

CENTRO UNIVERSITÁRIO FEI

Bruno Henrique de Almeida Borges

Lucas Agostinho Silva

Lucas Miguel Marciano

Matheus Ribeiro Castilho e Silva

Yasmin Marine Almeida

ROBÔ DE LIMPEZA INDUSTRIAL: Desenvolvimento e aplicação de um robô aspirador
autônomo em linhas de produção de ensaque

São Bernardo do Campo

2025

Bruno Henrique de Almeida Borges

Lucas Agostinho Silva

Lucas Miguel Marciano

Matheus Ribeiro Castilho e Silva

Yasmin Marine Almeida

ROBÔ DE LIMPEZA INDUSTRIAL: Desenvolvimento e aplicação de um robô aspirador
autônomo em linhas de produção de ensaque

Trabalho de Conclusão de Curso apresentado
ao Centro Universitário FEI, como parte dos
requisitos necessários para obtenção do título de
Bacharel em Engenharia de Robôs e Engenharia
Elétrica. Orientado pelo Prof. Dr. Fagner de
Assis Moura Pimentel.

São Bernardo do Campo

2025

Robô de limpeza industrial : Desenvolvimento e aplicação de um robô aspirador autônomo em linhas de produção de ensaque / Bruno Henrique de Almeida Borges...[et al.]. São Bernardo do Campo, 2025.

90 f. : il.

Trabalho de Conclusão de Curso - Centro Universitário FEI.
Orientador: Prof. Dr. Fagner de Assis Moura Pimentel.

1. Robô aspirador. 2. Automação. 3. Controle Remoto. 4. Linha de produção de ensaque. I. Almeida Borges, Bruno Henrique de. II. Agostinho Silva, Lucas. III. Marciano, Lucas Miguel. IV. Ribeiro Castilho e Silva, Matheus. V. Marine Almeida, Yasmin. VI. Assis Moura Pimentel, Fagner de, orient. VII. Título.

Elaborada pelo sistema de geração automática de ficha catalográfica da FEI com os dados fornecidos pelo(a) autor(a).

BRUNO HENRIQUE DE A BORGES
LUCAS MIGUEL MARCIANO
LUCAS AGOSTINHO SILVA
MATHEUS RIBEIRO C E SILVA
YASMIN MARINE ALMEIDA

**ROBÔ DE LIMPEZA INDUSTRIAL Desenvolvimento e aplicação
de um robô aspirador autônomo em linhas de produção de ensaque**

Trabalho de Conclusão de Curso apresentado ao Centro Universitário FEI, como parte dos requisitos necessários para obtenção do título de Bacharel em Ciência da Computação. Orientado pelo Prof. Dr. Fagner de Assis Moura Pimentel.

Comissão julgadora

Prof. Dr. Fagner de Assis Moura Pimentel
(Orientador)

Prof. Dr. Danilo Hernani Perico

Prof. Dr. Isaac Jesus da Silva

São Bernardo do Campo

2025

Dedicamos este Trabalho de Conclusão de Curso aos nossos familiares e professores, cuja dedicação, esforço e apoio incondicional nos acompanharam desde o início de nossa trajetória acadêmica. Esta conquista é o fruto, não apenas do incentivo e dos valores transmitidos por eles, mas também do reconhecimento de sua confiança em nosso potencial.

AGRADECIMENTOS

Com a finalização de Trabalho de Conclusão de Curso, não podemos finalizá-lo sem expressar nossa gratidão a todos que nos auxiliaram neste percurso tão significativo em nossas trajetórias acadêmicas.

Queremos expressar nossa gratidão ao nosso Professor Orientador, Dr. Fagner de Assis Moura Pimentel por toda assistência prestada, desde questões simples com o relatório até questões mais complexas que impactariam diretamente no nosso projeto.

Agradecemos ao professor Isaac Jesus da Silva pelo apoio prestado na parte de Impressão 3D do nosso robô, nos ajudando nos momentos mais críticos, além de ter aceitado integrar a banca de avaliação.

Agradecemos ao coordenador Danilo Hernani Perico por aceitar participar de nossa banca de avaliação, compartilhando conosco momentos e experiências que nos impactaram durante toda a nossa trajetória acadêmica e que continuarão a impactar durante nossas carreiras profissionais futuras. Atuando como um dos primeiros docentes que tivemos do curso de Engenharia de Robôs, e atualmente atuando como nosso coordenador de curso.

Agradecemos aos professores Fabio Delatore e Ricardo Germano Stolf, que nos ajudaram ativamente na parte elétrica, contribuindo para o desenvolvimento dos circuitos e o dimensionamento e produção da bateria.

Agradecemos também ao encarregado da manutenção técnica do ensaque da Braskem na região de São Paulo, Edivaldo Francisco Gará, que nos permitiu usar uma das linhas de ensaque da companhia como modelo para nosso projeto e autorizou a execução de medições para a elaboração de um mapa simulado com medidas autênticas.

LISTA DE ILUSTRAÇÕES

Figura 1 – Modelo da linha de ensaque encontrado no manual do fabricante (PALLE-TIZER SERIES 1500)	12
Figura 2 – Exemplos de robôs domésticos	13
Figura 3 – Exemplos de modelos de PPCR	16
Figura 4 – Exemplo de aplicação de Monte Carlo	20
Figura 5 – Possíveis formatos de um robô aspirador	24
Figura 6 – Mapa simulado	33
Figura 7 – Urdf do robô	33
Figura 8 – Navegação em Gazebo	34
Figura 9 – Robô dentro do mundo simulado	35
Figura 10 – Base com Espaçadores	38
Figura 11 – Entrada para os <i>Pellets</i>	38
Figura 12 – Rasgos para passagem das Rodas	39
Figura 13 – Andar inferior e intermediário com Espaçadores	39
Figura 14 – Carcaça completa	40
Figura 15 – Carcaça Completa + Cúpula	40
Figura 16 – Impressora 3D, Sermoon D1, imagem retirada do site oficial do fabricante.	41
Figura 17 – Simulação para impressão da Cúpula	42
Figura 18 – Base Inferior	42
Figura 19 – Base Inferior com espessura 3mm	43
Figura 20 – Base Intermediaria	43
Figura 21 – Base Intermediaria com espessura 5mm	44
Figura 22 – Espaçadores	44
Figura 23 – Base inferior em ABS	45
Figura 24 – Base superior em ABS	45
Figura 25 – Estrutura Completa	46
Figura 26 – Circuito da bateria	52
Figura 27 – Simulação do circuito da bateria	53

Figura 28 – Mundo simulado ao lado do mapa gerado	60
Figura 29 – Exemplo 1 de funcionamento do método de localização	61
Figura 30 – Exemplo 2 de funcionamento do método de localização	61
Figura 31 – Exemplo 3 de funcionamento do método de localização	62
Figura 32 – Exemplo de trajetória gerada por planejador em zig-zag horizontal	62
Figura 33 – Exemplo de trajetória gerada por planejador em zig-zag vertical	63
Figura 34 – Exemplo de trajetória gerada por planejador em espiral retangular	63
Figura 35 – Exemplo de área coberta durante o trajeto de zig-zag horizontal	66
Figura 36 – Exemplo de área coberta durante o trajeto de zig-zag vertical	67
Figura 37 – Exemplo de área coberta durante o trajeto de contorno	67
Figura 38 – Exemplo de área coberta durante o trajeto aleatório	69
Figura 39 – Exemplo de área coberta durante o trajeto aleatório	69
Figura 40 – Exemplo de área coberta durante o trajeto aleatório	69
Figura 41 – Circuito montado no Tinkercad	71
Figura 42 – Fluxograma do código usado nos testes com o sensor Ultrassom	72
Figura 43 – Circuito eletrico no Fritzing	72
Figura 44 – Robô identificando um obstaculo	73
Figura 45 – Sensor virando 25 graus a direita	73
Figura 46 – Sensor voltando ao seu eixo inicial	73
Figura 47 – Sensor virando 25 graus a esquerda	74
Figura 48 – Robô realizando o desvio para a direita	74
Figura 49 – Robô se movimentando para frente após o desvio	74
Figura 50 – Conversor AC-DC	75
Figura 51 – Limitador de corrente LM338	75
Figura 52 – Regulador de tensão	76
Figura 53 – Circuito de Carregamento de baterias de 12 volts	76
Figura 54 – Circuito de Carregamento de baterias de 12 volts, placa ilhada	77
Figura 55 – Simulação FEM, Andar Inferior com 1,565 Kg + 20%	79
Figura 56 – Simulação FEM, Andar Intermediário com 0,162Kg + 20%	80
Figura 57 – Simulação FEM, Andar Inferior com 3Kg + 20%	80
Figura 58 – Simulação FEM, Andar Superior com 110g + 20%	81
Figura 59 – Montagem Inicial dos Componentes	81
Figura 60 – Exemplos de robôs industriais	82

SUMÁRIO

1	INTRODUÇÃO	11
1.1	OBJETIVO	13
1.2	ESTRUTURA DO TRABALHO	13
2	CONCEITOS FUNDAMENTAIS	15
2.1	ROBÔ ASPIRADOR	15
2.2	LINHA DE PRODUÇÃO DE ENSAQUE	15
2.3	PATH PLANNING	15
2.4	PPCR (PATH PLANNING OF COVERAGE REGION)	16
2.5	SLAM (SIMULTANEOUS LOCALIZATION AND MAPPING)	17
2.6	PLANEJADOR GLOBAL	17
2.7	A* (A STAR)	17
2.8	PLANEJADOR LOCAL	18
2.9	DWB (DYNAMIC WINDOW APPROACH)	18
2.10	RPP (REGULATED PURE PURSUIT)	18
2.11	VP (VECTOR PURSUIT)	19
2.12	MAPEAMENTO	19
2.13	MÉTODO DE MONTE CARLO	19
2.14	ROS (ROBOT OPERATING SYSTEM)	20
2.15	RVIZ	21
2.16	SISTEMA DE ASPIRAÇÃO	21
2.17	COMUNICAÇÃO REMOTA	21
3	TRABALHOS RELACIONADOS	22
3.1	PATH PLANNING (PPCR)	22
3.2	MAPEAMENTO COM MONTE CARLO	23
3.3	ANÁLISE DE ÁREA EFETIVA DE LIMPEZA	24
3.4	COMUNICAÇÃO REMOTA	24
3.5	SISTEMA DE VÁCUO	25
3.6	CIRCUITOS DE PROTEÇÃO	25
3.7	DIMENSIONAMENTO DE BATERIA	26
3.8	ELÉTRICA	26
3.9	ALIMENTAÇÃO ELÉTRICA	27

4	METODOLOGIA	28
4.1	MATERIAIS	28
4.1.1	Software	28
4.1.2	Placas de controle	29
4.1.3	Controle remoto	29
4.1.4	Sensores	29
4.1.5	Corpo do robô	29
4.1.6	Motores, bateria e botão de emergência	30
4.1.7	Placa principal	30
4.2	MÉTODOS	32
4.2.1	Medidas da área de aplicação	32
4.2.2	Preparação do ambiente de simulação virtual	32
4.2.3	Simulação no GAZEBO	32
4.2.4	Codificação do planejamento e localização do robô em ROS	33
4.2.5	Subsistemas do robô	36
4.2.6	Dimensionamento do robô	36
4.2.7	Impressão do corpo	37
4.2.8	Desenhos no NX	37
4.2.9	Conversão de arquivos <i>.obj</i> para <i>.gcode</i>	40
4.2.10	Impressão das peças	42
4.2.11	Montagem do circuito	46
4.2.12	Criação do sistema de proteção	47
4.2.13	Sistema de baterias	48
4.2.14	Configuração dos motores	49
4.3	MÉTRICAS	49
4.3.1	Simulação	49
4.3.2	Área efetiva de limpeza, tempo médio de limpeza	50
4.3.3	Precisão e Alcance do sensor ultrassom	50
4.3.4	Tempo Médio de Operação da bateria	51
4.3.5	Eficiência no Controle de Motores	51
4.3.6	Carregador da bateria	52
4.3.7	Testes de Métodos de Elementos Finitos (FEM)	58

5	RESULTADOS	60
5.1	SIMULAÇÃO	60
5.1.1	Mapeamento	60
5.1.2	Localização	61
5.1.3	Definição do planejador global e planejador local	62
5.1.3.1	<i>Definição do planejador global</i>	65
5.1.3.2	<i>Definição do planejador local</i>	66
5.1.4	Definição do melhor tipo de trajeto	66
5.1.5	Tempo médio de limpeza e taxa de efetividade da navegação	70
5.2	ELÉTRICA	70
5.2.1	Círcuito elétrico	70
5.2.2	Carregador da bateria	75
5.3	MECÂNICA	78
5.3.1	Testes de resistência e deformação	78
5.3.2	Montagem inicial dos componentes	81
5.4	EFETIVIDADE NO MERCADO E CUSTOS	82
6	CONCLUSÃO E TRABALHOS FUTUROS	84
6.1	DISCUSSÃO E CONCLUSÃO	84
6.2	TRABALHOS FUTUROS	85
	REFERÊNCIAS	87

RESUMO

A linha de produção é um método de produção em série aplicado a produtos de várias dimensões. No entanto, existe um desafio frequente na contenção de vazamentos de itens extremamente pequenos, com apenas alguns milímetros de dimensão. No modelo utilizado para nosso estudo de caso o uso constante de esteiras e equipamentos pesados torna o ambiente propício para tais acontecimentos. Isso é um grande problema, pois ao longo do tempo acumula-se um grande volume de material, exigindo uma limpeza. No entanto, os espaços abaixo das máquinas e esteiras de uma linha de produção são normalmente fechados, visando a segurança dos trabalhadores. Assim, uma interrupção na produção deve acontecer, resultando em um prejuízo para a empresa. Este projeto tem como objetivo a criação e implementação de um robô aspirador totalmente autônomo, capaz de superar obstáculos tanto móveis quanto fixos, cumprir as normas de segurança obrigatórias e ter um tamanho bastante reduzido para facilitar a sua movimentação em espaços estreitos em áreas confinadas, removendo o produto acumulado no chão sem a necessidade de interromper a produção.

Para o desenvolvimento deste projeto foi feita a divisão das tarefas em três frentes, sendo elas programação, elétrica e mecânica. Todas as metodologias utilizadas para a confecção do projeto, testes realizados e seus resultados, conclusões e trabalhos futuros, estão detalhados ao longo deste documento.

Nas seções de programação, é realizada a criação de um mundo simulado baseado no local do estudo de caso, o método de mapeamento e os algoritmos de planejamento empregados, bem como os testes de eficácia do robô.

Nas seções de elétrica, é apresentado o circuito de funcionamento do robô, iniciado pelo dimensionamento da bateria, a escolha dos componentes e por fim, o sistema de proteção.

Nas seções de mecânica, é mostrado o desenvolvimento do corpo do robô, começando com a definição das limitações físicas e o layout do robô. Depois, a criação das peças usando o software NX, testes de resistência, seguidas pela impressão 3D de todas as partes e montagem do corpo do robô.

Palavras-chave: Robô aspirador; Controle Remoto; Linha de produção de ensaque; Automação.

1 INTRODUÇÃO

Embora a linha de produção seja um método frequentemente empregado por indústrias, cada uma tem uma configuração distinta de acordo com suas demandas. Para simplificar a explicação do projeto, a Braskem será usada como exemplo, mas o robô pode ser aplicado a qualquer outra que enfrente o mesmo desafio. A Braskem é uma companhia do setor petroquímico, cujo principal produto são os *pellets*¹.

Da mesma forma que outras empresas, ela enfrenta a perda de produto ao longo do processo, contudo, existe um setor onde essa questão se intensifica. Este local é conhecido como *linha de ensaque*² e é composto por vários equipamentos, incluindo paletizadoras automáticas, ensacadeiras e envolvedoras, interligados por esteiras. Portanto, torna-se imprescindível o isolamento de algumas áreas de maior risco para a proteção de seus trabalhadores.

No processo de ensaque, podem ocorrer erros como rasgos nos sacos ou excesso de velocidade em seu preenchimento, resultando na dispersão do produto pelo chão e em certas partes dos equipamentos mencionados. Isso torna necessária a limpeza regular da linha para remover esse produto disperso, com o objetivo de prevenir não apenas problemas internos, como falhas nos equipamentos, mas também danos ao meio ambiente.

Atualmente, o processo de limpeza da linha possui diversos agravantes. Os operadores de máquinas precisam acessar as partes enclausuradas para realizar a limpeza, exigindo a interrupção total da produção para prevenir riscos à saúde dos trabalhadores. Apesar da parada, ainda há locais que oferecem riscos aos funcionários.

Com o avanço da robótica e automação industrial, surge a oportunidade de automatizar esse procedimento através de um robô aspirador, reduzindo assim o perigo para as pessoas e incrementando a eficiência de produção.

A Figura 1 ilustra o modelo da linha de ensaque encontrado na Braskem. Como se pode observar, na parte central do equipamento há um elevador de carga destinado à elevação dos pellets. Este elevador é totalmente dependente de um sistema de contrapeso para sua movimentação, o que torna qualquer acesso à parte inferior do mesmo arriscado, pois uma falha grave no sistema de contrapeso pode resultar na queda do elevador.

Isso nos motivou a desenvolver um robô de limpeza industrial para atuar justamente na parte inferior dos equipamentos. Com sua utilização, eliminariámos a necessidade de interromper

¹*Pellet*: Nome dado ao plástico em sua forma granulada, normalmente encontrado no formato de esferas ou semiesféricas.

²*Linha de ensaque*: É a etapa final na produção dos pellets onde ele é ensacado, e tem sua embalagem lacrada.

a linha para limpeza, além de eliminar o perigo de acidentes e a necessidade de entrada de pessoas não apenas neste local, mas também em outras áreas isoladas da linha.

Figura 1 – Modelo da linha de ensaque encontrado no manual do fabricante (PALLETIZER SERIES 1500)



Fonte: NEWTEC BAG PALLETIZING, [s.d].

Tendo em vista a grande gama de robôs de limpeza existentes no mercado, o que torna inviável as empresas utilizarem um dos diversos modelos disponíveis no mercado?

Existe um sério obstáculo, esses robôs possuem diversas características que dificultam sua utilização em indústrias. Na Figura 2, notamos dois tipos de robôs domésticos: o Roomba 614, fabricado pela empresa **iRobot**, que representa os modelos mais populares e econômicos, e o ConnectLaser M7, fabricado pela empresa **Midea**, que se insere no grupo dos robôs aspiradores mais sofisticados atualmente disponíveis. Os principais atributos desses modelos são:

- **Populares e econômicos:** Programação de navegação básica que utiliza movimentos pré-estabelecidos, executados de forma aleatória e repetitiva, utilizando um único sensor bumper para identificar obstáculos. Valor comercial variando de R\$250,00 a R\$1.000,00.
- **Avançados e caros:** Realização de um mapeamento do local e planejamento da rota, utilizando um sensor a laser parecido com o LiDAR e controle através de um aplicativo. Preço de mercado variando entre R\$3.000,00 e R\$5.000,00.

Os principais impeditivos para os modelos mais econômicos são a sua movimentação programada para realizar um trajeto aleatório e o seu método de detecção de obstáculos, que é um simples sensor de contato (bumper). Isso eleva consideravelmente o tempo requerido para a limpeza e apresenta perigos, como a chance de o robô escapar da área de limpeza, podendo provocar incidentes em outros setores da empresa ou trombar com equipamentos móveis, gerando falhas.

Figura 2 – Exemplos de robôs domésticos



Fonte: Roomba 614 (IROBOT, s.d.) (esquerdo); *ConnectLaser M7* (MIDEA, s.d.) (direito), 2024

Embora os modelos mais sofisticados possuam a capacidade de mapeamento e navegação autônoma requeridas, é crucial que os robôs móveis em indústrias obedeçam a certas normas de segurança, como possuir um botão de emergência e, neste caso específico, devido à sua localização em um espaço confinado, um método de recuperação à distância do robô.

1.1 OBJETIVO

Desenvolver e implementar um robô aspirador autônomo industrial para uso em linhas de produção de ensaque e outras áreas com desafios semelhantes, com o objetivo de aumentar a produtividade e diminuir os riscos de acidentes de trabalho. Analisar mecanismos de segurança, incluindo botões de emergência, redundância de sensores, sistemas de limitação de sobrecarga, códigos de multiprocessamento, limitação da área de operação do robô e controle manual remoto.

1.2 ESTRUTURA DO TRABALHO

O restante deste trabalho é dividido da seguinte maneira:

No Capítulo 2, são apresentados todos os conceitos utilizados e relacionados ao tema abordado, para que o leitor possa entender com clareza as técnicas que estão sendo tratadas no trabalho e compreender os termos que são descritos posteriormente.

No Capítulo 3, são descritos trabalhos relacionados disponíveis na literatura, com o objetivo de apresentar o cenário atual de pesquisa da área.

O Capítulo 4 detalha a metodologia que foi utilizada para o desenvolvimento deste trabalho, demonstrando as técnicas utilizadas e os passos seguidos para atingir o objetivo final.

O Capítulo 5 contém os resultados obtidos após a execução das etapas contidas no Capítulo 4.

O Capítulo 6 apresenta as discussões finais e a conclusão, além dos trabalhos futuros.

2 CONCEITOS FUNDAMENTAIS

O desenvolvimento e a implementação de várias ferramentas foram essenciais para a construção de um robô autônomo destinado à limpeza de ambientes industriais. Foi necessário um estudo prévio de várias opções e métodos para a implementação dessas ferramentas, dentre os quais se destacam:

2.1 ROBÔ ASPIRADOR

O robô aspirador é um aspirador de pó independente, equipado com um sistema de sucção para a limpeza do chão. Juntamente com drivers, controladores e sensores, permite que o robô efetue a limpeza de maneira autônoma, conforme mencionado no artigo de (ECOVACS, s.d.) nomeado "*What are robot vacuum cleaners*". Normalmente, sua utilização é residencial, mas tivemos a intenção de desenvolver um modelo similar para uso industrial.

2.2 LINHA DE PRODUÇÃO DE ENSAQUE

Uma linha de produção é um método de produção em série onde os trabalhadores, auxiliados por máquinas, executam várias tarefas de maneira sequencial. Segundo o autor que atua na área de manutenção dedicada a essa linha, a área de ensaque é uma das fases finais de uma linha de produção, destinada a embalar um produto específico que foi produzido nas fases anteriores da linha.

2.3 PATH PLANNING

O *path planning* é uma técnica de planejamento de rota que permite a um robô ou veículo autônomo realizar um percurso da forma mais curta e livre de barreiras. Para tal, é usado um mapa do ambiente, a pose do robô ou veículo e o destino pretendido. Conforme estudado no artigo de (YANG et al., 2023), intitulado "*Path Planning Technique for Mobile Robots: A Review*". Ele apareceu pela primeira vez em 1956 através de E.W. Dijkstra, que criou o "Algoritmo de Dijkstra".

Trata-se de um método bastante flexível, com vários tipos de algoritmos, cada um com suas próprias vantagens e desvantagens, sendo os mais conhecidos:

- Busca baseada em campos potenciais;
- Busca baseada em grade;

- Busca baseada em amostragem;
- Busca baseada em otimização de trajetória.

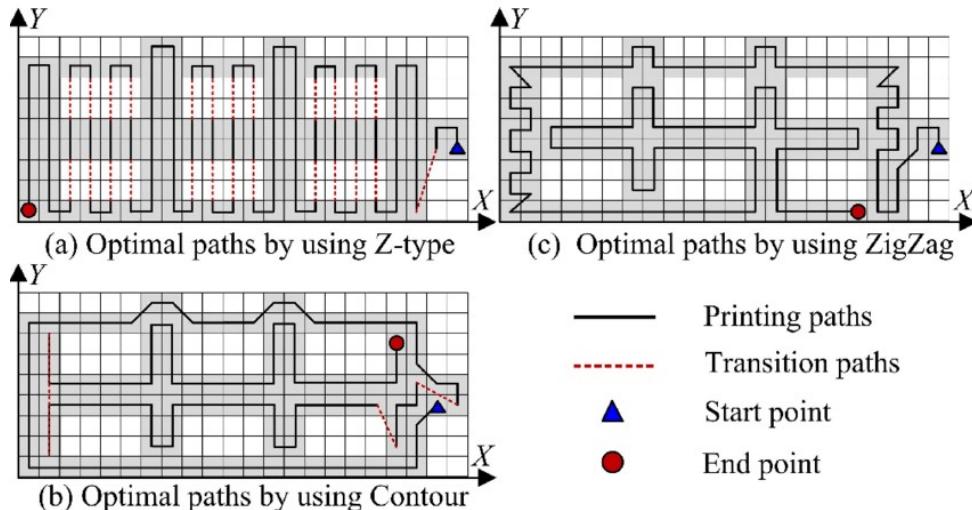
Há algoritmos para todos os tipos de aplicações e é crucial saber qual escolher, já que não existe um método perfeito. Portanto, é essencial entender as prioridades do seu projeto.

