SISTEMA AUTÔNOMO PARA AÇÕES DE MONITORAMENTO E MANEJO AGROPECUÁRIO – SUBSISTEMA FASE 1

Francisco Sergio Dias Martins Junior [1], Luís Fernando Caparroz Duarte[2], Bernardo Andrade[3]

RESUMO: Esta pesquisa faz parte de um projeto que visa desenvolver uma plataforma robótica de navegação autônoma para aplicações agrícolas. Em especial, este trabalho apresenta o desenvolvimento do subsistema responsável pelo controle geral da plataforma robótica e é baseado no *Robot Operational System 2* (ROS 2). Este subsistema foi elaborado utilizando um PC com arquitetura x86 executando Linux Ubuntu 22.04. O desenvolvimento foi realizado a partir da implementação da programação em Python para ROS 2 utilizando um modelo de descrição robótica denominado URDF (*Unified Robotic Description Format*), que foi gerado a partir de um *plugin* no *software* Blender. O subsistema recebe os dados de sensores e providencia os cálculos para possibilitar a movimentação autônoma do robô. A coleta de dados dos sensores foi implementada em um trabalho paralelo que culminou no desenvolvimento de um sistema embarcado baseado no sistema operacional FreeRTOS e na API micro-ROS, com comunicação via Wi-Fi. Os cálculos de posicionamento e localização, juntamente com a monitoração do movimento em tempo real, permitiram a movimentação autônoma da plataforma, utilizando um *software* de visualização para acompanhar os dados em tempo real. Os testes realizados em laboratório demostraram a correta operação do sistema para locomoção em curtas distâncias.

Palavras-chave: ROS; navegação; robótica.

1 INTRODUÇÃO

A automação robótica facilita diversas atividades humanas, especialmente em tarefas que envolvem esforço físico repetitivo, exposição a condições insalubres ou alta precisão, onde erros podem ter consequências graves. Na agricultura, por exemplo, trabalhadores enfrentam exposição constante ao sol, manuseio de produtos químicos e levantamento de peso (LEE et al., 2023). Nesse contexto, a robótica surge como solução eficaz, integrando software e hardware para operar de maneira segura e eficiente, respeitando restrições de tempo e lógica (BENSALEM et al., 2009).

Sistemas robóticos modernos utilizam ROS (Robot Operating System) para gerenciar tarefas complexas como navegação autônoma. O Navigation2 (Nav2) é uma dessas ferramentas, empregando técnicas de SLAM (*Simultaneous Localization and Mapping*) para mapear ambientes e localizar o robô em tempo real (MERZLYAKOV;

[1] Aluno do curso de Engenharia Eletrônica; franciscosdm.jr@gmail.com.
[2] Professor Doutor da UTFPR campus Cornélio Procópio; lfduarte@utfpr.edu.br.
[3] Aluno do curso de Engenharia Eletrônica; andrade.bernardo2010@gmail.com.

MACENSKI, 2021). Sensores como LiDAR, IMU e *encoders* são usados para detectar obstáculos e estimar a posição do robô. Além disso, o micro-ROS complementa o ROS 2, permitindo a comunicação com sistemas embarcados e garantindo a operação autônoma.

Estudos recentes exploram o uso de ROS em aplicações agrícolas. Idris, Aizzat Zakaria e Ibrahim (2020) desenvolveram uma plataforma autônoma com sensores GPS e IMU, melhorando a precisão da navegação em ambientes externos. Já Alam Khan et al. (2021) implementaram SLAM em um robô agrícola simulado no Gazebo, validando algoritmos de navegação e mapeamento em cenários virtuais antes de aplicá-los em campo.

O objetivo deste trabalho é implementar um subsistema de controle baseado em ROS 2 responsável pela coordenação das ações de uma plataforma robótica experimental, permitindo que os dados dos sensores da plataforma robótica sejam processados e as decisões sejam tomadas em tempo real.

2 METODOLOGIA

O subsistema foi desenvolvido utilizando a API ROS 2 Humble Hawksbill, operando em um PC com arquitetura x86 e o sistema operacional Linux Ubuntu 22.04. A programação foi feita em Python para a implementação das funcionalidades do ROS 2.

