Universidade Federal de São Carlos– UFSCar
Centro de Ciências Exatas e de Tecnologia– CCET
Departamento de Computação– DC
Programa de Pós-Graduação em Ciência da Computação– PPGCC

Fernanda Martins Luna

Framework Multissensorial Robusto para Localização de Robôs Móveis em Ambientes Internos

São Carlos
2026

Fernanda Martins Luna

Framework Multissensorial Robusto para Localização de Robôs Móveis em Ambientes Internos

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação do Centro de Ciências Exatas e de Tecnologia da Universidade Federal de São Carlos, como parte dos requisitos para a obtenção do título de Mestre em Ciência da Computação.

Área de concentração: Metodologias e Técnicas de Computação

Orientador: Prof. Dr. Roberto Santos Inoue

São Carlos
2026

Folha de Aprovação

Defesa de dissertação de mestrado do(a) candidato(a) Fernanda Martins Luna, realizada em 26/06/2026

Comissão Julgadora

Prof(a) Dr(a) Roberto Santos Inoue (UFSCar)

Prof(a) Dr(a) Denis Fernando Wolf (USP)

Prof(a) Dr(a) Orides Morandin Junior (UFSCar)

O relatório de defesa assinado pelos membros da comissão Julgadora encontra-se arquivado junto ao Programa de Pós-Graduação em Ciência da Computação

Este trabalho é dedicado às mentes curiosas e brilhantes que estão em constante busca por respostas e explicações sobre o sentido da vida e da existência do Universo, e que, apesar das incertezas e dos mistérios, nunca deixam de questionar, explorar e expandir os limites do conhecimento humano.

Agradecimentos

Agradeço ao professor Roberto Santos Inoue pela orientação ao longo desse trajeto, o qual foi repleto de muita aprendizagem e bons momentos.
Agradeço aos colegas do *Laboratory of Autonomous Robots and Intelligent Systems* (LARIS), que sempre me ajudaram ao longo dessa jornada, ao mesmo tempo em que a tornaram mais leve, com muitos momentos de descontração.
E acima de tudo, agradeço o constante amparo da minha família. Aos meus pais Eduardo e Sueli, pelo apoio e pela proteção, à minha irmã Nathalia, pelo carinho, e à minha avó, Irony, pelo cuidado. Ao meu companheiro Daniel, pela cumplicidade, amizade e companheirismo ao longo de todos os dias dessa jornada.
O presente trabalho foi financiado em parte pela Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Código de financiamento 001 e pelo Conselho Nacional de Desenvolvimento Científico e Tecnológico – CNPq.

"Para mim, é muito melhor compreender o Universo como ele realmente é do que persistir na ilusão, por mais satisfatória e reconfortante que ela seja."
(SAGAN, Carl; O mundo assombrado pelos demônios: A ciência vista como uma vela no escuro, 1995.)

Resumo

Robôs móveis autônomos têm sido amplamente empregados em ambientes internos, como armazéns, hospitais, laboratórios, instalações industriais e sistemas de intralogística. No entanto, a localização confiável desses sistemas ainda representa um desafio, especialmente em cenários sem disponibilidade de sinal do Sistema de Posicionamento Global, do inglês *Global Positioning System* (GPS), sujeitos à deriva da odometria, geometrias repetitivas, condições inadequadas de iluminação e medições sensoriais incertas. Embora abordagens de Localização e Mapeamento Simultâneos, do inglês *Simultaneous Localization and Mapping* (SLAM), baseadas no Filtro de Kalman Estendido, do inglês *Extended Kalman Filter* (EKF), sejam frequentemente utilizadas para fusão sensorial, seu desempenho pode ser degradado por erros de modelagem, linearizações locais e caracterização imperfeita dos ruídos.

Diante dessas limitações, esta dissertação propõe um *framework* robusto multissensorial para melhorar a localização de robôs móveis em ambientes internos estruturados. A abordagem integra odometria das rodas, observações de marcadores fiduciais ArUco e estimativas auxiliares de pose provenientes da Localização Monte Carlo Adaptativa, do inglês *Adaptative Monte Carlo Localization* (AMCL). O objetivo é aumentar a precisão, a consistência e a robustez da estimação de pose em condições reais de operação, nas quais os modelos nominais do sistema e dos sensores estão sujeitos a incertezas. Para isso, uma formulação baseada no Filtro de Kalman Robusto, do inglês *Robust Kalman Filter* (RKF), originalmente desenvolvida para sistemas lineares discretos no tempo com incertezas paramétricas limitadas em norma, é adaptada para uma estrutura não linear de EKF-SLAM com marcadores fiduciais.

A validação é realizada por meio de experimentos reais e simulações, utilizando um robô móvel equipado com sensores baseados em *Light Detection and Ranging* (LiDAR) 2D e 3D, câmera *Red-Green-Blue-Depth* (RGBD) e marcadores ArUco distribuídos em um ambiente interno sem sinal de GPS. Os resultados são comparados com o *BDU Extended SLAM* (BDUE-SLAM) e com o *Extended Kalman Filter SLAM* (EKF-SLAM)

convencional, considerando a estimação da trajetória, a redução da deriva e a consistência da localização.

Os resultados indicam que a incorporação de estratégias robustas de estimação reduz a sensibilidade do sistema a incertezas de modelo e de medição, contribuindo para uma localização mais confiável em ambientes internos estruturados. Assim, esta dissertação contribui com um *framework* de localização robusta para robôs móveis autônomos, preservando a estrutura recursiva e computacionalmente eficiente dos filtros de Kalman e ampliando sua aplicabilidade em cenários reais de navegação indoor.

Palavras-chave: Robôs móveis autônomos, localização *indoor*, EKF-SLAM, REKF-SLAM, marcadores fiduciais ArUco, fusão sensorial, filtro de kalman robusto.

Abstract

Autonomous mobile robots have been widely employed in indoor environments, such as warehouses, hospitals, laboratories, industrial facilities, and intralogistics systems. However, reliable localization of these systems remains a challenge, especially in scenarios without Global Positioning System (GPS) signal availability and subject to odometric drift, repetitive geometries, inadequate lighting conditions, and uncertain sensory measurements. Although Simultaneous Localization and Mapping (SLAM) approaches based on the Extended Kalman Filter (EKF) are frequently used for sensor fusion, their performance may be degraded by modeling errors, local linearizations, and imperfect noise characterization.

In view of these limitations, this dissertation proposes a robust multisensor framework to improve the localization of mobile robots in structured indoor environments. The approach integrates wheel odometry, observations of ArUco fiducial markers, and auxiliary pose estimates obtained from Adaptive Monte Carlo Localization (AMCL). The objective is to increase the accuracy, consistency, and robustness of pose estimation under real operating conditions, in which the nominal models of the system and sensors are subject to uncertainties. To this end, a formulation based on the Robust Kalman Filter (RKF), originally developed for discrete-time linear systems with norm-bounded parametric uncertainties, is adapted to a nonlinear EKF-SLAM structure with fiducial markers.

The validation is carried out through real experiments and simulations, using a mobile robot equipped with 2D and 3D LiDAR-based sensors, an RGBD camera, and ArUco markers distributed in an indoor environment without GPS signal. The results are compared with BDUE-SLAM and conventional EKF-SLAM, considering trajectory estimation, drift reduction, and localization consistency.

The results indicate that the incorporation of robust estimation strategies reduces the sensitivity of the system to model and measurement uncertainties, contributing to more reliable localization in structured indoor environments. Thus, this dissertation contributes a robust localization framework for autonomous mobile robots, preserving the

recursive and computationally efficient structure of Kalman filters while extending their applicability to real-world indoor navigation scenarios.

Keywords: Autonomous mobile robots, indoor localization, EKF-SLAM, REKF-SLAM, ArUco markers, sensor fusion, robust Kalman filter, BDUE-SLAM, RKF.

Lista de ilustrações

Figura 1 – Diferentes dicionários do marcador *Augmented Reality University of Cordoba* (ArUco). Fonte: Dicionário ArUco. 50
Figura 2 – Coordenadas do marcador ArUco. Fonte: elaborado pela autora. 51
Figura 3 – Processo de detecção dos marcadores ArUco. Fonte: (**??**). 51
Figura 4 – Funcionamento básico do encoder. Fonte: Adaptado de <https://www.futek.com/Incremental-Encoder-Signal-Converter>. 53
Figura 5 – Combinação da reconstrução de diversos mapas com o *Real-Time Appearance-Based Mapping* (RTAB-Map). Fonte: (LABBÉ; MICHAUD, 2014). . . 55
Figura 6 – Visualização do mapa bidimensional no *ROS Visualization* (RVIZ). Fonte: elaborado pela autora. 59
Figura 7 – Visualização do mapa tridimensional no RVIZ. Fonte: elaborado pela autora 59
Figura 8 – Braço robótico em um cenário simulado no Gazebo. Fonte: retirado de <https://gazebosim.org/showcase> 60
Figura 9 – Prédio em cenário simulado no Gazebo. Fonte: retirado de <https://gazebosim.org/showcase> 60
Figura 10 – Diagrama em blocos da arquitetura do sistema. Fonte: elaborado pela autora. 64
Figura 11 – Modelo planar de odometria do robô móvel diferencial. 68
Figura 12 – Ângulos e pose do marcador em relação ao robô móvel. 69
Figura 13 – Pipeline de sincronização baseado em *timestamps* e de processamento de predição-atualização para odometria, detecções de marcadores fiduciais e estimativas de pose do AMCL. 77
Figura 14 – Ambiente simulado no Gazebo utilizado para validar o algoritmo *Robust Extended Kalman Filter SLAM* (REKF-SLAM) com um robô móvel de acionamento diferencial. 82

Figura 15 – Robô TurtleBot 4 no ambiente simulado criado no Gazebo, que foi utilizado para validar o algoritmo REKF-SLAM com um robô móvel de acionamento diferencial. 83
Figura 16 – Comparação das trajetórias x, y e ψ ao longo do tempo para os dados simulados. 86
Figura 17 – Ampliação do intervalo de 115 s a 120 s em x, y e ψ ao longo do tempo para as trajetórias da Figura 16. 86
Figura 18 – Trajetória no plano x-y realizada pelo robô no ambiente simulado. . . . 87
Figura 19 – Ambiente utilizado nos testes experimentais com um robô móvel de acionamento diferencial. 88
Figura 20 – Robô móvel de acionamento diferencial utilizado para a validação experimental. 89
Figura 21 – Reconstrução com o RTAB-Map. 89
Figura 22 – Ambiente real com marcadores ArUco. 90
Figura 23 – Comparação das trajetórias x, y e ψ ao longo do tempo para os dados experimentais. 92
Figura 24 – Ampliação do intervalo de 125 s a 130 s em x, y e ψ ao longo do tempo para as trajetórias da Figura 23. 92
Figura 25 – Trajetória no plano x-y realizada pelo robô no mapa reconstruído utilizando RTAB-Map. 93
Figura 26 – Conversão do objeto Q em 3D para o ponto q nas coordenadas da câmera. Fonte: adaptado de (BRADSKI; KAEHLER, 2008). 114
Figura 27 – Efeitos da distorção radial. As linhas sólidas representam a área sem distorção, e as linhas pontilhadas mostram onde possui distorção radial, sendo que a é a distorção negativa e b a positiva. Fonte: Adaptado de (WENG; COHEN; HERNIOU, 1992). 117
Figura 28 – Efeitos da distorção tangencial. As linhas sólidas representam a área sem distorção, e as linhas pontilhadas mostram onde possui distorção tangencial, u e v são os eixos no sistema de coordenadas da câmera. Fonte: adaptado de (WENG; COHEN; HERNIOU, 1992). 118
Figura 29 – Janela de calibragem da câmera no ROS. Fonte: Imagem capturada pelo autor. 120
Figura 30 – Diagrama de blocos representando os componentes do *Robot Operating System* (ROS). As setas representam o sentido de comunicação entre eles. Fonte: adaptado de (QUIGLEY; GERKEY; SMART, 2015). . . . 123
Figura 31 – Estrutura básica de funcionamento do ROS. Fonte: (MARUYAMA; KATO; AZUMI, 2016). 124

Lista de tabelas

Tabela 1 – *Raíz do Erro Quadrático Médio* (RMSE) da estimação da trajetória em relação à trajetória de *ground truth* do Gazebo. 88
Tabela 2 – RMSE da estimação da trajetória em relação à trajetória de referência Point-LIO. 93

Lista de algoritmos

1 Algoritmo do REKF 67
2 Algoritmo do REKF-SLAM 78
3 Algoritmo do BDUE-SLAM 79

Lista de siglas

AMCL *Adaptative Monte Carlo Localization*

AWFG *Adaptive Weighted Factor Graph*

ArUco *Augmented Reality University of Cordoba*

AMR *Autonomous Mobile Robots*

BDUE *BDU Extended*

BDU *Bounded Data Uncertainties*

BDUE-SLAM *BDU Extended SLAM*

DDEKF *Decorrelated Distributed EKF*

DDS *Data Distribution Service*

EKF *Extended Kalman Filter*

REKF-SLAM *Robust Extended Kalman Filter SLAM*

EKF-SLAM *Extended Kalman Filter SLAM*

GPS *Global Positioning System*

GNSS *Global Navigation Satellite System*

IMU *Inertial Measurement Unit*

KF *Kalman Filter*

KLD *Kullback–Leibler Divergence*

LiDAR *Light Detection and Ranging*

LARIS *Laboratory of Autonomous Robots and Intelligent Systems*

LMI *Linear Matrix Inequality*

MCL *Monte Carlo Localization*

NDT *Normal Distributions Transform*

Nav2 *Navigation 2*

PI *Proporcional-Integral*

PF *Particle Filter*

RMSE *Raíz do Erro Quadrático Médio*

RKF *Robust Kalman Filter*

REKF *Robust Extended Kalman Filter*

REKF-SLAM *Robust Extended Kalman Filter SLAM*

RGBD *Red-Green-Blue-Depth*

ROS *Robot Operating System*

RVIZ *ROS Visualization*

RBPF *Rao-Blackwellized Particle Filter*

RLAM *Rapid Localization and Mapping*

RTAB-Map *Real-Time Appearance-Based Mapping*

SLAM *Simultaneous Localization and Mapping*

SPM *Squared Planar Markers*

UWB *Ultra-Wide Band*

UT *Unscented Transformation*

UKF *Unscented Kalman Filter*

Sumário

1 INTRODUÇÃO 25
1.1 Caracterização do Problema 27
1.2 Objetivos 28
1.3 Organização do Trabalho 29

2 REVISÃO BIBLIOGRÁFICA 31
2.1 Métodos Probabilísticos Recursivos para Localização 32
2.1.1 Métodos Baseados no Filtro de Kalman 32
2.1.2 Métodos baseados no Filtro de Partículas 35
2.2 Marcadores Fiduciais como Referências Visuais Artificiais . . . 37
2.2.1 Conceitos e Tipos de Marcadores Fiduciais 38
2.3 SLAM em Ambientes Internos 40
2.3.1 SLAM Baseado em Filtragem 40
2.3.2 SLAM Baseado em Otimização 43
2.4 Filtragem Robusta para Sistemas Incertos 44
2.5 Considerações Finais 46

3 CONCEITOS BÁSICOS 49
3.1 Marcadores Fiduciais ArUco 49
3.1.1 Composição 49
3.1.2 Processo de Detecção 51
3.2 Sensores 52
3.2.1 Encoders 52
3.2.2 Câmeras 54
3.2.3 Inertial Measurement Unit 55
3.2.4 Light Detection and Ranging 55
3.3 Localização e Mapeamento Simultâneos 56

3.4 Plataformas de Simulação . . . 58
3.4.1 Matlab . . . 58
3.4.2 Rviz . . . 58
3.4.3 Gazebo . . . 59

4 REKF-SLAM FRAMEWORK . . . 63
4.0.1 Robust Extended Kalman Filter . . . 64
4.0.2 Modelo de Propagação . . . 66
4.0.3 Modelo de Medição . . . 69
4.0.4 Estado Aumentado do REKF-SLAM . . . 71
4.0.5 Inicialização dos Marcadores . . . 72
4.0.6 Atualização Robusta das Medições e Implementação . . . 73
4.0.7 Implementação . . . 75

5 RESULTADOS . . . 81
5.0.1 Validação por Simulação . . . 82
5.0.2 Validação Experimental . . . 88

6 CONCLUSÕES E TRABALHOS FUTUROS . . . 95
6.1 Produção Intelectual . . . 96
6.2 Limitações e Dificuldades . . . 96
6.3 Contribuições . . . 98
6.4 Trabalhos Futuros . . . 98

REFERÊNCIAS . . . 101

APÊNDICES 111

APÊNDICE A – CALIBRAÇÃO DA CÂMERA . . . 113
A.1 Modelo da câmera linear . . . 113
A.1.1 Modelo de Câmera Não-Linear . . . 116
A.2 Distorção Radial . . . 116
A.3 Distorção Tangencial . . . 117
A.4 Calibração Utilizando um Tabuleiro de Xadrez . . . 119

APÊNDICE B – ROBOT OPERATING SYSTEM . . . 121
B.1 Estrutura de Funcionamento do ROS . . . 121
B.2 ROS 2 . . . 123

Capítulo 1

Introdução

Os robôs móveis autônomos, do inglês *Autonomous Mobile Robots* (AMR), têm se tornado cada vez mais relevantes em ambientes internos em razão dos avanços recentes em sensoriamento, computação embarcada, controle e navegação autônoma. Diferentemente dos veículos guiados automatizados tradicionais, que normalmente dependem de infraestrutura fixa ou de trajetórias predefinidas, os AMRs são projetados para operar em ambientes dinâmicos e compartilhados com seres humanos, utilizando capacidades embarcadas de percepção, tomada de decisão e planejamento de movimento. Essa flexibilidade tem possibilitado sua aplicação em uma ampla variedade de cenários internos, incluindo manufatura, armazéns, intralogística, instalações de *cross-docking*, hospitais e ambientes de serviços (ZACHARIAE et al., 2024; NAM et al., 2023; RIMON et al., 2025). Nesses contextos, os robôs móveis podem reduzir a carga associada a tarefas repetitivas ou fisicamente exigentes, auxiliar no transporte de materiais, aumentar a eficiência operacional e permitir que trabalhadores humanos se concentrem em atividades que demandam tomada de decisão em nível mais elevado (HERCIK et al., 2022).

Para que esses robôs operem de forma segura e confiável, é fundamental que sejam capazes de estimar continuamente sua pose em relação ao ambiente. A localização fornece as informações espaciais necessárias para navegação, planejamento de trajetórias, desvio de obstáculos e execução autônoma de tarefas. No entanto, a localização em ambientes internos permanece desafiadora. Ao contrário de cenários externos, nos quais sistemas globais de navegação por satélite podem frequentemente ser utilizados, ambientes internos são geralmente privados de sinal de GPS devido à atenuação e ao bloqueio de propagação. Assim, robôs móveis que operam nesses ambientes dependem principalmente de estratégias de localização baseadas em sensores embarcados, como odometria das rodas, sensoriamento inercial, correspondência de varreduras baseada em LiDAR e marcadores

visuais (ZHANG et al., 2023; TENG et al., 2024; CHAI; LI; LI, 2023).

Nesse contexto, a Localização e Mapeamento Simultâneos, ou SLAM, constitui uma capacidade central para AMRs, pois permite que o robô estime sua própria pose enquanto constrói incrementalmente uma representação do ambiente. Sistemas modernos de SLAM geralmente exploram modalidades sensoriais heterogêneas, incluindo sensores baseados em *Light Detection and Ranging* (LiDAR) e sensores baseados em visão, a fim de obter uma estimação de estados mais confiável (KANG et al., 2025; NAM et al., 2024; BAI; XING; LIU, 2023). Apesar desses avanços, ambientes internos estruturados, como edifícios de escritórios, armazéns e instalações industriais, ainda apresentam limitações relevantes para a percepção robótica. Esses locais frequentemente contêm corredores longos, padrões geométricos repetitivos, superfícies com pouca textura e condições inadequadas de iluminação, fatores que podem degradar tanto características visuais quanto informações de profundidade (DHAIGUDE; KULKARNI, 2025; LIN; YEH, 2022).

Além das limitações perceptivas do ambiente, os sensores utilizados na localização apresentam fontes específicas de erro. A odometria das rodas e as Unidades de Medição Inercial, *Inertial Measurement Unit* (IMU), fornecem estimativas contínuas de movimento, mas estão inerentemente sujeitas à deriva cumulativa (JIANG et al., 2024). Abordagens baseadas em LiDAR podem mitigar parte desse problema ao fornecer informações geométricas do ambiente; entretanto, o custo de sistemas LiDAR 3D de alta resolução e as limitações de campo de visão vertical de scanners 2D podem restringir sua aplicação prática, dependendo dos requisitos operacionais (LIM; LEE, 2009; MUÑOZ-SALINAS; MEDINA-CARNICER, 2020; SANCHEZ-LOPEZ et al., 2017). Consequentemente, alcançar uma localização robusta e globalmente consistente em cenários internos permanece um desafio relevante.

Diante dessas limitações, sistemas de localização em ambientes internos frequentemente recorrem a estruturas probabilísticas de fusão sensorial, capazes de combinar fontes heterogêneas de informação e, ao mesmo tempo, representar explicitamente as incertezas associadas às medições e ao movimento do robô (LIU et al., 2022). Essa abordagem é particularmente importante em SLAM, uma vez que a pose do robô e a representação do ambiente precisam ser atualizadas continuamente a partir de dados ruidosos, incompletos e provenientes de sensores com diferentes características de erro.

Nesse cenário, métodos recursivos de estimação de estados tornaram-se componentes fundamentais das arquiteturas modernas de localização e SLAM. A estimação de estados em sistemas estocásticos não lineares é comumente formulada por meio de técnicas recursivas de filtragem Bayesiana, entre as quais as abordagens baseadas no filtro de Kalman permanecem amplamente adotadas (FANG et al., 2022; TAO et al., 2021). Na robótica móvel, variantes como o Filtro de Kalman Estendido (do inglês, EKF), são frequentemente empregadas devido à sua estrutura recursiva, à eficiência computacional e à capacidade de incorporar modelos não lineares de movimento e observação por meio de linearizações

locais.

1.1 Caracterização do Problema

Embora estimadores convencionais baseados no filtro de Kalman sejam amplamente utilizados em localização e SLAM, eles pressupõem que o modelo em espaço de estados e a caracterização estatística dos ruídos de processo e de medição são conhecidos com precisão. Em aplicações práticas, essas hipóteses raramente são plenamente satisfeitas, devido a imprecisões de modelagem, erros residuais de calibração, dinâmicas não modeladas, degradação dos sensores e perturbações ambientais.

Como consequência, incertezas nos modelos do sistema e de observação podem deteriorar a consistência do filtro e comprometer a precisão da estimação. Esse efeito torna-se particularmente relevante em problemas não lineares de SLAM, nos quais as etapas de predição e correção dependem de linearizações locais em torno da estimativa corrente. Assim, pequenas discrepâncias entre o modelo nominal e o comportamento real do sistema podem ser propagadas ao longo do processo recursivo, afetando a confiabilidade da estimação. Essas limitações motivam a investigação de estratégias de filtragem robusta capazes de reduzir a sensibilidade do estimador a incertezas paramétricas, erros de modelagem e caracterizações imperfeitas dos ruídos.

Nesse contexto, a filtragem de Kalman robusta tem emergido como uma alternativa para reduzir a sensibilidade de estimadores recursivos a incertezas de modelo, caracterizações imperfeitas dos ruídos e medições não confiáveis. Esse aspecto é particularmente relevante em problemas de EKF-SLAM, nos quais erros de linearização, inicialização inadequada de covariâncias e condições sensoriais incertas podem comprometer a consistência tanto na etapa de predição quanto na correção baseada em marcadores visuais (GUSRIAL et al., 2025). Desafios semelhantes também ocorrem na estimação não linear de estados veiculares, em que incertezas de modelo, perturbações e hipóteses gaussianas aproximadas podem degradar o desempenho de estimadores convencionais baseados no filtro de Kalman (QI; FAN; LI, 2022). Portanto, a filtragem robusta de Kalman fornece uma estrutura adequada para melhorar a consistência do estimador sob incertezas paramétricas limitadas.

Mais recentemente, *frameworks* robustos de navegação multissensorial têm incorporado indicadores de confiabilidade dos sensores ao processo de estimação. Esses algoritmos permitem a rejeição de medições corrompidas e melhoram o desempenho da navegação em ambientes com maior incerteza (CALDAS et al., 2025; DOMINGUEZ; KEATON; SAYED, 2006; INOUE; TERRA; JR., 2008). Paralelamente, abordagens de filtragem de Kalman globalizadas e robustas em relação à distribuição têm sido desenvolvidas para mitigar os efeitos de discrepâncias no modelo probabilístico e de suposições estatísticas imprecisas. Essas abordagens reduzem a sensibilidade da estimação de estados às incer-

tezas nas distribuições de ruído subjacentes (XU et al., 2025). Apesar desses avanços, a integração da estimação recursiva robusta com o EKF-SLAM combinado com marcadores fiduciais sob incertezas paramétricas explícitas permanece relativamente pouco explorada, particularmente em cenários internos validados experimentalmente.

Diante desse cenário, este trabalho investiga o uso de uma estrutura robusta de estimação de estados aplicada ao problema de localização de robôs móveis em ambientes internos. O foco está no desenvolvimento de um *framework* capaz de integrar diferentes fontes de informação sensorial, como odometria das rodas, marcadores fiduciais ArUco e estimativas de pose provenientes do AMCL, em uma formulação de EKF-SLAM baseada no Filtro de Kalman Robusto. A proposta busca atender aplicações nas quais a localização precisa permanecer confiável mesmo sob incertezas de modelo, medições ruidosas, degradação sensorial e condições ambientais não ideais, como ocorre em armazéns, hospitais, laboratórios, ambientes industriais e instalações de serviços.

1.2 Objetivos

Este trabalho tem como objetivo principal desenvolver e avaliar um *framework* robusto de localização para robôs móveis autônomos em ambientes internos estruturados, combinando o problema de SLAM com uma formulação baseada no Filtro de Kalman Robusto. A proposta busca melhorar a precisão, a consistência e a resiliência da estimação de pose em cenários nos quais a localização é afetada por deriva odométrica, incertezas de modelagem, ruídos de medição, degradação sensorial e limitações perceptivas típicas de ambientes internos.

O *framework* proposto é voltado para aplicações em que robôs móveis precisam operar de forma autônoma e confiável em espaços fechados, como armazéns, instalações industriais, hospitais, laboratórios, edifícios de escritórios, ambientes de serviços e sistemas de intralogística. Nesses cenários, a ausência de sinal de GPS, a presença de corredores longos, geometrias repetitivas, obstáculos dinâmicos, superfícies com pouca textura e condições variáveis de iluminação podem comprometer o desempenho de métodos convencionais de localização. Assim, a abordagem desenvolvida visa fornecer uma solução adequada para contextos nos quais a robustez da estimação é tão importante quanto a precisão nominal do estimador.

Para isso, o trabalho propõe uma arquitetura multissensorial que integra incrementos de odometria das rodas, observações de marcadores fiduciais ArUco e estimativas auxiliares de pose fornecidas pelo AMCL. Nessa arquitetura, a odometria fornece informações incrementais de movimento, os marcadores ArUco atuam como referências visuais artificiais distribuídas no ambiente, e o AMCL contribui com uma estimativa global de pose obtida a partir de um mapa previamente construído. A fusão dessas fontes permite explorar informações complementares: continuidade temporal da odometria, restrições

espaciais absolutas dos marcadores fiduciais e referência global baseada em mapa.

A principal contribuição do trabalho consiste em estender a formulação RKF proposta por (ROCHA; TERRA, 2021), originalmente desenvolvida para sistemas lineares discretos no tempo sujeitos a incertezas paramétricas, para uma estrutura não linear de EKF-SLAM aplicada à localização de robôs móveis. Essa extensão permite representar explicitamente incertezas associadas tanto ao modelo de movimento quanto aos modelos de medição, atribuindo estruturas de incerteza distintas para diferentes modalidades sensoriais. Dessa forma, o *framework* permite tratar de maneira diferenciada as incertezas provenientes da odometria das rodas, das observações dos marcadores ArUco e das estimativas de pose do AMCL.

Diferentemente de abordagens convencionais baseadas em EKF-SLAM, que dependem fortemente da correta caracterização estatística dos ruídos de processo e de medição, a formulação proposta busca reduzir a sensibilidade do estimador a erros de modelagem e a caracterizações imperfeitas das covariâncias. Com isso, o método é particularmente adequado para aplicações reais em que os sensores apresentam comportamentos não ideais, as condições ambientais variam ao longo da operação e os modelos nominais utilizados pelo filtro não representam perfeitamente o sistema físico.

Assim, este trabalho busca demonstrar que a incorporação de uma estrutura robusta ao problema de EKF-SLAM com marcadores fiduciais e medições auxiliares de pose pode melhorar a confiabilidade da localização em ambientes internos. A proposta pretende contribuir para o desenvolvimento de sistemas de navegação mais seguros e consistentes para robôs móveis autônomos, especialmente em aplicações práticas nas quais a perda de localização, a deriva acumulada ou a inconsistência da estimativa podem comprometer a execução autônoma de tarefas.

1.3 Organização do Trabalho

Este trabalho está organizado da seguinte forma: o Capítulo 2 apresenta a revisão bibliográfica. Em seguida, o Capítulo 3 apresenta os principais conceitos que fundamentam a metodologia proposta neste trabalho, o Capítulo 4 descreve o *framework* REKF-SLAM desenvolvido, o Capítulo 5 apresenta os resultados obtidos, e o Capítulo 6 descreve as conclusões e trabalhos futuros desta Dissertação.

Capítulo 2

Revisão bibliográfica

Neste capítulo são revisadas as principais abordagens de localização e fusão sensorial aplicadas a robôs móveis em ambientes internos, com ênfase nas limitações que motivam o desenvolvimento do *framework* robusto proposto nesta dissertação. O foco da revisão está em métodos capazes de estimar a pose do robô em cenários sem acesso ao *Global Navigation Satellite System* (GNSS), sujeitos à deriva odométrica, ambiguidades perceptivas, geometrias repetitivas, degradação sensorial e incertezas de modelagem. A partir dessa análise, busca-se evidenciar a necessidade de uma abordagem que combine a eficiência recursiva do EKF-SLAM, o uso de referências visuais artificiais por meio de marcadores ArUco, medições auxiliares de pose provenientes do AMCL e mecanismos de filtragem robusta para lidar explicitamente com incertezas paramétricas.

Em ambientes internos, a utilização de sistemas de navegação global por satélite (GNSS) torna-se inviável ou altamente imprecisa devido ao bloqueio parcial ou total dos sinais por estruturas físicas, como paredes e edificações. Além disso, ambientes internos estruturados frequentemente apresentam características desafiadoras, como corredores estreitos e repetitivos, obstáculos desconhecidos e condições adversas de iluminação, o que aumenta a complexidade do problema de localização.

Diante disso, a literatura apresenta uma ampla variedade de abordagens para a estimação de estados em robôs móveis, que podem ser organizadas em métodos clássicos e métodos modernos. Os métodos clássicos incluem abordagens probabilísticas recursivas amplamente consolidadas, como aquelas baseadas no Filtro de Kalman (do inglês, *Kalman Filter* (KF)) e suas extensões, enquanto os métodos modernos englobam formulações mais recentes que exploram estratégias de otimização global, representações gráficas e técnicas de maior robustez e escalabilidade.

Portanto, este capítulo aborda inicialmente os algoritmos clássicos de fusão sensorial

baseados no Filtro de Kalman e no Filtro de Partículas (do inglês, *Particle Filter* (PF)), apresentados na Seção 2.1. Em seguida, serão discutidas as abordagens baseadas no uso de marcadores fiduciais como fontes de referência absoluta para correção da deriva acumulada, na Seção 2.2. Posteriormente, são apresentados os algoritmos de Localização e Mapeamento Simultâneos (do inglês, SLAM), na Seção 2.3, contemplando tanto abordagens baseadas em filtragem quanto em otimização. Por fim, a Seção 2.4 é dedicada às técnicas de filtragem robusta, que buscam mitigar limitações conhecidas dos métodos tradicionais em cenários reais de aplicação.

2.1 Métodos Probabilísticos Recursivos para Localização

Segundo (HALL; LLINAS, 1997), as técnicas de fusão de dados combinam as medidas adquiridas por diversos sensores para obter maior precisão do que seria possível usando apenas um deles. Para traduzir as entradas de medida de cada sensor de maneira confiável, é possível aplicar algoritmos probabilísticos de estimação de estado, que ajudam a prever o estado de um sistema que muda constantemente com base em observações e medições. Dessa forma, os principais algoritmos utilizados para realizar a estimação de estados na literatura são os algoritmos baseados no Filtro de Kalman e no Filtro de Partículas, que serão apresentados nas Subseções seguintes.

2.1.1 Métodos Baseados no Filtro de Kalman

A estimação de estados, também conhecida como filtragem, é fundamental para muitos sistemas de controle (ANDERSON; MOORE, 1979). O problema consiste em estimar o estado de um sistema dinâmico com base em dados de medição sujeitos a ruídos e incertezas. Dada sua importância, a estimação de estados tem sido extensivamente estudada ao longo das últimas décadas, especialmente após o início da década de 1960, quando o filtro de Kalman foi originalmente introduzido (Kalman, 1960). Nesse contexto, o KF tornou-se uma das abordagens mais populares e amplamente utilizadas para estimação recursiva de estados.

O KF estima o estado de processos em tempo discreto descritos por modelos estocásticos lineares de transição de estado e de medição. Entretanto, grande parte dos problemas em robótica envolve sistemas não lineares, nos quais a cinemática do robô, os modelos de observação e a relação entre sensores e estados não podem ser representados adequadamente por equações lineares. Nesses casos, uma extensão amplamente utilizada é o EKF, que aproxima localmente os modelos não lineares por meio de linearizações em torno da estimativa corrente do estado. Devido à sua estrutura recursiva e ao seu baixo custo

computacional em comparação com métodos mais complexos, o EKF é frequentemente empregado em aplicações de fusão de sensores e localização robótica.

Nesse sentido, em (HOUSEIN et al., 2022), os autores empregam o Filtro de Kalman Estendido EKF como método de localização para robôs móveis operando em ambientes internos, explorando a fusão de informações provenientes da odometria e de sensores LiDAR. A integração desses sensores por meio do EKF permite combinar o modelo de movimento do robô com medições exteroceptivas, resultando em estimativas de posição mais precisas e estáveis. Os resultados experimentais indicam que a aplicação do EKF proporciona uma melhoria significativa na estimativa da pose do robô quando comparada ao uso exclusivo da odometria bruta. Entretanto, os experimentos foram conduzidos em um ambiente de dimensões limitadas e com a utilização de um número restrito de marcadores visuais, o que pode restringir a generalização dos resultados para cenários mais extensos ou complexos.

