Universidade Federal de Minas Gerais (UFMG)
Av. Antônio Carlos 6627, CEP 31270-901,
Belo Horizonte, MG Brasil

DISSERTAÇÃO DE MESTRADO Nº 1232

Localização e Mapeamento para Robôs Móveis em Ambientes Confinados Baseado em Fusão de LiDAR com Odometrias de Rodas e Sensor Inercial

Gilmar Pereira da Cruz Júnior

Belo Horizonte - Minas Gerais
28 de maio de 2021

Gilmar Pereira da Cruz Júnior

Localização e Mapeamento para Robôs Móveis em Ambientes Confinados Baseado em Fusão de LiDAR com Odometrias de Rodas e Sensor Inercial

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Minas Gerais como requisito parcial para a obtenção do Título de Mestre em Engenharia Elétrica.

Universidade Federal de Minas Gerais - UFMG

Programa de Pós Graduação em Engenharia Elétrica - PPGEE

Orientador: Prof. Dr. Gustavo Medeiros Freitas

Belo Horizonte - Minas Gerais

Maio, 2021

Cruz Júnior, Gilmar Pereira da.
C957l Localização e mapeamento para robôs móveis em ambientes confinados baseado em fusão de LiDAR com odometrias de rodas e sensor inercial [recurso eletrônico] / Gilmar Pereira da Cruz Júnior. - 2021.
1 recurso online (131 f. : il., color.) : pdf.

Orientador: Gustavo Medeiros Freitas.

Dissertação (mestrado) - Universidade Federal de Minas Gerais, Escola de Engenharia.

Bibliografia: f. 124-131.
Exigências do sistema: Adobe Acrobat Reader.

1. Engenharia elétrica - Teses. 2. Kalman, Filtragem de - Teses. 3. Robôs móveis - Teses I. Freitas, Gustavo Medeiros. II. Universidade Federal de Minas Gerais. Escola de Engenharia. III. Título.

CDU: 621.3(043)

Ficha catalográfica elaborada pelo bibliotecário Reginaldo Cesar Vital dos Santos CRB/6 2165
Biblioteca Prof. Mário Werneck, Escola de Engenharia da UFMG

"Localização e Mapeamento para Robôs Móveis em Ambientes Confinados Baseado em Fusão de LiDAR com Odometria de Rodas e Sensor Inercial"

Gilmar Pereira da Cruz Júnior

Dissertação de Mestrado submetida à Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação em Engenharia Elétrica da Escola de Engenharia da Universidade Federal de Minas Gerais, como requisito para obtenção do grau de Mestre em Engenharia Elétrica.

Aprovada em 28 de maio de 2021.

Por:

Prof. Dr. Gustavo Medeiros Freitas
DEE (UFMG) - Orientador

Prof. Dr. Armando Alves Neto
DELT (UFMG)

Prof. Dr. Gustavo Pessin
Controle e Robótica (ITV - Instituto Tecnológico Vale)

Dedico este trabalho a meu pai Gilmar Pereira da Cruz, que me inspirou a sonhar alto e a nunca desistir!

Agradecimentos

Primeiramente agradeço a Deus por me sustentar e me guiar na vida, me permitindo alcançar objetivos, que sem ele não seria possível.

Gostaria de agradecer a Universidade Federal de Minas Gerais e ao Instituto Tecnológico Vale pelos recursos oferecidos para pesquisa realizada nesta dissertação, por meio do projeto "Dispositivo Robótico de Inspeção de Ambientes Restritos e Confinados". Agradeço também as equipes de pesquisadores e colegas, em especial ao Gustavo Pessin e ao Hector Azpúrua do ITV, e ao meu orientador Gustavo Medeiros Freitas pela sua dedicação na coordenação do projeto na UFMG, pelos seus ensinamentos e por me conceder essa oportunidade única.

Gostaria de agradecer à CNPq e a Fundação Cristiano Ottoni pelo apoio financeiro concedido por meio de bolsas de mestrado, possibilitando a minha dedicação aos estudos.

Gostaria de agradecer a minha família, em particular a minha mãe Imaculada Cruz pelo carinho incondicional.

E especialmente, eu agradeço imensamente à minha noiva Mariceli Pinheiro pelo companheirismo e amor, e pelo seu apoio durante minha graduação e mestrado, me erguendo nos momentos mais difíceis e me motivando cada vez mais a cada etapa vencida. Obrigado por estarmos juntos.

"Hoje, ainda almejamos saber por que estamos aqui e de onde viemos. O desejo profundo da humanidade pelo conhecimento é justificativa suficiente para nossa busca contínua."
(Stephen Hawking).

Resumo

A exploração e inspeção de espaços confinados utilizando dispositivos robóticos é uma opção viável e segura. Contudo, ambientes confinados apresentam grandes desafios à robótica, principalmente em relação à estimação da pose e geração de mapas, essenciais para navegação autônoma. A ausência de sinal de GPS, bem como possíveis interferências em magnetômetros, principalmente na região de mineração ou na presença de equipamentos elétricos de grandes potências, impossibilitam a utilização destes sensores para estimar posição e orientação do dispositivo robótico. Além disso, a baixa iluminação pode prejudicar a identificação de *features* visuais, e solos escorregadios acarretam acúmulo de erros no cálculo de odometria das rodas. Deste modo, uma solução mais adequada é a utilização de técnicas de localização e mapeamento simultâneos baseadas em sensores LiDAR, denominado LiDAR SLAM. Para o funcionamento adequado de um robô autônomo, a localização precisa ser estimada *online* de forma a realimentar o sistema de controle de navegação, e o mapa deve ser representativo e leve do ponto de vista computacional, facilitando a execução de algoritmos de planejamento de caminhos embarcados. Neste contexto, essa dissertação apresenta um estudo e investigação de 3 técnicas consideradas como estado da arte, LOAM-Velodyne, LeGO-LOAM e HDL-Graph-SLAM, de modo a determinar qual delas é a mais adequada para a implementação no EspeleoRobô, um dispositivo robótico de inspeção de ambientes confinados. Este robô móvel está sendo desenvolvido pelo Instituto Tecnológico Vale em parceria com a Universidade Federal de Minas Gerais. A comparação entre as técnicas de LiDAR SLAM utiliza métricas propostas para avaliar a precisão das estimações de localização e dos mapas gerados durante simulações em ambientes virtuais implementados com o software CoppeliaSim, e também em experimentos reais. A técnica com melhor performance, LeGO-LOAM, foi adaptada e embarcada no EspeleoRobô, possibilitando avaliar o desempenho da localização e mapeamento *online* durante testes *indoor* e *outdoor* na UFMG, e em experimentos de campo na Mina du Veloso. Dado que alguns dos ambientes explorados pelo EspeleoRobô, incluindo dutos e galerias, apresentam poucas *features* geométricas que impactam no desempenho das técnicas de LiDAR SLAM, gerando subestimação da distância percorrida pelo robô e consequentemente a deformação do mapa gerado, é proposto também um Filtro de Kalman Estendido (EKF - do inglês *Extended Kalman Filter*) aplicado ao SLAM. Esse filtro funde os dados da odometria LiDAR, acrescida de uma covariância adaptativa em função do número de *features* identificadas do ambiente, com a odometrias das rodas e os dados da IMU disponível no robô. A implementação foi analisada inicialmente em ambientes simulados, e em seguida validada em experimentos reais com o EspeleoRobô.

Palavras-chave: LiDAR SLAM, Filtro de Kalman Estendido, Robôs Móveis, Robôs de Serviço, Inspeção de Ambientes Confinados.

Abstract

The exploration and inspection of confined environments using robotic devices is a feasible and safe option. However, confined environments present several challenges to robotics, especially regarding pose estimation and map generation which are essential for autonomous navigation. Due to the lack of GPS signal and possible interference with magnetometers, especially in the mining region or in the presence of high-powered electrical equipment, it is impossible to use these sensors to estimate the position and orientation of the robotic device. Low lighting can impair the identification of visual features, and slippery terrains lead to the accumulation of wheel odometry errors. Thus, a more suitable solution is the Simultaneous Localization And Mapping based on LiDAR sensors, called LiDAR SLAM. For the proper functioning of an autonomous robot, the location needs to be estimated online in order to feedback the navigation control system, and the map must be representative and computationally lighweight, facilitating the execution of the path planning embedded algorithms. Therefore, this dissertation presents a study and investigation of 3 state-of-the-art techniques, LOAM-Velodyne, LeGO-LOAM, and HDL-Graph-SLAM, to determine which one is the most suitable for implementation in the EspeleoRobô, a robotic device for inspecting confined environments. This mobile robot is being developed by the Instituto Tecnológico Vale (ITV) in partnership with the Universidade Federal de Minas Gerais (UFMG). The comparison between LiDAR SLAM techniques uses metrics proposed to assess the accuracy of localization estimations and maps generated during simulations in virtual environments implemented with the CoppeliaSim software, and also in real-world experiments. The best performing technique, LeGO-LOAM, was adapted and embedded into the EspeleoRobô, allowing to evaluate the online performance of localization and mapping during indoor and outdoor tests at UFMG and in field experiments at Mina du Veloso. Given that some of the environments explored by EspeleoRobô, including ducts and galleries, present few geometric features that impact the performance of LiDAR SLAM techniques, generating an underestimation of the robot traveled distance and consequently the deformation of the generated map, we propose an Extended Kalman Filter (EKF) integrated to the SLAM technique. This filter merges data from the LiDAR odometry, which uses an adaptive covariance in respect to the number of environment features identify, the wheel odometry and the IMU available in the robot. The implementation was initially evaluated with simulations in virtual environments, and then validated during real-world experiments with the EspeleoRobô.

Keywords: *LiDAR SLAM, Extended Kalman Filter, Mobile Robots, Service Robots, Confined Environments Inspection.*

Lista de Figuras

Lista de Tabelas

Lista de Algoritmos

Lista de Abreviaturas e Siglas

CASE	*Conference on Automation Science and Engineering*
CML	*Current Mapping and Localization*
CTER	*Counter Tunnel Exploitation Robot*
CIR	Centro Instantâneo de Rotação
DARPA	*Defense Advanced Research Projects Agency*
DCS	*Dynamic Covariance Scaling*
EKF	*Extended Kalman Filter*
EUA	Estados Unidos da América
GPS	*Global Positioning System*
GR-LOAM	*Ground Robots - LiDAR Odometry And Mapping*
HT	*Hough Transform*
ICP	*Iterative Closest Point*
ICRA	*International Conference on Robotics and Automation*
IMLS-SLAM	*Implicit Moving Least Squares*-SLAM
IMU	*Inertial Measurenment Unit*
INS	*Inertial Navigation System*
IP67	*Ingress Protection* 67
ITV	Instituto Tecnológico Vale
KTIO	*Keyframe-based Thermal–Tnertial Odometry*
LADAR	*Laser Radar*
LeGO-LOAM	*Lightweight and Ground-Optimized LiDAR Odometry And Mapping*

LiDAR	*Light Detection And Ranging*
LIMO	*LiDAR-Monocular Visual Odometry*
LM	*Levemberg-Marquardt*
LOAM	*LiDAR Odometry And Mapping*
NDT	*Normal Distributions Transform*
NR-33	Norma Regulamentadora 33
OL	Odometria LiDAR
OV	Odometria Visual
PFH	*Point Features Histograms*
RBPF	*Rao-Blackwellized Particle Filter*
RDS	*Robotics Developer Studio*
RGB	*Red, Green, and Blue*
ROS	*Robot Operating System*
ROVIO	*Robust Visual-Inertial Odometry*
RWHT	*Range Weighted Hough Transform*
SLAM	*Simultaneous Localization And Mapping*
ToF	*Time of Flight*
UFMG	Universidade Federal de Minas Gerais
UKF	*Unscented Kalman Filter*
UWB	*Ultra-Wide Bandwidth*
V-LOAM	*Visual-LiDAR Odometry And Mapping*

Lista de Símbolos

A	Sistema de coordenadas A
$\mathbf{A}_k$	Matriz que mapeia a transição de estados de um sistema linear
$\mathbf{b}$	Vetor que expressa a equação de uma linha
$\mathbf{B}_k$	Matriz que mapeia as entradas de controle de um sistema linear
c	Velocidade média da luz
$\mathbf{c}$	Vetor unitário que dá a direção de uma linha
$\mathbf{c}_{k-1}^{(i)}$	Vetor unitário que dá a direção de uma linha i no instante $k-1$
$\mathbf{c}_{k}^{(i)}$	Vetor unitário que dá a direção de uma linha i no instante k
$\mathbf{C}_k$	Matriz que mapeia os estados de um sistema linear para a saída
d	Distância de objeto detectado por um sensor *laser*
d_1	Distância maior de um ponto detectado do ambiente utilizado para o cálculo do ângulo de segmentação
d_2	Distância menor de um ponto detectado do ambiente utilizado para o cálculo do ângulo de segmentação
d_b	Distância entre pontos de bordas correspondentes entre duas varreduras do sensor LiDAR
d_p	Distância entre pontos de superfícies planas correspondentes entre duas varreduras do sensor LiDAR
$det(\cdot)$	Determinante de uma matriz
$d(i,j)$	Distância de um ponto expresso nas coordenadas de um *pixel* i e j
$\mathbf{D}$	Matriz formada pelos vetores dados pela diferença entre pontos de uma nuvem de pontos e sua respectiva média no instante de tempo k

