculo leva em consideração a geometria do espaço, $R2$, permitindo ao algoritmo avaliar a proximidade entre as células e o destino. Além disso, é necessário verificar se a célula está livre, ou seja, se não está bloqueada, utilizando as informações do *Occupancy Grid Map*. Isso garante que o algoritmo leve em consideração a acessibilidade do caminho e evite obstáculos durante o planejamento da trajetória.

A combinação do cálculo de custo baseado na distância euclidiana e a verificação da disponibilidade das células no *Occupancy Grid Map* permite que o algoritmo A* encontre trajetórias eficientes e seguras para o robô, otimizando sua movimentação no ambiente.

$$D = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{3.1}$$

Figura 11 – Células testadas em cada movimento



Fonte: (PENG *et al.*, 2015)

A função utilizada para calcular o custo total no algoritmo A* leva em consideração não apenas a distância euclidiana, mas também a distância já percorrida. Assim, a função final pode ser representada como indicado em (3.2).

$$F(n) = G(n) + H(n) \tag{3.2}$$

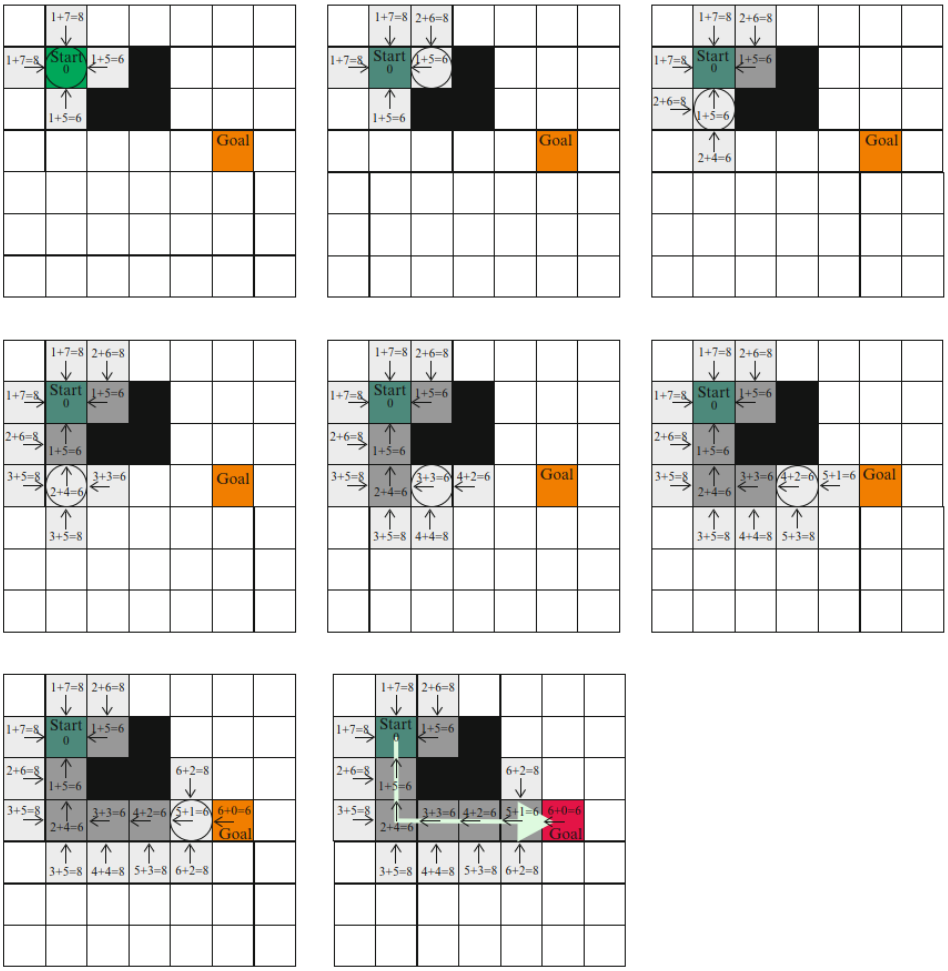
Nessa equação, a função $F(n)$ representa o custo total do movimento da célula atual para a próxima célula. A função $G(n)$ indica o custo de deslocamento do ponto inicial até a célula atual, considerando a distância percorrida até o momento. Já a função $H(n)$ é o valor da distância da célula atual até o destino, calculada utilizando a equação (3.1), que representa a distância euclidiana entre as coordenadas das células.

A combinação das funções $G(n)$ e $H(n)$ na função $F(n)$ permite ao algoritmo A* avaliar tanto o custo acumulado do movimento até o momento quanto uma estimativa do custo restante para alcançar o destino. Dessa forma, o algoritmo é capaz de selecionar a próxima célula com base no menor custo total, buscando eficiência e otimização na determinação da trajetória.

Na Figura (12), é apresentado um exemplo da aplicação do algoritmo A*. O ponto inicial é indicado como *Start*, e o ponto de destino como *Goal*. Nesse exemplo, em contraste com a referência citada em (PENG *et al.*, 2015), apenas quatro células são analisadas em cada movimento. Para cada célula, o algoritmo calcula o custo $G(n)$, que inicialmente é igual a 1, pois corresponde ao número de células percorridas a partir do ponto inicial. No entanto, o valor de $H(n)$ é diferente para cada célula, pois reflete a distância da célula até o destino, considerando a equação 3.1. Com base nos valores de $G(n)$ e $H(n)$, o algoritmo determina o custo total $F(n)$ e

escolhe a célula mais adequada para o próximo movimento. Esse processo é repetido para cada célula até que o destino seja alcançado.

Figura 12 – Exemplo do algoritmo A* sendo aplicado



Fonte: (KLANCAR, 2017)

3.3.2 Algoritmo no MATLAB

No contexto do *MATLAB*, é possível implementar o algoritmo A* e utilizar o *Occupancy Grid Map* para obter uma trajetória do ponto inicial ao ponto final. Ao executar o algoritmo, são estabelecidos três principais objetivos:

- Obter uma trajetória que alcance o destino desejado: O algoritmo A* busca encontrar o caminho mais curto ou mais eficiente do ponto inicial ao ponto final. Levando em consideração os custos de deslocamento e a estimativa heurística de distância, ele realiza iterações para determinar as células a serem percorridas e, assim, alcançar o destino desejado.
- Obter uma trajetória suave: A suavidade da trajetória é uma preocupação importante para

a navegação de robôs móveis. Isso resulta em um movimento mais estável e controlado do robô ao longo do caminho definido.