O uso mais frequente de algoritmos de *path planning* envolve um algoritmo global e outro local. O primeiro recebe o mapa geral e os locais pelos quais o robô deve passar, traçando uma rota entre eles, enquanto o segundo estabelece o caminho do ponto atual até o próximo objetivo, prevenindo obstáculos.

2.4 PPCR (PATH PLANNING OF COVERAGE REGION)

PPCR é um algoritmo de planejamento de rota que leva em conta a área já percorrida pelo robô em um mapa previamente conhecido para otimizar sua eficácia. Este algoritmo possibilita quatro modelos de deslocamento, alguns exemplos podem ser vistos na Figura 3 abaixo:

Figura 3 – Exemplos de modelos de PPCR



Fonte: Zibo Zuo, 2017.

- **Movimento aleatório:** O robô se desloca em linha reta até encontrar um obstáculo, gira em um ângulo aleatório e retorna à posição inicial, repetindo esse ciclo até cobrir completamente a área;

- **Entre paredes:** A locomoção ocorre seguindo as paredes do ambiente;
- **Zig-Zag:** Movimentação em linha reta, seguida de deslocamento de uma célula para o lado e, finalmente, retorno em paralelo à linha reta anterior;
- **Espiral:** Movimento para fora e para dentro, representado por uma espiral, que pode ter formatos circulares ou retangulares.

Vale ressaltar que esse método é apenas uma forma de realizar o *path planning* mas não representa um algoritmo pronto como "Algoritmo de Dijkstra".

2.5 SLAM (SIMULTANEOUS LOCALIZATION AND MAPPING)

O SLAM é um algoritmo de mapeamento e localização simultânea amplamente aplicado na robótica. Ele opera da seguinte forma: a partir dos dados brutos dos sensores, ele identifica obstáculos e pontos de interesse, que são pontos de referência que o robô usará para se orientar no mapa que está elaborando. Simultaneamente, esses pontos de interesse são os próprios obstáculos que compõem o mapa, permitindo assim a realização simultânea de localização e mapeamento. Isso pode ser observado no artigo *The Simultaneous Localization and Mapping (SLAM)-An Overview* redigido por (ALSADIK; KARAM, 2021).

2.6 PLANEJADOR GLOBAL

O planejador global utiliza o mapa de um ambiente para calcular a melhor rota. Isso é realizado ao dividir o mapa em várias regiões menores e atribuir um valor de custo para cada região, em geral cada planejador global define o custo de uma maneira própria, entretanto em geral, regiões que apresentam maior proximidade do objetivo e que necessitam de menor movimentação para alcançá-las tendem a ter um custo menor, após a definição dos custos individuais de cada região o planejador determina qual o caminho com o menor custo possível a ser seguido, no caso do estudo realizado nessa monografia, foram utilizados o A estrela e o algoritmo de djikstra que possuem uma alta semelhança na forma de cálculo dos custos.

2.7 A* (A STAR)

O algoritmo global de *path planning* A*, como explicado no artigo (LESTER, 2005) denominado "*A* Pathfinding for Beginners*", envolve a segmentação do mapa em grades e a definição de áreas que podem ser percorridas e áreas que não podem. Então, por meio de um cálculo heurístico entre a distância do robô até o nó inicial, distância do objetivo até o robô após a movimentação, juntamente com o custo do deslocamento realizado para alcançar o próximo ponto do percurso, determina-se o melhor trajeto a seguir para alcançar o objetivo.

Este algoritmo é conhecido por sempre selecionar a rota mais curta entre as metas estabelecidas. Uma dificuldade conhecida deste algoritmo é a distância que o robô tende a

percorrer para ultrapassar os obstáculos estabelecidos na grade. Isso pode ser resolvido ampliando o tamanho das grades ou estabelecendo uma distância de segurança através do código.

2.8 PLANEJADOR LOCAL

O planejador local é quem realiza a movimentação do robô, ele utiliza a rota gerada pelo planejador global e tenta segui-la da forma mais precisa possível. Ele também é responsável pelo desvio de obstáculos principalmente se eles forem móveis, uma vez que o mapa utilizado pelo planejador global considera todos os obstáculos como sendo fixos, os planejadores locais utilizados nesse estudo estão listados abaixo, DWB, RPP e VP respectivamente.

2.9 DWB (DYNAMIC WINDOW APPROACH)

Dynamic window approach diferente de A*, é um algoritmo de navegação local. Ele se fundamenta nas restrições de espaço físico, velocidade e aceleração do robô, formando uma zona de busca ao seu redor e determinando o melhor caminho dentro dessa zona de busca até o próximo objetivo estabelecido pelo algoritmo de planejamento global. A sua ampla personalização permite que seja aplicado em praticamente qualquer situação, conforme demonstrado no artigo de (FOX; BURGARD; THRUN, 1997) intitulado "*The Dynamic Window Approach to Collision Avoidance*".

2.10 RPP (REGULATED PURE PURSUIT)

Pure Pursuit assim como o *DWB*, trata-se de um algoritmo de planejamento local, porém com outra premissa: seguir da maneira mais próxima possível o caminho estipulado pelo algoritmo global, dando a ele uma maior precisão ao seguir o trajeto, mas gerando também uma maior dependência do algoritmo global. Essa é a ideia por trás de todo algoritmo de *Pursuit*. Conforme dito por Steve Macenski, o algoritmo de *Regulated Pure Pursuit* utiliza a ideia comentada anteriormente juntamente com parâmetros de regulagem da velocidade e aceleração próximos de obstáculos ou de objetivos, com a ideia para aplicação justamente em robôs em áreas industriais ou de trabalho, limitando assim a possibilidade de acidentes e minimizando os riscos atrelados a um robô em alta velocidade (MACENSKI et al., 2023).

2.11 VP (VECTOR PURSUIT)

Vector Pursuit conforme citado pelo autor Jeffrey Wit, assim como *Regulate Pure Pursuit*, o objetivo é seguir da melhor forma possível o caminho estipulado pelo algoritmo global. A aplicação deles difere no seguinte aspecto: o *Vector Pursuit* trata-se de um algoritmo com base geométrica, visando maior velocidade e melhor detecção de obstáculos ao longo do percurso do robô, conseguindo apresentar em muitos casos um acompanhamento mais fiel do trajeto do que o próprio *RPP* devido a maiores velocidades de giro e translação.

2.12 MAPEAMENTO

O mapeamento de robôs móveis, também conhecido como mapeamento robótico, foi desenvolvido na década de 80, fortemente influenciado pela observação da natureza. Esse método permitiu a criação de um mapa do ambiente e também permitiu que o robô se localizasse dentro dele, um procedimento muito parecido com o empregado no mundo animal. Isso resulta em duas fontes de dados, que serão utilizadas no mapeamento do robô:

- Fonte gerada pelo robô, como número de rotações em cada roda;
- Fonte produzida por estímulos externos identificados por sensores, tais como câmera, lidar, sonar, laser, entre outros.

Este método é minuciosamente explicado e detalhado no artigo titulado "*Robotic Mapping: A Survey*" escrito por (THRUN, 2003), contudo, apresenta algumas restrições. Por exemplo, a aprendizagem do mapa e o processo de localização não podem ser separados, os erros inerentes a todos os sensores, além da complexidade em identificar se o robô está em um local desconhecido ou já previamente mapeado.

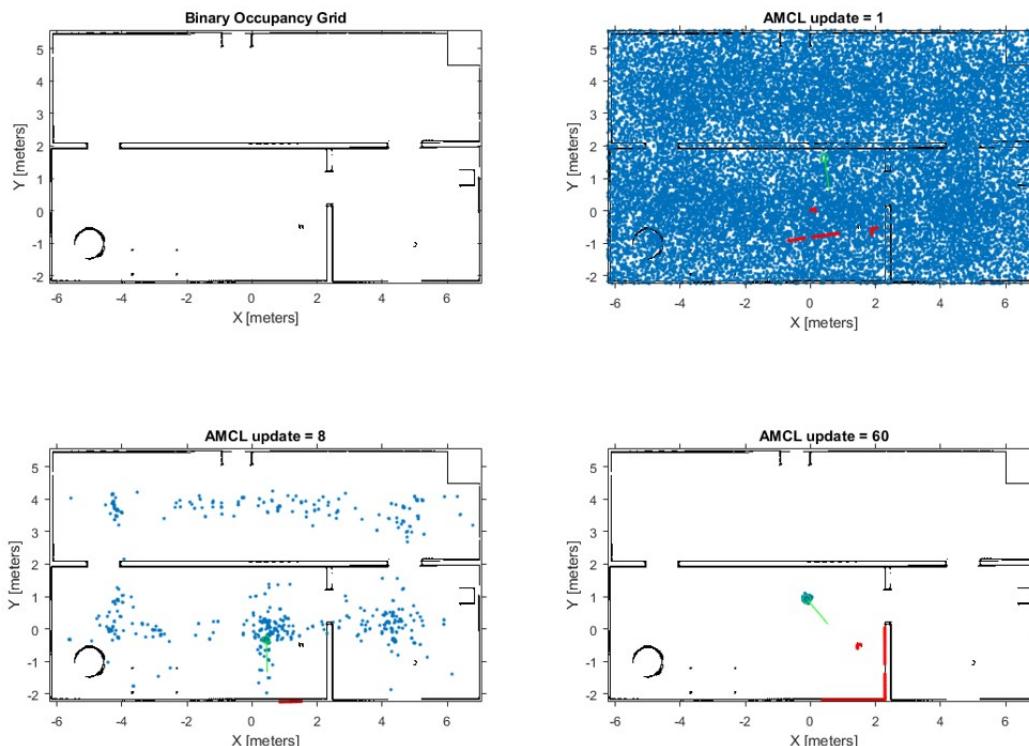
2.13 MÉTODO DE MONTE CARLO

Monte Carlo Localization (MCL) é um algoritmo de localização para robôs que emprega filtros de partículas. A partir de um mapa do local, o algoritmo é capaz de prever a posição e direção do robô, conforme ele se move e detecta obstáculos através de seus sensores. Conforme o artigo da empresa (AMAZON, s.d.) denominado "*What is The Monte Carlo Simulation?*", esse método foi desenvolvido na década de 40 por John von Neumann e Stanislaw Ulam.

Inicialmente, ele emprega um filtro de partículas para ilustrar as possíveis localizações do robô no mapa, começando com partículas distribuídas uniformemente e de forma aleatória por

todo o mapa, cada uma representando um possível local. Ao longo do tempo, a cada movimento do robô e análise de seus sensores, as partículas mais prováveis de estarem na posição do robô começam a se reunir em conjuntos nos pontos mais adequados para representar a verdadeira posição dele, até que haja informação suficiente para que todas as partículas se agrupem na real localização do robô, como na Figura 4.

Figura 4 – Exemplo de aplicação de Monte Carlo



Fonte:

Mihir Acharya, 2023.

2.14 ROS (ROBOT OPERATING SYSTEM)

Conforme explicado no artigo "*What Is a Robot Operating System (ROS)? Meaning, Working, Applications, and Benefits*" redigido por (KANADE, 2024). ROS é um sistema operacional de código aberto voltado para robôs, com um conjunto de bibliotecas e ferramentas que auxiliam na criação de aplicações robóticas.

Neste sistema, há pacotes que fornecem funcionalidades e aplicações, incluindo drivers de hardware, modelos de robôs, tipos de dados, planejamento, percepção, mapeamento e localização simultâneos, ferramentas de simulação, além de outros algoritmos comumente empregados.

As principais bibliotecas de ROS são nas linguagens de programação C++, Python e Lisp. No momento, apenas o Ubuntu Linux suporta o ROS, enquanto sistemas operacionais mais populares como Microsoft Windows e Mac OS são vistos como "experimentais".

2.15 RVIZ

Conforme descrito na documentação dele (**RVIZ**) o RVIZ trata-se de um software de vizualização 3D utilizado pelo ROS 2 para verificar tudo o que o robô está "vendo" e fazendo, ele será utilizado como um auxílio para aferir a qualidade do mapa feito via SLAM e para definição dos pontos de navegação do robô.

2.16 SISTEMA DE ASPIRAÇÃO

Inventado em 1901 pelo engenheiro britânico Hubert Cecil Booth, o sistema de vácuo teve sua primeira e mais famosa utilização no aspirador de pó. Inicialmente, ele operava com um motor a óleo, mas logo foi desenvolvida uma versão elétrica, conforme mencionado no artigo de (FERREIRA, 2021) denominado "*Curiosidades que, provavelmente, você não sabia sobre aspirador de pó*". Com o passar do tempo, este sistema passou por alterações e aprimoramentos, oferecendo várias formas, tamanhos e maneiras de ser utilizado.

2.17 COMUNICAÇÃO REMOTA

A comunicação remota é um processo que, por meio de conexões sem fio, possibilita o controle de um robô ou máquina por meio de um controle manual (físico ou virtual), realizado à distância. Nikola Tesla lançou em 1889 o primeiro dispositivo de comunicação à distância, um barco comandado por um controle remoto via rádio. Desde então, diversos sistemas e métodos de comunicação foram desenvolvidos, incluindo *Wi-Fi*, *Bluetooth*, infravermelho, entre outros. Cada tipo de conexão tem seus prós e contras, além de uma distância limite de alcance, ou seja, existe uma forma de conexão adequada para todas as demandas.

3 TRABALHOS RELACIONADOS

3.1 PATH PLANNING (PPCR)

O *path planning* foi utilizado como a base do nosso robô autônomo. Através deste tipo de algoritmo, o *software* é capaz de controlar os movimentos dos motores para que ele atinja várias áreas sem a necessidade de intervenção humana. Isso otimiza a sua varredura, permitindo que a mão de obra que seria deslocada para o seu controle possa ser empregada de forma mais eficaz em outros setores do ambiente industrial.

Para realizar esse trabalho, escolhemos utilizar o "*Path Planning Coverage Region*", que se adequa perfeitamente à nossa proposta. Com as informações obtidas, o robô pode se mover pelo ambiente sem encontrar barreiras e "varrer" toda a área que precisa ser aspirada.

Com o ambiente devidamente mapeado e segmentado por setores, o método a ser adotado deve ser selecionado, dentre as diversas alternativas mencionadas por (GYLLING; ELMARSSON, 2018) em seu trabalho "*Improving robotic vacuum cleaners*".

- a) ***The Pattern Method:*** Este é o método mais empregado por ser o menos complexo e estabelecer uma rota que não requer tanto do poder computacional do robô. Aqui, as áreas são vistas como células de uma matriz e, se os *scanners* do robô identificarem alguma mudança em relação ao mapa original, o próprio *software* se encarregará de criar uma nova célula. As áreas serão categorizadas em três tipos: abertas, fechadas e prioritárias. As áreas abertas correspondem a lugares onde o robô ainda não passou e limpou completamente. Prioritárias se assemelha às abertas, mas supera as abertas em termos de importância, ao considerar áreas isoladas, como uma célula que o robô não atravessou, mas que já foi percorrida por outras células. Em outras palavras, é como se ele precisasse limpar um local com resíduos de material. Finalmente, as fechadas, que se referem às células pelas quais o robô já passou.
- b) ***The Greedy Method:*** Nesta alternativa, devemos levar em conta 8 células ao redor do robô, onde cada célula inicialmente terá um valor inteiro, mas o valor que será apagado. Esses valores simbolizarão os obstáculos ao redor do robô (paredes são consideradas obstáculos). No próximo passo, o programa deve calcular o custo para executar o trajeto, sendo $CoverageCost = S - U$. S representa a quantidade de células ao seu redor, enquanto U representa a quantidade de células que não estão

cobertas. Ao realizar essa conta, o custo que for menor que 8 será selecionado, e as células adjacentes são aumentadas em 1. Esses passos são repetidos até que as células sejam limpas.

- c) ***The Genetic Method:*** Como o nome indica, o algoritmo trata as células como se fossem genes, formando cromossomos ao combinar um determinado número de genes. Inicia-se com uma das 8 posições disponíveis no robô, e diferentes cromossomos serão criados de forma aleatória, selecionando o próximo gene adjacente. O cálculo dessa maneira resultará em 8 possíveis rotas. e cada rota será avaliada com um valor, sendo selecionado o mais alto. Este procedimento pode levar à criação de novos e mais eficazes métodos para alcançar o objetivo. No entanto, este algoritmo pode levar muito tempo para limpar todo o ambiente, ou até mesmo não limpar determinados pontos.

Foi muito usado os estudos de (HOFNER; SCHMIDT, 1995) em sua pesquisa intitulada "*Path planning and guidance techniques for an autonomous mobile cleaning robot*".

Nesta pesquisa, conseguimos entender claramente as necessidades que o nosso *software* precisa satisfazer para funcionar adequadamente. Entre elas, estão as geometrias e dimensões do nosso robô, que são pontos iniciais para referência inicial. É importante lembrar que as alternativas mencionadas possuem desvantagens que precisam ser superadas, como a redundância de rotas.

Tudo isso é realizado através de cálculos e do uso de sensores, que nos ajudam a recolher dados físicos e transformá-los em unidades digitais que serão utilizadas no nosso projeto.

3.2 MAPEAMENTO COM MONTE CARLO

A estratégia selecionada para localizar o robô no ambiente foi a de Monte Carlo, baseada em pesquisas anteriores realizadas por (JANSON; SCHMERLING; PAVONE, s.d.) em seu trabalho "*Monte Carlo Motion Planning for Robot Trajectory Optimization Under Uncertainty*". O algoritmo de Monte Carlo é aplicado para aprimorar a varredura e análise do ambiente. Isso se deve ao fato de que, mesmo com um *path planning* bem elaborado, se o robô estiver em um ambiente real, ao invés de uma simulação onde todas as variáveis e eventos são adequadamente controlados, a ligação entre o robô e o ambiente precisa ser intensificada. O algoritmo de Monte Carlo usa os sensores do robô para tomar decisões em tempo real, incluindo estratégias para superar possíveis obstáculos em uma rota. Isso ocorre devido ao processo de decisão de

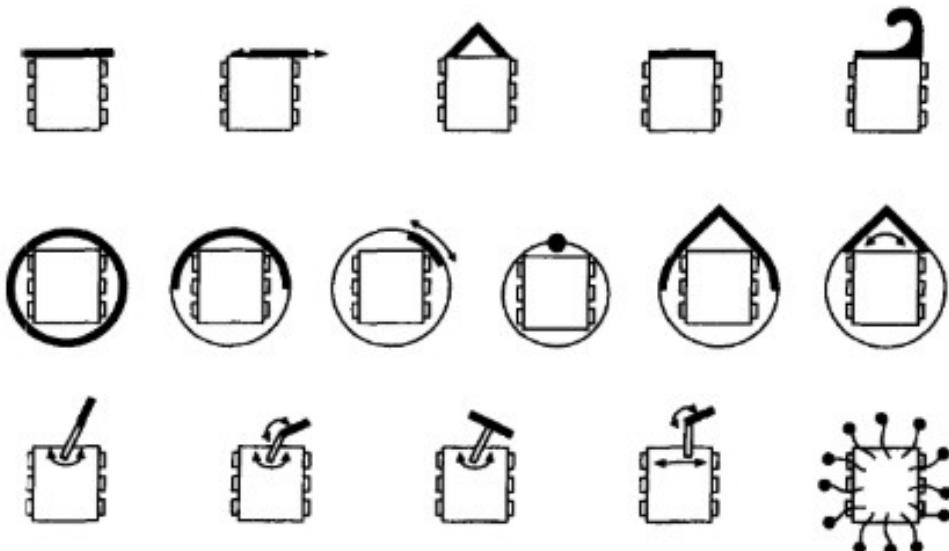
Markov, conforme ilustrado no trabalho de (DAM et al., 2022) chamado "*Monte-Carlo Robot Path Planning*", o *software* deve calcular o custo do caminho, através de uma recompensa.

3.3 ANÁLISE DE ÁREA EFETIVA DE LIMPEZA

Para o nosso projeto, era crucial que o robô fosse o mais compacto e eficaz possível. Uma das formas de alcançar isso era alterar a forma geométrica do robô. Isso pode envolver não só a alteração de sua geometria, mas também a inclusão de acessórios como um braço. Para uma limpeza mais eficaz e facilidade de deslocamento ao redor de obstáculos, o formato mais indicado era um corpo circular sem braço, conforme ilustrado brevemente no trabalho de (ULRICH; MONDADA; NICOUD, 1997) intitulado "*Autonomous vacuum cleaner*", alguns exemplos de formatos podem ser observados na Figura 5.

Não é por acaso que esse formato é empregado em robôs de limpeza doméstica de grandes companhias como iRobot e WAP. No entanto, além das vantagens, também surge um desafio: a limpeza de áreas onde o robô possui dificuldade para alcançar a sujeira, como cantos estreitos.

Figura 5 – Possíveis formatos de um robô aspirador



Fonte: Iwan Ulrich, 1997

3.4 COMUNICAÇÃO REMOTA

Pensamos em empregar um sistema de controle remoto que pudesse ser aplicado caso o mapeamento autônomo fosse interrompido por algum fator externo, como interferências, por exemplo.

Considerando isso, encontramos o trabalho "*Design and implementation of a cost effective vacuum cleaner robot*" escrito por (EREN; DOĞAN, 2022). Neste documento, identificamos uma lista básica de materiais empregados por eles, selecionados com base no custo-benefício, sem onerar o projeto final. Considerando que seu propósito era ser utilizado para alcançar áreas perigosas, sua perda ou quebra não deveria resultar em um prejuízo econômico considerável para a companhia. No entanto, a lista de materiais que utilizamos será discutida posteriormente na Metodologia.

E, finalmente, o mais importante para a realização da comunicação remota, o controle remoto via *Bluetooth*, escolhido devido à sua distância média de alcance, o suficiente para garantir a comunicação entre o supervisor responsável pela área e o robô durante seu funcionamento.

No artigo de (EREN; DOĞAN, 2022), é apresentado exemplos de como fabricar um controle remoto a partir de um celular comum, sem a exigência de um custo extra para criar um controle do zero, reduzindo o custo do projeto que visa a produção em grande escala.

3.5 SISTEMA DE VÁCUO

O nosso objetivo era desenvolver um robô para limpeza de *pellets* de polipropileno e polietileno. Para isso, voltamos nossa atenção para a pesquisa de (JOON; KOWALCZYK, s.d.). Intitulada, "*Design of Autonomous Mobile Robot for Cleaning in the Environment with Obstacles*".

Neste artigo, extraímos diversas utilidades que certamente enriquecerão significativamente o nosso projeto, como o sistema de sucção criado por eles.

Este sistema demonstra a possibilidade de criar uma solução personalizada para o sistema de sucção, que serviu como base para o desenvolvimento do nosso próprio sistema. Este sistema precisou passar por modificações, especialmente na sua capacidade de carregar um volume adequado de *pellets* e descarregá-los posteriormente.

3.6 CIRCUITOS DE PROTEÇÃO

Uma parte importante do circuito elétrico é a proteção. A proteção é responsável pela segurança dos equipamentos em caso de falha, sua principal função é realizar uma rápida parada no sistema quando ele sofrer curto-circuito, ou estiver realizando uma função de forma anormal que possa causar danos ou seja capaz de danificar o resto do sistema.

Para a definição do sistema de proteção, foi adotado como referência o artigo do (MOTA, RENATO) intitulado “Cálculos de curtos-circuitos para estudos e análises da proteção de sistemas elétricos”. Este artigo apresenta diretrizes para a escolha da proteção elétrica mais adequada em casos de curto-circuito. No contexto do presente projeto, essa proteção foi considerada a mais relevante, uma vez que, por se tratar de um circuito de baixa tensão, outras formas de proteção se mostram menos essenciais.

3.7 DIMENSIONAMENTO DE BATERIA

Para a escolha da bateria adequada a um projeto e a definição do seu dimensionamento, é fundamental compreender as necessidades específicas do circuito. Isso inclui aspectos como o tempo de operação necessário sem recarga e os parâmetros elétricos do circuito.

Para uma avaliação mais precisa do dimensionamento da bateria, foi utilizado como referência o artigo de Rado e Serikaku, intitulado “Desenvolvimento de um robô cortador de grama elétrico”. Esse artigo fornece uma base metodológica para identificar os parâmetros necessários ao dimensionamento da bateria (corrente e tensão de cada componente), bem como para determinar as exigências do circuito. A aplicação dessas diretrizes permitiu uma análise mais criteriosa das necessidades energéticas do projeto.

3.8 ELÉTRICA

Com o desenvolvimento da parte autônoma em andamento, iniciou-se a análise da parte elétrica do robô, que representa o início do trabalho com os componentes físicos do projeto. Para orientar o desenvolvimento do sistema elétrico, utilizou-se como base o estudo de Asafa et al. (2018), intitulado "Development of a Vacuum Cleaner Robot".

Este trabalho de pesquisa forneceu uma compreensão inicial dos conceitos fundamentais aplicados ao robô de limpeza. Além de oferecer insights sobre os movimentos e a construção mecânica do robô, a pesquisa destacou-se pela apresentação de um diagrama elétrico, que servirá como referência para o desenvolvimento do nosso próprio sistema.