$e(\cdot)$	Função de custo para cálculo de uma pose relativa
$e_k(\cdot)$	Função de custo para cálculo de uma pose relativa no instante de tempo k
$e_{k-1}(\cdot)$	Função de custo para cálculo de uma pose relativa no instante de tempo $k-1$
$\exp\{\cdot\}$	Exponencial natural
$\mathbf{E}$	Matriz diagonal retangular
e_{min}	Limiar mínimo para a função de custo $e(\cdot)$
$f(\cdot)$	Função f – avaliar o contexto
$f_a(\cdot)$	Função de transição de estados para o EKF SLAM
$f_a^{(i)}(\cdot)$	Função i-ésima de transição de estados para o EKF SLAM
f_b	Função não-linear das restrições impostas por pontos de bordas correspondentes
f_p	Função não-linear das restrições impostas pelos pontos de superfícies planas correspondentes
$\mathcal{F}_b$	Sub-conjunto de pontos de bordas
$\mathcal{F}_{b,k}$	Sub-conjunto de pontos de bordas no instante de tempo k
$\mathbb{F}_b$	Conjunto de pontos de bordas
$\mathbb{F}_{b,k-1}$	Conjunto de pontos de bordas no instante de tempo $k-1$
$\mathbb{F}_{b,k-1}^{\overline{\mathrm{O}}}$	Conjunto de pontos de bordas no instante de tempo $k-1$ com respeito ao sistema inercial $\overline{\mathrm{O}}$
$\mathbb{F}_{b,1}^{\overline{\mathrm{O}}}$	Conjunto de pontos de bordas no instante de tempo 1 com respeito ao sistema inercial $\overline{\mathrm{O}}$
$\mathbb{F}_{b,2}^{\overline{\mathrm{O}}}$	Conjunto de pontos de bordas no instante de tempo 2 com respeito ao sistema inercial $\overline{\mathrm{O}}$
$\mathcal{F}_p$	Sub-conjunto de pontos de superfícies planas
$\mathcal{F}_{p,k}$	Sub-conjunto de pontos de superfícies planas no instante de tempo k
$\mathbb{F}_p$	Conjunto de pontos de superfícies planas

$\mathbb{F}_{p,k-1}$	Conjunto de pontos de superfícies planas no instante de tempo $k-1$
$\mathbb{F}_{p,k-1}^{\overline{\mathrm{O}}}$	Conjunto de pontos de superfícies planas no instante de tempo $k-1$ com respeito ao sistema inercial $\overline{\mathrm{O}}$
$\mathbb{F}_{p,1}^{\overline{\mathrm{O}}}$	Conjunto de pontos de superfícies planas no instante de tempo 1 com respeito ao sistema inercial $\overline{\mathrm{O}}$
$\mathbb{F}_{p,2}^{\overline{\mathrm{O}}}$	Conjunto de pontos de superfícies planas no instante de tempo 2 com respeito ao sistema inercial $\overline{\mathrm{O}}$
$\mathbf{F}_k$	Matriz jacobiana de uma função não-linear de transição de estados
$g(\cdot)$	Função g – avaliar o contexto
g_b	Função heurística para adaptar a matriz de covariâncias da odometria LiDAR segundo o número de *features* de bordas
g_p	Função heurística para adaptar a matriz de covariâncias da odometria LiDAR segundo o número de *features* de superfícies planas
$g_u(\cdot)$	Função do modelo de medição da IMU
$g_{\mathrm{R}}(\cdot)$	Função do modelo de medição da odometria do robô
$g_{\mathrm{L}}(\cdot)$	Função do modelo de medição da odometria LiDAR
g_x	Ganho relativo à variância do deslocamento em x
g_y	Ganho relativo à variância do deslocamento em y
g_z	Ganho relativo à variância do deslocamento em z
g_ϕ	Ganho relativo à variância do deslocamento angular ϕ
g_θ	Ganho relativo à variância do deslocamento angular θ
g_ψ	Ganho relativo à variância do deslocamento angular ψ
$\mathbf{G}_k$	Matriz jacobiana de uma função não-linear que mapeia os estados para a saída
h	Quantidade de feixes horizontais de um sensor LiDAR
$\mathbf{H}$	Matriz de transformação homogênea $\in SE(3)$
$\mathbf{H}_{\mathrm{A}}^{\mathrm{O}}$	Matriz de transformação homogênea do sistema de coordenadas A para o sistema O

$\mathbf{H}_{\overline{\mathrm{O}}}^{\mathrm{O}}$	Matriz de transformação homogênea do sistema de coordenadas inercial $\overline{\mathrm{O}}$ para o sistema de coordenadas inercial O
$\mathbf{H}_{\mathrm{I}}^{\overline{\mathrm{O}}}$	Matriz de transformação homogênea da pose integrada I para o sistema de coordenadas inercial $\overline{\mathrm{O}}$
$\mathbf{H}_{\mathrm{R}}^{\mathrm{I}}$	Matriz de transformação homogênea do sistema de coordenadas do robô R para a pose integrada I
$\mathbf{H}_{\mathrm{R}}^{\mathrm{O}}$	Matriz de transformação homogênea do sistema de coordenadas do robô R para o sistema de coordenadas inercial O
$\mathbf{H}_{\mathrm{L}}^{\overline{\mathrm{O}}}$	Matriz de transformação homogênea referente à pose da odometria LiDAR L para o sistema inercial $\overline{\mathrm{O}}$
$\mathbf{H}_{\mathrm{M}}^{\overline{\mathrm{O}}}$	Matriz de transformação homogênea referente à pose do mapeamento LiDAR M para o sistema inercial $\overline{\mathrm{O}}$
$\mathbf{H}_{\mathrm{L}}^{\mathrm{D}}$	Matriz de transformação homogênea utilizada para ajuste da nuvem de pontos
$\mathbf{H}_{\mathrm{I}}$	Matriz de transformação homogênea referente à pose integrada I
$\mathbf{H}_{\mathrm{I}}^{\overline{\mathrm{O}}}$	Matriz de transformação homogênea referente à pose integrada I para o sistema inercial $\overline{\mathrm{O}}$
$\mathrm{i_m}$	Índice da linha de uma matriz de segmentação
$\mathbf{I}$	Matriz identidade
$\mathrm{j_m}$	Índice da coluna de matriz de segmentação
k	Instante de tempo discreto
$\mathbf{K}_k$	Matriz de ganhos para um filtro de Kalman no instante k
l_{max}	Distância máxima para um deslocamento de referência
m	Quantidades de pontos detectados por um sensor LiDAR
$\mathcal{M}_1$	Métrica 1
$\mathcal{M}_2$	Métrica 2
$\mathcal{M}_3$	Métrica 3
$\mathcal{M}_4$	Métrica 4
$\mathcal{M}_5$	Métrica 5

$\mathbf{M}$	Mapa
$\mathbf{M}_{k+1}$	Mapa no instante de tempo $k+1$
$\mathbf{M}_{k}$	Mapa no instante de tempo k
$\mathbf{M}_{\mathrm{L}}^{\overline{\mathrm{O}}}$	Mapa de pontos do LiDAR SLAM com respeito ao sistema de coordenadas inercial $\overline{\mathrm{O}}$
$\mathbf{M}_{\mathrm{L},k-1}^{\overline{\mathrm{O}}}$	Mapa de pontos do LiDAR SLAM no instante de tempo $k-1$ com respeito ao sistema de coordenadas inercial $\overline{\mathrm{O}}$
$\mathbf{M}_{\mathrm{L},k}^{\overline{\mathrm{O}}}$	Mapa de pontos do LiDAR SLAM no instante de tempo k com respeito ao sistema de coordenadas inercial $\overline{\mathrm{O}}$
$\mathbf{M}_{\mathrm{L},k}^{\mathrm{O}}$	Mapa de pontos do LiDAR SLAM no instante de tempo k com respeito ao sistema de coordenadas inercial O
$\mathbf{n}$	Vetor normal
$\mathbf{n}_{\mathrm{L},k}^{(i)}$	Vetor normal do plano i extraído de uma nuvem de pontos no instante de tempo k
$\mathbf{n}_{\mathrm{L},k-1}^{(i)}$	Vetor normal do plano i extraído de uma nuvem de pontos no instante de tempo $k-1$
n	Variável de iteração - avaliar contexto
n_b	Número de *features* de bordas identificadas do ambiente
n_p	Número de *features* de superfícies planas identificadas do ambiente
$\mathtt{n}_b$	Número máximo de *features* de bordas
$\mathtt{n}_p$	Número máximo de *features* de superfícies planas
N	Número total de poses do robô
N_n	Número de pares de planos extraídos de duas varreduras do sensor LiDAR
N_l	Número de pares de linhas extraídos de duas varreduras do sensor LiDAR
$\mathcal{N}(\cdot)$	Distribuição probabilística normal
$\mathbf{N}$	Matriz positiva definida de ganhos para uma função de custo
O	Sistema de coordenada inercial

$\overline{\mathrm{O}}$	Sistema de coordenada inercial do LiDAR SLAM
$\mathbb{O}_{1\times3}$	Vetor de zeros de dimensão 1×3
$\mathbb{P}(\cdot)$	Função de probabilidade
$\mathbf{p}_{\mathrm{CIR}}$	Coordenada do centro instantâneo de rotação
$\mathbf{p}_{3\times1}$	Vetor de posição de dimensão 3×1
$\mathbf{p}_{\mathrm{A}}$	Vetor de posição com respeito ao sistema de coordenadas A
$p_{\mathrm{A},x}$	Componente x do vetor $\mathbf{p}_{\mathrm{A}}$
$p_{\mathrm{A},y}$	Componente y do vetor $\mathbf{p}_{\mathrm{A}}$
$p_{\mathrm{A},z}$	Componente z do vetor $\mathbf{p}_{\mathrm{A}}$
$\mathbf{p}_{\mathrm{A}}^{\mathrm{O}}$	Vetor de posição do sistema de coordenadas A com respeito ao sistema inercial O
$p_{\mathrm{A},x}^{\mathrm{O}}$	Componente x do vetor $\mathbf{p}_{\mathrm{A}}^{\mathrm{O}}$
$p_{\mathrm{A},y}^{\mathrm{O}}$	Componente y do vetor $\mathbf{p}_{\mathrm{A}}^{\mathrm{O}}$
$p_{\mathrm{A},z}^{\mathrm{O}}$	Componente z do vetor $\mathbf{p}_{\mathrm{A}}^{\mathrm{O}}$
$\mathbf{p}_{\mathrm{B}}^{\mathrm{O}}$	Vetor da posição do sistema de coordenadas B com respeito ao sistema inercial O
$\overline{\mathbf{p}}_{\mathrm{B}}^{\mathrm{O}}$	Vetor da posição 4×1 do sistema de coordenadas B com respeito ao sistema inercial O
$\mathbf{p}_{\mathrm{B}}^{\mathrm{A}}$	Vetor da posição do sistema de coordenadas B com respeito ao sistema de coordenadas A
$\overline{\mathbf{p}}_{\mathrm{B}}^{\mathrm{A}}$	Vetor da posição 4×1 do sistema de coordenadas B com respeito ao sistema de coordenadas A
$\mathbf{p}_{\mathrm{R}}$	Vetor da posição do robô
$\mathbf{p}_{\mathrm{R}}^{\mathrm{O}}$	Vetor da posição do robô com respeito ao sistema inercial O
$p_{\mathrm{R},x}^{\mathrm{O}}$	Componente x do vetor $\mathbf{p}_{\mathrm{R}}^{\mathrm{O}}$
$p_{\mathrm{R},y}^{\mathrm{O}}$	Componente y do vetor $\mathbf{p}_{\mathrm{R}}^{\mathrm{O}}$
$\dot{\mathbf{p}}_{\mathrm{R}}^{\mathrm{O}}$	Vetor da velocidade do robô com respeito ao sistema inercial O
$\dot{p}_{\mathrm{R},x}^{\mathrm{O}}$	Componente x do vetor $\dot{\mathbf{p}}_{\mathrm{R}}^{\mathrm{O}}$