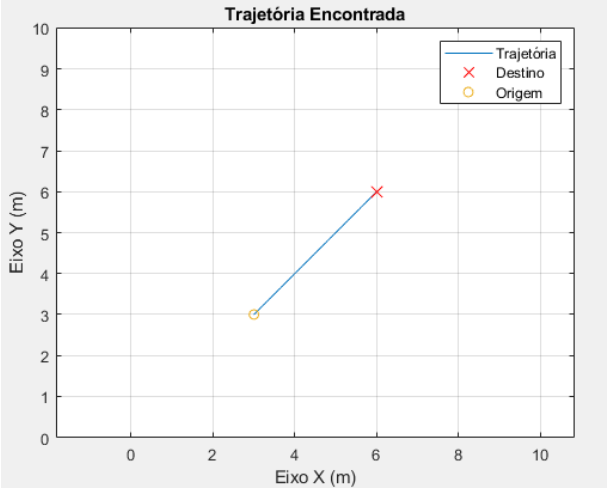
- Obter uma trajetória que desvie dos obstáculos: O algoritmo A* é capaz de contornar obstáculos no ambiente. Ao utilizar o *Occupancy Grid Map*, ele verifica quais células estão ocupadas e evita incluí-las na trajetória. Isso permite que o robô desvie dos obstáculos de forma inteligente, escolhendo caminhos livres que minimizem a colisão e garantam a segurança durante a navegação.

O primeiro caso, em que o mapa não contém obstáculos, pode ser visualizado na Figura (13). Nessa situação, o algoritmo A* é capaz de encontrar uma trajetória direta e sem desvios, pois não há obstruções no caminho entre o ponto inicial e o ponto final. Isso é evidenciado pela linha reta que conecta os dois pontos no mapa.

A ausência de obstáculos facilita a busca pelo caminho mais curto ou eficiente, pois não há necessidade de desviar ou contornar objetos. O algoritmo avalia as células disponíveis e escolhe aquelas que minimizam os custos de deslocamento e estimativa de distância, permitindo que o robô siga um trajeto direto em direção ao destino.

Esse cenário é comum em ambientes amplos e desimpedidos, nos quais o robô pode se movimentar livremente sem encontrar obstáculos físicos ou restrições de acesso. A trajetória obtida nesse caso geralmente é rápida e direta, permitindo que o robô chegue ao destino de forma eficiente.

Figura 13 – Caso 01, Sem Obstáculos - Ponto inicial (3,3) e Ponto Final (6,6)



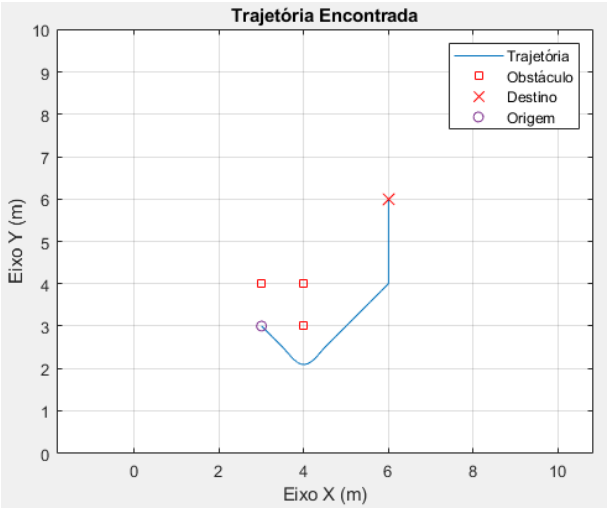
Fonte: O próprio autor

O segundo caso, representado na Figura (14), apresenta um ambiente com a presença de obstáculos. Nessa situação, o algoritmo A* é capaz de encontrar uma trajetória que se ajusta

aos obstáculos e desvia deles, permitindo que o robô alcance o ponto final de forma segura e eficiente.

Ao analisar a trajetória resultante, é possível observar que ela se adapta aos obstáculos, contornando-os para evitar colisões. Essa capacidade de desviar de obstáculos é fundamental para garantir a integridade do robô e evitar danos ou bloqueios durante o percurso. Para suavizar a trajetória, foi utilizado o filtro *Savitzky-Golay*. A aplicação do filtro resulta em uma trajetória mais suave, eliminando pequenas flutuações e ajustando os movimentos do robô de forma mais harmônica. Isso contribui para um deslocamento mais estável e controlado, reduzindo possíveis vibrações ou movimentos bruscos.

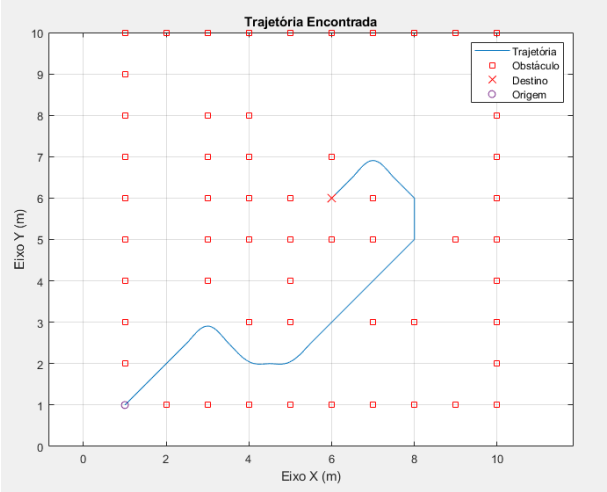
Figura 14 – Caso 02, Com Obstáculos - Ponto inicial (3,3) e Ponto Final (6,6)



Fonte: O próprio autor

O terceiro caso, ilustrado na Figura (15), apresenta um mapa com uma maior densidade de obstáculos em comparação aos casos anteriores. Apesar desse aumento na complexidade do ambiente, o algoritmo A* continua sendo capaz de encontrar uma trajetória viável para alcançar o ponto de destino.

Figura 15 – Caso 03, Com Obstáculos - Ponto Inicial (1,1) e Ponto Final (6,6)



Fonte: O próprio autor

3.4 Controle de Seguimento de Trajetórias

Para garantir que o robô móvel consiga seguir uma trajetória de maneira precisa e controlada, é necessário utilizar um controlador adequado. Um dos controladores comumente utilizados em robôs móveis diferenciais não holonômicos é o controlador Klancar, nomeado em referência ao autor que o propôs (KLANCAR *et al.*, 2005).