Entretanto, devido às condições mais severas do nosso ambiente de testes e operação, adaptações foram necessárias para atender às nossas demandas específicas. Entre essas adaptações, destacam-se o reforço das conexões elétricas, a inclusão de sistemas de segurança aprimorados, a duplicidade de conexões para garantir a funcionalidade em caso de falhas, e a implementação de

uma estação de descarte de resíduos (dock) integrada ao sistema de recarregamento da bateria. Essas modificações visam garantir a robustez e a eficiência do projeto em um ambiente desafiador.

3.9 ALIMENTAÇÃO ELÉTRICA

Para recarregar o robô uma fonte de corrente continua é necessária, tendo em vista a simplicidade e baixo custo foi optado por construir e utilizar uma fonte linear, uma vez que, as fontes de chaveadas são mais complexas, mais caras e tem algumas desvantagens como o baixo tempo de resposta a perturbações e a produção de ruídos na rede elétrica. Nesse caso serão os componentes para criar essa fonte são: um trafo 110 ou 220 volts para 12 volts, uma ponte de diodos, um circuito retificador de tensão e um limitador de corrente que é utilizado para evitar super aquecimento na bateria. Para auxiliar nesse projeto o Trabalho Final: Eletrônica analógica foi utilizado como material de apoio para o dimensionamento que foi usado na fonte desse robô.

4 METODOLOGIA

A seguir estão descritos os materiais, a metodologia e todas as métricas utilizadas durante o desenvolvimento do projeto.

4.1 MATERIAIS

Para realizar este projeto foi necessário adquirir alguns materiais para a confecção física do robô, os materiais são:

4.1.1 Software

- **Ubuntu 22.04 Jammy Jellyfish:** É um sistema operacional que serviu de base para instalação do ROS2 Humble Hawksbill.

- **ROS 2 Humble Hawksbill:** Como citado anteriormente no trabalho o ROS funciona como um sistema operacional de código aberto para aplicações robóticas, utilizamos a versão Humble Hawksbill por ter sido a última versão de longo suporte a ser compatível com o Ubuntu 22.04.

- **NAV 2:** Uma biblioteca de códigos de navegação inserida no ROS 2, ela foi utilizada por possuir uma implementação pronta dos métodos e códigos citados anteriormente como A*, Método de Monte Carlo e DWB, facilitando assim o processo de programação do robô.

- **SLAM toolbox:** Uma biblioteca de códigos também inserido em ROS 2 para implementação rápida do método SLAM.

- **Gazebo:** Simulador com compatibilidade com ROS 2 que será utilizado para os testes virtuais do robô.

- **NX:** Amplamente utilizado no mercado por conta da sua vasta gama de possibilidades e ferramentas integradas para desenvolvimento e simulação. Utilizado por pequenas *Start-Ups* até mesmo por multinacionais e principalmente por montadoras automobilísticas. Criado em 1847 pela Siemens® esse *software* é tanto CAD (*computer aided-design*) quanto CAM (*computer aided-manufacturing*), mas para as nossas necessidades utilizamos apenas o ambiente CAD, por conta de seu ecossistema mais voltado para desenho das peças no qual usamos para desenhar as partes mecânicas e estruturais do nosso robô.

4.1.2 Placas de controle

- **Raspberry Pi 4B 64 bits ARM:** Uma série de mini-computadores de placa única que pode ser ligado diretamente a um monitor e outros dispositivos como mouse e teclado, podendo ser utilizado para programação, automação e controle de robôs. Nós o utilizamos como cérebro do robô, aonde foi processada toda a lógica de mapeamento, localização e controle do robô, além da leitura dos sensores. O Raspberry tem a capacidade de funcionar com ubuntu 22.04, software utilizado para programação em ROS.

- **ESP32:** É uma série de microcontroladores de baixo custo e baixo consumo de energia que permite a criação de projetos complexos de IoT, robótica e automação residencial. Ele será ligado diretamente com o Raspberry, tendo como função de controlar os motores e receber os dados brutos do encoder para comunicá-los para o rasp. O ESP32 pode ser programado por diversos softwares como Arduino IDE, Espressif IoT Development Framework, Micropython, PlatformIO, etc.

4.1.3 Controle remoto

- **Controle de videogame XBOX:** Permite o controle manual do robô, sobrescrevendo os sistemas autônomos em necessidade de manobras não previstas ou limpezas em áreas não mapeadas. O modelo foi escolhido por conta de sua conectividade Bluetooth.

4.1.4 Sensores

- **RPLidar C1:** Sensor remoto que utiliza o princípio de triangulação do alcance de um laser para obter distâncias e outras informações sobre um objeto/obstáculo que esteja em uma determinada região. É o principal sensor utilizado para realizar a localização do robô e o mapeamento, existem diversos pacotes prontos em ROS para sua utilização.

4.1.5 Corpo do robô

- **Escovas laterais:** Escovas rotativas formadas por "mini-vassouras" ligadas aos motores laterais.

- **Carcaça:** Corpo físico do robô, o que definiu o formato e visual externo, além da posição dos componentes internos e formato do sistema de vácuo. O confeccionamos utilizando impressão 3D.

- **Rodas principais:** Utilizadas como meio de locomoção do robô, estão localizadas na parte traseira do robô e são movimentadas pelos motores com encoder.

- **Roda boba:** Utilizado em projetos robóticos a roda boba serve além de apoio, como uma forma de garantir o movimento gerado pelas rodas principais.

4.1.6 Motores, bateria e botão de emergência

- **Motores 12V DC com encoder:** Motor elétrico de baixa tensão com encoder que é um sensor eletromecânico com função de transformar posição em um sinal elétrico digital. Os motores são utilizados para locomoção do robô, já o encoder é utilizado na localização do robô conforme já citado anteriormente.

- **Motor de sucção 8V:** Motor responsável pela sucção da sujeira (neste caso os *pellets*).

- **Motores das escovas laterais 12V:** Motores encontrados em robôs de limpeza domésticos, localizado nas laterais frontais dos robôs com intuito de direcionar a sujeira dispersa para o centro onde se encontra a entrada do sistema de sucção.

- **Bateria 12V:** A bateria é responsável por fornecer a carga necessária para o funcionamento do robô.

- **Botão de emergência:** Responsável por realizar o desligamento emergência da alimentação do circuito em caso de necessidade de parada repentina.

4.1.7 Placa principal

A placa principal é uma Placa de Circuito Impresso (PCB do inglês *Printed Circuit Board*). A qual contém soquetes para a conexão das placas de controle e onde os demais subsistemas estarão posicionados, a placa foi desenhada no *software* Proteus, além de servir como um software de simulação para certificar o total funcionamento da placa.

- **Sensor de corrente INA219:** Sensor responsável por monitorar os parâmetros elétricos do sistema.

- **Transistores MOSFET:** Transistores utilizados para criar chaves (Switchs) que desconectam os motores e a bateria caso alguma grandeza elétrica saia dos parâmetros, evitando danos ao resto do sistema, além de poderem atuar como diodos que impedem o “refluxo” de corrente elétrica.

- **Relê:** Usado como uma chave para abrir e fechar o circuito elétrico através do efeito eletromagnético. Seu modelo básico possui dois terminais conectados por uma bobina. Ele

funciona como um eletroímã, pois a corrente conduzida através das bobinas cria um campo magnético em seu redor.

Quando o eletroímã é formado através da repulsão ou atração a chave altera seu estado (abrindo ou fechando). O relê é um componente semelhante ao transistor, também funcionando como uma chave, mas atuando de forma diferente.

- **Regulador de tensão:** Equipamento importante para muitos dos sistemas elétricos, ele é responsável em manter um determinado nível de tensão constante, pois diversos componentes necessitam de um valor de tensão específico para seu correto funcionamento.

Todo componente possui um datasheet onde tem todas suas informações principais, dentre elas suas condições de operação, que devem ser respeitadas para um funcionamento seguro.

- **Ponte H:** Usada principalmente em motores de corrente continua e permite que o motor funcione tanto no sentido horário quanto no anti-horário, através da mudança do sentido da corrente elétrica do circuito que altera a polaridade do motor e consequentemente sua rotação, essa mudança é possível com a abertura e fechamento de quatro chaves que funcionam em duas duplas.

- **Placa base para a impressão:** Essa é uma placa normalmente feita de fenolite ou fibra de vidro e posteriormente revestida com cobre para fazer as trilhas para a condução de corrente.

- **Estanho:** O principal material utilizado para a soldagem de componentes eletrônicos, responsável por segurar fisicamente os componentes e fazer os contatos elétricos. Ferro de solda: Uma ferramenta onde um pedaço de ferro é aquecido para fundir o estanho ao componente para criar a sustentação mecânica e o contato elétrico.

- **Resistores:** É um componente eletrônico passivo que regula o fluxo de corrente elétrica em um circuito, limitando-o conforme sua resistência elétrica medida em ohms. Ele é usado para controlar a voltagem e a corrente em uma variedade de dispositivos eletrônicos e circuitos, ajustando suas propriedades elétricas conforme necessário.

- **Capacitores:** É um componente eletrônico que armazena energia elétrica em um campo elétrico, usado para filtragem, acoplamento e armazenamento de carga em circuitos eletrônicos.

- **Transistores:** Um transistor é um dispositivo eletrônico semicondutor usado para amplificar ou controlar o fluxo de corrente em um circuito, essencial em eletrônica moderna.

4.2 MÉTODOS

4.2.1 Medidas da área de aplicação

Os autores deste trabalho obtiveram permissão por e-mail da empresa Braskem e, realizaram as medidas da área onde o robô seria instalado. O propósito desta fase era estabelecer um alicerce para as fases seguintes de construção de simulações e dimensionamento do projeto, já que o principal obstáculo na fase final seria o espaço físico para a passagem do robô.

Depois de realizar as medidas, foi estabelecido um limite máximo de 25 centímetros de largura, o ponto mais estreito pelo qual o robô precisaria passar. Isso serviu como referência para as próximas fases. Além disso, todos os obstáculos da região foram mapeados e suas respectivas distâncias foram traçadas para a elaboração do mapa tridimensional.

4.2.2 Preparação do ambiente de simulação virtual

Há várias opções para configurar o ambiente no Ubuntu com ROS 2, isso tornou essa uma das tarefas mais difíceis devido ao grande número de pacotes adicionais do ROS que precisaram ser instalados, como SLAM TOOLBOX e Nav2, já mencionados anteriormente.

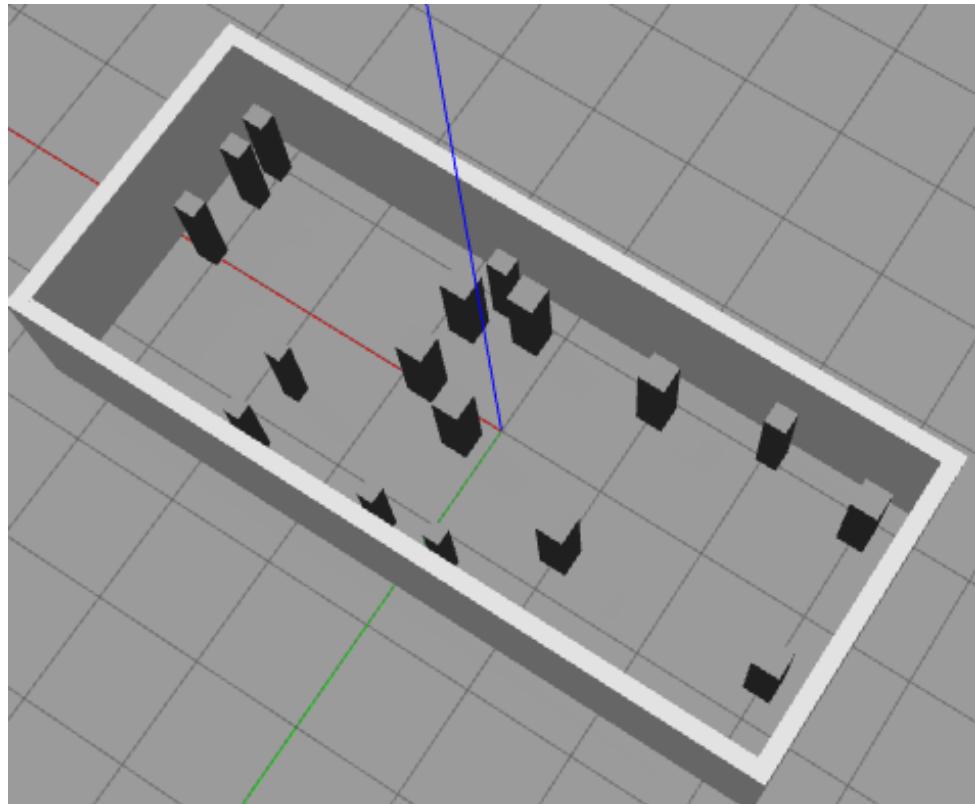
Também testamos vários métodos, incluindo docker e instalação local, mas o que obteve maior êxito foi a instalação através de uma máquina virtual, pois já tínhamos experiência prévia com esse método.

4.2.3 Simulação no GAZEBO

A partir das medições realizadas na área física do teste do robô, foi elaborado um mapa em escala 1:1 no programa GAZEBO utilizando as formas primitivas já disponíveis no simulador. O mapa gerado pode ser visto na Figura 6.

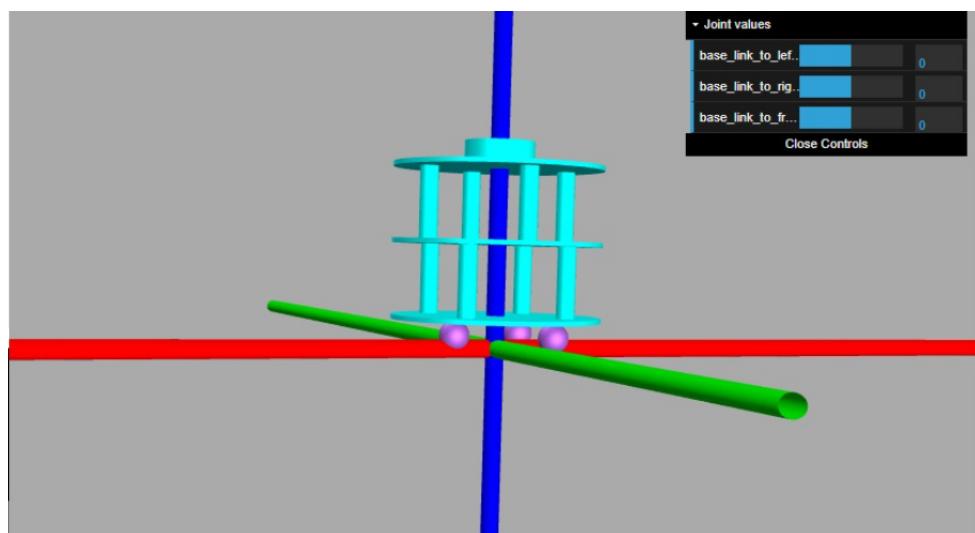
Em conjunto com o mapa, foi criado um modelo virtual do robô utilizando um conjunto de macros denominado xacro. Este conjunto tem como propósito simplificar e acelerar a criação do modelo virtual. O formato de arquivo escolhido foi o urdf. Novamente, utilizamos formas primitivas. Para simplificar o processo de modelagem, o torso dele, a parte central, foi aproximado a um cilindro, dado que a superfície física da aplicação não apresenta nenhuma proeminência que pudesse penetrar entre os discos dele. A Figura 7 contém a modelagem já concluída.

Figura 6 – Mapa simulado



Fonte: Autores

Figura 7 – Urdf do robô



Fonte: Autores

4.2.4 Codificação do planejamento e localização do robô em ROS

No ambiente virtual estabelecido pelo grupo, criou-se um pacote em ROS 2, onde foram gerados quatro arquivos de *launch*, além de alguns arquivos de configuração necessários para o funcionamento adequado da simulação. A implementação em si, por ser conceitos básicos

com pacotes já prontos, foi relativamente simples. Também aproveitamos códigos previamente criados para outras disciplinas, o que simplificou o processo.

Os dois primeiros arquivos *launch* gerados, que seriam reaproveitados em outros momentos, foram utilizados para iniciar a simulação no Gazebo com um mapa fornecido e para carregar o robô na simulação com seus respectivos controles e softwares necessários, como *teleop twist* e *RVIZ*.

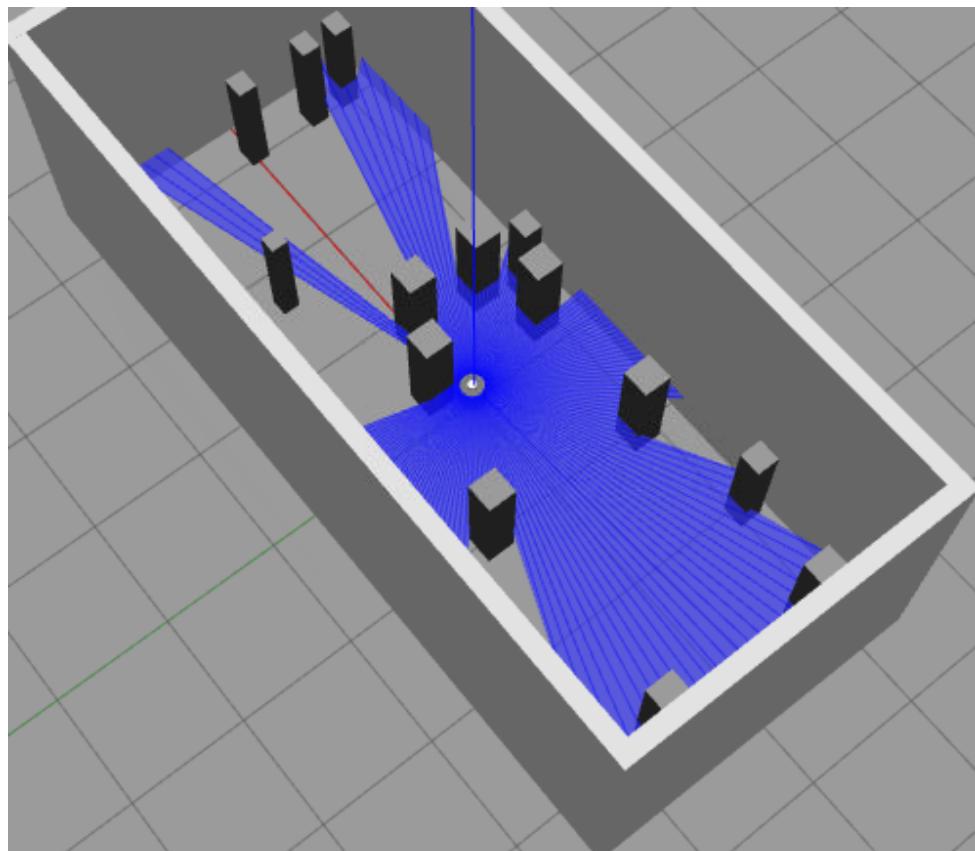
Outro arquivo de *launch* foi criado para o mapeamento com SLAM toolbox. Ele aproveita os dois arquivos mencionados anteriormente e também inicia os nós necessários para o mapeamento com o SLAM.

Finalmente, o *launch* para navegação, semelhante ao do SLAM, utiliza os dois primeiros mencionados e, em conjunto com o mapa gerado pelo SLAM, carrega os nós necessários para uma navegação autônoma por pontos definidos pelo usuário.

Juntamente com esses nós foram criados arquivos de configuração para o RVIZ e arquivos de parâmetros para o robô.

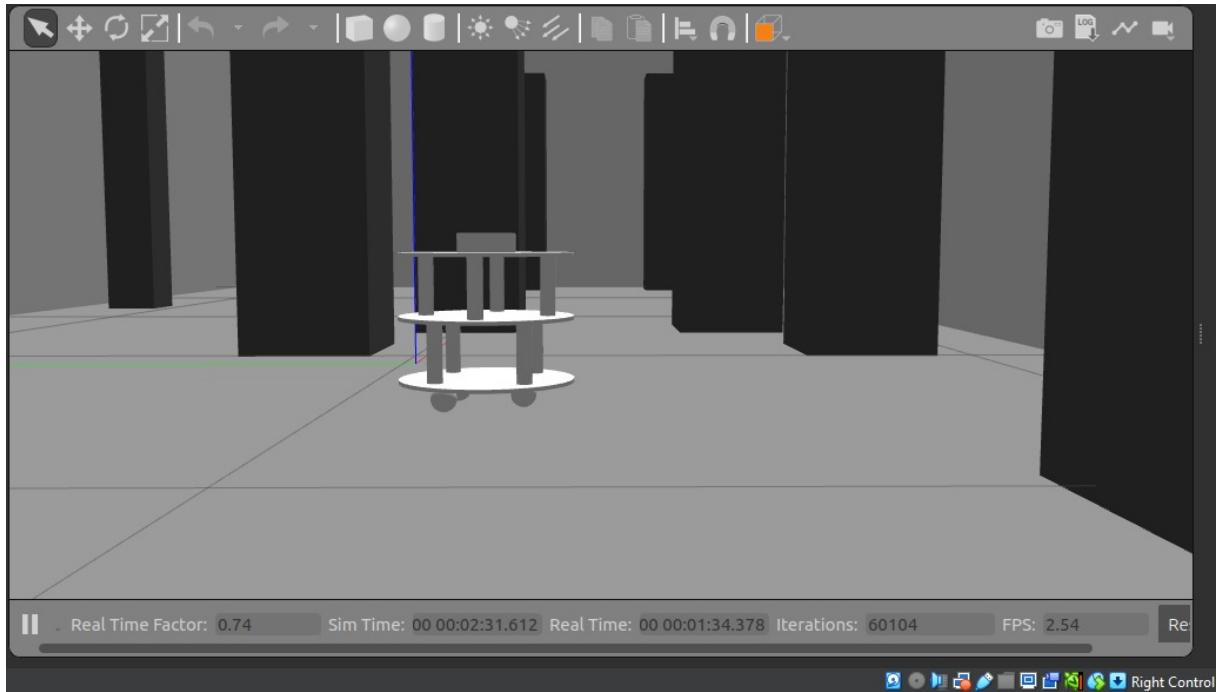
As Figuras 8 e 9 apresentam o robô já em Gazebo sendo simulado.

Figura 8 – Navegação em Gazebo



Fonte: Autores

Figura 9 – Robô dentro do mundo simulado



Fonte: Autores

Com o ambiente virtual operando, começamos a ajustar os parâmetros para a navegação do robô. Empregamos como referência os parâmetros presentes na documentação do robô *Turtlebot 3*, devido à semelhança de aplicação e formato entre os robôs. Posteriormente, era necessário ajustar esses valores em quatro circunstâncias:

- **Especificações do robô:** Definições básicas de operação, tais como velocidades máximas e aceleração. Existem também alguns parâmetros particulares de alguns planejadores, como o suavizador de velocidade usado pelo DWB.

- **Ajuste do mapa de custos:** Existem dois mapas de custos, um local e outro global. Ambos encarregados de determinar o quanto próximo o robô se aproximará dos obstáculos, esta fase é crucial, pois, se mal configurada, o robô não conseguirá transitar em áreas estreitas mesmo que exista espaço suficiente, ou, em situações extremas, poderá colidir contra obstáculos.

- **Estabelecimento do percurso:** Determina a posição inicial do robô e os locais pelos quais ele deve transitar. Estas localizações são definidas por uma coordenada X,Y e seu ângulo de rotação.

- **Definição dos planejadores:** Esta é a seção onde há a maior quantidade de modificações dos parâmetros. Na documentação do Nav2, existem as modificações necessárias para usar determinados planejadores. No entanto, outros parâmetros precisam ser adaptados de acordo

com as particularidades e restrições do robô. Devido ao grande volume de alterações, esta fase foi a que apresentou a maior parte dos problemas. Um deles foi a execução de curvas muito abertas pelo robô ao usar o DWB. Essa questão foi resolvida com soluções fornecidas por programadores que trabalham com o DWB no *GitHub*¹ e no *StackExchange*², onde os parâmetros de escala *BaseObstacle.scale*, *PathAlign.scale*, *Goallign.scale* e *RotateToGoal.scale* deveriam ser zerados, como medida temporária de solução.

Com a programação completa, era preciso definir o melhor planejador global e local, bem como o melhor percurso para a região. Portanto, realizamos vários experimentos para determinar a combinação mais adequada para esta situação, o que também resultou na determinação do tempo médio de limpeza. Os experimentos feitos e seus respectivos resultados estão disponíveis na Seção 6.1.

4.2.5 Subsistemas do robô

O robô possui dois subsistemas. O primeiro sendo o sistema de aspiração de pó em si, que consume a maior parte da bateria devido à presença de grande parte dos motores, três no total, um para produzir o vácuo através do sistema de tubos com reservatório e dois para mover as pás que ajudarão a direcionar os *pellets* para o aspirador. Com isso em mente, o código do aspirador de pó só é ativado durante o uso, com o objetivo de prolongar a vida útil da bateria.