Em (EMAN; RAMDANE, 2020), os autores também empregam o Filtro de Kalman Estendido para aprimorar a localização de robôs móveis em ambientes internos, considerando explicitamente modelos de movimento e observação não lineares. A avaliação do algoritmo é realizada por meio de experimentos conduzidos sob três condições distintas de ruído: ausência de ruído, presença de ruído com distribuição gaussiana e presença de ruído com distribuição não gaussiana. Os resultados indicam que no cenário sem ruído o algoritmo é capaz de estimar a trajetória do robô de forma exata. No caso de ruído gaussiano, o EKF apresenta bom desempenho, com erros de estimativa relativamente reduzidos. Já no cenário com ruído não gaussiano, embora o método ainda produza resultados aceitáveis, observa-se a ocorrência de desvios mais significativos em relação à trajetória real do robô, evidenciando as limitações do EKF frente a violações da hipótese de gaussianidade do ruído.

Além disso, em (LI et al., 2019), os autores implementam um sistema de fusão sensorial baseado no EKF, combinando informações de odometria provenientes de encoders nas rodas do robô com odometria visual obtida a partir de uma câmera monocular que detecta dois marcadores fiduciais. Essa integração permite explorar a complementaridade entre sensores proprioceptivos e exteroceptivos no processo de estimativa da pose. A abordagem proposta busca mitigar a acumulação de erros típica da odometria baseada em encoders, bem como a baixa confiabilidade da odometria visual em ambientes sujeitos a variações de iluminação e com poucos recursos visuais. Os resultados apresentados indicam que a fusão dessas fontes de informação por meio do EKF resulta em uma melhoria substancial na precisão da localização do robô quando comparada ao uso isolado de cada sensor, demonstrando a eficácia do método no contexto de aplicações de SLAM em ambientes internos.

As aproximações não-lineares realizadas pelo EKF podem introduzir erros na estimativa da média e da covariância devido ao processo de linearização. Como alternativa, é

possível utilizar o *Unscented Kalman Filter* (UKF), que utiliza pontos sigma gerados de maneira determinística para representar a distribuição de probabilidade do estado. Esses pontos são ajustados com base na média e na covariância estimadas e são propagados através das funções não-lineares para capturar a média e a covariância da distribuição transformada. O UKF é sensível à precisão das medidas de entrada e ao modelo de ruído, que é assumido Gaussiano. Dessa forma, medidas incorretas ou ruídos não uniformes podem afetar negativamente a estimativa do filtro. (ALI et al., 2022) utiliza o UKF para obter a localização de um robô móvel em ambientes internos a partir de um modelo de incerteza de medidas que corrige a covariância do erro de acordo com as medidas de distância. Além disso, o autor redefine os componentes da matriz de covariância do erro, com exceção dos elementos que estão na diagonal, para o valor zero. Essa alteração é feita considerando a influência mútua nas informações da velocidade linear e rotacional, que são utilizadas para estimativa e para obter as distâncias medidas pelos sensores. (ZHOU et al., 2022) aplica o UKF para obter a estimativa da posição do robô a partir da fusão de sensores como a câmera com profundidade, IMU, encoder, e LiDAR 2D. A partir dos resultados experimentais, esse método melhorou efetivamente a acurácia da estimativa de posição do robô, e o uso da fusão de sensores conseguiu garantir efetivamente a estabilidade do sistema.

Em alguns estudos como em (WAN; MERWE, 2000), (CHEN et al., 2019), e (ZHOU et al., 2022), são feitas comparações de desempenho e acurácia de estimativa entre o EKF e o UKF. Apesar da exigência reduzida de poder computacional no EKF, ele utiliza a expansão em série de Taylor em torno da estimativa atual, com uma aproximação linear de primeira ordem que é aplicada nas funções estado e observação. Essa expansão resulta na linearização do modelo de forma analítica, e pode levar a erros e resultados imprecisos, pois essa aproximação é válida apenas localmente e pode não capturar adequadamente as dinâmicas do sistema em regiões com alta não-linearidade. Para contornar essa problemática, o UKF faz a linearização de maneira estatística, utilizando um conjunto de pontos sigma que são escolhidos de maneira determinística e são projetados de forma a capturar a média e a covariância do estado do sistema. Esses pontos são propagados através das funções não-lineares utilizando uma técnica chamada *Unscented Transformation* (UT), que é projetada para capturar a média e a covariância até a segunda ordem de não-linearidade, de forma mais precisa do que o EKF, reduzindo os erros de aproximação. Dessa forma, o UKF consegue capturar não-linearidades de ordem superior de forma mais precisa, mas ainda assim, podem haver erros se as não-linearidades forem extremas. Como resultado, o UKF obtém resultados melhores de estimativa com a redução de erros. Entretanto, ele exige um poder computacional maior em comparação com o EKF.

Com o objetivo de alcançar um compromisso adequado entre custo computacional e precisão na estimação de estado, alguns trabalhos têm explorado a combinação do Filtro de Kalman Estendido e do Filtro de Kalman Unscented. Em (YATIGUL et al.,

2024), os autores propõem um sistema de localização para robôs móveis operando em ambientes internos que integra essas duas técnicas de filtragem, explorando as vantagens de cada abordagem. O sistema combina a eficiência computacional do EKF com a melhor capacidade do UKF em lidar com não linearidades mais acentuadas, resultando em um método de localização mais robusto e preciso. A abordagem realiza a fusão de informações provenientes de câmeras, encoders e sensores IMU, demonstrando melhorias consistentes na qualidade das estimativas de pose em comparação com o uso isolado de um único filtro.

De forma semelhante, (BAHAROM et al., 2020) apresenta um algoritmo de localização robusta de baixo custo e com precisão em nível de centímetros para robôs móveis autônomos em ambientes internos e externos. O trabalho proposto também realiza a fusão de dados de múltiplos sensores, além de aplicar algoritmos de SLAM e outros algoritmos de localização, como a Localização Monte Carlo (do inglês, *Monte Carlo Localization* (MCL)), além de combinar o EKF e o UKF para estimar a posição, velocidade e a atitude. Nesse caso, o UKF demonstrou melhor desempenho em convergência e precisão da estimativa em comparação com o EKF.

Assim, os métodos baseados no Filtro de Kalman permanecem relevantes para aplicações de localização em tempo real devido à sua estrutura recursiva e ao custo computacional reduzido. Entretanto, sua dependência de modelos nominais, covariâncias bem caracterizadas e hipóteses estatísticas aproximadas evidencia a necessidade de extensões robustas quando aplicados a cenários reais de navegação *indoor*.

2.1.2 Métodos baseados no Filtro de Partículas

O Filtro de Partículas, também conhecido como filtro de Bayes não paramétrico, constitui-se como uma abordagem de inferência Bayesiana que aproxima a distribuição a posteriori do estado por meio de um conjunto finito de amostras, denominadas partículas. Diferentemente dos filtros paramétricos, como o Filtro de Kalman, essa abordagem não impõe hipóteses explícitas sobre a forma da distribuição de probabilidade, permitindo uma representação mais flexível do estado.

Nesse método, a distribuição a posteriori é representada por um conjunto de amostras aleatórias do espaço de estados. Essa representação por amostragem possibilita a modelagem eficiente de sistemas altamente não lineares e de ruídos não gaussianos, constituindo uma das principais vantagens do Filtro de Partículas em aplicações de localização e SLAM (THRUN; BURGARD; FOX, 2005).

2.1.2.1 Localização Monte Carlo

A Localização de Monte Carlo constitui uma alternativa às técnicas clássicas baseadas em filtragem bayesiana paramétrica. Em contraste com métodos baseados no Filtro de Kalman, a MCL é capaz de representar distribuições de probabilidade multimodais, o

que possibilita a localização global do robô mesmo em situações de incerteza elevada ou ambiguidade perceptiva (DELLAERT et al., 1999).

Nessa abordagem, a crença sobre a pose do robô é representada por um conjunto de centenas ou milhares de partículas, cada uma correspondendo a uma hipótese possível de estado. Uma das principais vantagens da MCL é a redução significativa do consumo de memória quando comparada aos métodos de Localização de Markov baseados em grades, ao mesmo tempo em que mantém elevada precisão na estimativa da pose. Dessa forma, a MCL combina a robustez dos métodos probabilísticos globais com a eficiência computacional típica de abordagens baseadas em amostragem, posicionando-se como uma solução amplamente adotada para localização robótica em mapas conhecidos.

Em (SUN, 2022), é apresentado um estudo comparativo entre a localização estimada por meio da odometria e a obtida pelo algoritmo de MCL para robôs móveis operando em um mapa previamente conhecido. O estudo adota a metodologia de variáveis de controle, na qual ambos os métodos são executados simultaneamente sob as mesmas condições experimentais e utilizando os mesmos parâmetros de entrada, permitindo a comparação direta com a pose real do robô. Os resultados experimentais indicam que o MCL apresenta maior precisão na estimativa de localização quando comparado à odometria pura.

Em abordagens mais recentes, como em (ZHANG; SU, 2024), os autores utilizam a Localização de Monte Carlo (MCL) como método de localização para robôs móveis em ambientes internos e propõem aprimoramentos em relação à formulação clássica. A abordagem introduz modificações tanto na representação do mapa quanto no mecanismo de atualização das medições, com o objetivo de aumentar a precisão e a estabilidade da localização. Especificamente, em substituição aos mapas de grade de ocupação tradicionalmente empregados no MCL, são utilizados mapas baseados em Normal Distribution Transform (NDT), que oferecem uma representação mais compacta e contínua do ambiente. Além disso, os autores propõem ajustes no processo de atualização da ocupação para reduzir o impacto de interferências dinâmicas, como a presença de objetos móveis no cenário. Os resultados experimentais demonstram que o método proposto supera o MCL clássico, apresentando melhorias consideráveis tanto na precisão quanto na estabilidade da estimativa de pose, evidenciando o potencial do uso de mapas *Normal Distributions Transform* (NDT) e de estratégias adaptativas para localização robusta em ambientes internos.

2.1.2.2 Localização Monte Carlo Adaptativa

O algoritmo de Localização de Monte Carlo Adaptativa (do inglês, AMCL) é uma evolução do algoritmo de Localização de Monte Carlo, introduzindo um mecanismo de ajuste dinâmico do número de partículas utilizadas na representação da crença do estado. Nessa abordagem, um número reduzido de amostras é utilizado quando a distribuição de probabilidade da posição do robô está concentrada em uma região restrita do espaço de

estados, enquanto um maior número de partículas é utilizado em situações com maior incerteza.

Esse ajuste adaptativo é realizado por meio da técnica de KLD-Sampling (do inglês, *Kullback–Leibler Divergence* (KLD)) (FOX, 2003), que permite um controle maior do número de partículas, garantindo um compromisso entre precisão e custo computacional. Apesar dessas vantagens, o AMCL apresenta limitações conhecidas. Em particular, o intervalo de ajuste do número de partículas é restrito e nem sempre resulta em melhorias significativas de precisão. Além disso, quando ocorre uma convergência inicial para uma pose incorreta, a recuperação da localização correta pode tornar-se difícil. Estratégias que envolvem múltiplas execuções do AMCL tendem a mitigar esse problema, porém requerem maior esforço computacional.

Com o objetivo de superar essas limitações, trabalhos mais recentes têm proposto aprimoramentos no AMCL. Em (HE et al., 2023b), os autores apresentam uma abordagem voltada à localização de robôs móveis em ambientes dinâmicos e com baixa densidade de características perceptivas, alcançando desempenho superior ao AMCL clássico em termos de precisão e robustez. Em (YU; LI; PAN, 2021), o AMCL é integrado com marcadores fiduciais AprilTag, permitindo a incorporação de referências absolutas no ambiente. Os resultados indicam melhorias significativas na precisão e na estabilidade da localização quando comparado ao AMCL tradicional. De forma semelhante, (CHUNG; LIN, 2022) aborda o problema de localização em mapas com alta similaridade e simetria por meio da execução múltipla do AMCL, analisando diferentes hipóteses de pose e selecionando a estimativa mais consistente, o que resulta em aumento da precisão em relação ao método clássico.

Nesse contexto, o AMCL apresenta-se como uma fonte auxiliar relevante para sistemas de localização baseados em mapas previamente construídos. No entanto, sua utilização isolada pode ser limitada por ambiguidades perceptivas, convergência incorreta ou degradação do modelo de medição. Por essa razão, neste trabalho, suas estimativas são incorporadas como medições auxiliares em uma arquitetura de fusão sensorial, e não como única fonte de localização global.

2.2 Marcadores Fiduciais como Referências Visuais Artificiais

Uma alternativa amplamente empregada para lidar com ambientes com poucas características visuais ou com variações moderadas de iluminação é a utilização de marcadores fiduciais. Esses marcadores consistem em padrões visuais especificamente projetados para serem detectados e identificados de forma confiável por câmeras, possibilitando a estimação da pose relativa entre o robô e referências conhecidas no ambiente. Quando as posições dos marcadores são previamente conhecidas ou incorporadas ao mapa, essas observações

podem ser utilizadas como referências espaciais para corrigir a pose do robô em ambientes internos.

2.2.1 Conceitos e Tipos de Marcadores Fiduciais

Historicamente, uma das primeiras e mais influentes abordagens para rastreamento baseado em marcadores fiduciais foi o ARToolKit (KATO; BILLINGHURST, 1999). Esse sistema utiliza marcadores visuais compostos por uma borda externa preta e por padrões internos distintos, permitindo sua detecção e identificação por câmeras. Embora tenha sido amplamente utilizado em diversas aplicações, o ARToolKit apresenta algumas limitações. Entre elas, destacam-se a ocorrência de falsos positivos e a possibilidade de confusão entre marcadores durante o processo de detecção. Além disso, em muitas aplicações, o usuário precisa capturar imagens dos marcadores sob condições específicas de câmera e iluminação, bem como ajustar manualmente o limiar de escala de cinza utilizado na segmentação da imagem. Esses parâmetros devem ser definidos considerando o compromisso entre taxa de detecção e ocorrência de falsos positivos. Como consequência, apenas os marcadores que podem ser adequadamente segmentados sob o limiar previamente definido tendem a ser detectados de forma confiável. Outra limitação é que a distinção entre os marcadores pode diminuir à medida que o tamanho da biblioteca aumenta, aumentando a possibilidade de identificação incorreta.

Considerando essas limitações, o ARTag (FIALA, 2005) apresentou melhorias significativas de desempenho em relação ao ARToolKit. Embora tenha mantido a estrutura baseada em uma borda quadrada externa e em padrões internos para identificação, o ARTag introduziu uma abordagem mais robusta para o processamento do marcador. Seu método de detecção passou a explorar características geométricas associadas às bordas e aos cantos dos marcadores, reduzindo a dependência de limiares manuais de escala de cinza e aumentando a robustez em relação a variações de iluminação. No entanto, o ARTag também apresenta restrições. Sua estrutura de codificação é baseada em marcadores de 36 bits, permitindo a correção de, no máximo, dois bits errôneos. Além disso, essa capacidade de correção não é adaptada de acordo com a distância de Hamming, que compara códigos do mesmo tamanho, entre os marcadores efetivamente utilizados no conjunto, o que pode limitar a flexibilidade na construção de bibliotecas de marcadores.

O AprilTag (OLSON, 2011) foi lançado em seguida sob a licença *open-source*. Esse sistema incorporou avanços adicionais em relação ao ARTag, incluindo um algoritmo de segmentação de imagens baseado em grafos, que analisa padrões de gradiente com o objetivo de estimar linhas de forma mais precisa. Resultados experimentais demonstram que o AprilTag apresenta desempenho significativamente superior quando comparado ao ARToolKit e ao ARTag, como visto em (OLSON, 2011). Em seguida, foi proposto o STag (BENLIGIRAY; TOPAL; AKINLAR, 2019), um sistema voltado principalmente para a estabilidade das estimativas de posição. Sua principal distinção em relação aos

demais métodos reside na utilização de um padrão circular na região interna do marcador. A detecção desse padrão é realizada por meio de uma homografia refinada associada à técnica de *elliptical fitting*, o que resulta em maior precisão e estabilidade nas estimativas de localização.

Outro sistema amplamente utilizado é o ArUco, que também se baseia nos princípios do ARToolKit e do ARTag. Sua principal característica é a flexibilidade na geração e configuração de conjuntos de marcadores, permitindo ao usuário definir o tamanho e a quantidade de marcadores de acordo com a aplicação desejada. O algoritmo do ArUco realiza uma busca probabilística que maximiza simultaneamente a distância entre os marcadores de um conjunto e o número de transições de bits, aumentando a robustez da identificação. Além disso, o sistema propõe diferentes estratégias para lidar com oclusões, detectar automaticamente os marcadores e corrigir possíveis erros, sem recorrer ao uso de bits redundantes para detecção e correção, realizando a validação diretamente a partir do dicionário gerado. Em comparação com outros sistemas amplamente utilizados no estado da arte, o ArUco apresenta menor custo computacional e melhores resultados em termos de distância mínima entre marcadores e taxa de falsos positivos, conforme discutido em (GARRIDO-JURADO et al., 2014).

Em (KALAITZAKIS et al., 2020), é apresentada uma análise comparativa entre os principais sistemas de marcadores fiduciais discutidos neste trabalho. Os autores avaliaram o desempenho desses sistemas em aplicações de robótica e sistemas autônomos, considerando critérios como precisão de localização, robustez sob diferentes condições de iluminação, presença de desfoque de movimento e custo computacional. Os experimentos foram conduzidos utilizando imagens adquiridas sob diferentes rotações e distâncias. Os resultados indicam que os sistemas AprilTag, ArUco e STag apresentaram taxas superiores de detecção. Em particular, o AprilTag obteve melhor desempenho na estimativa da orientação, enquanto o STag apresentou maior precisão na estimativa da posição. Dessa forma, considerando as vantagens do ArUco em cenários oclusos e com baixa iluminação, bem como seu menor custo computacional e flexibilidade de configuração, esse método foi selecionado para esta pesquisa.

Em síntese, os marcadores fiduciais representam uma solução eficaz para fornecer referências espaciais para corrigir a pose do robô em ambientes internos, complementando sensores proprioceptivos e reduzindo a deriva acumulada em sistemas de localização. A escolha do sistema de marcadores impacta diretamente a robustez, o custo computacional e a precisão das estimativas de posição e orientação, tornando-se um componente relevante em arquiteturas de localização baseadas em fusão sensorial. Portanto, na próxima subseção, será apresentado como esses marcadores podem ser integrados a sistemas probabilísticos de localização, em particular nos algoritmos baseados no Filtro de Kalman.

Dessa forma, os marcadores ArUco são adotados neste trabalho como referências visuais artificiais capazes de fornecer restrições espaciais adicionais ao processo de estimação.

Entretanto, como sua detecção também está sujeita a ruídos, falhas de observação, erros de calibração e variações de iluminação, sua integração ao sistema de localização requer uma estrutura de fusão capaz de representar e tratar essas incertezas.

2.3 SLAM em Ambientes Internos

A técnica de SLAM permite que um robô estime sua própria posição ao mesmo tempo em que constrói um mapa do ambiente desconhecido, no qual está inserido. Essa estimativa é realizada a partir de um algoritmo que leva em consideração as equações que modelam as observações obtidas a partir dos sensores embarcados e a propagação do movimento do robô. A problemática do SLAM é fundamental na área de robótica móvel, uma vez que a localização e o mapeamento são fortemente interdependentes, visto que uma estimativa imprecisa da posição compromete a qualidade do mapa, enquanto um mapa inconsistente afeta diretamente a localização do robô.

De maneira geral, o problema de SLAM é formulado no contexto da estimativa probabilística, sendo comumente abordado por meio de técnicas de filtragem ou de suavização. As abordagens baseadas em filtragem são particularmente adequadas para a estimativa *on-line* da pose do robô e do mapa, uma vez que incorporam novas observações de forma recursiva à medida que o robô se desloca pelo ambiente. Entre os métodos mais utilizados nesse contexto destacam-se os que são baseados no Filtro de Kalman e no Filtro de Partículas.

Em contrapartida, as abordagens de suavização tratam o problema de SLAM como uma estimativa global, processando todo o conjunto de observações dos sensores ao longo do tempo. Essas abordagens formulam o SLAM como um problema de otimização, geralmente resolvido por meio de técnicas de mínimos quadrados e ajuste não linear (AL-SADIK; KARAM, 2021). Embora apresentem maior consistência global nas estimativas, esses métodos demandam maior consumo de memória e recursos computacionais, já que consideram explicitamente múltiplas poses do robô e suas correlações temporais, em contraste com os métodos de filtragem, que mantêm apenas a estimativa corrente do estado e sua incerteza associada. Um exemplo amplamente utilizado de abordagem baseada em suavização é o GraphSLAM, que será discutido nas próximas subseções.

2.3.1 SLAM Baseado em Filtragem

Uma das abordagens clássicas amplamente utilizadas é o SLAM baseado no Filtro de Kalman Estendido (do inglês, EKF-SLAM). Essa abordagem é derivada da formulação bayesiana de estimativa recursiva de estados e estrutura-se em dois estágios principais: predição e atualização por medições. A cada passo de tempo, os modelos não lineares que descrevem o movimento do robô e o processo de observação são aproximados por meio de

uma linearização local em torno do estado estimado corrente, utilizando a expansão de Taylor de primeira ordem.

Apesar de ser amplamente aceito, o EKF-SLAM apresenta limitações conhecidas. Em particular, os erros introduzidos pela linearização local podem se acumular ao longo do tempo, resultando em estimativas inconsistentes do estado. Além disso, à medida que o mapa cresce e o número de observações aumenta, o custo computacional do EKF-SLAM tende a se elevar significativamente, uma vez que a matriz de covariância conjunta do estado que relaciona o robô e o mapa cresce quadraticamente com o número de elementos estimados.

Ainda assim, diversos trabalhos recentes têm explorado o EKF-SLAM em conjunto com marcadores fiduciais como forma de mitigar algumas dessas limitações. Em (AFSHA; SAZID, 2024), é apresentada uma abordagem voltada para ambientes que não possuem sinal GNSS, na qual marcadores fiduciais são utilizados para fornecer referências absolutas no ambiente, resultando em desempenho confiável e maior precisão na localização em cenários controlados, além de manter baixo custo computacional. De forma semelhante, em (LIANG; STANCU; ZHANG, 2023), os autores avaliam o desempenho do EKF-SLAM em diferentes cenários experimentais e comparam seus resultados com o método *Squared Planar Markers* (SPM)-SLAM, uma abordagem baseada em otimização que utiliza exclusivamente informações visuais, obtendo melhorias significativas na acurácia quando os marcadores fiduciais são incorporados.

Nesse contexto, a introdução de marcadores fiduciais contribui de maneira decisiva para a resolução do problema de associação de dados, uma vez que a geometria conhecida desses marcadores permite a estimativa direta da posição e orientação a partir das observações da câmera. Dessa forma, a fusão entre EKF-SLAM e marcadores fiduciais configura uma estratégia eficaz para reduzir ambiguidades perceptivas e limitar a propagação de erros em sistemas de localização robótica. Entretanto, um desafio do EKF-SLAM é que à medida que o número de estados aumenta, a matriz de covariância cresce quadraticamente. Isso pode levar a problemas de estabilidade numérica e aumentar significativamente os requisitos computacionais. Em situações com muitas características ou em grandes ambientes, o EKF-SLAM pode se tornar computacionalmente intensivo e menos eficiente.

Para lidar com a problemática de escalabilidade do EKF-SLAM, em (PEI; ZHU; WU, 2020), os autores propõem uma abordagem de EKF-SLAM distribuído baseada no Decorrelated Distributed Extended Kalman Filter (*Decorrelated Distributed EKF* (DDEKF)), cujo principal objetivo é mitigar os efeitos da correlação cruzada entre estimativas locais provenientes de diferentes subsistemas. Os resultados experimentais indicam que a abordagem distribuída baseada no DDEKF é capaz de fornecer estimativas mais precisas da trajetória contínua do robô quando comparada a sistemas de SLAM centralizados e a outras abordagens do estado da arte. Entretanto, uma limitação relevante do método

reside na sua sensibilidade a alterações dinâmicas na configuração dos subsistemas, uma vez que mudanças abruptas na estrutura da rede ou na disponibilidade de agentes podem comprometer a estabilidade do filtro e a consistência das estimativas.

Outro método baseado em filtragem é o Filtro de Partículas, que utiliza um conjunto de partículas aleatórias para representar a distribuição de probabilidade da posição e orientação de um robô (FOX et al., 2001). Nesse sentido, em (ALAM; GULLU; GUNES, 2024), os autores propõem uma abordagem de SLAM baseada na combinação de Filtro de Partículas com marcadores fiduciais ArUco para a localização de um robô móvel autônomo. Diferentemente de métodos que utilizam diretamente o vetor de rotação estimado pelo ArUco, que pode apresentar imprecisões devido a ruídos de medição e limitações geométricas, a abordagem proposta utiliza recursões de filtragem progressiva e suavização regressiva para estimar de forma mais robusta o vetor de direção do robô. Os resultados experimentais demonstram que o método proposto apresenta desempenho superior quando comparado a uma abordagem de linha de base que não utiliza o Filtro de Partículas, alcançando reduções significativas nos erros médios de posição e orientação. Além disso, a solução mostrou-se computacionalmente eficiente, com boa adaptabilidade a cenários contendo obstáculos e trajetórias com curvas complexas, evidenciando o potencial da integração entre SLAM probabilístico e marcadores fiduciais para aplicações em ambientes internos.

No trabalho de (ZHANG; SHAN; LIU, 2025), os autores propõem um algoritmo de SLAM baseado em filtragem de partículas com inferência aproximada de filtros de partículas Rao-Blackwellized (do inglês, *Rao-Blackwellized Particle Filter* (RBPF)) (GRISETTI; STACHNISS; BURGARD, 2007), aplicado à utilização de marcadores visuais. Nessa abordagem, a pose do robô e a localização dos marcadores são estimadas de forma desacoplada, explorando a fatoração do espaço de estados característica dos filtros RBPF. As distribuições probabilísticas que não podem ser tratadas associadas à pose do robô e aos marcadores no ambiente são aproximadas por distribuições Gaussianas ótimas, permitindo uma estimativa mais eficiente e consistente.

Para solucionar o problema de associação de dados, os autores usam uma abordagem baseada em máxima verossimilhança, explorando a geometria conhecida dos marcadores visuais para reduzir ambiguidades durante o processo de correspondência. Os resultados experimentais demonstram que o algoritmo proposto alcança melhor desempenho e maior precisão quando comparado ao TUFastSLAM (LIN; YANG; LI, 2019), considerado um método de referência no estado da arte para SLAM com RBPF, e à técnica de Localização e Mapeamento Rápido (do inglês, *Rapid Localization and Mapping* (RLAM)) (CHARROUD et al., 2022), que é um método baseado na extração de características a partir de dados de LiDAR. Embora o RLAM apresente robustez frente a associações de dados desconhecidas e erros de correspondência, os resultados indicam que o método proposto em (ZHANG; SHAN; LIU, 2025) obtém maior acurácia global. Entretanto,

observa-se que o tempo médio de execução do algoritmo é superior ao do TUFastSLAM e do RLAM, evidenciando um compromisso entre desempenho computacional e precisão na estimativa de posição.

2.3.2 SLAM Baseado em Otimização

Os sistemas de SLAM baseados em otimização formulam os problemas de localização e mapeamento como uma estimativa global, frequentemente denominados SLAM completo, ou no inglês, full SLAM. Nesse caso, as posições do robô e as observações associadas são estimadas de maneira conjunta. Diferentemente das abordagens baseadas em filtragem recursiva, esses métodos objetivam minimizar globalmente um critério de erro definido sobre todo o histórico de medições, o que tende a resultar em maior consistência das estimativas (ZHENG et al., 2023).

Uma das principais representações utilizadas nesse contexto é o Graph SLAM, no qual o problema é modelado por meio de uma estrutura de grafo ou grafo de fatores que representa a formulação bayesiana do sistema. Nessa abordagem, as variáveis de estado são representadas como nós, enquanto as restrições impostas pelas medições e pelos modelos de movimento são representadas por arestas ou fatores.

Em (ZHANG et al., 2024), os autores propõem um método de localização baseado na fusão de sensores utilizando um grafo de fatores com pesos adaptativos (do inglês, *Adaptive Weighted Factor Graph* (AWFG)), voltado para robôs móveis operando em ambientes sem disponibilidade de sinal GNSS. O objetivo da abordagem é lidar com a degradação das características geométricas do ambiente e com a natureza assíncrona das informações provenientes dos sensores IMU, câmera e LiDAR. Os resultados experimentais são comparados com o Filtro de Kalman Estendido e com métodos baseados em Otimização por Enxame de Partículas, demonstrando que a abordagem proposta alcança melhorias significativas na precisão do posicionamento e na estabilidade do sistema, além de apresentar redução considerável no tempo computacional.

Em (ZHAO et al., 2024), os autores propõem uma estrutura de localização global para robôs móveis baseada em grafos, com o objetivo de superar a baixa estabilidade observada em métodos tradicionais de localização durante a navegação em nível de objetos. A abordagem desloca o problema de localização do nível puramente geométrico ou visual para uma representação semântica e topológica do ambiente, explorando relações espaciais entre objetos detectados. Dessa forma, técnicas de extração de informações semânticas são utilizadas para construir um mapa em nível de objetos, e a partir dele, são extraídos grafos topológicos que representam as relações espaciais entre esses objetos. Esses grafos são posteriormente processados e combinados por meio de estratégias de extração e união de grafos, resultando em uma estrutura hierárquica em forma de árvore. A correspondência entre grafos locais e globais é então utilizada para estimar a pose do robô de forma robusta, mesmo sob grandes variações de ponto de vista. Os resultados experimentais

demonstram que a abordagem proposta alcança alta precisão na localização global, sendo comparada com extensões do ORB-SLAM2 e com o método PoseNet2. Em todos os cenários avaliados, o método apresentou os menores erros médios de translação e rotação. Além disso, os autores destacam que a solução é robusta a variações de iluminação e a mudanças significativas no ambiente, evidenciando a eficácia do uso de grafos semânticos em nível de objetos para tarefas de relocalização global em ambientes internos.

2.4 Filtragem Robusta para Sistemas Incertos

Uma limitação bem conhecida do Filtro de Kalman é a suposição de que o modelo em espaço de estados do sistema e as estatísticas dos ruídos de processo e medição são conhecidas com precisão, condição que raramente é plenamente satisfeita em aplicações reais. Com isso, incertezas paramétricas frequentemente surgem da linearização, de dinâmicas não modeladas, ou da redução de modelos ou de parâmetros variantes, podendo degradar consideravelmente o desempenho da estimação.

Entre as abordagens mais relevantes de estimação robusta de estados para sistemas sujeitos a incertezas paramétricas limitadas em norma encontradas na literatura estão a filtragem $H\infty$ e as estratégias robustas de mínimos quadrados regularizados.

O objetivo da filtragem $H\infty$ é minimizar a norma $H\infty$ do mapeamento entre as perturbações e o erro de estimação. Nos últimos anos, essa técnica tem sido amplamente aplicada para o contexto de robótica móvel e sistemas de SLAM. Em (SHREYA; MISHRA; MUKHERJEE, 2024), é proposta uma variação do FastSLAM utilizando o filtro $H\infty$ para melhorar a estimação de marcadores visuais. No FastSLAM clássico, a pose do robô é estimada pelo Rao-Blackwellized Particle Filter, enquanto os marcadores são estimados pelo EKF. Uma desvantagem é que o EKF depende de uma boa caracterização estatística dos ruídos de processo e medição. Quando essas informações estão incorretas ou são desconhecidas, o filtro pode perder consistência ou até divergir. Sendo assim, o $H\infty$ se torna uma alternativa robusta a incertezas de modelo, ruídos desconhecidos e conhecimento estatístico impreciso das perturbações.

Além disso, uma solução para a localização de robôs móveis baseada em marcadores é proposta em (FAJARDO et al., 2021), utilizando filtragem $H\infty$ formulada por *Linear Matrix Inequality* (LMI)s. As observações são baseadas em marcadores detectados por um LiDAR 2D, e é feita uma linearização semelhante ao que ocorre no EKF. Entretanto, a principal limitação é o custo computacional. Como o problema LMI precisa ser resolvido a cada iteração, o filtro pode se tornar computacionalmente oneroso em ambientes grandes ou com muitos marcadores.

Em (ORTEGA-CONTRERAS et al., 2023), é abordado o acúmulo de erro ao longo do tempo pela odometria, levando em consideração o ruído na medição da posição das rodas pelos encoders, escorregamento e derrapagem das rodas, e o erro sistemático no raio ou

largura da plataforma. Os autores comparam o Filtro de Kalman, o Colored Measurement Noise Kalman Filter e o Filtro robusto $H\infty$, que apresentou o melhor desempenho.

É possível formular o problema de estimação como um problema robusto de mínimos quadrados regularizados (SAYED; NASCIMENTO, 1999). O objetivo é minimizar, no pior caso, a norma residual regularizada sobre o conjunto de incertezas admissíveis. Em geral, os filtros resultantes são recursivos e se assemelham ao filtro de Kalman clássico, o que é conveniente para aplicações on-line. O primeiro trabalho a empregar essa abordagem em filtragem robusta foi (SAYED, 2001). Esse trabalho introduz o chamado filtro com incertezas de dados limitadas, ou do inglês, *Bounded Data Uncertainties* (BDU), o qual considera que o sistema está sujeito a incertezas paramétricas limitadas em norma.

No entanto, (XU; MANNOR, 2009) apontam que considerar o efeito de pior caso das incertezas em projetos baseados em mínimos quadrados pode levar a filtros excessivamente conservadores. Para contornar esse problema, os autores propõem um estimador que combina os filtros de Kalman e BDU. (ISHIHARA; TERRA; CERRI, 2015) apresentaram um filtro robusto em um arranjo matricial simétrico. O projeto proposto não depende de ajuste de parâmetros e assume que todas as matrizes de parâmetros dos modelos do sistema-alvo e de sensoriamento estão sujeitas a incertezas. Entretanto, esse método envolve a inversão de um grande bloco matricial a cada instante de tempo.

Em uma abordagem mais recente, (ROCHA; TERRA, 2021) desenvolve um Filtro de Kalman Robusto para sistemas lineares discretos sujeitos a incertezas paramétricas limitadas em norma. O filtro é derivado por meio de mínimos quadrados regularizados e função penalidade, de modo que o parâmetro de penalidade permite ajustar o compromisso entre desempenho nominal e robustez. Os autores também apresentam condições de convergência e mostram que o método pode manter bom desempenho mesmo em cenários de incerteza significativa.

