Universidade Federal da Bahia
Escola Politécnica
Programa de Pós-Graduação em
Engenharia Elétrica e de Computação

DHAVIDY DE ALMEIDA SILVA
ERICK BRAGANÇA MARTINS SANTOS
LUCAS PINHEIRO SOARES

Integração de Robôs Móveis com LiDAR e ROS para Implementação de Localização e Mapeamento

Docente: Dr. Paulo César M. de Abreu Farias
Docente: Dr. Tiago Trindade Ribeiro

Salvador-Ba – Brasil
04 de Fev. de 2025

DHAVIDY DE ALMEIDA SILVA
ERICK BRAGANÇA MARTINS SANTOS
LUCAS PINHEIRO SOARES

Integração de Robôs Móveis com LiDAR e ROS para Implementação de Localização e Mapeamento

Docente: Dr. Paulo César M. de Abreu Farias
Docente: Dr. Tiago Trindade Ribeiro

Salvador-Ba – Brasil
04 de Fev. de 2025

Resumo

A utilização de robôs móveis autônomos em tarefas de monitoramento de locais perigosos tem tornados cada vez mais comuns. Um exemplo dessa aplicação é a exploração de ambientes de risco, que reduz a exposição de operadores a situações perigosas. Nesse contexto, métodos de localização e mapeamento são fundamentais para que veículos autônomos compreendam o ambiente ao seu redor e naveguem sem colidir com obstáculos. Para isso, é necessário dispor de um mapa para calcular a posição do veículo, ao mesmo tempo em que a posição do veículo é essencial para a construção desse mapa. Assim, este artigo propõe o desenvolvimento e a integração de um sistema de localização para um robô móvel do tipo car-like, utilizando Lidar 2D e ROS Noetic, capaz de mapear ambientes e se localizar com precisão. As principais melhorias incluem aprimoramentos na odometria e na aplicação do *Extended Kalman Filter* (EKF), além da utilização de pacotes como robot localization (EKF), gmapping (slam gmapping) e teleop twist Keyboard, bem como a biblioteca Boost.

Palavra-chave: Odometria, EKF, Lidar 2D.

Abstract

The use of autonomous mobile robots in monitoring tasks in dangerous locations with tornadoes is increasingly common. An example of this application is the exploration of risky environments, which reduces the exposure of operators to dangerous situations. In this context, localization and mapping methods are essential for autonomous vehicles to understand the environment around them and navigate without colliding with obstacles. To do this, it is necessary to have a map to calculate the vehicle's position, at the same time as the vehicle's position is essential for the construction of this map. Therefore, this article proposes the development and integration of a localization system for a car-like mobile robot, using 2D LiDAR and ROS Noetic, capable of mapping environments and locating itself accurately. The main improvements include improvements in odometry and the application of the Kalman Filter, in addition to the use of packages such as robot localization (EKF), gmapping (slam gmapping) and teleop twist Keyboard, as well as the Boost library.

Keywords: Odometry, Kalman Filter, 2D LiDAR.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

ROS	*Robot Operating System*
Lidar 2D	*Light Detection and Ranging*
SLAM	*Simultaneous Localization And Mapping*
UDP	*User Datagram Protocol*
EKF	*Extended Kalman Filter*

Sumário

1 Introdução

A utilização de robôs móveis autônomos em tarefas de monitoramento de locais perigosos vem se tornando cada vez mais comum. Exemplificando essa aplicação, destaca-se a exploração de ambientes confinados, que evita a exposição de operadores aos riscos presentes nesses locais.

Nesse contexto, um robô móvel autônomo precisa ser capaz de determinar sua posição, mapear o ambiente e planejar caminhos até uma localização desejada, como uma região de trabalho. Por meio do sensoriamento e controle de navegação, o robô pode seguir caminhos e trajetórias, além de desviar de obstáculos.(CRUZ Júnior, 2021).

Nesse sentido, os robôs móveis desenvolve um papel importante para o avanço da tecnologia, em diversos campos, podendo contribuir significativamente com o seres humanos para realização de tarefas que envolve risco e precisão. Nesta perspectiva, métodos de localização e mapeamento são fundamentais para que veículos autônomos consigam compreender o ambiente ao seu redor, permitindo que naveguem sem colidir com obstáculos. Para isso, é necessário contar com um mapa para calcular a posição do veículo e, ao mesmo tempo, a posição do veículo é crucial para a criação desse mapa. O desafio de estimar tanto o estado do mapa quanto a posição do veículo simultaneamente é conhecido como (VELOSO, 2019).

Para mais, (CRUZ Júnior, 2021), afirma que os problemas da localização e mapeamento são intrinsecamente conectados de tal forma que para construir um mapa é necessário ter a uma localização exata do dispositivo, e para determinar a sua posição é necessário ter um mapa. Desta forma uma solução é a determinação da localização e mapeamento simultâneos *Simultaneous Localization And Mapping* (SLAM). Existem muitas técnicas de SLAM e estas utilizam diferentes sensores como sonares, câmeras, *Light Detection and Ranging* (Lidar 2D), e também IMU e encoders de rodas para auxiliar na estimação da odometria. Sendo assim, este trabalho propôs a implementação de um sistema de localização para um robô móvel car-like, equipado com um Lidar 2D para detecção e mapeamento do ambiente.