$\dot{p}^{\mathrm{O}}_{\mathrm{R},y}$	Componente y do vetor $\dot{\mathbf{p}}^{\mathrm{O}}_{\mathrm{R}}$
$\dot{\mathbf{p}}^{\mathrm{O}}_{\mathrm{R},k}$	Vetor da velocidade do robô no instante de tempo k com respeito ao sistema inercial O
$\dot{p}^{\mathrm{O}}_{\mathrm{R},x,k}$	Componente x do vetor $\dot{\mathbf{p}}^{\mathrm{O}}_{\mathrm{R},k}$
$\dot{p}^{\mathrm{O}}_{\mathrm{R},y,k}$	Componente y do vetor $\dot{\mathbf{p}}^{\mathrm{O}}_{\mathrm{R},k}$
$\dot{\mathbf{p}}^{\mathrm{O}}_{\mathrm{R},k+1}$	Vetor da velocidade do robô no instante de tempo $k+1$ com respeito ao sistema inercial O
$\dot{p}^{\mathrm{O}}_{\mathrm{R},x,k+1}$	Componente x do vetor $\dot{\mathbf{p}}^{\mathrm{O}}_{\mathrm{R},k+1}$
$\dot{p}^{\mathrm{O}}_{\mathrm{R},y,k+1}$	Componente y do vetor $\dot{\mathbf{p}}^{\mathrm{O}}_{\mathrm{R},k+1}$
$\mathbf{p}_{\mathrm{R},k}$	Vetor da posição do robô instante de tempo k
$\mathbf{p}_{\mathrm{R},k+1}$	Vetor da posição do robô instante de tempo $k+1$
$\mathbf{p}_{\mathrm{R},k-n}$	Vetor da posição do robô instante de tempo $k-n$
$\mathbf{p}_{\mathrm{R},k+n}$	Vetor da posição do robô instante de tempo $k+n$
$\mathbf{p}_{\mathrm{R},1}$	Vetor da primeira posição do robô
$\mathbf{p}_{\mathrm{R},N}$	Vetor da última posição do robô
$\mathbf{p}_{g,k}$	Pose do *ground truth* no instante de tempo k
$\mathbf{p}_{\mathrm{L}}$	Vetor da posição estimada pela odometria LiDA
$\mathbf{p}^{\overline{\mathrm{O}}}_{\mathrm{L}}$	Vetor da posição estimada pela odometria LiDAR com respeito ao sistema inercial $\overline{\mathrm{O}}$
$\mathbf{p}^{\overline{\mathrm{O}}}_{\mathrm{L},k}$	Vetor da posição da odometria LiDAR no instante de tempo k com respeito ao sistema inercial $\overline{\mathrm{O}}$
$\mathbf{p}^{\overline{\mathrm{O}}}_{\mathrm{L},k-n}$	Vetor da posição da odometria LiDAR no instante de tempo $k-n$ com respeito ao sistema inercial $\overline{\mathrm{O}}$
$\mathbf{p}^{\mathrm{L},k-n}_{\mathrm{L},k}$	Vetor da posição $\mathbf{p}^{\overline{\mathrm{O}}}_{\mathrm{L},k}$ representada com respeito à pose $\mathbf{p}^{\overline{\mathrm{O}}}_{\mathrm{L},k-n}$
$\mathbf{p}^{\overline{\mathrm{O}}}_{\mathrm{M}}$	Vetor da posição estimada pelo mapeamento LiDAR com respeito ao sistema inercial $\overline{\mathrm{O}}$
$\mathbf{p}^{\overline{\mathrm{O}}}_{\mathrm{I}}$	Vetor da posição integrada do LiDAR SLAM com respeito ao sistema inercial $\overline{\mathrm{O}}$

$\mathbf{p}_{\mathrm{I},k}^{\overline{\mathrm{O}}}$	Vetor da posição integrada no instante de tempo k com respeito ao sistema inercial $\overline{\mathrm{O}}$
$\mathbf{p}_{\mathrm{I},k-n}^{\overline{\mathrm{O}}}$	Vetor da posição integrada no instante de tempo $k-n$ com respeito ao sistema inercial $\overline{\mathrm{O}}$
$\mathbf{p}_{\mathrm{I},k}^{\mathrm{I},k-n}$	Vetor da posição $\mathbf{p}_{\mathrm{I},k}^{\overline{\mathrm{O}}}$ representada com respeito à pose $\mathbf{p}_{\mathrm{I},k-n}^{\overline{\mathrm{O}}}$
$\mathbf{p}_{\mathrm{D}}$	Vetor da pose do sensor LiDAR ajustada pela IMU
$p_{\mathrm{D},x}$	Componente x do vetor $\mathbf{p}_{\mathrm{D}}$
$p_{\mathrm{D},y}$	Componente y do vetor $\mathbf{p}_{\mathrm{D}}$
$p_{\mathrm{D},z}$	Componente z do vetor $\mathbf{p}_{\mathrm{D}}$
$\mathbf{p}_{\mathrm{E}}$	Vetor da posição estimada pelo EKF aplicado ao LiDAR SLAM
$\mathbf{p}_{\mathrm{E},k+1\|k}$	Vetor da posição estimada a *priori* pelo EKF aplicado ao LiDAR SLAM
$\mathbf{p}_{\mathrm{E},k\|k}$	Vetor da posição estimada no instante de tempo k pelo EKF aplicado ao LiDAR SLAM
$p_{\mathrm{E},x,k\|k}$	Componente x do vetor $\mathbf{p}_{\mathrm{E},k\|k}$
$p_{\mathrm{E},y,k\|k}$	Componente y do vetor $\mathbf{p}_{\mathrm{E},k\|k}$
$p_{\mathrm{E},z,k\|k}$	Componente z do vetor $\mathbf{p}_{\mathrm{E},k\|k}$
$\mathbf{q}_{k-n,k+n}$	Vetor formado pelas posições do robô dadas nos instantes $k-n$ e $k+n$
$\mathbf{q}_{k,k-n}$	Vetor formado pelas posições do robô dadas nos instantes k e $k-n$
$\mathbf{Q}$	Matriz formada pelos vetores dados pela diferença entre pontos de uma nuvem de pontos e sua respectiva média no instante de tempo $k-1$
R	Sistema de coordenadas do robô
$\mathbb{R}^2$	Espaço de configuração $\mathbb{R}$ bidimensional
$\mathbb{R}^3$	Espaço de configuração $\mathbb{R}$ tridimensional
$\mathbf{R}$	Matriz de rotação
$\mathbf{R}_{\mathrm{A}}^{\mathrm{O}}$	Matriz de rotação do sistema de coordenadas A para o sistema de coordenadas inercial O
$\mathbf{R}_{\mathrm{B}}^{\mathrm{O}}$	Matriz de rotação do sistema de coordenadas B para o sistema de coordenadas inercial O

$\mathbf{R}_{\mathrm{B}}^{\mathrm{A}}$	Matriz de rotação do sistema de coordenadas B para o sistema de coordenadas A
$\mathbf{R}_x(\phi)$	Matriz de rotação elementar ao redor do eixo x dada pelo ângulo ϕ
$\mathbf{R}_y(\theta)$	Matriz de rotação elementar ao redor do eixo y dada pelo ângulo θ
$\mathbf{R}_z(\psi)$	Matriz de rotação elementar ao redor do eixo z dada pelo ângulo ψ
$\mathbf{R}_{\mathrm{R}}^{\mathrm{O}}$	Matriz de rotação do sistema de coordenadas do robô para o sistema de coordenadas inercial O
$\mathbf{R}_{\mathrm{L}}$	Matriz de rotação da pose do odometria LiDAR
$\mathbf{R}_{\mathrm{L}}^{\overline{\mathrm{O}}}$	Matriz de rotação da pose estimada pela odometria LiDAR L com respeito ao sistema inercial $\overline{\mathrm{O}}$
$\mathbf{R}_{\mathrm{L},k-n}^{\overline{\mathrm{O}}}$	Matriz de rotação da odometria LiDAR no instante de tempo $k-n$ com respeito ao sistema inercial $\overline{\mathrm{O}}$
$\mathbf{R}_{\overline{\mathrm{O}}}^{\mathrm{L},k-n}$	Matriz de rotação transposta de $\mathbf{R}_{\mathrm{L},k-n}^{\overline{\mathrm{O}}}$
$\mathbf{R}_{\mathrm{M}}^{\overline{\mathrm{O}}}$	Matriz de rotação da pose estimada pelo mapeamento LiDAR M com respeito ao sistema inercial $\overline{\mathrm{O}}$
$\mathbf{R}_{\mathrm{I}}^{\overline{\mathrm{O}}}$	Matriz de rotação da pose integrada I do LiDAR SLAM com respeito ao sistema inercial $\overline{\mathrm{O}}$
$\mathbf{R}_{\mathrm{I},k-n}^{\overline{\mathrm{O}}}$	Matriz de rotação pose integrada no instante de tempo $k-n$ com respeito ao sistema inercial $\overline{\mathrm{O}}$
s	Coeficiente de rugosidade de um ponto em função dos pontos vizinhos
s_{min}	Limiar mínimo para o coeficiente de rugosidade s
s_{max}	Limiar máximo para o coeficiente de rugosidade s
$\mathbf{s}_{\mathrm{I}}$	Ponto medido do ambiente pelo sensor LiDAR expresso por uma imagem 3D
$\mathbf{s}_{\mathrm{L}}$	Ponto medido do ambiente pelo sensor LiDAR
$s_{\mathrm{L},x}$	Coordenada x do vetor $\mathbf{s}_{\mathrm{L}}$
$s_{\mathrm{L},y}$	Coordenada y do vetor $\mathbf{s}_{\mathrm{L}}$
$s_{\mathrm{L},z}$	Coordenada z do vetor $\mathbf{s}_{\mathrm{L}}$

$\mathbf{s}_{\mathrm{L}}^{\mathrm{R}}$	Ponto medido do ambiente pelo sensor LiDAR com respeito ao sistema de coordenadas do robô R
$\mathbf{s}_{\mathrm{L}}^{\mathrm{O}}$	Ponto medido do ambiente pelo sensor LiDAR com respeito ao sistema de coordenadas inercial O
$\mathbf{s}_{\mathrm{L}}^{(1:m)}$	Conjunto de m vetores de medida do ambiente pelo sensor LiDAR
$\mathbf{s}_{\mathrm{L}}^{(i)}$	Vetor i-ésimo da medida do ambiente pelo sensor LiDAR
$\mathbf{s}_{\mathrm{L},k}$	Vetor de medida do ambiente pelo sensor LiDAR no instante de tempo k
$\mathbf{s}_{\mathrm{L},k}^{(1:m)}$	Conjunto de m vetores de medida do ambiente pelo sensor LiDAR no instante de tempo k
$\mathbf{s}_{\mathrm{L},k}^{(1)}$	Vetor da primeira medida do ambiente pelo sensor LiDAR no instante de tempo k
$\mathbf{s}_{\mathrm{L},k}^{(m)}$	Vetor m-ésimo da medida do ambiente pelo sensor LiDAR no instante de tempo k
$\mathbf{s}_{\mathrm{L},k}^{(j)}$	Vetor j-ésimo da medida do ambiente pelo sensor LiDAR no instante de tempo k
$\mathbf{s}_{\mathrm{L},k-1}^{(1)}$	Vetor da primeira medida do ambiente pelo sensor LiDAR no instante de tempo $k-1$
$\mathbf{s}_{\mathrm{L},k-1}^{(m)}$	Vetor m-ésimo da medida do ambiente pelo sensor LiDAR no instante de tempo $k-1$
$\mathbf{s}_{\mathrm{L},k-1}^{(j)}$	Vetor j-ésimo da medida do ambiente pelo sensor LiDAR no instante de tempo $k-1$
$\mathbf{s}_{\mathrm{L},1:k-1}$	Vetores de medidas do ambiente pelo sensor LiDAR do instante de tempo 1 até $k-1$
$\mathrm{S_L}$	Conjunto dos pontos medidos de um sensor LiDAR
$\mathbf{s}_{\mathrm{D}}^{(i)}$	Vetor da medida do i-ésimo ponto do sensor LiDAR ajustado
$\mathbf{s}_{\mathrm{D}}^{(j)}$	Vetor da medida do j-ésimo ponto do sensor LiDAR ajustado
$\mathbf{s}_{\mathrm{D}}^{(n)}$	Vetor da medida do n-ésimo ponto do sensor LiDAR ajustado
$\mathbf{s}_{\mathrm{D},k}^{(i)}$	Vetor da medida do i-ésimo ponto do sensor LiDAR no instante de tempo k ajustado

$\mathbf{s}_{\mathrm{D},k}^{(j)}$	Vetor da medida do j-ésimo ponto do sensor LiDAR no instante de tempo k ajustado
$\mathbf{s}_{\mathrm{D},k-1}^{(j)}$	Vetor da medida do j-ésimo ponto do sensor LiDAR no instante de tempo $k-1$ ajustado
$\mathbf{s}_{\mathrm{D},k-1}^{(n)}$	Vetor da medida do n-ésimo ponto do sensor LiDAR no instante de tempo $k-1$ ajustado
$\mathbf{s}_{\mathrm{R}}$	Ponto do sensor LiDAR com respeito ao sistema de coordenadas do robô R
S_{D}	Conjuntos dos pontos medidos do sensor LiDAR ajustados
S_{L}	Conjunto de pontos do sensor LiDAR
$\mathbb{S}^1$	Espaço de configurações esférico $\mathbb{S}$ unidimensional
t	Instante de tempo contínuo
$\mathbf{u}_k$	Vetor de entradas de controle no instante de tempo k
$\mathbf{u}_{k-1}$	Vetor de entradas de controle no instante de tempo $k-1$
$\mathbf{u}_{1:k}$	Vetores de entradas de controle do instante de tempo 1 até k
$\mathbf{U}$	Matriz unitária
$v_{\mathrm{R},d}$	Velocidade linear do flanco direito de um robô
$v_{\mathrm{R},e}$	Velocidade linear do flanco esquerdo de um robô
v_{R}	Velocidade linear do robô na direção do eixo x
$v_{\mathrm{R},k}$	Velocidade linear do robô na direção do eixo x no instante de tempo k
$\mathbf{V}$	Matriz unitária
$\mathbf{V}^*$	Matriz conjugada transposta da matriz unitária $\mathbf{V}$
x_{CIR}	Coordenada x do centro instantâneo de rotação
$\mathbf{x}_{\mathrm{A}}^{\mathrm{O}}$	Pose do sistema de coordenadas A com respeito ao sistema inercial O
$\mathbf{x}_{\mathrm{R}}$	Vetor da pose do robô
$\mathbf{x}_{\mathrm{R}}^{\mathrm{O}}$	Pose do robô com respeito ao sistema inercial O
$\mathbf{x}_{\mathrm{R},k-1}$	Pose do robô no instante de tempo $k-1$