A extensão para sistemas não lineares é discutida em (INOUE; TERRA; CERRI, 2016), que propõe um Filtro de Kalman Robusto Estendido para estimativa de atitude. O método combina regularização robusta e penalização em uma estrutura preditor-corretor, incorporando incertezas tanto no modelo de estados quanto no modelo de observação.

Mais recentemente, a filtragem robusta tem sido combinada com modelos sujeitos a saltos markovianos. Em (MARCOS; TERRA, 2020), um filtro robusto para sistemas lineares discretos com saltos markovianos é aplicado à estimativa de torção do eixo cardã em veículos pesados. O método lida com mudanças abruptas na dinâmica do sistema, como trocas de marcha, e com incertezas associadas à inclinação da via. Os resultados mostram que o filtro robusto pode produzir estimativas online precisas mesmo em cenários nos quais abordagens baseadas em LMIs não encontram solução factível.

Em (CALDAS et al., 2025), os autores propõem um Filtro de Kalman Robusto para sistemas lineares discretos com saltos markovianos, aplicado à fusão de sensores visuais, GPS e inerciais. A principal contribuição é selecionar, a cada instante, o modo marko-

viano mais adequado para representar a condição de operação dos sensores, reduzindo a degradação causada por incertezas paramétricas e medições corrompidas. Esse tipo de abordagem é especialmente relevante em navegação autônoma, na qual sensores como GPS e magnetômetros podem sofrer oclusões, interferências ou degradações temporárias.

2.5 Considerações Finais

A partir dos trabalhos revisados, observa-se que o problema de localização de robôs móveis em ambientes internos se mantém como um desafio na área de robótica móvel. Foram apresentados alguns trabalhos que propõem metodologias para a fusão sensorial baseada no Filtro de Kalman e no Filtro de Partículas, além de abordagens de SLAM e alternativas para observações em ambientes internos com baixa visibilidade. Por fim, foram apresentados trabalhos que visam mitigar as incertezas associadas aos sensores, a dinâmica do movimento, a estrutura do ambiente e as hipóteses assumidas pelos modelos de estimativa, as quais são comumente aproximadas em sistemas que utilizam o Filtro de Kalman.

Nesse sentido, os métodos clássicos baseados no Filtro de Kalman e em suas extensões continuam sendo amplamente utilizados em sistemas de localização robótica, devido à sua formulação recursiva, eficiência computacional e capacidade de realizar fusão de múltiplas fontes de informação. Em particular, o EKF apresenta-se como uma solução adequada para aplicações em tempo real, principalmente quando os modelos não lineares podem ser aproximados de forma satisfatória por linearizações locais. No entanto, sua dependência de modelos precisos, ruídos aproximadamente gaussianos e linearizações de primeira ordem pode comprometer a consistência das estimativas em cenários com incertezas mais severas, ruídos não gaussianos ou erros de modelagem. O UKF, por sua vez, melhora a propagação estatística de sistemas não lineares por meio da transformada *unscented*, mas exige maior esforço computacional e ainda depende da qualidade dos modelos probabilísticos adotados.

As abordagens baseadas em Filtro de Partículas, incluindo MCL e AMCL, oferecem maior flexibilidade para representar distribuições não gaussianas e multimodais, sendo particularmente úteis em problemas de localização global e em situações de elevada ambiguidade perceptiva. Entretanto, esses métodos podem demandar um número elevado de partículas para manter precisão adequada, o que aumenta o custo computacional. Além disso, a recuperação de poses incorretas e a sensibilidade à qualidade do modelo de medição ainda representam limitações relevantes, especialmente em ambientes dinâmicos ou com baixa densidade de características distintivas.

A utilização de marcadores fiduciais surge como uma estratégia complementar importante para reduzir a deriva acumulada por sensores proprioceptivos e fornecer referências absolutas no ambiente. Sistemas como AprilTag, STag e ArUco demonstram bom desem-

penho em aplicações de robótica móvel, principalmente em ambientes internos controlados. Entre essas alternativas, o ArUco destaca-se pela flexibilidade de configuração, baixo custo computacional e robustez em diferentes condições de detecção. Ainda assim, o desempenho de sistemas baseados em marcadores depende da visibilidade dos marcadores, da calibração da câmera, da geometria de instalação e da qualidade da estimativa de pose visual, fatores que podem introduzir incertezas adicionais no processo de localização.

No contexto de SLAM, as abordagens baseadas em filtragem, como o EKF-SLAM e métodos baseados em Filtro de Partículas, são atrativas para aplicações on-line, pois atualizam recursivamente a estimativa do robô e do mapa à medida que novas medições são recebidas. No entanto, o crescimento da dimensão do estado, a propagação de correlações entre robô e landmarks e os erros de linearização podem afetar a escalabilidade e a consistência das estimativas.

Diante dessas limitações, as técnicas de filtragem robusta baseadas em minimização de pior caso são uma alternativa relevante para lidar com incertezas de modelo e perturbações que não são adequadamente representadas pelas formulações probabilísticas nominais. Essa classe de métodos amplia a formulação tradicional do Filtro de Kalman ao substituir a dependência exclusiva de modelos nominais por uma estrutura capaz de considerar desvios estruturados nas matrizes do sistema e da observação.

A revisão apresentada evidencia que cada classe de método contribui parcialmente para o problema investigado nesta dissertação. Os filtros baseados em Kalman oferecem eficiência e formulação recursiva para fusão sensorial; os métodos baseados em partículas, como o AMCL, fornecem uma alternativa robusta para localização em mapas conhecidos; os marcadores fiduciais permitem introduzir referências visuais artificiais em ambientes internos; e os filtros robustos oferecem mecanismos para tratar incertezas de modelo e perturbações não adequadamente descritas por modelos probabilísticos nominais. A contribuição deste trabalho está na integração desses elementos em uma arquitetura única de localização robusta.

Portanto, a lacuna identificada na literatura está relacionada à necessidade de estratégias de localização que preservem a eficiência recursiva dos filtros baseados em Kalman, mas que apresentem menor sensibilidade a erros de modelagem, incertezas paramétricas e degradações nas medições. Embora o EKF-SLAM seja adequado para fusão sensorial em tempo real, sua consistência pode ser comprometida quando os modelos de movimento e observação não representam adequadamente o sistema real. Por outro lado, marcadores fiduciais e estimativas provenientes do AMCL podem fornecer restrições adicionais à pose do robô, mas também estão sujeitos a incertezas próprias. Essa combinação motiva o desenvolvimento de um *framework* robusto de localização para robôs móveis em ambientes internos, capaz de integrar odometria, observações ArUco e medições auxiliares de pose em uma estrutura não linear de EKF-SLAM baseada em filtragem robusta.

Capítulo 3

Conceitos Básicos

Este Capítulo explica os conceitos básicos sobre as principais ferramentas utilizadas neste trabalho. Na Seção 3.1 são apresentados os principais aspectos dos Marcadores fiduciais ArUco, como a sua composição, o processo de detecção e como é realizada a estimativa de pose. Em seguida, a Seção 3.2 aborda brevemente o funcionamento dos principais sensores utilizados a bordo dos robôs utilizados para a validação do framework proposto neste trabalho, e na Seção 3.3 introduzimos os elementos essenciais da técnica de SLAM, que também será parte fundamental desta pesquisa. Por fim, na Seção 3.4 apresentamos as principais ferramentas utilizadas para a realização de testes e simulações para validação do sistema.

3.1 Marcadores Fiduciais ArUco

Esta Seção explica o funcionamento dos marcadores fiduciais do tipo ArUco, que foram utilizados para corrigir a posição do robô ao longo da trajetória percorrida. Foram desenvolvidos algoritmos de detecção utilizando as linguagens C++ e Python, em conjunto com a biblioteca OpenCV.

3.1.1 Composição

Os marcadores ArUco são constituídos por uma borda externa preta e uma região interna responsável por codificar um padrão binário único, utilizado para a identificação individual de cada marcador. A quantidade de bits que compõe esse padrão varia de acordo com o dicionário selecionado na biblioteca. A Figura 1 apresenta exemplos de mar-

cadores ArUco pertencentes a dicionários de diferentes tamanhos, ilustrando a variação na complexidade dos padrões binários empregados.

Nesse sentido, a utilização de marcadores com maior número de bits aumenta a robustez do processo de detecção e identificação, reduzindo a probabilidade de erros e ambiguidades. No entanto, esse aumento implica uma maior exigência quanto à resolução da imagem e à qualidade das condições de aquisição, uma vez que padrões mais complexos demandam maior detalhamento espacial para serem corretamente identificados.

Figura 1 – Diferentes dicionários do marcador ArUco. Fonte: Dicionário ArUco.

Esse padrão binário facilita o processo de detecção pela câmera, uma vez que reduz a probabilidade de falsas detecções decorrentes da presença de outros padrões visuais no ambiente. O pacote ArUco disponibiliza uma biblioteca de código aberto que fornece um conjunto abrangente de funções para geração, detecção e identificação de marcadores fiduciais, fazendo uso de técnicas de visão computacional implementadas na biblioteca OpenCV[1]. As ferramentas oferecidas permitem a geração de diferentes conjuntos de marcadores, denominados dicionários, os quais são definidos de acordo com o contexto de aplicação. Cada dicionário agrupa marcadores que compartilham o mesmo tamanho e a mesma quantidade de identificadores, sendo esses parâmetros fixos dentro de um mesmo conjunto. Os marcadores podem ser definidos por matrizes binárias de dimensões 4×4, 5×5, 6×6 ou 7×7, enquanto os dicionários podem conter, tipicamente, 50, 100, 250 ou 1000 identificadores distintos, sendo que cada identificador está associado ao seu índice dentro do dicionário ao qual pertence, o que permite sua distinção inequívoca durante o processo de detecção e reconhecimento.

A Figura 2 ilustra o sistema de coordenadas padrão adotado para os marcadores ArUco. Nessa representação, o canto superior esquerdo do marcador é destacado por um quadrado, indicando a orientação do marcador no plano da imagem e permitindo a determinação de sua rotação. As coordenadas da posição relativa dos quatro vértices do marcador, definidas em relação ao centro geométrico, são denotadas por `p0`, `p1`, `p2` e `p3`. Esses pontos correspondem, respectivamente, aos cantos do marcador e são utilizados como referências no processo de estimação de pose, podendo ser definidos conforme a

[1] https://opencv.org/

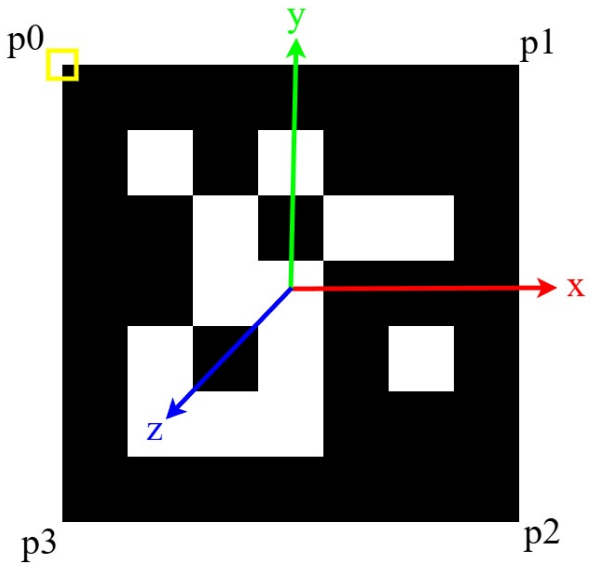

Figura 2 – Coordenadas do marcador ArUco. Fonte: elaborado pela autora.

seguir:

$$\mathrm{p}_0 = \begin{bmatrix} \frac{-s}{2} & \frac{s}{2} & 0 \end{bmatrix}^T, \tag{1}$$

$$\mathrm{p}_1 = \begin{bmatrix} \frac{s}{2} & \frac{s}{2} & 0 \end{bmatrix}^T, \tag{2}$$

$$\mathrm{p}_2 = \begin{bmatrix} \frac{s}{2} & \frac{-s}{2} & 0 \end{bmatrix}^T, \tag{3}$$

$$\mathrm{p}_3 = \begin{bmatrix} \frac{-s}{2} & \frac{-s}{2} & 0 \end{bmatrix}^T. \tag{4}$$

3.1.2 Processo de Detecção

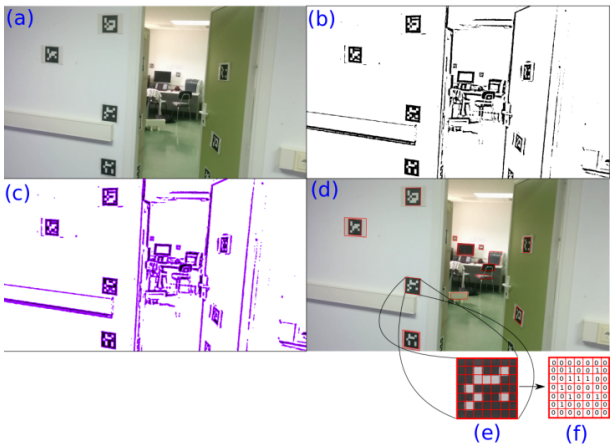

Figura 3 – Processo de detecção dos marcadores ArUco. Fonte: (**??**).

A Figura 3 mostra as etapas do processo de detecção dos marcadores ArUco. Inicialmente a imagem original é capturada pela câmera RGBD (Fig. 3a). Em seguida, a imagem passa por um processo de segmentação para identificar as bordas externas dos

marcadores, destacadas pela cor preta (Figura 3b). Seguindo o algoritmo proposto em (SUZUKI; ABE, 1985), são identificados os contornos nas imagens (Figuras 3c e 3d), e os detalhes irrelevantes no ambiente que não correspondem aos marcadores são removidos. Os contornos muito pequenos são retirados, e nos restantes aplica-se o algoritmo de (DOUGLAS; PEUCKER, 1973), que faz uma aproximação das figuras que mais correspondem a um polígono convexo de quatro cantos, sendo que as que não se assemelham o suficiente são descartadas. Em seguida, a projeção de perspectiva é removida ao computar a matriz de homografia, resultando na imagem representada em (Fig. 3e). Nessa imagem aplica-se o algoritmo de (OTSU, 1979), resultando em uma imagem binarizada (Fig. 3f) que é dividida em grades regulares, e cada elemento recebe um valor binário, seguindo a intensidade dos pixels contidos no interior de cada célula da grade. Para cada marcador identificado, é necessário determinar se ele pertence a um conjunto de marcadores válidos, e para tanto, obtêm-se quatro identificadores para cada marcador, que correspondem às quatro rotações possíveis da imagem canônica. Por fim, a localização dos cantos dos marcadores é estimada utilizando a resolução de subpixels, aplicando uma regressão linear dos pixels que compõem os cantos do marcador. Para maiores detalhes sobre o processo de identificação dos marcadores ArUco, veja (**??**). c

3.2 Sensores

Os sensores são fundamentais para obter informações acerca do ambiente em que o robô se encontra. Nesta Seção são abordados os conceitos fundamentais sobre alguns dos principais sensores utilizados em robótica móvel, sendo que a maioria deles também foram empregados nos robôs utilizados para a validação experimental do *framework*.

3.2.1 Encoders

O deslocamento do robô pode ser estimado por meio de encoders acoplados às rodas do robô móvel. Esses dispositivos fornecem saídas digitais correspondentes ao deslocamento linear ou angular das rodas, permitindo a inferência da movimentação do robô ao longo do tempo. Os encoders podem ser classificados em incrementais e absolutos: os primeiros detectam variações relativas na rotação a partir de uma posição de referência, enquanto os segundos fornecem diretamente a posição angular absoluta.

Apesar de ser amplamente utilizado, a odometria baseada em encoders apresenta limitações conhecidas. Esse método é suscetível ao acúmulo de erros decorrentes do drift, bem como ao deslizamento ou derrapagem das rodas, o que pode resultar em estimativas imprecisas, especialmente em terrenos irregulares ou superfícies com baixo atrito. Além disso, os erros de translação e orientação tendem a crescer proporcionalmente à distância total percorrida pelo robô, conforme discutido em (AQEL et al., 2016), reduzindo a confiabilidade da localização em aplicações de longo prazo. Para esta pesquisa, serão utilizados

os encoders incrementais, que medem o deslocamento angular do robô. A Figura 4 mostra o seu funcionamento básico. Nesse dispositivo, um feixe de luz atravessa as fendas do

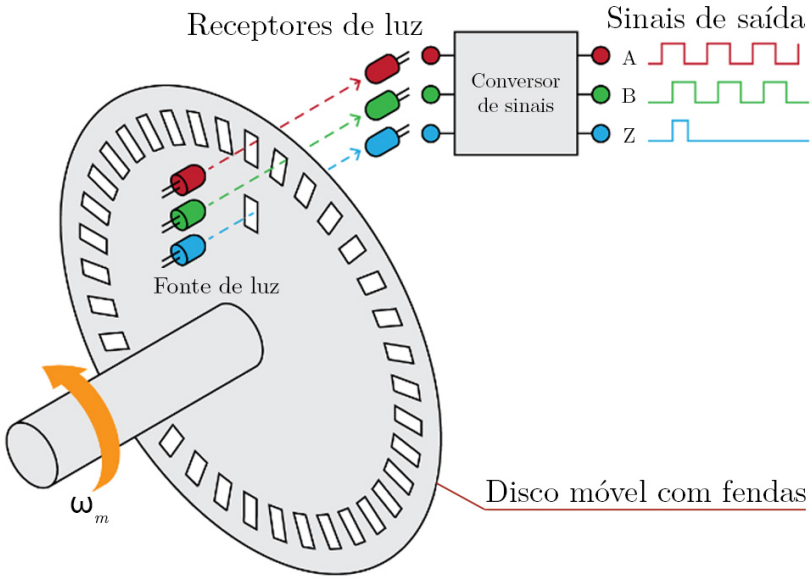

Figura 4 – Funcionamento básico do encoder. Fonte: Adaptado de <https://www.futek.com/Incremental-Encoder-Signal-Converter>.

disco e é detectado por um sensor. Quando o disco é girado, o sensor produz um pulso de saída e o número de pulsos é proporcional ao ângulo de giro do disco. A posição angular do disco e a rotação do eixo são determinadas pelo número de pulsos emitidos a partir da posição inicial. Com isso, a orientação ϕ e as coordenadas $[x, y]$ do robô móvel podem ser obtidas da seguinte forma:

$$\begin{aligned} \phi+ &= \frac{D_R - D_L}{L}, \\ x+ &= D_C \cos(\phi), \\ y+ &= D_C \sin(\phi), \end{aligned} \tag{5}$$

sendo que o deslocamento angular medido pelas rodas da direita e esquerda é representado por D_R e D_L, respectivamente, e a distância entre as rodas é representada por L. O deslocamento angular central do robô, dado por D_C, pode ser representado de acordo com a seguinte Equação:

$$D_C = \frac{D_R - D_L}{2}, \tag{6}$$

sendo que o deslocamento angular dos encoders pode ser medido por:

$$D = 2\pi R \frac{\Delta P}{N}, \tag{7}$$

em que o raio da roda é representado por R, o número de pulsos medidos durante a base de tempo do microcontrolador é representado por ΔP, e N representa o número de pulsos por revolução do encoder. O acionamento do motores e a leitura dos encoders são realizadas por uma placa de acionamento, utilizando um controlador *Proporcional-Integral* (PI) para controlar a velocidade angular de cada roda do robô. Esse controlador PI pode ser representado por:

$$u(t) = K_p + K_i \int_0^t e(\tau)d\tau, \tag{8}$$

em que u representa a lei de controle, K_p representa o ganho proporcional, K_i é o ganho integral, e é o erro, t representa o tempo e τ é o tempo de integração. Para o contexto da robótica, a interface entre a placa de acionamento dos motores e o sistema de navegação do robô pode ser realizada através do pacote *Nav2*[2], disponível para o ROS, e proposto em (MACENSKI et al., 2020). Para mais informações sobre o ROS, veja o Apêndice B. O pacote *Nav2* oferece diversas ferramentas para robôs móveis autônomos, como o planejamento de rotas, controle, localização e visualização. Ele utiliza árvores de comportamento em seu sistema para orquestrar servidores independentes de maneira modular. Esses servidores se comunicam com a árvore através de uma interface do ROS como uma ação ou um serviço. Sendo assim, um robô pode utilizar diferentes árvores para realizar diversas tarefas. Para o contexto desta pesquisa, o pacote recebe os comandos de velocidade angular e linear do robô e realiza a conversão para comandos de velocidade angulares, que são posteriormente enviadas para o acionamento dos motores. Além disso, ele realiza o envio das informações de odometria que são calculados no microcontrolador.

3.2.2 Câmeras

As câmeras com RGBD são uma opção vantajosa em relação às câmeras monoculares, por sua capacidade em perceber profundidade no ambiente, o que possibilita que ela capture objetos em diferentes planos e conte com uma melhoria significativa no reconhecimento de objetos e características visuais no ambiente, o que torna a estimativa da distância entre a câmera e os objetos no ambiente mais precisa. Em contraste, as câmeras monoculares são mais simples, por não capturarem informações de profundidade, apenas imagens bidimensionais. Entretanto, por este fator, possuem um custo menor e manuseamento simplificado.

Ao capturar imagens com profundidade, é possível aplicar algoritmos de reconstrução de nuvem de pontos para visualizar e obter características mais precisas do ambiente. Um algoritmo amplamente utilizando é o RTAB-Map[3], que é uma abordagem de SLAM baseada em otimização de grafos. Esse método utiliza a estratégia incremental de fechamento de loop com um conjunto de informações em memória para determinar qual

[2] <https://docs.nav2.org/>
[3] https://introlab.github.io/rtabmap/

é a probabilidade de que uma nova imagem apareça de um local já conhecido ou novo. Dessa forma, é possível reconstruir mapas bidimensionais ou tridimensionais a partir da imagem da câmera RGBD. A Figura 5 apresenta um exemplo de reconstrução de mapa tridimensional utilizando essa ferramenta.

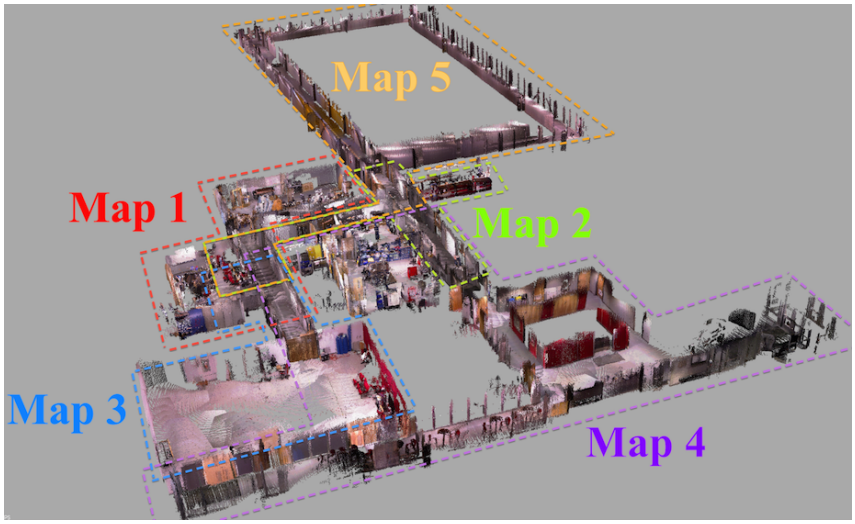

Figura 5 – Combinação da reconstrução de diversos mapas com o RTAB-Map. Fonte: (LABBÉ; MICHAUD, 2014).

3.2.3 Inertial Measurement Unit

A Unidade de Medição Inercial (IMU) é um dispositivo composto por acelerômetros, giroscópios e magnetômetros, sendo utilizado para estimar grandezas como aceleração linear, velocidade angular e orientação do robô. Uma das principais vantagens do uso da (IMU) é o fato de que essas unidades não dependem de sinais externos para fornecer estimativas do movimento, o que as torna particularmente adequadas para aplicações em ambientes internos.

Entretanto, a odometria baseada exclusivamente em sensores inerciais apresenta limitações significativas quando utilizada por períodos prolongados. As estimativas de velocidade e posição são obtidas por meio da integração sucessiva das medições de aceleração e velocidade angular, de modo que erros presentes nessas medições são progressivamente acumulados ao longo do tempo. Como consequência, a incerteza associada à estimativa de pose cresce de forma não limitada, resultando em degradação da precisão da localização, conforme discutido em (AQEL et al., 2016).

3.2.4 Light Detection and Ranging

Sensores do tipo Light Detection and Ranging (LiDAR) são amplamente utilizados para estimar a posição e a orientação de robôs móveis por meio do rastreamento de padrões

geométricos obtidos a partir da emissão de feixes laser e da análise de suas reflexões nos objetos presentes no ambiente. As medições fornecidas pelo LiDAR permitem a extração de informações precisas sobre distâncias e estrutura espacial, as quais podem ser exploradas em algoritmos de localização e mapeamento.

Em comparação com os sensores baseados em visão, o LiDAR apresenta menor sensibilidade às condições de iluminação do ambiente, o que o torna particularmente adequado para aplicações em cenários internos ou com iluminação variável. Além disso, sensores LiDAR modernos apresentam dimensões e peso compatíveis com a integração em plataformas robóticas móveis. Além disso, o LiDAR também é amplamente empregado em conjunto com algoritmos de SLAM, possibilitando a reconstrução do mapa do ambiente a partir das nuvens de pontos geradas pelo sensor, seja em duas ou três dimensões, contribuindo para estimativas de localização mais robustas e consistentes.

3.3 Localização e Mapeamento Simultâneos

A técnica de Localização e Mapeamento Simultâneos (SLAM) permite a um robô estimar simultaneamente sua própria localização e construir um mapa do ambiente no qual está inserido. Esse mapa consiste em uma representação das características perceptíveis do ambiente, tais como obstáculos, objetos e outros elementos estruturais, que podem ser obtidos por meio de sensores exteroceptivos, como câmeras e sensores LiDAR. Em ambientes complexos, essas características podem apresentar elevada similaridade visual ou geométrica, o que pode gerar ambiguidades na estimação da pose do robô e dificultar a identificação de locais previamente visitados, problema conhecido como ambiguidade perceptiva ou falha de fechamento de laço. Uma estratégia para mitigar esse problema consiste na inserção de marcadores visuais artificiais no ambiente, permitindo ao robô dispor de referências absolutas para a diferenciação de posições ao longo de sua trajetória. Nesse contexto, marcadores fiduciais do tipo ArUco são empregados neste projeto como auxílio ao processo de localização e mapeamento.

A construção de um mapa confiável do ambiente é fundamental para diversas tarefas em robótica móvel, incluindo o planejamento de trajetórias, a redução de erros na estimativa da pose do robô, a visualização do ambiente por operadores externos e a operação em cenários nos quais não há acesso a sistemas de posicionamento global, como ocorre tipicamente em ambientes internos.

Nesse sentido, segundo (THRUN; BURGARD; FOX, 2005), o problema de SLAM pode ser formulado de duas maneiras principais: o full SLAM e o SLAM online. No caso do full SLAM, busca-se estimar a distribuição de probabilidade a posteriori da trajetória completa do robô, juntamente com o mapa do ambiente, a partir de todas as observações

e entradas de controle disponíveis. Essa formulação pode ser expressa como

$$p(x_{r,k}, m \mid z_k, u_k), \tag{9}$$

em que $x_{r,k} = \{x_{r,1}, x_{r,2}, \ldots, x_{r,k}\}$ representa a sequência de estados do robô até o instante k, m denota o mapa do ambiente, z_k corresponde ao conjunto de medições realizadas e u_k às entradas de controle aplicadas ao sistema.

Em contraste, o SLAM online tem como objetivo estimar apenas o estado atual do robô e o mapa, descartando explicitamente a necessidade de manter toda a trajetória passada. Essa formulação é dada por

$$p(x_{r,k}, m \mid z_k, u_k), \tag{10}$$

sendo, portanto, mais adequada para aplicações em tempo real, nas quais as estimativas são atualizadas de forma recursiva a cada novo instante de tempo.

De maneira geral, algoritmos que abordam o full SLAM processam de forma conjunta todos os dados disponíveis, enquanto abordagens de SLAM online realizam a atualização incremental das estimativas à medida que novas medições são incorporadas. Além disso, o problema de SLAM pode ser classificado de acordo com a natureza das variáveis envolvidas, podendo apresentar componentes contínuos e discretos.

Nos problemas de estimativa contínua, o algoritmo estima variáveis como a pose do robô e a posição dos elementos do mapa em um espaço contínuo. Por outro lado, a estimativa discreta está associada ao problema de correspondência de dados, no qual cada nova observação deve ser associada a uma característica previamente mapeada ou identificada como uma nova entidade. Em muitos cenários práticos, o SLAM envolve simultaneamente variáveis contínuas e discretas, como ocorre quando a pose do robô é estimada de forma contínua, enquanto as associações entre medições e características do mapa são tratadas de maneira discreta.

Nesse contexto, o SLAM online pode ser interpretado como uma marginalização do full SLAM, obtida por meio da integração das poses passadas e da soma sobre todas as possíveis associações de dados, conforme a seguinte expressão:

$$p(x_{r,k}, m \mid z_k, u_k) = \int \cdots \int \sum_{c_1} \sum_{c_2} \cdots \sum_{c_{k-1}} p(x_{r,k}, m \mid z_k, u_k)\, dx_1\, dx_2 \cdots dx_{k-1}. \tag{11}$$

A complexidade dos problemas de SLAM é fortemente influenciada tanto pela elevada dimensionalidade do espaço contínuo de estados quanto pelo número de variáveis discretas associadas à correspondência de dados. Em ambientes extensos e com grande quantidade de características perceptíveis, esse espaço cresce rapidamente com o tempo, tornando a solução exata computacionalmente inviável. Dessa forma, algoritmos de SLAM aplicados em contextos reais geralmente recorrem a aproximações e simplificações, buscando um compromisso entre precisão e custo computacional.

3.4 Plataformas de Simulação

Nessa Seção são apresentados os softwares de simulação que foram utilizados para testar e validar o *framework* proposto neste trabalho.

3.4.1 Matlab

O MATLAB[4] é uma plataforma interativa de alto desempenho voltada à computação técnica, desenvolvida pela MathWorks. Uma de suas principais características é o fato de que o elemento fundamental de manipulação de dados é o vetor, o qual não requer dimensionamento prévio, o que facilita significativamente a operação com matrizes e estruturas de dados. Além disso, o MATLAB dispõe de uma linguagem de programação multifuncional voltada à computação numérica, sendo amplamente utilizado nas áreas de ciências exatas, tanto em pesquisa acadêmica quanto em aplicações industriais.

A plataforma oferece um conjunto abrangente de ferramentas, comandos e funções matemáticas que auxiliam na realização de cálculos numéricos e na execução de simulações baseadas em modelos matemáticos. Esses modelos podem ser expressos na forma de diagramas de blocos, representações em espaço de estados, esquemas funcionais ou código-fonte, permitindo a análise do comportamento dinâmico dos sistemas ao longo do tempo e a visualização dos resultados por meio de gráficos ou escopos de variáveis.

No contexto desta pesquisa, o MATLAB foi utilizado para a execução de simulações e testes para atingir os resultados esperados. Desta maneira, essa ferramenta viabiliza a execução de simulações numéricas das trajetórias percorridas pelo robô, tanto a partir de dados simulados, quanto dados experimentais. Essas simulações facilitaram a avaliação do comportamento das variáveis do sistema.

3.4.2 Rviz

O RVIZ é uma ferramenta que permite visualizar mensagens do ROS, informações sobre sensores, robôs e algoritmos, em um espaço tridimensional, em tempo real. Trata-se de uma ferramenta de extrema importância para depuração, visualização, e realização de testes que permitem melhor compreensão do funcionamento do sistema sendo implementado. Esta ferramenta é crucial para a validação e visualização das relações entre os sistemas de coordenadas e informações sobre o sistema que são publicadas no ROS2. Na Figura 6, é possível visualizar o mapa do ambiente utilizado para testes reconstruído em um plano bidimensional, e na Figura 7 o mapa tridimensional reconstruído.

[4] <https://www.mathworks.com/products/matlab.html>

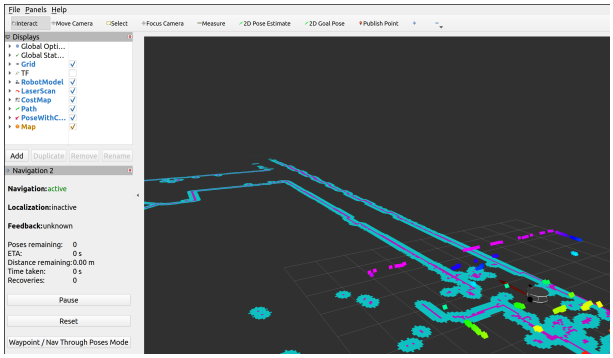

Figura 6 – Visualização do mapa bidimensional no RVIZ. Fonte: elaborado pela autora.

Figura 7 – Visualização do mapa tridimensional no RVIZ. Fonte: elaborado pela autora

3.4.3 Gazebo

O Gazebo (KOENIG; HOWARD, 2004) é um simulador tridimensional de robótica, de código aberto, amplamente utilizado para a criação de ambientes virtuais, a reprodução de cenários reais e a avaliação de sistemas robóticos em condições controladas. A ferramenta permite inserir modelos de robôs, configurar ambientes de simulação, representar propriedades físicas dos objetos, simular sensores e atuadores, além de testar algoritmos de navegação, controle e percepção de forma prática e segura.

Um dos principais objetivos do Gazebo é possibilitar a reprodução fiel de ambientes dinâmicos semelhantes aos encontrados por robôs em aplicações reais, como é apresentado na Figura 9, que mostra um cenário externo e interno na interface do Gazebo. Para isso, os objetos presentes na simulação podem possuir propriedades físicas, como massa, velocidade, geometria, atrito e colisão, permitindo que sejam empurrados, carregados, manipulados ou utilizados como obstáculos no ambiente. Dessa forma, o simulador oferece uma plataforma robusta para avaliar o comportamento de robôs diante de diferentes condições operacionais, sem a necessidade de expor diretamente o robô físico a riscos ou falhas durante as etapas iniciais de desenvolvimento.