O segundo subsistema é o sistema de controle remoto do robô. Quando o controle for ativado, a parte autônoma do robô é desligada e o controle passa a ser totalmente do operador. A comunicação é realizada através do módulo *bluetooth*, uma vez que a distância máxima que o controle requer não é extensa. O controle escolhido foi de um console, especificamente um Xbox, para simplificar inicialmente a conexão e o controle do robô. No entanto, isso poderia ser um ponto de melhoria no projeto. A verificação da efetividade desta fase consistia em confirmar se ao acionar o controle os movimentos autônomos do robô cessam, além de executar um teste de alcance máximo de comunicação do controle.

4.2.6 Dimensionamento do robô

Após obter todas as medidas por onde o robô deveria passar, foi desenvolvido 2 protótipos de medidas diferentes. Conforme visto no nosso ambiente de testes, que era a área de ensaque

¹*GitHub*: <https://github.com/ros-navigation/navigation2/issues/938>

²*StackExchange*: <https://robotics.stackexchange.com/questions/102284/dwb-controller-struggles-with-navigation-in-tight-scenarios>

dentro da Braskem, o nosso robô possui vários caminhos em que ele pode não passar dependendo de seu tamanho, por isso foi desenvolvido duas possibilidades.

A primeira sendo um protótipo menor com no máximo 200mm de altura e 150mm de diâmetro, com esse tamanho ele seria capaz de passar por debaixo da grade de segurança e se esgueirar por algumas frestas, mas essa possibilidade geraria uma capacidade de carga menor devido ao seu tamanho.

Já a segunda possibilidade era faze-lo com os mesmos 200mm de altura e até 250mm de diâmetro assim, poderíamos desce-lo por uma escada e ele teria uma área mais restrita, mas em compensação ele teria uma capacidade de carga maior.

4.2.7 Impressão do corpo

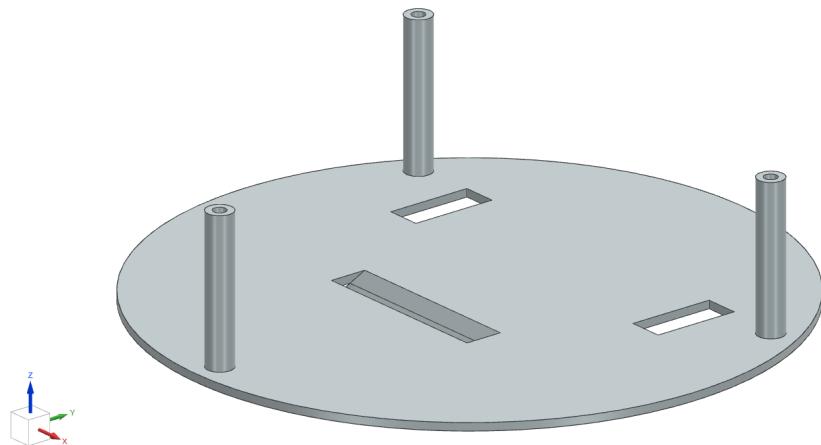
A impressão do robô foi realizada da seguinte maneira, com os desenhos e projetos devidamente feitos dentro do *software NX®*, os arquivos foram salvos em ".obj", esse tipo de arquivo é um tipo universal para impressoras 3D, no qual muitas fabricantes de impressoras utilizam. Já referente ao filamento utilizado para o projeto, foram usados os disponíveis para nós dentro dos laboratórios da FEI, nos quais se localizam as impressoras 3D.

4.2.8 Desenhos no NX

Conforme planejado, iniciamos a elaboração dos nossos primeiros rascunhos no NX. Como o nosso robô não poderia ter um diâmetro muito amplo devido ao limitado espaço disponível na linha da Braskem, para otimizar o espaço, decidimos elevar a altura do nosso robô. Como todo o equipamento é elevado do chão, o robô não corria o risco de colidir com a parte superior.

Ele foi construído com todos os andares de 230mm de diâmetro, o que ocasionou o descarte da opção de 150mm devido à necessidade de mais espaço para os componentes. Sua espessura é de 3mm e possui apenas 3 furos para a fixação dos espaçadores. Comentando brevemente sobre os elementos de fixação para as peças estruturais, optou-se por parafusos M5 para a fixação de todas as peças, tanto pelo seu diâmetro quanto pelo seu preço acessível e facilidade de compra. A primeira base do robô já com os espaçadores inferiores, pode ser vista na Figura 10.

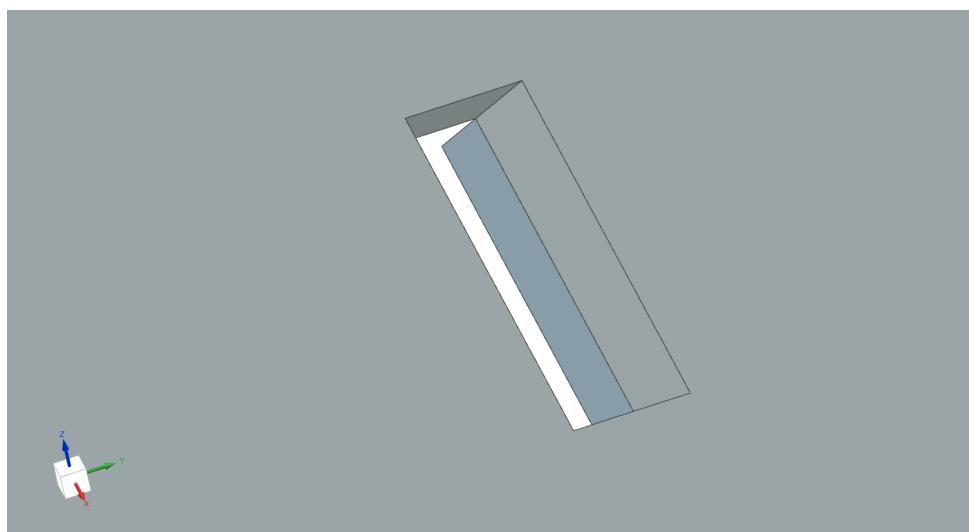
Figura 10 – Base com Espaçadores



Fonte: Autores

A decisão de utilizar apenas 3 perfurações em vez dos 4 furos habituais para posicionar a estrutura foi para manter a estabilidade da estrutura enquanto abrimos a parte traseira para facilitar a limpeza do reservatório de *pellets*. Também já definimos algumas posições na base, como a entrada dos *pellets*, podendo ser observada na Figura 11.

Figura 11 – Entrada para os *Pellets*

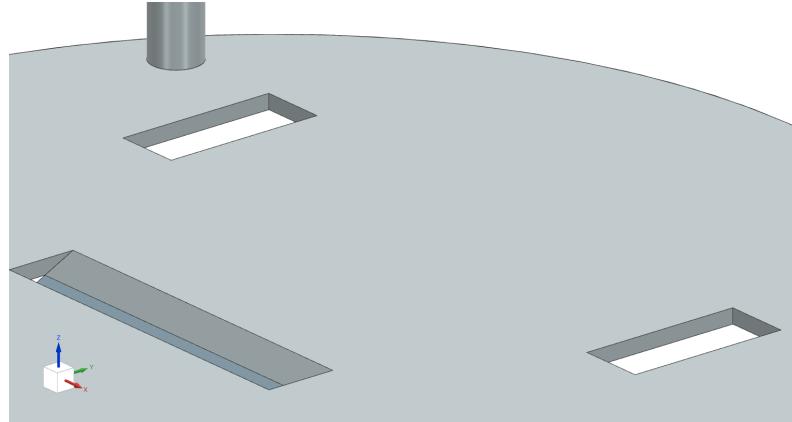


Fonte: Autores

Em seguida, colocamos as aberturas para que as rodas pudessem atravessar a base e mover o robô. Essa parte específica das rodas poderia ser alterada dependendo do tamanho das rodas e motores que precisaríamos utilizar. No entanto, contávamos que, por costume, neste tipo de robô, devido ao tamanho reduzido dos motores e rodas, elas deveriam ser posicionadas na parte interna e de forma recuada, como exemplificado na Figura 12, permitindo que uma

roda castor fosse colocada na frente, fazendo com que, apenas os dois motores conduzissem a movimentação do robô.

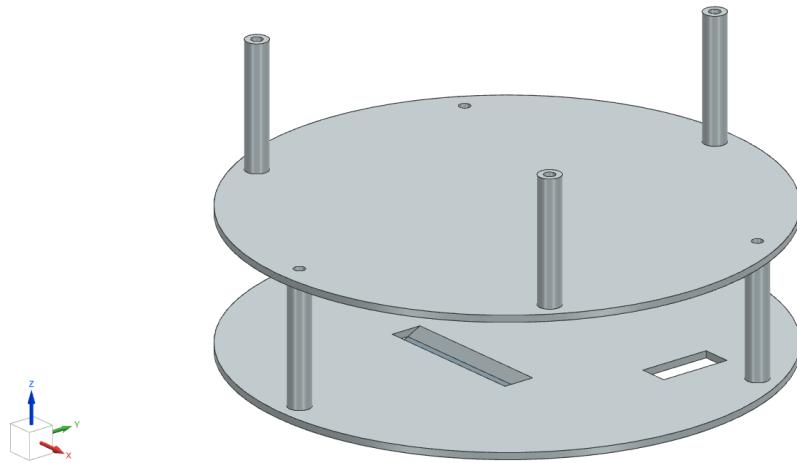
Figura 12 – Rasgos para passagem das Rodas



Fonte: Autores

Posteriormente, construímos o segundo andar do nosso robô, conforme ilustrado na Figura 13. Neste andar, planejamos instalar toda a nossa parte eletrônica, deixando a parte inferior reservada para os componentes mais pesados, como a bateria e os motores, que seriam utilizados tanto para mover quanto para sugar os *pellets*.

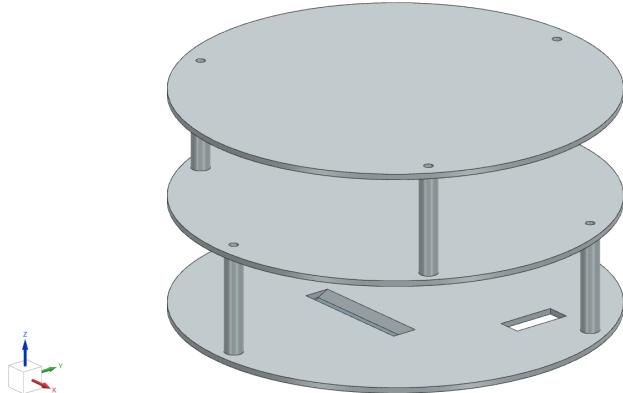
Figura 13 – Andar inferior e intermediário com Espaçadores



Fonte: Autores

Prosseguindo com os andares, o último andar, mostrado na Figura 14, era destinado à instalação do sensor *LiDAR* e de outros componentes, caso fosse necessário. O mais importante era que o *Lidar* fosse instalado neste andar, permitindo que ele escaneie todo o seu entorno sem que os espaçadores interferissem na leitura do laser.

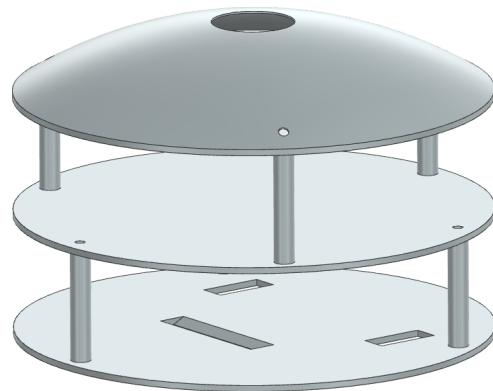
Figura 14 – Carcaça completa



Fonte: Autores

Na parte superior, optamos por colocar uma cúpula com um furo no centro, como o da Figura 15, para que o laser do *LiDAR* pudesse, mesmo com a cúpula, escanear o ambiente circundante. Esta peça era apenas um complemento ao nosso projeto, destinada tanto à proteção dos componentes principais contra quedas de material, quanto para que, se algum *pellet* caísse sobre o robô durante a limpeza do local, ele iria rebater na cúpula, evitando que o material ficasse preso em seus componentes e que, ao cair no solo, ele pudesse ser recuperado.

Figura 15 – Carcaça Completa + Cúpula



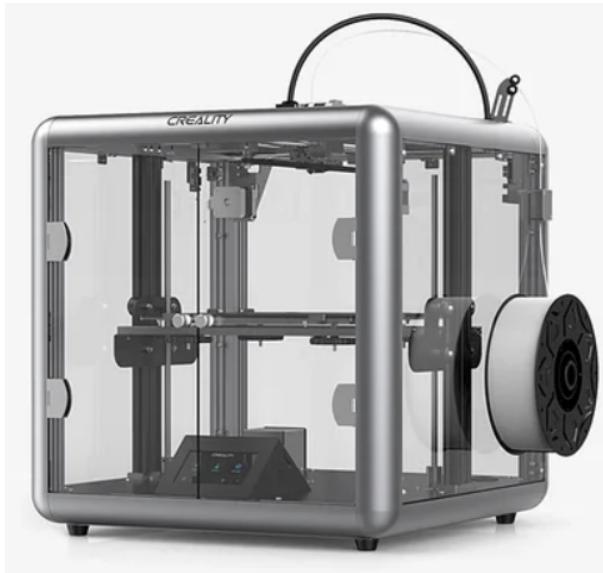
Fonte: Autores

4.2.9 Conversão de arquivos *.obj* para *.gcode*

Após a criação dos desenhos, a próxima etapa foi a conversão dos arquivos para *.gcode*. O software NX salva os arquivos por padrão em *.prt*, um formato criado pela própria *Siemens*

para arquivos *CAD*. No entanto, a impressora que dispúnhamos era uma Sermoon D1 da *Creality*, idêntica ao da Figura 16, localizada no laboratório de robôs móveis da FEI, que só aceitava arquivos no formato *.gcode*.

Figura 16 – Impressora 3D, Sermoon D1, imagem retirada do site oficial do fabricante.



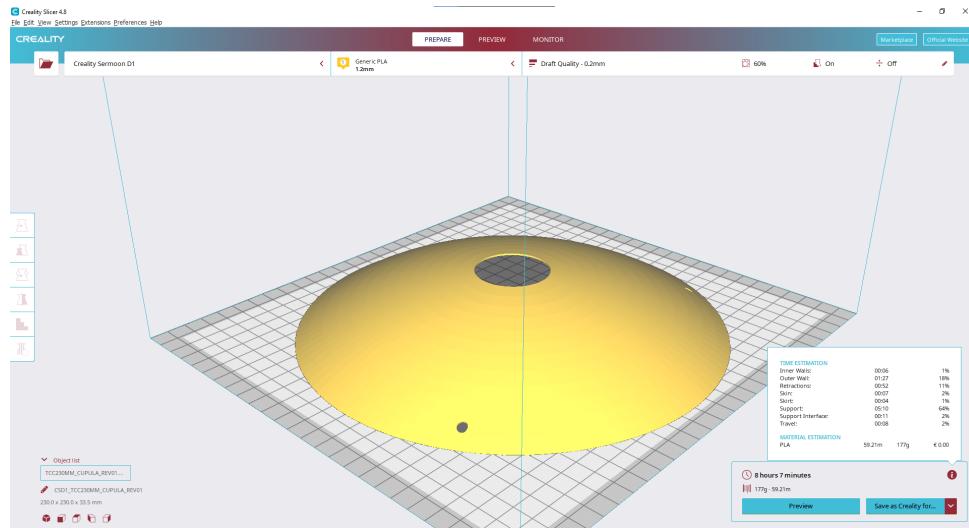
Fonte: (CREALITY, s.d.), 2024

O programa NX não possibilita criar um arquivo *.gcode* diretamente do nosso desenho *CAD*. Para isso, precisávamos convertê-lo em um arquivo *.obj*. Os arquivos *.obj* são amplamente usados em impressões 3D, pois são arquivos destinados a representações tridimensionais. É a partir deles que conseguimos criar os arquivos que as impressoras conseguem usar. Converter de *.prt* para *.obj* não representa um desafio, já que o próprio NX nos possibilita criá-lo a partir de nossos desenhos.

Com os arquivos *.obj* criados, foi necessário o uso de um software adicional para a impressão das partes do nosso robô. O *Creality Slicer* é um programa desenvolvido pela *Creality* para uso em suas impressoras. O arquivo que usamos é o *.obj*, e a partir dele pudemos simular o consumo de material e o tempo necessário para a produção da peça. Como podemos observar a seguir, uma representação da produção da nossa cúpula.

Na Figura 17, empregamos o PLA 1,2mm, ativando os suportes para evitar que a cúpula se colapse durante a impressão. Esta simulação teve uma duração estimada de 8h e 7min e utilização de aproximadamente 177g de material. Depois de simular a peça no *Creality Slicer*, pudemos salvá-la em formato *.gcode*. Todas as peças precisavam passar por essas fases, e qualquer alteração nos desenhos, todos os passos de conversão deveriam ser repetidos.

Figura 17 – Simulação para impressão da Cúpula

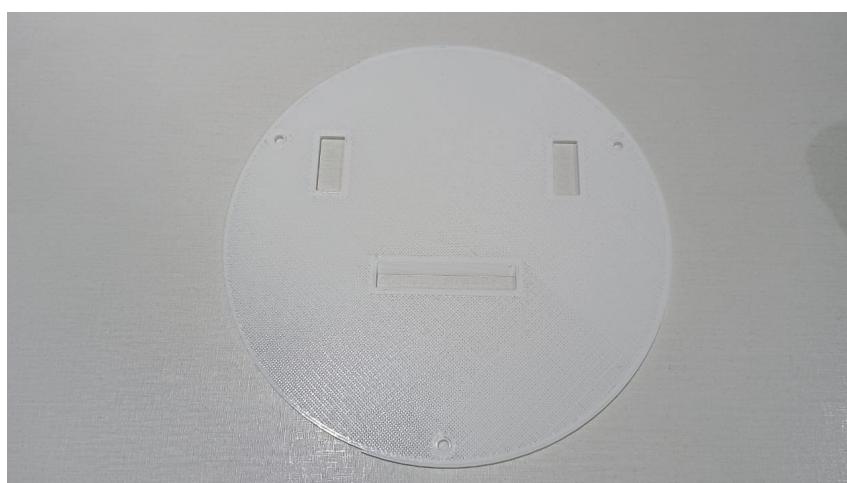


Fonte: Autores

4.2.10 Impressão das peças

Após salvar os arquivos em *.gcode*, iniciamos a impressão das partes. A primeira parte a ser impressa foi a base inferior do nosso robô, como nas Figuras 18 e 19. Já que ela era o elemento estrutural que mais sofreria deformações, sejam elas durante seu movimento dentro da área, ou principalmente por conta dessa peça precisar resistir aos maiores esforços mecânicos, já que todos os componentes mais pesados estavam planejados para estarem na parte mais baixa, diminuindo assim o seu centro de gravidade. Utilizando o PLA branco, conseguiríamos ver se a resistência desse material atenderia o nosso propósito, ou se os desenhos precisariam passar por alguma alteração.

Figura 18 – Base Inferior



Fonte: Autores

Figura 19 – Base Inferior com espessura 3mm



Fonte: Autores

O único material disponível no laboratório era o PLA branco, já que a impressora não conseguia imprimir corretamente em ABS, um material mais apropriado para nosso propósito. O ABS é um filamento que suporta mais tensões mecânicas do que o PLA, um material mais voltado para acabamentos. Após a impressão da primeira peça, optamos por aumentar sua espessura em 2mm, pois acreditamos que a base feita com o PLA com 3mm de espessura não teria margem de segurança o suficiente para prevenir uma quebra por conta das tensões em cima da peça, com a mudança feita a base totalizou 5mm de espessura. Com essa alteração realizada, foi impressa a base intermediária, como visto nas Figuras 20 e 21.

Figura 20 – Base Intermediaria



Fonte: Autores

Figura 21 – Base Intermediaria com espessura 5mm



Fonte: Autores

Com o sucesso da nossa segunda impressão, modificamos as demais bases para 5mm. A terceira impressão foi feita nos espaçadores das bases, como mostrado na Figura 22, uma vez que essas não necessitavam ser modificadas.

Figura 22 – Espaçadores



Fonte: Autores

Os espaçadores impressos demonstraram uma resistência maior do que esperávamos inicialmente, principalmente devido ao seu tamanho reduzido de apenas 10mm de diâmetro com 5mm de furo passante, resultando em uma peça com uma densidade de material bastante elevada e de excelente qualidade.

Durante o período de fabricação das peças, a impressora que usávamos precisou passar por manutenção, até que ela retornasse, as outras peças foram produzidas no laboratório *Maker* da nossa universidade, que é aberto ao público e todos os cursos podem utilizá-lo, desde que assinem um documento autorizando o uso e haja um técnico disponível no momento para acompanhar o

aluno. Como é de uso público, não tivemos a opção de escolher o material a ser usado devido à alta rotatividade de projetos. No entanto, o material padrão utilizado seria o ABS preto, que se adequaria perfeitamente ao nosso propósito.

Utilizando a alteração dos desenhos, modificamos a base inferior para acomodar as rodas recém-chegadas que eram maiores do que a marcação na peça original.

Com a disponibilidade do laboratório, a documentação assinada pelo nosso professor orientador e os desenhos devidamente corrigidos, a base inferior foi impressa novamente, como visto na Figura 23.

Figura 23 – Base inferior em ABS



Fonte: Autores

E por fim a a base superior, na Figura 24, para que pudéssemos começar a montagem da carcaça e da instalação de seus componentes.

Figura 24 – Base superior em ABS



Fonte: Autores

Após a impressão de todos os componentes estruturais básicos do nosso robô, foram comprados os elementos de fixação para que a montagem da estrutura pudesse ser feita, entre eles estavam os parafusos M5 maquina, com cabeça sextavada para facilitar o uso de uma ferramenta, como uma chave combinada Nº 8, por conta do espaço estreito e algumas arruelas M5 para distribuir melhor a carga pelo parafuso, evitando assim ao máximo deformações no material, como resultado final toda a estrutura básica foi montada como mostrado na Figura 25.

Figura 25 – Estrutura Completa



Fonte: Autores

Com a estrutura básica totalmente impressa, localizamos uma máquina no laboratório do Maker para a impressão da nossa cúpula. Até o momento em que estamos escrevendo este texto, ela está em produção. Assim como as outras peças produzidas no Maker, ela também será produzida em ABS preto, um detalhe significativo para o nosso projeto, pois é um item adicional destinado à proteção dos componentes superiores.

4.2.11 Montagem do circuito

Com a definição dos componentes principais, iniciou-se o processo de construção do circuito. A fonte de alimentação escolhida foram duas baterias de 12 volts, responsável por energizar todo o sistema, incluindo os motores (dois motores de pás, um motor aspirador e dois motores de tração) e o Arduino Uno. A primeira etapa do desenvolvimento concentrou-se nos motores de tração, que requerem uma ponte H para controlar tanto o sentido de rotação quanto a velocidade. Para os testes iniciais, a ponte H foi conectada à bateria, a dois motores menores, com a mesma tensão, e a um microcontrolador Arduino Uno responsável pelo comando. Na segunda etapa, realizou-se a validação dos motores de limpeza (motores de pás). Como ambos operam

em 12 volts, não foi necessário o uso de reguladores de tensão. Os motores foram conectados diretamente a um relé, utilizado para o acionamento, que por sua vez foi ligado à bateria e ao Arduino que realizou o teste de acionamento. A etapa seguinte foi dedicada ao teste do motor aspirador. Por se tratar de um motor de 8 volts, foi necessário integrar ao circuito um regulador de tensão (de 12V para 8V), além de um relé, antes de conectá-lo ao motor. Após a validação do sistema, por meio de testes preliminares, iniciou-se a montagem do circuito. Nessa fase, foram realizados os primeiros testes de integração entre o Arduino Uno e todo sistema elétrico citado, foi adicionado um sensor ultrassom, capaz de identificar objetos a frente do robô e direcioná-lo para a direita, ao circuito para um teste mais preciso do comando de cada componente, marcando um avanço significativo no desenvolvimento do projeto.

