INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE MINAS GERAIS - *CAMPUS* ITABIRITO
ENGENHARIA ELÉTRICA

Leandro Henrique Vidigal Sousa

DESENVOLVIMENTO DE UM VEÍCULO AUTÔNOMO EM ESCALA REDUZIDA

Itabirito - MG
2022

LEANDRO HENRIQUE VIDIGAL SOUSA

DESENVOLVIMENTO DE UM VEÍCULO AUTÔNOMO EM ESCALA REDUZIDA

Trabalho de conclusão de curso apresentado ao Curso de Engenharia Elétrica do Instituto Federal de Educação, Ciência e Tecnologia de Minas Gerais - *Campus* Itabirito para a obtenção do título de Engenheiro Eletricista.

Orientador: Prof. Me. Elias José de Rezende Freitas

Itabirito - MG
2022

FICHA CATALOGRÁFICA

S725d 2022

Sousa, Leandro Henrique Vidigal

Desenvolvimento de um veículo autônomo em escala reduzida / Leandro Henrique Vidigal Sousa. – 2022.

73 f. : il.

Trabalho de Conclusão de Curso (Bacharelado em Engenharia Elétrica) – Instituto Federal de Educação, Ciência e Tecnologia de Minas Gerais – Campus Avançado Itabirito, 2022.

Orientador: Me. Elias José de Resende Freitas.

1. Robôs. 2. Navegação autônoma. 3. Desvios de obstáculos. I. Sousa, Leandro Henrique Vidigal. II. Instituto Federal de Educação, Ciência e Tecnologia de Minas Gerais – Campus Avançado Itabirito. III. Título.

CDD 629.892

Elaborada pela Biblioteca Jarbas Nazareth de Souza – Instituto Federal de Educação, Ciência e Tecnologia de Minas Gerais – Campus Avançado Itabirito

Bibliotecário Responsável: Veríssimo Amaral Matias – CRB-6/3266

Leandro Henrique Vidigal Sousa

DESENVOLVIMENTO DE UM VEÍCULO AUTÔNOMO EM ESCALA REDUZIDA

Trabalho de conclusão de curso apresentado ao Curso de Engenharia Elétrica do Instituto Federal de Educação Ciência e Tecnologia de Minas Gerais - Campus Avançado Itabirito para a obtenção do título de Engenheiro Eletricista.

Aprovado em 16/12/2022 pela banca examinadora:

Prof. Me. Elias José de Rezende Freitas (IFMG - Campus Ibirité)
Orientador (presidente da banca avaliadora)

Prof. Me. Gabriel Cambraia Soares (IFMG - Campus Avançado de Itabirito)
Membro avaliador

Prof. Esp. Carlos Dias da Silva Junior (IFMG - Campus Ibirité)
Membro avaliador

Me. Victor Ricardo Fernandes Miranda (UFMG/SBM Offshore)
Membro avaliador

AGRADECIMENTOS

Agradeço, primeiramente, aos meus pais por terem se dedicado incansavelmente, cada um de seus dias durante a minha criação me proporcionando o que puderam de melhor para meu futuro.

Agradeço a minha namorada Chrystiane de Fátima Goulart, pelo carinho, atenção e dedicação, sempre me incentivando a continuidade dos meus estudos.

Desejo agradecer aos meus irmãos por estarem sempre disponíveis e permitirem ser a pessoas no qual eu sempre poderei contar.

Quero agradecer também aos meus professores, que prestaram, com amor, a atividade de transmitir seu conhecimento e também ao Instituto Federal de Educação, Ciência e Tecnologia de Minas Gerais por fornecer os recursos necessários para garantir a qualidade de ensino com excelência.

Agradeço ao meu orientador por dedicar seu tempo e atenção para o desenvolvimento desse trabalho e crescimento pessoal.

E por último a meus amigos que sempre estiveram preocupados com minha formação e me incentivaram nessa caminhada.

“Jamais considere seus estudos como uma obrigação, mas como uma oportunidade invejável para aprender a conhecer a beleza libertadora do intelecto para seu próprio prazer pessoal e para proveito da comunidade à qual seu futuro trabalho pertencer."

Albert Einstein

RESUMO

Robôs são construídos para realizar tarefas para a humanidade, sendo que dentre essas tarefas, pode-se destacar, as tarefas de locomoção de objetos e pessoas. Nesse contexto, vários tipos de robôs são desenvolvidos para realizar uma tarefa de navegação em uma trajetória, podendo ser, totalmente ou parcialmente autônomos. Esse trabalho de conclusão de curso propõe o desenvolvimento de um robô, veículo autônomo em escala reduzida, com capacidade de realizar uma trajetória de forma completamente autônoma, dado um conjunto de coordenadas. A estratégia utilizada para navegação autônoma do robô foi a utilização de campos potenciais virtuais que se baseiam na simulação de campos atrativos e repulsivos para orientação do robô no espaço. A validação do modelo foi realizada, inicialmente, por meio de um ambiente de simulação com posterior construção do veículo. Os resultados alcançados demonstram que a estratégia de navegação de robôs por campos potenciais é eficaz para a solução de problemas locomoção espacial, tanto em simulação como nos testes realizados com o robô desenvolvido. Nesses resultados, o robô além de conseguir realizar a missão de chegar à coordenada determinada também conseguiu realizar o desvio de obstáculos.

Palavras-chave: Robôs, Navegação Autônoma, Desvio de obstáculos.

ABSTRACT

Robots are built to perform tasks for humanity, highlighting the ability to move objects and people. In this context, several types of robots are developed to perform trekking navigation tasks and can be, totally or partially autonomous. This final paper proposes the development of a robot, an autonomous vehicle on a reduced scale, capable of performing trekking in completely autonomous ways, given a set of coordinates. The strategy used for autonomous navigation was based on virtual potential fields, so that a combination of an attractive and a repulsive field guides the robot on the ground. The validation of the model was performed, initially, through a simulation environment with subsequent construction of the vehicle. The results achieved demonstrate that the strategy of robot navigation by potential fields is effective for the solution of space locomotion problems, both in simulation and in the tests performed with the robot developed. In these results, the robot besides being able to accomplish the mission of reaching the determined coordinate was also able to avoid obstacles.

Keywords: Robots. Autonomous Navigation. Obstacle Avoidance.

LISTA DE ILUSTRAÇÕES

LISTA DE QUADROS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

LIDAR	*Light Detection and Ranging*
IFMG	Instituto Federal de Ciência e Tecnologia de Minas Gerais
GPS	*Global Positioning System*
IMU	*Inertial Measurement Unit*
RPA	Método Probabilístico de Mapeamento
RRT	Método de Rápida Exploração por Árvores Randomizadas
ROS	Robot Operating System
RF2O	*Range Flow-based 2D Odometry*
PWM	*Pulse Width Modulation*
PID	*Proportional–Integral–Derivative Controller*

SUMÁRIO

1 INTRODUÇÃO

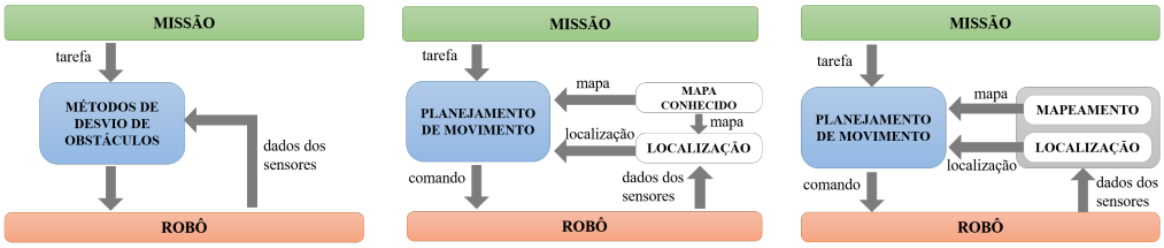
Novas soluções são desenvolvidas todos os dias com o avanço da tecnologia, nesse sentido, o desenvolvimento de robôs tem sido cada vez mais frequente na busca de resultados mais eficientes. De acordo com estudos publicados pela *Boston Consulting Group* (BCG), *Robotics Outlook* 2030 (LASSIG *et al.*, 2022), o mercado atual fatura cerca de 25 bilhões de dólares e irá aumentar esse faturamento em aproximadamente 10 vezes até 2030.

Nesse contexto, robôs móveis são desenvolvidos para solucionar problemas que envolvem a necessidade locomoção, assim, são desenvolvidos a partir de um um sistema de navegação em robótica que descreve como robôs móveis irão trafegar em um determinado local. Dessa forma, é necessário solucionar questões como, onde o robô está e para onde ele irá. Com isso, diversas estratégias foram desenvolvidas combinando métodos de localização, mapeamento e planejamento de movimento para que o robô execute tarefas previamente determinadas (ALVES *et al.*, 2011).

Atualmente, autores têm divulgado diversos trabalhos para obter melhorias em cada um desses métodos. Na IROS(*International Workshop on Intelligent Robots and Systems*), em 2021, diversos trabalhos foram divulgados nesse sentido: (WANG *et al.*, 2021) desenvolveram uma metodologia para realização de odometria e mapeamentos utilizando LIDAR e o modelo SLAM(*Simultaneous Localization and Mapping*), (ZHAO *et al.*, 2021) desenvolveram uma metodologia de fusão de sensores para obter uma odometria precisa, (KäSTNER *et al.*, 2021) apresentou em seu trabalho uma metodologia de desvio de obstáculo utilizando *Deep Learning* com métodos convencionais de planejamento de movimento.

Seja em um ambiente previamente conhecido ou não, estratégias são desenvolvidas para que seja possível a locomoção de um robô. De acordo com (FREITAS, 2017), as formas navegação podem ser agrupadas em três grupos, como ilustrado na Figura 1: (i) A navegação local reativa no qual é utilizado métodos de desvio de obstáculos em que o robô irá realizar suas ações baseadas nas condições do ambiente que são recebidas por informações obtidas por sensores, não existindo portanto, necessidade prévia de um mapa global do ambiente; (ii) Navegação local com com mapeamento em que um mapa local é gerado em tempo real e um planejador de caminhos é utilizado para fornecer a movimento a ser seguida, podendo, ser realizado a gravação em memória do mapa gerado ou não; (iii) Navegação com mapa global, em que previamente o robô tem a informação do ambiente por meio de um mapa fornecido, assim, o planejamento de movimento é realizado de forma antecipada e corrigida por métodos locais de desvios de obstáculos.

Figura 1 – Três estratégias de navegação: (a) Navegação local reativa. (b) Navegação global com conhecimento do mapa a priori. (c) Navegação local com mapeamento.



Fonte: FREITAS, 2017.

1.1 Motivação

Diante o rápido crescimento da robótica, é notório a aceleração no desenvolvimento de robôs móveis autônomos. Em 2019, a empresa Tesla apresentou um pacote de piloto automático em seus veículos que realiza a direção em estradas sem necessidade da mão do condutor sobre o volante. Essas soluções necessitam do desenvolvimento de métodos que, por meio de um sistema de navegação, irão possibilitar que o robô se localize, realize mapeamento e tenha um planejamento de movimento.

Figura 2 – Modelo de veículo autônomo de competição utilizado na F1TENTH.

Fonte: F1TENTH, 2022.

Nesse contexto, vários grupos têm realizado competições de veículos autônomos em escala reduzida afim de fomentar estudos para aplicação em veículos autônomos, buscando dessa forma, resolver problemas de navegação de veículos móveis terrestres de forma segura e cada vez

mais eficiente. Em (SRINIVASA *et al.*, 2019) os autores desenvolveram um protótipo de veículo autônomo móvel em escala reduzida de baixo custo compatível com competições realizadas ao redor do mundo, e além disso, foi disponibilizado uma plataforma de código aberto que contém um conjunto de informações para replicação do trabalho proposto, bem como, ambiente de simulação que contribui com estudos e pesquisas em robótica de veículos autônomos.

Na mesma linha, a Universidade da Pensilvânia em 2016 montou uma comunidade internacional que reúne pesquisadores, engenheiros e entusiastas de sistemas autônomos, denominada F1TENTH[1]. Desde então, essa comunidade tem realizado competições de corrida para veículos elétricos autônomos de escala reduzida, tal como o modelo apresentado na Figura 2.

Um ponto relevante dessa comunidade é que foi utilizado o ROS (*Robot Operating System*), um sistema para desenvolvimento de robôs. Nesse sentido, a comunidade construiu um sistema de simulação em ambiente ROS para veículos autônomos que possibilita realizar projetos em uma fase de testes para então iniciar a construção de um robô real, permitindo assim, migrar o que foi desenvolvido no simulador para o robô construído.

Esse ambiente permite o acesso a uma simulação de veículo com a cinemática de Ackermann, mapas de competições realizadas pelas F1TENTH, conjunto de sensores tipicamente utilizados em projetos de veículos autônomos e outras ferramentas que tornam a plataforma eficaz na produção de trabalhos nesse sentido. Dessa maneira, neste trabalho é utilizado como fonte a plataforma dessa comunidade para o desenvolvimento do protótipo.

Neste trabalho será abordado diversos problemas e soluções da robótica móvel ao explorar a utilização de metodologias desenvolvidas por diversos autores para o desenvolvimento de um protótipo de veículo elétrico autônomo em escala reduzida.

1.2 Objetivos

O objetivo geral do trabalho é desenvolver um veículo elétrico autônomo em escala reduzida.

Como objetivos específicos pode-se citar:

- testar um modelo de robô em ambiente de simulação para competições de corrida autônoma;
- automatizar um veículo de controle remoto adicionando sensores e eletrônica embarcada;
- implementar uma estratégia de navegação autônoma local com mapeamento;
- integrar o sistema do robô desenvolvido em ambiente de simulação com o veículo físico;
- adaptar e testar o robô em ambiente físico, bem como, analisar os resultados obtidos.

[1] <https://f1tenth.org/>

1.3 Contribuições

A principal contribuição deste trabalho é o desenvolvimento desde a mecânica, eletrônica até a comunicação, simulação e testes reais com um robô para competições de corrida autônoma. Além disso, este trabalho foi apresentado e publicado nos anais da Mostra Nacional de Robótica (MNR 2021):

1. SOUSA, L. H. V.; RIBEIRO, C. A.; FREITAS, E. J. R. Robô móvel autônomo para corrida F1TENTH In: Mostra Nacional de Robótica - Mostra Nacional de Robótica – 2021. , 2021. p.1 – 6.

1.4 Organização do Texto

Este trabalho está organizado da seguinte forma: O Capítulo 1 apresentou a introdução do trabalho, os objetivos a serem alcançados. O Capítulo 2 é dedicado para revisão bibliográfica, no qual serão apresentados os conceitos necessários para o desenvolvimento da metodologia. Em seguida, o Capítulo 3 é apresentado a metodologia necessária para desenvolver um veículo autônomo em escala reduzida. Tendo sido realizado o processo proposto da metodologia, o Capítulo 4 apresenta os resultados obtidos, bem como a análise de cada um deles. Por fim, no Capítulo 5, serão apresentados as considerações finais e proposta para trabalhos futuros.

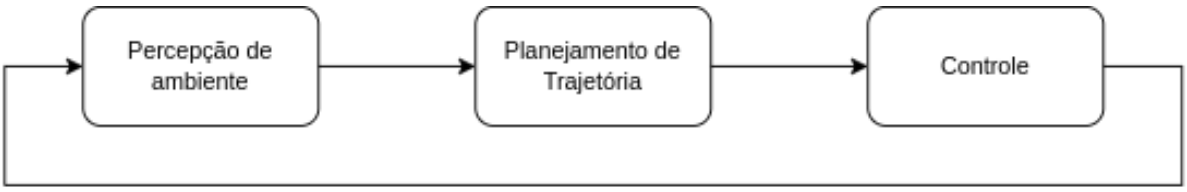
2 REFERENCIAL TEÓRICO

Neste capítulo, é apresentado uma revisão dos conceitos encontrados na literatura para desenvolvimento de robôs móveis. Para tanto, serão apresentados os conceitos de navegação de veículos autônomos (Seção 2.1) e por fim a explicação do sistema de desenvolvimentos de robôs, o ROS (Seção 2.2).

2.1 Navegação de veículos autônomos

De forma geral, veículos totalmente autônomos precisam de três módulos fundamentais para sua operação, que conforme Ghallabi (2020), são divididos em: Percepção de Ambiente, Planejador de Movimento e Controle, que seguem o fluxo demonstrado na Figura 3. A Percepção de ambiente irá retornar a informação do ambiente ao redor e sua localização, incluindo também, detecção de obstáculos, classificadores e outros. Para isso, diferentes tecnologias de sensoriamento são utilizadas, tais como: radares, câmeras, LIDARs e sensores de ultrassom. O Planejador de Movimento irá encontrar o caminho adequado pelo qual o robô deverá seguir entre dois pontos dentro de um sistema de coordenadas e determinará o desvio de obstáculos. Já o Controle irá calcular os comandos a serem aplicados ao robô para ele seguir um caminho planejado.

Figura 3 – Componentes fundamentais para direção autônoma.



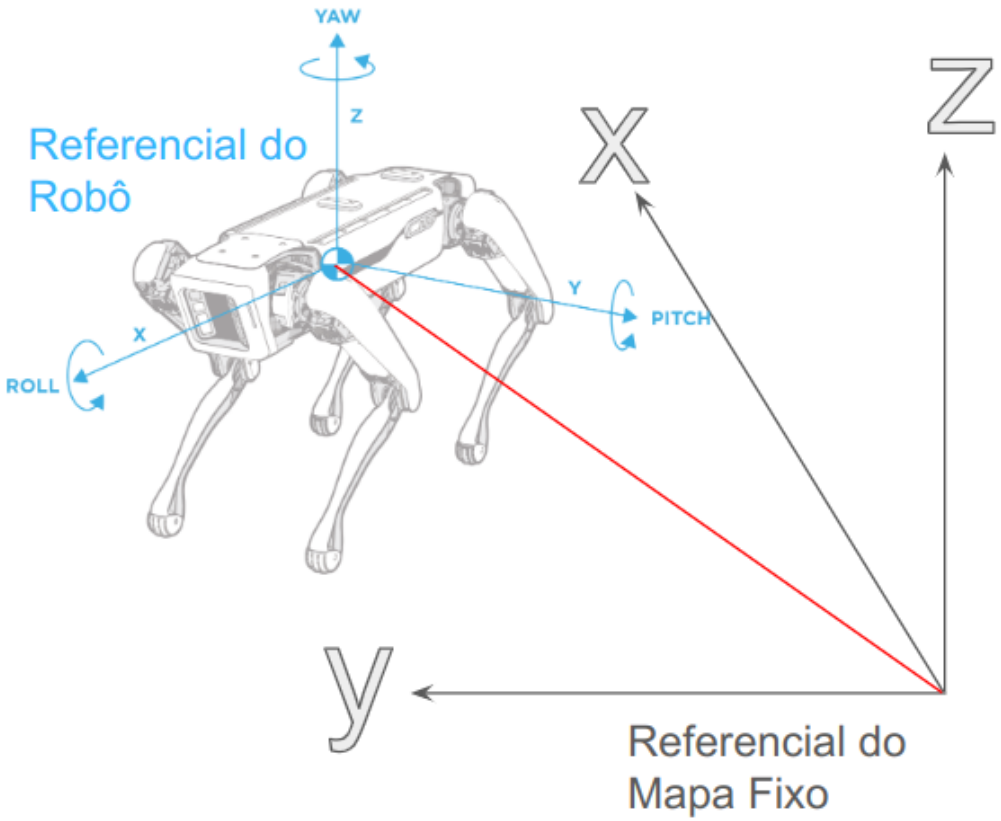
Fonte: *Adaptado de* GHALLABI, 2020.

2.1.1 *Localização*

A localização faz parte do sistema de Percepção de Ambiente de um robô, obtendo informações do seu movimento a partir da realização de odometria, enquanto navega-se em um ambiente. Essa odometria é realizada por meio da medição da velocidade e direção do veículo por sensores, tais como *Encoders*. Entretanto, essa medição sempre possuirá um determinado valor de erro que é acrescida ao longo do tempo, e por isso, outros sensores de recepção externa são utilizados para realizar a medição ambiente e, por meio da combinação desses sensores com os resultados obtidos pela odometria, é possível obter uma localização mais precisa do robô. Alguns desses sensores utilizados são, IMU (Unidade de medição Inercial) e LIDARs, que são combinados em métodos que foram desenvolvidos para realizar a localização de robôs, no qual, dentre eles, pode-se destacar os métodos probabilísticos, como o filtro de Kalman (PANIGRAHI; BISOY, 2021).

A localização de um robô precisa estar estabelecida dentro de um sistema de coordenadas, de maneira que se tenha uma mesma padronização para o desenvolvimento de projetos. Portanto, alguns sistemas de convenções de coordenadas são estabelecidos, tal como, o REP103 que fornece um referencial de unidades e convenções de coordenadas (FOOTE; PURVIS, 2010) e o REP105 que fornece a estrutura de coordenadas para aplicação em robótica (MEEUSSEN, 2010).

Figura 4 – Convenções de coordenadas.



Fonte: Elaborado pelo autor, 2022.

Portanto, ao se analisar um sistema coordenadas de um robô, deve-se ter em mente a configuração de orientação de eixos em relação ao robô conforme apresentado pela Figura 4, onde é possível perceber que sempre será considerado a comparação entre o referencial em relação ao robô com um referencial de um mapa que deve permanecer fixo. Ao se avaliar o referencial do robô, é comum convencionar que o eixo x sempre estará indicando para frente do robô, o eixo y para sua esquerda e eixo z para cima do robô. Em robô terrestres, o eixo de rotação *yaw* é o mais utilizado como referência para rotação com relação ao centro do robô. As coordenadas do mapa sempre estarão fixas em relação aos eixos de coordenadas e sempre será definida a partir de um modelo fornecido antecipadamente para o robô, quando não existe um mapa global fornecido. Normalmente, o referencial do mapa será definido pela posição e orientação inicial do robô.