$\mathbf{x}_{\mathrm{R},k}$	Pose do robô no instante de tempo k
$\mathbf{x}_{\mathrm{R},0:k-1}$	Conjunto de poses do robô do instante de tempo 0 até k
$\mathbf{x}_{\mathrm{R},1:k}$	Conjunto de poses do robô do instante de tempo 1 até k
$\mathbf{x}_{\mathrm{R},k\|k}$	Pose estimada do robô
$\mathbf{x}_{\mathrm{R},k+1\|k}$	Estimativa a *priori* da pose do robô
$\mathbf{x}_{\mathrm{R},k+1\|k+1}$	Estimativa a *posteriori* da pose do robô
$\mathbf{x}_{\mathrm{R}a,k}$	Pose aumentada do robô no instante de tempo k para o EKF SLAM
$\mathbf{x}_{\mathrm{L}}$	Pose da odometria LiDAR
$\mathbf{x}_{\mathrm{L}}^{\overline{\mathrm{O}}}$	Pose da odometria LiDAR com respeito ao sistema inercial $\overline{\mathrm{O}}$
$\mathbf{x}_{\mathrm{L},k}^{\overline{\mathrm{O}}}$	Pose da odometria LiDAR no instante de tempo k com respeito ao sistema inercial $\overline{\mathrm{O}}$
$\mathbf{x}_{\mathrm{L},k-n}^{\overline{\mathrm{O}}}$	Pose da odometria LiDAR no instante de tempo $k-n$ com respeito ao sistema inercial $\overline{\mathrm{O}}$
$\mathbf{x}_{\mathrm{L},k}^{\mathrm{L},k-n}$	Pose $\mathbf{x}_{\mathrm{L},k}^{\overline{\mathrm{O}}}$ representada com respeito à pose $\mathbf{x}_{\mathrm{L},k-n}^{\overline{\mathrm{O}}}$
$\mathbf{x}_{\mathrm{L}a}$	Pose relativa calculada pela integração dos dados da IMU
$\mathbf{x}_{\mathrm{M}}^{\overline{\mathrm{O}}}$	Pose estimada pelo mapeamento LiDAR com respeito ao sistema inercial $\overline{\mathrm{O}}$
$\mathbf{x}_{\mathrm{M},k-n}^{\overline{\mathrm{O}}}$	Pose estimada pelo mapeamento LiDAR no instante de tempo $k-n$ com respeito ao sistema inercial $\overline{\mathrm{O}}$
$\mathbf{x}_{\mathrm{I}}^{\overline{\mathrm{O}}}$	Pose integrada pelo LiDAR SLAM com respeito ao sistema inercial $\overline{\mathrm{O}}$
$\mathbf{x}_{\mathrm{I},k}^{\overline{\mathrm{O}}}$	Pose integrada no instante de tempo k com respeito ao sistema inercial $\overline{\mathrm{O}}$
$\mathbf{x}_{\mathrm{I},k-n}^{\overline{\mathrm{O}}}$	Pose integrada no instante de tempo $k-n$ com respeito ao sistema inercial $\overline{\mathrm{O}}$
$\mathbf{x}_{\mathrm{I},k}^{\mathrm{I},k-n}$	Pose $\mathbf{x}_{\mathrm{I},k}^{\overline{\mathrm{O}}}$ representada com respeito à pose $\mathbf{x}_{\mathrm{I},k-n}^{\overline{\mathrm{O}}}$
$\mathbf{x}_{\mathrm{E}}$	Pose estimada pelo EKF aplicado ao LiDAR SLAM
$\mathbf{x}_{\mathrm{E},k+1\|k}$	Pose estimada a *priori* pelo EKF aplicado ao LiDAR SLAM
$\mathbf{x}_{\mathrm{E},k+1\|k+1}$	Pose estimada a *posteriori* pelo EKF aplicado ao LiDAR SLAM

y_{CIR}	Coordenada y do centro instantâneo de rotação
$\mathbf{y}_{\mathrm{L},k}$	Vetor de saída da equação de estados do filtro de Kalman de tempo k
$\mathbf{y}_{\mathrm{L},k}^{(i)}$	Vetor da i-ésima saída da equação de estados do filtro de Kalman de tempo k
α	Fator de correção para o modelo *Skid Steering*
α_{res}	Resolução do ângulo de segmentação
α_{seg}	Ângulo de segmentação pré-determinado
α_{L}	Ângulo de segmentação calculado em relação aos pontos vizinhos
$\boldsymbol{\delta}_k$	Vetor da variável gaussiana que mapeia a aleatoriedade da medida no instante k
$\boldsymbol{\delta}_{k-1}^{(i)}$	Vetor da i-ésima variável gaussiana que mapeia a aleatoriedade da medida no instante $k-1$
Δt	Variação do tempo t
$\boldsymbol{\varepsilon}_k$	Vetor da variável gaussiana que mapeia a aleatoriedade na transição de estados no instante k
ζ	Escalar da equação de uma reta para parametrização de *features*
θ	Ângulo *pitch*
θ_{min}	Ângulo *pitch* inferior ou mínimo
θ_{res}	Ângulo de resolução *pitch*
θ_v	Ângulo *pitch* formado entre os vetores dados por dois pontos verticais do LiDAR em relação à origem do sensor
θ_h	Ângulo *pitch* formado entre os vetores dados por dois pontos horizontais do LiDAR em relação à origem do sensor
θ_g	Ângulo de limite para a análise dos pontos do piso
$\theta_{\mathrm{A}}^{\mathrm{O}}$	Componente de rotação θ do sistema A com respeito ao sistema inercial O
θ_{L}	Ângulo de rotação do eixo y de um sensor LiDAR
θ_{D}	Componente θ da pose do sensor LiDAR ajustada

θ_u	Componente θ da orientação da IMU
$\theta_{\mathrm{E},k+1\|k}$	Componente θ do vetor da posição estimada a *priori* pelo EKF aplicado ao LiDAR SLAM
$\boldsymbol{\mu}_{\mathrm{L},k}^{cj}$	Vetor da média da pose da célula cj para método NDT no instante de tempo k
$\boldsymbol{\mu}_{\mathrm{L},k-1}^{cj}$	Vetor da média da pose da célula cj para método NDT no instante de tempo $k-1$
$\boldsymbol{\mu}_{\mathrm{L}}$	Vetor da média da pose do sensor LiDAR
$\boldsymbol{\mu}_{\mathrm{L},k}$	Vetor da média da pose do sensor LiDAR no instante k
$\boldsymbol{\mu}_{\mathrm{L},k-1}$	Vetor da média da pose do sensor LiDAR no instante de tempo $k-1$
$\boldsymbol{\mu}_{\mathrm{L},k}^{cj}$	Vetor da média da pose do sensor LiDAR para a célula cj no instante k
$\sigma_{\phi,u}$	Variância da componente ϕ da orientação da IMU
$\sigma_{\theta,u}$	Variância da componente θ da orientação da IMU
$\sigma_{\psi,u}$	Variância da componente ψ da orientação da IMU
$\sigma_{\mathrm{R},x}$	Variância da componente x da pose do robô
$\sigma_{\mathrm{R},y}$	Variância da componente y da pose do robô
$\sigma_{\psi\mathrm{R}}$	Variância da componente ψ da pose do robô
$\boldsymbol{\Sigma}_{\mathrm{R}}$	Matriz de covariâncias da pose do robô
$\boldsymbol{\Sigma}_{\mathrm{R},k\|k}$	Matriz de covariâncias da pose estimada do robô
$\boldsymbol{\Sigma}_{\mathrm{R},k+1\|k}$	Matriz de covariâncias da estimativa a *priori* da pose do robô
$\boldsymbol{\Sigma}_{\mathrm{R},k+1\|k+1}$	Matriz de covariâncias da estimativa a *posteriori* da pose do robô
$\boldsymbol{\Sigma}_{\varepsilon,k}$	Matriz de covariâncias da variável gaussiana ε no instante k
$\boldsymbol{\Sigma}_{\delta,k}$	Matriz de covariâncias da variável gaussiana δ no instante k
$\boldsymbol{\Sigma}_{\mathrm{L}}$	Matriz de covariâncias da pose do sensor LiDAR
$\boldsymbol{\Sigma}_{\mathrm{L},k}^{cj}$	Matriz de covariâncias da pose do sensor LiDAR para a célula cj no instante k
$\boldsymbol{\Sigma}_{\mathrm{E},k+1\|k}$	Matriz de covariância estimada a *priori* pelo EKF aplicado ao LiDAR SLAM

$\boldsymbol{\Sigma}_{\mathrm{E},k+1\mid k+1}$	Matriz de covariância estimada a *posteriori* pelo EKF aplicado ao LiDAR SLAM
ϕ	Ângulo de *roll*
$\boldsymbol{\varphi}_{\mathrm{A}}^{\mathrm{O}}$	Representação mínima por ângulos de *roll*, *pitch* e *yaw* do sistema A com respeito ao sistema inercial O
$\phi_{\mathrm{A}}^{\mathrm{O}}$	Componente de rotação ϕ do sistema A com respeito ao sistema O
ϕ_{D}	Componente ϕ da pose do sensor LiDAR ajustada
ϕ_u	Componente ϕ da orientação da IMU
$\boldsymbol{\varphi}_{\mathrm{E}}$	Vetor da orientação estimada pelo EKF aplicado ao LiDAR SLAM
$\boldsymbol{\varphi}_{\mathrm{E},k+1\mid k+1}$	Vetor da orientação estimada a *posteriori* pelo EKF aplicado ao LiDAR SLAM
$\boldsymbol{\varphi}_u$	Vetor da orientação da IMU
ψ	Ângulo *yaw*
ψ_{res}	Ângulo de resolução *yaw*
$\psi_{\mathrm{A}}^{\mathrm{O}}$	Componente de rotação ψ do sistema A com respeito ao sistema O
ψ_{R}	Ângulo de rotação em torno do eixo z
$\psi_{\mathrm{R},k}$	Ângulo de rotação em torno do eixo z no instante de tempo k
$\psi_{\mathrm{R},k+1}$	Ângulo de rotação em torno do eixo z no instante de tempo $k+1$
$\dot{\psi}_{\mathrm{R}}$	Velocidade de rotação em torno do eixo z
ψ_{L}	Ângulo de rotação do eixo z de um sensor LiDAR
ψ_{D}	Componente ψ da pose do sensor LiDAR ajustada pela IMU
ψ_u	Componente ψ da orientação da IMU
$\psi_{\mathrm{E},k+1\mid k}$	Componente ψ da posição estimada a *priori* pelo EKF aplicado ao LiDAR SLAM
$\psi_{\mathrm{E},k\mid k}$	Componente ψ da posição estimada a pelo EKF aplicado ao LiDAR SLAM
ω_{R}	Velocidade angular do robô em torno do eixo x
$\omega_{\mathrm{R},k}$	Velocidade angular do robô em torno do eixo x no instante de tempo k

Sumário

Capítulo 1

Introdução

A utilização de robôs móveis autônomos em tarefas de inspeção e monitoramento de locais perigosos vem se tornando cada vez mais comum. Um exemplo de aplicação é a exploração de ambientes confinados, evitando a exposição de operadores aos riscos presentes neste tipo de ambiente. Um robô móvel autônomo deve ser capaz de determinar a sua posição, mapear um ambiente, e planejar caminhos até uma localização desejada, como exemplo, uma região de trabalho. Por meio do sensoriamento e do controle de navegação, o dispositivo pode seguir caminhos e trajetórias e desviar de obstáculos evitando avarias.

Entretanto, ambientes confinados oferecem muitas dificuldades para determinação da localização do robô assim como para o mapeamento, que são informações essenciais para a operação autônoma. A falta de sinal de GPS (*Global Positioning System*) devido à estrutura fechada que esses espaços possuem, impossibilita a utilização deste sensor para estimar a posição do dispositivo. Do mesmo modo, a existência de materiais ferromagnéticos, comuns em regiões de mineração, bem como a presença de equipamentos de grandes potências, podem causar interferências em magnetômetros que são frequentemente utilizados na composição de uma IMU (*Inertial Measurement Unit*), provocando erros de orientação. A presença água e lama, assim como sedimentos rochosos normais em cavernas naturais, tornam o solo lúbrico. Com isso, técnicas de odometrias que utilizam a velocidade das rodas são prejudicadas por acúmulo de erros devido ao escorregamento. Outra adversidade encontrada em ambientes confinados é a baixa iluminação que pode prejudicar a utilização de câmeras para aplicação de técnicas visuais. Os problemas são tão desafiadores que instigaram a realização de diferentes pesquisas e desenvolvimentos. Um exemplo disso é o desafio *Subterranean Challenge*[1] proposto pela agência norte-americana DARPA (*Defense Advanced Research Projects Agency*), que consiste numa competição de robôs móveis para a exploração de ambientes subterrâneos.