Além disso, foram empregados métodos de localização utilizando algoritmos, entre eles a implementação do EKF, podendo obter dados tanto de odometria, acelerômetro e giroscópio. Possibilitando a leitura do mapa criado pelo pacote Gmapping. Dessa forma, o sistema desenvolvido visa proporcionar uma navegação eficiente e adaptável a diferentes ambientes, contribuindo para aplicações autônoma. Em seguida, é possível observar o modelo de robô móvel car-like utilizado no projeto.

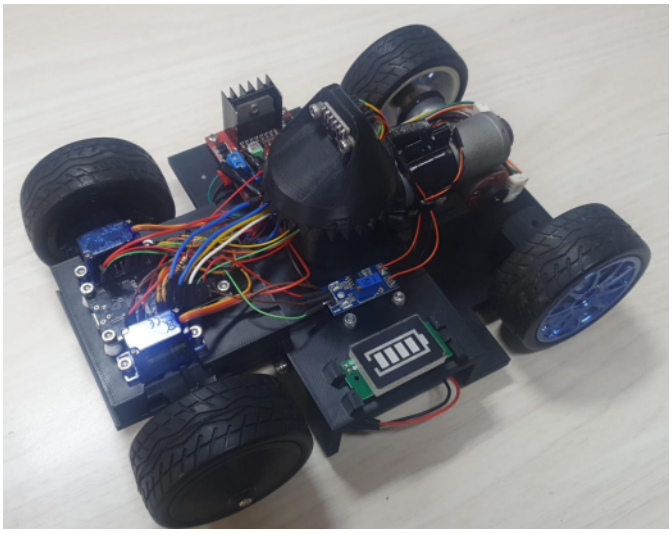


Fig. 1 – ROBÔ CAR-LIKE

A geometria de Ackermann é um conceito crucial na análise e no design de sistemas de direção para veículos, incluindo robôs móveis. Tal geometria refere-se a uma configuração específica das rodas de um veículo, onde as rodas dianteiras não possuem angulo de rotação igual entre si. Em um sistema de direção Ackermann, as rodas dianteiras são projetadas para convergir ao ponto chamado "ponto de Ackermann", ou Instantaneous Centre of Rotation (ICR), centro de rotação instantâneo, ao virar. Isso permite que o veículo realize curvas mais suaves e reduz o desgaste dos pneus durante manobras (SOARES, 2023). As relações trigonométricas na geometria de Ackermann são fundamentais para entender e modelar o movimento de veículos com direção car-like.(SOARES, 2023).

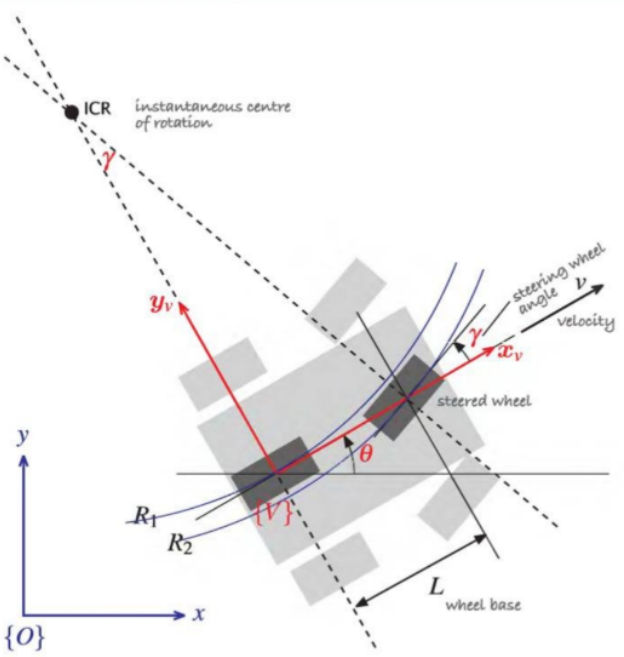


Fig. 2 – Robôs car-like

Geometria de Ackerman

O modelo de veiculo car-like, oferece uma representação precisa do movimento do robô. Essa característica é fundamental para o objetivo deste trabalho, garantindo uma confiabilidade na navegação e na construção do mapa do ambiente. Assim, o artigo propõe o desenvolvimento e integração de um sistema de localização de um robô móvel car-like uilizando Lidar 2D e ROS, capaz de mapear ambientes e se localizar com precisão.

2 Revisão Bibliográfica

A localização de robôs móveis em ambientes complexos é um campo amplamente estudado devido à sua relevância para a robótica, especialmente para robôs autônomos. Nesse contexto, diversos algoritmos e sensores têm sido desenvolvidos com o objetivo de aprimorar a precisão e a robustez frente a este problema.