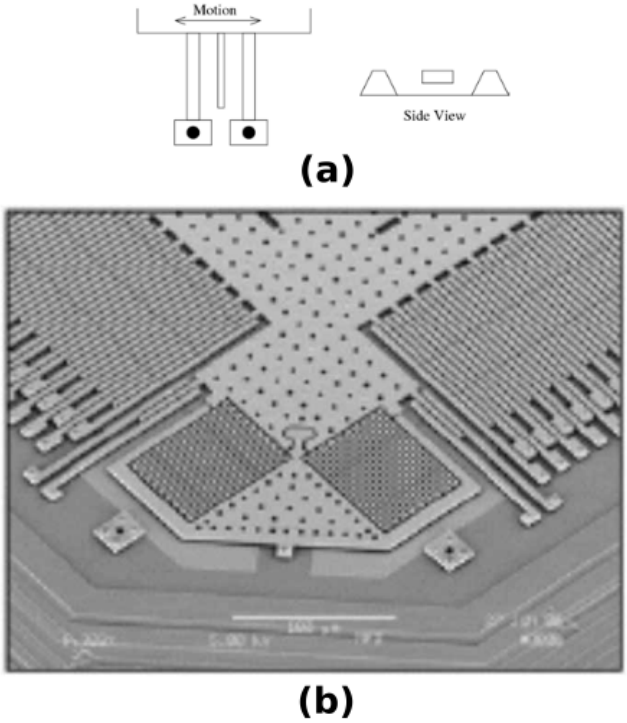
2.1.2 *Sensores*

Os sensores realizam medições importantes para que seja possível que o sistema de percepção do veículo tenha o conjunto de informações necessárias para se realizar uma navegação coerente. Sensores são elementos do robô que são responsáveis por realizar a medição das informações necessárias para garantir os componentes fundamentais da direção autônoma. Nesse cenário, vários desses sensores são tipicamente usados em diversos projetos de robô móveis autônomos terrestres e serão detalhados nessa seção.

A Unidade de Medição Inercial, ou do inglês *Inertial Measurement Unit* (IMU) é um dispositivo capaz de reportar informações de aceleração, orientação e força gravitacional. A tecnologia utilizada consiste na utilização da combinação de sensores de aceleração inercial e giroscópio (AHMAD; GHAZILLA; KHAIRI, 2013).

O sensor de aceleração inercial, comumente, utiliza o princípio de variação de capacitância para inferir a aceleração. A técnica utilizada é baseada na variação do meio dielétrico do capacitor a partir da variação da distância entre as suas placas condutoras. Essa variação é transformada em um sinal de frequência que pode ser correlacionado diretamente com o valor da aceleração (FAISAL; PURBOYO; ANSORI, 2020).

Figura 5 – Acelerômetro do tipo MEMS.



Fonte: OZGUNER; ACARMAN; REDMILL, 2011.

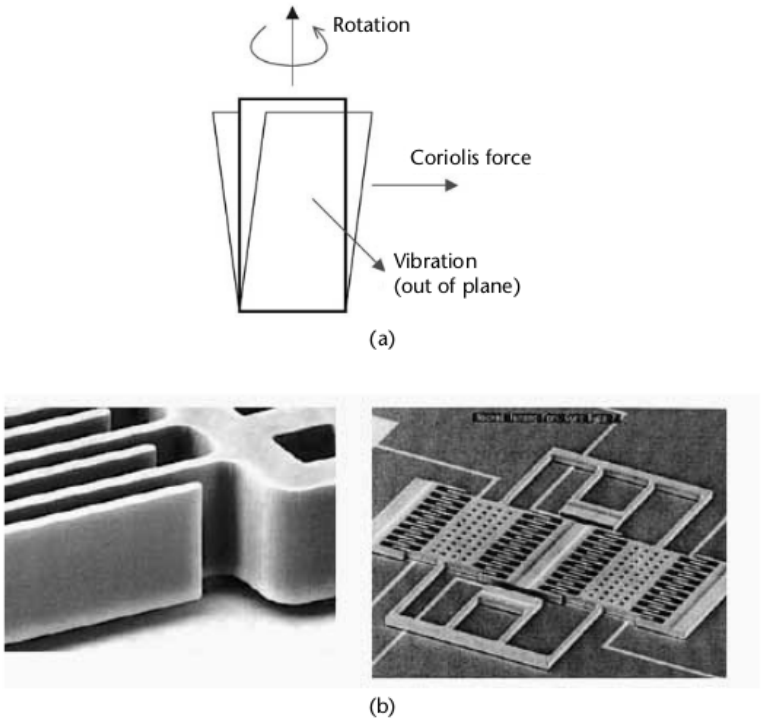
Na Figura 5(a) pode-se perceber que, quando existe um movimento, a placa central se move fazendo com que o valor da capacitância seja alterado. Na Figura 5(b) é visto um modelo construído do sensor, em sua estrutura, é visto a variação entre linhas paralelas de materiais

dielétricos e material condutores que cumprem o papel de realizar a medição da aceleração durante a variação do movimento.

Já o giroscópio tem a capacidade de realizar a medição da velocidade angular. Seu princípio de funcionamento é baseado na realização da medição da velocidade angular em um meio rotacional utilizando o conceito do efeito Coriolis. Para medição do efeito Coriolis, um giroscópio do tipo MEMS possui uma massa que vibra em um eixo a partir de um oscilador, assim, uma oscilação secundária é induzida em um eixo perpendicular no momento em que acontece um movimento de rotação com o dispositivo, então, o giroscópio sofre variações de capacitância para detectar esses deslocamentos (FAISAL; PURBOYO; ANSORI, 2020).

A Figura 6(a) apresenta o modelo do efeito Coriolis aplicado a um modelo de sensor. É possível perceber que o movimento de rotação no sensor faz com que exista uma vibração de sua placa. Na Figura 6(b) pode ser vista a construção do sensor em uma foto ampliada. As estruturas do sensor são ajustadas para que seja possível captar a variação da capacitância a partir da vibração das placas, um conjunto de placas são alinhadas paralelamente para poder mensurar o efeito de forma mais precisa.

Figura 6 – Modelo do Giroscópio tipo MEMS.



Fonte: OZGUNER; ACARMAN; REDMILL, 2011.

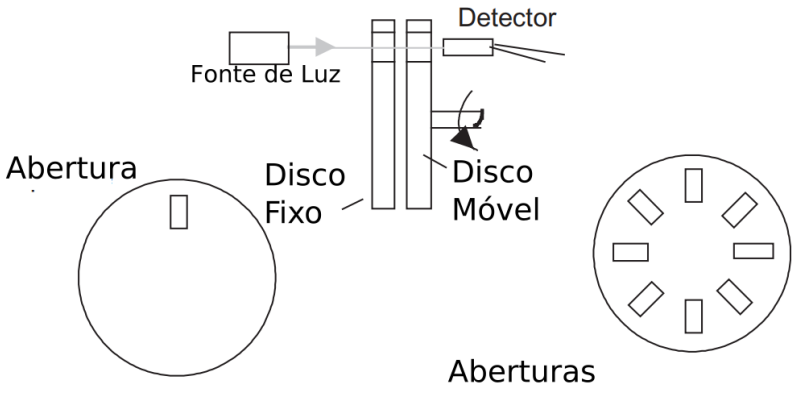
Com a IMU é possível retornar o valor da localização do veículo ao realizar a integração dos valores obtidos pelo sensor. Esse valor irá possuir alguma faixa de erro devido às incertezas do próprio dispositivo, não sendo, portanto, um sensor de uso singular para se determinar a localização.

Outro sensor comumente utilizado é o *Encoder*, que são dispositivos que fornecem uma saída digital que traduz um deslocamento angular ou linear. Existem dois tipos de *Encoders*: os incrementais, que fornecem o valor de uma mudança de posição a partir de uma posição inicial conhecida, e os absolutos, que fornecem a posição atual independente de um conhecimento anterior. O princípio de funcionamento é baseado na geração de um pulso de sinal digital na medida em que ocorre um determinado movimento, então esse pulso é contado e como a distância entre os pulsos é conhecida, pode-se determinar um deslocamento. Comumente, esses pulsos são gerados ou por sensores ópticos ou por sinal magnético utilizado do efeito hall (MORRIS; LANGARI, 2020).

Exemplos comuns de *Encoders* incrementais são os de disco fixo e disco móvel que possuem funcionamento baseado em uma fonte de luz que permanece por de trás de um disco fixo com uma abertura, enquanto um disco móvel com aberturas de dimensões idênticas e espaçadas igualmente se move fazendo com que exista uma condição de abertura para passagem de luz de forma padronizada. Essa luz é recebida por sensores ópticos que mandam o sinal.

Na Figura 7 é possível visualizar um modelo no qual uma luz emitida por uma fonte atravessa o disco fixo e o disco móvel do *Encoder*, e então é detectada pelo sensor. Na medida em que acontece o giro do disco móvel acontece a intermitência da medição da luz emitida.

Figura 7 – Modelo de *Encoder* incremental óptico

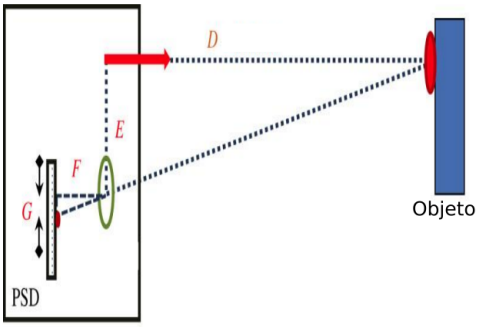


Fonte: *Adaptado de:* BOLTON, 2021.

O LIDAR (*Light Detection and Ranging*) é um sensor que realiza um escaneamento local por meio da emissão de pulsos de laser em uma superfície de determinado objeto para determinar a sua distância. Alguns métodos irão realizar a medição da distância por meio da diferença de tempo entre a luz emitida e a luz recebida, outros métodos de forma contínua irão medir a defasagem entre o valor emitido e recebido e também existem métodos que utilizam a triangulação, tal como mostrado na Figura 8.

Na Figura 8, a distância E representa a distância interna do sensor entre a fonte emitida e o ponto de convergência da luz recebida no disco, e a distância F representa a distância entre o ponto de convergência de luz e o sensor interno de medição, sendo, essas duas medidas fixas e

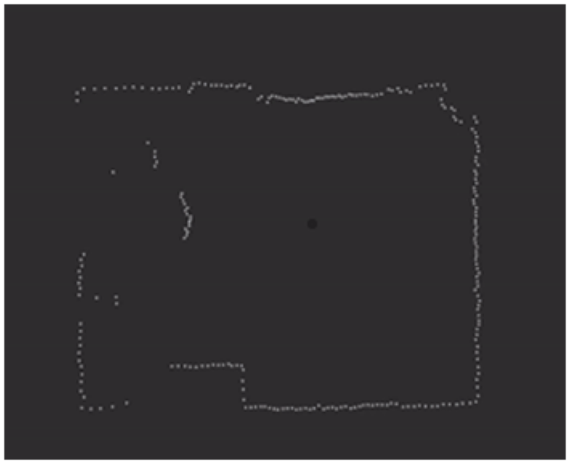
Figura 8 – Método de triangulação utilizado em sensor do tipo LIDAR.



Fonte: *Adaptado de:* MOHIUDDIN; HUSSAIN; ALI, 2021.

previamente conhecidas. A luz parte internamente do sensor e segue o caminho apresentado pela letra D até o ponto de interesse de medição, representando, portanto, a distância a ser medida. A luz será refletida em uma superfície e retornará ao sensor onde será detectada internamente por receptor, e, conforme a distância do objeto, retornará um valor proporcional representado pela letra G.

Figura 9 – Nuvem de pontos fornecido pelo sensor LIDAR.



Fonte: COMETLABS,2022.

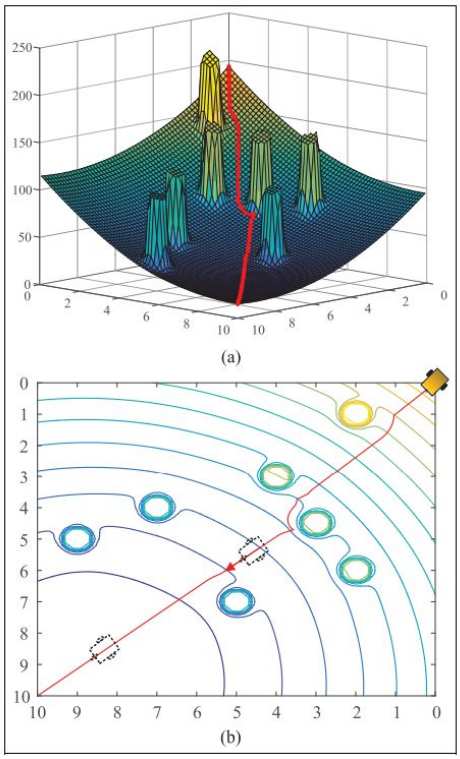
Esse principio é utilizado em conjunto com motores que fazem o movimento de rotação síncrona do sensor, resultando em um movimento circular que propicia a formação de uma uma nuvem de pontos que retorna a informação bidimensional do espaço ao redor do robô. Diversos trabalhos foram desenvolvidos utilizando esse modelo de LIDAR, como pode ser visto em Mohiuddin, Hussain e Ali (2021). Esse princípio fornece modelos de LIDARs de baixo custo que comumente são associados a motores que realizam movimento rotacional para fornecer uma mapa em duas dimensões do local por meio de um nuvem de pontos, tal como vemos na Figura 9.

2.1.3 *Planejador de movimento*

A função de um planejador de movimento é fornecer as coordenadas de uma rota que conecta um ponto inicial a um ponto final. Tipicamente um planejador de movimento pode ser dividido em duas partes, conforme Ozguner, Acarman e Redmill (2011):

- *Planejador Global de Movimento* - Tem como função definir o movimento necessário que o robô deverá realizar no caminho entre o ponto inicial da missão até o ponto final da missão.
- *Planejador Local de Movimentos* - Tem objetivo de selecionar o melhor movimento para o veículo diante as situações do ambiente em que o robô se encontra, sendo responsável pela realização de tarefas como o desvio de obstáculos.

Figura 10 – Método de planejamento de movimento por campos potenciais.(a) Apresentação tridimensional do gradiente do campo potencial sendo os picos a representação dos obstáculos e o vale o ponto de interesse;(b) Representação bidimensional do campo potencial onde os círculos são os obstáculos e os arcos convergem para o caminho de interesse.

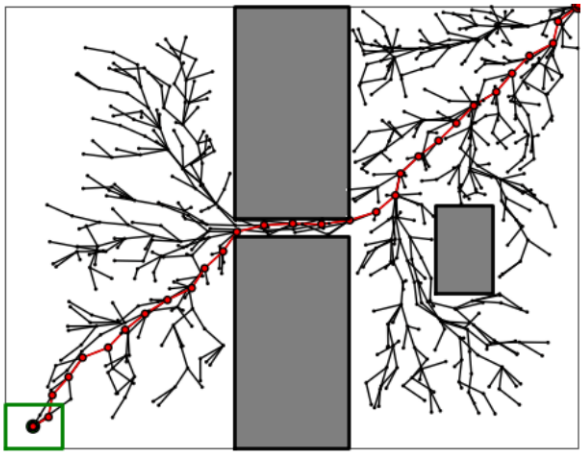


Fonte:LIN *et al.*, 2021

Nesse sentido, alguns métodos são desenvolvidos para a solução de planejadores globais e locais. Um desses métodos é a utilização de planejadores baseados em campos potenciais (KHATIB, 1985). A ideia é ter uma função de campo potencial atrativo para obter a caminho

global ao ponto de destino e ter um campo potencial repulsivo que incrementa na medida em que obstáculos se aproximam, mudando assim, a movimento localmente (BARRAQUAND; LANGLOIS; LATOMBE, 1992). Recentemente, autores têm evoluído os métodos de planejamento por campos potenciais, LIN *et al.* (2021) desenvolveram uma modificação no campo repulsivo do modelo do campo potencial clássico e demonstrou os resultados das curvas de campos representados pela Figura 10. Na figura é possível notar que a caminho em vermelho é definida por uma euclidiana distorcida pelos obstáculos do caminho. Na Figura 10(a), o gradiente máximo é encontrado no local de interesse de chegada.

Figura 11 – Método de planejamento RRT.



Fonte:ELBANHAWI; SIMIC, 2014

Ao longo do tempo, outros métodos também foram desenvolvidos para solução de planejamento de rotas, KAVRAKI *et al.* (1996), introduziram o método probabilístico de mapeamento (RPA) enquanto os autores LAVALLE *et al.* (1998) o método de rápida exploração por árvores randomizadas (RRT), conforme ilustrado na Figura 11, em que o método utiliza uma forma randomizada de criação de caminhos de forma a encontrar o menor percurso entre o ponto de partida e chegada.

2.2 *Robot Operating System (ROS)*

O ROS (*Robot Operating System*) é uma das principais ferramentas utilizadas em diversos trabalhos de robótica, isso porque, ele é uma plataforma que possibilita a construção descentralizada de projetos na área de robótica, possui um sistema que contém diversos mecanismos de auxílio na construção de robôs e permite a interação com diversas linguagens de programação.

O ROS é uma estrutura flexível para criação sistemas para robôs, ele possui um conjunto de ferramentas, bibliotecas e parâmetros que busca simplificar a atividade de criação de comportamentos complexos e robustos de robôs por meio de uma ampla variedade de plataformas de robótica. Podemos dividir o ROS nos principais segmentos:

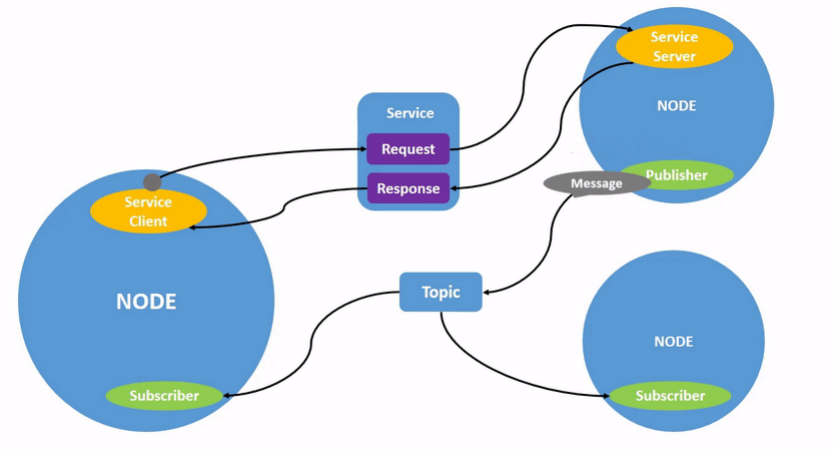
- Sistema de Comunicação - O ROS possui em sua estrutura um sistema de comunicação que, basicamente, por meio de *Publisher* e *Subscribers*, possibilitando a troca contínua de informações, ou por meio de *Services*, no qual um cliente irá solicitar uma informação a um servidor que enviará essa informação.

- Conjunto de Ferramentas - O ROS possui um conjunto de ferramentas integradas ao seu sistema, dentre elas pode-se citar, como exemplo, ambiente de simulação RVIZ e GAZEBO, no qual é possível nesse ambiente modelar um robô e visualizar seu funcionamento.

- Ecossistema - O ROS possui também um núcleo onde está integrado uma série de pacotes que podem ser utilizados para criação de robôs, além de permitir com linguagens de programação por meio de integração das bibliotecas roscpp (Linguagem programação em C),rospy (Linguagem de Programação em Phyton) e roslisp (Linguagem de programação Lisp) e também possui uma estrutura de funcionamento por meio de nós que permitem a descentralização dos processos de criação, permitindo dessa forma, a criação de projetos em código aberto por várias pessoas de forma simultânea.

O sistema de comunicação do ROS possui uma estrutura composta de nós, em cada nó pode ser realizada uma função específica. Entretanto, é necessário que esses nós se comuniquem, e, essa comunicação é realizada por meio de mensagens que são direcionadas a tópicos ou serviços, como exemplificado na Figura 12. No caso dos tópicos, eles são estruturas responsáveis por receber e transmitir a informação de um nó para outro nó de forma contínua, sendo que a mensagem em um nó é enviada por meio de *Publishers* e recebida por meio de *Subscribers*. Já o serviço é um tipo de comunicação entre um servidor e um cliente, nesse caso, o cliente requisita a informação a um servidor e a mesma é enviada por ele, sendo assim, essa ação ocorre somente uma vez quando existe a solicitação.

Figura 12 – Mensagem sendo publicada no tópico por um nó e realização de um serviço entre servidor e cliente.



Fonte: ROS,2022.

2.3 Trabalhos Relacionados

Esse trabalho tem como referência alguns trabalhos semelhantes que são descritos na Tabela 1. Em resumo, Vajnar (2017) desenvolveu em sua dissertação um veículo autônomo para competição na F1TENTH, ele utilizou como método de planejamento de movimento os CP (Campos Potenciais) e um algoritmo de procura denominado A*, além disso, o autor utilizou o método convencional SLAM (*Simultaneous Localization and Mapping*) para localização, em seu trabalho foi abordado os métodos para construção do veículo autônomo e detalhes do desenvolvimento do software na plataforma ROS, os sensores utilizados no trabalho são semelhantes aos que serão vistos neste trabalho, entretanto a odometria não utilizou *Encoder*. Um trabalho semelhante a esse foi desenvolvido por Kopecky (2019), em que o veículo possui as mesmas características construtivas do anterior, entretanto, o método de planejamento utilizado foi o FP (*Forward Pass*) e de localização o MCL (*Monte Carlo Localization*). Srinivasa *et al.* (2019) desenvolveram um veículo autônomo em escala reduzida de baixo custo, o qual teve como foco a construção do robô e o ambiente de simulação. Em seu método, a localização do veículo é realizada com a odometria obitda pela cinemática do veículo a partir dos dados do controlador pelo ângulo das rodas e velocidade linear do robô, todas as informações podem ser encontradas em: <https://mushr.io/>. Gotlib, Lukojc e Szczygielski (2019) desenvolveram um robô autônomo robusto e utilizou como método para localização no espaço o AMCL (*Adaptive Monte Carlo Localization*), diferente dos outros trabalhos, como método de localização do robô, os autores utilizaram diversos sensores, incluindo a utilização de *Encoder* para medição da velocidade linear do veículo.