O controlador Klancar é projetado para lidar com as restrições de movimento dos robôs móveis diferenciais, que são incapazes de se mover livremente em todas as direções. Essas restrições surgem devido à configuração das rodas do robô, que não permitem que ele realize movimentos laterais. O controlador Klancar utiliza técnicas de controle feedback para gerar comandos de velocidade apropriados para as rodas esquerda e direita do robô, de modo a seguir a trajetória desejada. Ele leva em consideração as informações sobre a posição e orientação atuais do robô, bem como a referência de trajetória fornecida. Ao comparar a posição e orientação atuais com a referência de trajetória, o controlador calcula os erros de posicionamento e orientação. Com base nesses erros, ele gera comandos de velocidade diferenciais para as rodas, ajustando-os de forma a minimizar os erros e permitir que o robô siga a trajetória desejada.

3.4.1 Controlador Klancar

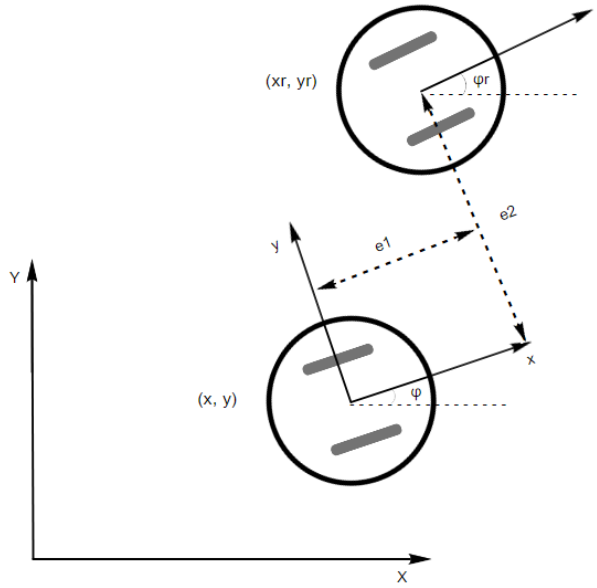
A trajetória de referência é representada por um vetor $q_r = (x_r, y_r, \theta_r)$, que indica as coordenadas e a orientação desejadas para o robô. Por outro lado, a posição atual do robô é representada por $q = (x, y, \theta)$, que representa as coordenadas e a orientação atual do robô. Ao

analisar a Figura (16), é possível observar a presença de um erro de posicionamento entre o robô real e o robô de referência. Esse erro é utilizado na análise do controlador, a fim de realizar os ajustes necessários para minimizá-lo. Inicialmente, considerando apenas o robô real em sua posição q, podemos perceber que os eixos do robô estão rotacionados em relação aos eixos X e Y do mundo por um ângulo θ. Essa rotação pode ser representada por uma matriz de rotação:

$$\begin{bmatrix} X \\ Y \\ \theta \end{bmatrix} = \begin{bmatrix} cos\theta & sen\theta & 0 \\ -sen\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} \tag{3.3}$$

Essa matriz de rotação, representada na equação matricial (3.3), permite converter as coordenadas de um ponto do referencial do robô para o referencial do mundo ou vice-versa. Ela descreve a relação entre as coordenadas de um ponto após a rotação do eixo z. Ao utilizar a matriz de rotação, é possível realizar transformações de coordenadas entre o sistema de referência do robô e o sistema de referência do mundo, levando em consideração a orientação do robô em relação aos eixos do sistema de coordenada absoluto.

Figura 16 – Erro da posição atual e posição de referência



Fonte: O próprio autor

O erro entre a posição atual do robô e a posição de referência pode ser representado por um vetor $q_e = (e_1, e_2, e_3)$, onde e_1, e_2 e e_3 representam os erros nas coordenadas x, y e θ,

respectivamente. Esses erros indicam a diferença entre a posição desejada e a posição atual do robô. A matriz do erro, denotada como E, é definida da seguinte forma:

$$\begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix} = \begin{bmatrix} cos\theta & sen\theta & 0 \\ -sen\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x_r - x \\ y_r - y \\ \theta_r - \theta \end{bmatrix} \tag{3.4}$$

A partir da equação matricialde posição (2.16) e a equação matricial de erro (3.4), é possível obter a seguinte matriz como indica (KLANCAR *et al.*, 2005):

$$\begin{bmatrix} \dot{e_1} \\ \dot{e_2} \\ \dot{e_3} \end{bmatrix} = \begin{bmatrix} cos(e_3) & 0 \\ sen(e_3) & 0 \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} u_{r1} \\ u_{r2} \end{bmatrix} + \begin{bmatrix} -1 & e_2 \\ 0 & -e_1 \\ 0 & -1 \end{bmatrix} \cdot \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \tag{3.5}$$

O valor de u_{r1} é a velocidade linear de referência, enquanto que u_{r2} é o valor da velocidade angular de referência. As entradas u_1 e u_2 podem ser expressas como:

$$u_1 = u_{r1} \cdot cos(e_3) - v_1 \tag{3.6}$$

$$u_2 = u_{r2} - v_2 \tag{3.7}$$

Os primeiros elementos da equação (3.6) são referentes a alimentação da trajetória de referência, enquanto que, a segunda parte é referente aos valores de velocidade do robô. Utilizando as informações da equação (3.6) é possível reescrever a matriz 3.5, resultando em:

$$\begin{bmatrix} \dot{e_1} \\ \dot{e_2} \\ \dot{e_3} \end{bmatrix} = \begin{bmatrix} 0 & u_2 & 0 \\ -u_2 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix} + \begin{bmatrix} 0 \\ sen(e_3) \\ 0 \end{bmatrix} \cdot u_{r1} + \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} \tag{3.8}$$

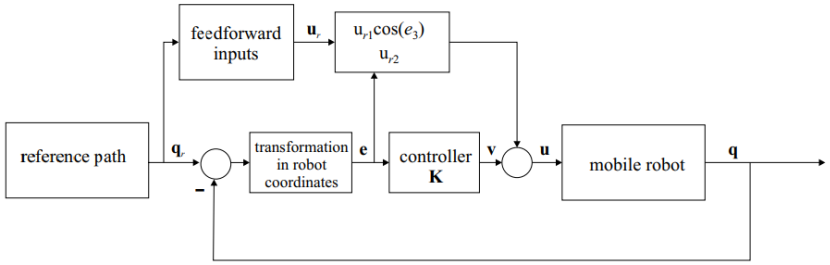
A equação matricial de erro (3.8) pode ser linearizada em torno do ponto de operação ($e_1 = 0$, $e_2 = 0$, $e_3 = 0$, $v_1 = 0$ e $v_2 = 0$) para obter um modelo linear.