Localização e Mapeamento Simultâneos SLAM é considerado um problema fundamental da robótica, no qual um robô que não possui conhecimento prévio de seu estado precisa se movimentar em um ambiente desconhecido. No SLAM, cabe ao robô construir gradativamente uma representação do ambiente enquanto se localiza em relação ao mapa criado simultâneamente (VELOSO, 2019).

O problema de Localização e Mapeamento Simultâneos é considerado um problema complexo porque para que um veículo consiga se localizar é preciso de uma representação consistente do mapa, enquanto que para adquirir uma representação do mapa é necessário uma estimativa da posição do veículo (VELOSO, 2019).

Deste modo, técnicas como estas permitem que o robô atualize continuamente sua posição no mapa enquanto explora o ambiente, permitindo que os robôs compreendam e interajam de forma autônoma em ambientes complexos e dinâmicos (SOARES, 2023). Porém, a iluminação precária dos ambientes confinados pode dificultar a identificação de features do ambiente pelas câmeras, gerando erros na estimação da pose e consequentemente na construção do mapa. Outra solução mais adequada é a aplicação de técnicas de SLAM baseadas em sensores Lidar 2D, chamada LiDAR SLAM, que possuem uma medida precisa de distância, na faixa de centímetros, e longo alcance, podendo ir de 10 a 300 metros, além de serem inerentes as condições de iluminação. Sensores do tipo Scannig LiDAR 3D possuem múltiplos feixes de lasers e são comumente utilizados em aplicações de robótica devido a sua representação densa do ambiente em 360 (CRUZ Júnior, 2021).

Outro método utilizado, é a implementação de EKF integrado ao SLAM para corrigir tanto a pose quanto o mapa fundindo a odometria LiDAR com a odometria das rodas. o EKF é um método estatístico que combina as informações provenientes dos sensores, permitindo uma estimativa precisa da posição e orientação do robô, bem como a construção do mapa do ambiente em tempo real. A abordagem EKF é utilizada para lidar com a incerteza e as não linearidades. (DANTAS Junior, 2023).

3 Definição do Problema de Estudo

A localização precisa de robôs móveis em ambientes complexos é um desafio significativo na robótica. Este estudo foca na integração de um robô móvel car-like com um sistema de localização e mapeamento utilizando Lidar 2D. A comunicação do robô será adaptada de *User Datagram Protocol* (UDP) para um tópico ROS, permitindo a implementação de algoritmos de SLAM. Para aumentar a precisão da localização, serão implementados sensores de acelerômetro e giroscópio, e dados de odometria serão utilizados em conjunto com o EKF.

3.1 Objetivo Geral

Implementar um sistema de localização para um robô móvel car-like equipado com Lidar 2D, utilizado ROS com algoritmos de SLAM para mapear e estimar a posição do robô

3.2 Objetivos Específicos

- Adaptar a comunicação do robô para tópicos ROS;
- Integrar sensores de aceçerômetro e giroscópio;
- Programar e implementar a odometria do robô;
- Aprimorar a odometria através da integrar os dados do snesores utilizando um EKF;
- Implementar um algoritmos de SLAM;
- Validar a solução em testes reais.

4 Desenvolvimento

O processo de seleção dos componentes para o projeto, considerou características específicas do robô, requisitos de desempenho, custo e eficiência energética, levando em conta as premissas previamente definidas. A seção aborda a escolha dos sistemas de direção, tração e energia, sensores, plataforma computacional e materiais complementares.(SOARES, 2023)

A integração do robô ao ROS, adaptação da comunicação do Lidar 2D para tópicos ROS, e a implementação de algoritmos de SLAM utilizando os dados de saída do Filtro de Kalman. Além disso, a programação e implementação de odometria serão realizadas para fornecer dados adicionais de movimentação do robô. Os sensores de acelerômetro e giroscópio serão implementados e seus dados, junto com os dados do Lidar 2D, serão utilizados pelo Filtro de Kalman, aumentando assim a precisão da localização. A modelagem do sistema será realizada no Gazebo, e a precisão do Filtro de Kalman será aumentada com a inclusão dos sensores. Espera-se que, ao final, o robô possa mapear e se localizar autonomamente em um ambiente de testes. Além disso, os resultados obtidos na simulação serão comparados com testes em ambiente real para validar a eficácia do sistema.

A avaliação da solução será realizada através de testes de mapeamento e localização em um ambiente controlado e em um ambiente real. Serão analisados a precisão da localização, a qualidade do mapa gerado e a eficiência dos algoritmos implementados, tanto na simulação quanto nos testes reais.

4.1 Materiais

O sensor de distância escolhido para constituir o Lidar 2D foi o modelo VL53L0X (STMICROELECTRONICS, 2022). Com ele foram comparados os sensores de distância mais disponíveis no mercado hobista, o sensores Sharp, o VL53L1X e os modelos de sensores ultrassônicos.