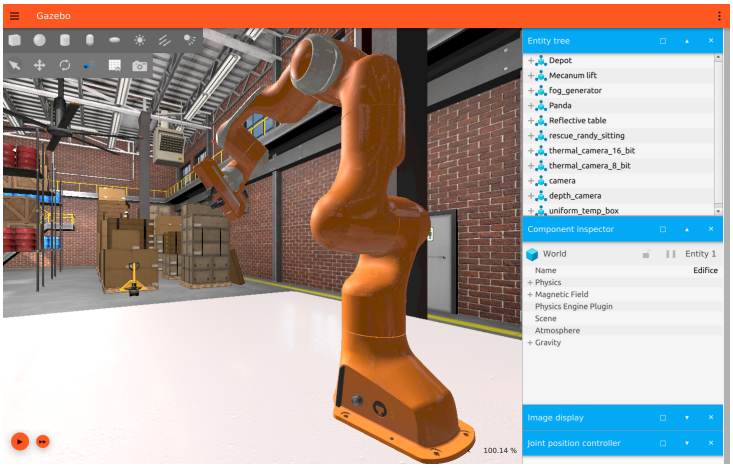

Figura 8 – Braço robótico em um cenário simulado no Gazebo. Fonte: retirado de <https://gazebosim.org/showcase>

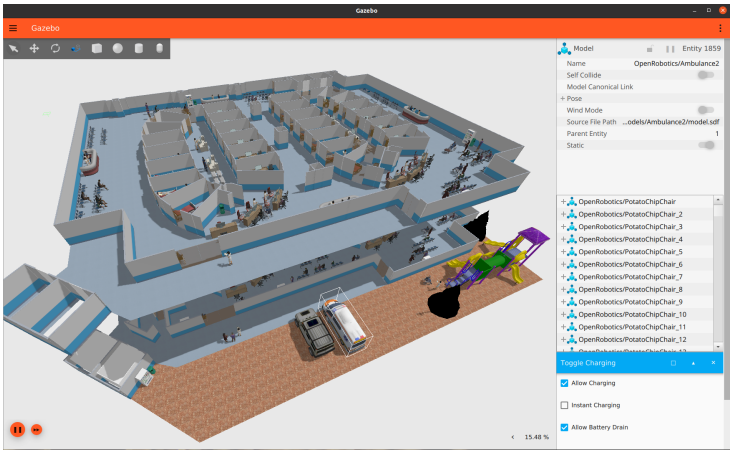

Figura 9 – Prédio em cenário simulado no Gazebo. Fonte: retirado de <https://gazebosim.org/showcase>

Por ser uma ferramenta *open-source*, o Gazebo conta com uma comunidade ativa de desenvolvedores e pesquisadores, que contribuem continuamente com melhorias, novos recursos, modelos e correções. Essa característica torna o simulador uma alternativa escalável e acessível para a realização de experimentos, validação de algoritmos e desenvolvimento de sistemas robóticos.

Na robótica móvel, o Gazebo permite simular robôs com rodas, esteiras ou pernas em ambientes virtuais compostos por pisos, paredes, obstáculos, rampas, iluminação, colisões, ruídos e diferentes tipos de sensores, como mostra a interface do Gazebo na Figura 8, que apresenta um braço robótico em ambiente fechado com paredes, caixas e estrutura semelhante a um depósito. A ferramenta oferece suporte à simulação de sensores como LiDAR, câmeras 2D e 3D, IMU e GPS, além de possibilitar a personalização do comportamento dos robôs e dos cenários. Com isso, é possível testar a interação entre o robô, os sensores utilizados, e o ambiente antes da implementação em hardware real.

Entre as principais aplicações do Gazebo na robótica móvel, destaca-se o teste de algoritmos de navegação autônoma. O simulador permite avaliar tarefas como seguimento de trajetórias, desvio de obstáculos, planejamento de rotas, navegação e execução de missões autônomas. Esse processo reduz custos e riscos, pois possibilita identificar falhas de software e ajustar parâmetros de controle antes da realização de experimentos com o robô físico.

O Gazebo também é amplamente utilizado em conjunto com o ROS e o ROS 2, compondo um ecossistema de desenvolvimento bastante consolidado na robótica. Nessa integração, comandos de velocidade, dados de sensores, informações de odometria e estados do robô podem ser trocados entre os nós do ROS e o ambiente de simulação. Essa integração permite que o mesmo software desenvolvido para a simulação seja posteriormente adaptado ou utilizado em plataformas reais.

Dessa forma, o Gazebo se consolida como uma ferramenta essencial para a realização dos testes e validações em ambiente simulado no contexto deste trabalho.

Capítulo 4

REKF-SLAM Framework

Neste trabalho, foi desenvolvido um *framework* robusto baseado no EKF-SLAM para melhorar a localização de robôs móveis de acionamento diferencial que operam em ambientes internos, sem dados de GPS. O *framework* estima conjuntamente a pose do robô e as poses dos marcadores ArUco observados no ambiente em que o robô percorre.

Diferentemente das formulações convencionais de EKF-SLAM, que usualmente representam erros de modelagem apenas por meio de ruído gaussiano aditivo (GUSRIAL et al., 2025), o *framework* desenvolvido considera explicitamente incertezas paramétricas estruturadas nos modelos de predição e correção localmente linearizados. Essas incertezas podem surgir do deslizamento das rodas, erros de calibração da odometria, medições atrasadas, imprecisões na calibração da câmera, inconsistências de sincronização e efeitos de linearização (INOUE; TERRA; JR., 2008; QI; FAN; LI, 2022). Ao incorporar tais estruturas de incerteza na recursão de filtragem por meio do algoritmo RKF desenvolvido em (ROCHA; TERRA, 2021), que se fundamenta na formulação do filtro BDU de (SAYED, 2001), o método proposto busca reduzir o excesso de confiança do estimador e melhorar a consistência da covariância, preservando a implementação recursiva em tempo de execução.

A arquitetura geral do *framework* é ilustrada na Figura 11. Conforme mostrado no diagrama, o sistema é organizado em duas etapas principais: predição e atualização. A etapa de predição utiliza informações de odometria das rodas para propagar o estado do robô ao longo do tempo. Seja k o instante de amostragem atual e $k+1$ o instante subsequente. A pose do robô é representada por

$$\mathbf{x}_{r,k} = \begin{bmatrix} x_{r,k} & y_{r,k} & \psi_{r,k} \end{bmatrix}^T, \tag{12}$$

em que $x_{r,k}$ e $y_{r,k}$ denotam as coordenadas cartesianas do robô no referencial global, e $\psi_{r,k}$

representa o seu ângulo de guinada.

A etapa de atualização incorpora informações externas de pose fornecidas pelo algoritmo AMCL, que estima a pose do robô a partir de medições obtidas por um sensor LiDAR bidimensional. Essa estimativa auxiliar de pose pode ser expressa como

$$\mathbf{x}_{r,k}^{\text{amcl}} = \begin{bmatrix} x_{r,k}^{\text{amcl}} & y_{r,k}^{\text{amcl}} & \psi_{r,k}^{\text{amcl}} \end{bmatrix}^T. \tag{13}$$

Essa medição é incorporada ao processo de filtragem como uma correção externa, contribuindo para a redução da deriva acumulada da odometria e para a melhoria da consistência global da trajetória estimada.

Além disso, a câmera detecta marcadores fiduciais ArUco distribuídos no ambiente. Cada marcador m_i, identificado por seu identificador único i, é assumido como tendo uma pose fixa em relação ao referencial do mapa. Essa pose é definida por sua posição cartesiana, dada por x_{m_i}, y_{m_i} e z_{m_i}, e por sua orientação em termos dos ângulos de rolagem, arfagem e guinada, denotados por ϕ_{m_i}, θ_{m_i} e ψ_{m_i}, respectivamente. Portanto, a pose do i-ésimo marcador é representada como

$$\mathbf{p}_{m_i} = \begin{bmatrix} x_{m_i} & y_{m_i} & z_{m_i} & \phi_{m_i} & \theta_{m_i} & \psi_{m_i} \end{bmatrix}^T. \tag{14}$$

Como resultado, o *framework* fornece a pose estimada do robô, denotada por $\widehat{\mathbf{x}}_{r,k}$, bem como as posições estimadas dos marcadores fiduciais observados ao longo do tempo, denotadas por $\widehat{\mathbf{p}}_{m_i,k}$.

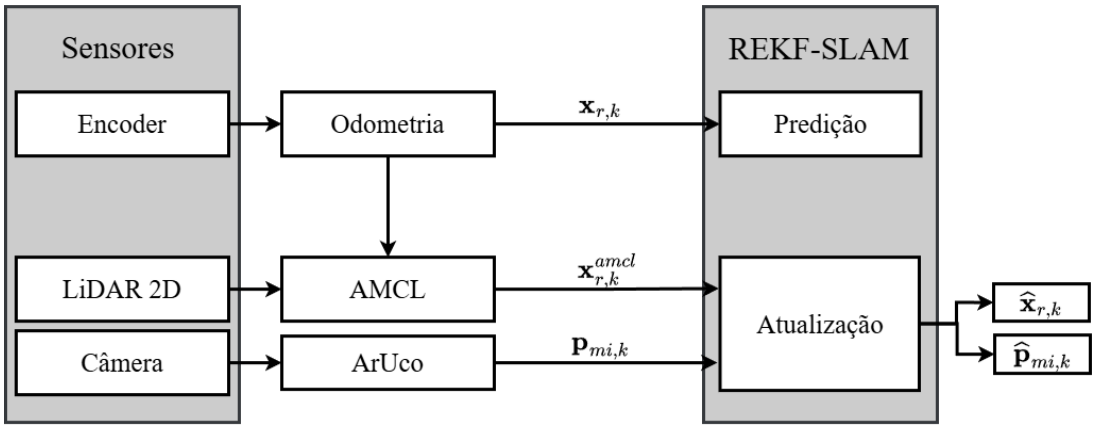

Figura 10 – Diagrama em blocos da arquitetura do sistema. Fonte: elaborado pela autora.

4.0.1 Robust Extended Kalman Filter

Nesta Subseção é apresentada a derivação do *Robust Extended Kalman Filter* (REKF) desenvolvido neste trabalho. A partir da formulação do RKF para sistemas lineares discretos no tempo sujeitos a incertezas, proposta em (ROCHA; TERRA, 2021), o estimador é estendido para sistemas não lineares por meio da linearização local dos modelos de

processo e de medição, seguindo a metodologia padrão do EKF descrita em (THRUN; BURGARD; FOX, 2005; CORKE, 2017). O tratamento robusto das incertezas paramétricas limitadas em norma é preservado, enquanto as matrizes do sistema linear são substituídas por suas correspondentes linearizadas localmente. O REKF resultante fornece uma estrutura geral de estimação robusta de estados não lineares, que serve como base para a formulação REKF-SLAM apresentada na Subseção 4.0.4, na qual o filtro é aplicado para o problema de SLAM combinado com marcadores fiduciais ArUco.

Considere o sistema não linear em tempo discreto

$$\mathbf{x}_k = \mathbf{f}\left(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, \mathbf{w}_{k-1}\right), \tag{15}$$

$$\mathbf{z}_k = \mathbf{h}(\mathbf{x}_k) + \mathbf{v}_k, \tag{16}$$

em que $\mathbf{x}_k \in \mathbb{R}^n$ é o vetor de estados, $\mathbf{u}_{k-1} \in \mathbb{R}^m$ é a entrada de controle, $\mathbf{z}_k \in \mathbb{R}^r$ é o vetor de medições, e $\mathbf{w}_{k-1}$ e $\mathbf{v}_k$ denotam, respectivamente, ruídos gaussianos de processo e de medição com média nula e matrizes de covariância dadas por

$$\mathbf{w}_{k-1} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_{k-1}), \qquad \mathbf{v}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_k).$$

Dada a estimativa filtrada $\widehat{\mathbf{x}}_{k-1|k-1}$, a etapa de predição nominal é obtida avaliando-se o modelo de movimento não linear no valor nominal do ruído de processo:

$$\widehat{\mathbf{x}}_{k|k-1} = \mathbf{f}\left(\widehat{\mathbf{x}}_{k-1|k-1}, \mathbf{u}_{k-1}, \mathbf{0}\right). \tag{17}$$

Uma expansão de Taylor de primeira ordem em torno de $\widehat{\mathbf{x}}_{k-1|k-1}$ resulta no modelo de predição localmente linearizado

$$\mathbf{x}_k \approx \widehat{\mathbf{x}}_{k|k-1} + \mathbf{F}_{k-1}\left(\mathbf{x}_{k-1} - \widehat{\mathbf{x}}_{k-1|k-1}\right) + \mathbf{G}_{k-1}\mathbf{w}_{k-1}, \tag{18}$$

em que $\mathbf{F}_{k-1}$ e $\mathbf{G}_{k-1}$ são definidas como:

$$\left(\mathbf{F}_{k-1} = \frac{\partial \mathbf{f}}{\partial \mathbf{x}}, \quad \mathbf{G}_{k-1} = \frac{\partial \mathbf{f}}{\partial \mathbf{w}}\right)\Bigg|_{\widehat{\mathbf{x}}_{k-1|k-1}, \mathbf{u}_{k-1}, \mathbf{0}}. \tag{19}$$

De forma semelhante, o modelo de observação não linear é linearizado em torno da estimativa predita do estado $\widehat{\mathbf{x}}_{k|k-1}$ por meio de uma aproximação de Taylor de primeira ordem

$$\mathbf{h}(\mathbf{x}_k) \approx \mathbf{h}(\widehat{\mathbf{x}}_{k|k-1}) + \mathbf{H}_k\left(\mathbf{x}_k - \widehat{\mathbf{x}}_{k|k-1}\right), \tag{20}$$

em que $\mathbf{H}_k$ é a Jacobiana de observação avaliada no estado predito:

$$\mathbf{H}_k = \left.\frac{\partial \mathbf{h}}{\partial \mathbf{x}}\right|_{\widehat{\mathbf{x}}_{k|k-1}}. \tag{21}$$

Expandindo a expressão linearizada, obtém-se

$$\mathbf{h}(\mathbf{x}_k) \approx \mathbf{h}(\widehat{\mathbf{x}}_{k|k-1}) + \mathbf{H}_k\mathbf{x}_k - \mathbf{H}_k\widehat{\mathbf{x}}_{k|k-1}.$$

Substituindo essa aproximação na equação de observação, $\mathbf{z}_k = \mathbf{h}(\mathbf{x}_k) + \mathbf{v}_k$, obtém-se a pseudomedição

$$\mathbf{y}_k = \mathbf{z}_k - \mathbf{h}(\widehat{\mathbf{x}}_{k|k-1}) + \mathbf{H}_k\widehat{\mathbf{x}}_{k|k-1} = \mathbf{H}_k\mathbf{x}_k + \mathbf{v}_k, \tag{22}$$

que converte o modelo de observação não linear em uma equação linear de pseudomedição adequada ao REKF.

Utilizando a pseudomedição definida na Equação (22), o REKF opera sobre o seguinte sistema linearizado incerto:

$$\mathbf{x}_k = (\mathbf{F}_{k-1}+\delta\mathbf{F}_{k-1})\mathbf{x}_{k-1} + (\mathbf{G}_{k-1}+\delta\mathbf{G}_{k-1})\mathbf{w}_{k-1}, \tag{23}$$

$$\mathbf{y}_k = (\mathbf{H}_k+\delta\mathbf{H}_k)\mathbf{x}_k + \mathbf{v}_k, \tag{24}$$

em que $\delta\mathbf{F}_{k-1}$, $\delta\mathbf{G}_{k-1}$ e $\delta\mathbf{H}_k$ denotam incertezas paramétricas estruturadas que afetam as matrizes localmente linearizadas. Assume-se que essas incertezas sejam limitadas em norma e modeladas como

$$\begin{bmatrix}\delta\mathbf{F}_{k-1} & \delta\mathbf{G}_{k-1}\end{bmatrix} = \mathbf{M}_{1,k}\boldsymbol{\Delta}_{1,k}\begin{bmatrix}\mathbf{E}_{F,k} & \mathbf{E}_{G,k}\end{bmatrix}, \tag{25}$$

$$\delta\mathbf{H}_k = \mathbf{M}_{2,k}\boldsymbol{\Delta}_{2,k}\begin{bmatrix}\mathbf{E}_{H,k}\end{bmatrix}, \tag{26}$$

sujeitas a

$$\|\boldsymbol{\Delta}_{1,k}\| \leq 1, \qquad \|\boldsymbol{\Delta}_{2,k}\| \leq 1. \tag{27}$$

Nessa formulação, p e q representam, respectivamente, as dimensões dos vetores de ruído de processo e de ruído de medição. Os inteiros s_1 e t_1 são dimensões auxiliares introduzidas pela fatoração da incerteza limitada em norma associada à equação de estado. De forma análoga, s_2 e t_2 definem as dimensões correspondentes para a fatoração da incerteza na equação de medição, associada a $\mathbf{M}_{2,k}$, $\boldsymbol{\Delta}_{2,k}$ e $\mathbf{E}_{H,k}$. Portanto, $\mathbf{M}_{1,k} \in \mathbb{R}^{n\times s_1}$, $\mathbf{M}_{2,k} \in \mathbb{R}^{r\times s_2}$, $\mathbf{E}_{F_k} \in \mathbb{R}^{t_1\times n}$, $\mathbf{E}_{G_k} \in \mathbb{R}^{t_1\times p}$, $\mathbf{E}_{H_k} \in \mathbb{R}^{t_2\times p}$ são matrizes conhecidas, enquanto $\boldsymbol{\Delta}_{1,k} \in \mathbb{R}^{s_1\times t_1}$ e $\boldsymbol{\Delta}_{2,k} \in \mathbb{R}^{s_2\times t_2}$ são matrizes de contração arbitrárias.

Seguindo a formulação robusta de mínimos quadrados regularizados, a correção do REKF é obtida a partir do seguinte problema de otimização min-max

$$\min_{\mathbf{x}_k,\mathbf{w}_{k-1},\mathbf{v}_k}\ \max_{\delta\mathbf{F}_{k-1},\delta\mathbf{G}_{k-1},\delta\mathbf{H}_k}\ \left\|\mathbf{x}_k - \widehat{\mathbf{x}}_{k|k-1}\right\|^2_{\mathbf{P}^{-1}_{k|k-1}} + \|\mathbf{w}_{k-1}\|^2_{\mathbf{Q}^{-1}_{k-1}} + \|\mathbf{v}_k\|^2_{\mathbf{R}^{-1}_k}, \tag{28}$$

sujeito aos modelos incertos em (23)–(27).

A recursão robusta introduz termos de ponderação dependentes da incerteza, controlados pelos parâmetros de ajuste $\mu > 0$ e $\xi > 0$. A implementação recursiva completa utilizada neste trabalho é resumida no Algoritmo 1.

4.0.2 Modelo de Propagação

A evolução do estado do robô ao longo do tempo pode ser descrita por meio de um modelo matemático que relaciona o estado atual, as entradas de controle aplicadas e as

Algoritmo 1 Algoritmo do REKF

Modelo: Considere o modelo linearizado incerto do sistema (23)–(24).

Entrada: Estimativa inicial $\hat{\mathbf{x}}_{0|0}$, covariância inicial $\mathbf{P}_{0|0} \succ 0$, covariância de processo $\mathbf{Q}_k \succ 0$, covariância de medição $\mathbf{R}_k \succ 0$, matrizes de incerteza $\mathbf{M}_{1,k}$, $\mathbf{M}_{2,k}$, $\mathbf{E}_{F_k}$, $\mathbf{E}_{G_k}$, $\mathbf{E}_{H_k}$, e parâmetros de ajuste $\mu > 0$, $\xi > 0$.

Para $k = 0, 1, \ldots, N-1$ **faça**

  Linearize os modelos não lineares de processo e de observação

  Calcule as Jacobianas $\mathbf{F}_k$, $\mathbf{G}_k$, e $\mathbf{H}_k$ utilizando (19) e (21)

  Calcule a pseudomedição $\mathbf{y}_k$ utilizando (22)

  Calcule as matrizes modificadas do sistema

  $\widehat{\lambda}_k = (1+\xi)\mu \left\| \text{diag}\left(\mathbf{M}_{1,k}^T\mathbf{M}_{1,k}, \mathbf{M}_{2,k}^T\mathbf{M}_{2,k}\right)\right\|$

  $\mathbf{\Phi}_{1,k} = \mu^{-1}\mathbf{I} - \widehat{\lambda}_k^{-1}\mathbf{M}_{1,k}\mathbf{M}_{1,k}^T$

  $\mathbf{\Phi}_{2,k} = \mu^{-1}\mathbf{I} - \widehat{\lambda}_k^{-1}\mathbf{M}_{2,k}\mathbf{M}_{2,k}^T$

  $\bar{\mathbf{Q}}_k = \widehat{\lambda}_k^{-1}\mathbf{I} + \mathbf{E}_{G_k}\mathbf{Q}_k\mathbf{E}_{G_k}^T$ $\bar{\mathbf{R}}_k = \widehat{\lambda}_k^{-1}\mathbf{I} + \mathbf{R}_k$

  $\widehat{\mathbf{Q}}_k = \mathbf{\Phi}_{1,k} + \mathbf{G}_k\left(\mathbf{Q}_k^{-1} + \widehat{\lambda}_k\mathbf{E}_{G_k}^T\mathbf{E}_{G_k}\right)^{-1}\mathbf{G}_k^T$

  $\widehat{\mathbf{R}}_k = \mathbf{\Phi}_{2,k} + \mathbf{R}_k$

  $\widehat{\mathbf{F}}_k = \mathbf{F}_k - \mathbf{G}_k\mathbf{Q}_k\mathbf{E}_{G_k}^T\bar{\mathbf{Q}}_k^{-1}\mathbf{E}_{F_k}$

  $\widehat{\mathbf{H}}_k = \mathbf{H}_k - \mathbf{R}_k\bar{\mathbf{R}}_k^{-1}\mathbf{E}_{H_k}$

  Atualize a estimativa robusta filtrada (correção)

  $\mathbf{P}_{k|k} = \left(\mathbf{P}_{k|k-1}^{-1} + \widehat{\mathbf{H}}_k^T\widehat{\mathbf{R}}_k^{-1}\widehat{\mathbf{H}}_k + \mathbf{E}_{H_k}^T\bar{\mathbf{R}}_k^{-1}\mathbf{E}_{H_k} + \mathbf{E}_{F_k}^T\bar{\mathbf{Q}}_k^{-1}\mathbf{E}_{F_k}\right)^{-1}$

  $\hat{\mathbf{x}}_{k|k} = \mathbf{P}_{k|k}\left(\mathbf{P}_{k|k-1}^{-1}\hat{\mathbf{x}}_{k|k-1} + \widehat{\mathbf{H}}_k^T\widehat{\mathbf{R}}_k^{-1}\mathbf{y}_k\right)$

  Atualize a estimativa robusta predita (predição)

  Calcule o estado predito $\hat{\mathbf{x}}_{k+1|k}$ utilizando (17)

  Calcule a covariância predita $\mathbf{P}_{k+1|k} = \widehat{\mathbf{F}}_k\mathbf{P}_{k|k}\widehat{\mathbf{F}}_k^T + \widehat{\mathbf{Q}}_k$

  Retorne $\hat{\mathbf{x}}_{k+1|k}, \mathbf{P}_{k+1|k}$

end Para

incertezas associadas ao processo. Esse tipo de modelagem permite prever a evolução do sistema com base em sua dinâmica e nas entradas conhecidas, sendo amplamente utilizado em problemas de estimação de estado e localização.

Segundo (THRUN; BURGARD; FOX, 2005), o modelo de propagação do estado pode ser formulado com base em um modelo de velocidade ou em um modelo de odometria. No caso do modelo de velocidade, assume-se que o robô é controlado diretamente por meio de uma velocidade translacional v_k e de uma velocidade rotacional ω_k. Na prática, entretanto, os modelos baseados em odometria tendem a apresentar maior precisão do que os modelos puramente baseados em comandos de velocidade, uma vez que a maioria dos robôs móveis não executa esses comandos com exatidão. A utilização de medições diretas, como a contagem das revoluções das rodas por meio de encoders, permite obter estimativas mais confiáveis do deslocamento do robô e reduzir o erro acumulado na propagação do estado.

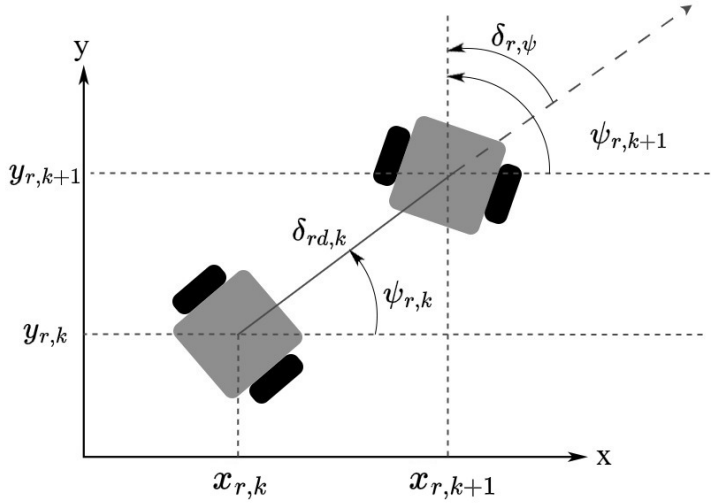

Figura 11 – Modelo planar de odometria do robô móvel diferencial.

Dessa forma, neste trabalho, a etapa de propagação utiliza incrementos de odometria das rodas de um robô móvel diferencial planar, conforme ilustrado na Figura 11.

O vetor de entrada de odometria é definido como

$$\mathbf{u}_{r,k-1} = \begin{bmatrix} \delta_{rd,k-1} & \delta_{r\psi,k-1} \end{bmatrix}^T, \tag{29}$$

em que $\delta_{rd,k}$ representa a distância incremental percorrida e $\delta_{r\psi,k}$ representa a variação incremental de orientação.

A incerteza da odometria é modelada como um ruído de processo gaussiano aditivo,

$$\mathbf{w}_{r,k-1} = \begin{bmatrix} w_{rd,k-1} & w_{r\psi,k-1} \end{bmatrix}^T \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_r), \tag{30}$$

com covariância

$$\mathbf{Q}_{r,k-1} = \operatorname{diag}\left(\sigma_{rd}^2, \sigma_{r\psi}^2\right). \tag{31}$$

Sejam c_{k-1} e s_{k-1} definidos como

$$c_{k-1} = \cos(\psi_{r,k-1}), \qquad s_{k-1} = \sin(\psi_{r,k-1}).$$

O modelo não linear de movimento do robô é escrito como

$$\begin{aligned} \mathbf{x}_{r,k} &= \mathbf{f}_r\left(\mathbf{x}_{r,k-1}, \mathbf{u}_{r,k-1}, \mathbf{w}_{r,k-1}\right) \\ &= \begin{bmatrix} x_{r,k-1} + (\delta_{rd,k-1} + w_{rd,k-1})c_{k-1} \\ y_{r,k-1} + (\delta_{rd,k-1} + w_{rd,k-1})s_{k-1} \\ \psi_{r,k-1} + \delta_{r\psi,k-1} + w_{r\psi,k-1} \end{bmatrix}. \end{aligned} \tag{32}$$

As Jacobianas do modelo de movimento são dadas pelas seguintes equações:

$$\mathbf{F}_{r,k-1} = \left.\frac{\partial \mathbf{f}_r}{\partial \mathbf{x}_r}\right|_{\widehat{\mathbf{x}}_{r,k-1|k-1}, \mathbf{u}_{r,k-1}, \mathbf{0}} = \begin{bmatrix} 1 & 0 & -\delta_{rd,k-1}s_{k-1} \\ 0 & 1 & \delta_{rd,k-1}c_{k-1} \\ 0 & 0 & 1 \end{bmatrix},$$

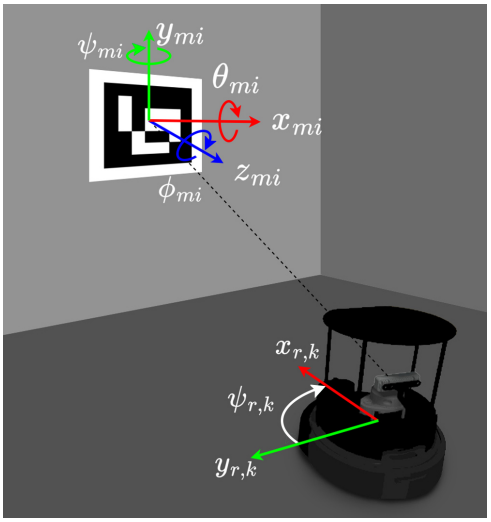

Figura 12 – Ângulos e pose do marcador em relação ao robô móvel.

$$\mathbf{G}_{r,k-1} = \left.\frac{\partial \mathbf{f}_r}{\partial \mathbf{w}_r}\right|_{\widehat{\mathbf{x}}_{r,k-1|k-1},\mathbf{u}_{r,k-1},\mathbf{0}} = \begin{bmatrix} c_{k-1} & 0 \\ s_{k-1} & 0 \\ 0 & 1 \end{bmatrix}.$$

4.0.3 Modelo de Medição

A Figura 12 ilustra a relação de coordenadas entre o robô TurtleBot 4 e um marcador fiducial ArUco. As variáveis $x_{r,k-1}$ e $y_{r,k-1}$ representam as coordenadas cartesianas do robô no referencial global, enquanto $\psi_{r,k-1}$ denota seu ângulo de guinada. Além disso, x_{m_i}, y_{m_i} e z_{m_i} definem a posição cartesiana do i-ésimo marcador, ao passo que ϕ_{m_i}, θ_{m_i} e ψ_{m_i} representam sua orientação em termos dos ângulos de rolagem, arfagem e guinada, respectivamente.

Seja $\mathcal{M}$ o conjunto de todos os marcadores fiduciais presentes no ambiente. No instante k, o robô observa um subconjunto $\mathcal{M}_k \subseteq \mathcal{M}$ contendo N_k marcadores visíveis. Para cada marcador observado, o pipeline de visão estima inicialmente a pose do marcador em relação ao referencial da câmera e, em seguida, transforma-a para o referencial da base do robô utilizando a transformação extrínseca calibrada entre a câmera e o robô. Assim, a medição associada ao marcador i é representada como

$$\mathbf{z}_{i,k} = \begin{bmatrix} x_{i,k}^{\text{rel}} & y_{i,k}^{\text{rel}} & z_{i,k}^{\text{rel}} & \phi_{i,k}^{\text{rel}} & \theta_{i,k}^{\text{rel}} & \psi_{i,k}^{\text{rel}} \end{bmatrix}^T, \tag{33}$$

em que todos os componentes são expressos em relação ao referencial da base do robô. Sob a hipótese de movimento planar, a observação predita do marcador é modelada como

$$\mathbf{h}_{i,k}(\mathbf{x}_{r,k}, \mathbf{p}_{m_i,k}) = \begin{bmatrix} \mathbf{R}_z^T(\psi_{r,k}) \left(\mathbf{p}_{m_i,k}^{xy} - \mathbf{p}_{r,k}^{xy} \right) \\ z_{m_i,k} \\ \phi_{m_i,k} \\ \theta_{m_i,k} \\ \psi_{m_i,k} - \psi_{r,k} \end{bmatrix}, \tag{34}$$

em que $\mathbf{p}_{r,k}^{xy} = [x_{r,k},\, y_{r,k}]^T$ representa a posição do robô no referencial do mapa, enquanto $\mathbf{p}_{m_i,k}^{xy} = [x_{m_i,k},\, y_{m_i,k}]^T$ representa a posição horizontal do marcador i no mesmo referencial. A matriz $\mathbf{R}_z(\psi_{r,k})$ é a matriz de rotação planar associada ao ângulo de guinada do robô e, portanto, $\mathbf{R}_z^T(\psi_{r,k})$ transforma vetores de deslocamento horizontal do referencial do mapa para o referencial da base do robô e pode ser definida como:

$$\mathbf{R}_z^T(\psi_{r,k}) = \begin{bmatrix} c_k & s_k \\ -s_k & c_k \end{bmatrix}.$$

O ruído de medição da câmera é modelado como

$$\mathbf{v}_{i,k} \sim \mathcal{N}\left(\mathbf{0}, \mathbf{R}_{z,i}^{\text{cam}}\right), \tag{35}$$

com covariância

$$\mathbf{R}_{z,i}^{\text{cam}} = \text{diag}\left(\sigma_{c,x}^2, \sigma_{c,y}^2, \sigma_{c,z}^2, \sigma_{c,\phi}^2, \sigma_{c,\theta}^2, \sigma_{c,\psi}^2\right). \tag{36}$$

A estrutura de navegação do ROS2 (MACENSKI et al., 2022), conhecida como pilha *Navigation 2* (Nav2) (MACENSKI et al., 2020), também pode fornecer uma estimativa global de pose por meio do AMCL. O AMCL localiza o robô em um mapa de ocupação 2D previamente construído, comparando medições em tempo real de um sensor LiDAR 2D com o mapa conhecido e estimando a posição e a orientação mais prováveis do robô por meio de uma abordagem baseada em filtro de partículas. Essa estimativa é incluída como observação auxiliar sempre que estiver disponível:

$$\mathbf{z}_k^{\text{amcl}} = \mathbf{h}^{\text{amcl}}(\mathbf{x}_{r,k}) + \mathbf{v}_k^{\text{amcl}}, \tag{37}$$

em que

$$\mathbf{h}^{\text{amcl}}(\mathbf{x}_{r,k}) = \begin{bmatrix} x_{r,k} & y_{r,k} & \psi_{r,k} \end{bmatrix}^T. \tag{38}$$

O ruído de medição do AMCL é modelado como

$$\mathbf{v}_k^{\text{amcl}} \sim \mathcal{N}\left(\mathbf{0}, \mathbf{R}_z^{\text{amcl}}\right),$$

com covariância

$$\mathbf{R}_z^{\text{amcl}} = \text{diag}\left(\sigma_{\text{amcl},x}^2, \sigma_{\text{amcl},y}^2, \sigma_{\text{amcl},\psi}^2\right). \tag{39}$$

Quando múltiplas observações fiduciais e a estimativa do AMCL estão simultaneamente disponíveis, elas são empilhadas no vetor unificado de observações

$$\mathbf{z}_k = \begin{bmatrix} \mathbf{z}_{1,k}^T & \cdots & \mathbf{z}_{N_k,k}^T & (\mathbf{z}_k^{\text{amcl}})^T \end{bmatrix}^T. \tag{40}$$

O modelo não linear de observação empilhado correspondente é

$$\mathbf{z}_k = \mathbf{h}_k(\mathbf{x}_k) + \mathbf{v}_k. \tag{41}$$

Após a linearização em torno de $\widehat{\mathbf{x}}_{k|k-1}$,

$$\mathbf{h}_k(\mathbf{x}_k) \approx \mathbf{h}_k(\widehat{\mathbf{x}}_{k|k-1}) + \mathbf{H}_k\left(\mathbf{x}_k - \widehat{\mathbf{x}}_{k|k-1}\right), \tag{42}$$

em que $\mathbf{H}_k$ é a Jacobiana de observação empilhada, definida conforme a Equação (21).