Apesar de ambos os trabalhos terem como objetivo o desenvolvimento de um veículo autônomo para navegação e competição, os componentes de hardware bem como o sistema de navegação desenvolvido para cada um dos robôs tem características distintas, mostrando assim, diversas formas de solucionar problemas de navegação autônoma. Neste trabalho, será apresentado o desenvolvimento de um veículo autônomo semelhante aos citados, e, como solução, no planejamento de movimento será abordado uma metodologia de CP e para sua localização KF(*Kalman Filter*).

Tabela 1 – Tabela comparativa de trabalhos relacionados.

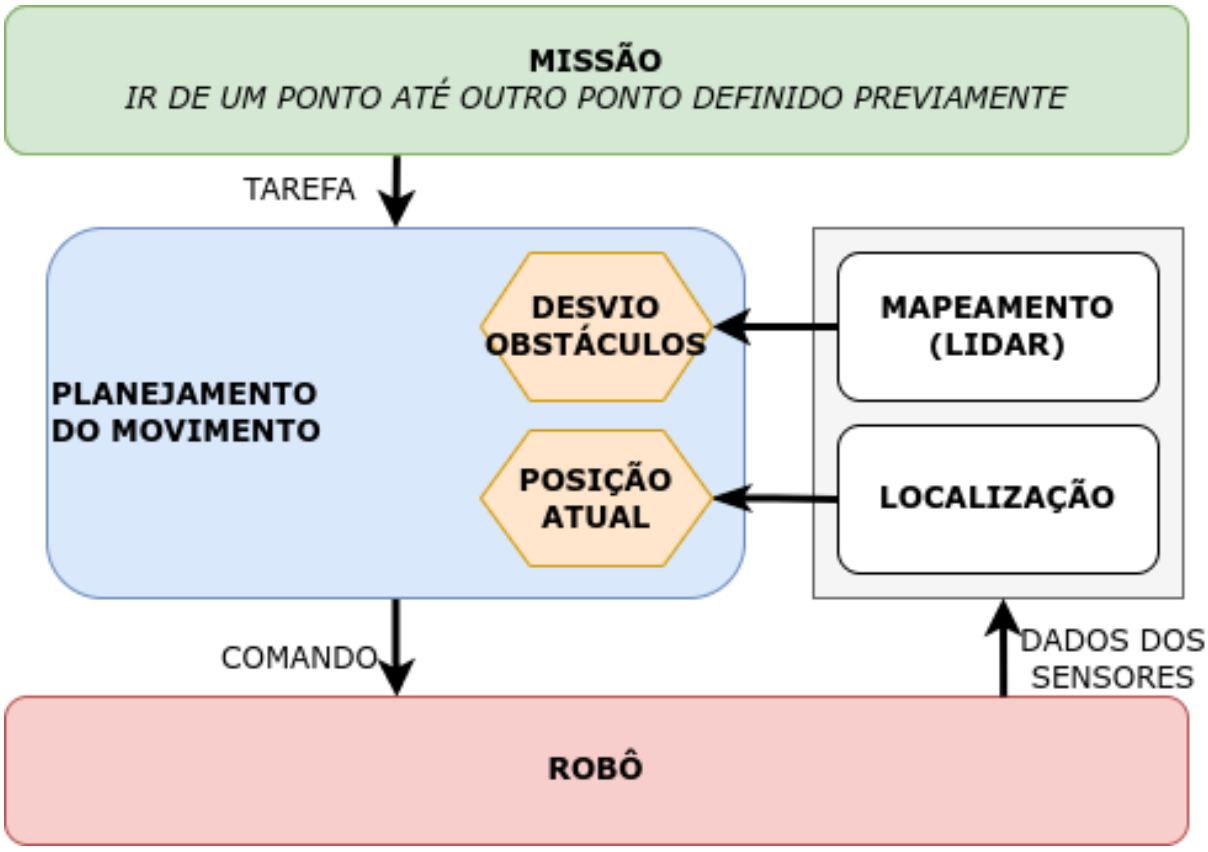
Autor	**Planejamento de Movimento / Localização**	**Unidade de Processamento**	**Sensores**			**Atuadores**		
			LIDAR	***Encoder***	**IMU**	**Motor**	**Servo**	**Controlador**
(VAJNAR, 2017)	CP+A*/SLAM	JETSON TK1	x	-	x	x	x	x
(KOPECKY, 2019)	FP/MCL	JETSON TX2	x	-	x	x	x	x
(SRINIVASA *et al.*, 2019)	-	JETSON NANO	x	-	-	x	x	x
(GOTLIB; LUKOJC; SZCZYGIELSKI, 2019)	-/AMCL	Odroid XU4	x	x	x	x	x	x
Este trabalho	CP/KF	Raspberry Pi4 B	x	x	x	x	x	x

3 METODOLOGIA

3.1 Estratégia de Navegação

O primeiro passo para o desenvolvimento de um veículo autônomo é decidir a estratégia de navegação. Com isso, é possível determinar quais serão os componentes de *hardware* e *software* para construção do robô. Nesse trabalho, tal como descrito em Freitas (2017), a estratégia de planejamento foi baseada no modelo de navegação local com mapeamento sem memória de armazenamento, como visto na Figura 13.

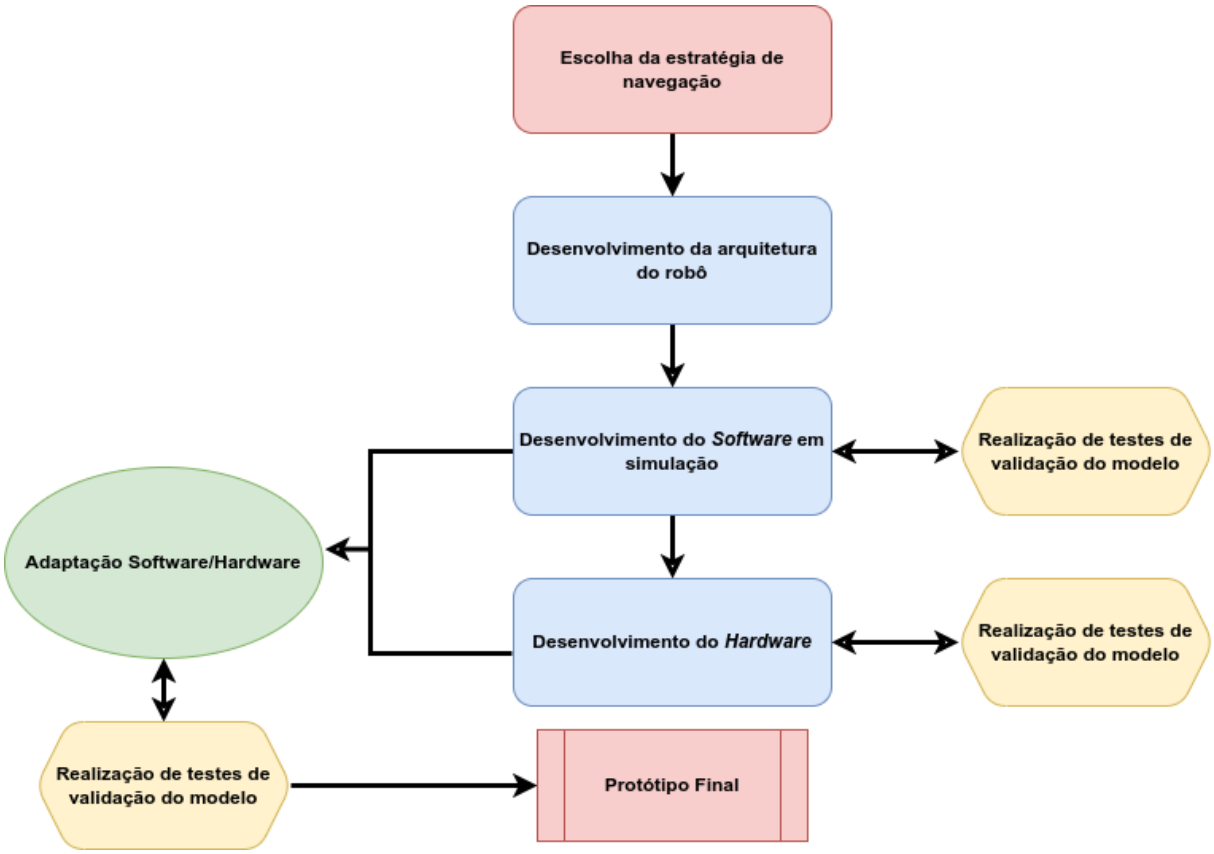
Figura 13 – Estratégia de Navegação.



Fonte: *Adaptado de:* FREITAS, 2017.

Dessa maneira, a Figura 14 apresenta um fluxo de etapas a serem seguidas para o desenvolvimento do trabalho. O trabalho foi desenvolvido em duas partes, sendo elas a parte de *hardware* e a parte de *software*. Na parte de *hardware* foi realizada a adaptação do veículo em escala reduzida e feito a estruturação das partes de eletrônica realizando a instalação dos dispositivos que integraram o robô. Na parte de *software* foi desenvolvido todo as partes de configuração do ROS e *driver* de sensores, microcomputadores e entre outros para que fosse possível fazer com que o robô tivesse todas as funções necessárias. As seções desses capítulos, apresentados seguir, estão organizadas de forma a descrever em detalhes como foi elaborado cada uma dessas partes.

Figura 14 – Fluxograma de etapas utilizado para o desenvolvimento do protótipo do robô.



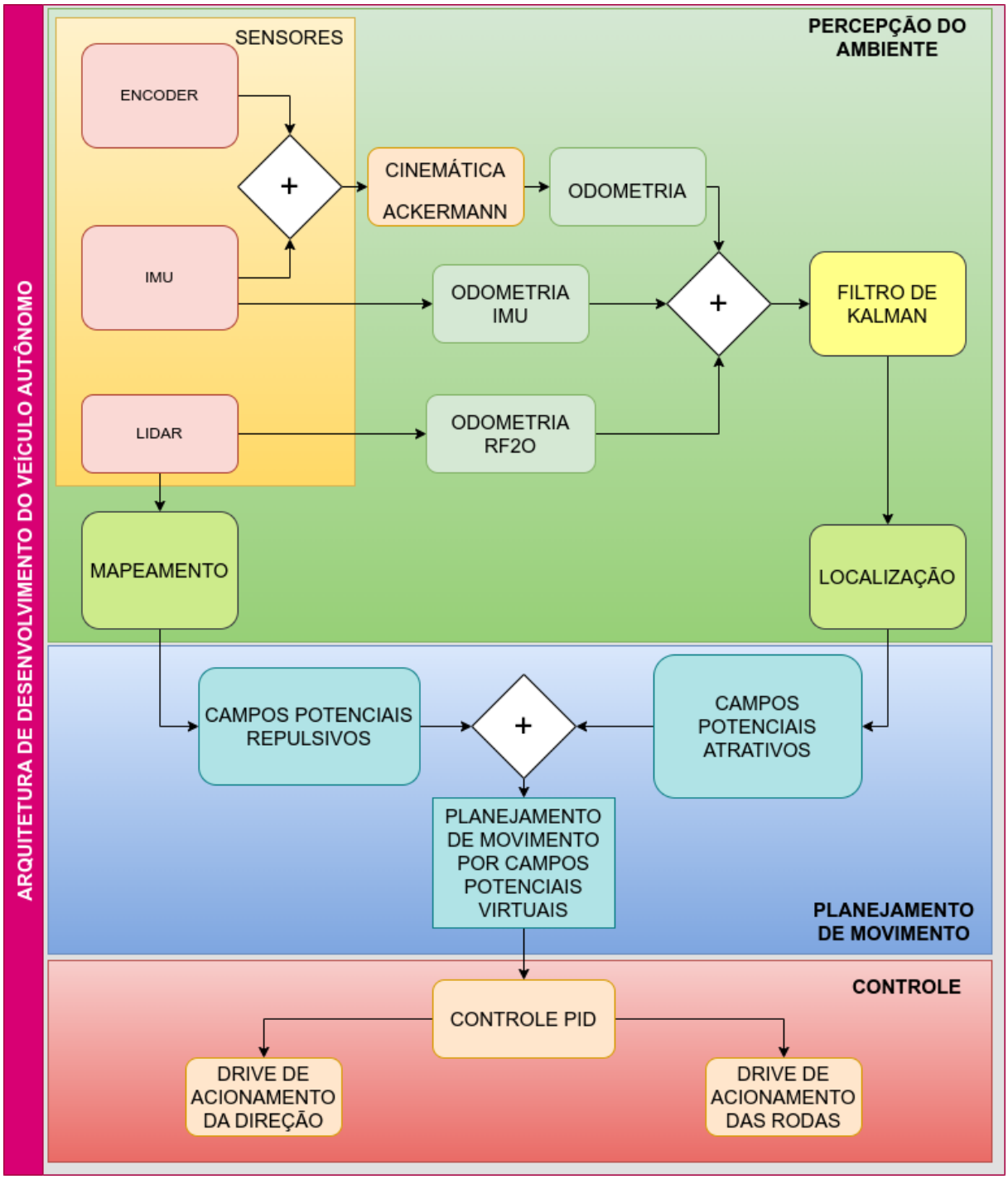
Fonte: Elaborado pelo autor, 2022.

3.2 *Desenvolvimento da Arquitetura do Robô*

A arquitetura do robô é projetada para o desenvolvimento de um planejador de movimento tendo como base o conceito de campus potenciais virtuais. A Figura 15 apresenta em detalhes como está configurada a arquitetura desenvolvida.

A arquitetura do robô é agrupada em três partes, sendo elas: percepção do ambiente, planejamento de movimento e controle. Na percepção do ambiente, os sensores LIDAR, IMU e Encoder são utilizados para determinar o mapeamento e a localização. A principal forma de se determinar a posição do veículo é com a utilização do *Encoder* e a IMU, o *Encoder* realiza a medição da velocidade da roda do veículo, enquanto a IMU realiza a medição de sua orientação, assim, esses valores são aplicados ao modelo da cinemática de *Ackermann* para se determinar a posição. Além disso, o sensor IMU também consegue fornecer os valores de odometria por meio da integração dos seus valores. Outro método utilizado para se obter a odometria é através do sensor LIDAR que por meio da metodologia RF2o consegue estimar a posição do veículo conforme poderá ser visto com mais detalhes nesta Seção do trabalho. Essas posições então são ajustadas por um filtro de Kalman que retorna a localização do veículo, além disso, o sensor LIDAR irá retornar uma nuvem de pontos do local fazendo o mapeamento local instantâneo em duas dimensões ao redor do veículo.

Figura 15 – Arquitetura Desenvolvida.



Fonte: Elaborado pelo autor, 2022.

O planejamento de movimento é então realizado por meio de campos potenciais virtuais que necessita da informação da localização e do mapeamento. Os dados fornecidos pelo mapeamento local determinam quais obstáculos existem ao redor do veículo, assim, é projetado um campo potencial repulsivo, inversamente proporcional à distância desses obstáculos. Já a localização fornece os dados necessários para se construir o planejador de campos potenciais atrativos, que será proporcional à distância geométrica entre a localização atual e o destino a que

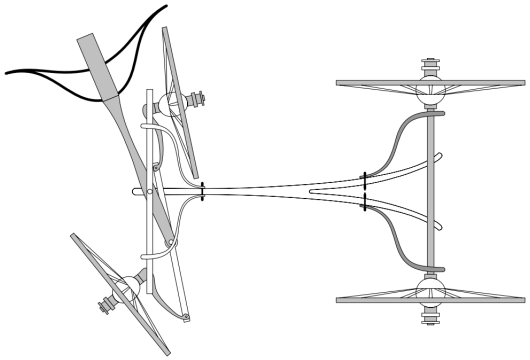
se deseja chegar. O somatório desses campos potenciais retornam um vetor direção e intensidade para o veículo, indicando qual caminho ele deve percorrer.

O vetor fornecido pelo planejamento é utilizado para realização do controle do acionamento do automodelo, assim, o veículo terá uma direção e uma intensidade no qual deverá navegar, então, esses dados são utilizados para realização de um controle PWM que irá fazer o acionamento do *driver* de direção do veículo e do acionamento das rodas para sua movimentação.

3.2.1 *Percepção do ambiente*

Os sensores *Encoder* e IMU são responsáveis por fornecer a principal referência de odometria do veículo, entretanto, essa medida sozinha apresenta erro acumulativo. Os principais dados de odometria são a velocidade atual do veículo e o ângulo em relação ao eixo z, conhecido como *yaw*. Então, esses valores são utilizados no cálculo da cinemática de Ackermann para determinação da posição atual do robô. Veículos de cinemática Ackermann, possuem quatro rodas, sendo duas dianteiras móveis e duas traseiras fixas. Tal modelo de veículo foi introduzido por *George Langensperger* em 1816, sendo *Rudolf Ackermann*, agente patenteador, ao qual é dado nome ao modelo cinemático desses veículos (JAZAR, 2008), modelo da Figura 16.

Figura 16 – Modelo de veículo introduzido por *George Langensperger*.



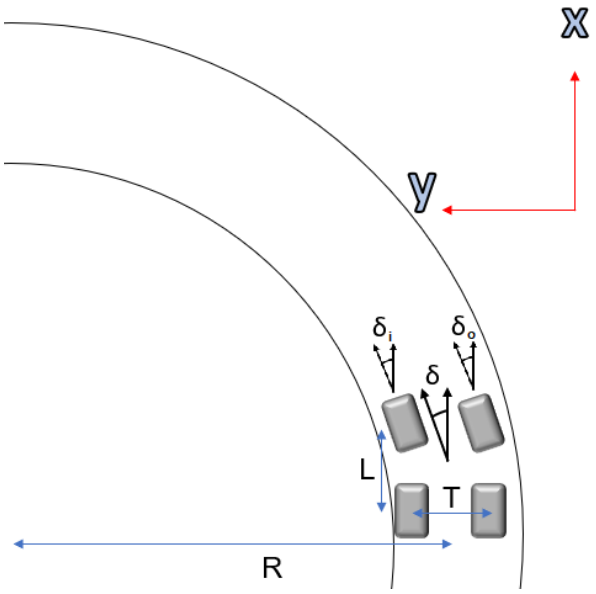
Fonte: JAZAR, 2008.

Uma característica importante desse modelo é a definição de que a roda interna ao raio da curva do veículo deveria sempre ter uma inclinação maior do que a curva de raio externo ao veículo, possibilitando assim a rotação do veículo. Para tanto, o modelo possui um mecanismo trapezoidal nas rodas dianteiras que possibilitam tal movimento.

Dessa maneira, o modelo cinemático de *Ackermann* é representado em detalhadas pela Figura 17.

O modelo matemático pode ser simplificado em condições ideais que descreve a direção do veículo descrita por:

Figura 17 – Modelo Cinemático Curvatura.



Fonte: Elaborado pelo autor, 2022

$$\delta_i = tan^{-1}\left(\frac{L}{R-\frac{T}{2}}\right) \tag{3.1a}$$

$$\delta_o = tan^{-1}\left(\frac{L}{R+\frac{T}{2}}\right) \tag{3.1b}$$

Tal modelo pode ainda ser simplificado ao trazer as rodas para o meio do veículo, assim:

$$\delta = tan^{-1}\left(\frac{L}{R}\right) \tag{3.2}$$

Dessa maneira, é possível obter um modelo matemático para esses veículos de forma que seja possível determinar a sua localização espacial. Tal posição é determinada por equações que utilizam a velocidade linear do robô em relação a sua parte frontal e a velocidade angular produzida pelas rodas. Para tal modelagem, deve-se considerar as posições do robô em um eixo (X,Y) e posição angular como ψ assim como é possível ver pela Figura 18.

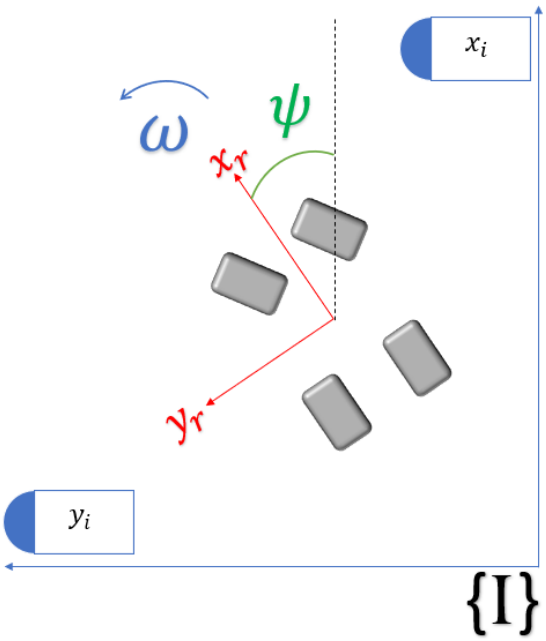
Assim, as equações que determinam as velocidades do veículo $(\vec{v_x},\vec{v_y})$ em relação ao eixo inicial x_i e y_i e a velocidade angular ω, são descritas por:

$$\vec{v_x} = v\cos\left(\frac{\pi}{2}-\omega\right), \tag{3.3a}$$

$$\vec{v_y} = v\sin\left(\frac{\pi}{2}-\omega\right), \tag{3.3b}$$

$$\omega = \frac{v}{L}\tan(\delta), \tag{3.3c}$$

Figura 18 – Sistema de coordenadas utilizados do robô em relação ao mapa. Em vermelho é apresentado o referencial do robô, já em azul o referencial inercial.



Fonte: Elaborado pelo autor, 2022

onde v é a velocidade linear do veículo em relação a sua própria referência x_r e L é a largura entre as rodas traseira e dianteira do robô.