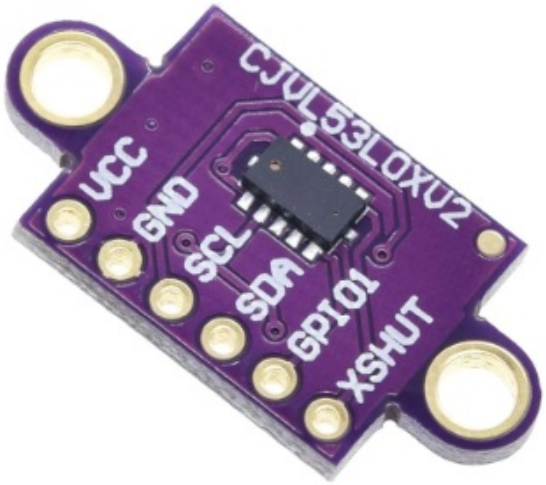


Fig. 3 – Sensor VL53L0X

A escolha do VL53L0X em detrimento dos outros sensores levou em conta principalmente seu custo, tamanho, consumo e taxa de amostragem, além da resolução da medição. O VL53L0X, desenvolvido pela STMicroelectronics, é um sensor do tipo ToF compacto e preciso. Possui uma resolução de até 1 mm e é capaz de medir distâncias de até 2 metros a uma velocidade de até 50Hz, ótimo para situações que demandam precisão sem a necessidade um alcance muito grande. Comparativamente, os sensores Sharp, embora sejam amplamente utilizados e ofereçam boas características de precisão, apresentam um consumo de energia mais elevado, custam mais caro e podem ser sensíveis a variações nas condições ambientais. Já o VL53L1X, também da STMicroelectronics, é uma versão mais avançada do VL53L0X, com maior alcance, porém, com custo também mais elevado e maior complexidade não justificam plenamente sua inclusão no projeto (SOARES, 2023)

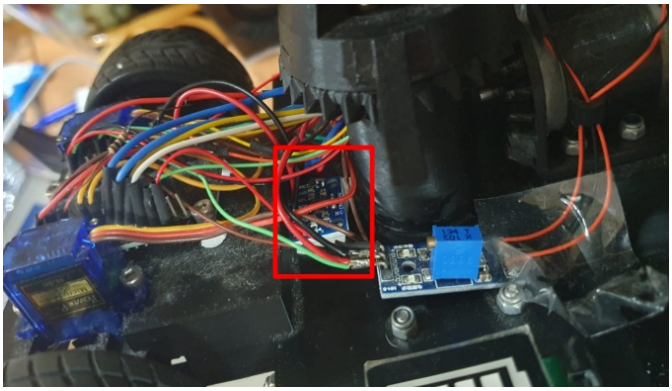


Fig. 4 – Adição do acelerômetro no Car-like

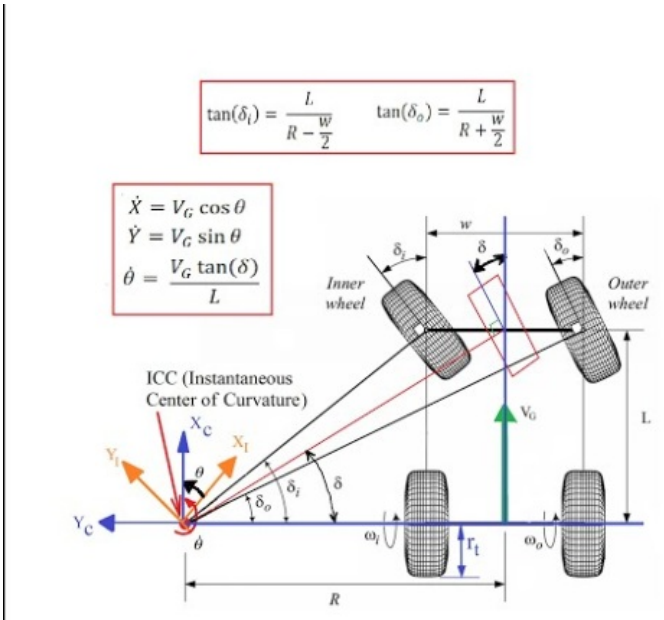
Como pode observar na imagem, foi adicionado um acelerômetro/giroscópio (IMU) em sua estrutura. O dispositivo está posicionado para medir a aceleração e a velocidade angular do robô durante o movimento.

```
[Microsegundos]_[Quantidade_de_Pontos]_[Dist0]_[Dist1]_..._[Dist15]

[Microssegundos]_[Velocidade]_[Esterçamento]_[AcelX]_[AcelY]_[AcelZ]_[RotX]_[RotY]_[RotZ]
```

Fig. 5 – Comunicação UDP

Logo após a implementação do acelerômetro, também modificou-se a comunicação UDP, que foi realizada através da biblioteca Boost.Asio, que permite criar sockets UDP para enviar/receber mensagens. As mensagens recebidas são processadas de acordo com o seu tipo. Mensagens que contém dados de velocidade, posição, aceleração e rotação (odometri e IMU) são publicadas nos tópicos /odom e /imu/data. Já os dados de distância que são medidos pelo LIDAR são publicados no tópico /scan.




```
float pos_rad = pos * M_PI / 180.0;
float rot = (vel / L) * tan(pos_rad);

// Atualiza a posição
xx += vel * dt * cos(theta);
yy += vel * dt * sin(theta);
theta += rot * dt;
```

Fig. 6 – Odometria e sua configuração

A odometria é usada para estimar a posição e orientação de um robô com base no movimento das rodas. A configuração envolve o uso de encoders para medir a rotação das rodas e calcular a posição ao longo dos eixos X e Y, além da orientação ao redor do eixo Z. Com equações cinemáticas simples, a odometria atualiza a posição do robô, mas pode acumular erros ao longo do tempo, especialmente em terrenos irregulares. Técnicas como SLAM podem ajudar a corrigir esses desvios. Onde vel é a velocidade linear, rot a velocidade angular e dt o intervalo de tempo no qual foi estipulado em 100 ms.