$$\Delta\dot{e} = \begin{bmatrix} 0 & u_{r2} & 0 \\ -u_{r2} & 0 & u_{r1} \\ 0 & 0 & 0 \end{bmatrix} \cdot \Delta e + \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} \cdot \Delta v \tag{3.9}$$

A Figura 17 ilustra o diagrama do controlador, incluindo as malhas de realimentação e de *feedforward*. Nesse diagrama, a referência q_r e os valores atuais do robô q são utilizados como entradas. A referência é empregada tanto na realimentação quanto no *feedforward* do controlador. Os erros e_1, e_2 e e_3 são calculados e fornecidos como entradas ao controlador. Para obter as velocidades de controle, o controlador utiliza a matriz K, que é a matriz do controlador proposta por *Klancar*.

$$\begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \begin{bmatrix} -k_1 & 0 & 0 \\ 0 & -sign(u_{r1})k_2 & -k_3 \end{bmatrix} \cdot \begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix} \tag{3.10}$$

Figura 17 – Esquemático de controle do robô móvel



Fonte: (KLANCAR *et al.*, 2005)

Para obter os parâmetros do controlador K, em (KLANCAR *et al.*, 2005) é comparado o polinômio característico real e o desejado, como indica a equação 3.11.

$$(s+2\zeta\omega_n)(s^2+2\zeta\omega_n s+\omega_n^2) \tag{3.11}$$

Assim como os sistemas de segunda ordem, o ζ é o coeficiente de amortecimento, com valores que variam de 0 a 1. E ω_n é a frequência natural, com valores maiores que 0. Considerando o modelo de espaço de estados 3.9 e a lei de controle 3.10, o polinômico característico de malha fechada com o espaço de estados é:

$$det(sI-A+BK) = s^3+(k_1+k_3)s^2+(k_1k_3+k_2u_{r1}+u_{r2}^2)s+k_1k_2eu_{r1}+k_3u_{r2}^2 \tag{3.12}$$

Comparando o grau dos polinômios de 3.11 e 3.12 é possível encontrar que:

$$k_1 + k_3 = 4\zeta\omega_n \tag{3.13}$$

$$k_1 k_3 + k_2 u_{r1} + u_{r2}^2 = 4\zeta^2\omega_n^2 + \omega_n^2 \tag{3.14}$$

$$k_1 k_2 u_{r1} + k_3 u_{r2}^2 = 2\zeta\omega_n^3 \tag{3.15}$$

A matriz K utilizada no controlador, composta pelos valores k_1, k_2 e k_3, é determinada com base em dois parâmetros importantes: g e ζ. Essas relações são obtidas a partir da equações 3.13, 3.14 e 3.15, como indicado em (KLANCAR *et al.*, 2005) e (LUCA *et al.*, 2002). Embora esses valores possam ser definidos pelo usuário, é importante ressaltar que o parâmetro $\omega_n(t)$, presente na equação (3.16), também desempenha um papel fundamental. Para calcular $\omega_n(t)$, pode-se utilizar a equação (3.18), conforme descrito em (KLANCAR *et al.*, 2005). Essa equação permite determinar a frequência natural ω_n do sistema, que está relacionada à rapidez com que o robô pode acompanhar a trajetória de referência.

Portanto, ao definir os valores de g, ζ e utilizar a equação (3.18), é possível calcular os parâmetros k_1, k_2 e k_3 da matriz K, que serão utilizados no controlador para ajustar o desempenho do robô e garantir o acompanhamento adequado da trajetória desejada.

$$k_1 = k_3 = 2\zeta\omega_n(t) \tag{3.16}$$

$$k_2 = g \cdot |u_{r1}(t)| \tag{3.17}$$

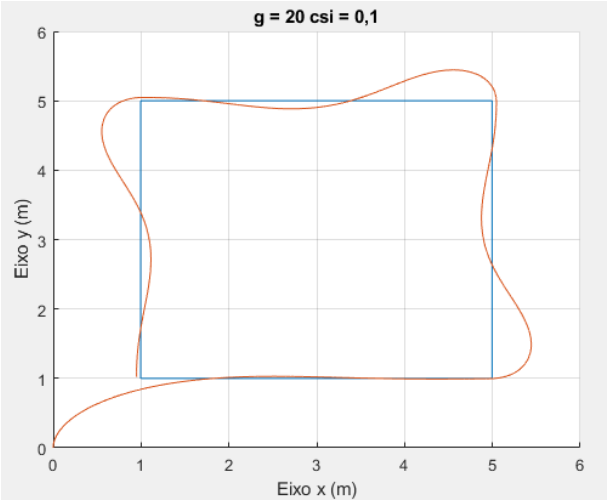
$$\omega_n(t) = \sqrt{(u_{r2}(t))^2 + g(u_{r1}(t))^2} \tag{3.18}$$

3.4.2 Seguimento de Trajetória

Durante os experimentos computacionais realizados para avaliar o desempenho do controlador *Klancar* e definir os parâmetros g e ζ, observou-se que a escolha de um valor baixo para ζ resultou em uma trajetória do robô com alta oscilação. Isso indica que o sistema está respondendo de forma instável, com respostas rápidas e excessivas, levando a movimentos oscilatórios indesejados. A alta oscilação na trajetória pode ser explicada pelo fato de que ζ é o fator de amortecimento do sistema. Um valor baixo para ζ implica em um baixo amortecimento,

o que permite que o sistema responda rapidamente às variações, mas também torna-o mais propenso a oscilações indesejadas. Portanto, ao observar uma trajetória com alta oscilação na figura (18), é possível concluir que é necessário aumentar o valor de ζ para melhorar a estabilidade do sistema e reduzir as oscilações.