Para identificar as variáveis de posição e orientação do robô pode-se integrar os dados de velocidade:

$$x = \int_0^t \vec{v}_x dt, \tag{3.4a}$$

$$y = \int_0^t \vec{v}_y dt, \tag{3.4b}$$

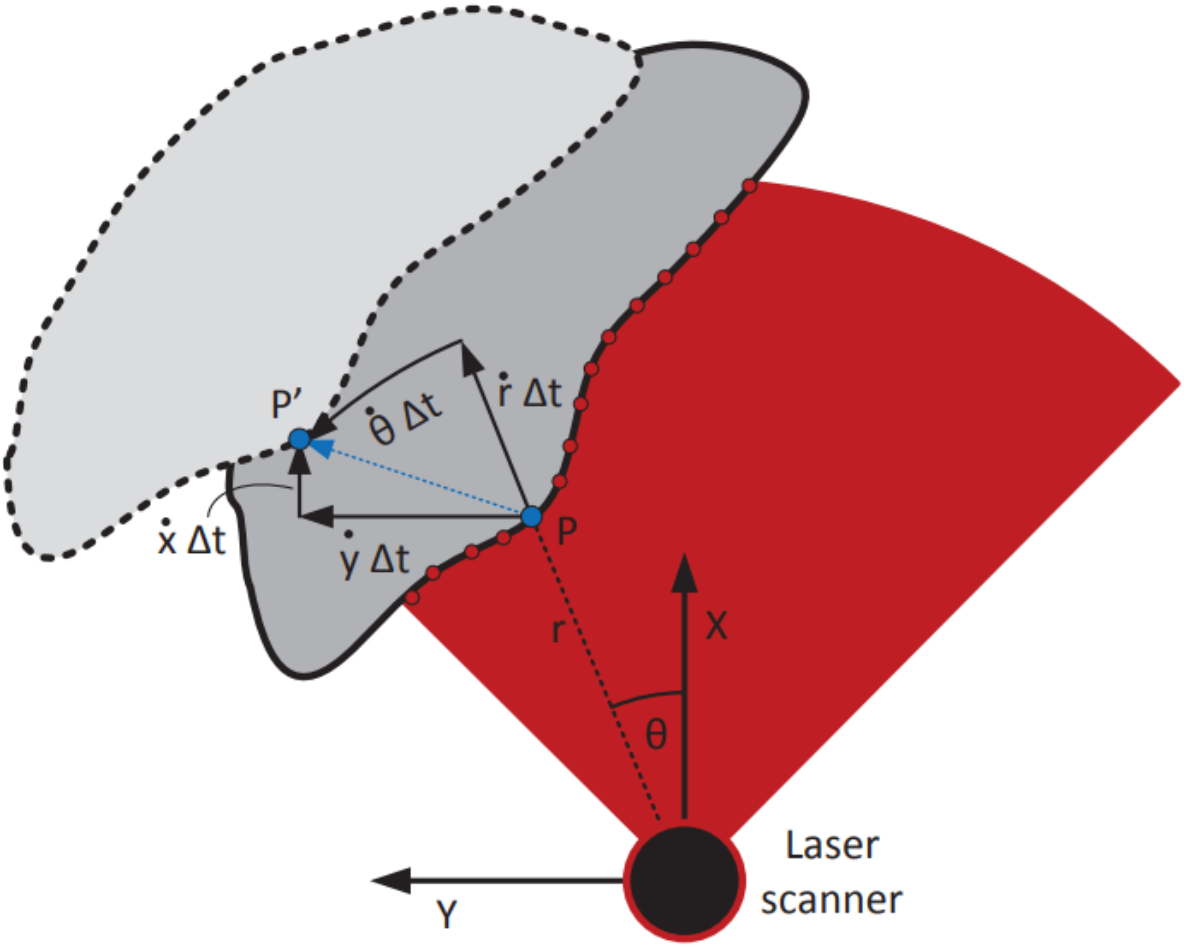
$$\psi = \int_0^t \omega dt, \tag{3.4c}$$

onde $[x, y]^T$ é o vetor posição do robô no espaço 2D e ψ é a orientação em relação ao eixo z.

Os dados de odometria também podem ser obtidos pelo sensor LIDAR por meio da técnica de odometria RF2o (*Range Flow-based 2D Odometry*) descrito em Jaimez, Monroy e Gonzalez-Jimenez (2016). Neste trabalho é apresentado um método rápido e preciso para estimar o movimento por meio do escaneamento do sensor LIDAR. A técnica apresentada consiste em calcular o movimento aparente do LIDAR com relação a variação de cada uma das distância medidas correspondente a cada feixe do sensor.

Como pode ser visto na Figura 19, o modelo compara a posição de um ponto P medido por um dos feixes do LIDAR com a nova posição medida em um ponto P' e calcula a equidistância

Figura 19 – Solução RF2o (*Range Flow-based 2D Odometry*). A área em vermelho é a representação da medição feita pelo LIDAR, em cinza escuro está representado o obstáculo e em cinza claro o mesmo obstáculo após um determinado movimento do robô.



Fonte: JAIMEZ; MONROY; GONZALEZ-JIMENEZ, 2016.

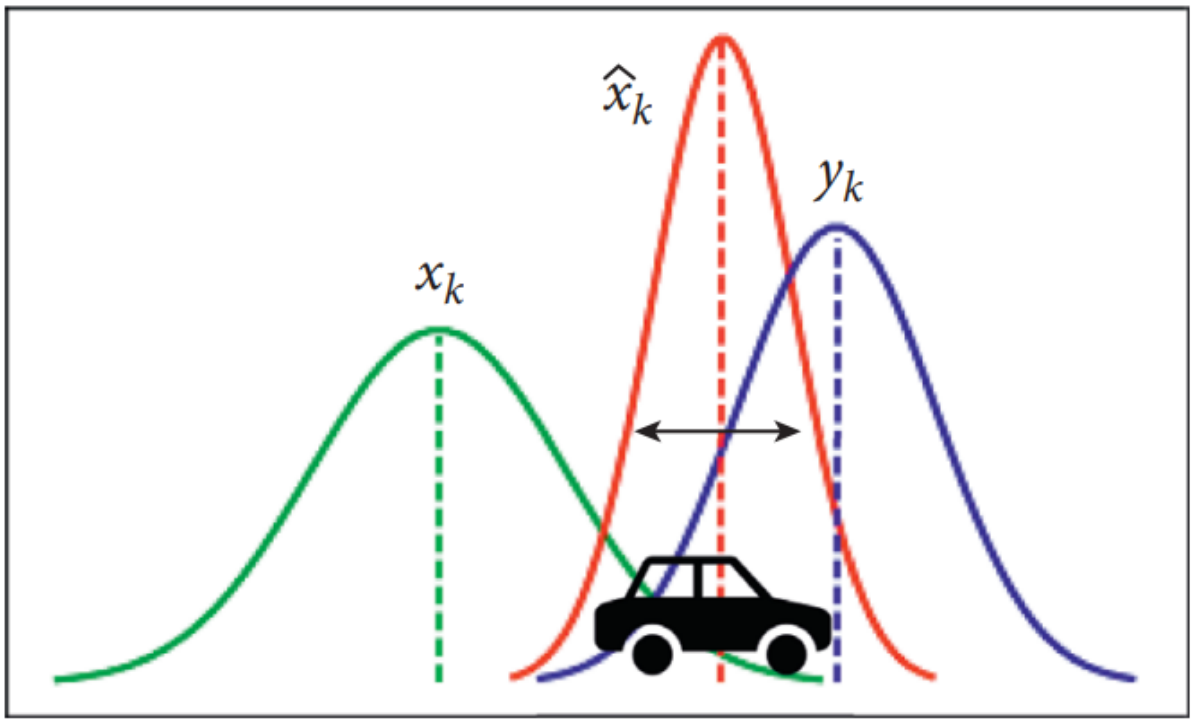
entre esses pontos para se determinar a posição do veículo. Esse modelo tem bons resultados em locais onde é possível detectar variações no ambiente, porém o erro é acumulado em ambientes pouco estruturados, como corredores.

Ainda, os dados da odometria podem ser obtidos pela integração dos dados fornecidos pela IMU, esses, porém, possuem também muito ruído de medição, na medida em que, os valores da integração apresentam erro acumulativo ao longo do tempo.

Para resolver as incertezas dessas medições, é, então, utilizado um filtro de Kalman que realiza o agrupamento desses sinais, extraindo um resultado que tenderá a se ajustar o melhor possível. A combinação das variâncias de cada resultado medido é combinada fazendo que com a variação total se reduza, no qual, tipicamente, a cada sinal medido é aplicado uma função de densidade do tipo Gaussiana, como visto na Figura 20, em que $\hat{X}_k$ é a posição estimada pelo Filtro de Kalman, X_k é a posição calculada por meio de um dos sensores e Y_k é a posição calculada por

meio de outro sensor.

Figura 20 – Princípio de funcionamento do Filtro de Kalman.



Fonte: CHEN *et al.*, 2021.

O mapeamento do veículo é realizado por meio do sensor LIDAR, a nuvem de pontos fornecida pelo sensor, tal como apresentado na Figura 9 pode fornecer uma informação instantânea em duas dimensões de todo o ambiente ao redor do veículo, dessa maneira, permitindo que o robô incremente no seu planejamento formas de solucionar problemas de desvio de obstáculo. É importante destacar, que essa informação não é salva na memória e como não é conhecido um mapa previamente, a resposta com relação ao mapeamento gerado também será dinâmica. Isso reduz consideravelmente o cálculo computacional, uma vez que, não é necessário trabalhar com conjuntos de dados armazenados em matriz, permitindo, assim, a realização de um planejamento reativo ao ambiente que o robô se encontra.

3.2.2 *Planejamento de Movimento*

O planejamento de movimento escolhido para este trabalho é baseado em campos vetoriais determinados a partir do mapeamento instantâneo. A ideia principal dessa solução é a criação de campos potenciais virtuais atrativos e repulsivos por meio de modelos matemáticos. Dessa forma, o algoritmo irá calcular campos de força atrativos para se chegar ao ponto de interesse, e campos repulsivos em locais que representam obstáculos para o carro, garantindo assim, que o robô seja repelido antes de uma colisão.

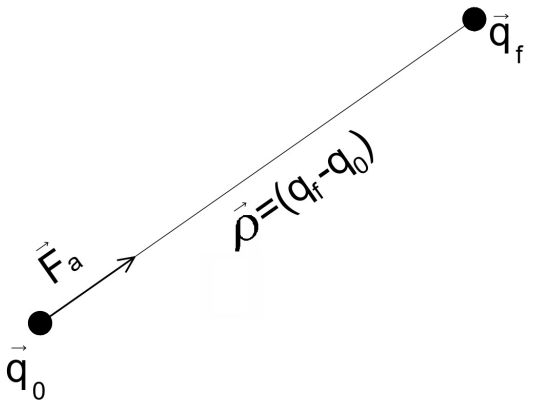
Pode-se definir assim, que a força virtual aplicada ao veículo será a somatória de contribuições de forças geradas pelos campos atrativos e repulsivos modelados, resultando para o robô a direção e o sentido pelo qual o robô deve seguir, assim, a equação 3.5 descreve essa força.

$$\vec{F}_t = \sum_{n=1}^{n} (\vec{F}_a + \vec{F}_r), \tag{3.5}$$

onde $\vec{F}_t$ é o vetor de força resultante, $\vec{F}_a$ o vetor de força atrativa e $\vec{F}_r$ o vetor de força repulsiva no robô.

O vetor de campo potencial atrativo é definido pela diferença entre a posição atual do veículo e a distância de interesse. Assim, para definir o ponto de interesse, considerando q_0, o ponto atual e q_f o ponto de interesse, pode-se considerar o vetor força atrativa, conforme Figura 21.

Figura 21 – Força atrativa determinada pela distância calculada entre a posição atual e a posição de interesse.



Fonte: Elaborado pelo autor, 2022

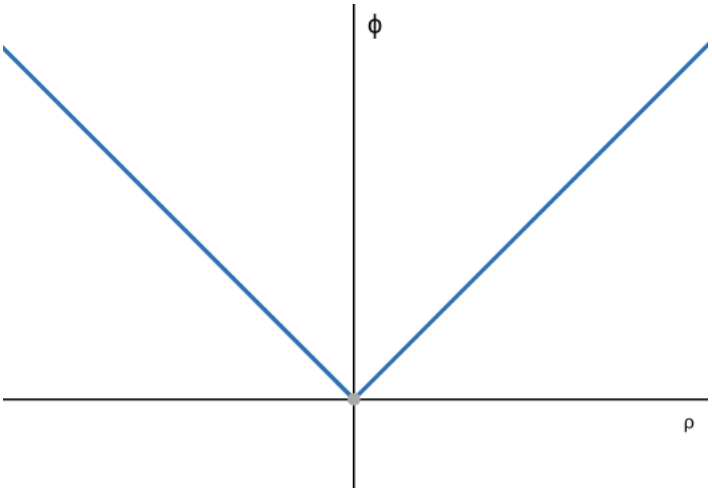
Ao se aplicar um campo potencial atrativo ϕ com característica de função modular linear é obtido o gráfico representado na Figura 22.

Esse campo é descrito pela Equação 3.6:

$$\phi_a = k|\vec{\rho}|, \tag{3.6}$$

no qual o valor de k é uma constante diretamente proporcional a velocidade do veículo. Sendo assim, a força atrativa pode ser calculada por meio do gradiente desse campo, sendo descrita como:

Figura 22 – Campo de função modular linear em que ϕ representa o campo e ρ a distância em relação ao ponto de interesse.



Fonte: Elaborado pelo autor, 2022

$$\vec{F_a} = \nabla \phi_a, \tag{3.7}$$

resolvendo essa equação:

$$\begin{aligned} \vec{F_a} &= k\nabla(|\vec{\rho}|) \\ &= k \begin{bmatrix} \frac{d}{d_x}\left(x^2+y^2\right)^{\frac{1}{2}} \\ \frac{d}{d_y}\left(x^2+y^2\right)^{\frac{1}{2}} \end{bmatrix} \\ &= k \left[\frac{x}{|\rho|}, \frac{y}{|\rho|}\right]^T \\ &= k\frac{\vec{\rho}}{|\rho|} \end{aligned} \tag{3.8}$$

em que k é a constante de ajuste e $\vec{p}$ é o vetor de equidistância entre os pontos.

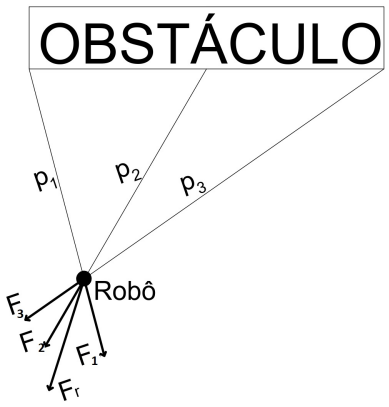
A força vetorial do campo repulsivo será obtida pelo somatório das forças repulsivas de cada um dos pontos fornecidos pelos obstáculos ao redor do robô, conforme é visto na Figura 23.

O campo vetorial repulsivo é calculado pelos dados obtidos diretamente do sensor LIDAR sendo que para cada feixe do sensor é obtido um campo vetorial definido por:

$$\phi_i = \eta \left[\frac{1}{\rho_i} - \frac{1}{d_0}\right]^2 \tag{3.9}$$

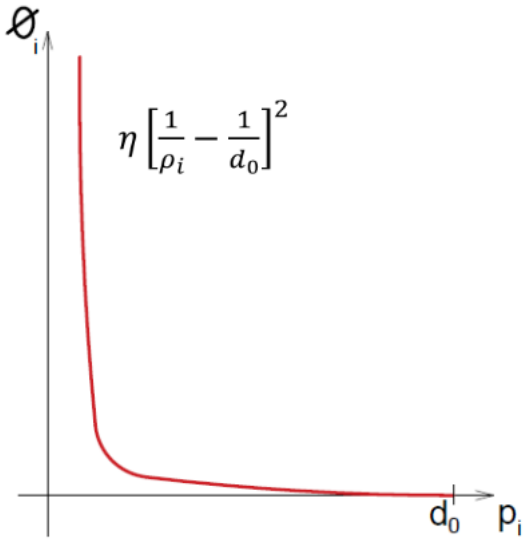
onde ρ_i é a distância euclidiana entre o robô e o obstáculo medido pelo feixe do sensor LIDAR, d_0 é a distância mínima para o qual o campo começa a ter efeito e η a constante de proporcionalidade, conforme ilustrado na Figura 24.

Figura 23 – Forças repulsivas decorrentes de obstáculos.



Fonte: Elaborado pelo autor, 2022.

Figura 24 – Representação do campo repulsivo aplicado ao robô. A função do campo possui uma curva acentuada que tem como objetivo a realização de uma barreira de proteção contra o obstáculo.



Fonte: Elaborado pelo autor, 2022.

A força repulsiva também será calculada pelo gradiente desse campo em cada um dos feixes do sensor LIDAR, conforme Equação 3.10.

$$\vec{F_i} = \nabla \phi_i, \tag{3.10}$$

ao resolver essa equação tem-se que:

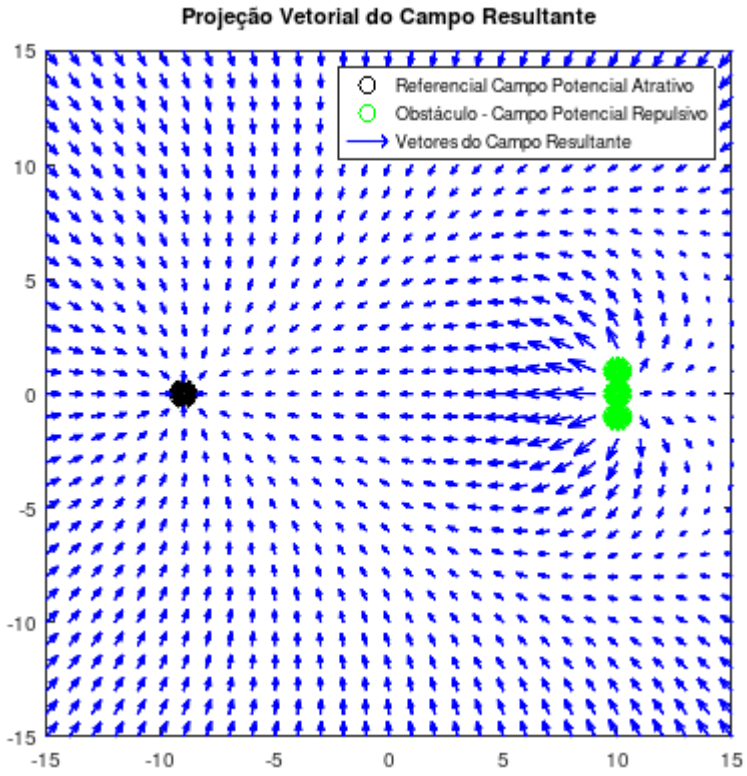
$$\vec{F_i} = \eta \left[\frac{2x(d_0 - \rho_i)}{d_0 {\rho_i}^2}, \frac{2y(d_0 - \rho_i)}{d_0 {\rho_i}^2} \right]^T \tag{3.11}$$

Com isso, pode-se determinar o valor total da força repulsiva com o somatório das forças de cada um dos feixes:

$$\vec{F}_r = \sum_{n=1}^{n} \vec{F}_i \tag{3.12}$$

Dessa maneira, aplicando esse somatório de forças relativas aos campos potenciais é possível obter um referencial de movimento para o robô, conforme o campo vetorial apresentado na Figura 25.

Figura 25 – Representação do campo resultante. As setas em azul representação a intensidade e a direção no qual o veículo deve seguir em cada um dos pontos, o ponto preto representa o ponto de interesse de chegada e os pontos em verde o obstáculo.



Fonte: Elaborado pelo autor, 2022.

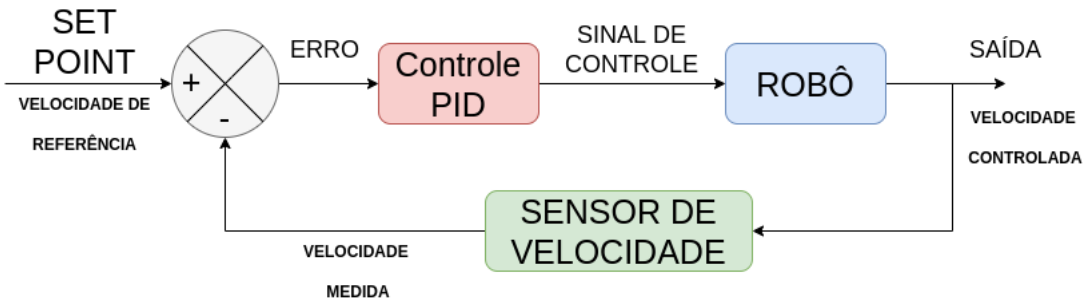
3.2.3 *Arquitetura de Controle*

O planejamento de movimento fornece os dados de *set point* necessários para realização da correta atuação dos dispositivos de locomoção do robô. Esses dados são utilizados para fazer o controle de direção e velocidade do automodelo. Assim, para cada um deles foi arquitetado um controle do tipo PID (Proporcional, Integral e Derivativo).

O controle de velocidade está diretamente ligado ao movimento de rotação das rodas do automodelo. Assim, o sensor *Encoder* é utilizado como *feedback* de sinal para o fechamento da

malha de controle. Esse sensor retorna a velocidade a partir do número de pulsos em determinado tempo. Assim, a Figura 26 apresenta malha de controle de velocidade, sendo que o erro é a diferença entre o sinal medido e o *set point* e o controle PID enviará um sinal para minimizar esse erro no robô.

Figura 26 – Arquitetura do controle de velocidade.

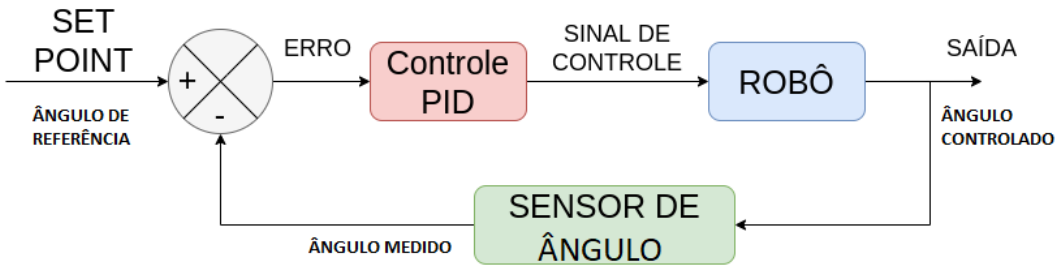


Fonte: Elaborado pelo autor, 2022.

A mesma arquitetura de controle é utilizada para o controle da direção do veículo, entretanto, a variável controlada não é a velocidade mas o ângulo das rodas, conforme Figura 27. Sendo que, existe integrado ao motor das rodas um potenciômetro em que é possível realizar a medição do ângulo atual.

O esquema de controle ficará:

Figura 27 – Arquitetura do controle de direção.



Fonte: Elaborado pelo autor, 2022.