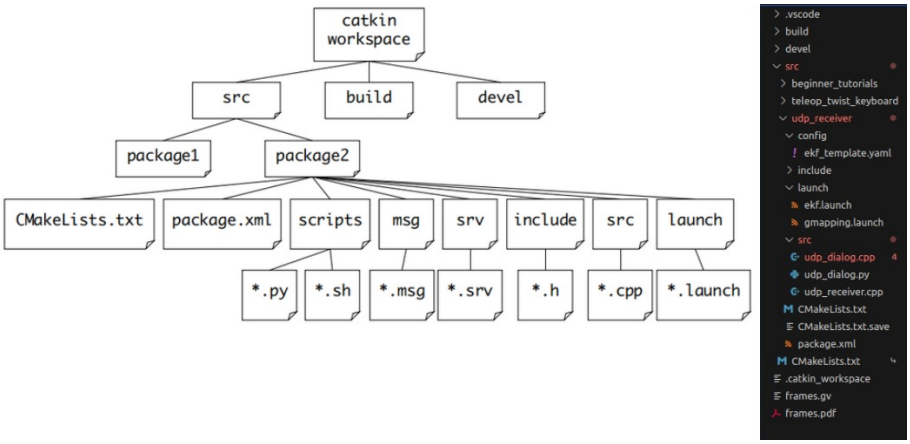


Fig. 7 – Estrutura do ROS

A estrutura do *Robot Operating System* (ROS) é composta por nós, que executam tarefas específicas, e pacotes, que organizam os nós, mensagens, serviços e arquivos de configuração. A comunicação entre os nós é feita por tópicos, que são canais unidirecionais de dados, e por serviços ou ações, que permitem interações bidirecionais. Essa estrutura modular facilita a integração de sensores, atuadores e algoritmos, tornando o sistema escalável e flexível.

Juntos desempenham papéis vitais no processo de navegação autônoma do robô. O udp-dialog.cpp gerencia a comunicação e a publicação de dados sensoriais, enquanto o ekf-template.yaml configura o EKF para estimar com precisão a localização do robô. O gmapping.launch inicia o algoritmo de mapeamento em tempo real, e o ekf.launch garante que o EKF e a comunicação UDP sejam executados de forma integrada. Com isso, o sistema permite que o robô se localize, crie mapas e se mova de forma autônoma, integrando dados de múltiplos sensores e publicando informações no ROS de maneira eficaz.

```yaml
# Odometry Input (odom0)
odom0: /odom
odom0_config: [true,  true,  true,
               true,  true,  true,
               true,  true,  true,
               true,  true,  true,
               false, false, false]
odom0_queue_size: 2
odom0_nodelay: false
odom0_differential: false
odom0_relative: false
odom0_pose_rejection_threshold: 5
odom0_twist_rejection_threshold: 1
# IMU Input (imu0)
imu0: /imu/data
imu0_config: [false, false, false,
              true,  true,  true,
              false, false, false,
              true,  true,  true,
              true,  true,  true]
imu0_queue_size: 5
imu0_nodelay: false
imu0_differential: false
imu0_relative: true
imu0_pose_rejection_threshold: 0.8
imu0_twist_rejection_threshold: 0.8
imu0_linear_acceleration_rejection_threshold: 0.8
imu0_remove_gravitational_acceleration: true
```

```yaml
# Filter Configuration
frequency: 10
sensor_timeout: 0.1
silent_tf_failure: false
two_d_mode: true
transform_time_offset: 0.0
transform_timeout: 0.0
print_diagnostics: true

# Transform Settings
publish_tf: true
publish_acceleration: false
permit_corrected_publication: false

# Frame Configuration
map_frame: map
odom_frame: odom
base_link_frame: base_link
world_frame: odom
```

Fig. 8 – Configuração do Kalman

Como pode ser observado na figura 8, o filtro de kalman está configurado para operar em 2D, combinando dados de odometria e IMU para estimar a posição e orientação do Car-like. Ele rejeita medições anômalas com limiares específicos e publica transformações para navegação.