4.0.4 Estado Aumentado do REKF-SLAM

O vetor de estados do REKF-SLAM contém conjuntamente a pose do robô e as poses de todos os marcadores fiduciais mapeados:

$$\mathbf{x}_k = \begin{bmatrix} \mathbf{x}_{r,k}^T & \mathbf{p}_{m_1}^T & \cdots & \mathbf{p}_{m_M}^T \end{bmatrix}^T \in \mathbb{R}^{3+6M}, \tag{43}$$

em que M é o número de marcadores atualmente armazenados no mapa. A matriz de covariância associada é particionada como

$$\mathbf{P}_k = \begin{bmatrix} \mathbf{P}_{rr,k} & \mathbf{P}_{rm,k} \\ \mathbf{P}_{rm,k}^T & \mathbf{P}_{mm,k} \end{bmatrix}, \tag{44}$$

em que $\mathbf{P}_{rr,k}$ é a covariância do estado do robô, $\mathbf{P}_{mm,k}$ contém as covariâncias dos marcadores e $\mathbf{P}_{rm,k}$ armazena as correlações cruzadas entre o robô e os marcadores. Como os marcadores são considerados estáticos, somente o estado do robô evolui durante a predição. Portanto, o estado aumentado predito é

$$\widehat{\mathbf{x}}_{k+1|k} = \begin{bmatrix} \mathbf{f}_r\left(\widehat{\mathbf{x}}_{r,k|k}, \mathbf{u}_{r,k}, \mathbf{0}\right) \\ \mathbf{p}_{m_1,k} \\ \vdots \\ \mathbf{p}_{m_M,k} \end{bmatrix}. \tag{45}$$

As Jacobianas aumentadas correspondentes são

$$\mathbf{F}_k = \begin{bmatrix} \mathbf{F}_{r,k} & \mathbf{0}_{3\times 6M} \\ \mathbf{0}_{6M\times 3} & \mathbf{I}_{6M} \end{bmatrix}, \tag{46}$$

$$\mathbf{G}_k = \begin{bmatrix} \mathbf{G}_{r,k} \\ \mathbf{0}_{6M\times 2} \end{bmatrix}. \tag{47}$$

em que $\mathbf{F}_{r,k}$ e $\mathbf{G}_{r,k}$ podem ser definidas, respectivamente, como

$$\mathbf{F}_{r,k} = \begin{bmatrix} 1 & 0 & -\delta_{rd,k}\sin(\psi_{r,k}) \\ 0 & 1 & \delta_{rd,k}\cos(\psi_{r,k}) \\ 0 & 0 & 1 \end{bmatrix}, \tag{48}$$

$$\mathbf{G}_{r,k} = \begin{bmatrix} \cos(\psi_{r,k}) & 0 \\ \sin(\psi_{r,k}) & 0 \\ 0 & 1 \end{bmatrix}. \tag{49}$$

A predição da covariância do estado aumentado é dada por

$$\mathbf{P}_{k+1|k} = \mathbf{F}_k\mathbf{P}_{k|k}\mathbf{F}_k^T + \mathbf{G}_k\mathbf{Q}_{r,k}\mathbf{G}_k^T. \tag{50}$$

4.0.5 Inicialização dos Marcadores

Sempre que um marcador fiducial ainda não observado no mapa é detectado, sua pose no referencial do mapa é inicializada a partir da observação relativa do marcador expressa no referencial da base do robô e acrescentada ao vetor de estados aumentado. Seja a medição de pose relativa associada ao marcador i no instante k definida como

$$\mathbf{z}_{i,k} = \begin{bmatrix} z_{x,i,k} & z_{y,i,k} & z_{z,i,k} & z_{\phi,i,k} & z_{\theta,i,k} & z_{\psi,i,k} \end{bmatrix}^T. \tag{51}$$

Os componentes $z_{x,i,k}$, $z_{y,i,k}$ e $z_{z,i,k}$ denotam a posição medida do marcador i em relação ao referencial da base do robô, enquanto $z_{\phi,i,k}$, $z_{\theta,i,k}$ e $z_{\psi,i,k}$ denotam os ângulos relativos correspondentes de rolagem, arfagem e guinada.

A pose do marcador é inicializada por meio da função de composição não linear

$$\mathbf{p}_{m_i,k} = \mathbf{g}_m\left(\mathbf{x}_{r,k}, \mathbf{z}_{i,k}\right), \tag{52}$$

em que $\mathbf{x}_{r,k} = [x_{r,k},\, y_{r,k},\, \psi_{r,k}]^T$ denota a estimativa da pose do robô no referencial do mapa, e $\mathbf{z}_{i,k}$ é a observação relativa do marcador no referencial da base do robô.

Considerando o movimento planar do robô, a função de inicialização do marcador é definida como

$$\mathbf{g}_m(\mathbf{x}_{r,k}, \mathbf{z}_{i,k}) = \begin{bmatrix} x_{r,k} + z_{x,i,k}c_k - z_{y,i,k}s_k \\ y_{r,k} + z_{x,i,k}s_k + z_{y,i,k}c_k \\ z_{z,i,k} \\ z_{\phi,i,k} \\ z_{\theta,i,k} \\ \psi_{r,k} + z_{\psi,i,k} \end{bmatrix}. \tag{53}$$

Após a inicialização, a matriz de covariância do estado é aumentada por meio das Jacobianas de $\mathbf{g}_m$ em relação ao estado do robô e à observação relativa do marcador:

$$\mathbf{G}_{mx,k} = \frac{\partial \mathbf{g}_m}{\partial \mathbf{x}_{r,k}}, \qquad \mathbf{G}_{mz,k} = \frac{\partial \mathbf{g}_m}{\partial \mathbf{z}_{i,k}}. \tag{54}$$

A Jacobiana em relação ao estado do robô é

$$\mathbf{G}_{mx,k} = \begin{bmatrix} \mathbf{I}_2 & \mathbf{j}_{i,k} \\ \mathbf{0}_{3\times 2} & \mathbf{0}_{3\times 1} \\ \mathbf{0}_{1\times 2} & 1 \end{bmatrix}, \tag{55}$$

em que $\mathbf{j}_{i,k}$ é definido como

$$\mathbf{j}_{i,k} = \begin{bmatrix} -z_{x,i,k}s_k - z_{y,i,k}c_k \\ z_{x,i,k}c_k - z_{y,i,k}s_k \end{bmatrix}. \tag{56}$$

A Jacobiana em relação à observação relativa do marcador pode ser definida como

$$\mathbf{G}_{mz,k} = \begin{bmatrix} \mathbf{R}_{MB,k} & \mathbf{0}_{3\times 3} \\ \mathbf{0}_{3\times 3} & \mathbf{I}_3 \end{bmatrix}, \tag{57}$$

em que $\mathbf{R}_{MB,k}$ denota a matriz de rotação do referencial da base do robô para o referencial do mapa:

$$\mathbf{R}_{MB,k} = \begin{bmatrix} c_k & -s_k & 0 \\ s_k & c_k & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{58}$$

A covariância associada ao marcador inicializado é então obtida por meio da propagação de incerteza de primeira ordem:

$$\mathbf{P}_{m_i,k} = \mathbf{G}_{mx,k}\mathbf{P}_{rr,k}\mathbf{G}_{mx,k}^T + \mathbf{G}_{mz,k}\mathbf{R}_{z,i}^{\text{rel}}\mathbf{G}_{mz,k}^T, \tag{59}$$

em que $\mathbf{P}_{rr,k}$ é o bloco de covariância da pose do robô antes do aumento do estado, e $\mathbf{R}_{z,i}^{\text{rel}}$ é a matriz de covariância associada à observação relativa do marcador expressa no referencial da base do robô.

A covariância cruzada entre o marcador recém-inicializado e o estado aumentado anterior é calculada como

$$\mathbf{P}_{m_ix,k} = \mathbf{G}_{mx,k}\mathbf{P}_{rx,k}, \tag{60}$$

em que $\mathbf{P}_{rx,k}$ denota o bloco de covariância entre a pose do robô e o estado aumentado anterior. Portanto, a matriz de covariância aumentada torna-se

$$\mathbf{P}_{\text{aug},k} = \begin{bmatrix} \mathbf{P}_k & \mathbf{P}_{m_ix,k}^T \\ \mathbf{P}_{m_ix,k} & \mathbf{P}_{m_i,k} \end{bmatrix}. \tag{61}$$

Nessa expressão, $\mathbf{P}_k$ é a matriz de covariância do estado antes do aumento, $\mathbf{P}_{m_i,k}$ é o bloco de covariância do marcador recém-inicializado e $\mathbf{P}_{m_ix,k}$ é a covariância cruzada entre o novo marcador e o estado aumentado anterior.

Esse modelo de inicialização segue uma aproximação planar de EKF-SLAM. A posição horizontal do marcador é transformada do referencial da base do robô para o referencial do mapa utilizando o ângulo de guinada do robô, enquanto a coordenada vertical, a rolagem e a arfagem são incorporadas como grandezas associadas ao marcador após a transformação extrínseca entre a câmera e o robô.

4.0.6 Atualização Robusta das Medições e Implementação

Sempre que um marcador fiducial previamente mapeado é novamente observado, o filtro executa uma etapa de correção utilizando o modelo robusto pseudolinear de medição definido na Equação (22). A observação predita associada ao marcador i é calculada a partir do estado predito atual como

$$\begin{aligned}\widehat{\mathbf{z}}_{i,k} &= \mathbf{h}_{i,k}\left(\widehat{\mathbf{x}}_{r,k|k-1}, \widehat{\mathbf{p}}_{m_i,k|k-1}\right) \\ &= \begin{bmatrix} c_k(\widehat{x}_{m_i,k|k-1} - \widehat{x}_{r,k|k-1}) + s_k(\widehat{y}_{m_i,k|k-1} - \widehat{y}_{r,k|k-1}) \\ -s_k(\widehat{x}_{m_i,k|k-1} - \widehat{x}_{r,k|k-1}) + c_k(\widehat{y}_{m_i,k|k-1} - \widehat{y}_{r,k|k-1}) \\ \widehat{z}_{m_i,k|k-1} \\ \widehat{\phi}_{m_i,k|k-1} \\ \widehat{\theta}_{m_i,k|k-1} \\ \widehat{\psi}_{m_i,k|k-1} - \widehat{\psi}_{r,k|k-1} \end{bmatrix}.\end{aligned} \tag{62}$$

O vetor de inovação associado ao marcador i é definido como

$$\boldsymbol{\nu}_{i,k} = \mathbf{z}_{i,k} - \widehat{\mathbf{z}}_{i,k},$$

em que $\mathbf{z}_{i,k}$ é a observação relativa do marcador expressa no referencial da base do robô.

Para todas as observações disponíveis, o vetor de inovação empilhado é escrito como

$$\boldsymbol{\nu}_k = \mathbf{z}_k - \mathbf{h}_k\left(\widehat{\mathbf{x}}_{k|k-1}\right). \tag{63}$$

A pseudomedição empilhada é dada por

$$\mathbf{y}_k = \boldsymbol{\nu}_k + \mathbf{H}_k\widehat{\mathbf{x}}_{k|k-1}. \tag{64}$$

Essa representação reescreve o modelo de medição não linear localmente linearizado na forma exigida pela formulação de filtragem robusta, sendo equivalente à Equação (22).

A Jacobiana de medição associada ao marcador i é esparsa e contém apenas blocos não nulos correspondentes à pose do robô e ao marcador observado:

$$\begin{aligned}\mathbf{H}_{i,k} &= \left.\frac{\partial \mathbf{h}_{i,k}}{\partial \mathbf{x}_k}\right|_{\widehat{\mathbf{x}}_{k|k-1}} \\ &= \begin{bmatrix}\mathbf{H}_{r,ik} & \cdots & \mathbf{0} & \cdots & \mathbf{H}_{m,ik} & \cdots & \mathbf{0}\end{bmatrix}.\end{aligned} \tag{65}$$

Define-se os deslocamentos relativos no referencial do mapa como

$$\begin{aligned}\delta x_{i,k} &= \widehat{x}_{m_i,k|k-1} - \widehat{x}_{r,k|k-1}, \\ \delta y_{i,k} &= \widehat{y}_{m_i,k|k-1} - \widehat{y}_{r,k|k-1}.\end{aligned} \tag{66}$$

Assim, o bloco da Jacobiana em relação ao estado do robô é definida por

$$\mathbf{H}_{r,ik} = \begin{bmatrix} -c_k & -s_k & -\delta x_{i,k}s_k + \delta y_{i,k}c_k \\ s_k & -c_k & -\delta x_{i,k}c_k - \delta y_{i,k}s_k \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix}. \tag{67}$$

O bloco da Jacobiana em relação ao estado do marcador é definida como

$$\mathbf{H}_{m,ik} = \begin{bmatrix} \mathbf{R}_{BM,k} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{I}_3 \end{bmatrix}.$$

A matriz $\mathbf{R}_{BM,k}$ transforma os componentes de posição do referencial do mapa para o referencial da base do robô e é definida como

$$\mathbf{R}_{BM,k} = \begin{bmatrix} c_k & s_k & 0 \\ -s_k & c_k & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

Além disso, o ruído de medição dos marcadores fiduciais é modelado como

$$\mathbf{v}_{i,k} \sim \mathcal{N}\left(\mathbf{0}, \mathbf{R}_{z,i}^{\text{rel}}\right),$$

com covariância

$$\mathbf{R}_{z,i}^{\text{rel}} = \text{diag}\left(\sigma_{c,x}^2, \sigma_{c,y}^2, \sigma_{c,z}^2, \sigma_{c,\phi}^2, \sigma_{c,\theta}^2, \sigma_{c,\psi}^2\right).$$

Neste caso, $\mathbf{R}_{z,i}^{\text{rel}}$ representa a covariância da observação relativa do marcador fiducial expressa no referencial da base do robô, após a transformação extrínseca entre a câmera e o robô. Além das observações visuais, o sistema pode incorporar a estimativa da pose do robô fornecida pela estrutura Nav2 do ROS 2 por meio do AMCL. Essa estimativa é modelada como uma observação direta da pose planar do robô no referencial do mapa, conforme estabelecido nas Equações (37) e (38).

Se N_k marcadores e estimativas do AMCL estiverem simultaneamente disponíveis, a Jacobiana completa de observação empilhada é dada por

$$\mathbf{H}_k = \begin{bmatrix} \mathbf{H}_{1,k} \\ \vdots \\ \mathbf{H}_{N_k,k} \\ \mathbf{H}_k^{\text{amcl}} \end{bmatrix}, \quad \mathbf{H}_k^{\text{amcl}} = \begin{bmatrix} \mathbf{I}_{3\times3} & \mathbf{0}_{3\times6M} \end{bmatrix}. \tag{68}$$

A matriz de covariância empilhada correspondente é

$$\mathbf{R}_k = \text{diag}\left(\mathbf{R}_{z,1}^{\text{rel}}, \ldots, \mathbf{R}_{z,N_k}^{\text{rel}}, \mathbf{R}_z^{\text{amcl}}\right). \tag{69}$$

Se a estimativa do AMCL não estiver disponível, os blocos correspondentes a ele são removidos da formulação empilhada.

4.0.7 Implementação

Para melhorar a robustez diante de detecções visuais não confiáveis, a covariância associada às observações fiduciais é escalonada adaptativamente de acordo com o número

de marcadores visíveis e a distância entre o marcador e a câmera. A distância euclidiana entre o robô e o marcador i é calculada como

$$d_{i,k} = \left\| \begin{bmatrix} x_{i,k}^{\text{rel}} & y_{i,k}^{\text{rel}} & z_{i,k}^{\text{rel}} \end{bmatrix} \right\|, \tag{70}$$

e os fatores de escalonamento são definidos como

$$\alpha_{xy}(d_{i,k}) = 1 + \beta_{xy} d_{i,k}^2, \tag{71}$$

$$\alpha_z(d_{i,k}) = 1 + \beta_z d_{i,k}^2, \tag{72}$$

$$\alpha_\theta(d_{i,k}) = 1 + \beta_\theta d_{i,k}^2. \tag{73}$$

A covariância adaptada torna-se

$$\tilde{\mathbf{R}}_{z,i}^{\text{cam}} = \mathbf{D}_{i,k} \mathbf{R}_{z,i}^{\text{cam}} \mathbf{D}_{i,k},$$

em que $\mathbf{D}_{i,k}$ pode ser definida como

$$\mathbf{D}_{i,k} = \text{diag}\left(\alpha_{xy}, \alpha_{xy}, \alpha_z, \alpha_\theta, \alpha_\theta, \alpha_\theta\right). \tag{74}$$

Antes da etapa de correção robusta, a consistência das medições empilhadas é avaliada por meio da distância de Mahalanobis, conforme (BRAUN et al., 2022). O vetor de inovação empilhado é definido como

$$\boldsymbol{\nu}_k = \mathbf{z}_k - \mathbf{h}_k\left(\hat{\mathbf{x}}_{k|k-1}\right), \tag{75}$$

em que $\mathbf{z}_k$ denota o vetor de medições empilhado e $\mathbf{h}_k(\hat{\mathbf{x}}_{k|k-1})$ é a observação predita empilhada correspondente.

A matriz de covariância da inovação é calculada como

$$\mathbf{S}_k = \mathbf{H}_k \mathbf{P}_{k|k-1} \mathbf{H}_k^T + \mathbf{R}_k, \tag{76}$$

e o quadrado da distância de Mahalanobis é dado por

$$d_{M,k}^2 = \boldsymbol{\nu}_k^T \mathbf{S}_k^{-1} \boldsymbol{\nu}_k. \tag{77}$$

O conjunto de medições é então aceito se

$$d_{M,k}^2 \leq \gamma,$$

em que γ é um limiar de consistência selecionado de acordo com a dimensão do vetor de inovação. Se $d_{M,k}^2 > \gamma$, o conjunto de medições é rejeitado, e o filtro mantém o estado e a covariância preditos.

As leituras de odometria, as detecções de marcadores fiduciais e as estimativas de pose do AMCL são adquiridas em frequências distintas. Portanto, todas as mensagens recebidas são sincronizadas por meio de processamento baseado em *timestamps*, conforme

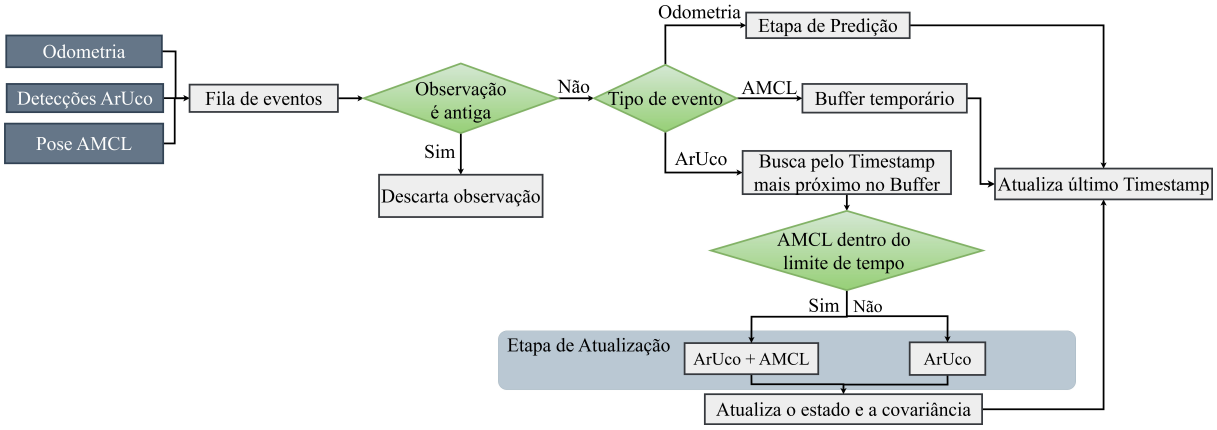

Figura 13 – Pipeline de sincronização baseado em *timestamps* e de processamento de predição-atualização para odometria, detecções de marcadores fiduciais e estimativas de pose do AMCL.

ilustrado na Figura 13. Inicialmente, as leituras de odometria, as detecções ArUco e as estimativas de pose do AMCL são recebidas pelo robô e inseridas em uma fila de eventos ordenada temporalmente. Os eventos de odometria são processados com prioridade para preservar a sequência causal de predição-atualização. Se a marca de tempo ou *timestamp* de um evento for anterior ao último instante processado, a medição correspondente é descartada. Caso contrário, as mensagens são classificadas de acordo com o tipo de evento.

Para os eventos de odometria, a etapa de predição é executada utilizando os incrementos de odometria correspondentes. As estimativas de pose do AMCL são armazenadas em um *buffer* separado. Por sua vez, as detecções ArUco são agrupadas segundo uma tolerância de conjunto predefinida, e em seguida é feita uma busca no *buffer* do AMCL para encontrar a estimativa de pose com o *timestamp* mais próximo. Para cada conjunto de marcadores fiduciais, a estimativa do AMCL mais próxima é selecionada sempre que

$$|t_k^{\text{amcl}} - t_k^{\text{batch}}| \leq \Delta t_{\max}, \tag{78}$$

em que t_k^{batch} denota o *timestamp* representativo do conjunto de marcadores fiduciais, t_k^{amcl} é o *timestamp* da estimativa do AMCL, e $\Delta t_{\max}$ é a tolerância de sincronização. A etapa de correção é então realizada utilizando tanto o conjunto de marcadores fiduciais quanto a estimativa selecionada do AMCL. Caso nenhuma estimativa do AMCL satisfaça essa condição, a etapa de correção é realizada somente com as observações dos marcadores fiduciais, assim como de maneira inversa. Por fim, a estimativa do estado e a matriz de covariância são atualizadas, e o *timestamp* do último evento processado é devidamente renovada.

Por fim, o procedimento completo do REKF-SLAM é resumido no Algoritmo 2.

Algoritmo 2 Algoritmo do REKF-SLAM

Modelo: Considere o modelo do sistema REKF-SLAM (32), (41) e o modelo de incerteza (23)–(24).
Entrada: Estimativa inicial $\widehat{\mathbf{x}}_{0|0}$, covariância inicial $\mathbf{P}_{0|0}$, entradas de odometria $\{\mathbf{u}_k\}$, observações dos marcadores $\{\mathbf{z}_{i,k}\}$ e observações do AMCL $\{\mathbf{z}_k^{\text{amcl}}\}$
Para $k = 0, 1, \ldots, N-1$ **faça**
    Linearize os modelos não lineares de processo e de observação
    Calcule as Jacobianas $\mathbf{F}_k$, $\mathbf{G}_k$ e $\mathbf{H}_k$ utilizando (46), (47) e (68)
    Calcule a pseudomedição $\mathbf{y}_k$ utilizando (22)
    Processamento dos marcos
    Para cada marcador observado $\mathbf{z}_{i,k}$ **faça**
        Calcule a distância de Mahalanobis de acordo com (77)
        Se $d_{M,k}^2 > \gamma$ **então**
            Rejeite as leituras do ArUco
        Senão
            Se o marcador m_i não estiver no mapa **então**
                Inicialize o marco conforme (53)
                Calcule $\mathbf{G}_{mx,k}$ por meio de (55) e $\mathbf{G}_{mz,k}$ por meio de (57)
                Aumente o vetor de estados
                Aumente a matriz de covariância
            end Se
            Calcule a observação predita conforme (62)
            Calcule a Jacobiana de observação $\mathbf{H}_{i,k}$ por meio de (65)
            Adapte a covariância do ArUco de acordo com a distância e a visibilidade do marcador
        end Se
    end Para
    Montagem das medições
    Se a estimativa do AMCL estiver disponível e sincronizada **então**
        Adicione $\mathbf{z}_k^{\text{amcl}}$, $\mathbf{H}_k^{\text{amcl}}$ e $\mathbf{R}_z^{\text{amcl}}$
    end Se
    Empilhe todas as observações conforme (40)
    Construa as matrizes empilhadas $\mathbf{H}_k$ e $\mathbf{R}_k$
    Calcule a inovação por meio de (75)
    Calcule a covariância da inovação conforme (76)
    Calcule as matrizes modificadas do sistema $\widehat{\mathbf{Q}}_k$, $\widehat{\mathbf{R}}_k$, $\widehat{\mathbf{F}}_k$ e $\widehat{\mathbf{H}}_k$ (Algoritmo 1 – Linhas 7–14)
    Atualize a estimativa robusta filtrada (correção) $\widehat{\mathbf{x}}_{k|k}$ e $\mathbf{P}_{k|k}$ (Algoritmo 1 – Linhas 16 e 17)
    Atualize a estimativa robusta predita (predição)
    Calcule o estado predito $\widehat{\mathbf{x}}_{k+1|k}$ utilizando (32)
    Calcule a covariância predita $\mathbf{P}_{k+1|k}$ (Algoritmo 1 – Linha 20)
    Retorne $\widehat{\mathbf{x}}_{k+1|k}, \mathbf{P}_{k+1|k}$
end Para

Algoritmo 3 Algoritmo do BDUE-SLAM

Modelo: Considere o modelo do sistema BDUE-SLAM (32), (41) e o modelo de incerteza (23)–(24).

Entrada: Estimativa inicial $\widehat{\mathbf{x}}_{0|0}$, covariância inicial $\mathbf{P}_{0|0}$, entradas de odometria $\{\mathbf{u}_k\}$, observações dos marcadores $\{\mathbf{z}_{i,k}\}$ e observações opcionais do AMCL $\{\mathbf{z}_k^{\text{amcl}}\}$.

Para $k = 0, 1, \ldots, N-1$ **faça**
  Linearize os modelos não lineares de processo e de observação
  Calcule as Jacobianas $\mathbf{F}_k$, $\mathbf{G}_k$ e $\mathbf{H}_k$ utilizando (46), (47) e (68)
  Calcule a pseudomedição $\mathbf{y}_k$ utilizando (22)
  Processamento dos marcos
  Para cada marcador observado $\mathbf{z}_{i,k}$ **faça**
    Calcule a distância de Mahalanobis de acordo com (77)
    Se $d_{M,k}^2 > \gamma$ **então**
      Rejeite as leituras do ArUco
    Senão
      Se o marcador m_i não estiver no mapa **então**
        Inicialize o marco conforme (53)
        Calcule $\mathbf{G}_{mx,k}$ por meio de (55) e $\mathbf{G}_{mz,k}$ por meio de (57)
        Aumente o vetor de estados
        Aumente a matriz de covariância
      end Se
      Calcule a observação predita conforme (62)
      Calcule a Jacobiana de observação $\mathbf{H}_{i,k}$ por meio de (65)
      Adapte a covariância do ArUco de acordo com a distância e a visibilidade do marcador
    end Se
  end Para
  Montagem das medições
  Se a estimativa do AMCL estiver disponível e sincronizada **então**
    Adicione $\mathbf{z}_k^{\text{amcl}}$, $\mathbf{H}_k^{\text{amcl}}$ e $\mathbf{R}_z^{\text{amcl}}$
  end Se
  Empilhe todas as observações conforme (40)
  Construa as matrizes empilhadas $\mathbf{H}_k$ e $\mathbf{R}_k$
  Calcule a inovação por meio de (75)
  Calcule a covariância da inovação conforme (76)
  Calcule as matrizes modificadas do sistema
  Se $\mathbf{H}_k\mathbf{M}_k = \mathbf{0}$ **então**
    Defina $\widehat{\lambda}_k = 0$
  Senão
    $\widehat{\lambda}_k = (1+\alpha_f)\left\|\mathbf{M}_k^T\mathbf{H}_k^T\mathbf{R}_k^{-1}\mathbf{H}_k\mathbf{M}_k\right\|$
  end Se
  $\widehat{\mathbf{P}}_{k|k} = \left(\mathbf{P}_{k|k}^{-1} + \widehat{\lambda}_k\mathbf{E}_{F,k}^T\mathbf{E}_{F,k}\right)^{-1}$
  $\widehat{\mathbf{R}}_k = \mathbf{R}_k - \widehat{\lambda}_k^{-1}\mathbf{H}_k\mathbf{M}_k\mathbf{M}_k^T\mathbf{H}_k^T$
  $\widehat{\mathbf{Q}}_k = \left(\mathbf{Q}_k^{-1} + \widehat{\lambda}_k\mathbf{E}_{G,k}^T\left(\mathbf{I} + \widehat{\lambda}_k\mathbf{E}_{F,k}\mathbf{P}_{k|k}\mathbf{E}_{F,k}^T\right)^{-1}\mathbf{E}_{G,k}\right)^{-1}$
  $\widehat{\mathbf{G}}_k = \mathbf{G}_k - \widehat{\lambda}_k\mathbf{F}_k\widehat{\mathbf{P}}_{k|k}\mathbf{E}_{F,k}^T\mathbf{E}_{G,k}$
  $\widehat{\mathbf{F}}_k = \left(\mathbf{F}_k - \widehat{\lambda}_k\widehat{\mathbf{G}}_k\widehat{\mathbf{Q}}_k\mathbf{E}_{G,k}^T\mathbf{E}_{F,k}\right)\left(\mathbf{I} - \widehat{\lambda}_k\widehat{\mathbf{P}}_{k|k}\mathbf{E}_{F,k}^T\mathbf{E}_{F,k}\right)$
  Atualize a estimativa robusta filtrada (correção)
  $\mathbf{R}_{e,k} = \mathbf{H}_k\mathbf{P}_{k|k-1}\mathbf{H}_k^T + \widehat{\mathbf{R}}_k$
  $\mathbf{K}_k = \mathbf{P}_{k|k-1}\mathbf{H}_k^T\mathbf{R}_{e,k}^{-1}$
  $\widehat{\mathbf{x}}_{k|k} = \widehat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k\left(\mathbf{z}_k - \mathbf{h}_k\left(\widehat{\mathbf{x}}_{k|k-1}\right)\right)$
  $\mathbf{P}_{k|k} = \left(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k\right)\mathbf{P}_{k|k-1}$
  Atualize a estimativa robusta predita (predição)
  Calcule o estado predito $\widehat{\mathbf{x}}_{k+1|k}$ utilizando (32)
  $\mathbf{P}_{k+1|k} = \widehat{\mathbf{F}}_k\widehat{\mathbf{P}}_{k|k}\widehat{\mathbf{F}}_k^T + \widehat{\mathbf{G}}_k\widehat{\mathbf{Q}}_k\widehat{\mathbf{G}}_k^T$
  Retorne $\widehat{\mathbf{x}}_{k+1|k}, \mathbf{P}_{k+1|k}$
end Para

Capítulo 5

Resultados

Este Capítulo avalia o desempenho do *framework* REKF-SLAM proposto, por meio de validações baseadas em simulação e em experimentos reais. Os resultados obtidos são comparados com um filtro padrão EKF-SLAM e com uma adaptação estendida do filtro BDU para o problema de SLAM, denominada BDUE-SLAM, sendo que a sua formulação é resumida no Algoritmo 3.

A acurácia da estimação da trajetória do robô é quantificada por meio do RMSE, calculada em relação à trajetória de referência usada como ground-truth. Para uma variável escalar genérica ρ, o RMSE é definido como

$$\mathrm{RMSE}(\rho) = \left(\frac{1}{N} \sum_{k=1}^{N} \left(\hat{\rho}_k - \rho_k^{\mathrm{ref}} \right)^2 \right)^{1/2},$$

em que $\hat{\rho}_k$ é o valor estimado, ρ_k^{ref} é o valor de referência correspondente, e N é o número de amostras avaliadas. O erro de guinada foi calculado utilizando a diferença angular ajustada, com o objetivo de evitar descontinuidades.

O restante deste Capítulo está organizado da seguinte forma: A Seção 5.0.1 apresenta os resultados dos experimentos realizados em ambiente simulado. Esses experimentos permitiram uma comparação controlada e quantitativa do algoritmo proposto e dos filtros avaliados. Em seguida, a Seção 5.0.2 apresenta a validação conduzida com um robô móvel diferencial real operando em um ambiente interno com marcadores fiduciais ArUco. As trajetórias resultantes e os erros de estimação são analisados a partir dos experimentos, a fim de avaliar a efetividade da formulação robusta proposta na melhoria da acurácia de localização e na mitigação da deriva do robô.

5.0.1 Validação por Simulação

Para validar o algoritmo REKF-SLAM, foram conduzidas simulações no Gazebo com o ROS 2 Humble (MACENSKI et al., 2022), utilizando o modelo TurtleBot 4 Standard. O ambiente simulado foi projetado para reproduzir as principais características estruturais e a disposição dos marcadores no cenário interno real, ao mesmo tempo em que fornece condições controladas e repetíveis para a avaliação da trajetória. O ambiente criado no Gazebo é apresentado nas Figuras 14-15. Os marcadores fiduciais foram gerados a partir de um dicionário ArUco 7×7, com comprimento lateral de 0.15 m, e distribuídos em diferentes alturas ao longo do ambiente, de modo a expor o sistema de localização a diferentes geometrias heterogêneas de observação entre câmera-marcador. Essa escolha de projeto é sustentada por estudos anteriores que mostram que a incerteza da pose dos marcadores fiduciais é afetada pela distância entre câmera-marcador, pelo ângulo de observação, pela posição do marcador e pela área aparente do marcador na imagem. Assim, o posicionamento dos marcadores tem impacto direto no desempenho da localização em ambientes internos (ADÁMEK et al., 2023; HINDERER; SCHEFFLER; YANG, 2025).

Figura 14 – Ambiente simulado no Gazebo utilizado para validar o algoritmo REKF-SLAM com um robô móvel de acionamento diferencial.

O modelo de simulação TurtleBot 4 Standard foi utilizado com o ROS 2 Nav2, utilizando um mapa previamente conhecido, que foi gerado antes do experimento. O robô utiliza um modelo de locomoção diferencial, e os dados da odometria das rodas foram utilizados como entrada para a etapa de predição dos algoritmos de filtragem. Os dados simulados do LiDAR foram utilizados para realizar a localização baseada em mapa via AMCL, enquanto o fluxo de imagens da câmera simulada foi utilizado para detectar os marcadores ArUco ao longo da trajetória do robô. Portanto, a configuração simulada preserva a mesma estrutura de estimação adotada nos experimentos reais, combinando odometria, estimativas de pose obtidas pelo AMCL, e observações de marcadores fiduci-

Figura 15 – Robô TurtleBot 4 no ambiente simulado criado no Gazebo, que foi utilizado para validar o algoritmo REKF-SLAM com um robô móvel de acionamento diferencial.

ais. No vídeo disponibilizado[1], é possível visualizar o robô se deslocando de acordo com o algoritmo do Nav2, e visualizando os marcadores ArUco na parede do corredor, em ambiente simulado.