Um controlador PID, descrito pela Equação 3.13, possui três constantes de ajuste, K_p a constante proporcional do controlador que realiza o ajuste conforme o erro proporcional do sistema, K_i a constante integrativa do controlador que realiza o ajuste conforme o erro acumulado do sistema e K_d a constante derivativa do controlador que realiza o ajuste conforme a taxa de variação do erro do sistema, e também, $e(t)$ que representa o erro no domínio do tempo e $u(t)$ que representa o sinal de controle.

$$u(t) = K_p e(t) + K_i \int e(t)\,dt + K_d \frac{de(t)}{dt} \tag{3.13}$$

3.3 *Desenvolvimento do software*

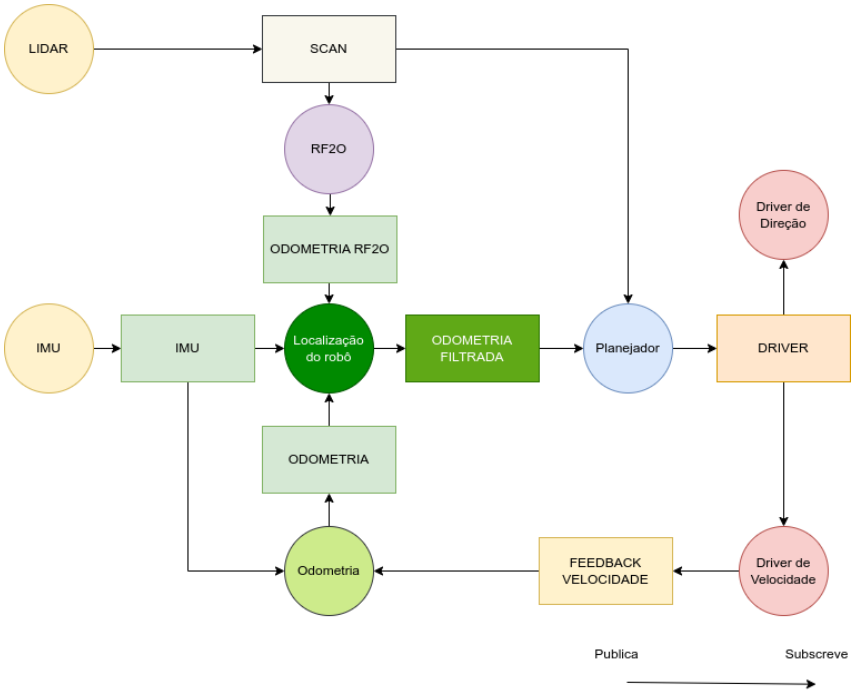
É ideal desenvolver todo *software* do robô em uma simulação, no qual seu robô será virtual, e assim, é possível construir toda a estrutura do *software* para o robô e realizar testes prévios para sua validação.

O ambiente de simulação da F1TENTH é instalado para possibilitar o desenvolvimento de toda estrutura de *software*. Esse simulador utiliza o ROS como estrutura de criação e, portanto, todo esse sistema deve ser instalado dentro de um sistema operacional Linux.

3.3.1 Estrutura de *Software*

O ambiente de simulação da F1TENTH possui todos os recursos necessários para realização dos testes do modelo proposto para navegação do robô. Entretanto, é importante destacar que é necessário realizar os ajustes corretos para que o modelo se adapte ao que está sendo proposto.

Figura 28 – Estrutura de nós e tópicos desenvolvidos.



Fonte: Elaborado pelo autor, 2022.

Como já definido na Seção 2.2, o sistema de comunicação do ROS é baseado em nós que publicam e subscrevem informações de tópicos. Assim, essa estrutura deve ser definida para o adequado funcionamento do robô. A Figura 28 descreve a estrutura desenvolvida em *software*, os círculos representam os nós, enquanto retângulos os tópicos, no qual, cada um deles é descrito no Quadro 1. De forma resumida, o esquema segue a arquitetura desenvolvida na Seção anterior. Em amarelo estão os sensores do sistema, na cor verde está a parte da estrutura responsável pela

Quadro 1 – Componentes de comunicação do ROS.

Nome	**Tipo**	**Função**
LIDAR	Nó	Publica os dados do sensor LIDAR.
SCAN	Tópico	Armazena dados do sensor LIDAR.
IMU	Nó	Publica os dados da IMU.
IMU	Tópico	Armazena dados da IMU.
Odometria	Nó	Subscreve dados do ângulo da IMU e velocidade do robô para calcular a odometria pelo modelo Ackermann.
Odometria	Tópico	Armazena a informação da odometria do veículo.
RF20	Nó	Subscreve a informação de mapeamento fornecida pelo LIDAR para calcular a odometria e publica esse valor.
ODOMETRIA RF2O	Tópico	Armazena a informação da odometria por RF2O.
Localização do Robô	Nó	Subscreve as informações de odometria dos tópicos da IMU, ODOMETRIA e ODOMETRIA RF2O para aplicar filtro de Kalman e publicar a Odometria Filtrada.
Odometria Filtrada	Tópico	Armazena os dados da odometria filtrada.
Planejador	Nó	Subscreve as informações da odometria filtrada e do escaneamento do LIDAR para realizar cálculo do planejamento de navegação publicando um sinal de controle para o *driver*.
Driver	Tópico	Armazena a informação de setpoint de velocidade e direção fornecidas pelo planejador.
***Driver* de Direção**	Nó	Realiza o controle de direção do veículo subscrevendo a informação da direção desejado do *driver*.
***Driver* de Velocidade**	Nó	Realiza o controle de velocidade do veículo subscrevendo a informação da direção desejada do *driver* e publica a informação de velocidade medida.
FEEDBACK	Tópico	Armazena a informação de velocidade do veículo.

realização da odometria, em azul o planejador de navegação e por último em vermelho tem-se a parte de *driver* de acionamento do veículo.

3.3.2 Implementação do *software*

Um conjunto de códigos foi desenvolvido para o desenvolvimento do robô, bem como, alguns códigos amplamente conhecidos e divulgados por outros autores foram utilizados por

meio de pacotes do ROS. Dentre os algoritmos desenvolvidos neste trabalho, foi desenvolvido códigos para o *driver* e para a odometria do veículo, o código do planejador de movimento, sendo também utilizado códigos desenvolvidos por outros autores, conforme apresentado no Quadro 2.

Quadro 2 – Algoritmos utilizados na estrutura de *software* do ROS.

Nome	**Autor**	**Fonte do pacote**
YDLidarSDK	Shenzhen Co.,Ltd.	https://github.com/YDLIDAR/sdk
YDLidarROS	Shenzhen Co.,Ltd.	https://github.com/YDLIDAR/ydlidar_ros_driver
IMU	Dheera Venkatraman	https://github.com/dheera/ros-imu-bno055
Odometreia RF2o	Javier Monroy	https://github.com/MAPIRlab/rf2o_laser_odometry
Localização do Robô	Charles River Inc.	https://github.com/cra-ros-pkg/robot_localization
Odometria	Próprio	https://github.com/vidigaleandro/F1bot
***Driver* Velocidade**	Próprio	https://github.com/vidigaleandro/F1bot
***Driver* Direção**	Próprio	https://github.com/vidigaleandro/F1bot
Planejador	Próprio	https://github.com/vidigaleandro/F1bot

Na sequência, são apresentados pseudo-códigos para os códigos desenvolvidos. O primeiro a ser apresentado é o código da odometria.

Algoritmo 1: Odometria

Entrada: v = velocidade medida; ψ = direção medida; $X_{inicial}$=0; $Y_{inicial}$=0;

Saída: $X_{calculado}$;$Y_{calculado}$;

enquanto *Ativo* **faça**

    $\vec{v}_x$ = CalculaVelocidadeX(v,ψ) ; // Equação 3.3a

    $\vec{v}_y$ = CalculaVelocidadeY(v,ψ) ; // Equação 3.3b

    x = Integra($\vec{v}_x$, X) ; // Equação 3.4a

    y = Integra($\vec{v}_y$, Y) ; // Equação 3.4b

fim

No Algoritmo 1 tem-se como entrada os valores da velocidade medida pelo sensor *Encoder* e ângulo *yaw*(ψ) medido pela IMU e valores iniciais desejados para posição do veículo, obtendo, como saída os valores das posições x e y calculados. O código segue em um *loop* que calcula a velocidade do robô em relação ao eixo X e ao eixo Y, então essas velocidades são integradas e são enviadas como saída do código.

Os algoritmos de controle de velocidade e direção são construídos de forma semelhante, entretanto, em um dos casos será avaliado a velocidade de movimento enquanto o outro a posição de direção, uma vez que o primeiro recebe um sinal pulsante de acordo com a velocidade e o outro recebe um valor proporcional a posição das rodas.

Algoritmo 2: *Driver* de Velocidade

Entrada: E_k = Tempo de pulso *Encoder*; $v_{desejado}$ = velocidade desejado; K_p=ganho proporcional; K_i=ganho integral;K_d=ganho derivativo;$f = FrequenciaPWM$;$k = constante\_de\_proporcionalidade\_da\_velocidade$;

Saída: v_{medida};

enquanto *Ativo* **faça**

- $v_{medida} = k/(E_k - E_{k-1})$;
- $erro = v_{desejado}$ - v_{medida};
- $u = CalculaPID(erro, K_p, K_i, K_d)$; `// Equação 3.13`
- $Mpwm = ConfigPWM(u, f)$;
- $E_{k-1} = E_k$

fim

No Algoritmo 2, o cálculo da velocidade é realizado por meio do inverso da diferença de tempo entre os pulsos fornecidos pelo *Encoder* multiplicado por uma constante de proporcionalidade k que é utilizado como ajuste de calibração da velocidade. Então, é calculado o erro entre o valor de velocidade desejado e o valor medido para então se determinar o cálculo da variável de controle que é utilizada como saída para acionamento do motor. No Algoritmo 2 e Algoritmo 3 a função ConfigPWM é utilizada para configurar a saída pwm da porta que enviará o sinal, em que u é o valor dutycicle do PWM e f é a frequência do PWM.

Algoritmo 3: *Driver* de direção

Entrada: p_{medida} = Posição da direção; $p_{desejado}$ = Posição desejado; K_p=ganho proporcional; K_i=ganho integral;K_d=ganho derivativo;$f = FrequenciaPWM$;$k = constante\_de\_proporcionalidade\_da\_posicao$;

Saída: v_{medida};

enquanto *Ativo* **faça**

- $p_{atual} = kp_{medida}$;
- $erro = p_{desejado}$ - p_{medida};
- $u = CalculaPID(erro, K_p, K_i, K_d)$; `// Equação 3.13`
- $Mpwm = ConfigPWM(u, f)$;

fim

Por último, é apresentado o Algoritmo 4 desenvolvido para realização do planejamento de movimento do veículo.

Algoritmo 4: Planejamento de Movimento

Entrada: X_{atual} = Posição atual em relação a X; Y_{atual} = Posição atual em relação a Y;$X_{desejado}$ = Posição desejado em relação a X; $Y_{desejado}$ = Posição desejado em relação a Y; ψ=ângulo em relação ao mapa; K_a=Ganho força atrativa;η=Ganho força repulsiva;d_0 = Distância mínima de atuação do campo repulsivo;ρ_i = Distância euclidiana robô/obstáculo;v_{max} = Velocidade Máxima limitada para o veículo;v_{min} = Velocidade Mínima limitada para o veículo; δ_{min} = ângulo mínimo rodas do veículo; δ_{max} = ângulo mínimo rodas do veículo

Saída: $v_{desejado}$;$p_{desejado}$

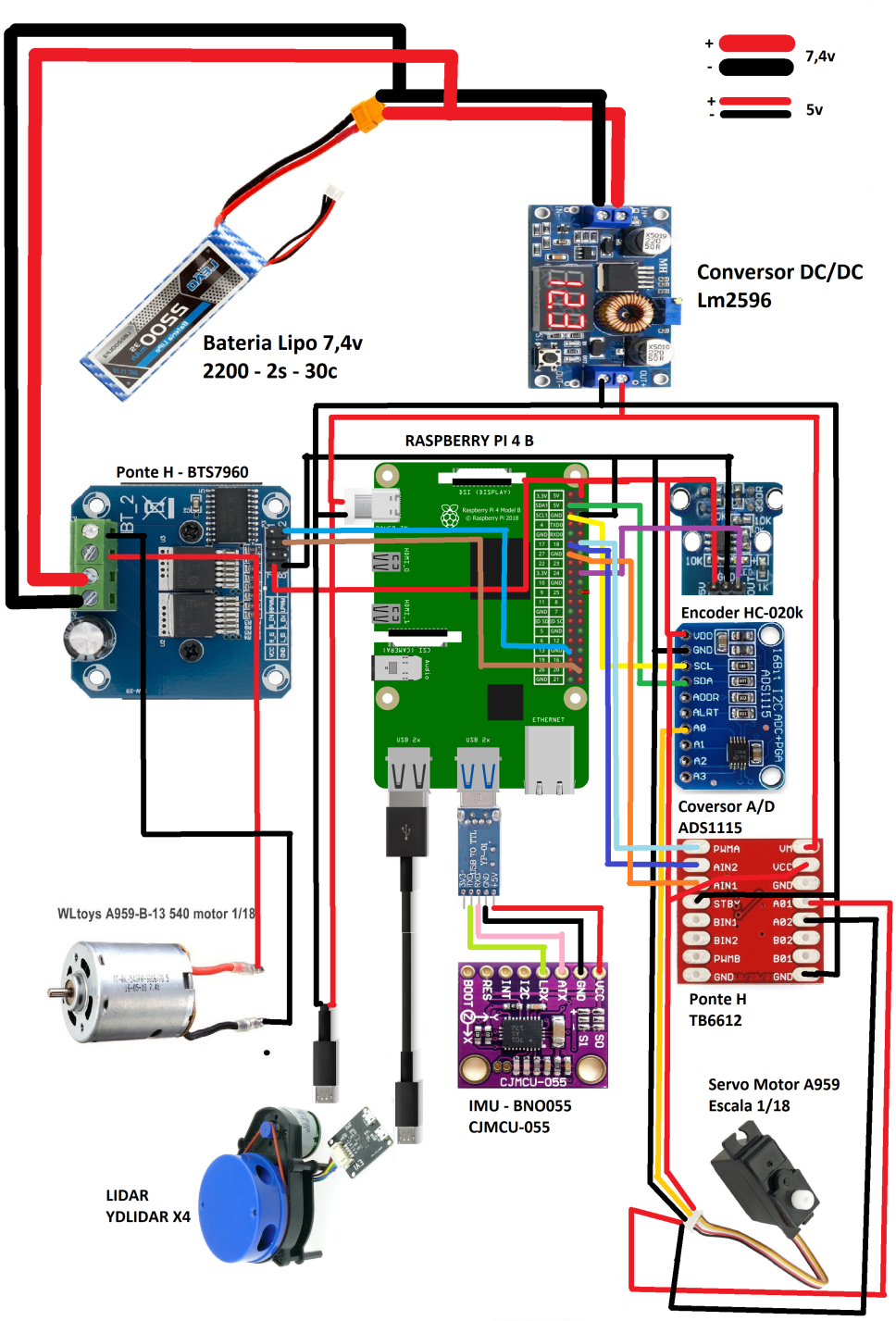
1 **enquanto** *Ativo* **faça**
2 $\vec{F_r} = CalculaFRepulsiva(\rho_i, \eta, d_0)$; // Equação 3.12
3 $\vec{F_a} = CalculaFAtrativa(X_{atual}, Y_{atual}, X_{desejado}, Y_{desejado})$; // Equação 3.8
4 $\vec{F_t} = CalculaFResultate(\vec{F_a}, \vec{F_r})$; // Equação 3.5
5 $p_{desejado} = arctg(\vec{F_t}) - \psi$;
6 **se** $p_{desejado} > \delta_{max}$ **então**
7 $p_{desejado} = \delta_{max}$
8 **fim**
9 **se** $p_{desejado} < -\delta_{max}$ **então**
10 $p_{desejado} = -\delta_{max}$
11 **fim**
12 $v_{desejado} = v_{min} + v_{max}((\delta_{max} - p_{desejado})/\delta_{max})$;
13 **fim**

O planejamento é baseado em campos potenciais onde uma força virtual resultante é utilizada para se calcular a direção do veículo. Na linha 5 do pseudo-código é realizado o cálculo da posição que deve ser enviada para a roda do veículo por meio da diferença entre a direção indicada e a direção atual. Entretanto, esse valor é logo limitado, uma vez que o ângulo desejado para as rodas não pode ser maior que o ângulo máximo que ele pode se movimentar. Por fim, a velocidade do veículo é configurada de forma proporcional ao ângulo desejado para as rodas, quanto maior o ângulo configurado, menor será a velocidade e automaticamente a velocidade do robô é reduzida durante as curvas.

3.4 Desenvolvimento do *Hardware*

Nesta seção serão apresentados os conjuntos de dispositivos de *hardware*, bem como a instalação de cada um deles para o desenvolvimento do robô. Para tanto, a Figura 29 apresenta a estrutura de *hardware* que compõe o conjunto de ligações e componentes utilizados para construção do veículo.

Figura 29 – Estrutura do *hardware* desenvolvido.



Fonte: Elaborado pelo autor, 2022.

3.4.1 Construção do robô

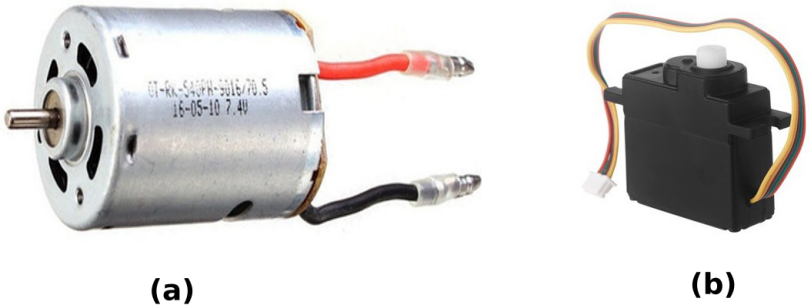
O robô foi desenvolvido tendo como base um chassi de um automodelo Rádio Controlado (RC), que basicamente são carros de escala reduzida controlados por ondas de rádio. Foi selecionado o modelo Storm Monster Truck com um tamanho em escala reduzida na proporção de 1/18 apresentado na Figura 30.

Figura 30 – Chassi Storm Monster Truck.

Fonte: Elaborado pelo autor, 2022.

Esse chassi possui integrado a sua estrutura um motor para movimentação de corrente contínua de 12V modelo 540-A959 e um servo motor de menor potência, modelo A959, para realizar o direcionamento das rodas, sendo que, esse segundo, possui ainda um sistema de potenciômetro interno com finalidade de ajustar o ângulo de direção das rodas, tal como apresentado na Figura 31.

Figura 31 – (a) Motor de Velocidade. (b) Servo de Direção do Veículo.

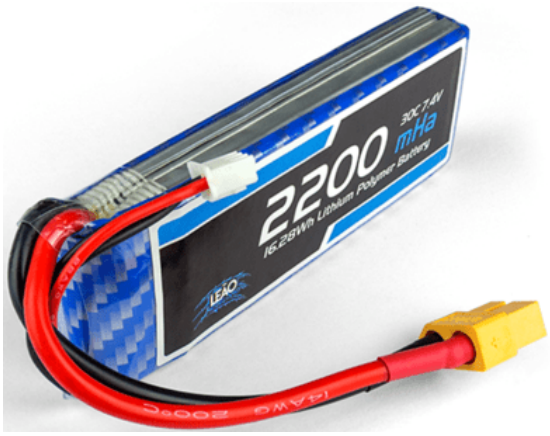


Fonte: Elaborado pelo autor, 2022.

O chassi comporta todo o restante da estrutura de *hardware* do robô, um sistema de alimentação é instalado em seu interior, possuindo uma parte potência em que os componentes recebem a tensão diretamente da bateria, que possui uma tensão em plena carga de aproximadamente 8V e circula uma corrente de 7A para suprir o acionamento do automodelo. Além disso, um circuito de controle é alimentado por um conversor de tensão que fornecerá energia para restante dos componentes a uma tensão de 5V e corrente aproximada de 3A.

A bateria possui as seguintes especificações: recarregável de Polímero de Lítio (Lipo) de 7,4 volts nominal, capacidade de corrente 2200mah e descarga máxima de 30 vezes a capacidade de corrente (30C), sua característica principal para o projeto é a baixa resistência interna, que permite fornecer altas correntes e possui alta densidade de energia e está apresentada na Figura 32.

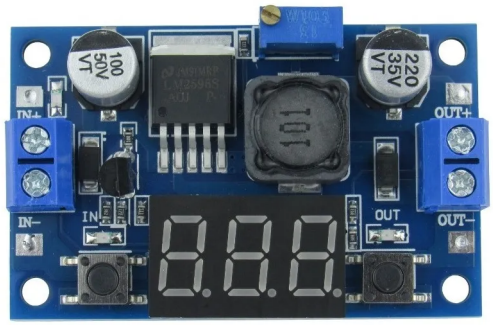
Figura 32 – Bateria Lipo.



Fonte: Elaborado pelo autor, 2022.

Como a tensão fornecida é de 7,4V, não sendo compatível com os demais dispositivos de *hardware*, além de não fornecer uma tensão estável, um conversor Buck de tipo abaixador de tensão de contínua para contínua (DC/DC) modelo Lm2596 de 3A foi destinado para reduzir a tensão para 5V, conforme pode ser visto na Figura 33. Uma importante escolha é a utilização de *display* para medir a tensão de entrada, assim, sendo possível verificar se a bateria está sem carga.

Figura 33 – Conversor Buck.

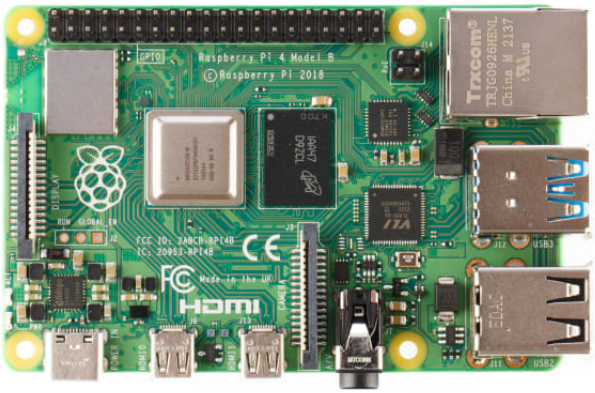


Fonte: Elaborado pelo autor, 2022.