src > udp_receiver > launch > gmapping.launch

```xml
<launch>
  <!-- Inicia o Gmapping -->
  <node name="slam_gmapping" pkg="gmapping" type="slam_gmapping" output="screen">
    <param name="base_frame" value="base_link" /> <!-- Frame base do robô -->
    <param name="odom_frame" value="odom" />       <!-- Frame da odometria -->
    <param name="map_frame" value="map" />         <!-- Frame do mapa -->
    <param name="maxUrange" value="1.0" />        <!-- Alcance máximo do LIDAR (em metros) -->
    <param name="minimumScore" value="6" />      <!-- Qualidade mínima para uma leitura ser usada -->
    <param name="linearUpdate" value="0.1" />      <!-- Atualização do mapa após mover 20 cm -->
    <param name="angularUpdate" value="0.1" />     <!-- Atualização do mapa após girar 0.1 rad -->
    <param name="particles" value="30" />          <!-- Número de partículas usadas no filtro -->
    <param name="xmin" value="-5.0" />            <!-- Limites do mapa (em metros) -->
    <param name="ymin" value="-5.0" />
    <param name="xmax" value="5.0" />
    <param name="ymax" value="5.0" />
    <param name="delta" value="0.05" />            <!-- Resolução do mapa (em metros/pixel) -->
    <param name="iterations" value="10" />
    <!-- Configuração do tópico do scan -->
    <remap from="scan" to="/scan" />
  </node>
</launch>
```

Fig. 9 – Gmapping

O Gmapping é um algoritmo de SLAM (Simultaneous Localization and Mapping) que permite ao robô criar mapas do ambiente enquanto se localiza dentro dele. Ele usa os dados do LIDAR (ou outro sensor de distância) para detectar obstáculos e construir

um mapa em tempo real. Além disso, o Gmapping consegue estimar a posição do robô no mapa, mesmo enquanto ele se move. Ele faz isso através de um filtro de partículas, que ajuda o robô a se ajustar à medida que novos dados são recebidos, melhorando a precisão do mapa à medida que o robô explora o ambiente.

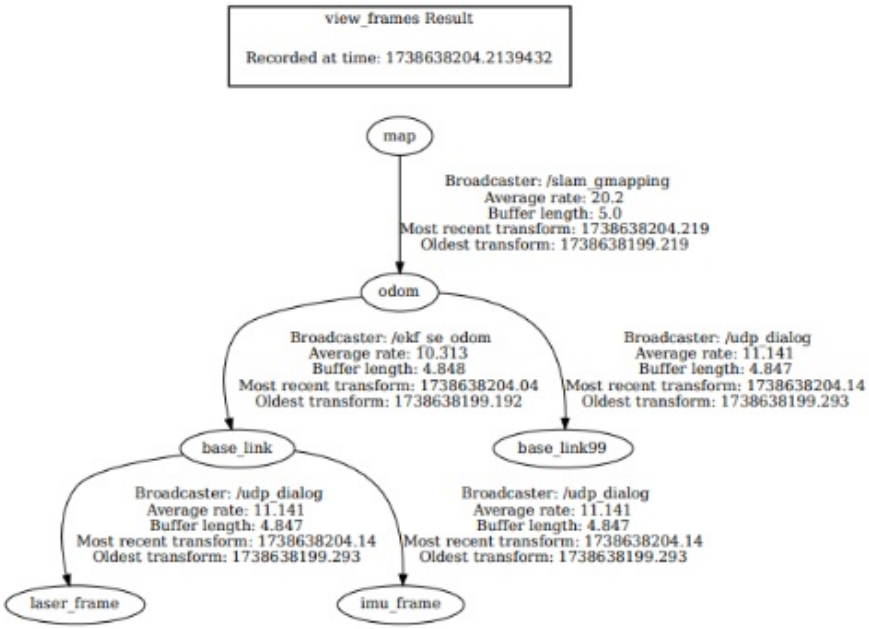


Fig. 10 – /tf

A /tf mostra a visualização das transformações de coordenadas dentro do sistema ROS, representadas pelo tópico /tf. Esse tópico é responsável por gerenciar e publicar as transformações entre diferentes quadros de referência (frames) no ambiente. Através do /tf, é possível saber a posição e orientação de um objeto (como o robô) em relação a outro, permitindo que o sistema tenha uma visão precisa de como os sensores e componentes do robô se relacionam no espaço. Na prática, o /tf é crucial para tarefas como a integração de dados de sensores (por exemplo, do Lidar 2D ou IMU), já que ele permite converter as medições para um sistema de coordenadas comum, facilitando o mapeamento e a navegação autônoma do robô.

Entre os principais frames detectados, temos o map, que é gerenciado pelo SLAM (slam-gmapping), com uma taxa média de atualização de 20,2 Hz e um buffer de transformações de 5 segundos. Isso indica que o mapeamento está sendo atualizado regularmente.

Já o odom é publicado pelo *Extended Kalman Filter* (EKF), que mantém a odometria do sistema com uma taxa de 10,313 Hz e um buffer de 4,848 segundos. Esse frame é fundamental para estimar a posição do robô ao longo do tempo. O mesmo broadcaster também é responsável pelo base-link, que representa o centro do Car-like, garantindo que sua posição seja consistente dentro do sistema de coordenadas.

Além disso, há frames adicionais, como base-link99, laser-frame e imu-frame, que são publicados pelo nó udp-dialog com uma taxa média de 11,141 Hz. Esses frames sugerem a presença de sensores remotos transmitindo dados via UDP, o que contribui para uma maior integração de informações.