4.2.12 Criação do sistema de proteção

No desenvolvimento do projeto do robô aspirador, foi implementado um circuito de proteção essencial para garantir a segurança e o bom funcionamento dos componentes eletrônicos, utilizando duas abordagens principais: a proteção por fusível e a utilização de um botão de emergência. O fusível foi colocado entre a bateria de 12V e os circuitos do robô, com o objetivo de proteger os componentes contra sobrecorrentes que poderiam causar danos, como aos motores e ao Arduino. Um fusível de 7A foi escolhido, interrompendo automaticamente o fluxo de corrente em casos de sobrecarga e evitando falhas no sistema. Durante os testes, o fusível funcionou adequadamente, desligando-se corretamente ao detectar correntes excessivas e protegendo os circuitos. Além disso, foi integrado ao projeto um botão de emergência, que possibilitou a desativação imediata do robô em situações de falha ou risco de danos, como em defeitos nos motores ou no sistema de controle. O botão foi instalado diretamente no circuito de controle do Arduino, de modo que, ao ser acionado, cortava a alimentação elétrica do robô e interrompia sua operação de forma rápida e segura. Essa medida foi validada nos testes, funcionando de maneira eficaz e permitindo uma parada imediata do robô quando necessário. Assim, a combinação dessas duas medidas de proteção – fusível e botão de emergência – assegurou a operação segura e confiável do robô aspirador, protegendo os componentes eletrônicos e garantindo a integridade do sistema durante seu funcionamento, minimizando riscos de danos e falhas operacionais.

4.2.13 Sistema de baterias

O cálculo do tempo de duração das baterias foi realizado levando em consideração os componentes principais do sistema: o microcontrolador Arduino Uno, os motores de tração, os motores de pás e o motor de sucção. O sistema foi alimentado por duas baterias de 12V e 2200mAh (ou 2,2Ah) conectadas em paralelo, o que resultou em uma capacidade combinada de 4,4Ah, mantendo a tensão de 12V. Esse arranjo foi adotado para garantir maior autonomia ao robô, pois a conexão em paralelo aumenta a capacidade de carga da bateria, sem alterar a tensão fornecida. Para calcular a duração das baterias, primeiramente foi necessário determinar o consumo de corrente máxima de cada componente do robô. O motor de tração, que possui dois motores de 12V com consumo de 1,5A cada, apresentou um consumo total de 3A. O motor de pás, com dois motores de 12V consumindo 1A cada, apresentou um consumo total de 2A. O motor de sucção, operando a 8V e consumindo 0,5A, foi alimentado por um regulador de tensão. Considerando a perda de eficiência no processo de conversão de tensão, o consumo de corrente necessário da bateria foi de aproximadamente 0,33A para o motor de sucção. Por fim, o Arduino Uno, com consumo de 0,05A, também foi considerado no cálculo total de consumo. O consumo total de corrente foi então calculado somando o consumo individual de cada componente:

$$\begin{aligned} I_{\text{total}} &= I_{\text{motores de tração}} + I_{\text{motores de pá}} + I_{\text{bateria}} + I_{\text{arduino}} = 3 + 2 + 0,75 + 0,05 \\ &= 5,61 \text{A} \end{aligned}$$

Com a corrente total consumida de 5,61A e a capacidade das baterias de 4,4Ah (considerando duas baterias conectadas em paralelo), o tempo estimado de operação foi calculado utilizando a fórmula:

$$T = \text{Capacidade total da bateria} / \text{Corrente total consumida} = 4,4 \text{Ah} / 5,61 \text{A} = 0,785 \text{h}$$

Portanto, o tempo estimado de duração das baterias foi de aproximadamente 0,785 horas, ou cerca de 47 minutos. É importante destacar que esse cálculo oferece uma estimativa teórica, considerando condições ideais de operação. Na prática, a autonomia do robô pode ser afetada por diversos fatores, como a eficiência real do regulador de tensão, que pode variar dependendo do tipo de regulador utilizado, a variação no consumo dos motores em função da carga e das condições de operação (por exemplo, em superfícies irregulares), e a condição das baterias, que pode ser influenciada pela temperatura, idade e taxa de descarga. Portanto, é fundamental realizar testes práticos para validar o tempo de duração da bateria sob as condições operacionais específicas, garantindo que o robô funcione de maneira eficiente e dentro dos parâmetros desejados.

4.2.14 Configuração dos motores

O motor responsável pela função de aspiração opera com uma tensão nominal de 8 volts e consome uma corrente de 0,5 amperes. Para integrá-lo ao sistema, ele é conectado a um relé que é ativado por um sinal de 5 volts proveniente do Arduino Uno. Além disso, o motor é alimentado por um regulador de tensão, que converte os 12 volts fornecidos pela bateria para os 8 volts necessários para seu funcionamento seguro e eficiente.

Por outro lado, o motor que opera a pá apresenta uma tensão nominal de 12 volts e um consumo de corrente de 1 ampere. Esse motor é conectado diretamente ao relé, sem necessidade de regulação adicional, uma vez que sua tensão de operação é compatível com a saída da bateria.

Os motores de locomoção são alimentados por meio de uma ponte H, que realiza o controle da direção e velocidade desses motores. A ponte H está conectada tanto à bateria, responsável por fornecer a energia necessária, quanto ao sistema de comunicação (Arduino Uno), que regula os comandos enviados para o controle dos motores juntamente com o sensor de objetos.

Essa configuração assegura o funcionamento integrado e coordenado de todos os componentes, permitindo a execução das funções previstas para o dispositivo. A escolha dos componentes, como as células de bateria e os reguladores de tensão, foi feita com base nas 50 especificações de consumo e tensão de cada motor, visando eficiência energética e compatibilidade técnica. O uso de relés para ativação dos motores garante maior controle e segurança na operação, enquanto o regulador de tensão e a ponte H asseguram a distribuição adequada da energia para os diferentes subsistemas. Por fim, foi adicionado o botão de liga/desliga e o botão de emergência, ligados diretamente a bateria.

4.3 MÉTRICAS

4.3.1 Simulação

Para todos os aspectos de mapeamento, planejamento, localização e navegação, a principal forma de avaliar esses processos era por meio do simulador, que permitia uma visão de como o robô se comportaria no mundo real e a eficácia dos processos, cada um com uma métrica distinta:

- **Mapeamento:** Ao comparar o mapa gerado pelo robô através do código com o mapa criado em escala 1:1 da área de atuação, podíamos determinar se a fase de mapeamento correspondia ao resultado esperado. Poderiam haver diferenças devido a falhas nos sensores ou

na rota selecionada para o mapeamento. Para que essa fase fosse bem-sucedida, o mapa virtual deveria mostrar os obstáculos nos mesmos pontos do local real analisado e com dimensões equivalentes.

- **Planejamento:** Os locais pelos quais o robô deveria transitar foram inseridos manualmente no código. O teste realizado era se ele conseguia superar todos os pontos estabelecidos com êxito, sem se prender em estruturas ou se chocar com obstáculos, dentro de um tempo plausível quando comparado com a duração da bateria e seguindo o caminho estipulado pelo algoritmo global com a maior precisão possível.

- **Localização:** Ao comparar a interface gráfica do simulador GAZEBO com as coordenadas internas do código do robô, pudemos determinar se a discrepância entre elas poderia afetar negativamente a orientação do robô. Na própria interface com o RVIZ pode-se visualizar quão dispersas as partículas utilizadas pelo método de Monte Carlo estão do robô real, podendo assim ter uma confirmação se o algoritmo está funcionando corretamente.

- **Navegação:** Nos testes conduzidos para a obtenção do tempo médio, conforme explicado na Seção 4.3.2, obtivemos dois indicadores de tempo: o tempo de execução (tempo necessário para um procedimento de limpeza completo do robô) e o tempo fora do trajeto (tempo em que o robô permaneceu parado ou se movia fora do trajeto planejado). Dessa forma, podemos determinar a taxa de efetividade da navegação, que é a relação entre o tempo total de execução e o tempo fora do percurso.

4.3.2 Área efetiva de limpeza, tempo médio de limpeza

Com a simulação em total funcionamento, a utilizamos para realizar uma série de testes com fins de obter os dois principais fatores para a comprovação da efetividade do robô e do cumprimento dos objetivos estabelecidos, são eles:

- **Área efetiva de limpeza:** Comparação entre a área total e a área coberta pelo robô ao realizar o percurso de limpeza.

- **Tempo médio de limpeza:** Através da obtenção do tempo de execução do robô em um total de dez testes individuais, calculamos a média de tempo para a realização de uma limpeza.

4.3.3 Precisão e Alcance do sensor ultrassom

O sensor ultrassom utilizado no robô aspirador teve a função crucial de detectar obstáculos e garantir a navegação eficiente do dispositivo. Para assegurar que o robô conseguisse evitar

obstáculos de maneira eficaz, foram realizados testes que avaliaram tanto a precisão quanto o alcance do sensor em diferentes distâncias.

A precisão do sensor ultrassom foi verificada por meio da comparação entre a distância real do obstáculo e a distância medida pelo sensor em uma variedade de cenários, considerando diferentes tipos de superfícies e materiais refletivos. Foi testado o comportamento do sensor em distâncias que variaram de 5 cm a 3 metros, com o intuito de validar sua capacidade de detectar objetos próximos e distantes com a acuracidade necessária para a navegação autônoma.

Adicionalmente, o alcance do sensor foi medido em diferentes condições, incluindo diferentes ângulos de incidência e possíveis interferências, a fim de garantir que o robô fosse capaz de identificar obstáculos e ajustar sua trajetória antes de uma colisão. O desempenho do sensor foi considerado satisfatório caso ele tenha conseguido detectar obstáculos com precisão de até 2 cm em distâncias de 5 cm a 3 metros, com tempo de resposta inferior a 100 ms.

Esses testes foram essenciais para garantir que o sistema de navegação do robô aspirador funcionasse adequadamente, proporcionando segurança e eficiência durante sua operação em ambientes variados.

4.3.4 Tempo Médio de Operação da bateria

A autonomia do robô aspirador foi avaliada com base no tempo médio de operação com uma carga completa da bateria. Foi medido o tempo em que o robô permaneceu em funcionamento contínuo até a bateria atingir seu nível mínimo de carga. A análise considerou diferentes condições de operação, como variação de superfícies e intensidades de uso, para simular o comportamento do robô em situações de seu cotidiano. O objetivo foi garantir que a autonomia atendesse a uma duração satisfatória para o uso no ambiente esperado (área industrial de embalagem). O desempenho da bateria foi considerado adequado para realizar a limpeza do local, levando em conta os parâmetros: área estimada do local e obstáculos fixos.

4.3.5 Eficiência no Controle de Motores

A eficiência da ponte H no controle dos motores de tração e sucção foi verificada por meio de testes que analisaram sua capacidade de converter sinais elétricos em comandos precisos para o funcionamento dos motores DC. Durante os experimentos, foi monitorada a resposta dos motores a diferentes sinais enviados pela ponte H, avaliando o desempenho na regulação da velocidade, direção e torque. A análise focou em aspectos como a estabilidade do controle, a

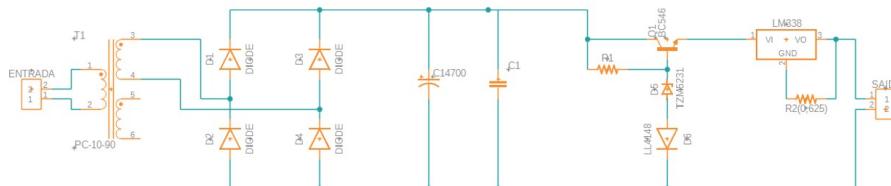
precisão na execução dos comandos e a capacidade da ponte H de manter o controle eficiente dos motores, mesmo sob variações de carga e condições operacionais. A eficiência foi considerada satisfatória quando os motores apresentaram um controle suave e constante, sem falhas ou perda significativa de desempenho durante o funcionamento.

4.3.6 Carregador da bateria

Com a bateria dimensionada para o projeto, é necessário um carregador para energizá-la. Tendo isso em mente verificamos no *datasheet* da Melasta, a fabricante das células íon de lítio que usamos, para identificar os parâmetros de carregamentos das células, e com isso chegamos a conclusão que a bateria pode ser carregada com uma fonte de 12 volts e até 28 amperes, contudo utilizar a potência máxima de carga que seria de 336 watts, vai causar super-aquecimento na bateria se não houver um circuito eletrônico adicional, que usualmente fica na bateria. Como o desenvolvimento desse circuito eletrônico está fora do escopo do projeto, o mais adequado é limitar a corrente de carga, e por isso em conversas com o Professor Fábio Delatore, docente do Centro Universitário FEI, foi aconselhável limitar a carga da bateria a 2 Amperes resultando em uma potência de 24 watts.

A Figura 26 abaixo mostra o Circuito de Carregamento da bateria feito no *software* Fusion360.

Figura 26 – Circuito da bateria

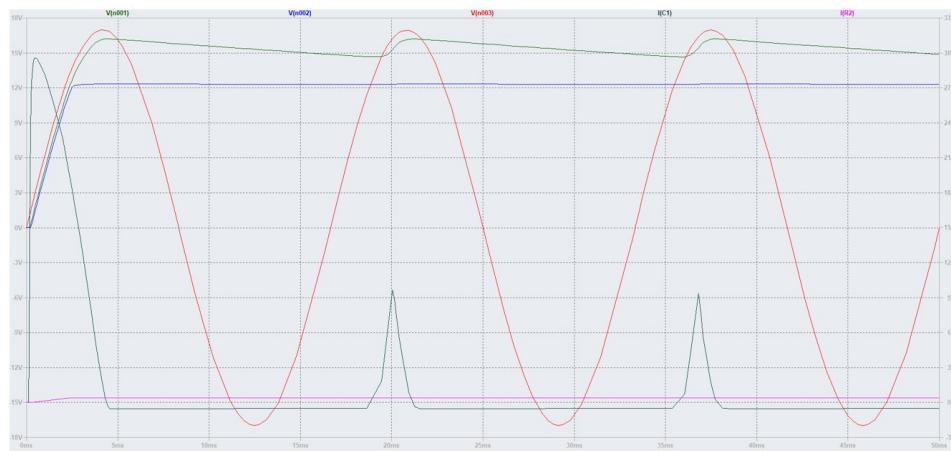


Fonte: Autores

A topologia escolhida para a fonte foi uma linear. A escolha por uma fonte linear ao invés de uma fonte chaveada vem do fato de que a fonte chaveada, embora mais eficiente, é mais complexa, aumentando o risco de falhas a tornando menos confiável. Fora isso, a fonte linear apresenta uma resposta mais rápida a variações na carga e a ausência de um circuito de chaveamento faz com que a fonte não gere ruído de alta frequência na rede elétrica. O dimensionamento de uma fonte chaveada com seu próprio circuito de chaveamento ficaria para um futuro estudo.

Para transformar os 127 volts de corrente alternada (Alternating current-AC) da rede elétrica em 12 volts de corrente contínua (Direct Current-DC) para carregar a bateria, o circuito pode ser dividido em 3 partes, redução de tensão, conversão de corrente contínua e nivelamento da corrente contínua, como na simulação da Figura 27.

Figura 27 – Simulação do circuito da bateria



Fonte: Autores

Para abaixar a tensão de 127 volts para 12 volts utiliza-se um transformador convencional. Poderia ser utilizado um transformador de tap central como mostra a imagem e como foi utilizado no trabalho final de eletrônica analógica, onde os autores utilizaram um transformador de tap central para reduzir a ponte de diodos em apenas 2, contudo, o transformador de tab central é mais caro, por isso o circuito foi desenhado para operar com um transformador convencional.

Para dimensionar o transformador, alguns passos são necessários. Em primeiro lugar, achar a relação de transformação conforme a equação abaixo:

$$r = V1/V2 = N1/N2 \quad (1)$$

Onde:

r= relação de transformação

V1= Tensão no primário

V2= Tensão no secundário

N1= Número de espiras no primário

N2= Número de espiras no Secundário

Uma vez tendo a relação de transformação, é possível determinar o número de espiras com base na equação:

$$N = V/4,44 * f * B * Ac \quad (2)$$

Onde: V = tensão do enrolamento (127 V para o primário e 12 V para o secundário); f = frequência (60 Hz para a rede elétrica brasileira); B = densidade de fluxo magnético (1,2 a 1,5 T é comum para aço-silício); Ac = área efetiva do núcleo (em m²).

Tendo as Espiras do primário e secundário, é necessário calcular as correstes para dimensionar os fios e como já temos a corrente no secundário, só precisamos determinar a corrente no primário e para isso usamos a potência no secundário, calculado pela fórmula abaixo:

$$P = V * I \quad (3)$$

A potência no enrolamento secundário é necessária para determinar a corrente no primário através da equação de conversão de potência dada por:

$$Ip = Ps/Vp \quad (4)$$

Uma vez tendo todos esses parâmetros é possível construir um transformador para essa fonte, porém, foi utilizado nesse projeto um transformador já pronto da própria universidade, isso nos concede uma margem de operação, uma vez que o transformador fornecido é mais robusto conseguindo suportar amperagens maiores tanto no primário quanto no secundário.

O transformador sozinho é responsável por converter a tensão AC de 127 volts em 12 volts ainda AC e para converter para DC temos o conjunto de 4 diodos D1 a D4. Esses diodos formam uma estrutura conhecida como ponte retificadora e ela funciona da seguinte forma: há dois terminais de entrada destinados a tensão DC e por consequência a corrente AC. Como o diodo é um componente semicondutor, suas características elétricas fazem com que o mesmo só conduza em uma direção, do ânodo (polo positivo) para o catodo (polo negativo). Por conta disso, quando a entrada AC está em seu semicírculo positivo a corrente flui através dos diodos D1 e D4, os diodos D2 e D3 estão bloqueados, impedindo o fluxo reverso. Uma vez que a entrada AC inverte sua polaridade, os Diodos D2 e D3 deixam de estar bloqueados e a corrente flui por eles e os diodos D1 e D4 estão bloqueados. Como resultado da presença da ponte de diodos temos uma corrente DC pulsante com mesma amplitude da corrente AC original.

Como mencionado, a resultante da ponte retificadora de diodos é uma tensão e corrente DC pulsante, a fim de resolver isso é que os capacitores de 4.700uF e o de 100nF estão presentes no circuito. O capacitor de 4.700uF serve como um filtro que diminui a ondulação da corrente e

da tensão, conforme a tensão aumenta, o capacitor vai carregando e quando a tensão começa a cair o capacitor passa a descarregar. Esse fenômeno faz com que a ondulação da tensão (Vripple) diminua deixando a tensão DC mais linear, contudo, esse capacitor não consegue fazer toda a filtragem sozinho. Para completar a filtragem o capacitor de 100nF faz praticamente a mesma função, só que para as ondulações de alta frequência, resultando em uma corrente contínua com uma ondulação muito baixa, chegando a ser desprezível.

Após os dois capacitores temos uma corrente contínua com uma ondulação muito baixa, contudo, devido ao fato de estarmos lidando com uma corrente e tensão contínua que foram originadas de uma tensão alternada de 12 volts eficazes, a tensão contínua resultante deve obedecer a seguinte formula:

$$V_{pico} = V_{eficaz} * \sqrt{2} \quad (5)$$

A VPico equivale a tensão resultante DC que temos e pelo fato de estarmos lidando com uma bateria de íons de lítio com uma densidade energética elevada, deve ser respeitado com rigor os dados técnicos fornecidos pela fabricante. Como temos um conjunto de 3 células em série e cada célula tem 4,2 volts, a bateria nos fornece uma tensão de 12,7 volts e essa é a tensão máxima de carregamento que pode ser utilizada, seguindo a equação dada temos uma tensão de pico de 16,97 volts.

Para controlar a tensão e a corrente que chega na bateria temos o último trecho do circuito que é o regulador simples. Esse regulador usa um diodo Zener em série com um resistor. O resistor limita a corrente que chega no diodo Zener e a corrente de base do transistor que controla o fluxo de corrente, sendo assim, para dimensionar esse resistor precisamos da corrente de base do transistor e a corrente mínima do diodo Zener, sendo assim os cálculos ficam da seguinte forma:

Cálculo da corrente de base IB.

$$I_b = I_c / (\beta) \quad (6)$$

Onde:

I_C =Corrente de coletor

β =Ganho de corrente

Como é preferível que a corrente da fonte seja de 2 Amperes, a corrente I_c também será de 2 Amperes, caso em que termos o I_B igual a 200 mA. A corrente que passa pelo diodo Zener

I_Z depende das especificações, segundo o datasheet do 1N4742AF esse diodo Zener pode ter de 1 mA a 76 mA de corrente, para esse projeto usaremos 10 mA, sendo assim, teremos a corrente do resistor igual a:

$$I_r = I_z + I_b = 10mA + 200mA = 210mA \quad (7)$$

A tensão que cai sobre o resistor é a diferença entre seus dois terminais, sabendo que a tensão de entrada é 16,97V e a tensão do Zener é de 12V a diferença de tensão será:

$$V_r = 16,97V - 12 = 4,97 \quad (8)$$

Tendo a corrente que passa pelo resistor e a tensão que cai sobre ele é possível determinar a resistência com base na lei de Ohm:

$$R = V_r/I_r = 4,97V/0,210A = 23,67V \quad (9)$$

Com estes parâmetros podemos determinar que o valor da resistência R é de 24 ohms, esse não é um simples arredondamento e sim o valor de um resistor comercial.

Como explicado anteriormente, o diodo Zener mantem a tensão da carga em 12 volts e o transistor controla o fluxo de corrente, contudo, há uma complicação. O transistor possui uma queda de tensão entre a base e o emissor, essa tensão VBE é subtraída da tensão do Zener e possui um valor típico de 0,7 volts, isso resultaria em uma tensão de 11,3 volts e para corrigir isso um novo diodo convencional é inserido em série com o diodo Zener, esse quinto diodo do circuito é utilizado para somar a sua tensão (0,7 V) a tensão do zener e assim a tensão resultante disponível para carregar a bateria são os exatos 12 volts DC com uma ondulação muito baixa.

Utilizando o software LTspice para simular o circuito temos o seguinte resultado:

Com a simulação no software LTspice é possível ver todo o comportamento do circuito. A linha vermelha representa a tensão alternada, pós redução de tensão, a linha verde é a tensão contínua pós os capacitores com um VRipple elevado de cerca de 1,5 volts. A linha azul escura é a tensão pós o regulador de tensão simples mantendo a tensão contínua em 12 volts. A linha azul clara é a corrente do capacitor que começa alta para carregar e depois normaliza, a linha rosa é a corrente de carga que para essa simulação foi limitada pela carga de teste.

Há alguns comportamentos perigosos que devem ser limitados para evitar problemas. O primeiro ponto é limitar a corrente que o capacitor irá demandar da rede. Quando está descarregado ele é equivalente à um curto-circuito, isso demanda uma corrente muito alta. Várias

soluções podem ser feitas para corrigir isso, entre elas colocar um resistor em serie com a entrada; colocar um termistor NTC, entre outras soluções, contudo, para determinar qual resistência iremos usar para limitar a corrente, precisamos saber o limite dos diodos da ponte retificadora composta pelos diodos 1N5408 que tem corrente de pico de 200 amperes, a fim de manter uma boa margem de segurança, adotaremos essa corrente como 100 amperes.

Dado que a corrente que passa pelos diodos não pode ultrapassar os 100 amperes, podemos calcular a resistência com base na lei de Ohm:

$$V = R * I \quad (10)$$

$$R = 12/100 = 0,12 \quad (11)$$

Essa resistência é a mínima que deve ser colocada em serie com a ponte de diodos para garantir que não sejam queimados no processo de partida da fonte, porém, o fio de cobre já possui uma resistência interna calculada por:

$$R = \rho * L/A \quad (12)$$

Onde:

- ρ = Resistividade elétrica do material (cobre aproximadamente $1,68 \times 10^{-8}$ ohns*m)
- L = Comprimento do condutor
- A = área de sessão transversal

Dado que o condutor tem um $A = 0,5 \text{ mm}^2$ e assumindo um L de 1 cm, temos uma resistência interna do fio de $0,366 \Omega$ aproximadamente. Concluímos assim que o fio de cobre já limita a corrente do capacitor sem danificar os diodos. Esse comportamento também é visível na simulação do LTspice onde a corrente de partida do diodo é de 30 amperes aproximadamente.

Outro ponto de atenção é a saída do circuito, a tensão é fixa em 12 volts, mas a corrente não é limitada, se a carga demandar mais corrente vai haver queda de tensão, para mitigar isso temos o circuito com o LM338 com um resistor externo de $0,625 \Omega$ dado que a corrente de saída do LM338 é dada pela equação:

$$I_{out} = 1,25/R \quad (13)$$

Sabendo que a corrente de saída deve ser de 2 amperes, podemos afirmar que a resistência necessária é de $0,625 \Omega$. Essa resistência não pode ser resolvida com um fio como feito

anteriormente, pois precisaríamos de um condutor de mais de 18 metros, portanto, o resistor fixo será melhor e pode ser feito de duas formas: com o próprio resistor de $0,625\ \Omega$, que é mais difícil de encontrar ou com resistores em paralelo como por exemplo 2 de $1,2\ \Omega$ por exemplo. Pegando o trabalho utilizado para estudo dessa topologia, os autores optaram por outra solução para limitar a corrente, uma fonte de corrente feita com um segundo transistor NPN. Seria possível limitar a correm por meio de um resistor em serie com a saída, mas essa solução causaria muita dissipação de potência em forma de calor. Tendo em vista que a topologia escolhida não é a mais eficiente, a solução com, apenas o resistor não é viável; a formulação de uma fonte de corrente com um segundo transistor é mais complexa que o circuito com o LM338, sendo esse componente o mais adequado para esse trabalho.