Para o funcionamento adequado de um sistema autônomo de um robô é preciso que a localização seja estimada *online* de forma a realimentar o sistema de controle de navegação,

[1] https://www.subtchallenge.com/

e que o mapa represente corretamente as informações do ambiente possibilitando um processamento eficiente pelos algoritmos de planejamento de caminhos. Os problemas da localização e mapeamento são intrinsecamente conectados de tal forma que para construir um mapa é necessário ter a uma localização exata do dispositivo, e para determinar a sua posição é necessário ter um mapa. Desta forma uma solução é a determinação da localização e mapeamento simultâneos (do inglês *Simultaneous Localization And Mapping* – SLAM). Existem muitas técnicas de SLAM e estas utilizam diferentes sensores como sonares, câmeras, LiDAR (*Light Detection And Ranging*), e também IMU e encoders de rodas para auxiliar na estimação da odometria.

Dentre as técnicas de SLAM, as abordagens visuais são largamente empregadas devido à robustez na percepção [Cadena et al., 2016]. Porém, a iluminação precária dos ambientes confinados pode dificultar a identificação de *features* do ambiente pelas câmeras, gerando erros na estimação da pose e consequentemente na construção do mapa. Outra solução mais adequada é a aplicação de técnicas de SLAM baseadas em sensores LiDAR, chamada LiDAR SLAM, que possuem uma medida precisa de distância, na faixa de centímetros, e longo alcance, podendo ir de 10 a 300 metros, além de serem inerentes as condições de iluminação. Sensores do tipo *Scannig* LiDAR 3D possuem múltiplos feixes de *lasers* e são comumente utilizados em aplicações de robótica devido a sua representação densa do ambiente em 360° [Siciliano et al., 2010]. Desta forma, com o uso desse sensor é possível criar representações geometricamente precisas do ambiente, permitindo gerar mapas com alcance maior que as técnicas visuais, além de serem mais esparsos, o que reduz o processamento computacional exigido pelo sistema de navegação autônoma para o cálculo de caminhos e trajetórias. Dadas essas circunstâncias, esta dissertação tem como foco o estudo e análise de técnicas de LiDAR SLAM com o objetivo de implementar e adaptar essa metodologia em um robô de inspeção de ambientes confinados.

As técnicas de LiDAR SLAM apresentam um bom desempenho na maioria dos ambientes confinados, quando estes possuem um número suficiente de *features* geométricas identificáveis. Porém, em dutos e galerias, que possuem estruturas longas e homogêneas, ocorre a subestimação do caminho percorrido pelo robô por apresentarem poucas *features* geométricas e consequentemente a construção de um mapa de comprimento inferior ao real, como descrito em [Ebadi et al., 2020]. A presença de poeira e neblinas podem ocasionar erros de superestimação. Em [Kolvenbach et al., 2020] os experimentos realizados em um túnel apresentaram erros de $42,8\%$ superior à extensão real. Já em [Chang et al., 2020] ocorreu estimação superior ao trajeto realizado pelo robô ao realizar o mapeamento de uma galeria, com erros extraordinariamente maiores que às demais técnicas apresentadas, mesmo utilizando a fusão dos dados do sensor IMU com a pose gerada pelo LiDAR SLAM. Uma solução estudada nesta dissertação consiste na integração da odometria gerada somente com os dado do LiDAR, chamada de odometria LiDAR, com a odometrias rodas e os dados da IMU, aplicada entre o *Back-End* e o *Front-End* da técnica LiDAR SLAM,

porém, utilizando uma covariância adaptativa dada em função do número de *features* identificadas do ambiente.

1.1 Motivação

Segundo a norma regulamentadora brasileira NR-33 [BRASIL, 2006], ambientes confinados são espaços insalubres não projetados para ocupação humana, com acesso restrito, baixa ventilação e luminosidade. Muitos desses ambientes estão presentes na indústria e na mineração como: galerias de dreno, moinhos de bola, silos e barragens de contenção; que exigem manutenção constante para manter o seu estado de conservação e pleno funcionamento, atendendo certos requisitos de segurança. Outro ambiente confinado muito encontrado em regiões de mineração são as cavernas naturais formadas pelo curso de águas, que necessitam ser exploradas para fins de preservação do bioma local ou, quando permitido, extração mineral. A investigação de cavernas naturais é uma atividade realizada por profissionais da espeleologia que realizam o estudo da formação dessas cavidades e seus conteúdos [Bögli, 2012]. Contudo, esta é uma atividade perigosa que oferece risco a vida dos exploradores devido à presença de animais peçonhentos, gases tóxicos, excrementos de morcegos, além de possíveis ocorrências de desabamentos da estrutura.

O uso de robôs móveis para realização de tarefas como inspeção, monitoramento e exploração é uma opção mais segura, pois evita a exposição dos operadores envolvidos ao risco. Contudo, do ponto de vista da robótica estes ambientes apresentam grandes desafios ao trabalho e à navegação dos dispositivos em seu interior.

A operação remota, controlada por profissionais à distância, necessitam da comunicação entre uma base e o robô. Essa comunicação pode ser feita por cabos, formando o que é chamado de cordão umbilical, ou via sinal de rádio. Estas duas opções oferecem problemas devido à limitação da distância máxima de percurso, dada pelas interferências na locomoção pelo arrasto dos cabos e influência de sinais eletromagnéticos na comunicação, além de apresentarem risco de perda do equipamento.

Outra opção é a navegação autônoma que permite maior mobilidade ao dispositivo na execução das tarefas a serem desempenhadas. Neste contexto, o Instituto Tecnológico Vale (ITV) e a Universidade Federal de Minas Gerais (UFMG), vêm desenvolvendo um robô móvel intitulado de EspeleoRobô (Figura 1), que irá atuar na inspeção e monitoramento de áreas confinadas [Azpurua et al., 2019].

O EspeleoRobô foi projetado inicialmente para ser teleoperado por um profissional treinado, com as inspeções sendo realizadas de forma visual por meio de câmeras acopladas ao robô. Com o intuito de aumentar o grau de autonomia do dispositivo, estão sendo realizadas pesquisas sobre técnicas de localização e mapeamento, planejamento de caminhos, e controle de navegação [Azpúrua et al., 2021]. Estas atividades fazem parte do projeto

Figura 1 – EspeleoRobô realizando inspeções em diferentes ambientes. Fonte: adaptado de [Azpurua et al., 2019].

"Dispositivo Robótico de Inspeção de Ambientes Restritos e Confinados", no qual o estudo realizado nesta dissertação está inserido.

1.2 Objetivos

O sistema de navegação autônoma de um robô necessita da localização precisa do dispositivo, publicada *online* para realimentar o controle de navegação, além de mapas representativos e leves o suficiente para permitir um processamento rápido e eficiente do algoritmo de planejamento de caminhos. Desta forma, é crucial que a técnica de LiDAR SLAM embarcada no robô seja leve computacionalmente para ser executada de forma *online*. Portanto, o objetivo dessa dissertação é investigar 3 técnicas de SLAM baseadas em sensores LiDAR consideradas como estado da arte, LOAM-Velodyne, LeGO-LOAM e HDL-Graph-SLAM, avaliando os seus desempnehos para determinar qual delas é a mais adequada para implementação, adaptação e uso no EspeleoRobô. Desse modo, foram propostas métricas para avaliar a precisão das estimações de pose e dos mapas gerados.

Entre os objetivos específicos desta dissertação, é possível mencionar:

- Processamento dos dados fornecidos pelos sensores embarcados no EspeleoRobô, possibilitando a geração de nuvens de pontos;
- Implementação de ferramentas para simulação, incluindo modelos virtuais de ambientes de operação, e também do robô e dos sensores utilizados;
- Investigação de diferentes técnicas LiDAR SLAM, e definição da técnica mais adequada para ser embarcada no EspeleoRobô e integrada com o seu sistema de

navegação autônomo;

- Adaptação e implementação da técnica LiDAR SLAM de melhor desempenho no EspeleoRobô, fornecendo dados da pose e mapa para outros algoritmos implementados no robô de forma *online*;
- Implementação de um filtro EKF com covariância adaptativa integrado ao LiDAR SLAM para correção da odometria e do mapa em ambientes com poucas *features* geométricas;
- Validação experimental das técnicas LiDAR SLAM e do filtro EKF utilizando simulações e experimentos reais;
- Disponibilização dos algoritmos implementados, encapsulados em pacotes ROS (do inglês – *Robot Operating System*)[Quigley et al., 2009].

1.3 Contribuições

Uma contribuição desta dissertação consiste no estudo e análise de técnicas de SLAM baseadas em sensores LiDAR consideradas como estado da arte, permitindo a escolha de uma técnica para a adaptação e implementação no EspeleoRobô, e assim possibilitando a operação autônoma do mesmo.

As técnicas de LiDAR SLAM possuem bom desempenho em espaços confinados, porém, em ambientes com poucas *features* geométricas podem apresentar erros na estimação da pose do robô e consequentemente na construção do mapa. Desta forma, foi proposto e implementado um filtro de Kalman Estendido integrado ao SLAM para corrigir tanto a pose quanto o mapa, fundindo a odometria LiDAR com a odometria das rodas e IMU. Para melhorar o desempenho do filtro, foram determinadas funções para as covariâncias da odometria LiDAR com base no número de *features* identificadas do ambiente. Desta forma, é possível utilizar um robô autônomo para realizar inspeções dentro de um duto ou galeria, por exemplo.

Os resultados dos estudos e desenvolvimentos realizados durante o mestrado foram publicados em congressos e periódico científico:

- [Duarte et al., 2019] C. F. A. Duarte, I. F. S. Amaral, L. P. V. Dutra, G. P. d. Cruz Júnior, H. Azpúrua, and G. M. Freitas. “Utilização do Robot Operating System (ROS) em Conjunto com o Kit Didático Lego Mindstorms no Ensino de Robótica Móvel”. In Anais da Sociedade Brasileira de Automática, volume 1. Simpósio Brasileiro de Automação Inteligente, 2019. Neste artigo foi realizada uma primeira implementação de um filtro de Kalman aplicado à localização e mapeamento com robôs móveis.

- [Cruz Júnior et al., 2020] G. P. d. Cruz Júnior, L. V. d. C. Matos, H. Azpúrua, G. Pessin, and G. M. Freitas. "Investigação de Técnicas LiDAR SLAM para um Dispositivo Robótico de Inspeção de Ambientes Confinados". In Anais do Congresso Brasileiro de Automática 2020, dec 2020. Neste artigo foi apresentada uma comparação de 3 técnicas de LiDAR SLAM: LOAM-Velodyne, LeGO-LOAM e HDL-Graph-SLAM, onde foi determinada a técnica de melhor desempenho para uso no EspeleoRobô.

- [Rezende et al., 2020] A. M. C. Rezende, G. P. C. Junior, R. Fernandes, V. R. F. Miranda, H. Azpurua, G. Pessin, and G. M. Freitas. "*Indoor Localization and Navigation Control Strategies for a Mobile Robot Designed to Inspect Confined Environments*". In *2020 IEEE 16th International Conference on Automation Science and Engineering* (CASE), *pages* 1427–1433. IEEE, *aug* 2020. Neste artigo foi realizada uma comparação da técnica de LiDAR SLAM implementada no EspeleoRobô com outras técnicas de odometria como das rodas, visual e via sinal de rádio, em ambientes *indoors*.

- [Azpúrua et al., 2021] H. Azpúrua, A. Rezende, G. Potje, L. W. R. Filho, G. P. C. Júnior, R. Fernandes, V. Miranda, F. Rocha, F. L. M. de Sousa, L. G. D. de Barros, E. R. Nascimento, D. G. Macharet, G. Pessin, and G. Freitas. "*Towards semi-autonomous robotic inspection and mapping in confined spaces with the EspeleoRobô*". *Journal of Intelligent and Robotic Systems*, 2021. Neste artigo a técnica LiDAR SLAM implementada foi aplicada ao sistema de navegação autônoma do EspeleoRobô, fornecendo dados para os algoritmos de planejamento de caminhos e controle de navegação.

As contribuições desta dissertação estão inseridas no contexto do projeto "Dispositivo Robótico de Inspeção de Ambientes Restritos e Confinados", realizado em parceria entre o ITV e a UFMG, e contaram com a participação de várias pessoas que fazem parte das equipes de pesquisadores de ambas as instituições.

1.4 Estrutura da Dissertação

O restante desta dissertação está organizada em mais 6 Capítulos. No Capítulo 2 é apresentada uma revisão bibliográfica dos trabalhos publicados sobre robôs de serviços utilizados em ambientes confinados, técnicas de SLAM dcom foco em LiDAR SLAM, e também estratégias de fusão sensorial.