As incertezas das matrizes do sistema adotadas na implementação do REKF-SLAM foram determinadas empiricamente. Especificamente, seus elementos foram ajustados por meio de uma série de procedimentos experimentais de calibração, nos quais diferentes níveis de incerteza foram avaliados. Os valores finais foram selecionados de acordo com a melhoria obtida na precisão da estimativa da trajetória e na consistência do filtro em comparação com o EKF-SLAM nominal.

Os parâmetros de ajuste robusto do REKF-SLAM foram definidos como $\mu = 0.3$ e $\xi = 0.1$, seguindo a orientação de (ROCHA; TERRA, 2021) de que $\xi \in (0, 1)$ geralmente conduz a resultados melhores e de que valores menores que μ aumentam a robustez quando o modelo está sujeito a incertezas significativas.

A matriz que modela a incerteza da transição de estados foi aplicada apenas no bloco da pose do robô:

$$\mathbf{M}_{1,k} = \begin{bmatrix} 0.001 & 0 \\ 0.0001 & 0 \\ 0 & 0.001 \\ \mathbf{0}_{6M\times 2} & \end{bmatrix} \in \mathbb{R}^{(3+6M)\times 2}.$$

Assim, nenhuma incerteza estrutural direta foi atribuída aos estados dos marcadores inicializados por meio de $\mathbf{M}_{1,k}$.

Para cada observação de marcador ArUco, a matriz que modela a incerteza de medição

[1] <https://youtu.be/dtI-DBVQAVA>

foi definida como

$$\mathbf{M}_2^A = \begin{bmatrix} 0.003 & 0 \\ 0.003 & 0 \\ 0.006 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \in \mathbb{R}^{6\times 2}.$$

Para a medição de pose AMCL, a matriz que modela a incerteza correspondente foi definida como

$$\mathbf{M}_2^M = \begin{bmatrix} 0.001 & 0 \\ 0.001 & 0 \\ 0 & 0.020 \end{bmatrix} \in \mathbb{R}^{3\times 2}.$$

Consequentemente, quando N_A marcadores ArUco e uma medição AMCL estão disponíveis, a matriz completa de incerteza de medição é dada por

$$\mathbf{M}_{2,k} = \begin{bmatrix} \mathbf{M}_2^A \\ \vdots \\ \mathbf{M}_2^A \\ \mathbf{M}_2^M \end{bmatrix} \in \mathbb{R}^{(6N_A+3)\times 2}.$$

Se apenas medições ArUco ou apenas medições AMCL estiverem disponíveis, então $r = 6N_A$ ou $r = 3$, respectivamente.

As matrizes que definem as direções das incertezas limitadas em norma nos modelos linearizados de transição e de medição foram especificadas como

$$\begin{aligned} \mathbf{E}_{F,k} &= \mathbf{0}_{2\times(3+6M)}, \\ \mathbf{E}_{G,k} &= \mathbf{0}_{2\times 2}, \\ \mathbf{E}_{H,k} &= \begin{bmatrix} 0.002 & 0.002 & 0 & \mathbf{0}_{1\times 6M} \\ 0 & 0 & 0.010 & \mathbf{0}_{1\times 6M} \end{bmatrix} \in \mathbb{R}^{2\times(3+6M)}. \end{aligned}$$

A formulação do filtro BDUE-SLAM acomoda incertezas paramétricas limitadas em norma apenas na matriz de transição de estados e na matriz de entrada do ruído de processo. Portanto, a descrição da incerteza fica restrita ao modelo de processo, enquanto as incertezas que afetam o modelo de observação não são explicitamente incorporadas ao projeto do filtro. Em todos os experimentos, o parâmetro α foi definido como 0.05.

A estrutura de incerteza é representada pela matriz $\mathbf{M}_k$, e pelas matrizes de ponderação $\mathbf{E}_{F,k}$ e $\mathbf{E}_{G,k}$, que estão associadas, respectivamente, aos modelos linearizados de transição de estados e de entrada. A matriz $\mathbf{M}_k$ foi aplicada apenas ao bloco da pose do

robô:

$$\mathbf{M}_k = \begin{bmatrix} 0.001 & 0 \\ 0.001 & 0 \\ 0 & 0.001 \\ \mathbf{0}_{6M\times 2} & \end{bmatrix} \in \mathbb{R}^{(3+6M)\times 2}.$$

A matriz associada às direções de incerteza no modelo linearizado de transição de estados foi definida como

$$\mathbf{E}_{F,k} = \begin{bmatrix} 0.0005 & 0.0005 & 0 & \mathbf{0}_{1\times 6M} \\ 0 & 0 & 0.0005 & \mathbf{0}_{1\times 6M} \end{bmatrix} \in \mathbb{R}^{2\times(3+6M)}.$$

De modo análogo, a matriz associada às direções de incerteza da entrada de odometria foi definida como

$$\mathbf{E}_{G,k} = \begin{bmatrix} 0.005 & 0 \\ 0 & 0.005 \end{bmatrix} \in \mathbb{R}^{2\times 2}.$$

Consequentemente, a correção robusta do BDUE-SLAM atua exclusivamente sobre a pose planar do robô e sobre os incrementos translacional e angular. Dessa forma, nenhum perfil de incerteza foi atribuído aos blocos de medição ArUco e AMCL, cuja confiabilidade varia com a visibilidade dos marcadores, com a geometria câmera-marcador e com a consistência da estimativa de localização baseada em mapa. Essa restrição é precisamente tratada pela formulação proposta com base em (ROCHA; TERRA, 2021), a qual acomoda incertezas limitadas em norma em todas as matrizes paramétricas, incluindo o modelo de medição, permitindo que perfis de incerteza específicos para cada sensor sejam atribuídos às observações ArUco e AMCL.

O ambiente simulado fornece diretamente uma trajetória de *ground truth* a partir do Gazebo. Isso permite uma avaliação quantitativa e qualitativa controlada do erro de estimação, sem depender de um sistema auxiliar de odometria. Todas as trajetórias estimadas foram interpoladas para os instantes de tempo da trajetória de *ground truth* e avaliadas apenas sobre um intervalo de tempo em comum. As trajetórias obtidas ao longo do tempo são apresentadas na Figura 16, em que o quadrado vermelho destaca o intervalo entre 150 s e 154 s para melhor visualização, que é ampliado na Figura 17. A Figura ampliada destaca as diferenças locais entre os estimadores. Em torno de 152 s, o REKF-SLAM acompanha a trajetória de referência de forma mais próxima, enquanto o EKF-SLAM apresenta desvios transitórios maiores, sugerindo menor sensibilidade a incertezas locais de modelagem e de medição. Embora o BDUE-SLAM também acompanhe a trajetória de *ground truth* de forma próxima em determinados intervalos, uma inspeção detalhada mostra que o REKF-SLAM fornece desempenho global de trajetória mais consistente.

Esses intervalos foram selecionados aleatoriamente ao longo da trajetória e foram empregados exclusivamente para facilitar a avaliação visual das diferenças locais entre os

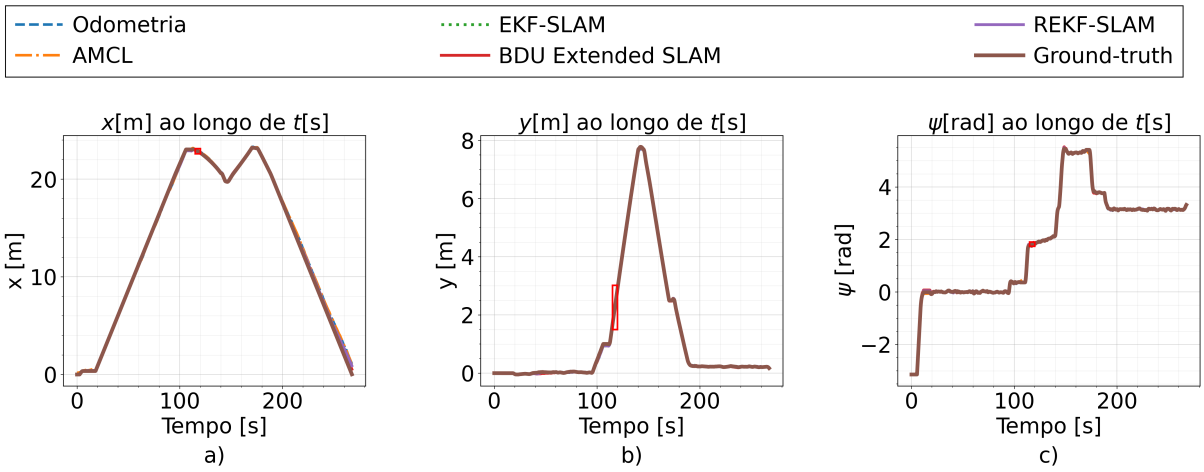

Figura 16 – Comparação das trajetórias x, y e ψ ao longo do tempo para os dados simulados.

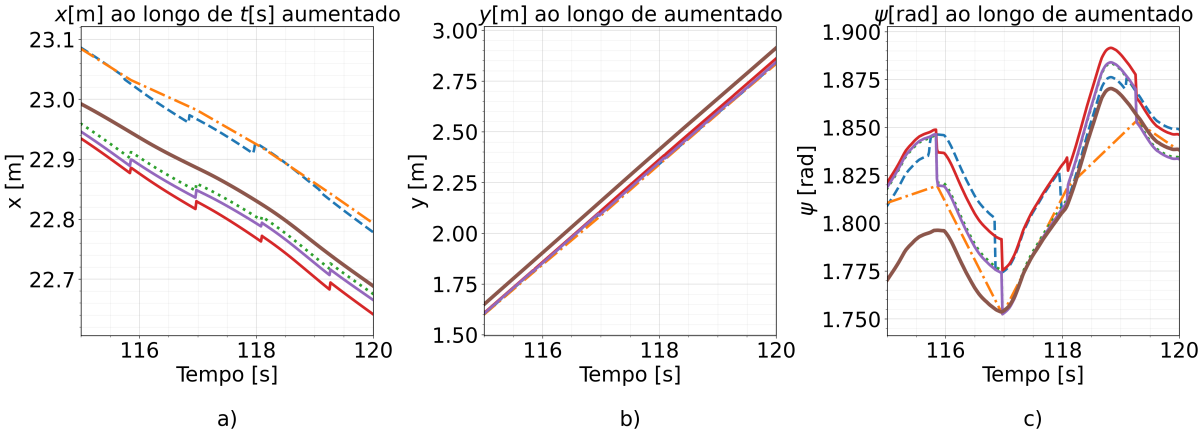

Figura 17 – Ampliação do intervalo de 115 s a 120 s em x, y e ψ ao longo do tempo para as trajetórias da Figura 16.

estados estimados. A seleção não foi baseada no desempenho relativo de nenhum dos métodos e, portanto, não introduz um viés preferencial em favor de um estimador específico. A avaliação global de desempenho é sustentada pelas métricas quantitativas calculadas sobre a trajetória completa.

É importante ressaltar que os valores das matrizes de incerteza foram ajustados empiricamente com base em sucessivos ensaios experimentais. A configuração final foi definida após a avaliação de diferentes conjuntos de parâmetros, sendo selecionada aquela que proporcionou o desempenho mais satisfatório do REKF-SLAM em comparação com o EKF-SLAM nominal.

A Figura 18 apresenta a sobreposição de todas as trajetórias sobre o mapa do Gazebo, que tem como objetivo emular o cenário interno real utilizado para a validação experimental.

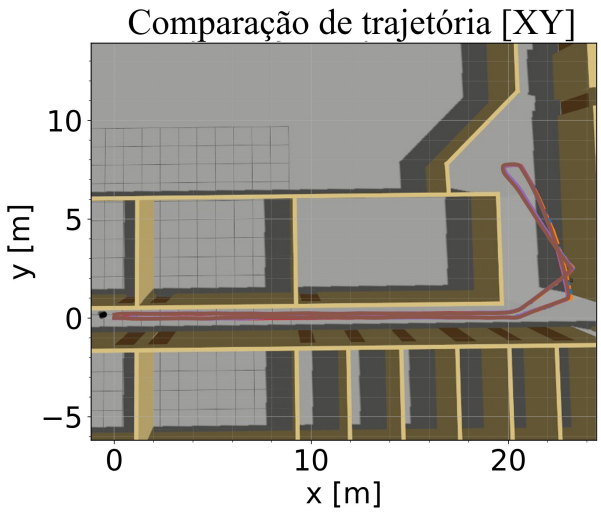

Figura 18 – Trajetória no plano x-y realizada pelo robô no ambiente simulado.

Os resultados de simulação apresentados na Tabela 1 indicam que o *framework* proposto forneceu a melhor acurácia translacional entre os métodos avaliados. Em comparação com o EKF-SLAM padrão, o REKF-SLAM proposto reduziu o RMSE(x) de 0.2170 m para 0.1274 m e o RMSE(y) de 0.0350 m para 0.0245 m, representando uma melhoria global de aproximadamente 41%. O BDUE-SLAM, por outro lado, permaneceu próximo ao filtro nominal nas componentes translacionais, melhorando o RMSE da posição planar em apenas cerca de 3%, e a sua única vantagem foi uma redução pontual no erro de guinada, RMSE(ψ), de 0.0183 rad para 0.0175 rad, inferior a 0.05 graus. Esse comportamento é consistente com a limitação estrutural discutida anteriormente. As fontes dominantes de erro nesse cenário encontram-se no lado da medição, mais especificamente no ruído da pose dos marcadores, que aumenta com a distância câmera-marcador e com o ângulo de observação, e nas inconsistências residuais da estimativa do AMCL. Ao atribuir perfis explícitos de incerteza para cada bloco de medição, a formulação proposta converte essa liberdade adicional de modelagem em uma redução substancial da deriva translacional, ao custo de um pequeno aumento no erro de guinada, de 0.0058 rad, aproximadamente 0.33 graus. Essa tendência está alinhada com o *benchmark* linear relatado em RKF, no qual o mesmo filtro superou o filtro *BDU Extended* (BDUE) em aproximadamente 12 dB na variância do erro em regime permanente.

Embora o REKF de referência tenha alcançado o menor RMSE de guinada, a formulação proposta produziu erros translacionais substancialmente menores. Isso sugere que a estrutura de incerteza adotada beneficia principalmente a estimação da posição, em vez da correção angular.

A formulação REKF-SLAM utilizada para comparação também melhorou a estimativa EKF-SLAM, embora com efeito menor nas componentes translacionais. Entretanto, essa formulação alcançou o menor erro de guinada entre os filtros avaliados, reduzindo o RMSE(ψ) de 0.0183 rad para 0.0175 rad. Esses resultados sugerem que, no cenário

Tabela 1 – RMSE da estimação da trajetória em relação à trajetória de *ground truth* do Gazebo.

Método	x (m)	y (m)	ψ (rad)
EKF-SLAM	0.2170	0.0350	0.0183
BDUE-SLAM	0.2100	0.0344	**0.0175**
REKF-SLAM	**0.1274**	**0.0245**	0.0241

simulado, a formulação robusta proposta foi mais efetiva na redução da deriva de posição, enquanto o BDUE-SLAM preservou uma consistência angular ligeiramente superior.

5.0.2 Validação Experimental

O cenário experimental consistiu em um ambiente interno instrumentado com 20 marcadores fiduciais sob condições de iluminação não uniformes. Todos os marcadores foram gerados a partir de um dicionário ArUco 7×7 e possuíam comprimento lateral de 0.15 m. Eles foram impressos em folhas A4 e fixados verticalmente às paredes a aproximadamente 0.63 m acima do piso, correspondendo a uma altura relativa de cerca de 0.42 m em relação à plataforma robótica, conforme mostrado na Figura 19. Os marcadores foram mantidos em altura fixa para assegurar que permanecessem dentro do campo de visão da câmera. Os experimentos foram realizados em um notebook equipado com processador Intel Core i5-12450HX e 16 GB de RAM, executando o Ubuntu 22.04 e ROS 2 Humble. O robô móvel utilizado nos testes é apresentado na Figura 20. O *framework* EKF-SLAM proposto foi desenvolvido em Python e C++, e a detecção *online* dos marcadores foi realizada com OpenCV em uma câmera RGBD Intel RealSense D435i.

Figura 19 – Ambiente utilizado nos testes experimentais com um robô móvel de acionamento diferencial.

Como nenhum sistema externo de posicionamento absoluto estava disponível durante os experimentos, a trajetória de referência foi estimada utilizando o Point-LIO (HE et al., 2023a), um método de odometria LiDAR-inercial com alta consistência e acurácia local. O Point-LIO foi executado utilizando as medições de um LiDAR 3D Unitree L2 fixado ao robô móvel durante os experimentos. Embora essa estimativa não possa ser conside-

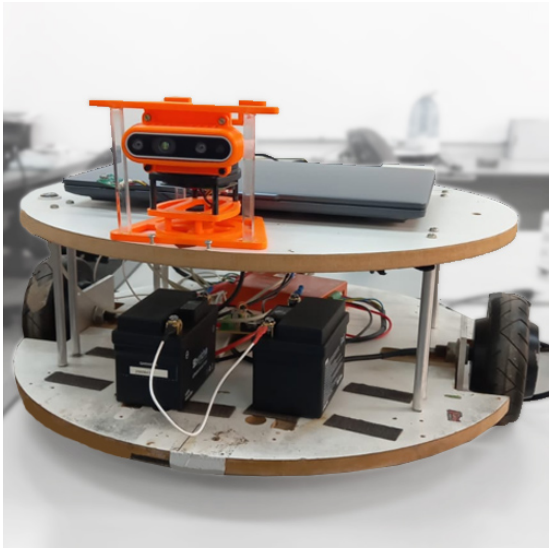

Figura 20 – Robô móvel de acionamento diferencial utilizado para a validação experimental.

rada um *ground truth* absoluto, ela fornece uma referência confiável para comparações de desempenho relativo entre os métodos avaliados. Durante a aquisição dos dados, o robô navegou de forma autônoma utilizando o ROS 2 Nav2 e um mapa estático bidimensional de ocupação previamente gerado com um sensor laser RPLIDAR A1. Essa configuração forneceu um cenário representativo de navegação interna para avaliar o método proposto em condições práticas de operação.

Figura 21 – Reconstrução com o RTAB-Map.

O desempenho de localização foi avaliado ao programar o robô móvel para seguir uma trajetória predefinida que cobria a região contendo os marcadores fiduciais. Além disso, para validar o desempenho do algoritmo, foi utilizado o RTAB-Map (LABBÉ; MICHAUD, 2019) a partir das informações obtidas pela profundidade da câmera, para reconstruir o mapa tridimensional da região onde o robô percorreu. É possível observar o resultado dessa ferramenta de acordo com a Figura 21, sendo que na Figura 22 é possível visualizar o ambiente real.

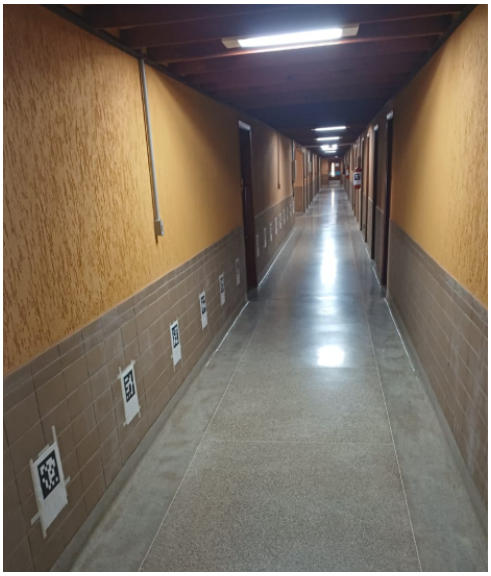

Figura 22 – Ambiente real com marcadores ArUco.

A odometria das rodas, a estimativa EKF-SLAM padrão, o BDUE-SLAM e as estimativas REKF-SLAM foram registrados sobre a mesma execução da trajetória. As trajetórias resultantes foram então comparadas com a trajetória de referência do Point-LIO para quantificar os efeitos das observações dos marcadores fiduciais e da filtragem robusta na redução da deriva e na consistência global da estimação de pose.

O REKF-SLAM foi implementado utilizando uma arquitetura de filtragem robusta preditor-corretor para modelos de espaço de estados incertos. A dimensão correspondente do estado é $n = 3 + 6M$, em que M é o número de marcadores inicializados no mapa. De modo semelhante às simulações, os parâmetros de ajuste robusto foram definidos como $\mu = 0.3$ e $\xi = 0.1$.

A matriz que modela a incerteza na transição de estados foi aplicada apenas ao bloco da pose do robô:

$$\mathbf{M}_{1,k} = \begin{bmatrix} 0.001 & 0 \\ 0.0001 & 0 \\ 0 & 0.001 \\ \mathbf{0}_{6M\times 2} & \end{bmatrix} \in \mathbb{R}^{(3+6M)\times 2}.$$

Para cada observação ArUco, o perfil de incerteza correspondente foi definido como

$$\mathbf{M}_2^A = \begin{bmatrix} 0.003 & 0 \\ 0 & 0 \\ 0.002 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \in \mathbb{R}^{6\times 2}.$$

Para a medição de pose AMCL, o perfil de incerteza correspondente foi definido como

$$\mathbf{M}_2^M = \begin{bmatrix} 0.001 & 0 \\ 0.020 & 0 \\ 0 & 0 \end{bmatrix} \in \mathbb{R}^{3\times 2}.$$

Consequentemente, quando N_A marcadores ArUco e uma medição AMCL estão disponíveis, a matriz completa de incerteza de medição é dada por

$$\mathbf{M}_{2,k} = \begin{bmatrix} \mathbf{M}_2^A \\ \vdots \\ \mathbf{M}_2^A \\ \mathbf{M}_2^M \end{bmatrix} \in \mathbb{R}^{(6N_A+3)\times 2}.$$

Se apenas medições ArUco ou apenas medições AMCL estiverem disponíveis, então $r = 6N_A$ ou $r = 3$, respectivamente.

As matrizes que definem as direções das incertezas limitadas em norma nos modelos linearizados de transição e de medição foram definidas como

$$\begin{aligned} \mathbf{E}_{F,k} &= \mathbf{0}_{2\times(3+6M)}, \\ \mathbf{E}_{G,k} &= \mathbf{0}_{2\times 2}, \\ \mathbf{E}_{H,k} &= \begin{bmatrix} 0.002 & 0.002 & 0 & \mathbf{0}_{1\times 6M} \\ 0 & 0 & 0.010 & \mathbf{0}_{1\times 6M} \end{bmatrix} \in \mathbb{R}^{2\times(3+6M)}. \end{aligned}$$

Essa parametrização concentra a correção robusta na pose planar do robô, nas medições de posição ArUco e na estimativa global de pose AMCL, preservando simultaneamente a estrutura recursiva e, consequentemente, a simplicidade computacional do estimador EKF-SLAM.

O *benchmark* BDUE-SLAM foi configurado exatamente como na Seção 5.0.1, com $\alpha = 0.05$ e as mesmas matrizes $\mathbf{M}_k$, $\mathbf{E}_{F,k}$ e $\mathbf{E}_{G,k}$. Como anteriormente, sua correção robusta é restrita à pose planar do robô e aos canais de entrada da odometria, sem uma estrutura de incerteza disponível para os blocos de medição ArUco e AMCL.

O desempenho do REKF-SLAM foi avaliado comparando as trajetórias estimadas com a trajetória de referência fornecida pelo Point-LIO. Como as trajetórias comparadas são geradas em diferentes taxas de amostragem, todas as estimativas foram interpoladas para os instantes de tempo da referência e avaliadas apenas sobre o intervalo temporal comum. A Figura 23 compara a evolução temporal de x, y e ψ para todos os estimadores. Assim como na análise de simulação, uma região de interesse entre 125 s e 130 s é indicada na Figura 23 por um quadrado vermelho e ampliada na Figura 24.

Entre os métodos avaliados, o REKF-SLAM permanece mais próximo da referência Point-LIO durante a maior parte do experimento. Na Figura 24, o trecho ampliado revela que o REKF-SLAM apresenta menores desvios transitórios, particularmente na

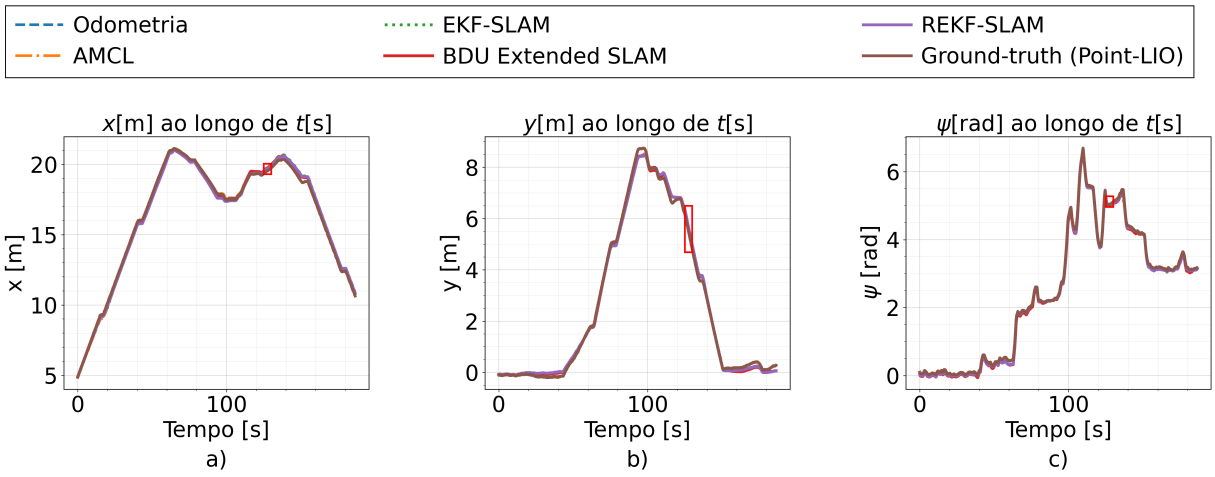

Figura 23 – Comparação das trajetórias x, y e ψ ao longo do tempo para os dados experimentais.

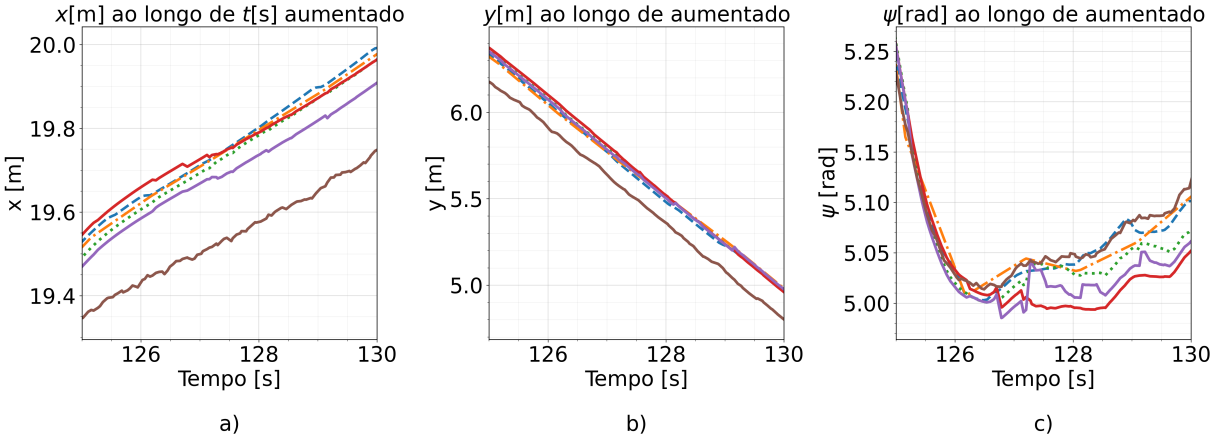

Figura 24 – Ampliação do intervalo de 125 s a 130 s em x, y e ψ ao longo do tempo para as trajetórias da Figura 23.

coordenada x, indicando que o mecanismo de correção reduz efetivamente os erros locais de estimação.

A Tabela 2 resume os erros de estimação de trajetória obtidos com o EKF-SLAM padrão, com a adaptação BDUE-SLAM de (SAYED, 2001) e com a formulação REKF-SLAM proposta. Conforme apresentado na tabela, o REKF-SLAM alcançou o melhor desempenho translacional no cenário experimental, fornecendo uma melhoria global de aproximadamente 18.9% em relação ao EKF-SLAM. Em contraste, o BDUE-SLAM apresentou apenas ganhos pontuais em relação à trajetória de referência, com melhoria de 1% no RMSE da posição planar. Esse resultado confirma, sob condições reais de sensoriamento, que modelar a incerteza apenas no modelo de movimento é insuficiente quando os erros de medição constituem a fonte dominante de incerteza na localização. Embora a formulação proposta tenha produzido um RMSE de guinada superior ao dos demais filtros,

Tabela 2 – RMSE da estimação da trajetória em relação à trajetória de referência Point-LIO.

Método	x (m)	y (m)	ψ (rad)
EKF-SLAM	0.2170	0.1359	**0.0526**
BDUE-SLAM	0.2132	0.1356	0.0528
REKF-SLAM	**0.1634**	**0.1282**	0.0673

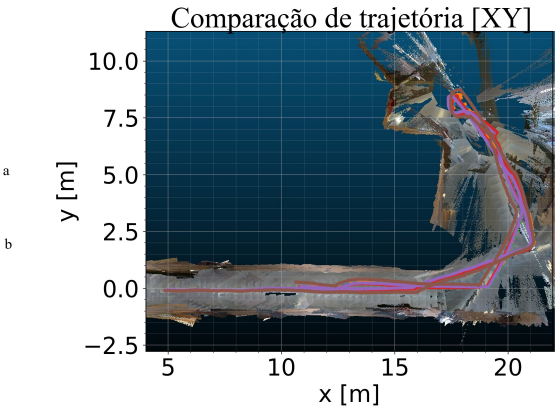

Figura 25 – Trajetória no plano x-y realizada pelo robô no mapa reconstruído utilizando RTAB-Map.

0.0673 rad contra 0.0526 rad, correspondente a um aumento de aproximadamente 0.85 graus, a redução substancial da deriva de posição resultou em uma acurácia global maior de localização para a navegação em ambientes internos.

Por fim, a Figura 25 apresenta as trajetórias no plano x-y do REKF-SLAM proposto, BDUE-SLAM, EKF-SLAM, AMCL, odometria e da referência Point-LIO, sobrepostas ao mapa reconstruído com RTAB-Map utilizando dados de profundidade da câmera Real-Sense D435i. É possível visualizar o mapa reconstruído com o RTAB-Map, os marcadores ArUco sendo detectados em tempo real e a trajetória percorrida com o robô utilizando o EKF-SLAM em um vídeo disponibilizado[2].

[2] <https://www.youtube.com/watch?v=e3HNV0N4Eg0>

Capítulo 6

Conclusões e Trabalhos Futuros

Este trabalho apresentou um *framework* REKF-SLAM para melhorar a localização de robôs móveis em ambientes internos, integrando incrementos de odometria das rodas, observações de marcadores fiduciais ArUco e estimativas auxiliares de pose fornecidas pelo algoritmo AMCL. A formulação proposta representa a pose do robô em um espaço de estados planar e incorpora os marcadores fiduciais como marcadores de seis dimensões no vetor de estados aumentado. Ao explorar as referências espaciais fornecidas por marcadores previamente observados e a informação de pose global proveniente do AMCL, o sistema reduz a deriva acumulada associada à odometria e melhora a consistência da trajetória estimada.

A principal vantagem da formulação proposta é a representação explícita da incerteza tanto no modelo de movimento quanto nos modelos de medição, permitindo que diferentes perfis de incerteza sejam atribuídos às observações ArUco e AMCL.

A formulação proposta, construída com base no filtro robusto preditor-corretor RKF (ROCHA; TERRA, 2021), apresentou as melhorias mais consistentes na acurácia translacional. Tanto nas avaliações simuladas quanto nas experimentais, o uso de matrizes estruturadas de incerteza permitiu que o filtro considerasse explicitamente as incertezas no modelo de movimento do robô e nos blocos heterogêneos de medição. Isso foi particularmente relevante para as observações ArUco e AMCL, cuja confiabilidade pode variar de acordo com a visibilidade dos marcadores, a geometria câmera-marcador, as condições de iluminação e a consistência da estimativa de localização baseada em mapa. Como resultado, o REKF-SLAM proposto reduziu a deriva de posição, preservando a estrutura recursiva e a simplicidade computacional do estimador EKF-SLAM.

Na simulação, o *framework* proposto reduziu o RMSE da posição planar em aproximadamente 41% em relação ao EKF-SLAM convencional. Na avaliação experimental, a

redução alcançou aproximadamente 19%, demonstrando que a formulação proposta permanece efetiva sob condições reais de sensoriamento.

Além disso, os resultados também revelaram uma degradação moderada na acurácia da estimação de guinada, com aumento de aproximadamente 0.85 graus no RMSE(ψ) no cenário experimental, indicando que trabalhos futuros podem continuar a investigação por estruturas alternativas de incerteza capazes de melhorar simultaneamente o desempenho da estimação translacional e angular.

A comparação com o BDUE-SLAM, no qual as matrizes de incerteza são limitadas ao modelo de movimento linearizado, conforme proposto por (SAYED, 2001), reforçou essa conclusão. Restringir a estrutura de incerteza aos canais de transição de estados e de entrada resultou apenas em ganhos pontuais em relação ao EKF-SLAM nominal em ambos os cenários, preservando, no máximo, uma consistência angular ligeiramente melhor. Portanto, as melhorias translacionais obtidas com o método proposto podem ser atribuídas à modelagem da incerteza no lado das medições, essa configuração não possui equivalência na formulação do filtro BDUE.

6.1 Produção Intelectual

Durante o desenvolvimento desta pesquisa, foi publicado um artigo em conferência e foi feita a submissão de um artigo para revista, que estão listados respectivamente:

- Luna, F., Caldas, K., Campos, L., Vivaldini, K., Pazelli, T., Inoue, R.: Improving indoor mobile robot localization using integrated EKF-SLAM, AMCL and ArUco fiducial markers. In: **2026 Brazilian Conference on Robotics (CROS)**, vol. 2, pp. 1–6 (2026).

- Luna, F., Caldas, K., Campos, L., Terra, M., Vivaldini, K., Pazelli, T., Inoue, R.: Robust EKF-SLAM for Indoor Mobile Robot Localization subject to Parametric Uncertainties. **Submetido a Journal** (2026).