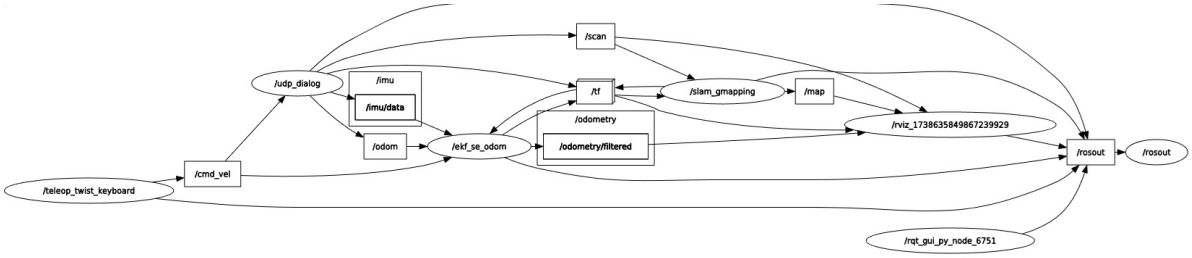


Fig. 11 – Rosgraph

O Rosgraph é uma ferramenta do *Robot Operating System* (ROS) que gera um grafo visual representando os nós (módulos) e tópicos (canais de comunicação) de um sistema ROS em execução. Ele permite visualizar como os diferentes nós estão interconectados e como trocam informações entre si. No gráfico, os nós são representados como círculos ou retângulos, enquanto as arestas conectam os nós aos tópicos que publicam ou assinam. Esse grafo ajuda a entender a arquitetura do sistema e a diagnosticar problemas, mostrando a estrutura da comunicação entre sensores, atuadores e algoritmos que estão sendo executados. O Rosgraph pode ser visualizado através de um arquivo '.svg', que é gerado automaticamente e pode ser aberto em um navegador ou visualizador de imagens vetoriais.

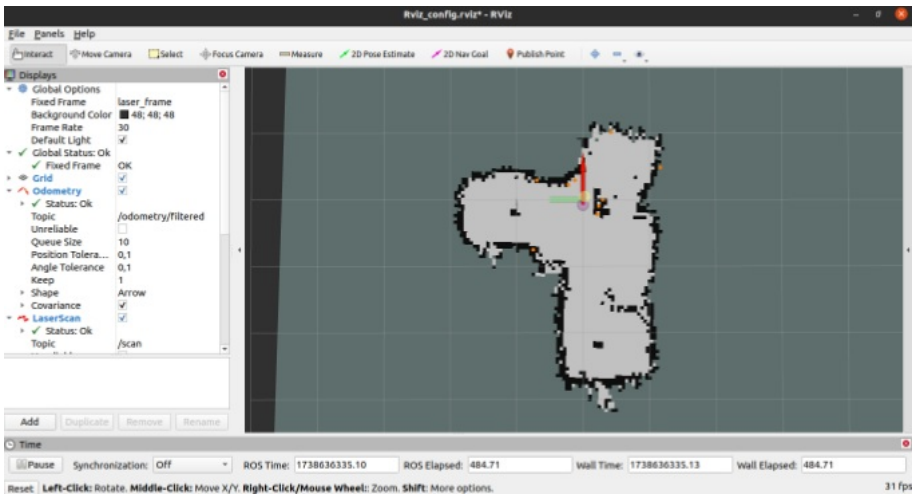


Fig. 12 – Rviz

O Rviz é uma ferramenta visual do ROS usada para visualizar e depurar dados de robôs em tempo real. Ele permite que você visualize tópicos, como mapas, sensores, modelos de robôs e outros dados relevantes, em uma interface gráfica. A principal função do Rviz é fornecer uma representação 3D do ambiente e do estado do robô, ajudando os desenvolvedores a entender o comportamento do sistema e a fazer ajustes conforme necessário.

Com o Rviz, é possível exibir informações como a posição do robô, dados de sensores como LIDAR, câmeras e IMU, além de permitir a visualização de transformações de coordenadas (TF) e o andamento de algoritmos de SLAM. Ele é uma ferramenta essencial para monitoramento e depuração de sistemas de robótica no ROS.

5 Análise de Resultados

O Car-like operou a no máximo 75 mm/s e com curvas restritas a 6 graus, o que dificultou o mapeamento em espaços mais complexos. Isso aconteceu porque o controlador PID não conseguia lidar bem com velocidades mais baixas, impactando a eficiência em áreas com poucos pontos de referência. Com isso foi selecionado um espaço mais confinado e com mais pontos de referência como pode ser observado na figura 13.