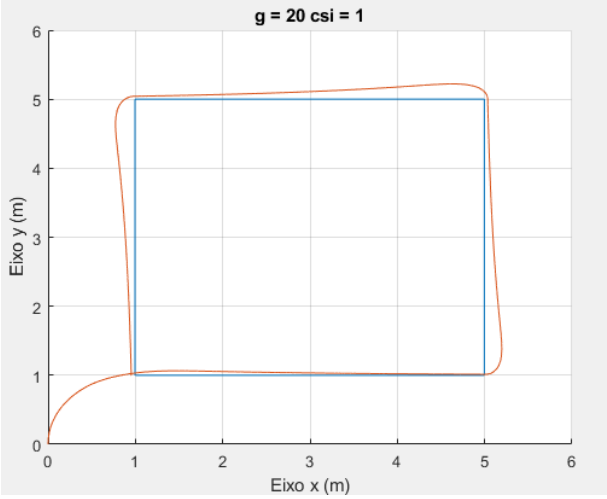
Figura 18 – Teste 01 - g = 20 csi = 0,1



Fonte: O próprio autor

Na Figura (19), observamos que ao definir $\zeta = 1$, a trajetória do robô se aproxima da referência desejada. No entanto, é possível notar que há um erro constante entre a trajetória do robô e a trajetória de referência. Esse erro constante pode ser resultado de uma compensação insuficiente do controlador em relação a algumas características do sistema ou perturbações presentes no ambiente. Um valor de $\zeta = 1$ indica um amortecimento crítico, o que significa que o sistema responde de forma rápida e sem oscilações excessivas, mas pode não ser capaz de eliminar completamente o erro constante.

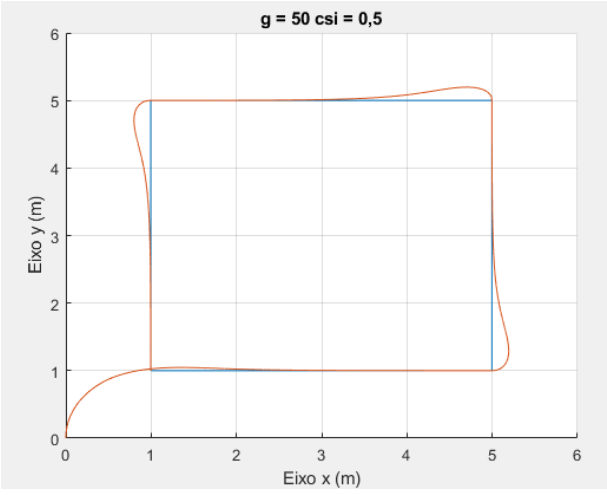
Figura 19 – Trajetória 02 - g = 20 csi = 1



Fonte: O próprio autor

Na Figura (20), podemos observar que ao utilizar os parâmetros $g = 50$ e $\zeta = 0,5$, obtemos um bom desempenho do controlador *Klancar* para a trajetória indicada. Nessa configuração, é possível notar que o robô segue a trajetória de referência de forma suave e precisa, minimizando tanto as oscilações quanto o erro constante. O parâmetro g influencia a magnitude da resposta do sistema em relação ao erro constante. Um valor maior de g permite uma compensação mais eficiente, reduzindo o erro constante. No caso em questão, o valor de $g = 50$ foi escolhido para obter um controle mais preciso e minimizar o erro. Por sua vez, o parâmetro ζ está relacionado ao amortecimento do sistema. Um valor de $\zeta = 0,5$ indica um amortecimento moderado, equilibrando a rapidez da resposta do sistema e a estabilidade. Essa configuração permite que o robô siga a trajetória de referência de forma suave, evitando oscilações indesejadas.

Figura 20 – Trajetória 03 - g = 50 csi = 0.5



Fonte: O próprio autor

3.4.3 Desvio de Obstáculos

O algoritmo A* permite criar a trajetória evitando os obstáculos do mapa, essa é a trajetória de referência utilizada para realizar o controle do robô móvel. Com isso, foi realizado três experimentos com o algoritmo A* em conjunto com o controlador *Klancar*. Os parâmetros utilizados durante os testes são:

- $g = 50$. Esse parâmetro foi ajustado para controlar a influência do erro constante no sistema. Um valor alto para g permite uma compensação mais rápida do erro;
- $\zeta = 0,5$. Esse parâmetro controla o amortecimento do sistema. Um valor menor para ζ resulta em um amortecimento menor e uma resposta mais rápida, enquanto um valor maior aumenta o amortecimento e suaviza a resposta;
- $v_{max} = 2m/s$. Esse parâmetro define a velocidade linear máxima permitida para o robô. Ele limita a velocidade de deslocamento do robô ao longo da trajetória;
- $\omega_{max} = 2rad/s$. Esse parâmetro define a velocidade angular máxima permitida para o robô. Ele limita a taxa de rotação do robô ao seguir a trajetória;
- $T_{amostragem} = 10ms$. Esse parâmetro determina o intervalo de tempo entre as amostras do controlador. Ele define a frequência com que o controlador é atualizado e os cálculos de controle são realizados.

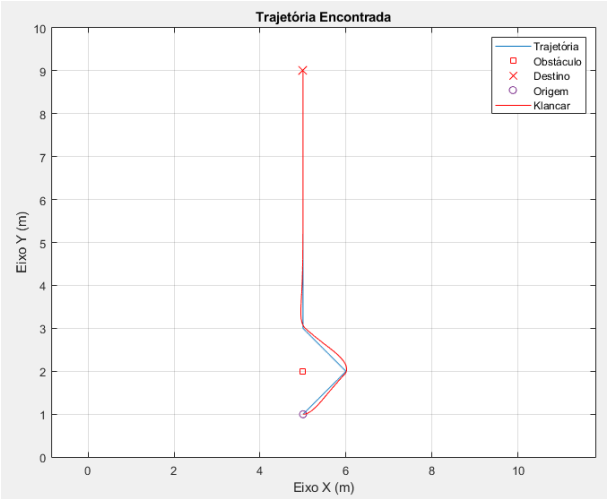
A primeira trajetória foi considerado o posição inicial $(5,1)$ para um $\theta = 0$ e o posição de destino $(5,9)$. Foi colocado um obstáculo ao longo do trajeto para ilustrar que a trajetória busca desviar do mesmo e alcançar o destino. Com a trajetória criada, os dados foram usados no controlador *Klancar*, que permitiu o seguimento da trajetória, como indica na Figura (21).

De acordo com os dados de velocidade linear e angular apresentados nas Figuras (22) e (23), é possível confirmar que as velocidades da trajetória 01 analisada estão de acordo com as velocidades de referência definidas pelo algoritmo A*. Observando a Figura (11), podemos verificar que, para as células 1, 3, 5 e 7, a velocidade linear de referência é de $0,2m/s$, enquanto para as outras células a velocidade linear de referência é de $0,283m/s$.