6.2 Limitações e Dificuldades

Inicialmente, este trabalho foi projetado considerando aplicações em veículos aéreos não tripulados. Entretanto, durante o desenvolvimento da pesquisa, verificou-se que a validação experimental com esse tipo de plataforma apresentaria limitações práticas importantes, principalmente relacionadas à repetibilidade dos ensaios, à disponibilidade de modelos de simulação compatíveis, à estabilidade da odometria embarcada e à complexidade operacional dos testes em ambientes internos. Esses fatores poderiam dificultar a avaliação sistemática dos algoritmos de estimação propostos, uma vez que variações não

controladas da plataforma poderiam interferir diretamente na análise do desempenho do estimador.

Diante dessas limitações, optou-se pela utilização de um robô móvel diferencial como plataforma experimental. Essa escolha permitiu maior controle sobre as trajetórias executadas, maior segurança durante os ensaios e melhor reprodutibilidade das condições de teste. Além disso, o robô móvel apresentou maior praticidade para a realização de experimentos em corredores internos, possibilitando a repetição de trajetórias, a aquisição contínua de dados sensoriais e a comparação mais consistente entre diferentes configurações dos algoritmos avaliados.

Outro aspecto relevante foi a flexibilidade da plataforma terrestre para a integração de sensores, componentes auxiliares e modificações estruturais conforme as necessidades do experimento. Essa característica foi essencial durante a validação em ambiente interno, especialmente porque o corredor utilizado nos testes não possuía iluminação natural. Em ensaios realizados no período noturno, a baixa luminosidade dificultava a detecção dos marcadores fiduciais pela câmera. Para contornar esse problema, foi acoplada uma fonte de iluminação à parte frontal do robô, permitindo a realização dos experimentos em diferentes condições de iluminação e aumentando a robustez do procedimento experimental.

Assim, embora a aplicação inicial considerada envolvesse veículos aéreos não tripulados, a adoção de um robô móvel diferencial mostrou-se mais adequada para a etapa de validação experimental deste trabalho. Essa escolha favoreceu a praticidade dos testes, a repetibilidade dos experimentos e o controle das condições de aquisição, permitindo uma avaliação mais consistente do *framework* de localização proposto.

Outra dificuldade relevante esteve relacionada ao ajuste dos parâmetros das matrizes de incerteza utilizadas nos filtros REKF-SLAM e BDUE. A obtenção de valores adequados exigiu uma etapa extensa de testes, tanto no ambiente simulado quanto no experimental, a fim de identificar combinações capazes de melhorar o desempenho dos estimadores. Para isso, foram gerados *rosbags* contendo os tópicos necessários para posterior reprodução dos dados em conjunto com os algoritmos de detecção de marcadores e de filtragem. Esse procedimento tornou o processo de avaliação mais lento, pois cada *rosbag* precisou ser executado diversas vezes para testar diferentes configurações paramétricas. Além disso, como a reprodução dos dados não pôde ser acelerada de forma apropriada sem comprometer a sincronização dos tópicos, a análise demandou tempo considerável. Dessa forma, a etapa de ajuste das matrizes do sistema constituiu uma das fases mais trabalhosas da validação experimental e simulada.

Por fim, uma última limitação deste trabalho está relacionada à obtenção de uma trajetória de referência para a validação do sistema no robô físico. Na ausência de um sistema externo de posicionamento absoluto, como câmeras de captura de movimento ou outras plataformas dedicadas à geração de *ground truth*, foi utilizado um LiDAR 4D em conjunto com o algoritmo Point-LIO como referência para a avaliação experimental.

Embora essa abordagem forneça uma estimativa de trajetória com boa consistência local, ela não pode ser considerada um *ground truth* absoluto. Isso ocorre porque a qualidade da nuvem de pontos pode ser afetada por características do ambiente, oclusões, reflexões, degradação das medições e eventuais falhas do próprio sensor. Portanto, os resultados experimentais devem ser interpretados como uma comparação relativa em relação a uma referência LiDAR-inercial, e não como uma validação baseada em uma medição absoluta e livre de incertezas.

6.3 Contribuições

As principais contribuições deste trabalho são:

- ❏ Desenvolvimento e implementação de um *framework* robusto de localização para robôs móveis em ambientes internos, baseado na fusão de sensores heterogêneos e na representação explícita de incertezas paramétricas e erros de modelagem;
- ❏ Extensão de uma formulação de Filtro de Kalman Robusto, originalmente desenvolvida para sistemas lineares discretos no tempo, para uma estrutura não linear de EKF-SLAM aplicada à robótica móvel;
- ❏ Integração de odometria das rodas, observações de marcadores fiduciais ArUco e estimativas auxiliares de pose provenientes do AMCL em uma arquitetura multissensorial de localização;
- ❏ Investigação do uso de matrizes robustas em diferentes estruturas de incerteza, considerando sua aplicação apenas ao estado do robô e sua aplicação combinada ao estado do robô e às medições;
- ❏ Avaliação comparativa do *framework* proposto em relação ao EKF-SLAM convencional e ao BDUE-SLAM, considerando métricas de trajetória, redução de deriva e consistência da localização;
- ❏ Validação da abordagem proposta em simulações e experimentos reais conduzidos em ambiente interno sem acesso ao GNSS, demonstrando sua aplicabilidade em condições realísticas de navegação.

6.4 Trabalhos Futuros

Para os trabalhos futuros, é possível considerar as seguintes possibilidades de pesquisa, que estão listadas mas não limitadas a:

- ❏ Investigar estratégias adaptativas para selecionar e ponderar observações de marcadores fiduciais de acordo com a qualidade da detecção, o ângulo de observação, a distância e a distribuição dos marcadores no ambiente;
- ❏ Avaliar diferentes famílias de marcadores fiduciais, como AprilTag, e configurações alternativas de posicionamento dos marcadores, a fim de analisar sua influência na robustez da atualização do SLAM;
- ❏ Estender o filtro REKF-SLAM proposto para testes em Veículos Aéreos Não Tripulados e em outras plataformas robóticas, considerando diferentes configurações de sensores e variações nas condições do ambiente.
- ❏ Realizar novos testes com o REKF-SLAM proposto utilizando outras configurações de sensores, como sistemas multi-câmera, IMU e sensores *Ultra-Wide Band* (UWB).

Referências

ADÁMEK, R. et al. Analytical models for pose estimate variance of planar fiducial markers for mobile robot localisation. **Sensors**, v. 23, n. 12, p. 5746, 2023. Disponível em: <https://doi.org/10.3390/s23125746>. Acesso em: 21 jul. 2026.

AFSHA, S.; SAZID, M. M. AR-TurtleSLAM: EKF-based localization and mapping using ArUco feature detection on mobile robots. In: **2024 IEEE International Conference on Computing, Applications and Systems (COMPAS)**. Piscataway, NJ: IEEE, 2024. p. 1–6. Disponível em: <https://doi.org/10.1109/COMPAS60761.2024.10797053>. Acesso em: 21 jul. 2026.

ALAM, M. S.; GULLU, A. I.; GUNES, A. Fiducial markers and particle filter based localization and navigation framework for an autonomous mobile robot. **SN Computer Science**, v. 5, n. 6, p. 748, 2024. Disponível em: <https://doi.org/10.1007/s42979-024-03090-y>. Acesso em: 21 jul. 2026.

ALI, R. et al. An indoor mobile robot localization in perspective of analysis and performance using unscented Kalman filter. **Trends in Sciences**, Nakhon Si Thammarat, Thailand, v. 19, n. 7, p. 3097, mar. 2022. Disponível em: <https://doi.org/10.48048/tis.2022.3097>. Acesso em: 21 jul. 2026.

ALSADIK, B.; KARAM, S. The simultaneous localization and mapping (SLAM): An overview. **Journal of Applied Science and Technology Trends**, v. 2, n. 2, p. 147–158, jul. 2021. Disponível em: <https://doi.org/10.38094/jastt204117>. Acesso em: 21 jul. 2026.

ANDERSON, B. D. O.; MOORE, J. B. **Optimal Filtering**. Englewood Cliffs, NJ: Prentice-Hall, 1979.

AQEL, M. O. A. et al. Review of visual odometry: types, approaches, challenges, and applications. **SpringerPlus**, v. 5, n. 1, p. 1897, 2016. Disponível em: <https://doi.org/10.1186/s40064-016-3573-7>. Acesso em: 21 jul. 2026.

BAHAROM, A. K. et al. Towards modelling autonomous mobile robot localization by using sensor fusion algorithms. In: **2020 IEEE 10th International Conference on System Engineering and Technology (ICSET)**. Piscataway, NJ: IEEE, 2020. p. 185–190. Disponível em: <https://doi.org/10.1109/ICSET51301.2020.9265372>. Acesso em: 21 jul. 2026.

BAI, Y.; XING, F.; LIU, W. Research on autonomous localization method of coal mine underground mobile robot based on multi sensor fusion. In: **2023 IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC)**. Piscataway, NJ: IEEE, 2023. p. 1–6. Disponível em: <https://doi.org/10.1109/ICSPCC59353.2023.10400323>. Acesso em: 21 jul. 2026.

BENLIGIRAY, B.; TOPAL, C.; AKINLAR, C. STag: A stable fiducial marker system. **Image and Vision Computing**, v. 89, p. 158–169, 2019. Disponível em: <https://doi.org/10.1016/j.imavis.2019.06.007>. Acesso em: 21 jul. 2026.

BRADSKI, G.; KAEHLER, A. **Learning OpenCV: Computer vision with the OpenCV library**. Sebastopol, CA: O'Reilly Media, 2008.

BRAUN, J. et al. A robot localization proposal for the RobotAtFactory 4.0: A novel robotics competition within the industry 4.0 concept. **Frontiers in Robotics and AI**, v. 9, 2022. Disponível em: <https://doi.org/10.3389/frobt.2022.1023590>. Acesso em: 21 jul. 2026.

BROWN, D. C. Close-range camera calibration. **Photogrammetric Engineering**, v. 37, n. 8, p. 855–866, 1971.

CALDAS, K. A. Q. et al. Discrete-time Markovian jump linear robust filtering for visual and GPS aided magneto-inertial navigation. **IEEE Access**, v. 13, p. 7590–7602, 2025. Disponível em: <https://doi.org/10.1109/ACCESS.2025.3527449>. Acesso em: 21 jul. 2026.

CHAI, W.; LI, C.; LI, Q. Multi-sensor fusion-based indoor single-track semantic map construction and localization. **IEEE Sensors Journal**, v. 23, n. 3, p. 2470–2480, 2023. Disponível em: <https://doi.org/10.1109/JSEN.2022.3226821>. Acesso em: 21 jul. 2026.

CHARROUD, A. et al. Rapid localization and mapping method based on adaptive particle filters. **Sensors**, v. 22, n. 23, p. 9439, 2022. Disponível em: <https://doi.org/10.3390/s22239439>. Acesso em: 21 jul. 2026.

CHEN, J. et al. Power system fusion state estimation based on a switched system model: Comparison between EKF and UKF. In: **2019 Chinese Control Conference (CCC)**. [s.n.], 2019. p. 7280–7284. Disponível em: <https://doi.org/10.23919/ChiCC.2019.8865322>. Acesso em: 21 jul. 2026.

CHUNG, M.-A.; LIN, C.-W. An improved localization of mobile robotic system based on AMCL algorithm. **IEEE Sensors Journal**, v. 22, n. 1, p. 900–908, 2022. Disponível em: <https://doi.org/10.1109/JSEN.2021.3126605>. Acesso em: 21 jul. 2026.

CORKE, P. I. **Robotics, Vision and Control: Fundamental Algorithms in MATLAB**. 2. ed. Cham: Springer, 2017. v. 118. (Springer Tracts in Advanced Robotics, v. 118). Disponível em: <https://doi.org/10.1007/978-3-319-54413-7>. Acesso em: 21 jul. 2026.

CRAIG, J. J. **Robótica**. 3. ed. São Paulo: Pearson Education do Brasil, 2013. ISBN 978-85-8143-128-4.

DELLAERT, F. et al. Monte Carlo localization for mobile robots. In: **Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No.99CH36288C)**. Piscataway, NJ: IEEE, 1999. v. 2, p. 1322–1328. Disponível em: <https://doi.org/10.1109/ROBOT.1999.772544>. Acesso em: 21 jul. 2026.

DHAIGUDE, V.; KULKARNI, A. A comprehensive review of sensor fusion techniques in SLAM: Trends, challenges and future directions. In: **2025 1st International Conference on AIML-Applications for Engineering & Technology (ICAET)**. Piscataway, NJ: IEEE, 2025. p. 1–8. Disponível em: <https://doi.org/10.1109/ICAET63349.2025.10932171>. Acesso em: 21 jul. 2026.

DOMINGUEZ, S.; KEATON, T.; SAYED, A. A robust finger tracking method for multimodal wearable computer interfacing. **IEEE Transactions on Multimedia**, v. 8, n. 5, p. 956–972, 2006. Disponível em: <https://doi.org/10.1109/TMM.2006.879872>. Acesso em: 21 jul. 2026.

DOUGLAS, D. H.; PEUCKER, T. K. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. **Cartographica: The International Journal for Geographic Information and Geovisualization**, v. 10, n. 2, p. 112–122, 1973. Disponível em: <https://doi.org/10.3138/FM57-6770-U75U-7727>. Acesso em: 21 jul. 2026.

EMAN, A.; RAMDANE, H. Mobile robot localization using extended Kalman filter. In: **2020 3rd International Conference on Computer Applications & Information Security (ICCAIS)**. Piscataway, NJ: IEEE, 2020. p. 1–5. Disponível em: <https://doi.org/10.1109/ICCAIS48893.2020.9096805>. Acesso em: 21 jul. 2026.

FAJARDO, J. et al. LMI methods for extended $\mathcal{H}\infty$ filters for landmark-based mobile robot localization. In: **2021 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)**. Piscataway, NJ: IEEE, 2021. p. 511–517. Disponível em: <https://doi.org/10.1109/AIM46487.2021.9517617>. Acesso em: 21 jul. 2026.

FANG, Y. et al. Adaptive unscented Kalman filter for robot navigation problem (adaptive unscented Kalman filter using incorporating intuitionistic fuzzy logic for concurrent localization and mapping). **IEEE Access**, v. 10, p. 101869–101879, 2022. Disponível em: <https://doi.org/10.1109/ACCESS.2022.3207925>. Acesso em: 21 jul. 2026.

FIALA, M. ARTag, a fiducial marker system using digital techniques. In: **2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)**. Piscataway, NJ: IEEE, 2005. v. 2, p. 590–596. Disponível em: <https://doi.org/10.1109/CVPR.2005.74>. Acesso em: 21 jul. 2026.

FÖRSTNER, W.; WROBEL, B. P. **Photogrammetric Computer Vision: Statistics, Geometry, Orientation and Reconstruction**. Cham: Springer, 2016. Disponível em: <https://doi.org/10.1007/978-3-319-11550-4>. Acesso em: 21 jul. 2026.

FOX, D. Adapting the sample size in particle filters through KLD-sampling. **The International Journal of Robotics Research**, v. 22, n. 12, p. 985–1003, 2003. Disponível em: <https://doi.org/10.1177/0278364903022012001>. Acesso em: 21 jul. 2026.

FOX, D. et al. Particle filters for mobile robot localization. In: DOUCET, A.; FREITAS, N. d.; GORDON, N. (Ed.). **Sequential Monte Carlo Methods in Practice**. New York, NY: Springer, 2001. p. 401–428. Disponível em: <https://doi.org/10.1007/978-1-4757-3437-9_19>. Acesso em: 21 jul. 2026.

GARRIDO-JURADO, S. et al. Automatic generation and detection of highly reliable fiducial markers under occlusion. **Pattern Recognition**, v. 47, n. 6, p. 2280–2292, 2014. Disponível em: <https://doi.org/10.1016/j.patcog.2014.01.005>. Acesso em: 21 jul. 2026.

GRISETTI, G.; STACHNISS, C.; BURGARD, W. Improved techniques for grid mapping with Rao-Blackwellized particle filters. **IEEE Transactions on Robotics**, v. 23, n. 1, p. 34–46, 2007. Disponível em: <https://doi.org/10.1109/TRO.2006.889486>. Acesso em: 21 jul. 2026.

GUSRIAL, M. H. et al. Review of Kalman filter variants for SLAM in mobile robotics with linearization and covariance initialization. **Journal of Mechatronics, Electrical Power, and Vehicular Technology**, v. 16, n. 1, p. 69–83, 2025. Disponível em: <https://doi.org/10.55981/j.mev.2025.1096>. Acesso em: 21 jul. 2026.

HALL, D.; LLINAS, J. An introduction to multisensor data fusion. **Proceedings of the IEEE**, v. 85, n. 1, p. 6–23, 1997. Disponível em: <https://doi.org/10.1109/5.554205>. Acesso em: 21 jul. 2026.

HE, D. et al. Point-LIO: Robust high-bandwidth light detection and ranging inertial odometry. **Advanced Intelligent Systems**, v. 5, n. 7, p. 2200459, 2023. Disponível em: <https://doi.org/10.1002/aisy.202200459>. Acesso em: 21 jul. 2026.

HE, S. et al. An enhanced adaptive Monte Carlo localization for service robots in dynamic and featureless environments. **Journal of Intelligent & Robotic Systems**, v. 108, n. 1, p. 6, 2023. Disponível em: <https://doi.org/10.1007/s10846-023-01858-7>. Acesso em: 21 jul. 2026.

HERCIK, R. et al. Implementation of autonomous mobile robot in SmartFactory. **Applied Sciences**, v. 12, n. 17, p. 8912, 2022. Disponível em: <https://doi.org/10.3390/app12178912>. Acesso em: 21 jul. 2026.

HINDERER, S.; SCHEFFLER, M.; YANG, B. **Investigation of ArUco Marker Placement for Planar Indoor Localization**. 2025. Disponível em: <https://arxiv.org/abs/2509.17345>. Acesso em: 21 jul. 2026.

HOUSEIN, A. A. et al. Extended Kalman filter sensor fusion in practice for mobile robot localization. **International Journal of Advanced Computer Science and Applications**, v. 13, n. 2, p. 33–38, 2022. Disponível em: <https://doi.org/10.14569/IJACSA.2022.0130204>. Acesso em: 21 jul. 2026.

INOUE, R. S.; TERRA, M. H.; CERRI, J. P. Extended robust Kalman filter for attitude estimation. **IET Control Theory & Applications**, v. 10, n. 2, p. 162–172, 2016. Disponível em: <https://doi.org/10.1049/iet-cta.2015.0235>. Acesso em: 21 jul. 2026.

INOUE, R. S.; TERRA, M. H.; JR., V. G. Robust state-space estimation for mobile robot localization. In: **2008 IEEE Latin American Robotic Symposium**. Piscataway,

NJ: IEEE, 2008. p. 85–90. Disponível em: <https://doi.org/10.1109/LARS.2008.31>. Acesso em: 21 jul. 2026.

ISHIHARA, J. Y.; TERRA, M. H.; CERRI, J. P. Optimal robust filtering for systems subject to uncertainties. **Automatica**, v. 52, p. 111–117, 2015. Disponível em: <https://doi.org/10.1016/j.automatica.2014.10.120>. Acesso em: 21 jul. 2026.

JIANG, F. et al. Robust indoor localization with ranging-IMU fusion. In: **2024 IEEE International Conference on Robotics and Automation (ICRA)**. Piscataway, NJ: IEEE, 2024. p. 11963–11969. Disponível em: <https://doi.org/10.1109/ICRA57147.2024.10611274>. Acesso em: 21 jul. 2026.

KALAITZAKIS, M. et al. Experimental comparison of fiducial markers for pose estimation. In: **2020 International Conference on Unmanned Aircraft Systems (ICUAS)**. Piscataway, NJ: IEEE, 2020. p. 781–789. Disponível em: <https://doi.org/10.1109/ICUAS48674.2020.9213977>. Acesso em: 21 jul. 2026.

Kalman, R. E. A new approach to linear filtering and prediction problems. **Journal of Basic Engineering**, v. 82, n. 1, p. 35–45, mar. 1960. Disponível em: <https://doi.org/10.1115/1.3662552>. Acesso em: 21 jul. 2026.

KANG, H. et al. Adaptive measurement model-based fusion of capacitive proximity sensor and LiDAR for improved mobile robot perception. **IEEE Robotics and Automation Letters**, v. 10, n. 1, p. 836–843, 2025. Disponível em: <https://doi.org/10.1109/LRA.2024.3511432>. Acesso em: 21 jul. 2026.

KATO, H.; BILLINGHURST, M. Marker tracking and HMD calibration for a video-based augmented reality conferencing system. In: **Proceedings 2nd IEEE and ACM International Workshop on Augmented Reality (IWAR'99)**. Piscataway, NJ: IEEE, 1999. p. 85–94. Disponível em: <https://doi.org/10.1109/IWAR.1999.803809>. Acesso em: 21 jul. 2026.

KOENIG, N.; HOWARD, A. Design and use paradigms for Gazebo, an open-source multi-robot simulator. In: **2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)**. Piscataway, NJ: IEEE, 2004. v. 3, p. 2149–2154. Disponível em: <https://doi.org/10.1109/IROS.2004.1389727>. Acesso em: 21 jul. 2026.

LABBÉ, M.; MICHAUD, F. Online global loop closure detection for large-scale multi-session graph-based SLAM. In: **2014 IEEE/RSJ International Conference on Intelligent Robots and Systems**. Piscataway, NJ: IEEE, 2014. p. 2661–2666. Disponível em: <https://doi.org/10.1109/IROS.2014.6942926>. Acesso em: 21 jul. 2026.

LABBÉ, M.; MICHAUD, F. RTAB-Map as an open-source LiDAR and visual simultaneous localization and mapping library for large-scale and long-term online operation. **Journal of Field Robotics**, v. 36, n. 2, p. 416–446, 2019. Disponível em: <https://doi.org/10.1002/rob.21831>. Acesso em: 21 jul. 2026.

LI, Y. et al. Multi-sensor fusion localization of indoor mobile robot. In: **2019 IEEE International Conference on Real-time Computing and Robotics (RCAR)**. Piscataway, NJ: IEEE, 2019. p. 481–486. Disponível em: <https://doi.org/10.1109/RCAR47638.2019.9044006>. Acesso em: 21 jul. 2026.

LIANG, Y.; STANCU, A.; ZHANG, Z. Simultaneous localisation and mapping with fiducial markers based on extended Kalman filter. In: **2023 6th International Conference on Intelligent Robotics and Control Engineering (IRCE)**. Piscataway, NJ: IEEE, 2023. p. 154–162. Disponível em: <https://doi.org/10.1109/IRCE59430.2023.10254782>. Acesso em: 21 jul. 2026.

LIM, H.; LEE, Y. S. Real-time single camera SLAM using fiducial markers. In: IEEE. **2009 ICCAS-SICE**. [S.l.], 2009. p. 177–182.

LIN, H.-Y.; YEH, M.-C. Drift-free visual SLAM for mobile robot localization by integrating UWB technology. **IEEE Access**, v. 10, p. 93636–93645, 2022. Disponível em: <https://doi.org/10.1109/ACCESS.2022.3203438>. Acesso em: 21 jul. 2026.

LIN, M.; YANG, C.; LI, D. An improved transformed unscented Kalman filter FastSLAM with adaptive genetic resampling. **IEEE Transactions on Industrial Electronics**, v. 66, n. 5, p. 3583–3594, 2019. Disponível em: <https://doi.org/10.1109/TIE.2018.2854557>. Acesso em: 21 jul. 2026.

LIU, Y. et al. Improved LiDAR localization method for mobile robots based on multi-sensing. **Remote Sensing**, v. 14, n. 23, p. 6133, 2022. Disponível em: <https://doi.org/10.3390/rs14236133>. Acesso em: 21 jul. 2026.

MACENSKI, S. et al. Robot operating system 2: Design, architecture, and uses in the wild. **Science Robotics**, v. 7, n. 66, p. eabm6074, 2022. Disponível em: <https://doi.org/10.1126/scirobotics.abm6074>. Acesso em: 21 jul. 2026.

MACENSKI, S. et al. **The Marathon 2: A Navigation System**. 2020. Disponível em: <https://arxiv.org/abs/2003.00368>. Acesso em: 21 jul. 2026.

MARCOS, L. B.; TERRA, M. H. Markovian filtering for driveshaft torsion estimation in heavy vehicles. **Control Engineering Practice**, v. 102, p. 104552, 2020. Disponível em: <https://doi.org/10.1016/j.conengprac.2020.104552>. Acesso em: 21 jul. 2026.

MARUYAMA, Y.; KATO, S.; AZUMI, T. Exploring the performance of ROS2. In: **Proceedings of the 13th International Conference on Embedded Software**. New York, NY: Association for Computing Machinery, 2016. (EMSOFT 2016), p. 1–10. Disponível em: <https://doi.org/10.1145/2968478.2968502>. Acesso em: 21 jul. 2026.

MUÑOZ-SALINAS, R.; MEDINA-CARNICER, R. UcoSLAM: Simultaneous localization and mapping by fusion of keypoints and squared planar markers. **Pattern Recognition**, v. 101, p. 107193, 2020. Disponível em: <https://doi.org/10.1016/j.patcog.2019.107193>. Acesso em: 21 jul. 2026.

NAM, D. V. et al. Fusion consistency for industrial robot navigation: An integrated SLAM framework with multiple 2d LiDAR-visual-inertial sensors. **Computers and Electrical Engineering**, v. 120, p. 109607, 2024. Disponível em: <https://doi.org/10.1016/j.compeleceng.2024.109607>. Acesso em: 21 jul. 2026.

NAM, T. Q. et al. Development of an autonomous mobile robot system for hospital logistics in quarantine zones. In: NGUYEN, T. D. L. et al. (Ed.). **Intelligent Systems and Networks**. Singapore: Springer Nature Singapore, 2023. p. 271–281.

O'KANE, J. M. **A Gentle Introduction to ROS**. Charleston, SC: CreateSpace Independent Publishing Platform, 2013.

OLSON, E. AprilTag: A robust and flexible visual fiducial system. In: **2011 IEEE International Conference on Robotics and Automation**. Piscataway, NJ: IEEE, 2011. p. 3400–3407. Disponível em: <https://doi.org/10.1109/ICRA.2011.5979561>. Acesso em: 21 jul. 2026.

ORTEGA-CONTRERAS, J. A. et al. Tracking a holonomic mobile robot with systematic odometry errors. In: **2023 International Conference on Control, Artificial Intelligence, Robotics & Optimization (ICCAIRO)**. Piscataway, NJ: IEEE, 2023. p. 99–104. Disponível em: <https://doi.org/10.1109/ICCAIRO58903.2023.00023>. Acesso em: 21 jul. 2026.

OTSU, N. A threshold selection method from gray-level histograms. **IEEE Transactions on Systems, Man, and Cybernetics**, v. 9, n. 1, p. 62–66, 1979. Disponível em: <https://doi.org/10.1109/TSMC.1979.4310076>. Acesso em: 21 jul. 2026.

PEI, F.; ZHU, M.; WU, X. A decorrelated distributed EKF-SLAM system for the autonomous navigation of mobile robots. **Journal of Intelligent & Robotic Systems**, v. 98, n. 3, p. 819–829, 2020. Disponível em: <https://doi.org/10.1007/s10846-019-01069-z>. Acesso em: 21 jul. 2026.

QI, G.; FAN, X.; LI, H. A comparative study of the unscented Kalman filter and particle filter estimation methods for the measurement of the road adhesion coefficient. **Mechanical Sciences**, v. 13, n. 2, p. 735–749, 2022. Disponível em: <https://doi.org/10.5194/ms-13-735-2022>. Acesso em: 21 jul. 2026.

QUIGLEY, M. et al. ROS: an open-source robot operating system. In: **ICRA Workshop on Open Source Software**. [S.l.: s.n.], 2009.

QUIGLEY, M.; GERKEY, B.; SMART, W. **Programming Robots with ROS: A Practical Introduction to the Robot Operating System**. Sebastopol, CA: O'Reilly Media, 2015.

RIMON, M. J. I. et al. Linecourier: A semi-autonomous mobile robot for indoor logistics. In: **2025 IEEE 2nd International Conference on Computing, Applications and Systems (COMPAS)**. Piscataway, NJ: IEEE, 2025. p. 1–6. Disponível em: <https://doi.org/10.1109/COMPAS67506.2025.11381788>. Acesso em: 21 jul. 2026.

ROCHA, K. D.; TERRA, M. H. Robust Kalman filter for systems subject to parametric uncertainties. **Systems & Control Letters**, v. 157, p. 105034, 2021. Disponível em: <https://doi.org/10.1016/j.sysconle.2021.105034>. Acesso em: 21 jul. 2026.

SANCHEZ-LOPEZ, J. L. et al. Visual marker-based multi-sensor fusion state estimation. **IFAC-PapersOnLine**, v. 50, n. 1, p. 16003–16008, jul. 2017. Disponível em: <https://doi.org/10.1016/j.ifacol.2017.08.1911>. Acesso em: 21 jul. 2026.

SAYED, A. H. A framework for state-space estimation with uncertain models. **IEEE Transactions on Automatic Control**, v. 46, n. 7, p. 998–1013, 2001. Disponível em: <https://doi.org/10.1109/9.935054>. Acesso em: 21 jul. 2026.

SAYED, A. H.; NASCIMENTO, V. H. Design criteria for uncertain models with structured and unstructured uncertainties. In: GARULLI, A.; TESI, A. (Ed.). **Robustness in identification and control**. London: Springer London, 1999. p. 159–173.

SHREYA, V. R.; MISHRA, A.; MUKHERJEE, K. Feature estimation and parameter tuning using h∞ filter on FastSLAM framework. In: **2024 6th International Conference on Energy, Power and Environment (ICEPE)**. Piscataway, NJ: IEEE, 2024. p. 1–6. Disponível em: <https://doi.org/10.1109/ICEPE63236.2024.10668921>. Acesso em: 21 jul. 2026.

SUN, Y. A comparative study on the Monte Carlo localization and the odometry localization. In: **2022 IEEE International Conference on Electrical Engineering, Big Data and Algorithms (EEBDA)**. Piscataway, NJ: IEEE, 2022. p. 1074–1077. Disponível em: <https://doi.org/10.1109/EEBDA53927.2022.9744872>. Acesso em: 21 jul. 2026.

SUZUKI, S.; ABE, K. Topological structural analysis of digitized binary images by border following. **Computer Vision, Graphics, and Image Processing**, v. 30, n. 1, p. 32–46, 1985. Disponível em: <https://doi.org/10.1016/0734-189X(85)90016-7>. Acesso em: 21 jul. 2026.

TAO, L. et al. Comparative evaluation of Kalman filters and motion models in vehicular state estimation and path prediction. **Journal of Navigation**, v. 74, n. 5, p. 1142–1160, 2021. Disponível em: <https://doi.org/10.1017/S0373463321000370>. Acesso em: 21 jul. 2026.

TENG, X. et al. Multi-sensor fusion based wheeled robot research on indoor positioning method. **Results in Engineering**, v. 22, p. 102268, 2024. Disponível em: <https://doi.org/10.1016/j.rineng.2024.102268>. Acesso em: 21 jul. 2026.

THRUN, S.; BURGARD, W.; FOX, D. **Probabilistic Robotics**. Cambridge, MA: MIT Press, 2005. (Intelligent Robotics and Autonomous Agents series).

WAN, E.; MERWE, R. V. D. The unscented Kalman filter for nonlinear estimation. In: **Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No.00EX373)**. Piscataway, NJ: IEEE, 2000. p. 153–158. Disponível em: <https://doi.org/10.1109/ASSPCC.2000.882463>. Acesso em: 21 jul. 2026.

WENG, J.; COHEN, P.; HERNIOU, M. Camera calibration with distortion models and accuracy evaluation. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 14, n. 10, p. 965–980, 1992. Disponível em: <https://doi.org/10.1109/34.159901>. Acesso em: 21 jul. 2026.

XU, H.; MANNOR, S. A Kalman filter design based on the performance/robustness tradeoff. **IEEE Transactions on Automatic Control**, v. 54, n. 5, p. 1171–1175, 2009. Disponível em: <https://doi.org/10.1109/TAC.2009.2017816>. Acesso em: 21 jul. 2026.

XU, Y. et al. On globalized robust Kalman filter under model uncertainty. **IEEE Transactions on Automatic Control**, v. 70, n. 2, p. 1147–1160, 2025. Disponível em: <https://doi.org/10.1109/TAC.2024.3451048>. Acesso em: 21 jul. 2026.

YATIGUL, R. et al. Enhancing indoor mobile robot localization through the integration of multi-sensor fusion algorithms. In: **2024 1st International Conference on Robotics, Engineering, Science, and Technology (RESTCON)**. Piscataway, NJ: IEEE, 2024. p. 101–105. Disponível em: <https://doi.org/10.1109/RESTCON60981.2024.10463559>. Acesso em: 21 jul. 2026.

YU, L.; LI, M.; PAN, G. Indoor localization based on fusion of AprilTag and adaptive Monte Carlo. In: **2021 IEEE 5th Information Technology,Networking,Electronic and Automation Control Conference (ITNEC)**. Piscataway, NJ: IEEE, 2021. v. 5, p. 464–468. Disponível em: <https://doi.org/10.1109/ITNEC52019.2021.9587205>. Acesso em: 21 jul. 2026.

ZACHARIAE, A. et al. Human-robot interactions in autonomous hospital transports. **Robotics and Autonomous Systems**, v. 179, p. 104755, 2024. Disponível em: <https://doi.org/10.1016/j.robot.2024.104755>. Acesso em: 21 jul. 2026.

ZHANG, L. et al. A multi-sensor fusion positioning approach for indoor mobile robot using factor graph. **Measurement**, v. 216, p. 112926, 2023. Disponível em: <https://doi.org/10.1016/j.measurement.2023.112926>. Acesso em: 21 jul. 2026.

ZHANG, L. et al. Indoor mobile robot localization applying IMU/stereo camera/LiDAR and graph-based optimization. **IEEE Sensors Journal**, v. 24, n. 13, p. 21466–21478, 2024. Disponível em: <https://doi.org/10.1109/JSEN.2024.3400269>. Acesso em: 21 jul. 2026.

ZHANG, M.; SU, Z. Indoor localization method for mobile robots based on Monte Carlo localization. In: **Proceedings of the 2024 3rd International Symposium on Control Engineering and Robotics**. New York, NY, USA: Association for Computing Machinery, 2024. (ISCER '24), p. 408–413. Disponível em: <https://doi.org/10.1145/3679409.3679485>. Acesso em: 21 jul. 2026.

ZHANG, S.; SHAN, J.; LIU, Y. Approximate inference particle filtering for mobile robot SLAM. **IEEE Transactions on Automation Science and Engineering**, v. 22, p. 7967–7978, 2025. Disponível em: <https://doi.org/10.1109/TASE.2024.3475735>. Acesso em: 21 jul. 2026.