No Capítulo 3 são descritos os fundamentos teóricos necessários para a compreensão do SLAM no contexto da robótica móvel e as técnicas que compõem a estrutura do LiDAR SLAM.

No Capítulo 4 é apresentada a metodologia aplicada na dissertação para a análise, adaptação e fusão de técnica LiDAR SLAM, descrevendo as métricas utilizadas na avaliação do desempenho das estratégias investigadas, a técnica de SLAM implementada no EspeleoRobô e o desenvolvimento do filtro EKF proposto.

No Capítulo 5 é descrito o arcabouço experimental desenvolvido em simulador, as aplicações implementadas em ROS, assim como os cenários utilizados nos experimentos.

No Capítulo 6 são apresentados os resultados da comparação das técnicas de LiDAR SLAM, dos experimentos realizados *online* com o EspeleoRobô, e os resultados dos experimentos realizados com o filtro EKF integrado ao LiDAR SLAM.

No Capítulo 7 é realizada a conclusão da dissertação, apresentado os trabalhos futuros propostos.

Capítulo 2

Revisão Bibliográfica

Este Capítulo apresenta inicialmente uma revisão bibliográfica sobre os robôs de serviços desenvolvidos para execução de tarefas em ambientes confinados e subterrâneos, apresentando os sensores utilizados para localização do dispositivo e mapeamento do ambiente. Na sequência são apresentadas técnicas de localização e mapeamento simultâneos baseadas em sensores LiDAR, descritas sobre um contexto histórico. Por último, é realizada uma discussão sobre a fusão sensorial aplicada às técnicas de SLAM que têm o propósito de melhorar o seu desempenho em situações adversas encontradas em diferentes ambientes explorados por robôs móveis.

2.1 Robôs de Serviços em Ambientes Confinados

Os robôs de serviços são definidos segundo Wirtz et al. [2018] como interfaces autônomas e adaptáveis baseadas em um sistema que interage, comunica e fornece serviços aos clientes de uma organização. A rápida melhoria na tecnologia, como sensores mais preciso e baratos, junto ao avanço da robótica vêm oferecendo avanços no setor. Uma das motivações para o desenvolvimento de plataformas robóticas móveis é a sua utilização em ambientes que possuem restrições à presença de seres humanos, devido ao risco de danos físicos bem como à vida [Freitas et al., 2020]. Já as aplicações para proteção, segurança e resgate podem ser extremamente úteis em cenários de emergência, como acidentes de mineração ou colapsos de túneis, onde equipes de robôs podem ser usadas para realizar exploração cooperativa, intervenção ou missões logísticas [Tardioli et al., 2016].

Com o foco em aplicações de serviços de inspeção de ambientes subterrâneos de difícil acesso, como as cavernas naturais investigadas pelo EspeleoRobô, Larson et al. [2014] desenvolveram dois robôs para exploração, mapeamento e caracterização de túneis suspeitos de contrabando fronteiriços nos EUA, como ilustra a Figura 2. Uma das plataformas, denominada *Counter Tunnel Exploitation Robot* (CTER), é um robô móvel pequeno, articulado e reconfigurável preparado para entrar em orifícios pequenos. Este robô possui

uma câmera montada em uma das extremidades capaz de rotacionar 180° para uma visão de 360° e na outra extremidade dispõe de braços para transporte de pequenas cargas úteis. Como uma segunda opção os autores desenvolveram outro robô denominado iRobot PackBot que carrega um conjunto de sensores chamado de MultiSenseSL, dentre os quais possui uma câmera estéreo e um Hokuyo UTM-30LX-EW que rotaciona em torno do eixo longitudinal para mapeamento 3D.

Figura 2 – Robô CTER, robô iRobot PackBot, e *MultiSense-SL*. Fonte: [Larson et al., 2014].

Para inspeção de estruturas de tubos, que são estruturas homogêneas semelhante aos dutos e galerias inspecionados pelo EspeleoRobô, Tâche et al. [2011] apresentam um robô escalador com rodas magnéticas, denominado MagneBike (Figura 3(a)). Este pequeno dispositivo foi projetado para trabalhar em ambientes ferromagnéticos, logo, não é possível a utilização de IMU's devido às interferências nos magnetômetros. Desta forma, o robô foi equipado com um sensor LiDAR da Hokuyo URG-04LX montado em um motor para gerar modelos 3D. Portanto, a localização do dispositivo é estimada por meio da fusão da odometria LiDAR com a odometria dada pelo giro das rodas. Dado um ambiente semelhante, Sun et al. [2011] e [Wang et al., 2013] descrevem robôs projetados para realizar limpeza de dutos de ar, como ilustram as Figuras 3(b) e 3(c). O primeiro robô está equipado com dois sensores LiDAR 2D para construir um mapa 3D por meio de técnicas de SLAM e o segundo utiliza um par de câmeras trabalhando com uma estratégia visual.

(a) MagneBike (b) Robô de Dutos de Ar (c) Robô de Dutos de Ar

Figura 3 – (a) Robô MagneBike [Tâche et al., 2011]; (b) Robô de limpeza de dutos de ar-condicionado utilizando LiDAR [Sun et al., 2011]; e (c) Robô de limpeza de dutos de ar-condicionado utilizando câmeras [Wang et al., 2013].

Devido à complexidade da atuação de robôs móveis em ambientes confinados, bem como o incentivo à pesquisa e desenvolvimento da robótica para serviços de resgate em casos de acidentes em ambientes subterrâneos, a agência norte-americana de defesa DARPA lançou a competição *Subterranean challenge*, no qual 9 equipes participam[1]: CERBERUS, CoSTAR, CTU-CRAS-NORLAB, CRETISE, CSIRO Data61, Explorer, MARBLE, PLUTO, e Robotika.

Motivados por essa competição Dang et al. [2019] apresentam uma visão integral para o projeto de robôs de inspeção de ambientes subterrâneos com foco em veículos aéreos. Neste trabalho são descritas as contribuições da fusão de diferentes sensores para uma estimação robusta da localização e mapeamento, com objetivo análogo ao proposto para a fusão apresentada nesta dissertação, além de apresentar um novo método de planejamento de caminhos em ambientes ramificados e como túneis. O robô utilizado é o DJI Matrice 100 quadrotor (Figura 4) que carrega um computador Intel NUC Core-i7, um LiDAR da Velodyne PuckLITE, uma câmera FLIR Blackfly, uma IMU VectorNav VN-100 e câmera térmica FLIR Boson.

Figura 4 – Robô DJI Matrice 100 quadrotor desenvolvido para o desafio DARPA *SubT Challange* e um mapa gerado em um dos experimentos. Fonte: [Dang et al., 2019]

Outro trabalho desenvolvido para a competição *Subterranean Challenge* é apresentado por Rouček et al. [2019] que descrevem uma proposta de exploração de ambientes subterrâneos utilizando multi-robôs. Os autores apresentam os algoritmos implementados, as estratégias desenvolvidas e as plataformas robóticas usadas pela equipe CTU-CRAS-NORLAB que ficou em terceiro lugar na rodada de circuitos de túneis. Para compor a equipe foram utilizados 7 robôs, como mostra a Figura 5, sendo: dois Tracked Absolem que são capazes de se locomoverem em terrenos difíceis e estão equipados cada um com um sensor *laser* SICK LMS151, utilizado para o SLAM, e uma omini-câmera PointGrey Ladubyg3 para detecção de objetos; um robô Husky A200 que se movimenta mais rapidamente e está equipado com um sensor RoboSense 3D LiDAR para o SLAM e cinco câmeras Bluefox RGB, posicionadas em 360° de visão, para identificação de objetos no ambiente; duas plataformas PhantomX Mark II que são dispositivos rastejadores de seis pernas capazes de se locomoverem em terrenos adversos e que estão equipados cada um com duas câmeras Intel Realsense, uma T265 para o Visual SLAM embarcado e uma D435i

[1] https://www.subtchallenge.com/teams.html

RGB-D para construir o mapa e detectar os objetos na cena; e dois drones quadrotor UAVs baseados no kit F450 do DJI que não possuem restrições de terrenos e estão equipados com um sensor RPLidar A3 usado para SLAM e uma câmera Bluefox RGB para detecção dos objetos.

Figura 5 – Robôs utilizados pela equipe CTU-CRAS-NORLAB na DARPA *Subterranean Challenge*. Fonte: [Rouček et al., 2019]

2.2 Localização e Mapeamento Simultâneo – Uma Perspectiva do LiDAR SLAM

Um dos elementos fundamentais para operação autônoma de um robô móvel consiste na sua localização. Esta informação é um requisito importante para realizar o mapeamento do ambiente e é indispensável para o controle de navegação. Considerando os diversos desafios ligados às operações em ambientes confinados e às dificuldades de obter a localização do robô, por exemplo, usando um sensor GPS, a posição e orientação podem ser calculadas a partir dos movimentos do dispositivo. Esta estratégia é conhecida como odometria e consiste em desenvolver um sistema matemático para descrever os movimentos do robô, integrando-os ao longo do tempo, com o objetivo de criar um modelo da posição e orientação atual [Dudek and Jenkin, 2010].

A construção de mapas é outra tarefa essencial na robótica móvel e os seus problemas, em muitos casos, estão associados às dificuldades da localização [Grisetti et al., 2007]. A localização e mapeamento simultâneos (SLAM) é baseada no fato que, para localizar o robô em um ambiente é necessário um mapa, e para construir um mapa é essencial a localização precisa do robô. Sem um mapa a estimação da pose desvia rapidamente do valor real. Contudo, o emprego do mapa do ambiente e a identificação de pontos de referência já visitados, por meio de técnicas de *loop closure*, reduz o erro na estimação da localização. Historicamente uma primeira solução para o problema do SLAM foi divulgada em 1986 no *IEEE International Conference on Robotics and Automation* (ICRA) em São Francisco, Califórnia. Diversos pesquisadores levantaram a questão trazendo algumas soluções como Smith and Cheeseman [1986] e Durrant-Whyte [1988] que descreveram a

relação entre os pontos de referência e as incertezas de forma geométrica, e Crowley [1989] que propôs um algoritmo baseado em Filtro de Kalman.

Segundo Cadena et al. [2016], os primeiros anos de pesquisas sobre o SLAM se classificam como "Era Clássica", de 1986 a 2004. Esse período foi marcado pela sua formulação probabilística, no qual se destacaram o filtro EKF, o filtro de partículas Rao-Blackwellized e a Estimação por Máxima Verossimilhança. Um segundo período, datado de 2004 a 2015, foi denominado de "Era da Análise Algorítmica", no qual foram estudadas as propriedades fundamentais do SLAM como observabilidade, convergência e consistência, principalmente cobertas em [Dissanayake et al., 2011]. Essa época foi marcada também pelos desenvolvimentos de bibliotecas de código aberto. Uma terceira era chamada de "Era da Percepção Robusta", de 2015 em diante, é caracterizada pela robustez no desempenho das técnicas desenvolvidas, pela compreensão em alto nível do ambiente, incluindo a identificação de recursos, geometria, semântica e física, bem como por uma percepção orientada às tarefas, filtrando as informações irrelevantes.

No final da era clássica os autores Thrun et al. [2000] apresentam um novo método baseado na Máxima Verossimilhança para solucionar o problema de localização e mapeamento de forma simultânea – chamado de *Current Mapping and Localization* (CML) – em ambientes que possuem circuitos fechados, executado *online* com um computador de baixo custo e aplicado a multi-robôs. Como uma extensão da metodologia, os autores apresentam neste trabalho uma técnica de geração de mapas 3D, além de testes de desempenho utilizando somente os dados do sensor *laser* para mostrar a robustez no mapeamento sem a necessidade de uma odometria.

Entrando na era da análise algorítmica, uma das técnicas de SLAM amplamente conhecida é descrita em [Grisetti et al., 2007], onde é apresentado um algoritmo de código aberto para SLAM chamado GMapping (*Grid Mapping*), desenvolvido para o uso de sensores *laser scanning* 2D, com sua metodologia baseada no filtro de partículas *Rao-Blackwellized* (do inglês *Rao-Blackwellized Particle Filter* – RBPF). Um grande problema desse filtro é a sua complexidade, que é medida em números de partículas necessárias para construir um mapa preciso. Grisetti et al. [2007] apresentam duas técnicas para melhorar o desempenho do RBPF, uma que propõe um modelo de distribuição considerando a precisão dos sensores e outra por meio de uma técnica de reamostragem adaptativa das partículas.

Com o desenvolvimento de muitas técnicas de SLAM alguns trabalhos tiveram o foco no estudo e investigação das metodologias já consolidadas. Kurt-Yavuz and Yavuz [2012] comparam os principais algoritmos de SLAM, sendo eles: EKF SLAM, *Unscented Kalman Filter* (UKF) SLAM, EKF-based FastSLAM versão 2.0 e UKF-based FastSLAM (uFastSLAM). Já Vincent et al. [2010] comparam as técnicas de SLAM como Monte Carlo e Graph-SLAM aplicadas em ambientes *indoor* sem sinal de GPS. Para isso os

autores criaram um conjunto de dados, chamado RAWSEEDS *benchmark*, que possuem informações salvas dos sensores *lasers*, odometria, imagens de câmeras e um *ground truth* com precisão de 5 cm. Com a presença de novos recursos computacionais, Costa et al. [2010] propõe e avalia um algoritmo baseado no ICP que resolve o problema de mínimos locais. Para realizar os experimentos os autores utilizaram um robô Pioneer P3DX no simulador *Robotics Developer Studio* (RDS) da Microsoft.