Ao analisar as Figuras (22) e (23), podemos verificar que as velocidades lineares e angulares não estão saturadas e apresentam valores correspondentes aos movimentos realizados pelo robô durante a trajetória. Isso indica que o controlador *Klancar* conseguiu ajustar as velocidades do robô de acordo com as velocidades de referência, permitindo que o robô siga a trajetória planejada de forma adequada. É importante ressaltar que a ausência de saturação nas

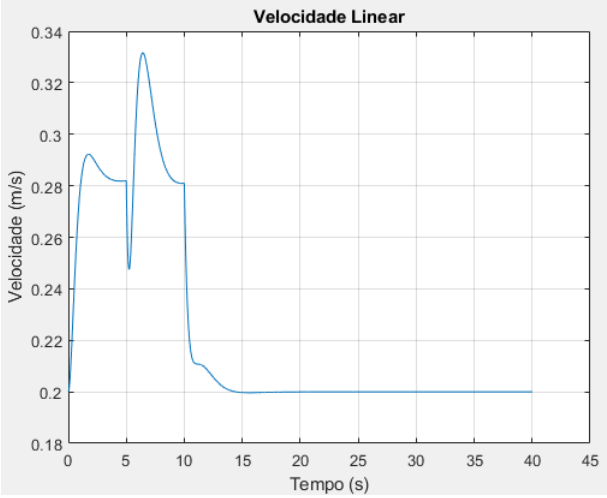
velocidades é um indicativo positivo, pois indica que o controlador está ajustando as velocidades de acordo com as necessidades da trajetória e do sistema em geral. Isso contribui para um seguimento preciso da trajetória e um desempenho satisfatório do robô móvel.

Figura 21 – Seguimento de Trajetória - Trajetória 01



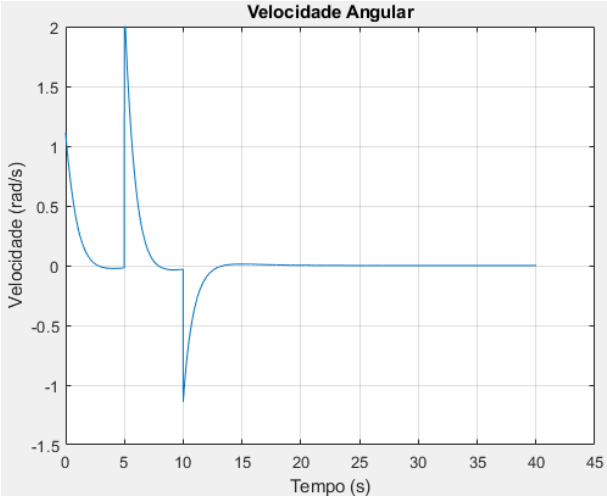
Fonte: O próprio autor

Figura 22 – Velocidade Linear - Trajetória 01



Fonte: O próprio autor

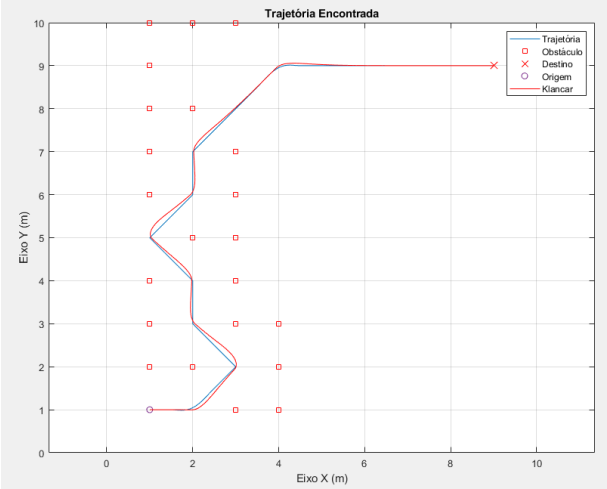
Figura 23 – Velocidade Angular - Trajetória 02



Fonte: O próprio autor

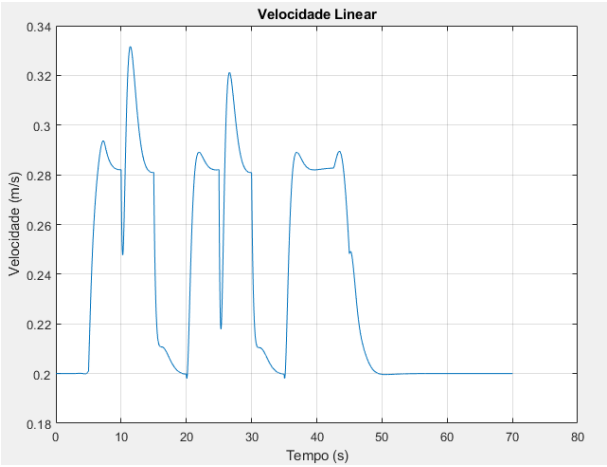
A segunda trajetória, com posição inicial $(1,1)$ e posição de destino $(9,9)$, apresentou diversos obstáculos ao longo do caminho. Na Figura (24), podemos observar que a trajetória planejada pelo algoritmo A* foi obtida e o controlador *Klancar* conseguiu seguir essa trajetória, contornando os obstáculos de forma eficiente. Analisando as velocidades, podemos notar que a velocidade angular apresenta vários picos, especialmente nas mudanças de direção que o robô está realizando. Isso ocorre devido à diferença de velocidade entre as rodas do robô, como indicado na equação (2.12). Nas outras movimentações, onde não há mudança de direção, as velocidades das rodas são iguais, ou seja, $\omega = 0$. A Figura (19) ilustra a velocidade linear da trajetória 02 analisada. Podemos observar que nos pontos de mudança de direção, o controlador tenta compensar essa alteração aumentando a velocidade do robô, de forma a permitir que ele retorne à trajetória de referência. Essa variação de velocidade é notória ao observar a Figura (25), que representa a velocidade linear.