O ROS 2 é composto por diversos pacotes que facilitam a comunicação entre os componentes do sistema robótico. A troca de informações entre esses componentes é feita através de tópicos que conectam diferentes *nodes*, utilizando interfaces que definem os tipos de dados transmitidos. Dois componentes principais do ROS 2 foram utilizados: o *Robot State Publisher*, que lê a descrição do robô a partir de um arquivo URDF (*Unified Robot Description Format*), e o *Joint State Publisher*, que fornece informações sobre a posição e orientação das juntas do robô.

Para gerar o arquivo URDF, a plataforma robótica foi modelada em Blender v4.0, replicando com precisão as dimensões do protótipo físico. O plugin Phobos foi utilizado para exportar o modelo 3D, permitindo sua integração no ROS 2. A visualização do modelo, bem como o monitoramento em tempo real do movimento e posicionamento da plataforma robótica, foram feitos com a ferramenta RViz, um visualizador robusto do ROS 2.

A arquitetura do subsistema foi desenhada utilizando a ferramenta *Lifecycle Node* do ROS 2, que permite dividir o sistema em diferentes fases, facilitando o controle do estado da execução e minimizando erros durante o processo. O sistema foi separado em duas partes principais: um subsistema servidor e um subsistema cliente. O cliente foi projetado para gerenciar as transições de estados no servidor e capturar erros durante a execução.

A plataforma robótica possui um chassis de metal com dimensões de 24,8 cm (comprimento), 10,6 cm (largura) e 3,88 cm (altura), com duas esteiras de plástico conectadas a 7 rodas de metal, uma delas dentada para tração. O sistema inclui uma placa de circuito impresso com IMU BMI323, microcontrolador ESP32-S3, LiDAR A1M8-R6 e dois *encoders* MLX90290, acoplados a motores com redução 18,8:1, configurando uma locomoção diferencial e seu desenvolvimento está descrito em (ANDRADE, DUARTE e MARTINS JUNIOR, 2024).

O sistema embarcado foi desenvolvido em conjunto, utilizando FreeRTOS e micro-ROS com comunicação Wi-Fi, responsável pela transmissão e recepção de dados entre os subsistemas. Os comandos de velocidade foram calculados com base em dados de odometria e orientação, usando um filtro complementar para fusão dos dados da IMU e acelerômetro, corrigindo as derivas de sensor.

A localização e posicionamento foram refinados pelo *Robot Localization* do ROS 2, utilizando um filtro de Kalman estendido que combina os dados de odometria e IMU para fornecer as atualizações para o *Joint State Publisher*.

Para navegação, foram utilizadas as APIs Nav2 e SLAM Toolbox. O Nav2 gerencia a tomada de decisões de movimento, enquanto o SLAM Toolbox realiza mapeamento e localização simultâneos. A integração foi feita com arquivos de lançamento específicos (*Navigation Launch* e *Online Async Launch*), gerando mapas globais e locais para navegação autônoma, com dados do LiDAR e odometria. O sensor LiDAR realiza a triangulação para obter medidas de distância, alimentando os mapas para evitar obstáculos.

Após a conclusão do sistema de navegação, um script Python foi desenvolvido para controlar a movimentação autônoma da plataforma. O robô foi programado para se locomover por pontos de referência predefinidos no mapa, ajustando sua trajetória conforme necessário para evitar obstáculos e atingir seus objetivos de navegação.

A Figura 1 ilustra as conexões entre os componentes do sistema. Cada tópico de comunicação é representado por blocos, onde as interfaces de dados são responsáveis pela transmissão de informações. Os subsistemas de cliente e servidor, o Nav2 e o sistema embarcado micro-ROS são representados como *nodes* no ROS 2, mostrando a interação entre eles. O *Robot Localization* está integrado diretamente ao Nav2, reforçando a fusão de dados de navegação e localização.