ZHANG, Y.-J. **2D Computer Vision**. Singapore: World Scientific, 2022. Disponível em: <https://doi.org/10.1142/12497>. Acesso em: 21 jul. 2026.

ZHANG, Z. A flexible new technique for camera calibration. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 22, n. 11, p. 1330–1334, 2000. Disponível em: <https://doi.org/10.1109/34.888718>. Acesso em: 21 jul. 2026.

ZHAO, L. et al. Graph-based robust localization of object-level map for mobile robotic navigation. **IEEE Transactions on Industrial Electronics**, v. 71, n. 1, p. 697–707, 2024. Disponível em: <https://doi.org/10.1109/TIE.2023.3245208>. Acesso em: 21 jul. 2026.

ZHENG, S. et al. Simultaneous localization and mapping (SLAM) for autonomous driving: Concept and analysis. **Remote Sensing**, v. 15, n. 4, p. 1156, 2023. Disponível em: <https://doi.org/10.3390/rs15041156>. Acesso em: 21 jul. 2026.

ZHOU, W. et al. Laser vision fusion based on unscented Kalman filtering for pose estimation of indoor mobile robot. In: **2022 IEEE International Conference on Robotics and Biomimetics (ROBIO)**. Piscataway, NJ: IEEE, 2022. p. 741–747. Disponível em: <https://doi.org/10.1109/ROBIO55434.2022.10011754>. Acesso em: 21 jul. 2026.

Apêndices

APÊNDICE A

Calibração da câmera

A.1 Modelo da câmera linear

Considere como Q um objeto no espaço tridimensional com as coordenadas fixas $[x\ y\ z]^T$, e q sendo um ponto nas coordenadas da câmera correspondente a Q, no espaço bidimensional, representado como $[x_c\ y_c\ w]^T$. O objetivo da calibração da câmera é relacionar esses pontos, a fim de calcular os parâmetros internos e externos da câmera. Sendo assim, a conversão de Q, no sistema de coordenadas fixo, para q, no sistema de coordenadas da câmera, pode ser representada pela Equação $q = MQ$, sendo que:

$$q = \begin{bmatrix} {}^c x \\ {}^c y \\ w \end{bmatrix}, Q = \begin{bmatrix} x \\ y \\ z \end{bmatrix}, M = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}, \tag{79}$$

em que $w = z$, considerando que o ponto q está em coordenadas homogêneas e M é a matriz de parâmetros internos da câmera, sendo que f_x e f_y são as distâncias focais da câmera em pixels nos eixos x e y, e c_x e c_y são as coordenadas do ponto principal da imagem. Para cada imagem que a câmera obtém de um determinado objeto, a posição do objeto em relação ao sistema de coordenadas da câmera pode ser expressa em termos de rotação e translação, como na Figura 26.

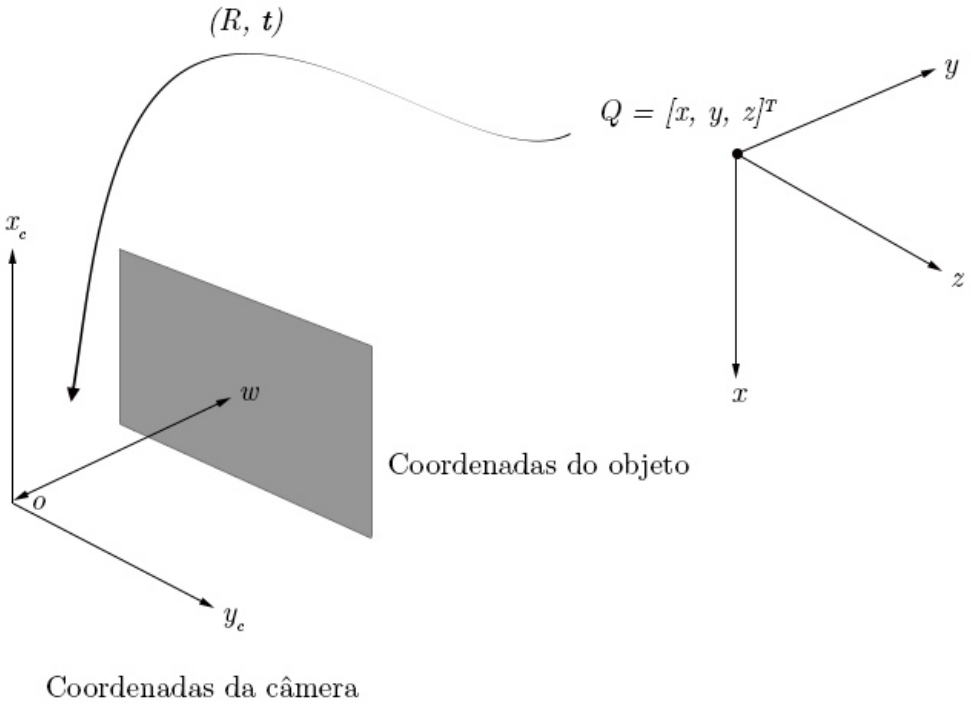

Figura 26 – Conversão do objeto Q em 3D para o ponto q nas coordenadas da câmera. Fonte: adaptado de (BRADSKI; KAEHLER, 2008).

A matriz de rotação R de dimensão 3×3 em que seus elementos internos representam a orientação do sistema de coordenadas da câmera em relação ao sistema de coordenadas global pode ser representada da seguinte forma:

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{24} & r_{25} & r_{26} \end{bmatrix}, \tag{80}$$

sendo r_{ij} os elementos internos da matriz de rotação. Para maiores detalhes veja (FÖRSTNER; WROBEL, 2016) e (CRAIG, 2013). O vetor de translação $\boldsymbol{t}$, de dimensão 3×1, que especifica o deslocamento do sistema de coordenadas da câmera em relação ao sistema de coordenadas global pode ser representado da seguinte forma:

$$\boldsymbol{t} \equiv \begin{bmatrix} t_x & t_y & t_z \end{bmatrix}^T, \tag{81}$$

sendo que os elementos t_x, t_y e t_z representam a translação ao longo dos eixos x, y e z do sistema de coordenadas global. Utilizando os ângulos de Euler, a matriz R pode ser representada como uma função de θ, ϕ, e ψ:

$$R = \begin{bmatrix} \cos\psi\cos\theta & \sin\psi\cos\theta & -\sin\theta \\ -\sin\psi\cos\phi + \cos\psi\sin\theta\sin\phi & \cos\psi\cos\phi + \sin\psi\sin\theta\sin\phi & \cos\theta\sin\phi \\ \sin\psi\sin\phi + \cos\psi\sin\theta\cos\phi & -\cos\psi\sin\phi + \sin\psi\sin\theta\cos\phi & \cos\theta\cos\phi \end{bmatrix}. \tag{82}$$

Dessa forma, a matriz R e o vetor $\boldsymbol{t}$ possuem três graus de liberdade, e os parâmetros externos da câmera são os ângulos θ, ψ e ϕ em R, e t_x, t_y, e t_z em $\boldsymbol{t}$. Segundo (ZHANG, 2022), o sistema de coordenadas fixo pode ser sobreposto pelo sistema de coordenadas da câmera, aplicando uma série de transformações nas coordenadas homogêneas de um ponto Q no espaço, como:

$$A = M \begin{bmatrix} R & \boldsymbol{t} \end{bmatrix} \equiv \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{24} & r_{25} & r_{26} & t_z \end{bmatrix}, \tag{83}$$

em que M é a matriz de transformação da projeção de imagem, representada na Equação (79), R é a matriz de rotação da câmera, e $\boldsymbol{t}$ é o vetor de translação da câmera, representado na Equação (81). A matriz de calibração da câmera, dada por A, incorpora as transformações de projeção, rotação e translação necessárias para mapear as coordenadas de um ponto 3D no espaço para as coordenadas da imagem capturadas pela câmera. Portanto, os elementos $a_{11}, \ldots, a_{33}$ representam componentes da matriz R, os elementos a_{14}, a_{24}, ea_{34} são os componentes do vetor $\boldsymbol{t}$, e o restante a_{ij} incluem parâmetros de escala e deslocamento, combinando a rotação, translação e projeção. O elemento $a_{44} = 1$, de forma a manter a consistência das coordenadas homogêneas. As coordenadas da imagem na forma homogênea são definidas pela Fórmula:

$$C_h = AW_q, \tag{84}$$

sendo que W_q são as coordenadas homogêneas de Q. Pode-se expressar essa relação como:

$$\underbrace{\begin{bmatrix} C_{h1} \\ C_{h2} \\ C_{h3} \\ C_{h4} \end{bmatrix}}_{C_h} = \underbrace{\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}}_{W_q}. \tag{85}$$

As coordenadas do plano de imagem da câmera podem ser expressadas na forma cartesiana como:

$$x = \frac{C_{h1}}{C_{h4}} \; y = \frac{C_{h2}}{C_{h4}} \tag{86}$$

essas equações são derivadas da normalização das coordenadas homogêneas C_{h1} e C_{h2} por C_{h4} para obter as coordenadas cartesianas no plano da imagem. Substituindo essas equações na Equação (85):

$$xC_{h4} \equiv Ch1 = a_{11}X + a_{12}Y + a_{13}Z + a_{14}, \tag{87}$$

sendo que C_{h4} é definido como:

$$C_{h4} = a_{41}X + a_{42}Y + a_{43}Z + a_{44}, \tag{88}$$

ao substituir C_{h4} na Equação (87):

$$x(a_{41}X + a_{42}Y + a_{43}Z + a_{44}) = a_{11}X + a_{12}Y + a_{13}Z + a_{14}, \tag{89}$$

ao realizar essa substituição em yC_{h4}, obtêm-se a seguinte Equação:

$$yC_{h4} \equiv C_{h2} = a_{21}X + a_{22}Y + a_{23}Z + a_{24}, \tag{90}$$

e ao substituir C_{h4}, obtêm-se:

$$y(a_{41}X + a_{42}Y + a_{43}Z + a_{44}) = a_{21}X + a_{22}Y + a_{23}Z + a_{24}. \tag{91}$$

A expansão de C_{h3} é omitida por ser relacionada ao eixo z. Ao substituir C_{h4} nas equações (87) e (90), obtêm-se as seguintes equações:

$$(a_{11} - a_{41}x)X + (a_{12} - a_{42}x)Y + (a_{13} - a_{43}x)Z + (a_{14} - a_{44}x) = 0 \tag{92}$$

$$(a_{21} - a_{41}y)X + (a_{22} - a_{42}y)Y + (a_{23} - a_{43}y)Z + (a_{24} - a_{44}y) = 0. \tag{93}$$

Dessa forma, ao utilizar a câmera para capturar pontos de coordenadas em 3D no mundo, na forma $[x_i\ y_i\ z_i]^T$, é possível obter as coordenadas no plano de imagem correspondentes, na forma $[x_i\ y_i]^T$, em que $i = 1, 2, \ldots, M$, sendo que $M \geq 6$.

A.1.1 Modelo de Câmera Não-Linear

O modelo de câmera pinhole é bastante utilizado para fins de simplificação da abstração matemática do funcionamento da câmera, no contexto de calibração. Entretanto, na prática, dada a necessidade e consequente complexidade das imagens geradas, a relação que define a projeção da câmera deixa de ser linear, e esse modelo se torna menos eficiente para descrever o funcionamento da câmera.

No sistema óptico mais próximo à realidade, as câmeras são equipadas com lentes, pois elas possuem uma capacidade maior de controlar a entrada de luz que entra na câmera, resultando em uma imagem com maior qualidade. Por outro lado, elas causam distorções significativas na imagem, que podem afetar na medição das coordenadas do robô em sistemas que utilizam as câmeras como sensores para obtenção da localização. Dessa forma, se faz necessário o uso de um modelo não-linear que integre as distorções da lente, em comparação ao modelo linear.

A.2 Distorção Radial

A distorção radial causa um deslocamento para dentro ou para fora de um determinado ponto da imagem a partir de sua localização ideal. Em geral, ela é causada por um erro

na curvatura da superfície da lente, e se torna mais visível à distância do eixo óptico ao longo do raio da lente. O modelo matemático dessa distorção pode ser representado da seguinte forma:

$$x' = x(k_1 r^2 + k_2 r^4 + k_3 r^6) \tag{94}$$

$$y' = y(k_1 r^2 + k_2 r^4 + k_3 r^6), \tag{95}$$

em que $[x\ y]^T$ representam a localização original das coordenadas do ponto distorcido no eixos x e y da imagem, e $[x'\ y']^T$ representam as coordenadas do ponto corrigido. k_1, k_2 e k_3 representam os coeficientes de distorção radial, e a distância entre o ponto na imagem e o centro da imagem é dada por r, que pode ser definido por:

$$r = (x_i^2 + y_i^2)^{\frac{1}{2}}. \tag{96}$$

Um deslocamento radial negativo dos pontos da imagem é chamado de deslocamento de barril. Ele faz com que os pontos externos se aglomerem cada vez mais, e a escala diminua. Já um deslocamento radial positivo é chamado de distorção *pincushion*, e faz com que os pontos externos se dispersem e a escala aumente. A Figura 27 ilustra os efeitos da distorção radial.

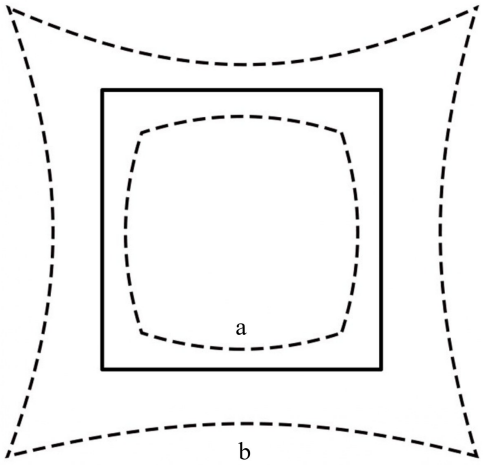

Figura 27 – Efeitos da distorção radial. As linhas sólidas representam a área sem distorção, e as linhas pontilhadas mostram onde possui distorção radial, sendo que a é a distorção negativa e b a positiva. Fonte: Adaptado de (WENG; COHEN; HERNIOU, 1992).

A.3 Distorção Tangencial

A distorção tangencial em geral é causada pelos centros ópticos não colineares nas lentes, de maneira que o ponto de imagem se move tangencialmente ao plano de imagem.

Como consequência, algumas áreas na imagem podem parecer mais próximas do que realmente são. É importante notar que a existe um eixo máximo de distorção em uma determinada direção, e um eixo mínimo na direção perpendicular a ela. A Figura 28 ilustra os efeitos da distorção tangencial na câmera.

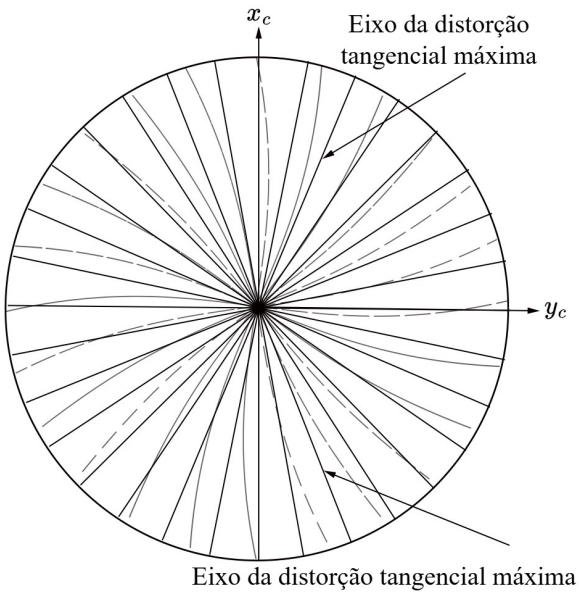

Figura 28 – Efeitos da distorção tangencial. As linhas sólidas representam a área sem distorção, e as linhas pontilhadas mostram onde possui distorção tangencial, u e v são os eixos no sistema de coordenadas da câmera. Fonte: adaptado de (WENG; COHEN; HERNIOU, 1992).

Para representar a distorção tangencial de forma matemática, adicionam-se os parâmetros p_1 e p_2 na distorção radial, resultando na seguinte Fórmula:

$$x_c = x + [2p_1y + p_2(r^2 + 2x^2)] \tag{97}$$

$$y_c = y + [p_1(r^2 + 2y^2) + 2p_2x] \tag{98}$$

Em que p_1 e p_2 representam os coeficientes de distorção da lente, seguindo o que foi visto em (BROWN, 1971). Os parâmetros internos da câmera podem ser definidos como: a distância focal f, as coordenadas do ponto principal c_x e c_y, os coeficientes de distorção radial e tangencial, dados por k_1, k_2 e k_3, e p_1 e p_2, respectivamente. Os parâmetros externos da câmera podem diferir a cada imagem capturada pela câmera, mas os internos não se alteram. Portanto, só é necessário realizar a calibragem dos parâmetros internos apenas uma vez. Dessa forma, existem cinco coeficientes de distorção necessários, que podem ser representados como um único vetor, descrito como:

$$d_C = \begin{bmatrix} k_1 & k_2 & k_3 & p_1 & p_2 & k_3 \end{bmatrix}^T, \tag{99}$$

sendo d_C o vetor de coeficientes de distorção, e quatro parâmetros intrínsecos, que podem ser definidos pelo vetor i_c, representado como:

$$i_c = \begin{bmatrix} f_x & f_y & c_x & c_y \end{bmatrix}. \tag{100}$$

A.4 Calibração Utilizando um Tabuleiro de Xadrez

Como proposto por (ZHANG, 2000), é possível realizar a calibração da câmera utilizando o padrão visual de um tabuleiro de xadrez devido a sua geometria regular. Esse processo é feito através da homografia planar, em que os pontos contidos em uma superfície plana bidimensional são mapeados para a imagem da câmera. Considere $\tilde{q}$ como o vetor aumentado de q, dado por $[x\ y\ 1]^T$, e $\tilde{Q}$ o vetor aumentado de Q, representado como $[x\ y\ z\ 1]^T$. A relação entre o ponto tridimensional Q, e o ponto bidimensional q é dada por:

$$\tilde{q} = sH\tilde{Q}, \tag{101}$$

sendo que s representa um fator de escala arbitrário. As coordenadas do ponto principal são $[c_x\ c_y]^T$, f_x e f_y são os fatores de escala na imagem. É possível definir a matriz W que une a matriz de rotação e o vetor de translação como:

$$W = \begin{bmatrix} R & \boldsymbol{t} \end{bmatrix}. \tag{102}$$

Nesse sentido, ao multiplicar a matriz M, definida na Equação (79), com $W\tilde{Q}$, obtêm-se a seguinte Equação:

$$\tilde{q} = sMW\tilde{Q}. \tag{103}$$

Assumindo que o plano do objeto possui o eixo $Z = 0$, a Equação que representa a homografia H que mapeia os pontos de um objeto plano para a imagem, pode ser descrita por $H = sM[r_1\ r_2\ \boldsymbol{t}]$, sendo que:

$$\tilde{q} = sH\tilde{Q}'. \tag{104}$$

Portanto, a Equação (104) é utilizada para computar a matriz de homografia a partir de múltiplas imagens de um mesmo objeto, a fim de computar as translações e rotações para cada imagem vista, assim como os parâmetros intrínsecos da câmera, com o objetivo de relacionar os pontos de um objeto plano para os pontos no plano de imagem.

Para realizar a calibragem da câmera aplicando os conceitos descritos foi utilizado o pacote *camera_calibration*[1], disponibilizado pelo ROS. Esse pacote facilita a calibração de câmeras monoculares ou estereoscópicas, de maneira integrada com o ROS, para que no final, as informações da câmera possam ser salvas em um arquivo de texto ou computadas

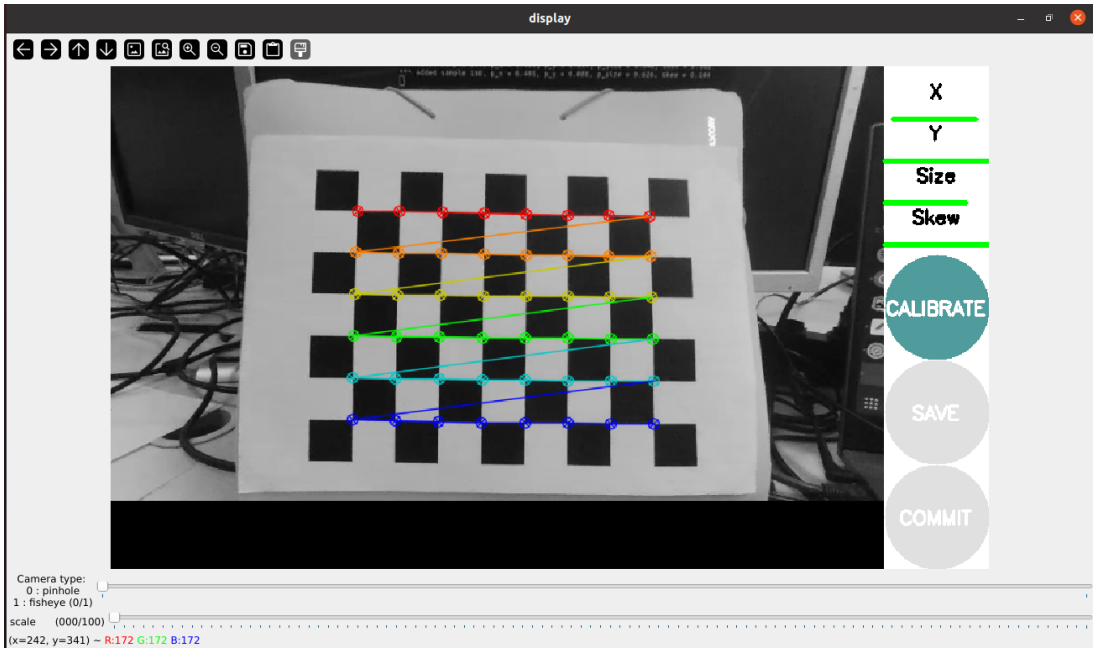

Figura 29 – Janela de calibragem da câmera no ROS. Fonte: Imagem capturada pelo autor.

no tópico *camera_info*, que guarda as informações de calibragem da câmera. A Figura 29 mostra a interface do programa na etapa de calibração:

Na Figura 29, as imagens do tabuleiro são capturadas automaticamente enquanto o usuário movimenta a câmera nos eixos X e Y, e altera os ângulos de inclinação e distância em relação ao tabuleiro, correspondente ao eixo Z. Ao movimentar a câmera em determinadas direções, uma barra na parte inferior de cada eixo se preenche conforme a quantidade suficiente de imagens já foi obtida a partir da câmera. Assim que as barras já forem preenchidas o suficiente, é possível acionar o botão de calibração, que resultará em um documento contendo a matriz de calibração da câmera e os coeficientes de distorção. É possível salvar essas informações no tópico *camera_info* do ROS, que guarda os dados de calibração diretamente nas informações da câmera.

[1] <https://github.com/ros-perception/image_pipeline>

APÊNDICE B

Robot Operating System

O ROS (QUIGLEY et al., 2009) é um *framework* com código aberto de desenvolvimento de software para robôs, amplamente utilizado no mundo todo. Ele possui um grande conjunto de bibliotecas e ferramentas destinados a facilitar e a auxiliar no desenvolvimento de aplicações para robôs, enquanto oferece as funcionalidades de um sistema operacional, como abstração de hardware, controle de dispositivos a baixo nível, troca de mensagens entre processos e gerenciamento de pacotes (O'KANE, 2013), além de se destacar por apresentar um elevado número de componentes e oferecer suporte para robôs de diferentes modelos, além de possuir uma ampla e crescente comunidade de usuários contribuindo para a sua expansão. Por fim, o ROS possui aplicações para vários domínios, incluindo robôs aéreos, terrestres, manipuladores industriais, drones, humanoides, entre outros.

B.1 Estrutura de Funcionamento do ROS

O processo computacional do ROS é comandado pela comunicação entre nós de comunicação, que promovem o isolamento de falhas, modularidade e reutilização de código. Eles utilizam a camada de middleware do ROS para realizar comunicações, e são executados de maneira independente. Dessa forma, um computador pode manter mais de um nó em execução simultaneamente. Eles também se comunicam com sensores e atuadores robóticos, a fim de trocar informações utilizando redes serializadas.

Para que um nó seja executado, é necessário manter em execução uma instância do tipo mestre, denominada *roscore*. Ela se torna responsável por identificar e registrar os nós ativos, coordenar a troca de mensagens e invocar serviços. Desta maneira, em um sistema distribuído, o nó mestre deve ser executado em um computador para que outros nós se

comuniquem de forma remota, a partir de uma comunicação primordial com o *roscore*. Para que os dados sejam armazenados durante a comunicação, utiliza-se o servidor de parâmetros, que é disponibilizado pelo nó mestre, sendo que todos os outros conseguem acessar e modificar os valores armazenados.

Os nós se comunicam através da troca de mensagens, que são estruturas de dados simples contendo um campo de informações contendo dados de diversos tipos, como inteiro, ponto flutuante, booleano, entre outros. As mensagens são interpretadas utilizando uma chave associativa, e são recebidas como um fluxo de bits. Sendo assim, é necessário consultar a chave que descreve o tipo da mensagem para saber como realizar a conversão de bits e reconstruir a estrutura de dados correspondente. Quando um nó publica uma mensagem, ela se torna disponível para qualquer outro nó na rede. Para que ele possa receber uma mensagem, ele se inscreve à mensagem publicada. Ao publicar uma mensagem, um nó é denominado publicador (do inglês, publisher), e ao se inscrever a uma mensagem, é chamado de subscritor (do inglês, subscriber), mas eles podem realizar ambas as funções ao mesmo tempo.

O transporte das mensagens é feito utilizando os tópicos, que permitem que vários nós publiquem e se inscrevam em determinados tópicos. Dessa forma, um subscritor não tem conhecimento de qual nó publica aquele tópico, apenas sabe o nome do tópico. Nesse sentido, para realizar uma comunicação, um nó publicador inicia um tópico ao anunciar ao *roscore* a existência do tópico e sua mensagem correspondente. Para que isso seja feito, o publicador instancia um objeto da classe *ros::publisher*, que é parte da distribuição do ROS. Utilizar os objetos dessa classe evita que o programador escreva códigos para realizar a comunicação entre os nós. A frequência de publicação é configurável de acordo com a necessidade do usuário, mas se torna limitada à capacidade do sistema.

Além disso, o ROS permite que a comunicação interna seja feita utilizando a estrutura de mensagens de requisição e resposta (do inglês, request/response), através dos serviços. Eles são utilizados de acordo com a necessidade do projeto, mas são mais comumente usados em sistemas distribuídos. Desta maneira, os serviços invocam procedimentos remotos de maneira síncrona, permitindo que um nó invoque uma função que é executada em outro nodo. As entradas e saídas são determinadas previamente, e o servidor responsável pelo serviço especifica um *callback* para responder à requisição, e o informa. Em seguida, o cliente acessa o serviço através de um proxy local. Para que a requisição e a resposta sejam feitas, o tipo de dados da requisição e da resposta devem ser declarados em um arquivo dentro da estrutura de pastas do ROS. Um nó atua como um servidor no qual o nó cliente pode requisitar o serviço, e se o servidor completar a rotina do serviço, ele enviará os resultados para o cliente.

Os serviços são úteis para comunicações que precisam ser feitas ocasionalmente e necessitam de respostas síncronas, as respostas do *callback* de um serviço são rápidas, mas para os casos que necessitam de respostas assíncronas e situações que demandam

mais tempo, é possível utilizar as ações, que são assíncronas e utilizados para situações a longo prazo. Para tanto, as ações definem um objetivo para iniciar um processo, e enviam um resultado quando ele está completo. Enquanto isso ocorre, as ações utilizam um sistema de *feedback* para enviar atualizações sobre o progresso do processo em execução em relação ao objetivo, com a possibilidade de que ele seja cancelado, se for solicitado. As ações são implementadas utilizando os tópicos, e podem ser definidas como um protocolos de alto nível que especificam como um conjunto de tópicos deve ser utilizado, através da combinação de comandos.

Por último, as informações passadas através de mensagens podem ser armazenadas em um arquivo *bag*, que permite que esses dados sejam processados, analisados e visualizados de maneira *offline*. Os *bags* são comumente criados pela ferramenta *rosbag*, que se inscreve a um ou mais tópicos do ROS, e armazena os dados serializados das mensagens no arquivo. Em suma, os principais componentes que formam a estrutura de comunicação do ROS podem ser representados pelo seguinte diagrama:

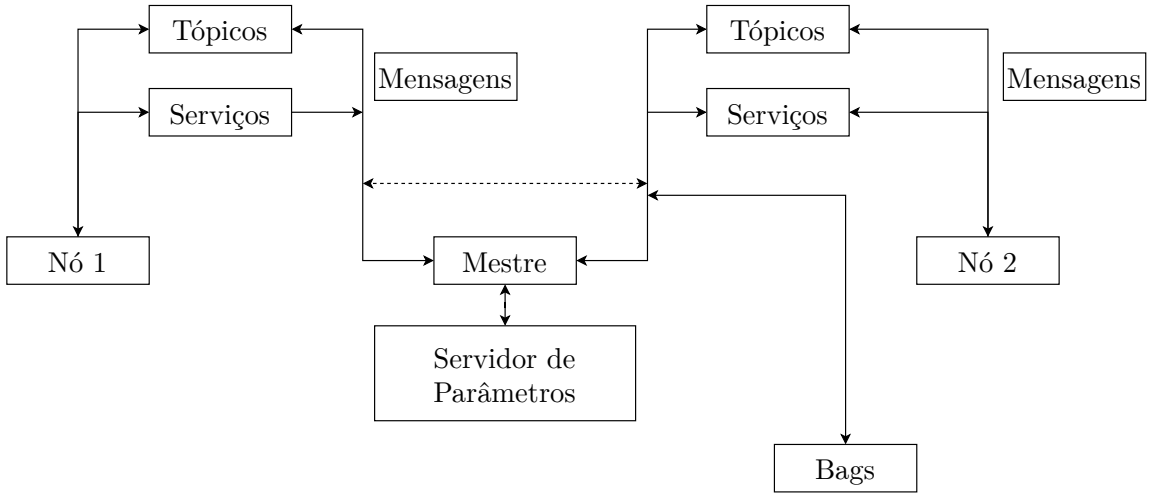

Figura 30 – Diagrama de blocos representando os componentes do ROS. As setas representam o sentido de comunicação entre eles. Fonte: adaptado de (QUIGLEY; GERKEY; SMART, 2015).

B.2 ROS 2

A primeira distribuição do ROS 2 a ser lançada foi o *Ardent Apalone*, em 2017. Desde então, o ROS 2 vêm estabelecendo cada vez mais sua base no desenvolvimento de software para robôs, visto que trouxe diversas melhorias em relação à primeira versão. O ROS 2 trouxe mudanças significativas na arquitetura do sistema. Enquanto o ROS 1 utiliza um nó mestre, o ROS 2 utiliza o *Data Distribution Service* (DDS), que promete eficiência, confiança, baixa latência e escalabilidade. Com a ausência do *roscore*, o sistema deixa de

ser centralizado, o que permite que os nós encontrem outros de maneira independente. Além disso, a nova versão do ROS oferece um suporte mais abrangente para diversas linguagens de programação, como Java e *C#*, e para outras plataformas além do Linux, como o Windows e o MacOS, além de sistemas multi-robôs.

Em contraste com o ROS 1, no ROS 2 foram feitas alterações na execução multi-threads e no processamento em tempo real. Na nova versão, o ROS suporta múltiplos nós em execução paralela, melhorando o desempenho e o aproveitamento de processadores *multi-core* de maneira mais eficiente. A nova versão também oferece um desempenho melhor em sistemas que exigem comunicação em baixa latência e em redes determinísticas.

No ROS 1 os serviços não são executados de forma assíncrona e não possuem um mecanismo de *feedback* ou cancelamento, apenas as ações executam essa função em conjunto com os tópicos. No ROS 2 as ações ainda utilizam os tópicos, mas agora incluem serviços assíncronos para definir um objetivo, cancelar, ou solicitar um resultado, de maneira independente aos tópicos. Para melhorar o entendimento, a Figura 31 ilustra a arquitetura do ROS 1 em comparação com o ROS 2.

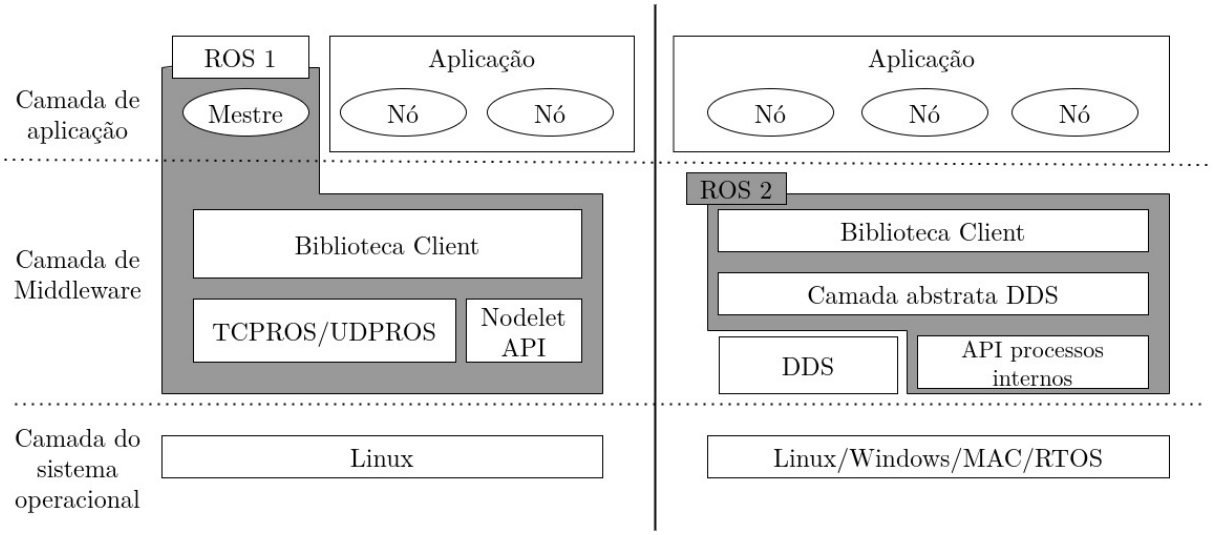

Figura 31 – Estrutura básica de funcionamento do ROS. Fonte: (MARUYAMA; KATO; AZUMI, 2016).