Figura 24 – Seguimento de Trajetória - Trajetória 02



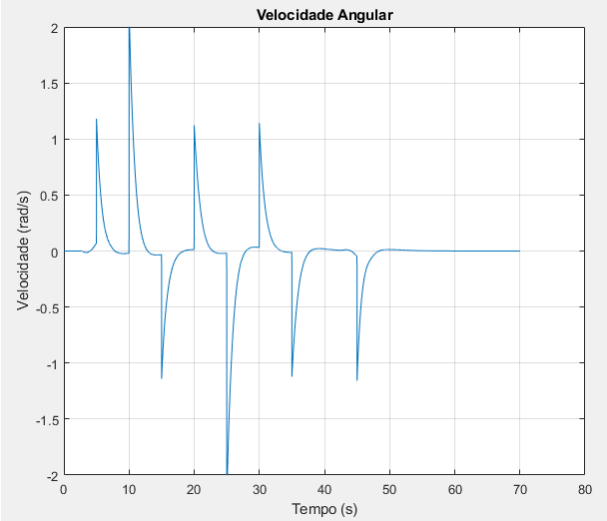
Fonte: O próprio autor

Figura 25 – Velocidade Linear - Trajetória 02



Fonte: O próprio autor

Figura 26 – Velocidade Angular - Trajetória 02

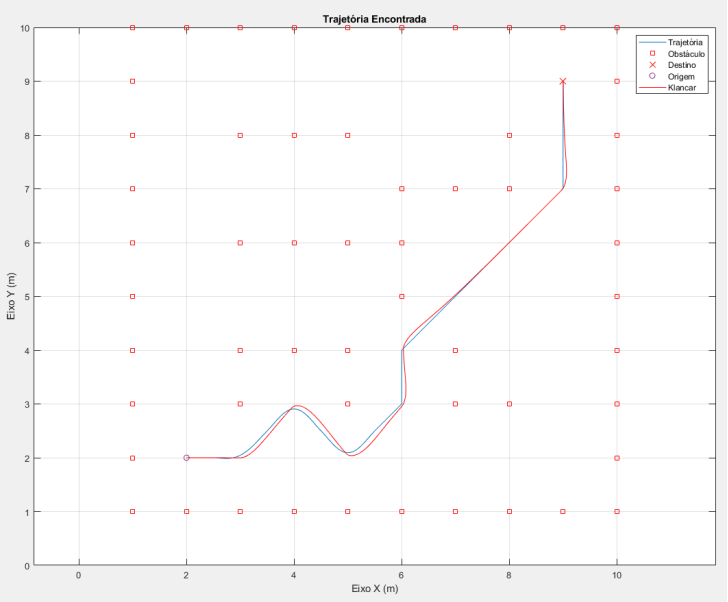


Fonte: O próprio autor

A terceira trajetória foi planejada com um posição inicial $(2,2)$ e um ângulo inicial de $\theta = 0$. O objetivo era chegar ao ponto de destino $(9,9)$. Assim como nas trajetórias anteriores, foram adicionados obstáculos ao longo do mapa. Ao analisar a Figura (27), podemos observar que o controlador *Klancar* foi capaz de seguir a trajetória planejada, contornando os obstáculos de forma eficiente. Diferente das outras trajetórias, foi adotado uma velocidade angular máxima, ω_{max}, de $4 rad/s$. Nos outros experimentos, a velocidade angular chegava no ponto de saturação, dessa forma, mudou-se seu valor máximo, para verificar o efeito de sua resposta.

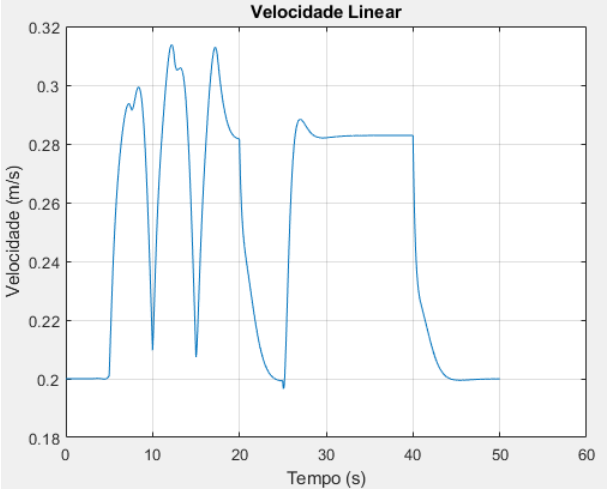
Dessa forma, a terceira trajetória também demonstra a eficiência do controlador *Klancar* em seguir trajetórias complexas, mesmo em situações em que é necessário realizar ajustes iniciais de alinhamento do robô.

Figura 27 – Seguimento de Trajetória - Trajetória 03



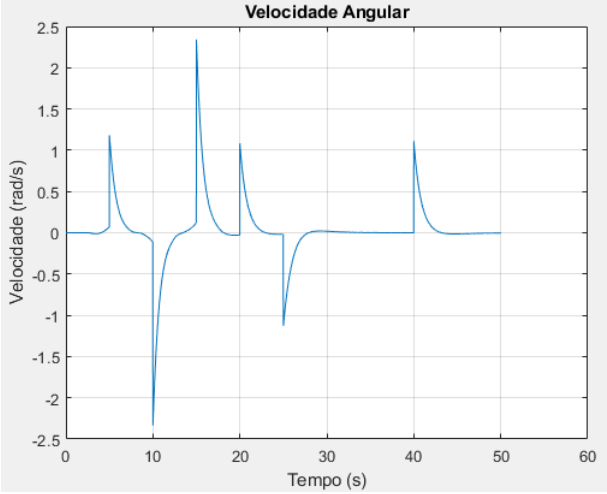
Fonte: O próprio autor

Figura 28 – Velocidade Linear - Trajetória 03



Fonte: O próprio autor

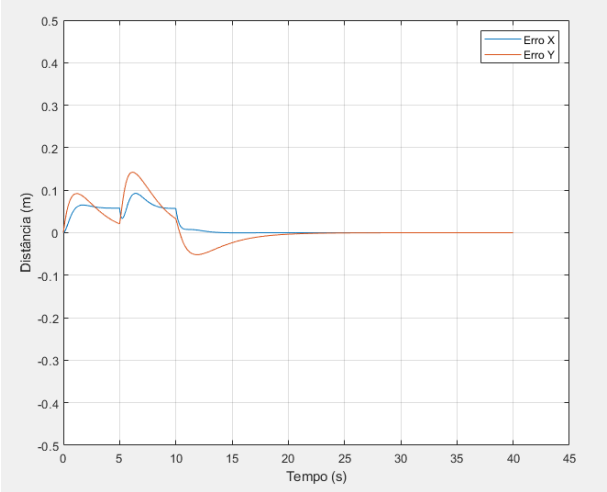
Figura 29 – Velocidade Angular - Trajetória 03



Fonte: O próprio autor

Na Figura 30 é possível verificar a questão do erro da posição do robô e da trajetória, como indica a equação matricial 3.4. Observa-se que esses erros tendem a aumentar no início da trajetória devido à orientação inicial do robô ser diferente daquela exigida pela trajetória. Como resultado, o robô precisa ajustar sua orientação para se alinhar à trajetória, o que leva à diminuição progressiva dos erros ao longo do percurso.