Essencialmente, o conversor Buck, terá como funcionalidade a alimentação direta do Raspberry, do LIDAR, e do motor de direção do veículo. A potência desses dispositivos é relativamente média em relação a todo *hardware*, utilizando, portanto, praticamente o limite de 3 amperes fornecido pelo conversor.

O Raspberry Pi4 model B, apresentado na Figura 34, é um microcomputador escolhido com processamento suficiente para controlar todos os dispositivos de *hardware*. O modelo possui um conjunto de 40 GPIOs para interligação com dispositivos, além de possuir 4 entradas USB que também são projetados para uso nesse trabalho para comunicação com o LIDAR e a IMU, possui ainda, comunicação Ethernet e Wifi, previsto como meio de configuração do dispositivo.

Figura 34 – Raspberry Pi4 Model B.



Fonte: Elaborado pelo autor, 2022.

O LIDAR desse projeto é o modelo YDLIDARX4, ele tem a capacidade de retornar uma nuvem em duas dimensões de 505 pontos distribuídos ao redor de 360 graus. Na Figura 35 é possível perceber que ele possui um dispositivo de interface que possui duas entradas, uma para alimentação e a outra para comunicação de dados via USB.

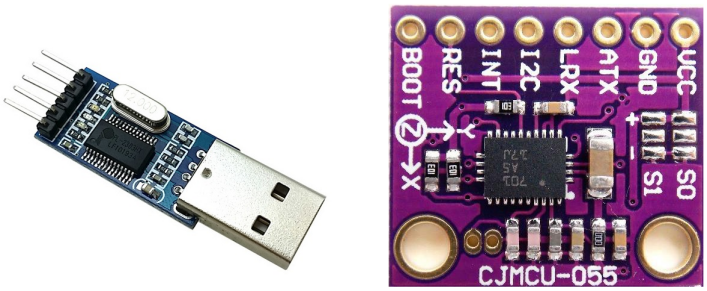
Figura 35 – Sensor LIDAR modelo YDLidar X4.

Fonte: Elaborado pelo autor, 2022.

Outro dispositivo que tem a comunicação realizada por USB é a IMU. Nesse caso, é necessário realizar uma adaptação para realização da comunicação de forma serial com o USB, isso porque, no trabalho optou-se por utilizar um pacote do ROS para comunicação com a IMU que está configurado para receber o sinal por USB. Portanto, um módulo TTL-PL2303 conversor USB para Serial RS232 é necessário para essa comunicação com o IMU modelo BNO055, então é necessário a interligação do módulo com os pinos RX/TX da IMU, ambos são apresentados na

Figura 36.

Figura 36 – (a) Módulo Serial modelo TTL-PL2303; (b) IMU modelo BNO055.



(a) (b)

Fonte: Elaborado pelo autor, 2022.

Uma configuração importante nesse tipo de comunicação da IMU é realizar a alteração do dispositivo para tipo UART realizando um ponto de solda do ponto S1 com o positivo.

Uma adaptação necessária na estrutura do robô é a colocação de um *Encoder* rotacional para que seja possível realizar a medição da velocidade proporcionada pelo motor de movimento do veículo. Para tanto, no eixo principal do auto modelo foi previsto a instalação do disco de Encoder, fazendo com que o movimento do robô fosse transmitido para esse disco, ainda, o sensor infravermelho do Encoder foi instalado na parte superior do disco para realização da contagem de pulsos.

Figura 37 – *Encoder* modelo HC-020K com disco.

Fonte: Elaborado pelo autor, 2022.

O modelo de *Encoder* utilizado foi o H modelo HC-020K e em que o disco possui 20 furos, funciona a 3 fios, sendo um deles positivo, o outro negativo e um terceiro fio do sinal, tal como apresentado na Figura 37.

O *Encoder* fornece o sinal necessário para se realizar o controle de velocidade de movimento do veículo que é feito por uma Ponte H. A ponte H tem finalidade de controle da velocidade

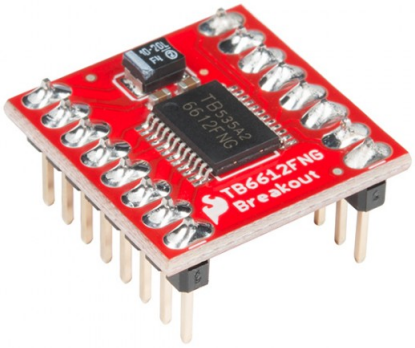
do motor por meio do ajuste da tensão média no motor por meio da utilização do PWM. Assim, uma saída analógica do Raspberry é configurada para fornecer largura de pulsos adequadas para se alterar o valor médio da tensão do motor. Dessa maneira, o valor da tensão no motor será proporcional com a largura de pulso estabelecida e consequentemente a velocidade do motor devido a característica linear do motor de corrente contínua.

O modelo de ponte H selecionado para controle da velocidade de movimento do veículo é a BTS7960, Figura 38, de corrente máxima de 43 Amperes e frequência máxima de PWM de 25khz, ideal para as altas correntes que o motor de velocidade do robô pode requerer. Também é utilizado outra ponte H de menor porte para controle da direção das rodas do veículo. O modelo utilizado para esse controle é o TB6612, Figura 39, e tem a mesma função da ponte H anteriormente apresentada.

Figura 38 – Ponte H modelo BTS7960.

Fonte: Elaborado pelo autor, 2022.

Figura 39 – Ponte H modelo TB6612.



Fonte: Elaborado pelo autor, 2022.

Todavia, é importante ressaltar que a forma de controle da direção do motor é realizada de maneira diferente quando comparado com o controle de movimento, uma vez que, o sensoriamento

do movimento é realizado pela velocidade e o sensoriamento da direção das rodas é obtida por meio da medição do ângulo das rodas com um potenciômetro. O potenciômetro interno presente no servo motor da direção é utilizado para criar um circuito divisor de tensão em que o valor de tensão medido é proporcional à posição atual da direção das rodas. Essa tensão possui um valor analógico, e portanto, é necessário realizar a conversão desse valor para digital. Para isso, foi escolhido um conversor analógico para digital (Conversor A/D) para realizar essa conversão, modelo selecionado foi o ADS1115, ele fornece uma saída de 16Bits e possui comunicação I2C compatível com o Raspberry, conforme ilustrado na Figura 40.

Figura 40 – Conversor A/D modelo ADS1115.

Fonte: Elaborado pelo autor, 2022

3.5 Integração entre *Software* e *Hardware*

3.5.1 Instalação e configuração do sistema

O sistema utilizado para o robô foi o Ubuntu server versão 18.04, sendo uma arquitetura Linux compatível com o ROS Melodic, que foi utilizado para execução da programação do robô.

Após a configuração do Raspberry, outro ponto de configuração importante é a habilitação do sistema Wifi do Raspberry para realizar a comunicação com o robô para realização de comandos remotos.

3.5.2 Calibração do robô

Alguns parâmetros precisam ser ajustados no robô para que o robô tenha o melhor resultado possível, dentre as possíveis calibrações do robô, têm-se:

O *driver* de velocidade tem tanto ajuste do PID quanto o ajuste da relação entre a velocidade real veículo com relação a diferença de tempo entre pulsos do *Encoder* do veículo, descrito pela Equação 3.14.

Quadro 3 – Ajustes de Calibração.

Nome	**Variáveis**
***Driver* de Velocidade**	Constantes do PID e Constante de velocidade.
***Driver* de Direção**	Constantes do PID e Constante de direção.
Planejador de Movimento	Constantes de Força Atrativa e Repulsiva.
IMU	Ajustes do acelerômetro e giroscópio.

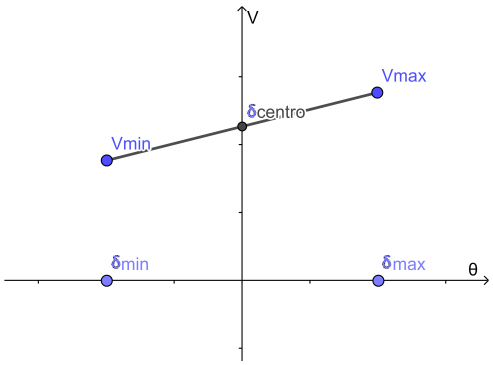
$$v = \frac{k_v}{\Delta t_{pulsos}}, \tag{3.14}$$

em que, v é a velocidade real do veículo, Δt_{pulsos} é a diferença entre pulsos do *Encoder* e k_v é a constante de velocidade que determinará a proporção da relação entre tempo de pulso e velocidade do veículo, assim, quanto menor o tempo entre os pulsos maior será a velocidade, sendo que, para ajuste desta constante, é utilizado valores empíricos e, com base em testes, esse valor vai sendo ajustado. Esses testes são baseados no movimento retilíneo uniforme realizado pelo robô, assim, sabendo que os motores de corrente contínua têm resposta linear, dessa forma, para uma velocidade constante pode-se assumir que:

$$v = d/\Delta t, \tag{3.15}$$

em que v é a velocidade real do veículo, Δt é o tempo percorrido em uma determinada distância e d a distância percorrida. Assim, ao se aplicar um valor constante de tensão no motor tem-se uma saída linear medida entre as diferenças de tempo de pulsos do *Encoder*. Pode-se determinar então que, mantida uma velocidade constante no veículo, a velocidade real pode ser encontrada de forma empírica, tornando possível o ajuste da constante k_v do veículo.

Figura 41 – Função de Ajuste da Direção



Fonte: Elaborado pelo autor, 2022.

O *driver* de direção tem um ajuste diferente devido ao sensor retornar um valor de tensão proporcional ao ângulo das rodas. Esse valor de tensão tem uma característica linear, entretanto,

a direção das rodas do veículo varia em uma faixa entre o valor negativo a um valor positivo, passando por um ponto zero, conforme pode ser visto na Figura 41.

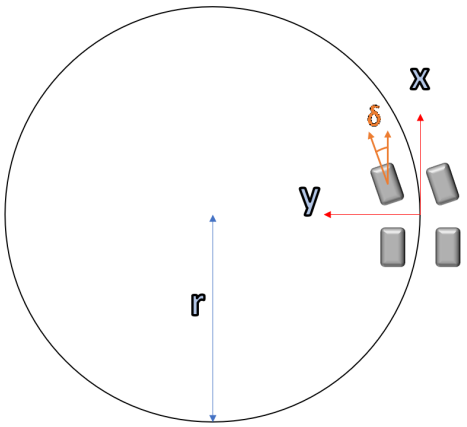
Pela equação da reta, tem-se que:

$$m = \frac{V_{max} - V_{min}}{\delta_{max} - \delta_{min}}, \tag{3.16}$$

em que o valor de m representa a inclinação da reta, V_{max} o valor máximo de tensão, V_{min} o valor mínimo de tensão, δ_{max} o valor do ângulo máximo e δ_{min} o valor do ângulo mínimo.

Entretanto, para esse cálculo é necessário ter os valores exatos de δ. Para medir esses valores é necessário ajustar esse valor fazendo o processo reverso pelo modelo Ackermann. Um controle é realizado para um valor desconhecido de ângulo, com esse ângulo é realizado o movimento circular com o veículo para se calcular δ, como mostrado na Figura 42.

Figura 42 – Método de Ajuste da Direção



Fonte: Elaborado pelo autor, 2022.

Dessa maneira é possível encontrar um valor de δ para cada raio de curvatura realizado pelo robô, assim, tendo obtido o valor do raio de curvatura após aplicação do teste, a Equação 3.2 pode ser utilizada para se obter o valor de δ.

O último valor importante para equação de calibração é o valor de tensão quando o veículo está com a direção no centro. Assim, a equação para ajuste determinação da posição é calculada por:

$$\delta = \frac{V_{atual} - V_{\delta_{centro}}}{m}. \tag{3.17}$$

O PID tanto para o *driver* de velocidade quanto para o *driver* de direção é ajustado de forma empírica com realização de teste aumentando e diminuindo os valores do PID de forma gradual até que se encontre um valor adequado.

As constantes de força atrativa e repulsiva do veículos também são realizadas de forma empírica, na realização de testes com o robô, essas constantes foram ajustadas de forma que o robô realize o trajeto da melhor maneira, desviando dos obstáculo e se mantendo estável para realização do trajeto.

Para ajuste dos dados da IMU, existe uma função dentro do pacote do ROS para calibração automática do sensor. Para isso, é necessário manter o sensor alinhado com o plano do chão e alinhado com o centro do veículo, assim o mesmo irá retornar uma posição e direção inicial para realização dos testes.

4 RESULTADOS

Os resultados se dividem entre os resultados obtidos em simulação e os resultados obtidos pelo auto-modelo construído.

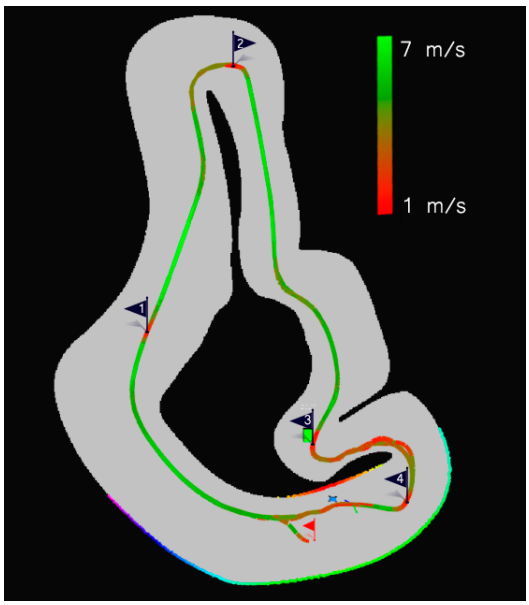
4.1 Resultados da Simulação

Os testes de simulação foram realizados utilizando o simulador da F1TENTH. Para realização dos testes foi necessário selecionar no simulador mapas para que o robô pudesse ser testado. Foram escolhidas quatro pistas fornecidas pelo simulador e em cada uma delas foram realizadas quatro voltas completas para conferência dos resultados.

Para aferição desses resultados, utilizou-se, como metodologia visual do simulador, a ferramenta RVIZ presente na plataforma ROS. Nesse sentido, foram criados vetores para representar as repostas dos campos potenciais virtuais (Repulsivo, Atrativo e Resultante). Além disso, foram criadas linhas que marcam o trajeto percorrido pelo veículo por uma faixa de velocidade em escala por cor, sendo a cor vermelho para velocidades mais baixas e cor verde velocidades mais altas.

Outro ponto de marcação nos resultados foi a colocação de pontos de chegadas, onde foi colocado o ponto de interesse de chegada do robô, em que os pontos de campo virtual atrativo é selecionado de maneira sequencial, ou seja, um novo ponto é dado ao robô após passar pelo *checkpoints*. Uma bandeira vermelha também foi indicada no simulador para aferição do ponto de partida.

Figura 43 – Simulação no mapa Berlim.



Fonte: Elaborado pelo autor, 2022.

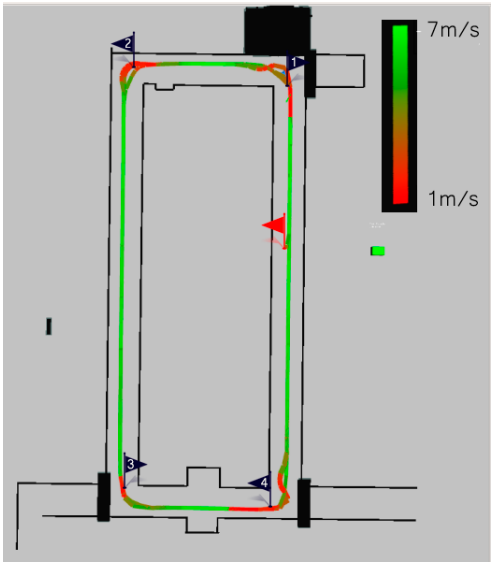
Nos mapas foram testados os modelos de navegação utilizando o métodos de planejamento

de movimento por campos potenciais proposto neste trabalho, o sistema de localização do simulador retorna a localização ideal do robô não sendo necessário utilizar nenhum método para definição da odometria.

Em primeira análise, no mapa Berlim verificado na Figura 43, o percurso realizado pelo robô é mantido bem próximo ao centro da pista. Pode-se aferir que nesse caso o resultado apresentado pela solução foi satisfatório, uma vez que, o robô praticamente não reduz de velocidade para cumprimento do trajeto apresentando uma forma suave até o primeiro ponto de chegada. Para o segundo ponto de chegada, o veículo reduz a velocidade para realização da curva conforme esperado no modelo proposto. No terceiro ponto, percebe-se que existe uma parede logo em frente, com isso, o robô precisa fazer uma curva mais acentuada para chegar ao quarto ponto de chegada tendo uma redução mais brusca da velocidade. Outro ponto importante de se notar é que, devido ao obstáculo, o campo virtual repulsivo tem maior efeito na resposta do robô, causando maiores oscilações do movimento.

No mapa Levine, apresentado na Figura 44, é visto um comportamento diferente do modelo quando comparado com os outros mapas utilizados. Devido ao formato retangular do mapa, o veículo demonstra grande estabilidade nas retas e altas velocidades, enquanto nas curvas é perceptível uma maior variação do veículo, bem como, velocidades consideravelmente reduzidas.

Figura 44 – Simulação no mapa Levine.

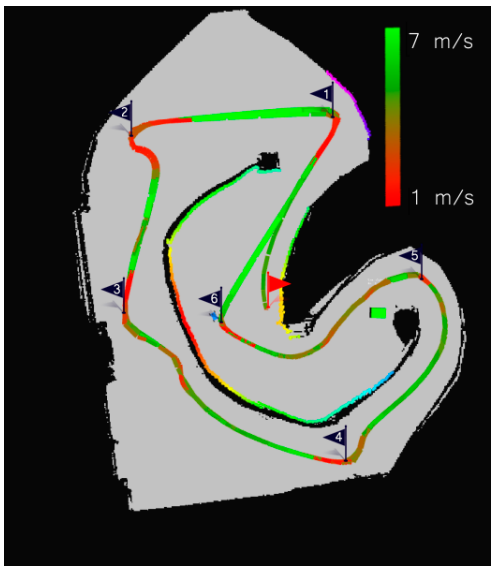


Fonte: Elaborado pelo autor, 2022.

Como pode ser visto na figura apresentada, o planejador desenvolvido demonstra velocidades altas nas retas com a ausência de obstáculos, chegando próximo a $7m/s$. Já nas curvas, é necessária uma redução considerável para que seja possível realizar a curva acentuada, ficando em velocidades próximas a $1m/s$.

Uma outra condição avaliada foi a resposta do robô a obstáculos na pista. Nesse caso, o mapa MTL, Figura 45, possui curvas muito acentuadas onde as paredes são os obstáculos

Figura 45 – Simulação no mapa MTL.

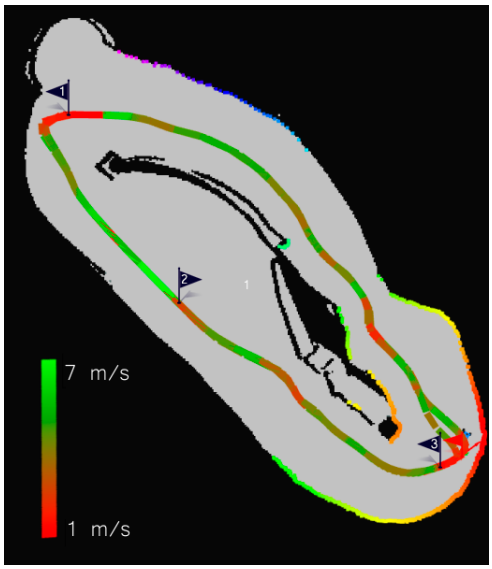


Fonte: Elaborado pelo autor, 2022.

tornando o trajeto desafiador.

Portanto, nesse mapa vários pontos de chegadas foram necessários para realização da navegação pelo robô. Assim, é possível perceber também que houve uma maior oscilação no movimento do veículo para alcançar os pontos de chegada. Por isso, o robô teve uma velocidade mais baixa para completar esse percurso, além disso, em poucas partes do trajeto o veículo conseguiu alcançar velocidades mais acentuadas.

Figura 46 – Simulação no mapa Columbia.



Fonte: Elaborado pelo autor, 2022.

O último teste foi realizado na pista Columbia, apresentada na Figura 46, onde foram utilizados três pontos de chegada. Nesse mapa, o veículo teve grandes variações de velocidade

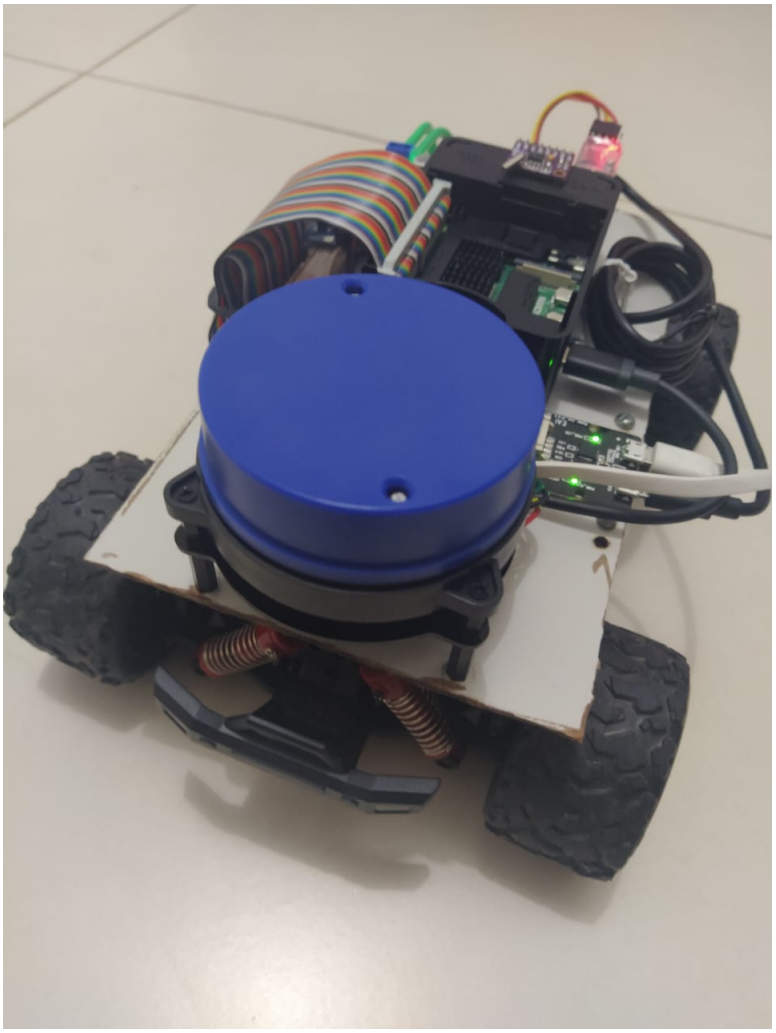
no percurso, essa oscilação pode ser vista nas várias oscilações de cores entre verde e vermelho da imagem apresentada. Ainda assim, mostrou-se satisfatório a movimentação do veículo que percorreu toda a pista.