4.3.7 Testes de Métodos de Elementos Finitos (FEM)

Para assegurar a segurança e resistência mecânica de todas as partes estruturais do nosso robô, examinamos as propriedades dos seus materiais por meio de uma simulação de FEM (Método dos Elementos Finitos ou *Finite Element Method*), uma técnica simples e prática que permite visualizar como as peças se deformam quando uma força é aplicada sobre elas.

- **FEM (*Finite Element Method*):** Esta análise foi feita diretamente no *software NX*, usando um complemento chamado *Nastran*. Com ele, podemos usar qualquer arquivo tridimensional e separá-lo em partes menores conhecidas como elementos finitos. Quando aplicada uma força nesse modelo de elementos finitos, podemos obter várias informações, tais como tensões, deformações e deslocamentos. Também podemos realizar análises térmicas, acústicas ou eletromagnéticas, porém, essas três últimas não são de nosso interesse.

- **Arquivos 3D's:** Conforme planejado anteriormente, nossos desenhos 3D das peças não foram utilizados apenas para a impressão, mas também foram reutilizados na simulação. Se algum resultado não fosse adequado para assegurar a resistência estrutural dos nossos componentes, podemos fazer as modificações nos arquivos já utilizados para as impressões.

- **Coeficiente de Segurança:** Dentro dos vários campos da engenharia, todas utilizam-se de coeficientes de seguranças, no qual esses coeficientes são utilizados para assegurar estruturas, componentes ou afins contra falhas devido a incertezas, como por exemplo a dos materiais empregados que variam de projeto para projeto.

Quanto maior a incerteza do material ou importância do componente, por exemplo, mais elevado pode ser o seu coeficiente de segurança, que representa uma força extra aplicada à força à qual a peça ou sistema será submetido, de acordo com o projeto.

Como utilizamos materiais que já estamos acostumados a usar, como ABS e PLA, amplamente empregados em impressões 3D, juntamente com camadas de 5mm de espessura, a probabilidade de algum componente se partir devido ao peso é mínima, já que não estamos lidando com cargas pesadas, limitando-nos a 3kg em circunstâncias críticas. Tendo isso em vista, foi empregado um coeficiente de apenas 1,2 (20% sobre a carga definida), e testamos qualquer peça que atinja no máximo 2 milímetros de deformação, já que a partir deste ponto pode afetar a leitura dos sensores.

5 RESULTADOS

Neste capítulo estão presentes todos os resultados das métricas definidas na Seção 4.3.

5.1 SIMULAÇÃO

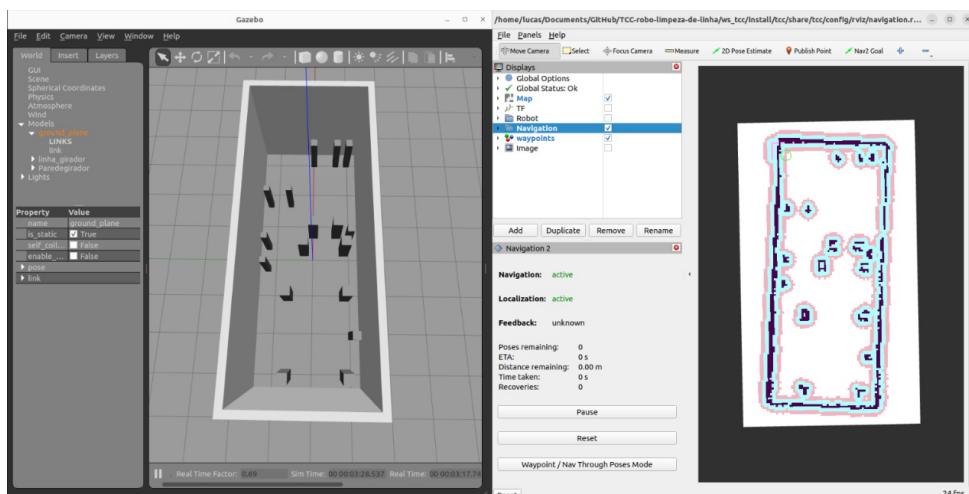
Esta parte demonstra a eficácia da simulação e dos métodos de mapeamento e localização, estabelece a combinação mais eficaz de planejador local e global, e o tipo de percurso mais adequado, além de determinar o tempo médio de limpeza e a efetividade da navegação.

Um repositório no *GitHub*¹ foi criado onde há todos os arquivos de programação desenvolvidos e utilizados durante a realização do projeto. Também criamos uma playlist não listada no *YouTube*² contendo vídeos com a realização do mapeamento e alguns dos 64 testes realizados durante a obtenção dos resultados encontrados nas Seções 5.1.3, 5.1.4 e 5.1.5.

5.1.1 Mapeamento

Posicionamos o robô dentro do mundo criado com base na área do estudo de caso. O mapa é gerado parcialmente de acordo com as detecções de paredes e obstáculos realizadas pelo sensor LiDAR, para obter o mapa completo, movimentamos o robô pela arena utilizando comandos via teclado. Uma vez percorrido o suficiente da área para o registro do local completo, finalizamos o processo de mapeamento e um mapa completo como visto na Figura 28 é gerado.

Figura 28 – Mundo simulado ao lado do mapa gerado



Fonte: Autores

¹*GitHub*: <https://github.com/LuAgostinho/TCC-robo-limpeza-de-linha>

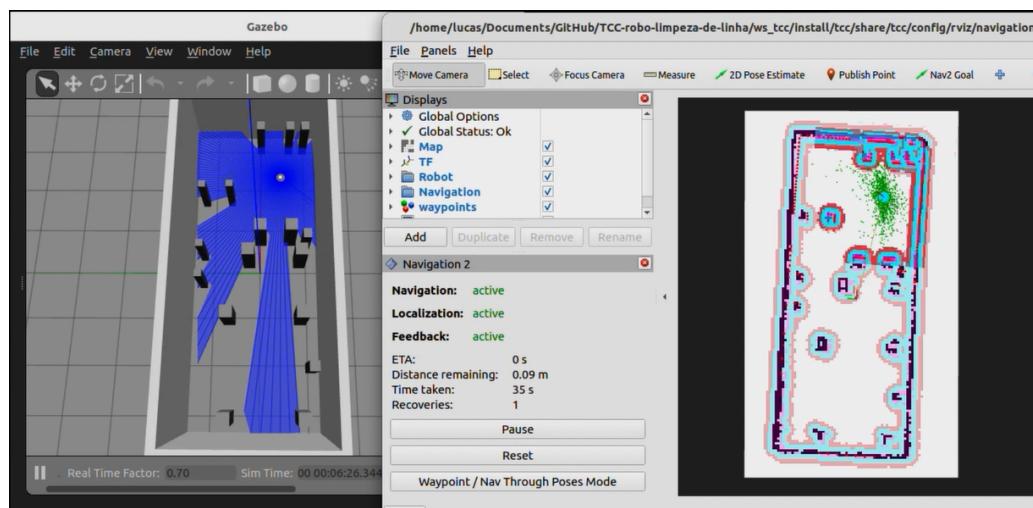
²*YouTube*: <https://www.youtube.com/playlist?list=PLjjUCjqIRs7aSMpOdP-UQYknQHS8CXove>

5.1.2 Localização

Com o mapeamento realizado, podemos testar a eficiência do método de localização. Com o robô dentro do local, podemos movimentá-lo livremente, enquanto a simulação do robô ocorre no Gazebo, observamos o RVIZ, onde é mostrado o mapa do local, o robô representado por um grande círculo azul claro e as partículas de Monte Carlo destacadas em verde.

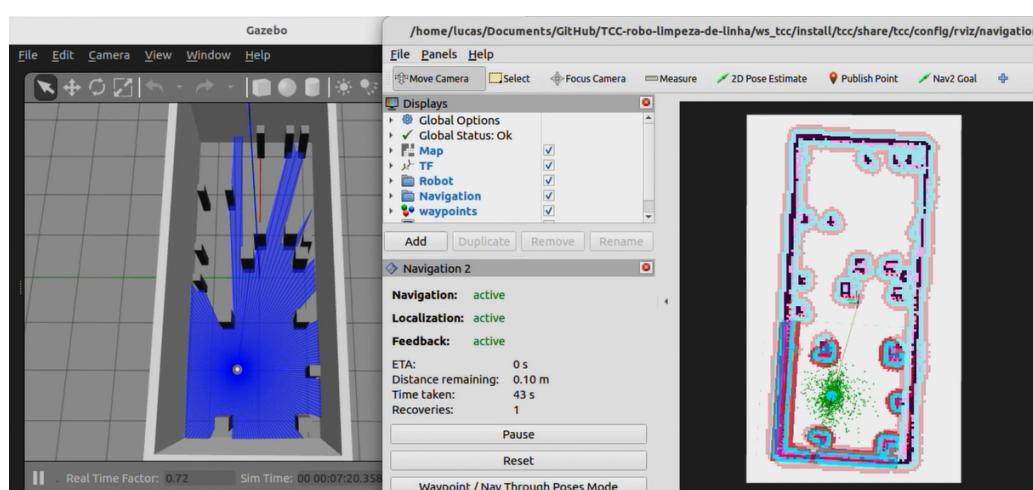
Para a localização estar representada de forma correta, o local do robô no Gazebo e no RVIZ obrigatoriamente devem ser o mesmo e as partículas de Monte Carlo têm de se concentrarem no local onde o robô se encontra, assim como mostrado na Figuras 29, 30 e 31.

Figura 29 – Exemplo 1 de funcionamento do método de localização



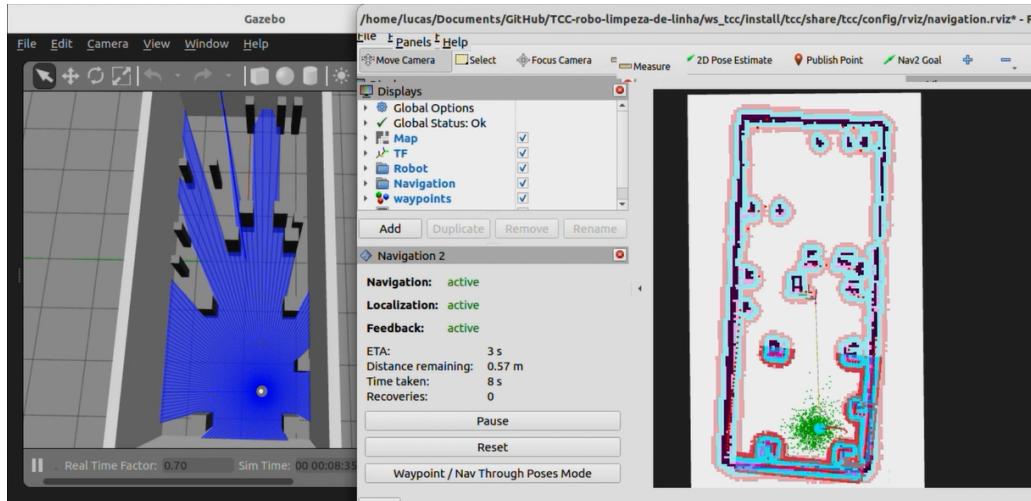
Fonte: Autores

Figura 30 – Exemplo 2 de funcionamento do método de localização



Fonte: Autores

Figura 31 – Exemplo 3 de funcionamento do método de localização



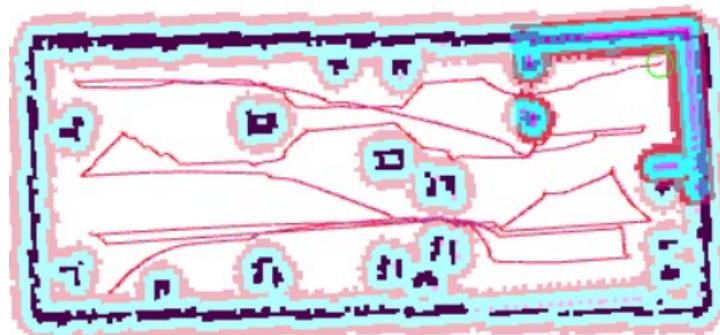
Fonte: Autores

5.1.3 Definição do planejador global e planejador local

Para determinar a combinação ideal de planejador global e local, realizamos experimentos com cada uma das seis combinações possíveis entre os planejadores que escolhemos, sendo dois globais (A* expandido e Algoritmo de Dijkstra) e três locais (DWB, Vector Pursuit e Regulated Pure Pursuit). A trajetória poderia ser realizada de quatro formas diferentes:

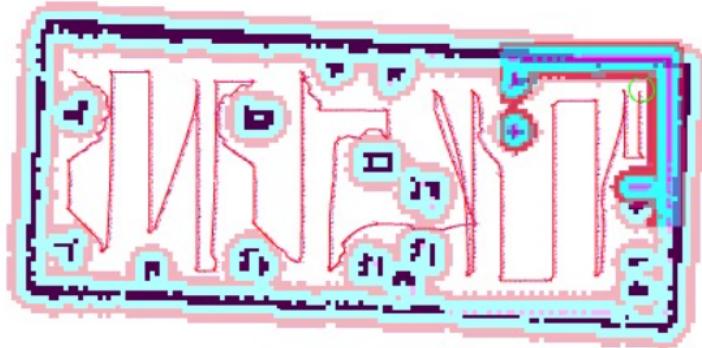
- **Movimentação aleatória:** Esta alternativa é exatamente o oposto do que o projeto propõe, porém realizamos alguns testes para fins de comparação.
- **Entre paredes:** Rejeitada por haver tanto espaços abertos sem contato com paredes quanto espaços estreitos com vários obstáculos.
- **Zig-Zag:** Testada em duas direções:
 - a) Zig-Zag horizontal: Retas longas e poucas curvas, exemplificado na Figura 32;
 - b) Zig-Zag vertical: Retas curtas e muitas curvas, como na Figura 33.

Figura 32 – Exemplo de trajetória gerada por planejador em zig-zag horizontal



Fonte: Autores

Figura 33 – Exemplo de trajetória gerada por planejador em zig-zag vertical



Fonte: Autores

- **Espiral ou contorno:** Considerando que o formato da área a ser percorrida é retangular, a trajetória espiral circular foi desconsiderada. Porém a espiral com forma retangular foi testada, conforme ilustrado na Figura 34, e os resultados podem ser conferidos a partir da Tabela 1.

Figura 34 – Exemplo de trajetória gerada por planejador em espiral retangular



Fonte: Autores

No total, nove testes foram conduzidos para cada combinação de planejador global e local, sendo três para cada tipo de percurso. Nas Tabelas 1 a 6 encontram-se as médias obtidas dos resultados desses três testes de cada tipo de trajetória para todas as combinações de planejadores. O objetivo desta etapa era obter a combinação mais eficaz de tempo de execução e taxa de efetividade, independentemente do tipo de percurso.

A taxa de efetividade é calculada através da relação entre o tempo total da simulação e o tempo em que o robô permaneceu fora do percurso definido. Para obter o tempo total em que o robô permaneceu fora do percurso, cronometramos os momentos em que o robô se distanciou da rota planejada, ficou imóvel por longos períodos ou entrou em modo de *recovery* (recuperação) constantemente sem um avanço significativo, depois os valores obtidos foram somados e multiplicados pelo fator da simulação, fator esse que indica quantas vezes a mais ou a

menos a simulação está rodando comparada com o tempo real. Os melhores resultados de tempo de execução e taxa de efetividade para cada trajeto estão destacados em negrito.

Tabela 1 – Resultados obtidos utilizando A* expandido e DWB

Trajetória	Tempo de execução (minutos)	Tempo no percurso (minutos)	Tempo fora do percurso (minutos)	Efetividade (%)
Horizontal	14:58	9:57	5:01	66,48
Vertical	19:04	12:44	6:20	66,78
Contorno	20:07	9:14	10:53	45,89

A combinação A* expandido e DWB apresentou valores de tempo de execução consideravelmente altos, além de uma efetividade baixa, principalmente no trajeto de contorno, como demonstrado na Tabela 1.

Tabela 2 – Resultados obtidos utilizando A* expandido e Vector Pursuit

Trajetória	Tempo de execução (minutos)	Tempo no percurso (minutos)	Tempo fora do percurso (minutos)	Efetividade (%)
Horizontal	11:07	9:25	1:42	84,71
Vertical	9:42	7:47	1:55	80,24
Contorno	8:16	7:28	0:48	90,32

Na Tabela 2 encontramos a combinação de A* expandido e VP, podemos observar valores promissores de tempo de execução e uma alta taxa de efetividade, o tipo de combinação de resultados que desejávamos. Também podemos observar uma inversão quanto ao resultado da efetividade no trajeto de contorno em comparação com os testes realizados com DWB, sendo o maior percentual de precisão entre os trajetos.

Tabela 3 – Resultados obtidos utilizando A* expandido e Regulated Pure Pursuit

Trajetória	Tempo de execução (minutos)	Tempo no percurso (minutos)	Tempo fora do percurso (minutos)	Efetividade (%)
Horizontal	19:35	14:23	5:12	73,45
Vertical	35:25	15:21	10:04	71,58
Contorno	20:06	15:00	5:06	74,63

A Tabela 3 apresenta a combinação de planejadores com os piores resultados de tempo de execução, sendo ela A* expandido com RPP.

Voltando para os resultados com DWB porém utilizando o Algoritmo de Dijkstra, na Tabela 4, obtivemos um tempo de execução consideravelmente mais baixo, assim como uma efetividade maior em relação ao A* expandido, porém ainda apresentou problemas quanto ao trajeto de contorno.

Tabela 4 – Resultados obtidos utilizando Algoritmo de Dijkstra e DWB

Trajetória	Tempo de execução (minutos)	Tempo no percurso (minutos)	Tempo fora do percurso (minutos)	Efetividade (%)
Horizontal	10:57	7:59	2:58	72,89
Vertical	16:46	13:10	3:36	78,53
Contorno	18:21	10:37	7:44	57,86

Tabela 5 – Resultados obtidos utilizando Algoritmo de Dijkstra e Vector Pursuit

Trajetória	Tempo de execução (minutos)	Tempo no percurso (minutos)	Tempo fora do percurso (minutos)	Efetividade (%)
Horizontal	9:09	7:45	1:24	84,70
Vertical	8:13	6:57	1:16	84,58
Contorno	7:12	6:32	0:40	90,74

A utilização do VP com o algoritmo de Dijkstra, encontrada na Tabela 5, resultou em uma efetividade similar para os trajetos horizontal e contorno, e uma pequena melhora no trajeto vertical. A grande mudança em relação a utilização do A* expandido foi no tempo de execução, onde houve uma redução de ao menos um minuto nos três tipos de trajeto.

Tabela 6 – Resultados obtidos utilizando Algoritmo de Dijkstra e Regulated Pure Pursuit

Trajetória	Tempo de execução (minutos)	Tempo no percurso (minutos)	Tempo fora do percurso (minutos)	Efetividade (%)
Horizontal	18:37	12:39	5:58	67,95
Vertical	30:40	25:53	4:47	84,40
Contorno	19:07	14:51	4:16	77,68

Na Tabela 6, observamos a combinação do algoritmo de Dijkstra com RPP, nela elevados tempos de execução voltam a aparecer, sendo o segundo pior grupo de resultados.

5.1.3.1 Definição do planejador global

O planejador global A* expandido tem como característica realizar uma maior proximidade em relação a obstáculos e curvas mais acentuadas em seu trajeto, como nosso objetivo era realizar a limpeza na maior área possível isso seria benéfico. Porém, essa proximidade ocasionou um grande número de travamentos durante sua execução, interferindo assim no tempo de execução e na taxa de efetividade.

Esse fato resultou no algoritmo de Dijkstra como melhor planejador global, com um tempo de execução menor para os três planejadores locais e em todos os trajetos em relação ao A* expandido, além de uma taxa de efetividade similar ou maior em todas as situações, exceto utilizando o planejador local Regulated Pure Pursuit na trajetória de zig-zag horizontal.

5.1.3.2 Definição do planejador local

Focando nos resultados dos planejadores locais e os separando quanto aos dois planejadores globais utilizados, observamos resultados semelhantes durante as comparações.

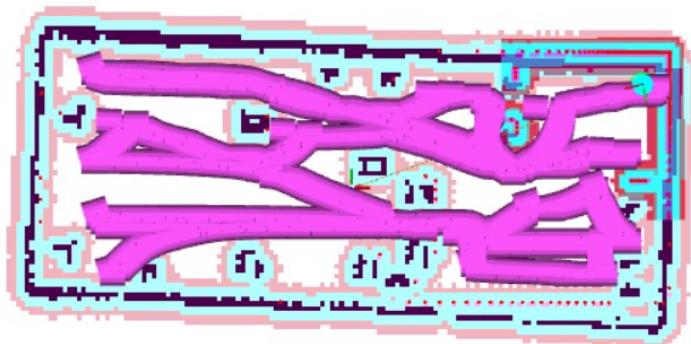
Em ambas as situações, o planejador local DWB teve um tempo de execução moderado e uma taxa de efetividade baixa, enquanto o planejador RPP obteve um resultado oposto, com um tempo elevado superando até mesmo a marca de 30 minutos e uma efetividade moderada. Enquanto isso, o planejador VP atingiu o menor tempo de execução e as melhores taxas de efetividade em todos os trajetos testados, sendo assim o melhor planejador local.

5.1.4 Definição do melhor tipo de trajeto

Com a combinação ideal de planejador global e local estabelecida, sendo ela Algoritmo de Dijkstra e Vector Pursuit respectivamente, nosso objetivo era conduzir mais experimentos para os três tipos de percursos, com o objetivo de determinar a forma de deslocamento mais eficaz para o robô, levando em conta a disposição da área a ser percorrida.

No entanto, durante a primeira etapa de testes, notamos um problema na área de cobertura do robô durante a trajetória em zig-zag na direção horizontal. Na Figura 35 está representado a área coberta pelo robô nesse trajeto, sendo destacado em rosa toda a área onde ocorreria a limpeza, já incluindo o alcance extra fornecido pelas pás laterais. É possível perceber vastas regiões que não foram percorridas devido à extensa distância entre os pontos estabelecidos. Tentamos incluir alguns pontos intermediários para o robô, mas isso resultou em outros problemas, como a frequente colisão com obstáculos próximos a alguns desses novos pontos e o travamento do robô devido à dificuldade em seguir o trajeto definido.

Figura 35 – Exemplo de área coberta durante o trajeto de zig-zag horizontal



Fonte: Autores

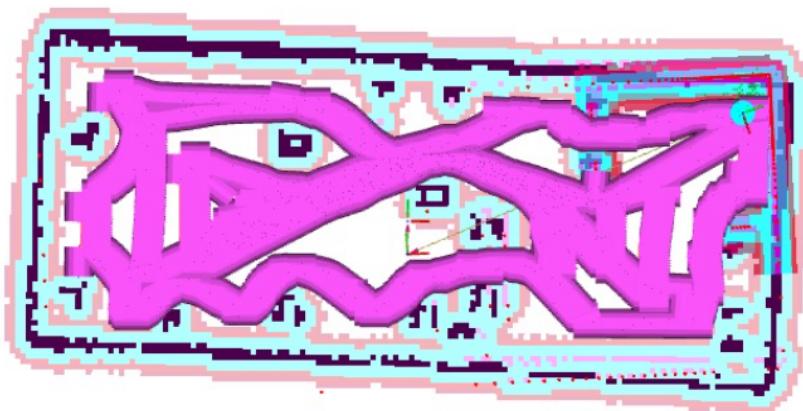
Felizmente, essa questão não se manifestou na trajetória de zig-zag vertical nem na trajetória em espiral retangular, conforme ilustrado respectivamente nas Figuras 36 e 37.

Figura 36 – Exemplo de área coberta durante o trajeto de zig-zag vertical



Fonte: Autores

Figura 37 – Exemplo de área coberta durante o trajeto de contorno



Fonte: Autores

Considerando que estávamos a procura do trajeto mais eficiente, optamos por prosseguir com os testes somente nos dois percursos onde há uma grande cobertura de área. Portanto, ao reutilizar os dois grupos de testes iniciais que originaram as médias mostradas na Tabela 5, dobramos o número de testes executados para alcançar um resultado mais consistente para cada modelo de trajetória. Essas informações podem ser encontradas nas Tabelas 7 e 8.

Identificamos a melhor rota como sendo o zig-zag vertical. Contudo, isso não ocorreu por causa do tempo de execução ou taxa de efetividade, já que os valores obtidos por ambos são bastante similares, mas sim à consistência na movimentação do robô notada nos testes, esse fato é explicado melhor no próximo parágrafo.