Fig. 13 – Contorno do local a ser mapeado

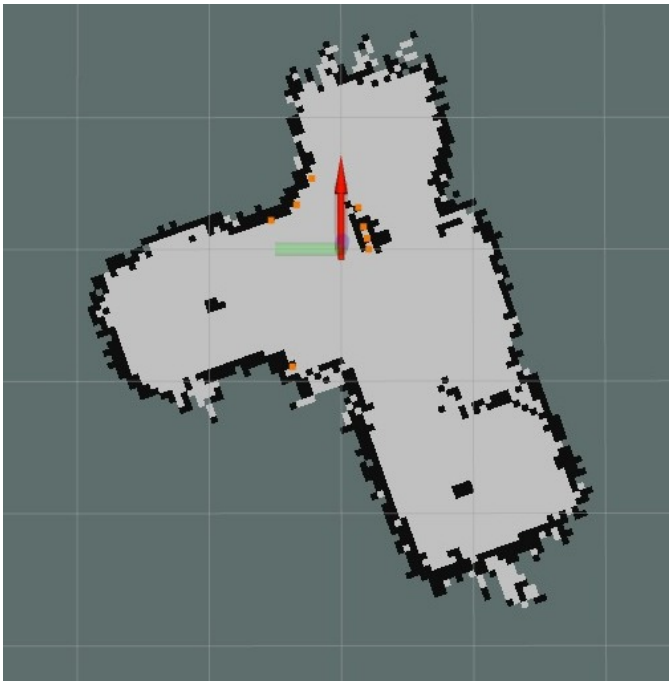


Fig. 14 – Mapa gerado no Rviz

A localização do Car-like dependia muito de referências visuais, como paredes e obstáculos, para corrigir a odometria. Foi necessário manter curvas próximas às paredes e colocar obstáculos no caminho para ajudar na orientação. Em ambientes abertos ou com poucos detalhes, isso causava erros na estimativa da posição.

Fig. 15 – Comparação do contorno do local com o mapa gerado no Rviz

Ao comparar o mapa gerado no Rviz com o real, ficou evidente que o sistema conseguiu mapear o ambiente de forma geral, mas apresentou erros em áreas onde o Car-like fez curvas ou se moveu rápido demais. Isso indica que a odometria criou um mapa errado, especialmente quando não havia boas referências visuais para corrigir a posição.

6 Cronograma e Recursos

6.1 Cronograma

As 8 semanas efetivas, entre 17/11/2024 e 25/01/2024, período até a data final de apresentaçao, serão utilizadas para as seguintes atividades:

1. Implementação da odometria e instalação do acelerômetro e giroscópio no robô;
2. Geração dos tópicos a partir da comunicação UDP;
3. Modelagem do robô no Gazebo;
4. Programação e implementação do EKF e SLAM;
5. Testes e correções e relatório de projeto.

Tab. 1 – Cronograma

Atividade	**Semanas**		
	1ª e 2ª	3ª a 6ª	7ª e 8ª
Incrementos do robô	X		
Geração de tópicos	X		
Modelagem no Gazebo	X		
Filtro de Kalman e SLAM		X	
Testes e Relatório			X

Fonte: Produzido pelo autor

6.2 Recursos

1. Robô móvel car-like com Lidar 2D;
2. Computador com ROS instalado;
3. Sensores de acelerômetro e giroscópio a serem implementados;
4. Simulador Gazebo;

7 Conclusão

O mapeamento e a localização do car-like tiveram alguns problemas ao longo dos testes. Pois as limitações de velocidade e curvas dificultaram a navegação em ambientes mais complexos, tornando o processo menos eficiente, além disso o SLAM tem uma dependência de referências visuais para se localizar, o que acarretou erros em espaços mais abertos. Também foi possível notar que ao fazer curvas ou se mover rapidamente, o mapa gerado não ficou tão preciso, acumulando pequenos desvios. Mesmo com o uso do EKF para integrar os sensores, a precisão ainda não foi ideal, mostrando será necessário ser feito ajustes nos parâmetros e o uso de mais sensores, que podem melhorar os resultados. Conclui-se que a limitação do Lidar 2D decorre de sua baixa taxa de aquisição e ângulo de visão reduzido, comprometendo sua eficiência.

Referências

CRUZ Júnior, G. P. Investigação de técnicas lidar slam para um dispositivo robótico de inspeção de ambientes confinados. *Revista da SBA*, v. 2, n. 1, 2021. Citado 2 vezes nas páginas 8 e 11.

DANTAS Junior, J. S. C. Slam de um robô móvel terrestre utilizando sensor lidar e odometria com filtro de kalman estendido. *Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Centro de Tecnologia, Curso de Engenharia Elétrica, Fortaleza*, 2023. Citado na página 11.

SOARES, L. P. Projeto e construção de um robô móvel car-like equipado com lidar 2d. *Curso de Graduação em Engenharia Elétrica da Universidade Federal da Bahia - Universidade de Salvador*, 2023. Citado 4 vezes nas páginas 9, 11, 13 e 14.

STMICROELECTRONICS. Time-of-flight ranging sensor. *ESP32 Series Datasheet*, 2022. Citado na página 13.

VELOSO, P. H. O. Desenvolvimento de um protótipo para veículos autônomos com localização e mapeamento simultâneos. *Instituto Federal de Educação, Ciência e Tecnologia de Minas Gerais – Campus Formiga.*, 2019. Citado 2 vezes nas páginas 8 e 11.