De forma geral, é possível perceber que em todas as pistas simuladas, o modelo desenvolvido para movimento e desvio de obstáculo teve um resultado satisfatório, esses resultados também podem ser vistos de forma dinâmica por meio do link <https://youtu.be/iJUIsDVP0R8>.

4.2 Resultados do *Hardware*

Todo *hardware* foi desenvolvido conforme proposto na Metodologia. O resultado foi positivo pois todos os componentes funcionaram como previstos, a imagem do veículo construído pode ser vista na Figura 47.

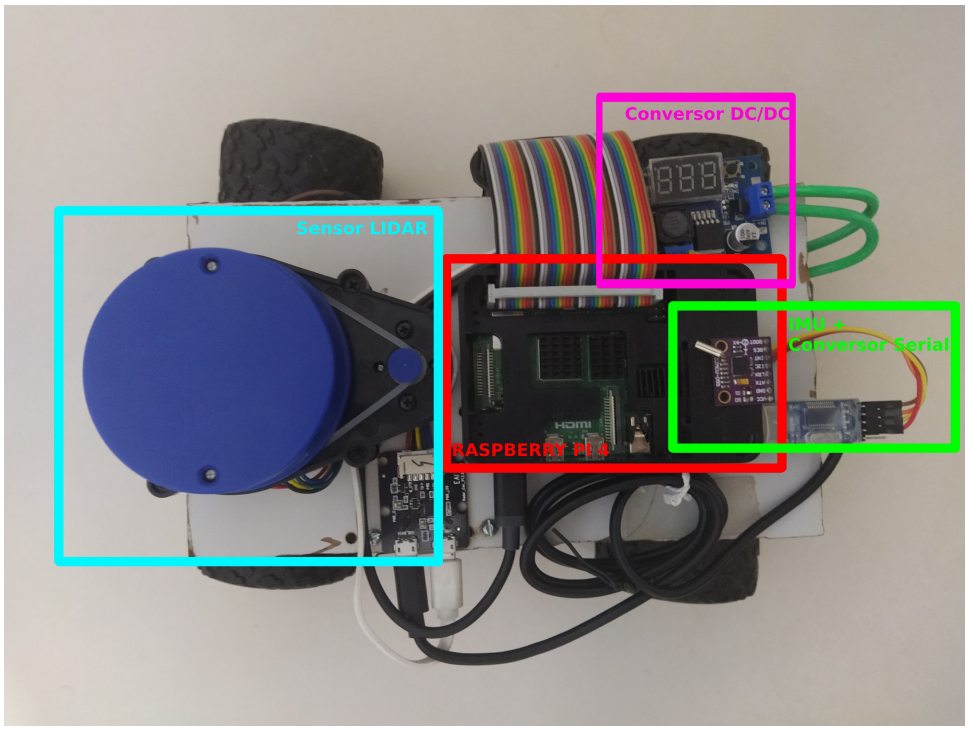
Figura 47 – Modelo construído de veículo autônomo.



Fonte: Elaborado pelo autor, 2022.

Em mais detalhes, nas imagens 48 e 49 é possível ver como foi distribuído cada um dos componentes do robô.

Figura 48 – Componentes da parte superior do robô. Na parte frontal se encontra o sensor LIDAR, no centro foi disposto o *Raspberry* PI 4, na parte traseira foi colocado a IMU junto com o conversor serial, na lateral do veículo foi instalado o conversor DC/DC.



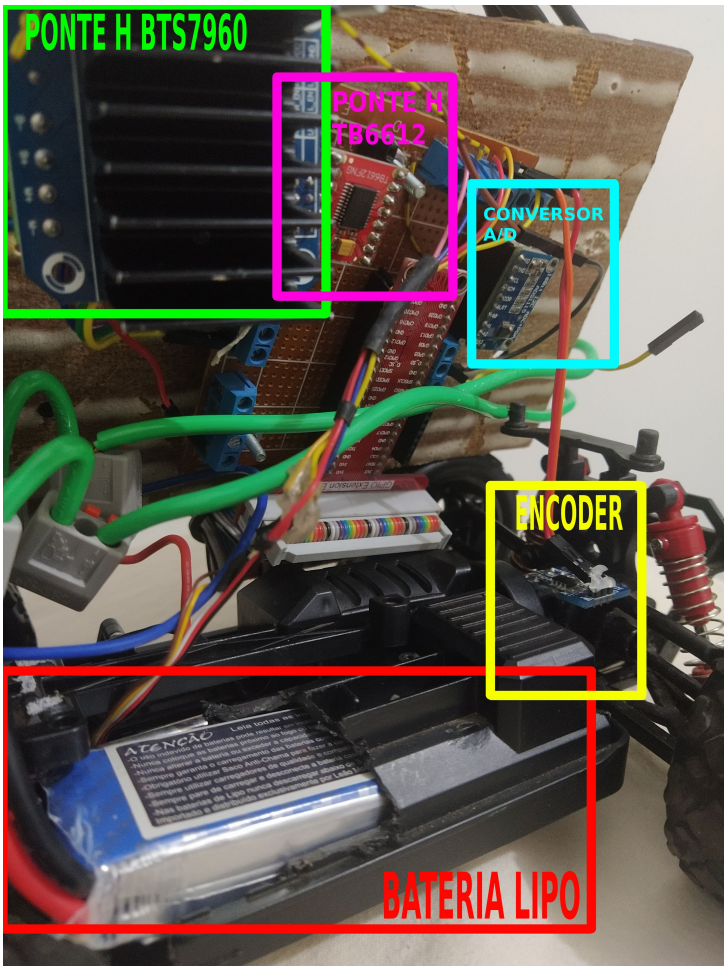
Fonte: Elaborado pelo autor, 2022.

No centro da parte superior do veículo foi instalado, sobre uma chapa de MDF, o *Raspberry Pi* onde são conectados os cabos de comunicação com os demais componentes, ao *Raspberry Pi* foi instalado um extensor que possibilitou transferir as entradas e saídas do dispositivo para parte interna do veículo. O conversor DC/DC também foi instalado na parte superior do veículo, e como resultado, possibilitou a visualização em tempo real da tensão da bateria. Na parte de trás do veículo foi instalado a IMU que está alinhada com o centro do veículo de forma a indicar de forma correta a direção do veículo a sua frente. É muito importante que a posição esteja bem alinhada, esse correto alinhamento retornou um valor correto em relação a orientação real do veículo. Também alinhado à linha central simétrica do veículo foi instalado o LIDAR em sua parte frontal e, estando acima de todos os componentes, para que nenhuma outra parte do veículo influenciasse na medição.

Na parte interior do veículo, os componentes foram instalados na parte inferior de uma chapa de MDF por meio de parafusos e porcas. A instalação desses componentes não teve uma boa distribuição e o motivo foi o pequeno espaço interno devido a dimensão do veículo não propiciar as alterações necessárias, dessa forma, no centro do veículo foi desenvolvido uma placa extensora para realizar a conexão com os componentes. Nessa placa foi instalado a ponte H TB6612 e o conversor Analógico digital. Ainda, na placa foram conectados borne para conexão com os fios necessários aos quais foram feitas as ligações da ponte H BTS7960, *Encoder* e

conexões com o motor de direção. O *Encoder* teve o disco instalado no eixo central de rotação do veículo que mostrou ser um bom local de instalação pôr o eixo traduzir a velocidade do veículo diretamente por um único ponto. A bateria foi instalada na parte inferior onde foi necessário realizar uma adaptação do veículo realizando um corte na estrutura do chassi para que coubesse sua colocação, as emendas dos fios de bateria foram realizadas com conectores de emenda.

Figura 49 – Vista interior do robô desenvolvido, que mostra os principais componentes: *Encoder*, Ponte-H BTS7960, Ponte-H TB6612, Conversor A/D, Bateria LI-PO.



Fonte: Elaborado pelo autor, 2022.

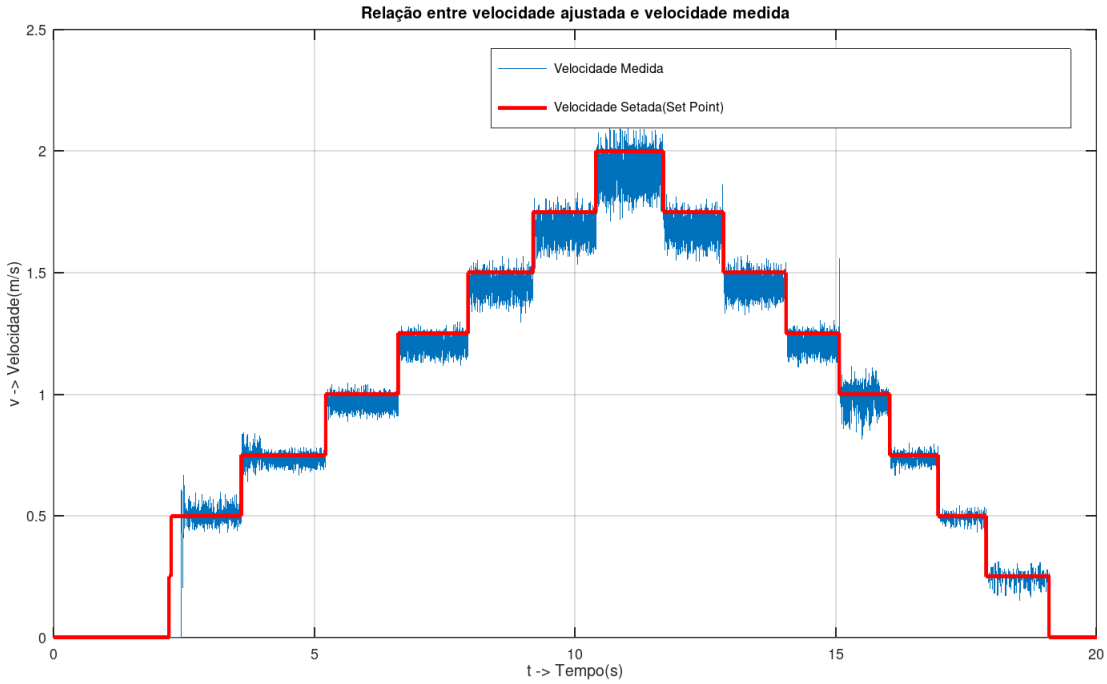
4.3 Resultados da Calibração

Os métodos de calibração foram realizados conforme discutido na Seção 3.5.2, tanto para o controle de velocidade quanto para o controle de direção do veículo. A seguir será visto os resultados obtidos por esses ajustes.

Para o ajuste de velocidade, foi extraído por testes, a curva que relaciona a velocidade pretendida com a velocidade alcançada, sendo os testes realizados diretamente no controle dos drivers. Como resultado, é possível perceber que durante a realização do controle de velocidade, a velocidade medida possui uma flutuação relevante, pode se notar que essa flutuação é resultante

da característica do sistema devido a resposta pulsante do *Encoder* e é afetada diretamente pela velocidade. Na curva apresentada fica claro que a medida que a velocidade do veículo aumenta as flutuações se tornam cada vez maiores.

Figura 50 – Curva de calibração da velocidade do robô. Em azul é vista a velocidade medida por meio do *Encoder* do robô enquanto em vermelho o valor configurado.



Fonte: Elaborado pelo autor, 2022.

Apesar das flutuações, em todo caso, o controle se mostrou eficaz, como pode ser visto, foi utilizado como referência de valores, um sinal degrau com variação em $0,25m/s$ e em todos casos o valor convergiu para o valor pretendido de forma rápida.

Os valores de calibração foram obtidos após teste descritos conforme o Quadro 4, que formam cronometrados por gravação realizada por vídeo com posterior aferição dos valores.

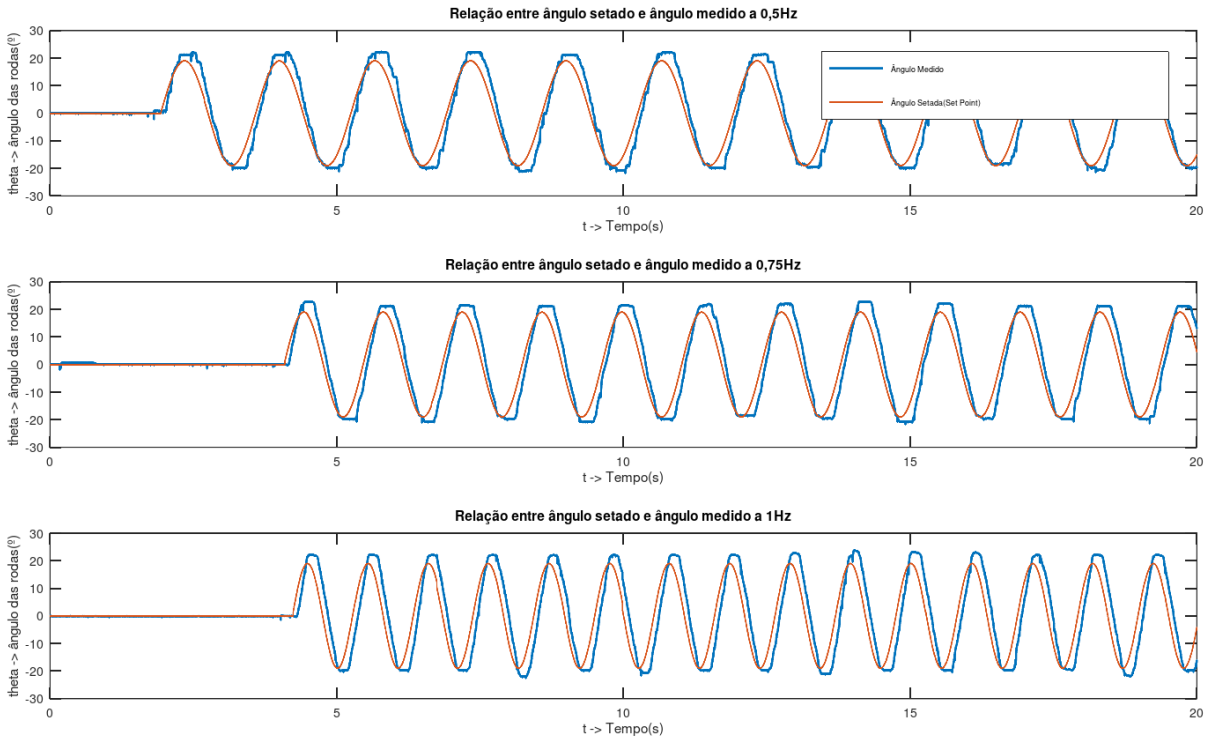
Quadro 4 – Ajustes de calibração em 10 metros percorridos.

Tempo (s)	**Velocidade Prevista**	K_v	**Velocidade Real**
19,47s	0,5m/s	0,075	0,51m/s
13,02s	0,75m/s	0,075	0,76m/s
10.05s	1,00m/s	0,075	0,99m/s
8,14s	1,25m/s	0,075	1,22m/s
6,83s	1,5m/s	0,075	1,46m/s
5,91s	1,75m/s	0,075	1,69m/s
5,3s	2,00m/s	0,075	1,88m/s

É possível perceber que na medida em que a velocidade aumenta, existe uma pequena redução real na velocidade alcançada pelo veículo. Essa condição pode ser notada por meio da Figura 50, onde é possível perceber que existe uma pequena perda média em relação a velocidade ajustada com o aumento da velocidade, assim, o valor do controle não converge exatamente para o valor definido, decaindo, na medida em que a velocidade ajustada aumenta. Esse tipo de problema pode ser mitigado com a utilização de um *Encoder* mais preciso que teria menos problemas com o aumento da velocidade.

Também foi obtido o gráfico em relação a calibração do sistema de direção do veículo, apresentado na Figura 51, em que foi definido como *setpoint* para a direção das rodas do veículo uma função senoidal variando a direção da rodas do veículo entre o ângulo de -19º a 19º.

Figura 51 – Curvas senoidais de calibração da direção do robô. Em azul está o ângulo de direção medido e em laranja o ângulo pré selecionado. A frequência de referência é aumentada a cada curva que segue abaixo.



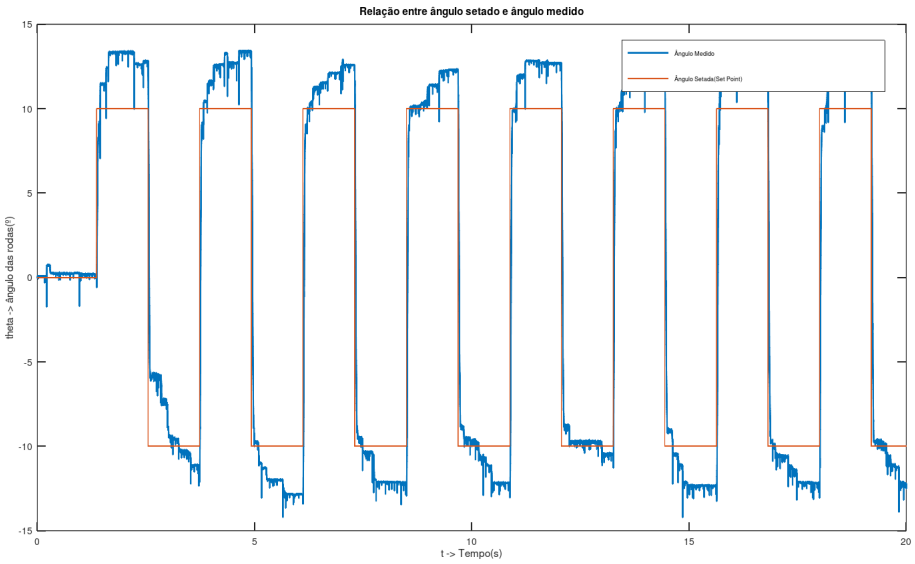
Fonte: Elaborado pelo autor, 2022.

Para realização do teste foi utilizado três frequências de referência, sendo que, para todas as frequências o valor medido convergiu para a senóide prevista como referência. Porém, é possível perceber que na medida em que a frequência aumenta, existe um atraso em relação a resposta do sinal. Isso se deve à própria limitação física do sistema mecânico da direção que não permite uma resposta com tanta velocidade.

Um outro teste realizado foi em relação a estabilidade do valor medido, então um sinal retangular foi usado para verificação, apresentado na Figura 52. Nesse teste, é possível que existe

uma relevante variação entre o valor desejado e o valor real, isso aconteceu em decorrência a condição de folga mecânica existente na direção do robô. Como esse ângulo não foi usado para cálculos de odometria essa divergência não se mostrou significativa para o andamento do trabalho.

Figura 52 – Curva retangular de calibração da direção do robô. Em azul está o valor medido enquanto em laranja o valor de referência.



Fonte: Elaborado pelo autor, 2022.

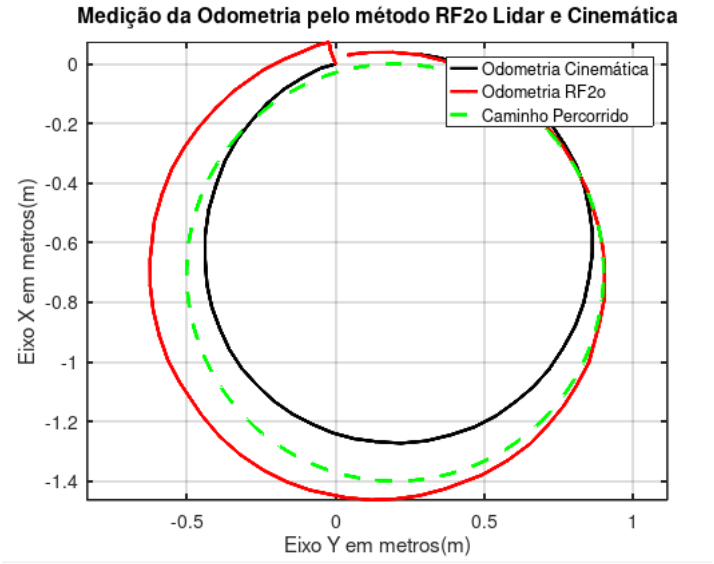
4.4 Resultados da Localização

Para obtenção da localização, a metodologia proposta foi aplicada para avaliação dos resultados. Nesses testes, foi realizada a medição individual das posições obtidas pelo método RF2O e odometria estimada pela cinemática do veículo.

Os testes foram feitos com o robô realizando um movimento circular do veículo de raio 0.7 cm, mantendo uma velocidade e ângulo de direção constantes, conforme apresentado na Figura 53. Pode-se perceber que os dados de odometria calculada não retornaram o valor exato do caminho executado pelo robô, mas os valores ficaram bem próximos. Na odometria obtida a partir da cinemática do veículo é possível perceber que inicialmente o valor ficou mais próximo do real e o mesmo partiu exatamente do ponto inicial (0,0). Já na odometria obtida pelo algoritmo RF2o pode-se perceber que existe um erro inicial na partida do robô e esse erro se torna mais relevante na medida em que é acumulado ao longo do tempo.

Como já explicado mencionado neste trabalho, é comum a existência de erros nas medições de posição do robô e por isso é utilizado métodos para fusão de sensores para melhorar esses valores. Portanto, neste trabalho foi realizado teste de aplicação do filtro de Kalman, assim, os valores foram aplicados a um filtro de Kalman e foi obtido com resultado a imagem da Figura 54.