Tabela 7 – Resultados obtidos com trajetória zig-zag vertical

Nº do teste	Tempo de execução (minutos)	Tempo no percurso (minutos)	Tempo fora do percurso (minutos)	Efetividade (%)
Teste 1	8:34	6:54	1:40	80,55
Teste 2	7:55	7:11	0:44	90,74
Teste 3	8:10	6:43	1:27	82,25
Teste 4	7:31	6:53	0:38	91,57
Teste 5	7:42	6:16	1:26	81,38
Teste 6	7:20	6:09	1:11	83,86

Tabela 8 – Resultados obtidos com trajetória espiral retangular

Nº do teste	Tempo de execução (minutos)	Tempo no percurso (minutos)	Tempo fora do percurso (minutos)	Efetividade (%)
Teste 1	7:05	6:42	0:23	94,59
Teste 2	7:27	6:32	0:55	87,91
Teste 3	7:04	6:21	0:43	89,86
Teste 4	7:21	6:18	1:03	85,71
Teste 5	8:42	6:44	1:58	77,39
Teste 6	7:55	6:51	1:04	86,53

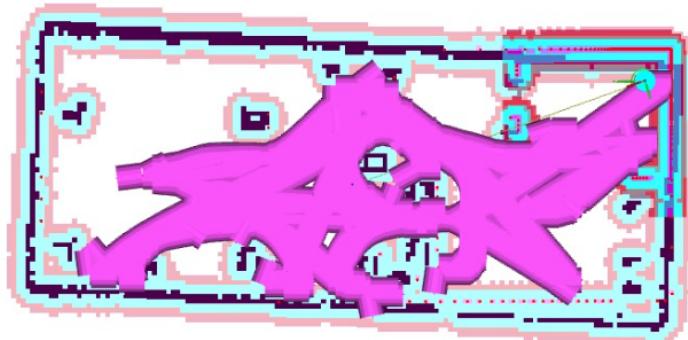
Esta nova série de testes revelou um problema específico da trajetória de contorno que não foi tão frequente nos três primeiros testes, uma redução acentuada da velocidade ou, em casos raros, uma parada total por alguns segundos ao atravessar áreas estreitas. Isso é bastante comum nessa trajetória. Por outro lado, o percurso vertical em zig-zag não possui tantas passagens por áreas estreitas, garantindo uma repetibilidade superior com maior confiabilidade, caso esse que atrelamos à consistência da rota citada anteriormente.

Queremos enfatizar que este teste sempre deve ser repetido quando a disposição do local, o tamanho ou a quantidade de obstáculos forem muito diferentes das situações previamente testadas.

Para fins de comparação foram feitos também testes com o algoritmo aleatório usando a melhor combinação dos planejadores em um código que gerava constantemente pontos aleatórios do mapa para o robô navegar, visto que esse trajeto não possui fim definido, limitamos o tempo dos testes para 30 minutos, sendo esse o atual tempo gasto pelos funcionários para a realização de uma limpeza, as áreas cobertas podem ser vistas nas Figuras 38, 39 e 40.

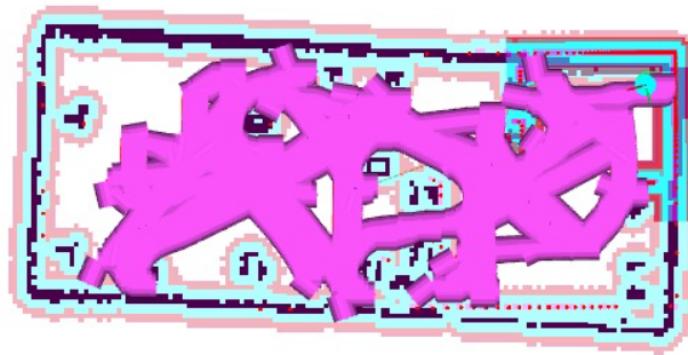
Percebe-se que a área coberta não é constante em nenhum dos 3 testes realizados, além de que as métricas utilizadas não se aplicam justamente por não ter um percurso pré-definido, não podendo assim existir um tempo fora do percurso.

Figura 38 – Exemplo de área coberta durante o trajeto aleatório



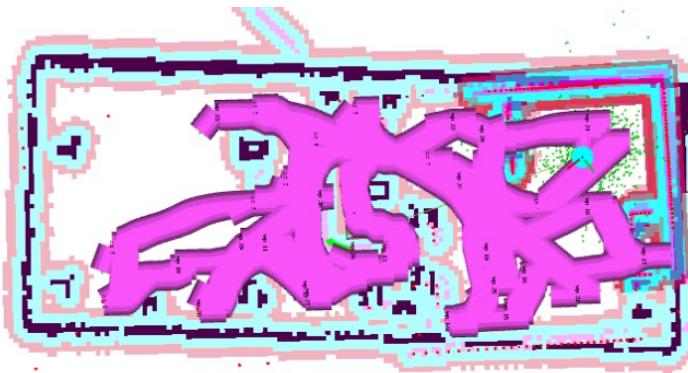
Fonte: Autores

Figura 39 – Exemplo de área coberta durante o trajeto aleatório



Fonte: Autores

Figura 40 – Exemplo de área coberta durante o trajeto aleatório



Fonte: Autores

Outra questão é a consistência dos testes, durante a execução deles o robô apresentou um número maior de travamentos e pontos onde os protocolos de recuperação precisaram agir, isso juntamente com o fato de alguns pontos serem gerados dentro de obstáculos tornou esse trajeto inviável, conforme havia sido comentado anteriormente.

5.1.5 Tempo médio de limpeza e taxa de efetividade da navegação

Estabelecido a melhor combinação entre os planejadores e a trajetória, executamos mais testes até chegarmos a um total de dez resultados, apresentados na Tabela 9. Posteriormente, foram empregados para determinar a média de tempo que o robô levou para executar o procedimento e a taxa de efetividade sobre o trajeto planejado e o realizado.

Tabela 9 – Resultados de testes para tempo médio de limpeza e efetividade

Nº do teste	Tempo de execução (minutos)	Tempo no percurso (minutos)	Tempo fora do percurso (minutos)	Efetividade (%)
Teste 1	8:34	6:54	1:40	80,55
Teste 2	7:55	7:11	0:44	90,74
Teste 3	8:10	6:43	1:27	82,25
Teste 4	7:31	6:53	0:38	91,57
Teste 5	7:42	6:16	1:26	81,38
Teste 6	7:20	6:09	1:11	83,86
Teste 7	10:21	9:18	1:03	89,86
Teste 8	7:54	6:44	1:10	85,23
Teste 9	6:59	6:12	0:47	88,78
Teste 10	7:11	6:30	0:41	90,49
Média:	7:58	6:53	1:05	86,47

Obtemos um tempo médio de execução de 7 minutos e 58 segundos com uma efetividade média de 86,47%, com isso possuímos os dados necessários para realizar a verificação da eficiência dos códigos de localização e navegação, e posteriormente a comparação com processo atual de limpeza.

5.2 ELÉTRICA

Esse subcapítulo explica os resultados elétricos encontrados após a realização das métricas abordadas no capítulo anterior.

5.2.1 Circuito elétrico

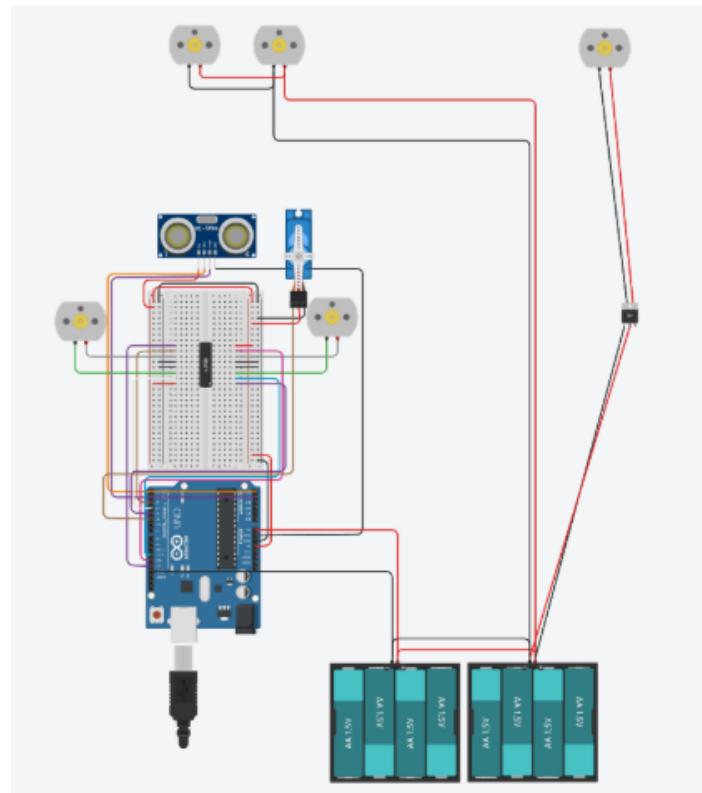
O desenvolvimento e a integração dos componentes do robô aspirador foram validados por meio de uma série de testes realizados para avaliar o desempenho de cada parte do sistema. Os motores de tração, controlados pela ponte H, demonstraram funcionamento adequado, permitindo que o robô se movesse para frente e para os lados, sem falhas no controle de direção ou aquecimento excessivo. Os motores de pás, responsáveis pela movimentação das escovas de

limpeza, foram acionados de forma eficiente através do botão de liga / desliga, operando em 12V, e apresentaram boa performance, com rotação estável e sem sobrecarga.

O motor aspirador, que requer uma alimentação de 8V, foi integrado com sucesso ao sistema por meio de um regulador de tensão, funcionando de maneira contínua e sem variações de potência, garantindo a eficácia na aspiração. Após a validação individual dos componentes, foi realizada a integração do sistema, onde o Arduino Uno controlou de forma coordenada os motores de tração e o sensor ultrassom. O robô demonstrou bom desempenho geral, com os motores operando de forma sincronizada e o sensor ultrassom detectando obstáculos com uma distância menor que 15 centímetros e desviando o robô para a direita ou esquerda, ajustando sua trajetória rapidamente, em menos de 2 segundos.

O primeiro teste consistiu em adicionar o circuito ao Tinkercad, nele foram adicionados 8 pilhas de 1,5V para simular a bateria conectadas diretamente ao Arduino Uno, e aos motores de pá e aspirador, o Arduino Uno liga-se diretamente ao L293D (No programa não havia disponível o Shield da ponte h, então os componentes foram conectados a um protobord). Nesse primeiro teste foram testados o funcionamento dos componentes (Figura 41).

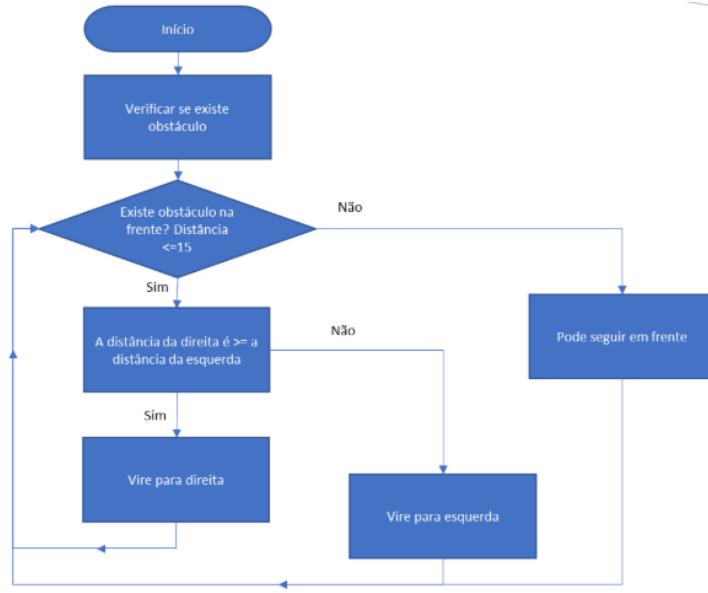
Figura 41 – Circuito montado no Tinkercad



Fonte: Autores

O código do circuito do primeiro teste possui o fluxograma da figura 42, onde o robô anda para frente até se deparar com um obstáculo, após isso ele da ré de aproximadamente 5cm e analisa sua direita e esquerda para saber para qual direção ele deve virar, caso ambos os lados estiverem livres o robo virará para a direita.

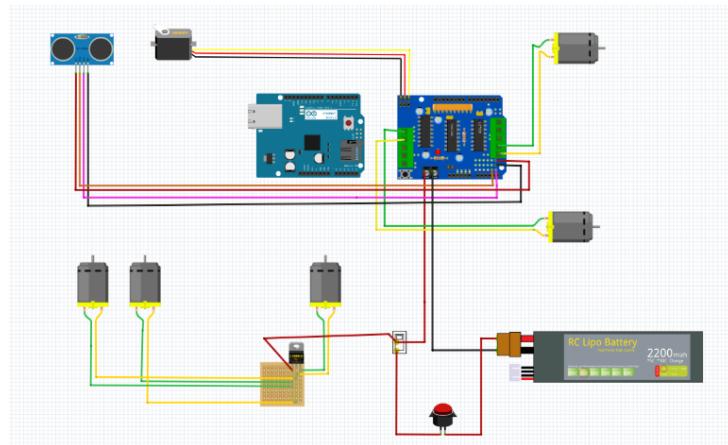
Figura 42 – Fluxograma do código usado nos testes com o sensor Ultrassom



Fonte: Autores

No segundo teste o circuito foi adicionado ao Fritzing onde os componentes foram ligados na configuração final já adicionando o botão de emergencia e o botão liga e desliga.

Figura 43 – Circuito eletrico no Fritzing



Fonte: Autores

O ultimo teste realizado foi em uma sala de 5x10m, que possuía móveis domésticos comuns, como sofá, mesa de jantar e cadeiras. Durante o teste, o sensor ultrassom desempenhou um papel crucial na navegação do robô, sendo responsável por identificar obstáculos à frente

e permitir que o robô desviasse de maneira eficiente. Quando o sensor detectava um objeto a uma distância menor que 15 centímetros, o robô automaticamente ajustava sua trajetória, desviando para a direita ou esquerda para evitar colisões. A resposta do sistema foi precisa e rápida, com o robô alterando sua direção em menos de 2 segundos após a detecção do obstáculo. Esse comportamento foi observado de maneira consistente ao longo de todo o teste, com o robô conseguindo navegar ao redor dos móveis sem muitas dificuldades como mostrado na sequencia de imagens entre as Figuras 44 a 49. (Video de teste encontrado no link: <https://youtu.be/cFOoXvu2HqY>)

Figura 44 – Robô identificando um obstáculo



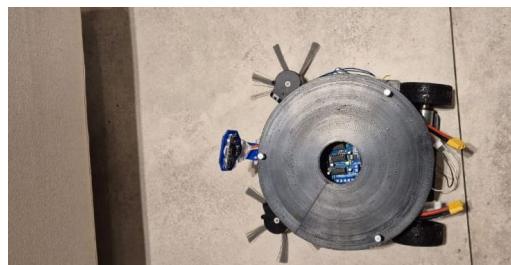
Fonte: Autores

Figura 45 – Sensor virando 25 graus a direita



Fonte: Autores

Figura 46 – Sensor voltando ao seu eixo inicial



Fonte: Autores

Figura 47 – Sensor virando 25 graus a esquerda



Fonte: Autores

Figura 48 – Robô realizando o desvio para a direita



Fonte: Autores

Figura 49 – Robô se movimentando para frente após o desvio



Fonte: Autores

A navegação foi fluida, e o robô cobriu toda a área da sala de forma eficiente, evitando obstáculos e realizando a limpeza. A autonomia do robô foi validada com uma operação contínua de aproximadamente 40 minutos, o que foi suficiente para a limpeza completa da sala.

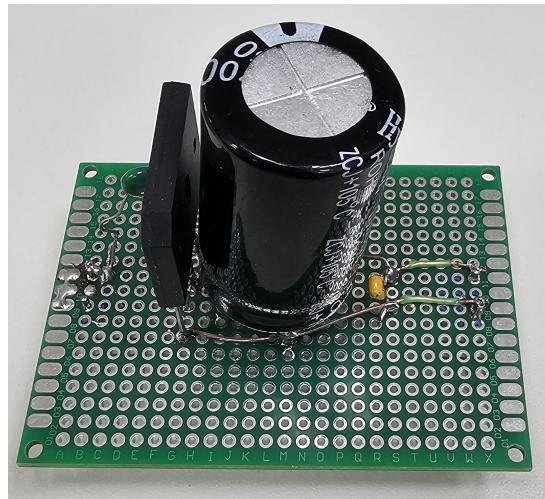
Durante os testes de integração do sistema, também foi realizada a tentativa de implementar o sistema de mapeamento simulado no GAZEBO, porém, não obtivemos sucesso em sua integração com o restante dos componentes eletrônicos. Diante dessa limitação, optou-se por continuar o funcionamento do robô com a utilização do sensor ultrassom, que garantiu uma navegação eficiente e a execução da limpeza de forma autônoma. A equipe está ciente dos desafios apresentados pela implementação do sistema de mapeamento e está comprometida em

estudar as causas desses problemas, visando o desenvolvimento de um novo projeto que integre com sucesso essa funcionalidade em futuras versões do robô. Teste final encontrado no link: <https://youtu.be/53AlQyGaMqw>

5.2.2 Carregador da bateria

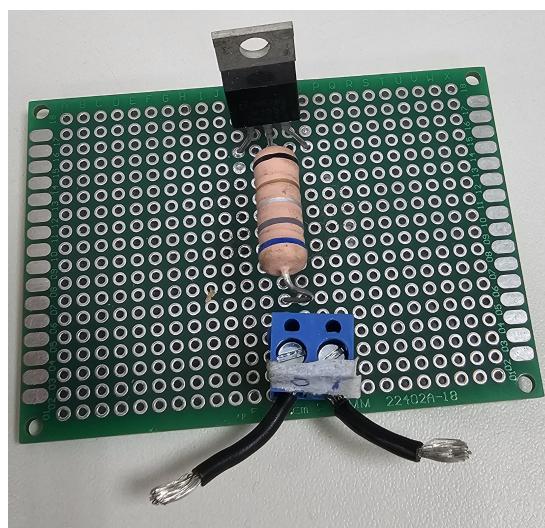
O carregador de bateria foi dividido em 3 partes sendo elas o conversor AC-DC, o limitador de corrente com o LM338 e por fim o Retificador de tensão com o transistor bipolar TIP31C representados nas Figuras 50, 51 e 52 respectivamente.

Figura 50 – Conversor AC-DC



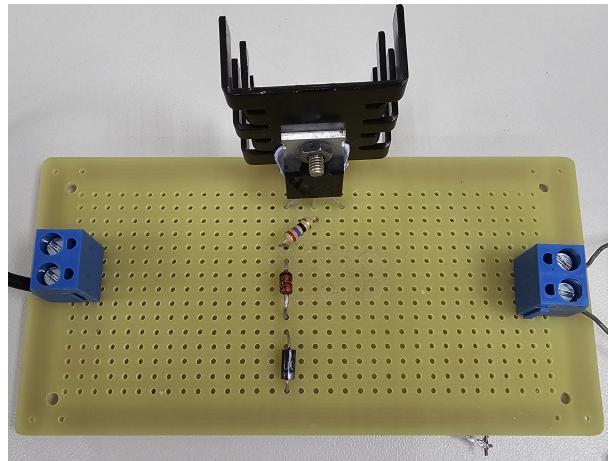
Fonte: Autores

Figura 51 – Limitador de corrente LM338



Fonte: Autores

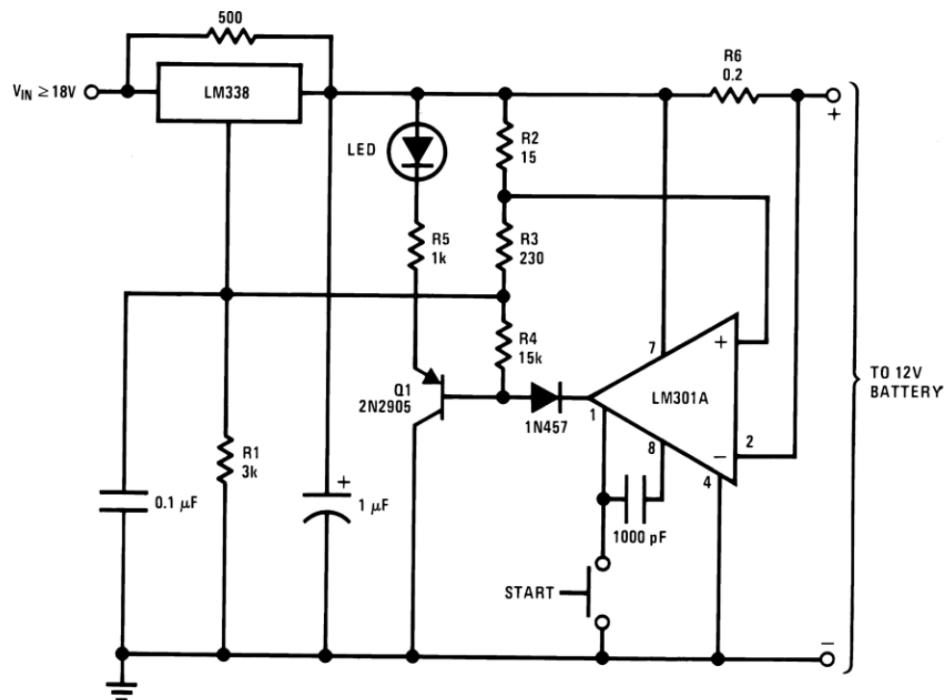
Figura 52 – Regulador de tensão



Fonte: Autores

Durante o teste da fonte de alimentação o circuito apresentou problemas após a conversão AC-DC, por isso foi feito uma análise adicional dos circuitos exemplos disponíveis no *datasheet* do LM338 (INSTRUMENTS, 2016), nele encontramos o circuito encontrado na Figura 53.

Figura 53 – Circuito de Carregamento de baterias de 12 volts



Copyright © 2016, Texas Instruments Incorporated

Fonte: (INSTRUMENTS, 2016)

Esse circuito usa o regulador LM338 para retificar a tensão de entrada para 12 volts, em adição a isso temos o Amplificador Operacional LM301A que serve como um comparador de tensão, uma vez que a carga da bateria chega ao fim o transistor Q1 interrompe a carga

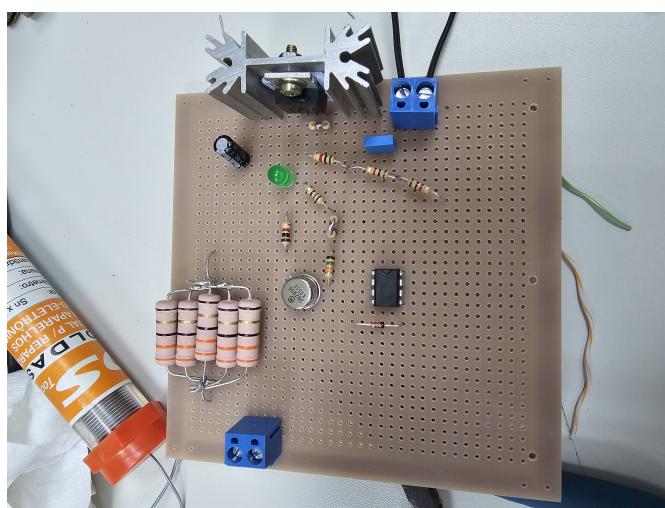
evitando sobrecarga e super aquecimento. O resistor R6 faz uma limitação de tensão. O circuito apresentado sofreu algumas mudanças entre o apresentado no material da *Texas Instruments* e o nosso circuito.

Dentre as mudanças, será necessário a troca do resistor R6, que no circuito apresentado na imagem é de $0,2\ \Omega$, por um de $5,6\ \Omega$ ou por um de $6,8\ \Omega$ que são valores comerciais, esses resistores limitam a corrente em $2,14\ A$ ou $1,76\ A$ respectivamente, contudo, Devido a alta corrente que passa por esse resistor seria necessário um resistor que suporte alta potencia dissipada, dada a dificuldade de encontrar um resistor com as especificações, foi feita a troca de um resistor para uma troca de uma associação em paralelo de 5 resistores de $33\ \Omega$ chegando ao valor de $6,8\ \Omega$. Como são 5 resistores idênticos a corrente de $1,76\ A$ fica dividida entre esses 5 resistores.

Além desses ajustes foi feito a troca do Amplificador Operacional LM301A pelo TL081 essa mudança foi feita por conveniência, uma vez que o TL081 foi mais fácil de encontrar que o LM301A, mas essa troca também nos permite eliminar o capacitor de $1000\ pF$, uma vez que, esse capacitor é usado para compensação de frequência, o TL081 dispensa esse capacitor pois dentro de seu circuito integrado ele já possui esse capacitor integrado.