Figura 30 – Erro X e Y da Trajetória 01



Fonte: O próprio autor

3.5 Conclusão do Capítulo

O capítulo apresentou a aplicação do algoritmo A* em conjunto com o controlador Klancar para o seguimento de trajetórias em um robô móvel. Foram realizados três experimentos, nos quais trajetórias complexas foram geradas considerando evitar obstáculos no ambiente. Os

resultados obtidos demonstraram que o algoritmo A* foi capaz de criar trajetórias adequadas, desviando dos obstáculos e levando o robô ao ponto de destino. Essas trajetórias foram utilizadas como referência para o controlador Klancar.

O controlador Klancar mostrou-se eficiente no seguimento das trajetórias geradas pelo algoritmo A*. O robô conseguiu seguir as trajetórias com precisão, mantendo-se próximos à referência e evitando desvios significativos. Além disso, os valores de velocidade linear e angular foram adequados, não apresentando saturação constante e permitindo o seguimento suave das trajetórias.

Os resultados obtidos reforçam a eficácia da abordagem combinada do algoritmo A* com o controlador Klancar para o seguimento de trajetórias em robôs móveis. Essa combinação permite que o robô navegue de forma autônoma em ambientes complexos, evitando obstáculos e seguindo trajetórias desejadas.

No próximo capítulo, serão apresentadas as considerações finais do trabalho, destacando os principais resultados alcançados.

4 CONCLUSÃO

Neste trabalho, foi abordada a cinemática do robô móvel, analisando as equações de posição que descrevem seu movimento no espaço. Em seguida, foi desenvolvido um algoritmo de planejamento de trajetória com base em um mapa de ocupação (*Occupancy Grid Map*), o qual se mostrou efetivo ao alcançar o destino desejado e evitar colisões com obstáculos ao longo do caminho. Além disso, foi realizada uma suavização da trajetória para garantir um movimento mais suave e natural do robô.

O controlador *Klancar* foi escolhido e seus parâmetros foram ajustados para o sistema em estudo. Verificou-se que os valores dos parâmetros ζ e g desempenham um papel significativo no seguimento da trajetória pelo robô. Após testes e análises, os valores $\zeta = 0,5$ e $g = 50$ foram selecionados, pois proporcionaram resultados na qual o erro de posição ficava em valores aceitáveis, como indica a Figura 30.

O seguimento da trajetória planejada pelo algoritmo A* foi realizado em conjunto com o controlador *Klancar* em diferentes cenários com obstáculos. Os resultados obtidos demonstraram um bom desempenho do controlador, pois o robô foi capaz de evitar colisões com os obstáculos ao longo do percurso, e as velocidades do robô se mantiveram em níveis razoáveis, sem atingir constantemente sua saturação, mantendo-se próximas dos valores de referência estabelecidos, além dos erros se manterem próximos.

É importante ressaltar que a escolha adequada das dimensões das células do mapa de ocupação e das dimensões do robô são fatores cruciais para permitir a correta navegação do robô ao longo da trajetória planejada. Além disso, é possível adicionar uma margem de segurança para o robô, levando em consideração curvas e perturbações, a fim de evitar possíveis colisões causadas por um possível *overshoot* do controle.

REFERÊNCIAS

ELFES, A. Using occupancy grids for mobile robot perception and navigation. **Computer**, p. 46–57, 1989.

GALLI R. BARBER, S. G. M.; MORENO, L. Path planning using matlab-ros integration applied to mobile robots. **IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)**, p. 98–103, 2017.

GASPARETTO, A.; BOSCARIOL, P.; LANZUTTI, A.; VIDONI, R. Trajectory planning in robotics. **Mathematics in Computer Science**, v. 6, 09 2012.

GONZALEZ, C. M. R.; KLOETZER, M. A matlab-based interactive simulator for teaching mobile robotics. **IEEE International Conference on Automation Science and Engineering (CASE)**, p. 310–315, 2015.

JIANGDAGGER, Z.-P.; NIJMEIJER, H. Tracking control of mobile robots: A case study in backstepping. **Automatica**, Elsevier, v. 33, n. 7, p. 1393–1399, 1997.

JR, U. d. M. P.; CARVALHO, M. P.; CONCEIÇÃO, A. G. S. Campos potenciais artificiais aplicado ao planejamento de trajetórias do braço robótico jaco. In: **Congresso Brasileiro de Automática-CBA**. [S.l.: s.n.], 2019. v. 1, n. 1.

KARUR, K.; SHARMA, N.; DHARMATTI, C.; SIEGEL, J. E. A survey of path planning algorithms for mobile robots. **Vehicles**, MDPI, v. 3, n. 3, p. 448–468, 2021.

KLANCAR, G. **Wheeled Mobile Robotics**. [S.l.]: Butterworth-Heinemann, 2017. ISBN 9780128042045.

KLANCAR, G.; MATKO, D.; BLAZIC, S. Mobile robot control on a reference path. In: IEEE. **Proceedings of the 2005 IEEE International Symposium on, Mediterrean Conference on Control and Automation Intelligent Control, 2005.** [S.l.], 2005. p. 1343–1348.

LUCA, A. D.; ORIOLO, G.; VENDITTELLI, M. Control of wheeled mobile robots: An experimental overview. **RAMSETE: articulated and mobile robotics for services and technologies**, Springer, p. 181–226, 2002.

PENG, J.; HUANG, Y.; LUO, G. Robot path planning based on improved a* algorithm. **Cybernetics and Information Technologies**, Walter de Gruyter GmbH, v. 15, n. 2, p. 171–180, 2015.

SALEM, F. A. Dynamic and kinematic models and control for differential drive mobile robots. **International Journal of Current Engineering and Technology**, v. 3, n. 2, p. 253–263, 2013.

SVENSTRUP, M.; BAK, T.; ANDERSEN, H. J. Trajectory planning for robots in dynamic human environments. In: IEEE. **2010 IEEE/RSJ International Conference on Intelligent Robots and Systems**. [S.l.], 2010. p. 4293–4298.