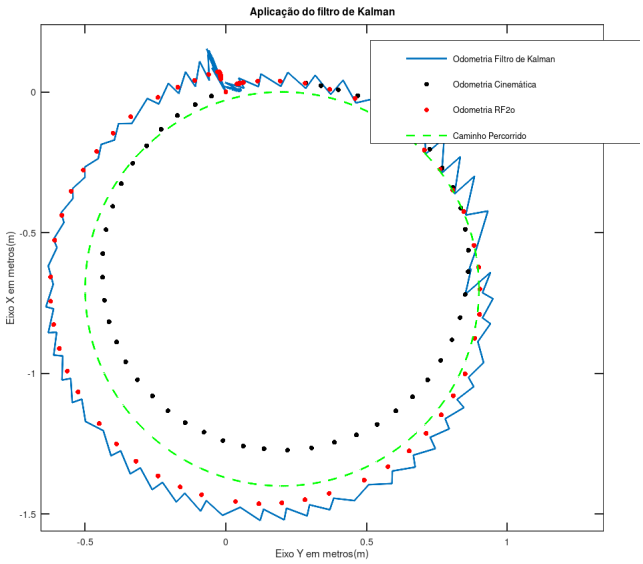
Figura 53 – Comparação entre odometrias obtidas pelo Método RF2o e Cinemática. A odometria pelo método RF2o está representada em vermelho, pelo método da cinemática do veículo em preto e o caminho real percorrido pela linha tracejada em verde.



Fonte: Elaborado pelo autor, 2022.

Contudo, os resultados obtidos pelo filtro de Kalman não foram satisfatórios, como pode ser visto pela imagem, o filtro não teve uma resposta que convergiu para o caminho real. Após a realização de vários testes, foi constatado que alguns parâmetros do filtro de Kalman possuem ajustes que demandam prática ou técnicas muito complexas, o que dificultou o seu ajuste. Assim, optou-se por utilizar como odometria apenas o valor obtido pela cinemática do veículo e em trabalhos futuros pretende-se desenvolver uma metodologia de sintonização desses parâmetros.

Figura 54 – Teste de aplicação do filtro de Kalman. Em azul é apresentado o caminho projetado para o movimento do veículo ao aplicar o filtro.



Fonte: Elaborado pelo autor, 2022.

4.5 Resultados do planejamento de movimento

Para teste do sistema de movimento no veículo autônomo, os testes foram divididos em duas etapas: a primeira delas foi o teste de realização de uma trajetória definida e a segunda, a realização do teste de reação a obstáculos.

Os testes foram realizados na quadra de esportes do IFMG Campus Avançado Itabirito, o local foi escolhido por ser amplo e existir marcações com referências de medidas.

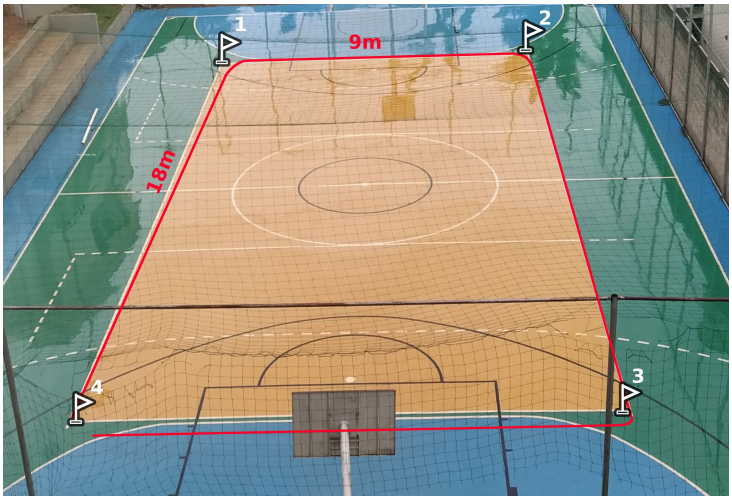
Para realização do teste de planejamento de movimento, foi definido uma coordenada como *checkpoint* para cada vértice do retângulo central da quadra, que corresponde às medidas oficiais de uma quadra de vôlei, sendo definido ao robô as coordenadas referentes ao Quadro 5.

Quadro 5 – Coordenadas definidas para o teste de movimento do robô.

Aresta	**Coordenadas**
Aresta 1	$(x = 18m,\ y = \ 0m)$
Aresta 2	$(x = 18m,\ y = -9m)$
Aresta 3	$(x = \ 0m,\ y = -9m)$
Aresta 4	$(x = \ 0m,\ y = \ 0m)$

Como resultado, a Figura 55 apresenta o caminho percorrido pelo veículo na quadra (em vermelho), sendo que as bandeiras de 1 a 4 representam os pontos desejados de chegada, respectivos às arestas de 1 até 4. Além disso, foi determinado um raio de um metro para consideração de chegada em cada ponto desejado.

Figura 55 – Caminho percorrido na quadra pelo robô. Em vermelho está a representação aproximada do caminho percorrido.



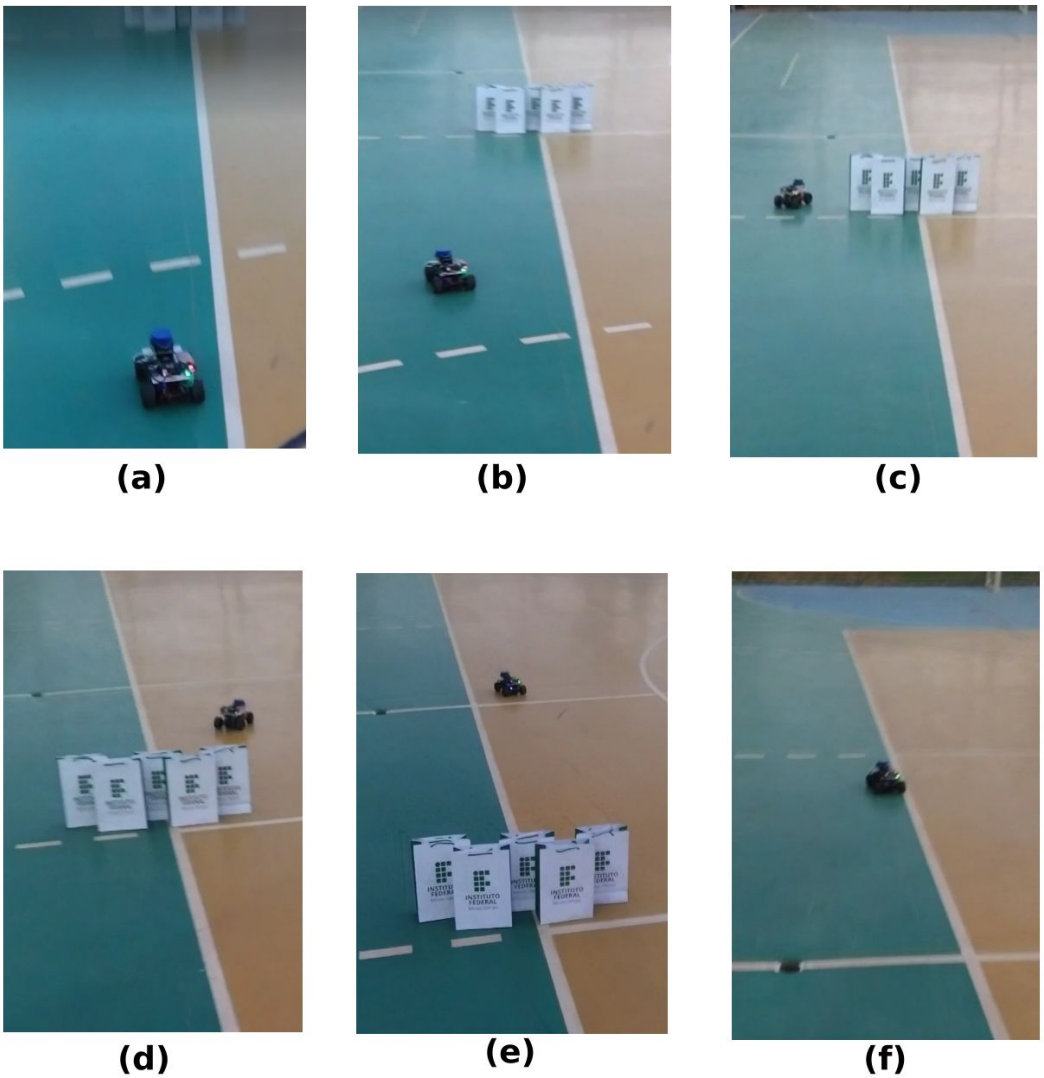
Fonte: Elaborado pelo autor, 2022.

Como pode ser visto ainda na Figura 55, o caminho percorrido pelo veículo (em vermelho) foi bem fiel ao caminho definido para locomoção (limites da quadra), mostrando que o sistema

de localização apresentou um bom ajuste de odometria. Pode-se perceber ainda que houveram dois desvios da linha de encontro aos pontos de chegada 1 e 4. No ponto 1 foi visto um pequeno desvio para esquerda que não chegou a ultrapassar 30 cm, sendo considerado um erro normal. Um outro erro foi após passar pelo ponto três, como o veículo já havia percorrido cerca de 45 metros, a odometria já havia acumulado um erro e é possível perceber que é ultrapassado o valor no eixo x durante o retorno, com um erro de aproximadamente 50 cm. Esses resultados podem ser vistos em detalhes no vídeo: https://youtu.be/bsLI31VnDQU.

Para realização do teste de desvio de obstáculo, o veículo foi colocado em linha reta e foi adicionado um obstáculo à frente do veículo para observar sua reação ao obstáculo, como resultado, o conjunto de imagens da Figura 56 representa o caminho percorrido.

Figura 56 – Desvio de obstáculo realizado pelo robô, sendo apresentado a progressão temporal do movimento de (a) à (f).



Fonte: Elaborado pelo autor, 2022.

Na Figura 56 é possível perceber que o veículo está se movimentando na sequência de (a) até (f). Na Figura 56(a) o veículo inicia seu movimento já realizando um desvio a esquerda,

isso acontece porque o objeto que se encontra a 6 metros à frente já é percebido pelo robô. Na Figura 56(b) o robô já está mais distante do objeto já prevendo o movimento necessário para dar a volta no objeto. Na Figura 56(c) o veículo já está fazendo a curva para voltar a linha central de referência. Na Figura 56(d) o objeto passa da linha central de referência até que estabiliza seu movimento retornando a linha central, vista na Figura 56(e). Na Figura 56(e) o veículo retorna a reta central estabilizando o movimento. O percurso de desvio pode ser visualizado pelo link: <https://youtube.com/shorts/B1dMoqXmQtM>.

De maneira geral, os testes foram realizados em uma velocidade aproximada de 0,5m/s, sendo que não foi possível alcançar velocidades superiores. Após as análises realizadas, foi identificado que o software desenvolvido gerou um alto custo computacional o que exigiu maior velocidade de processamento, isso foi notado, na medida em que, os testes realizados em partes do software de maneira individualizadas se comportavam com velocidade reduzida quando comparado com os testes realizados com todo o conjunto. Assim, foi percebido que o micro-computador *Raspberry* Pi 4 não foi capaz de processar de forma eficiente toda a informação ao mesmo tempo, sendo que, quanto maior o número de funções configuradas para o robô maior foi o erro apresentado, tanta nas medições, quanto na execução de suas tarefas. Assim, velocidades superiores exigiram um processo superior de processamento, o que não foi alcançado pelo *Raspberry*. Percebe-se dessa maneira, que os resultados podem ser melhorados a partir de otimizações a serem realizadas no *software* e no *hardware* do dispositivo para que o veículo consiga atingir velocidades mais elevadas. Pode-se concluir que, ainda que com falhas, o resultado de realizar a navegação local com mapeamento foi realizada e melhorias podem ser desenvolvidas para se obter melhor eficiência na realização dessa tarefa.

5 CONCLUSÃO E TRABALHOS FUTUROS

Neste trabalho foi proposto uma estratégia de planejamento de movimento para um veículo autônomo em escala reduzida. A estratégia utilizada foi baseada em um planejador de movimento, utilizando o conceito de campos potências. Essa estratégia utilizou sensores para determinar a localização do robô e por meio de um mapeamento local foi realizado o desvio de obstáculos.

Para validação da proposta foi utilizado um ambiente aberto e amplo no qual foi possível determinar pontos para que o robô pudesse se locomover até o local desejado, além disso, foi realizado teste com obstáculo para verificar a reação do robô ao ambiente.

Foi verificado que a utilização de Campos Potenciais, de acordo com a metodologia proposta no trabalho, permitiu que o robô realizasse o movimento previsto pelo planejador alcançando o local determinado na tarefa proposta ao veículo autônomo.

Foi verificado também que a estratégia proposta obteve sucesso no desvio dos obstáculos, de maneira que ao detectar um obstáculo, o robô foi capaz de contornar esse objeto.

As configurações realizadas no robô propiciaram que o robô tivesse uma boa resposta em relação a odometria obtida, os erros de orientação e distância em relação a localização do robô foram consideravelmente baixos, mesmo que, não tenha sido utilizado o filtro de Kalman para otimização da localização.

Quando comparada a estratégia com outros planejadores, por exemplo, com o RRT, a estratégia de planejamento de movimento com Campos Potenciais proposta neste trabalho se mostrou eficiente e de fácil implementação.

Nota-se que em simulação os resultados foram melhores quando comparados à aplicação real, essa limitação ocorreu devido a condições de *hardware* do robô que diminuiu a sua eficiência.

Assim, mesmo tendo sido alcançado os resultados esperados, pode-se elencar algumas dificuldades ainda a serem superadas:

- O filtro de Kalman não foi aplicado no modelo do veículo autônomo apesar de ter sido testado. Os testes mostraram que é necessário realizar um ajuste adequado dos parâmetros do filtro de Kalman para se obter a real localização do veículo de forma precisa, mostrando assim, ser necessário aplicação de metodologias mais complexas.

- O robô apresentou limitações de *hardware* que comprometeram o desenvolvimento do veículo em diversas situações. As principais limitações foram relacionadas ao processamento, onde foi percebido que houve um atraso de resposta do sistema quando houve uma necessidade maior de processamento pelo microcomputador e necessidade de um veículo de maior dimensão para melhor ajustes dos componentes utilizados.

- Os testes foram limitados a uma velocidade máxima do robô em 0,5 m/s na aplicação real.

A estratégia em simulação permitiu o veículo alcançar próximas de 7m/s o que mostra que limitações na construção do robô necessitam de adequação para melhorar sua eficiência.

5.1 Trabalhos Futuros

Algumas propostas de trabalhos futuros são sugeridas para aprimorar esse trabalho desenvolvido:

- desenvolver uma estratégia para parametrização adequada de filtro de Kalman que seja simples e eficaz, possibilitando visualizar o ajuste;
- realizar melhorias no desenvolvimento de *hardware* do robô, utilizando um melhor microcomputador e realização da troca do chassi para se ter uma melhor distribuição dos elementos;
- desenvolvimento de um ambiente de simulação próprio que tenha ajustes mais compatíveis com o *hardware* desenvolvido e que possua elementos mais condizentes com os sensores aplicados, levando em consideração a dinâmica do robô;
- participar das competições com veículos autônomos em escala reduzida, a fim de estimular o desenvolvimento de novas pesquisas nessa área no IFMG.

REFERÊNCIAS

AHMAD, R. A. R. N.; GHAZILLA, N. M.; KHAIRI. Reviews on various inertial measurement unit(imu) sensor applications. **International Journal of Signal Processing Systems**, Engineering and Technology Publishing, v. 1, n. 2, p. 256–262, 2013. Citado na página 20.

ALVES, J. M. S. F. R.; ROSáRIO, H. F.; FILHO, L. K. A.; RINCóN, R. A. T.; YAMASAKI1. Conceptual Bases of Robot Navigation Modeling, Control and Applications. **Advances in Robot Navigation**, InTech, 2011. Citado na página 14.

BARRAQUAND, J.; LANGLOIS, B.; LATOMBE, J.-C. Numerical potential field techniques for robot path planning. **IEEE Transactions on Systems, Man, and Cybernetics**, v. 22, n. 2, p. 224–241, 1992. Citado na página 25.

BOLTON, W. **Instrumentation and Control Systems**. 3. ed. [S.l.]: Newnes, 2021. ISBN 0128234717; 9780128234716. Citado na página 22.

CHEN, Y.; XIE, X.; YU, B.; LI, Y.; LIN, K. Multitarget vehicle tracking and motion state estimation using a novel driving environment perception system of intelligent vehicles. **Journal of Advanced Transportation**, v. 2021, p. 1–16, 09 2021. Citado na página 35.

COMETLABS. **CometLabs**. 2022. Disponível em: <https://blog.cometlabs.io/>. Citado na página 23.

ELBANHAWI, M.; SIMIC, M. Sampling-based robot motion planning: A review. **IEEE Access**, v. 2, p. 56–77, 2014. Citado na página 25.

F1TENTH. **F1TENTH**. 2022. Disponível em: <https://f1tenth.org/>. Citado na página 15.

FAISAL, T. W.; PURBOYO, A. S. R.; ANSORI. A review of accelerometer sensor and gyroscope sensor in imu sensors on motion capture) sensor applications. **Medwell Journals**, Journal of Engineering and Applied Sciences, v. 15, n. 3, p. 826–829, 2020. Citado 2 vezes nas páginas 20 e 21.

FOOTE, M. T.; PURVIS. **Standard Units of Measure and Coordinate Conventions**. 2010. Disponível em: <https://www.ros.org/reps/rep-0103.html>. Citado na página 19.

FREITAS, E. J. de R. **UMA ESTRATÉGIA PARA NAVEGAÇÃO DE ROBÔS DE SERVIÇO SEMIAUTÔNOMOS USANDO INFORMAÇÃO LOCAL E PLANEJADORES PROBABILÍSTICOS**. Dissertação (Mestrado) — Universidade Federal de Minas Gerais (UFMG), 2017. Citado 3 vezes nas páginas 14, 15 e 28.

GHALLABI, F. **Precise self-localization of autonomous vehicles using lidar sensors and highly accurate digital maps on highway roads**. Dissertação (Mestrado) — Université Paris sciences et lettres, 2020. Citado na página 18.

GOTLIB, A.; LUKOJC, K.; SZCZYGIELSKI, M. Localization-based software architecture for 1:10 scale autonomous car. p. 7–11, 2019. Citado na página 27.

JAIMEZ, M.; MONROY, J.; GONZALEZ-JIMENEZ, J. Planar odometry from a radial laser scanner. a range flow-based approach. 05 2016. Citado 2 vezes nas páginas 33 e 34.

JAZAR, G. N. **Engenharia de Controle Moderno**. [S.l.]: Springer, 2008. ISBN 9780387742434,0387742433. Citado na página 31.

KAVRAKI, L.; SVESTKA, P.; LATOMBE, J.-C.; OVERMARS, M. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. **IEEE Transactions on Robotics and Automation**, v. 12, n. 4, p. 566–580, 1996. Citado na página 25.

KHATIB, O. Real-time obstacle avoidance for manipulators and mobile robots. v. 2, p. 500–505, 1985. Citado na página 24.

KOPECKY, D. **Localization and advanced control for autonomous model cars**. Dissertação (Mestrado) — Czech Technical University in Prague, 2019. Citado na página 27.

KäSTNER, L. *et al.* Connecting deep-reinforcement-learning-based obstacle avoidance with conventional global planners using waypoint generators. In: **2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)**. [S.l.: s.n.], 2021. p. 1213–1220. Citado na página 14.

LASSIG, E. R.; SISSIMATOS, T.; BUCHNER, M.; LORENZ, I.; WICKER. **Robotics Outlook 2030: How Intelligence and Mobility Will Shape the Future**. 2022. Disponível em: <https://www.bcg.com/pt-br/publications/2021/how-intelligence-and-mobility-will-shape-the-future-of-the-robotics-industry>. Citado na página 14.

LAVALLE, S. M. *et al.* Rapidly-exploring random trees: A new tool for path planning. Ames, IA, USA, 1998. Citado na página 25.

LIN, Z.; YUE, M.; CHEN, G.; SUN, J. Path planning of mobile robot with pso-based apf and fuzzy-based dwa subject to moving obstacles. **Transactions of the Institute of Measurement and Control**, v. 44, p. 014233122110247, 07 2021. Citado 2 vezes nas páginas 24 e 25.

MEEUSSEN, W. **Coordinate Frames for Mobile Platforms**. 2010. Disponível em: <https://www.ros.org/reps/rep-0105.html>. Citado na página 19.

MOHIUDDIN, M. D. A. S. G.; HUSSAIN, M. A.; ALI. An Efficient Lidar Sensing System For Self-Driving Cars. **International Journal of Scientific Research & Engineering Trends**, IJSRET, 2021. Citado na página 23.

MORRIS, R. A. S.; LANGARI. **Measurement and Instrumentation: Theory and Application**. 3. ed. [S.l.]: ACADEMIC PRESS, 2020. ISBN 9780128171417; 0128171413. Citado na página 22.

OZGUNER, T. U.; ACARMAN, K.; REDMILL. **Autonomous Ground Vehicles**. [S.l.]: Artech House, 2011. ISBN 978-1-60807-192-0. Citado 3 vezes nas páginas 20, 21 e 24.

PANIGRAHI, S. K. P. K.; BISOY. Localization strategies for autonomous mobile robots: A review. **Journal of King Saud University**, Elsevier B.V, 2021. Citado na página 18.

ROS. **ROS**. 2022. Disponível em: <https://docs.ros.org/>. Citado na página 26.

SRINIVASA, S. S. *et al.* Mushr: A low-cost, open-source robotic racecar for education and research. **arXiv preprint arXiv:1908.08031**, 2019. Citado 2 vezes nas páginas 16 e 27.

VAJNAR, M. **Model car for the F1/10 autonomous car racing competition**. Dissertação (Mestrado) — Czech Technical University in Prague, 2017. Citado na página 27.

WANG, H.; WANG, C.; CHEN, C.-L.; XIE, L. F-loam : Fast lidar odometry and mapping. p. 4390–4396, 2021. Citado na página 14.

ZHAO, S.; ZHANG, H.; WANG, P.; NOGUEIRA, L.; SCHERER, S. Super odometry: Imu-centric lidar-visual-inertial estimator for challenging environments. In: **2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)**. [S.l.: s.n.], 2021. p. 8729–8736. Citado na página 14.