Uma vez o circuito montado e soldado na placa ilhada ele ficou como na Figura 54:

Figura 54 – Circuito de Carregamento de baterias de 12 volts, placa ilhada



Fonte: Autores

Devido a nova placa que apresenta componentes como o Amplificador Operacional e o próprio LM338, o trafo de 12 volts se torna inapropriado, pois conforme a equação do cálculo de tensão de pico:

$$V(\text{pico}) = V(\text{rms}) * \sqrt{2} \quad (14)$$

o trafo de 12 volts nos entrega aproximadamente 17 volts após a conversão AC-DC e conforme a imagem 38, precisamos de uma entrada de tensão de 18 volts ou mais, por isso foi feito a troca para um trafo mais robusto de 18 volts e 5 amperes. Utilizando a equação de tensão de pico temos aproximadamente 25 volts DC com esse trafo, o suficiente para alimentar o circuito de carga.

A tensão de saída varia de acordo com o carregamento da bateria. A tensão começa baixa em torno de 9 volts e vai subindo conforme o carregamento avança, chegando a um pico de aproximadamente 14 volts quando a bateria esta carregada. O amplificador operacional compara a tensão e quando a bateria esta em sua tensão máxima e, por consequência, carregada por completo, o carregamento é encerrado. O LED verde presente na placa serve de identificador de carga, quando apagado o carregamento está em andamento e quando aceso o carregamento foi encerrado.

5.3 MECÂNICA

Apresentação dos testes de resistência e deformação sobre os andares internos do robô, além da preparação para a montagem do circuito elétrico no corpo.

5.3.1 Testes de resistência e deformação

Com toda a parte estrutural preparada, como complemento, realizamos algumas simulações para comprovar a resistência do mesmo. Todos os componentes foram pesados justamente com o reservatório preenchido com água.

Abaixo nas Tabelas 10 e 11, estão as somatórias de peso dos componentes que seriam posicionados nos andares, a compilação dos pesos é importante para podermos fazer a simulação mais fidedigna possível.

Tabela 10 – Tabela de peso dos componentes do andar inferior

Componente	Peso (gramas)
Motores de locomoção	400
Motores das escovas laterais	110
Motor de sucção	80
Baterias	290
Reservatório preenchido com água	650
Diversos (fios, conectores, materiais de fixação, etc.)	35
Total:	1.565

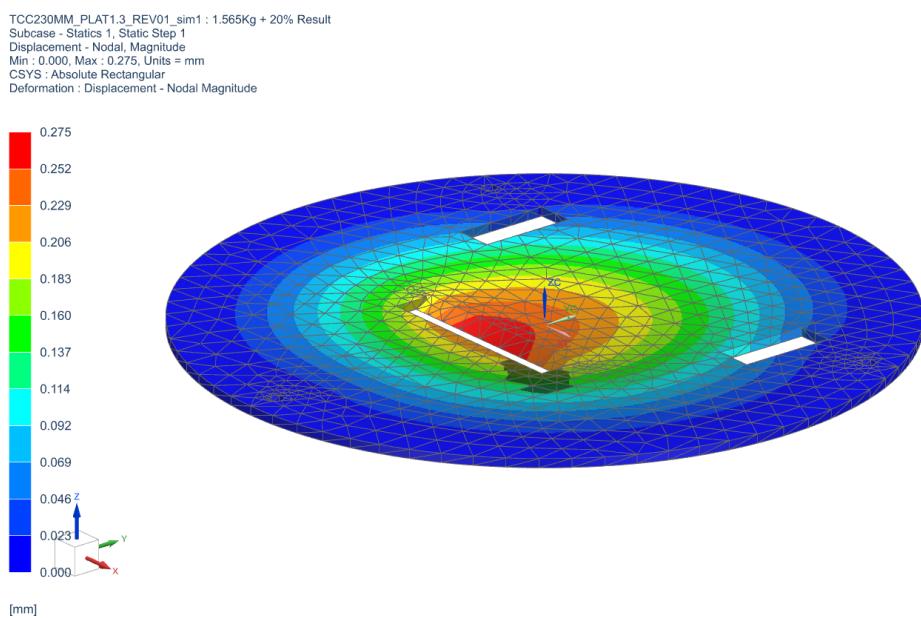
Tabela 11 – Tabela de peso dos componentes do andar intermediário

Componente	Peso (gramas)
Raspberry Pi 4B 64 bits ARM	50
ESP32	2
Pontes H	60
Diversos (fios, conectores, materiais de fixação, etc.)	50
Total:	162

A partir dos pesos e dos projetos 3D fizemos as simulações para o FEM (*Finite Element Method*), no qual se baseia em usarmos o nosso modelo das peças para avaliarmos as tensões e deformações que essas peças sofreriam.

Com isso a nossa simulação foi com o andar inferior, que de acordo com o nosso levantamento dos pesos dos componentes, totalizou 1,565 Kg, além disso devido a nossa confiança no material empregado que era o ABS, junto com peso baixo dos componentes, o nosso coeficiente de segurança foi estipulado em 1,2, ou seja estamos incrementando 20% no peso original, esse esforço foi posicionado no centro de cada peça, com seu respectivo peso, com o sentido em -Z, ou seja, sua direção está voltada para baixo. Com isso a nossa primeira simulação pode ser vista na Figura 55, onde a deformação no ponto de maior stress foi de apenas 0,275mm, comprovando a segurança do nosso modelo.

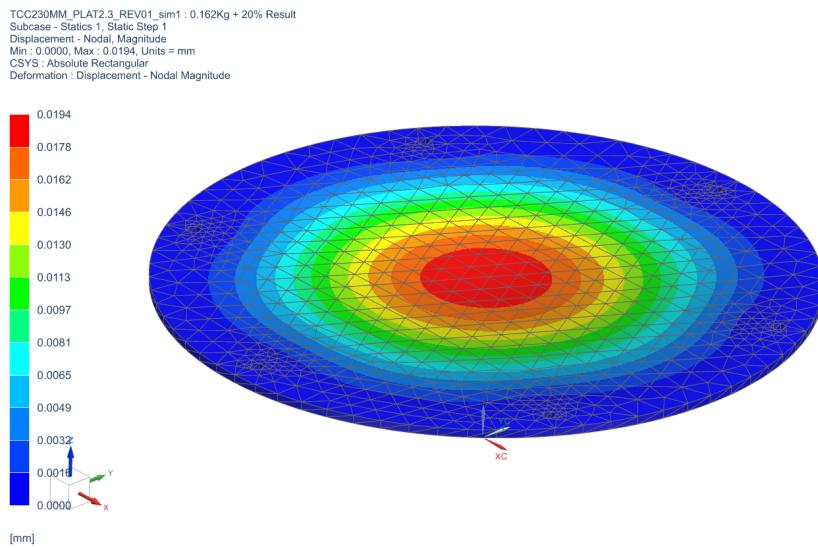
Figura 55 – Simulação FEM, Andar Inferior com 1,565 Kg + 20%



Fonte: Autores

A mesma lógica foi implementada no andar intermediário, com o peso de 0,162Kg e obtendo uma deformação máxima de 0,019mm, como podemos ver na Figura 56.

Figura 56 – Simulação FEM, Andar Intermediário com 0,162Kg + 20%

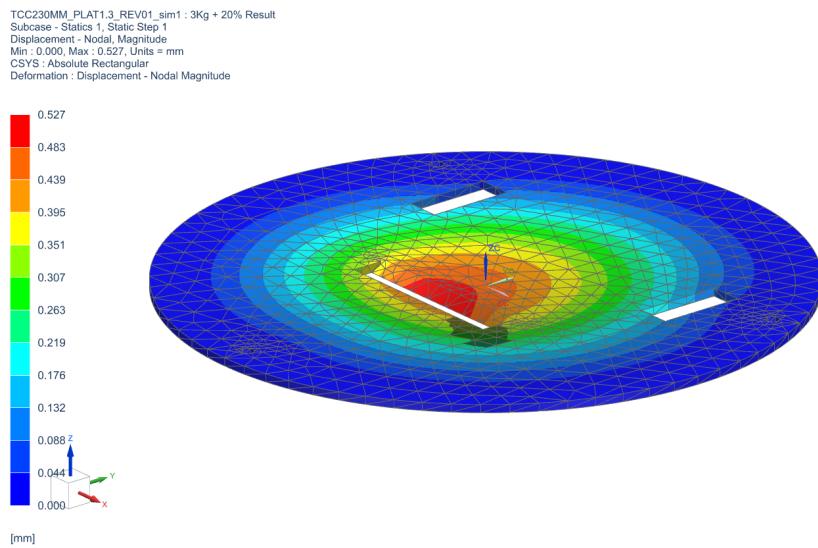


Fonte: Autores

Como um último teste de resistência, subtemos o andar inferior a um peso de 3kg, no qual é o peso aproximado de quanto o robô pesará com todos os componentes e com o seu *dispenser* cheio.

O resultado obtido foi uma deformação máxima de apenas 0,5mm, assim garantimos que a base aguentaria o peso do robô inteiro. Esse resultado pode ser encontrado na Figura 57.

Figura 57 – Simulação FEM, Andar Inferior com 3Kg + 20%

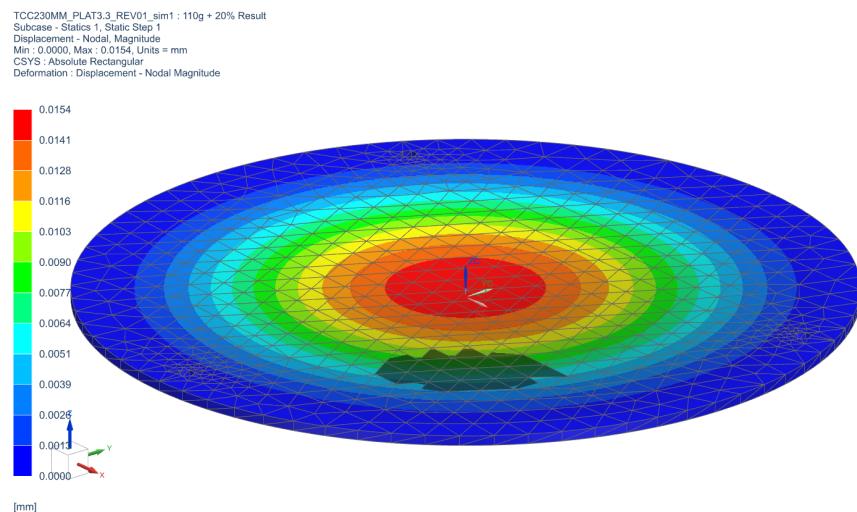


Fonte: Autores

Finalmente, no último andar, como planejado, haverá apenas o *LiDAR* que possui um peso de apenas 110g, esse peso gerará uma deformação ínfima no nosso material, mas testamos para assegurar a resistência de todos os andares do nosso robô, com a simulação comprovamos

que foi gerada uma deformação desprezível de apenas 0,0154mm, como pode ser visto na Figura 58.

Figura 58 – Simulação FEM, Andar Superior com 110g + 20%



Fonte: Autores

5.3.2 Montagem inicial dos componentes

Após os testes de resistência e deformação, continuamos com a montagem dos componentes que possuíam posições pré-determinadas e que ocupariam mais espaço, para melhor acomodação dos demais componentes, entre eles motores de tração, motor de sucção junto com seu local de armazenamento e o *LiDAR* já que ele teria um andar exclusivo para ele, confirmando a altura da cúpula. Como mostrado na Figura 59.

Figura 59 – Montagem Inicial dos Componentes



Fonte: Autores

5.4 EFETIVIDADE NO MERCADO E CUSTOS

Ainda que o circuito elétrico não tenha sido adequadamente instalado, compramos todos os componentes essenciais e temos uma estimativa de custo para a construção completa do protótipo do robô, que varia de R\$2.500,00 a R\$3.000,00.

A única companhia que identificamos que comercializa modelos industriais de robôs aspiradores é a Makita, cujos robôs são robustos, resistentes e significativamente caros. Atualmente existem dois modelos que podem ser vistos na Figura 60, sendo o da esquerda o DRC200 lançado em 2018 e o da direita seu sucessor mais avançado DRC300, anunciado no ano de desenvolvimento deste projeto (2024). A criação deste novo modelo comprova que existe um mercado interessado neste tipo de robô. Suas principais características incluem:

- DRC200: Programação de navegação básica com ações pré-estabelecidas, executadas de forma aleatória e repetitiva, semelhante aos modelos domésticos mais básicos, ou seguindo o padrão de movimento em zig-zag. Inclui alguns sensores ultrassônicos e um sensor bumper para identificar obstáculos, bem como um controle remoto. Preço de mercado variando de R\$6.900,00 a R\$8.500,00.

- DRC300: Atualmente, o modelo mais completo dispõe de um LiDAR, uma câmera, seis sensores ultrassônicos, cinco sensores de altura e um bumper. Há dois métodos de limpeza, um aleatório e outro que envolve a execução de um mapeamento e o acompanhamento da rota de limpeza mais eficaz. Preço de mercado variando entre R\$12.500,00 e R\$15.000,00 para compras internacionais sem a adição dos valores tributários, enquanto em compras nacionais possuem o valor inicial de R\$20.000,00.

Figura 60 – Exemplos de robôs industriais



Fonte: DRC200 (MAKITA, s.d.[a]) (esquerdo); DCR300 (MAKITA, s.d.[b]) (direito), 2024

Ambos os robôs foram projetados para a indústria, contudo, o modelo DRC200, que foi criado em 2018, dois anos antes das primeiras normas de robôs industriais móveis, não cumpre todos os requisitos, como o botão de emergência, a detecção de obstáculos à distância e o controle manual à distância.

Ou seja, além de nosso robô, um único modelo comercializado atualmente estaria em conformidade com as normas, o DRC300. Seu inconveniente seria o elevado custo, cerca de 4 a 5 vezes superior ao valor investido neste projeto.

6 CONCLUSÃO E TRABALHOS FUTUROS

Considerações finais entre os resultados esperados e obtidos pelo projeto, juntamente com a apresentação de aprimoramentos propostos com base nas dificuldades que encontramos e aprendizados obtidos durante sua realização.

6.1 DISCUSSÃO E CONCLUSÃO

Ao longo do desenvolvimento do projeto, conseguimos, por meio das simulações, adequações às normas e metodologias, comprovar a eficácia do robô simulado para aplicação no estudo de caso na empresa, entretanto, vale ressaltar que os resultados obtidos na simulação devem ser testados no local para garantir o bom funcionamento e a comprovação das conclusões levantadas.

Considerado que a proposta do projeto era melhorar o processo de limpeza de linhas, era esperado que o robô conseguisse, de forma completamente autônoma, realizar a limpeza da região desejada em um tempo menor ou igual a 30 minutos, sendo este o atual tempo gasto em cada paralisação. Como comprovado com os testes de navegação em ambiente simulado, este objetivo foi alcançado, com um tempo médio de limpeza de aproximadamente 8 minutos. Com um de tempo de limpeza tão curto, é possível fazer o retorno do robô pelo mesmo caminho utilizado na ida, otimizando a eficiência da limpeza e recolhendo novos detritos que foram derrubados no local após a passagem do robô.

Juntamente com isso, o projeto também deveria reduzir o risco ocupacional dos integrantes da empresa que realizam esse procedimento e aumentar a produção ao eliminar a necessidade da parada ou intertravamento da linha para limpeza.

A simulação da navegação comprovou que, utilizando os algoritmos desenvolvidos, o robô estaria apto a adentrar até os espaços mais estreitos da linha para realizar a limpeza, isso juntamente com o tempo médio estimado da limpeza, que atende nossa duração de bateria e é mais rápido que o procedimento atual e uma precisão acima de 85% que irá garantir a boa limpeza da área comprova que a sua aplicação é viável.

Observando os pontos citados acima, o projeto não só aumenta a produção gerando maior captura para as empresas, como também reduz o risco de acidentes que podem gerar indenizações, tudo isso enquanto otimiza o processo da limpeza.

No que diz respeito à parte mecânica do projeto, alcançamos o resultado esperado conforme demonstrado em nosso progresso. Apesar do contratempo causado pela manutenção

das impressoras do laboratório, conseguimos resolver essa questão em um período adequado para não comprometer a produção da estrutura básica para o projeto, onde seriam instalados os componentes essenciais para o seu funcionamento adequado.

Inicialmente estava planejado a criação de uma parede que envolveria todo o nosso robô, essa estrutura teria mais um papel estético do que uma função de proteção, considerando que é improvável que algum fragmento consiga penetrar pelas laterais do robô, dado que ele seria elevado do chão através das rodas. No entanto, decidimos remover a parede do projeto por três motivos que consideramos mais importantes. O primeiro é a redução do peso do robô, o segundo é que, sem as paredes, os espaçadores ficariam à mostra, assim se precisássemos removê-lo da área de limpeza, poderíamos facilmente fazê-lo com um gancho, por fim aumentar o fluxo de ar na parte interna do robô por conta de componentes que aquecem facilmente como o *Raspberry Pi*. Os materiais escolhidos demonstraram excelente resistência durante a montagem com o circuito.

Com base nos resultados das simulações, sabemos que as funcionalidades planejadas para o robô são eficazes, assim estão aptas para serem implementadas e testadas no robô real. Além da finalização da parte mecânica, que comporta todo o circuito elétrico e suporta o peso do robô em sua capacidade máxima de carga.

6.2 TRABALHOS FUTUROS

Ainda temos alguns pontos a serem aprimorados no projeto, bem como ideias para o desenvolvimento de uma segunda versão do robô com funcionalidades extras, uma vez que já estamos cientes dos desafios que teremos que superar e dos aspectos que devem ser priorizados durante o processo de desenvolvimento.

Na versão vigente, é imprescindível concluir a parte elétrica com a instalação completa e operacional, a fim de integrar a programação ao robô real.

Ao considerar uma versão posterior, poderíamos remover a restrição de tamanho que temos atualmente e igualá-lo aos robôs disponíveis no mercado, liberando espaço para um sistema de baterias mais eficiente, motores mais potentes e a inclusão de outros componentes que consideramos relevantes, tais como:

Sensores ultrassônicos: Um método alternativo para identificar obstáculos, trabalhando em conjunto com o LIDAR para melhor localização e redundância de sensores;

- **Hub USB:** Para aumentar a quantidade de integrações possíveis com o Rasp, abrindo espaço para instalação de mais melhorias;

- **Display LCD:** Mostrar dados do robô e facilitar a transição entre funções ou programas de navegação, sem a exigência de ligar o *Raspberry Pi* a um monitor e alterar manualmente oferecendo também *feedback* das funções internas e problemas do robô para o usuário;

- **Botões do tipo chave táctil:** Possibilitar a escolha das funções e mapas exibidos na tela;

- **Controle remoto próprio:** Design específico para o robô, sem botões supérfluos e possivelmente com um *display* com a mesma função do que seria encontrado no robô, porém com comandos remotos;

- **Base para carregamento e esvaziamento autônomo do reservatório:** Com a mesma função das bases de carregamento dos robôs domésticos mais sofisticados, onde o robô retornaria ao final da limpeza de maneira autônoma, iniciaria seu recarregamento, e teria seu reservatório esvaziado.

O aumento no tamanho do robô também permitiria retomar a ideia de usar a parede e adicionar entradas para ventilação dos componentes. Dessa forma, chegaríamos a um design mais refinado e parecido com os robôs aspiradores disponíveis no mercado, distanciando-nos do conceito de protótipo, mas mantendo sua característica esperada de ser um dispositivo mais robusto do que os demais.

Ao considerar o resfriamento do *Raspberry*, que acaba sendo um ponto de aquecimento para o nosso robô, seria útil considerar a possibilidade de reutilizar o ar que passa pelo sistema de aspiração e redirecioná-lo para o *Raspberry*, proporcionando assim uma outra forma de resfriamento além dos dissipadores já instalados.

REFERÊNCIAS

ALSADIK, Bashar; KARAM, Samer. The Simultaneous Localization and Mapping (SLAM)-An Overview. **Journal of Applied Science and Technology Trends**, v. 2, p. 120–131, nov. 2021. DOI: 10.38094/jastt204117.

AMAZON. **What is The Monte Carlo Simulation?** Disponível em: <https://aws.amazon.com/what-is/monte-carlo-simulation/>. (acessado em: 07.05.2024).

CREALITY, Shenzhen. **Impressora 3D, Creality Sermoon D1.** Disponível em: <https://www.creality.com/products/sermoon-d1-3d-printer>. (acessado em: 25.11.2024).

DAM, Tuan et al. Monte-Carlo Robot Path Planning. **IEEE ROBOTICS AND AUTOMATION LETTERS**, IEEE, p. 1–8, 2022.

ECOVACS. **How Does a Robot Vacuum Cleaner Work?** Disponível em: <https://www.ecovacs.com/uk/blog/how-does-a-robot-vacuum-cleaner-work>. (acessado em: 23.05.2024).

EREN, Anıl; DOĞAN, Hatice. Design and implementation of a cost effective vacuum cleaner robot. **Turkish Journal of Engineering**, Murat YAKAR, v. 6, n. 2, p. 166–177, 2022.

FERREIRA, Rosana. **Curiosidades que, provavelmente, você não sabia sobre aspirador de pó.** 2021. Disponível em: <https://revistacasaejardim.globo.com/Casa-e-Jardim/Dicas/Compras/noticia/2021/09/curiosidades-que-provavelmente-voce-nao-sabia-sobre-aspirador-de-po.html?~:text=Em%201901%2C%20surgiu%20o%20primeiro,que%20aspirasse%20a%20sujeira%20diretamente>. (acessado em: 16.05.2024).

FOX, D.; BURGARD, W.; THRUN, S. The dynamic window approach to collision avoidance. **IEEE Robotics & Automation Magazine**, v. 4, n. 1, p. 23–33, 1997. DOI: 10.1109/100.580977.

GYLLING, Andreas; ELMARSSON, Emil. Improving robotic vacuum cleaners. **EXAMENSARBETE INOM TECHNOLOGY, GRUNDNIVÅ, 15 HP**, KTH, p. 1–66, 2018.

HOFNER, Christian; SCHMIDT, Günther. Path planning and guidance techniques for an autonomous mobile cleaning robot. **Robotics and autonomous systems**, Elsevier, v. 14, n. 2-3, p. 199–212, 1995.

INSTRUMENTS, Texas. **LM138 and LM338 5-Amp Adjustable Regulators Datasheet (Rev. C).** [S.l.: s.n.], 2016. Datasheet. Available at <https://www.ti.com/lit/ds/symlink/lm338.pdf>.

IROBOT. **Robô Aspirador Roomba® 614.** Disponível em: <https://www.amazon.com.br/Roomba-614-Aspirador-Inteligente-iRobot/dp/B07W7W52QL>. (acessado em: 30.11.2024).

JANSON, Lucas; SCHMERLING, Edward; PAVONE, Marco. **Monte Carlo Motion Planning for Robot Trajectory Optimization Under Uncertainty.** Disponível em: <https://web.stanford.edu/~pavone/papers/JSP.ISRR15.pdf>. (acessado em: 18.04.2024).

JOON, Arpit; KOWALCZYK, Wojciech. **Design of Autonomous Mobile Robot for Cleaning in the Environment with Obstacles.** Disponível em: <https://doi.org/10.3390/app11178076>. (acessado em: 11.04.2024).

KANADE, Vijay. **What Is a Robot Operating System (ROS)? Meaning, Working, Applications, and Benefits.** 2024. Disponível em: <https://www.spiceworks.com/tech/artificial-intelligence/articles/what-is-robot-operating-system/>. (acessado em: 01.06.2024).

LESTER, Patrick. **A* Pathfinding for Beginners.** 2005. Disponível em: <https://csis.pace.edu/~benjamin/teaching/cs627/webfiles/Astar.pdf>. (acessado em: 03.06.2024).

MACENSKI, Steve et al. **Regulated Pure Pursuit for Robot Path Tracking.** [S.l.: s.n.], 2023. arXiv: 2305.20026 [cs.RO]. Disponível em: <https://arxiv.org/abs/2305.20026>.

MAKITA. **Robô Aspirador DRC200.** Disponível em: <https://www.makita.com.br/catalogoFDetalhes.asp?codParamD=602>. (acessado em: 30.11.2024).

MAKITA. **Robô Aspirador DRC300.** Disponível em: <https://www.makita.com.br/catalogoFDetalhes.asp?codParamD=1635>. (acessado em: 30.11.2024).

MIDEA. **Robô Aspirador ConnectLaser M7.** Disponível em: https://www.mideastore.com.br/robo-aspirador-connectlaser-m7-passa-pano-preto-midea/p?idsku=165&gad_source=1. (acessado em: 30.11.2024).

THRUN, Sebastian. Robotic mapping: a survey. In: EXPLORING Artificial Intelligence in the New Millennium. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2003. P. 1–35. ISBN 1558608117.

ULRICH, Iwan; MONDADA, Francesco; NICOUD, J.d. Autonomous Vacuum Cleaner. **Robotics and Autonomous Systems**, v. 19, mar. 1997. DOI: 10.1016/S0921-8890(96)00053-X.

YANG, Liwei et al. Path Planning Technique for Mobile Robots: A Review. **Machines**, v. 11, n. 10, 2023. ISSN 2075-1702. DOI: 10.3390/machines11100980. Disponível em: <https://www.mdpi.com/2075-1702/11/10/980>.