Entrando um pouco na era da percepção robusta Zhang and Singh [2014] apresentam a técnica LOAM (*LiDAR Odometry And Mapping*) que é investigada nesta dissertação. Essa é uma proposta *online* de baixo consumo computacional, baixo desvio e baseada no algoritmo ICP. Nela os autores apresentam uma metodologia para o SLAM 3D utilizando um LiDAR 2D rotativo. A ideia chave dessa abordagem é dividir o problema do SLAM em dois algoritmos buscando otimizar um grande número de variáveis de forma simultânea. Um algoritmo executa a odometria LiDAR calculando a pose relativa entre duas varreduras do sensor, porém com baixa fidelidade e alta frequência, e o outro algoritmo executa o registro dos dados, bem como a estimação da pose do dispositivo robótico com respeito ao mapa global, de forma mais precisa e a um frequência mais baixa.

Utilizando os sensores *scanning* LiDAR 3D as técnicas LiDAR SLAM se apropriam de metodologias que antes eram empregadas pelas técnicas de Visual SLAM. Nesse sentindo, Deschaud [2018] apresentam uma nova abordagem para SLAM baseada nos métodos RGB-D lidando com as esparsividadse dos dados da nuvem de pontos gerada por um sensor LiDAR 3D, chamado de *Implicit Moving Least Squares*-SLAM (IMLS-SLAM). A técnica possui baixo desvio, em torno de $0,4\%$, para um percurso de 4 km sem ciclos fechados e usando um sensor Velodyne HDL32. A metodologia também foi testada nos conjuntos de dados da KITTI *benchmark*[2] apresentado [Geiger et al., 2012], ficando entre as 20 melhores técnicas com um desvio de $0,69\%$.

Outro trabalho relevante investigado nessa dissertação é apresentado em [Shan and Englot, 2018] que descreve a técnica LeGO-LOAM (Lightweight and GroundOptimized LiDAR Odometry And Mapping). A metodologia apresentada é baseada na técnica LOAM, porém, otimizada para veículos terrestre e utilizando um sensor LiDAR 3D. Os resultados obtidos pelos autores foram superiores à técnica de referência devido à maior leveza em relação ao consumo computacional exigido e aplicação da técnica de *loop closure*. Em uma abordagem semelhante, Neuhaus et al. [2018] propõem uma metodologia para a odometria LiDAR com baixo desvio e online que integra múltiplos feixes do LiDAR com um sensor inercial, chamada MC2SLAM. Os resultados obtidos com a técnica também foram testados com os dados do KITTI *benchmark* ficando entre as 5 melhores técnicas baseadas somente no sensor LiDAR.

Mais uma metodologia investigada nesta dissertação é apresentada por Koide

[2] http://www.cvlibs.net/datasets/kitti/

et al. [2019] que descrevem uma técnica de LiDAR SLAM 3D aplicável em grandes áreas e que permite a detecção e rastreamento de pessoas. Esta técnica é chamada de HDL-Graph-SLAM em referência ao sensor Velodyne HDL32 utilizado, e está disponibilizada em algoritmos com código aberto. A metodologia é baseada no Graph-SLAM e utiliza um sensor inercial para correção da distorção ocasionada na ego-movimentação[3], fundindo com os dados da odometria LiDAR por meio de um filtro UKF.

Investigação de Técnicas de LiDAR SLAM para um Robô de Inspeção de Ambientes Confinados

Um dos obejtivos desta dissertação é o estudo e investigação de três técnicas de LiDAR SLAM consideradas com estado da arte: LOAM [Zhang and Singh, 2014], LeGO-LOAM [Shan and Englot, 2018] e HDL-Graph-SLAM [Koide et al., 2019].

A análise destas técnicas de LiDAR SLAM tem como finalidade a escolha e implementação de uma delas no EspeleoRobô. A abordagem escolhida deve fornecer informações sobre a localização do dispositivo de forma *online*, além de um mapa geometricamente representativo do ambiente, aos demais algoritmos do sistema de navegação autônoma em desenvolvimento.

2.3 Fusão Sensorial Aplicada ao SLAM

Muitas técnicas consideradas como estado da arte têm focado na solução do problema do SLAM utilizando sensores específicos como RADAR, LiDAR, câmeras mono e estéreos. Debeunne and Vivet [2020] apresentam uma revisão das técnicas de SLAM Visual e LiDAR descrevendo a fusão das duas metodologias. Os autores ressaltam que no caso de sensores heterogêneos é interessante notar as vantagens e desvantagens de cada modalidade. Sensores LiDAR são bons para detecção de obstáculos e rastreamentos, porém, são sensíveis a espaços com geometria homogênea como longos corredores. Por outro lado, as câmeras são boas para obter uma detecção semântica do ambiente. Contudo, são altamente sensíveis à variação da iluminação. A fusão das duas técnicas permite explorar as vantagens dos dispositivos heterogêneos de instrumentação.

Uma das técnicas de Visual SLAM que foi beneficiada pela fusão com o sensor LiDAR é descrita por Graeter et al. [2018] como LIMO (LiDAR Monocular Visual Odometry). Esta estratégia utiliza a nuvem de pontos do sensor LiDAR como medida de profundidade associada à imagem da câmera. De uma forma diferente, uma técnica de LiDAR SLAM beneficiada pelo Visual-SLAM é apresentada por Zhang and Singh [2018], denominada V-LOAM (*Visual-LiDAR Odometry And Mapping*). Neste caso, a nuvem de

[3] Ego-movimentação é definido como o movimento do robô estimado durante o tempo de aquisição de dados [Deschaud, 2018]

pontos do sensor LiDAR é associada ao mapa de profundidade gerado pela técnica de Visual SLAM, registrada com respeito ao mesmo sistema de coordenadas.

Para ambientes subterrâneos foram desenvolvidos diversos trabalhos incentivados pelo desafio da agência de defesa norte-americana DARPA *Subterranean Challenge*, como já abordado anteriormente no texto. Nesse sentido, Khattak et al. [2020] utiliza os sensores LiDAR, câmera RGB, câmera térmica e IMU para realizar mapeamento e localização multimodal, como mostra o diagrama da Figura 6. Para isso, são aplicadas as técnicas LOAM [Zhang and Singh, 2014] para o SLAM fundindo os dados da IMU e LiDAR; *Robust Visual-Inertial Odometry* (ROVIO) [Bloesch et al., 2015] para fusão entre a câmera e IMU; e *Keyframe-based Thermal–Inertial Odometry* (KTIO) [Khattak et al., 2019] para fusão da câmera térmica e IMU. As 3 técnicas são integradas em cascata utilizando um filtro de Kalman Estendido.

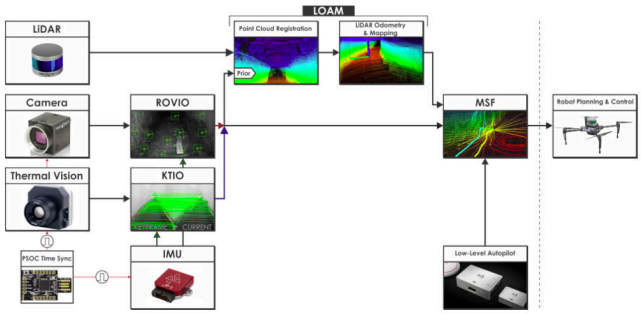


Figura 6 – Diagrama do sistema de mapeamento e localização multimodal utilizando diferentes sensores para uso em ambientes subterrâneos. Fonte: [Dang et al., 2019]

Outro trabalho que busca otimizar as vantagens de cada sensor é descrito em [Hening et al., 2017], que apresenta um algoritmo para fusão de dados entre os sensores LiDAR, *Inertial Navigation System* (INS) e GPS usando um Filtro de Kalman Adaptativo. Esta abordagem é baseada na utilização de uma matriz de medidas de ruídos adaptativa, dada em relação à alguns parâmetros como a precisão das técnicas de estimação. No caso da técnica LiDAR, o parâmetro é medido pelo erro do algoritmo ICP dado em função do número de *features* detectadas. Já em [Wisth et al., 2021] é apresentado um sistema de odometria multissensorial para plataformas robóticas móveis juntando as estimações dadas por uma técnica Visual, LiDAR e IMU, com ilustra a Figura 7(a). Nos experimentos realizados pelos autores, o sistema lidou bem com as diferentes restrições dos ambientes testados usando a melhor informação de cada sensor por meio de uma função de custo robusta baseada no *Dynamic Covariance Scaling* (DCS).

Uma nova metodologia que elimina os objetos móveis identificados do ambiente de forma a obter um desempenho mais robusto do LiDAR SLAM é proposta em [Dang et al., 2021]. A técnica apresentada funde os dados do sensor LiDAR com um sensor mmWRADAR. Outra abordagem com um escopo semelhante ao apresentado nessa dissertação é descrita

(a) Sistema para odometria

(b) Rover de 6 rodas

Figura 7 – (a) Sistema de odometria composto por um sensor LiDAR, uma camera RGBD e uma IMU. Fonte: [Wisth et al., 2021]. (b) Rover de 6 rodas com um braço articulado de 6 graus de liberdade. Fonte: [Su et al., 2021]

por Su et al. [2021], denominada GR-LOAM (*Ground Robots LiDAR Odometry And Mapping*). A metodologia estima da pose de um rover (Figura 7(b)) dada pela fusão dos dados dos sensores LiDAR, IMU e pelas medidas dos *enconders* das rodas. Uma das funções da odometria das rodas é refinar a estimação da pose dada pelo LiDAR reduzindo o desvio e melhorando o alinhamento das *features* com o mapa. Para obter bons desempenhos em diferentes cenários, os autores propuseram a utilização de funções denominadas *favor factors* para otimizar os pesos correspondentes a cada sensor.

Fusão da Odometria LiDAR com Odometria das Rodas e IMU Realizada Nesta Dissertação

A fusão sensorial proposta nesta dissertação adota uma estratégia semelhante à apresentada em [Su et al., 2021], que consiste em fornecer uma pose corrigida para o alinhamento entre mapas local e global do ambiente. Porém, a técnica proposta aqui tem como foco a aplicação em ambientes com poucas *features* geométricas como dutos e galerias, e também em espaços mistos incluindo ambientes *indoors* como as instalações de prédios que apresentam longos corredores homogêneos, além de ambientes *outdoors*.

Da mesma forma que em [Hening et al., 2017], a estratégia proposta aqui utiliza uma covariância adaptativa com um filtro de Kalman para fundir diferentes sensores. Hening et al. [2017] utilizam uma única matriz de covariância definida em função dos dados do GPS e erros do algoritmo ICP. Já a metodologia aplicada nesta dissertação utiliza matrizes de covariâncias diferentes para cada estratégia de localização. A odometria das rodas possui uma matriz de covariâncias que depende da velocidade angular em z do robô, alterando os valores das variâncias para movimentos em linha reta e em rotações. Para a IMU é utilizado uma covariância constante para a orientação, fornecida pelo fabricante do sensor. E por fim, a odometria LiDAR possui uma matriz de covariâncias que depende do número de *features* de borda e plano identificadas no ambiente, adaptando as variâncias da posição e orientação estimadas.

Capítulo 3

Fundamentos Teóricos

Este Capítulo descreve os fundamentos teóricos utilizados para descrever a movimentação de robôs móveis, bem como para compreender as técnicas de localização e mapeamento simultâneos baseadas em sensores LiDAR. A primeira Seção apresenta a cinemática de corpos rígidos descrevendo as transformações necessárias para representar a pose de um dispositivo robótico com respeito a diferentes sistemas de coordenadas. Já a segunda Seção descreve o modelo cinemático de robôs com rodas utilizado no EspeleoRobô para calcular os seus movimentos. Na terceira Seção são relatados os principais filtros Bayesianos aplicados a localização. Em seguida, na quarta Seção são apresentadas as principais estratégias de SLAM por filtros probabilísticos. E por fim, a quinta Seção retrata algumas das técnicas que fazem parte dos fundamentos do LiDAR SLAM.