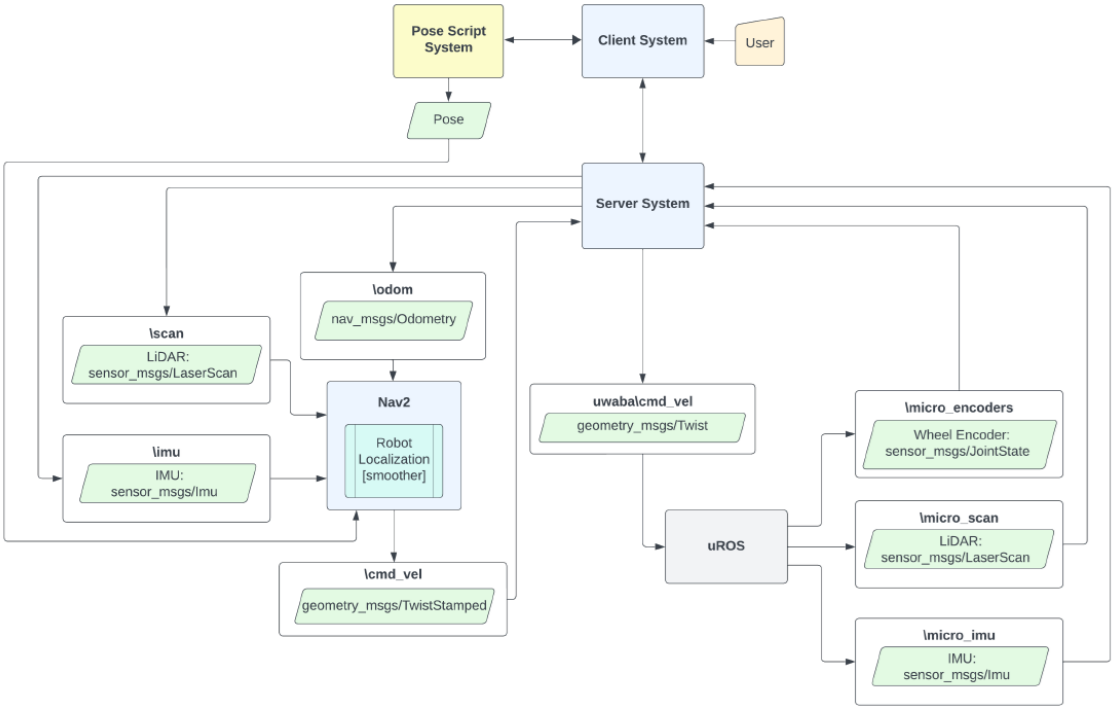


Figura 1: Diagrama de blocos representando o sistema completo. Fonte: próprio autor.

3 RESULTADOS E DISCUSSÃO

A implementação do subsistema de controle geral da plataforma foi capaz de adquirir e processar os dados para conseguir concretizar os objetivos voltados à automação robótica. Na Figura 2, nota-se a possibilidade de observar o funcionamento em tempo real da plataforma robótica conforme sua movimentação no mundo real estando denotadas as distâncias exatas de movimentação graças aos cálculos corretos de odometria e orientação em conjunto com os filtros devidamente implementados. Com este resultado conclui-se que os dados estão sendo coletados e processados de forma correta, graças ao *feedback* observado no RViz.

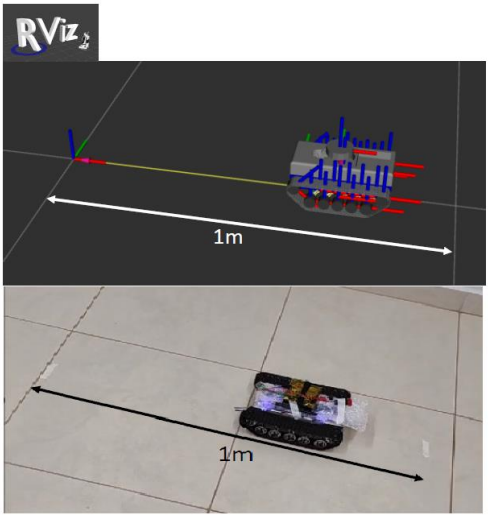
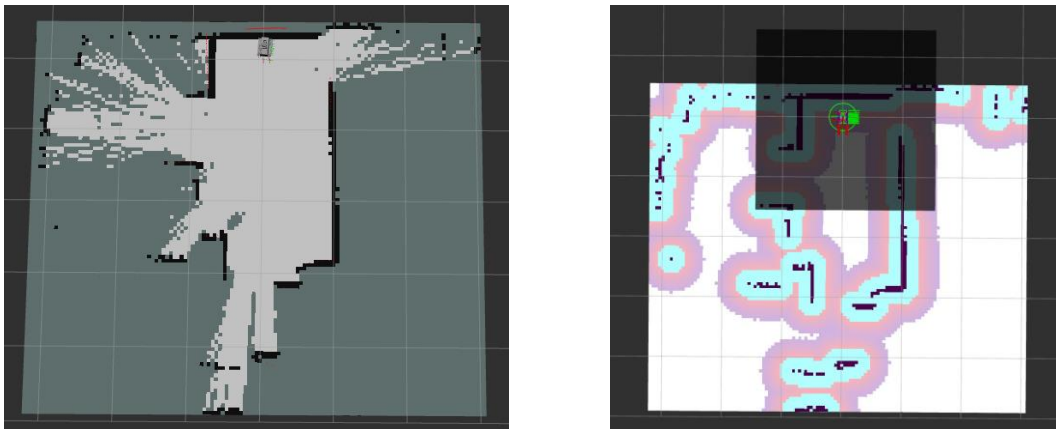


Figura 2: Comparação de distâncias. Fonte: próprio autor.

A Figura 3a demonstra o mapeamento realizado através das ferramentas SLAM e Nav2 com sucesso. Na Figura 3b pode-se observar que o Nav2 conseguiu abstrair o mapa e gerar os respectivos mapas global e local para inserir os custos associados aos elementos encontrados durante o mapeamento. Uma vez mapeados, foi possível introduzir o *script* para que a plataforma robótica se movimente de forma autônoma no ambiente.



(a) Mapa principal gerado a partir do SLAM (b) Mapas de custos gerado pelo Nav2

Figura 3: Mapas gerados a partir do SLAM do ambiente onde a plataforma robótica se encontrou. Fonte: próprio autor.

O código desenvolvido neste trabalho é aberto e está disponível no GitHub: [https://github.com/uSANMA/ROS 2-uwaba-prototype].

4 CONSIDERAÇÕES FINAIS

Este trabalho apresentou a implementação de um subsistema de controle de uma plataforma robótica autônoma baseado em ROS 2. O subsistema desenvolvido mostrou-

se eficaz no processamento dos dados em tempo real e viabilizou a navegação autônoma da plataforma em ambientes simulados. Embora ainda haja aprimoramentos a serem realizados, os resultados obtidos até o momento indicam que a integração de sensores como o LiDAR e IMU, em conjunto com os algoritmos de odometria e orientação, proporciona um controle preciso da locomoção da plataforma.

Esse sistema tem o potencial de contribuir significativamente para a automação de atividades agrícolas, trazendo benefícios como redução do esforço físico dos trabalhadores e maior precisão nas operações. O desenvolvimento deste subsistema abre portas para futuras expansões e aplicações práticas no campo, com a perspectiva de integrar novas funcionalidades e melhorar a eficiência do sistema.

AGRADECIMENTOS

Agradecimentos à Fundação Araucária pela bolsa de iniciação científica e à Universidade Tecnológica Federal do Paraná por disponibilizar os recursos de infraestrutura utilizados no desenvolvimento neste trabalho.

REFERÊNCIAS

ANDRADE, B.; DUARTE, L. F. C.; MARTINS JUNIOR, F. S. D. Plataforma robótica experimental para auxílio no manejo agrícola. In: XIV Seminário de Extensão e Inovação & XXIX Seminário de Iniciação Científica e Tecnológica da UTFPR, 2024, Francisco Beltrão. Anais do XIV Seminário de Extensão e Inovação & XXIX Seminário de Iniciação Científica e Tecnológica da UTFPR. 2024.

BENSALEM, Saddek et al. Designing autonomous robots. IEEE Robotics & Automation Magazine, v. 16, n. 1, p. 67–77, 2009. DOI: 10.1109/MRA.2008.931631.

IDRIS, Harun Dzulquornain; AIZZAT ZAKARIA, Muhammad; IBRAHIM, Ahmad Najmuddin. Outdoor Position Estimation of a Mobile Platform for Precision Farming and Agriculture Automation. In: 2020 IEEE 7th International Conference on Engineering Technologies and Applied Sciences (ICETAS). 2020. P. 1–6. DOI: 10.1109/ICETAS51660.2020.9484294.

LEE, Jeng-Dao et al. Development and Implementation of Automatic Agricultural Spraying Robot. In: 2023 9th International Conference on Applied System Innovation (ICASI). 2023. P. 59–61. DOI: 10.1109/ICASI57738.2023.10179586.

MERZLYAKOV, Alexey; MACENSKI, Steve. A Comparison of Modern General-Purpose Visual SLAM Approaches. In: 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). 2021. P. 9190–9197. DOI: 10.1109/IROS51168.2021.9636615.