3.1 Cinemática de Corpos Rígidos

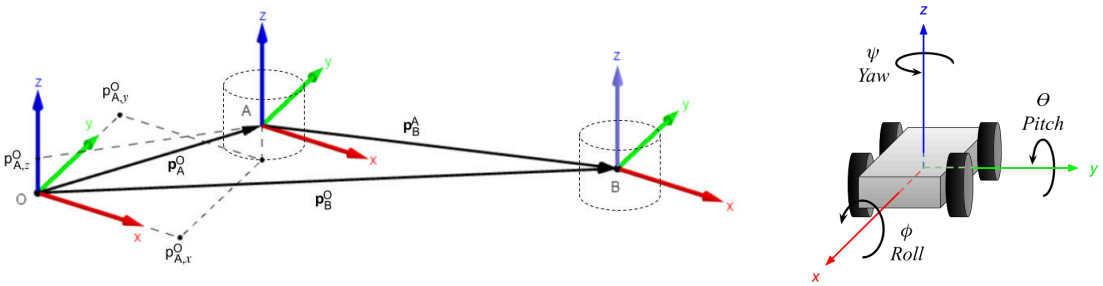
A cinemática estuda a movimentação de corpos rígidos desconsiderando as ações de forças e/ou torques, descrevendo a sua posição e orientação, e as derivadas de ordem superior. Um corpo rígido é definido como um conjunto de partículas agrupadas que possuem distâncias fixas entre si. Posto isso, a sua movimentação pode ser estudada considerando um único ponto, por exemplo, o seu centro de massa, facilitando a interpretação dos modelos matemáticos. Em robótica móvel a cinemática se concentra em determinar a posição e orientação do robô em função do deslocamento das rodas.

Para descrever a posição e orientação de um corpo rígido com respeito a um sistema de coordenadas inercial O, é necessário acoplar um sistema de coordenadas local ao corpo. Como as dimensões de um corpo rígido não se alteram, para fins de simplificação este corpo é considerado como um ponto localizado na origem do seu sistema de coordenadas. Portanto, para representar a posição de um corpo rígido em A com respeito ao sistema O,

é utilizado um vetor $\mathbf{p}_{\mathrm{A}}^{\mathrm{O}} \in \mathbb{R}^3$, como mostra a Equação 3.1:

$$\mathbf{p}_{\mathrm{A}}^{\mathrm{O}} = \begin{bmatrix} p_{\mathrm{A},x}^{\mathrm{O}} \\ p_{\mathrm{A},y}^{\mathrm{O}} \\ p_{\mathrm{A},z}^{\mathrm{O}} \end{bmatrix}, \tag{3.1}$$

onde $p_{\mathrm{A},x}^{\mathrm{O}}$, $p_{\mathrm{A},y}^{\mathrm{O}}$ e $p_{\mathrm{A},z}^{\mathrm{O}}$ são as componentes x, y e z de um sistema de coordenadas cartesiano, conforme ilustrado na Figura 8(a).



(a) Posição (b) Orientação

Figura 8 – (a) Representação gráfica da posição de um corpo rígido acoplado ao sistema de coordenadas A com respeito ao sistema de coordenadas inercial O, e da posição de um corpo rígido em B com respeito ao sistema de coordenadas A e com respeito sistema de coordenadas inercial O. (b) Representação dos ângulos *roll*, *pitch* e *yaw*, com respeito a um robô móvel.

A orientação de um corpo em A é definida pela rotação do seu sistema de coordenadas com respeito a um sistema de coordenadas de referencia O, dada por uma matriz de rotação que pertencente ao grupo especial ortonormal de dimensão 3 (do inglês – *Special Orthonormal*), de forma que $\mathbf{R}_{\mathrm{A}}^{\mathrm{O}} \in SO(3)$:

$$SO(3) = \{\mathbf{R} \in \mathbb{R}^{3\times3} : \mathbf{R}\mathbf{R}^T = \mathbf{I} \text{ e } det(\mathbf{R}) = 1\}, \tag{3.2}$$

onde $\mathbf{I} \in \mathbb{R}^{3\times3}$ é uma matriz identidade.

Na robótica existem diferentes formas para expressar a orientação de um corpo rígido, além da matriz de rotação $\mathbf{R}$. As mais utilizadas são as representações mínimas por ângulos de Euler que decompõe a matriz de rotação em 3 rotações elementares em torno dos eixos do sistema de coordenadas do corpo, e também por ângulos de *roll*, *pitch* e *yaw* que decompõe a matriz de rotação em 3 rotações elementares ao redor do sistema de coordenadas inercial, conforme ilustrado na Figura 8(b).

A orientação do sistema A com respeito ao sistema O utilizando a representação mínima por ângulos de *roll*, *pitch* e *yaw* é dada por:

$$\boldsymbol{\varphi}_{\mathrm{A}}^{\mathrm{O}} = \begin{bmatrix} \phi_{\mathrm{A}}^{\mathrm{O}} \\ \theta_{\mathrm{A}}^{\mathrm{O}} \\ \psi_{\mathrm{A}}^{\mathrm{O}} \end{bmatrix}, \tag{3.3}$$

onde $\phi_{\mathrm{A}}^{\mathrm{O}}$ esta representando o ângulo de *roll*, $\theta_{\mathrm{A}}^{\mathrm{O}}$ o ângulo *pitch* e $\psi_{\mathrm{A}}^{\mathrm{O}}$ o ângulo *yaw*. A matriz $\mathbf{R}_{\mathrm{A}}^{\mathrm{O}}$ equivalente é resultante da multiplicação das matrizes de rotações elementares de acordo com:

$$\mathbf{R}_{\mathrm{A}}^{\mathrm{O}} = \mathbf{R}_z(\psi)\mathbf{R}_y(\theta)\mathbf{R}_x(\phi). \tag{3.4}$$

onde $\mathbf{R}_x(\phi)$, $\mathbf{R}_y(\theta)$ e $\mathbf{R}_z(\psi)$ são as matrizes de rotações elementares ao redor dos eixos x, y e z dadas pelos ângulos ϕ, θ e ψ respectivamente.

A pose de um corpo rígido em A com respeito ao sistema de coordenadas O é composta pela posição $\mathbf{p}_{\mathrm{A}}^{\mathrm{O}}$ e pela orientação $\mathbf{R}_{\mathrm{A}}^{\mathrm{O}}$, definida como:

$$\mathbf{x}_{\mathrm{A}}^{\mathrm{O}} = \left(\mathbf{p}_{\mathrm{A}}^{\mathrm{O}}, \mathbf{R}_{\mathrm{A}}^{\mathrm{O}}\right), \tag{3.5}$$

que também pode ser definida em função da representação mínima de orientação $\boldsymbol{\varphi}_{\mathrm{A}}^{\mathrm{O}}$, conforme Equação 3.6:

$$\mathbf{x}_{\mathrm{A}}^{\mathrm{O}} = \begin{bmatrix} \mathbf{p}_{\mathrm{A}}^{\mathrm{O}} \\ \boldsymbol{\varphi}_{\mathrm{A}}^{\mathrm{O}} \end{bmatrix}. \tag{3.6}$$

É possível representar a pose de um corpo rígido B com respeito ao sistema A no sistema inercial O utilizando as transformações descritas nas Equações 3.7 e 3.8:

$$\mathbf{p}_{\mathrm{B}}^{\mathrm{O}} = \mathbf{p}_{\mathrm{A}}^{\mathrm{O}} + \mathbf{R}_{\mathrm{A}}^{\mathrm{O}}\mathbf{p}_{\mathrm{B}}^{\mathrm{A}}, \tag{3.7}$$

$$\mathbf{R}_{\mathrm{B}}^{\mathrm{O}} = \mathbf{R}_{\mathrm{A}}^{\mathrm{O}}\mathbf{R}_{\mathrm{B}}^{\mathrm{A}}, \tag{3.8}$$

onde $\mathbf{p}_{\mathrm{B}}^{\mathrm{O}}$ é a posição do ponto B com respeito ao sistema inercial O, e $\mathbf{R}_{\mathrm{B}}^{\mathrm{O}}$ é a matriz de rotação do sistema B para o sistema O, conforme ilustrado na Figura 8(a).

Outra forma de se representar a transformação entre sistemas de coordenadas é utilizando a matriz de transformação homogênea que pertencente ao grupo especial euclidiano de dimensão 3, de forma que $\mathbf{H} \in SE(3)$:

$$\mathbf{H} = \begin{bmatrix} \mathbf{R}_{3\times3} & \mathbf{p}_{3\times1} \\ \mathbb{O}_{1\times3} & 1 \end{bmatrix}, \tag{3.9}$$

onde $\mathbf{O} \in \mathbb{R}^{1\times3}$ é um vetor de zeros. Desta forma, a posição de um ponto B dado em relação ao sistema A com repeito ao sistema inercial O pode ser determinada como:

$$\overline{\mathbf{p}}_{\mathrm{B}}^{\mathrm{O}} = \begin{bmatrix} \mathbf{R}_{\mathrm{A}}^{\mathrm{O}} & \mathbf{p}_{\mathrm{A}}^{\mathrm{O}} \\ \mathbb{O}_{1\times3} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{p}_{\mathrm{B}}^{\mathrm{A}} \\ 1 \end{bmatrix} = \mathbf{H}_{\mathrm{A}}^{\mathrm{O}}\overline{\mathbf{p}}_{\mathrm{B}}^{\mathrm{A}}, \tag{3.10}$$

onde $\overline{\mathbf{p}}_{\mathrm{B}}^{\mathrm{O}} = [p_{\mathrm{B,x}}^{\mathrm{O}}\ p_{\mathrm{B,y}}^{\mathrm{O}}\ p_{\mathrm{B,z}}^{\mathrm{O}}\ 1]^T$ e $\overline{\mathbf{p}}_{\mathrm{B}}^{\mathrm{A}} = [p_{\mathrm{B,x}}^{\mathrm{A}}\ p_{\mathrm{B,y}}^{\mathrm{A}}\ p_{\mathrm{B,z}}^{\mathrm{A}}\ 1]^T$, e $\mathbf{H}_{\mathrm{A}}^{\mathrm{O}} \in SE(3)$ é a matriz de transformação homogênea do sistema de coordenadas A com respeito ao sistema O.

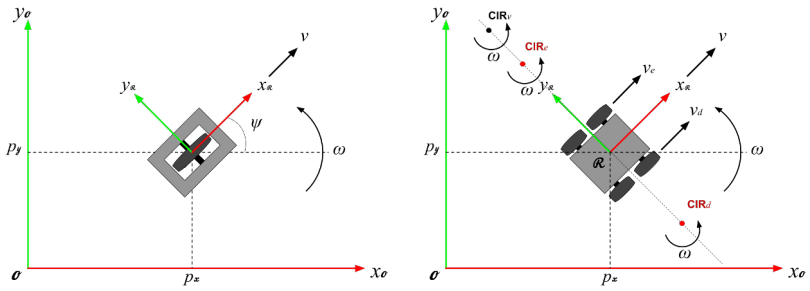
3.2 Modelos Cinemáticos de Robôs com Rodas

O modelo cinemático de um robô descreve o comportamento do sistema mecânico e as restrições impostas à sua movimentação. Um robô móvel se movimentando em um

plano possui um espaço de configurações $\mathbb{R}^2 \times \mathbb{S}^1$, onde $\mathbb{R}^2$ é o espaço das configurações de um plano e $\mathbb{S}^1$ é o espaço das configurações de um círculo. Sua configuração é representada pelo vetor $\mathbf{x}_\mathrm{R}$, como mostra a Equação 3.11:

$$\mathbf{x}_\mathrm{R} = \begin{bmatrix} p^\mathrm{O}_{\mathrm{R},x} \\ p^\mathrm{O}_{\mathrm{R},y} \\ \psi_\mathrm{R} \end{bmatrix}, \tag{3.11}$$

onde $p^\mathrm{O}_{\mathrm{R},x}$ e $p^\mathrm{O}_{\mathrm{R},y}$ são coordenadas planares da origem do sistema de coordenadas R acoplado ao dispositivo com respeito ao sistema inercial O, e ψ_R é o ângulo de rotação em torno do eixo z do robô conforme ilustrado na Figura 9.



(a) Modelo Monociclo (b) Modelo *Skid Steering*

Figura 9 – Robôs móveis do tipo (a) monociclo e (b) *skid steering* com sistema de coordenadas R representado com respeito ao sistema inercial O.

O modelo cinemático utilizado para descrever a movimentação do EspeleoRobô é o *skid-steering*, que pode ser derivado do modelo de um monociclo. O monociclo apresenta limitações à movimentação que impedem que o dispositivo realize movimentos laterais. Essas restrições chamadas de não-holonômicas são definidas com:

$$\dot{p}^\mathrm{O}_{\mathrm{R},x} \sin\psi_\mathrm{R} - \dot{p}^\mathrm{O}_{\mathrm{R},y} \cos\psi_\mathrm{R} = 0. \tag{3.12}$$

Partindo da Equação 3.12 de restrição não-holonômica, é possível obter um modelo que relaciona as velocidades linear v_R com relação ao eixo x do robô e angular ω_R com relação ao eixo z do robô com as velocidades lineares $\dot{p}^\mathrm{O}_{\mathrm{R},x}$ e $\dot{p}^\mathrm{O}_{\mathrm{R},y}$ e a velocidade angular $\dot{\psi}_\mathrm{R}$, tal que:

$$\begin{bmatrix} \dot{p}^\mathrm{O}_{\mathrm{R},x} \\ \dot{p}^\mathrm{O}_{\mathrm{R},y} \\ \dot{\psi}_\mathrm{R} \end{bmatrix} = \begin{bmatrix} \cos(\psi_\mathrm{R}) & 0 \\ \sin(\psi_\mathrm{R}) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_\mathrm{R} \\ \omega_\mathrm{R} \end{bmatrix}. \tag{3.13}$$

Dada a integração numérica pela aproximação de primeira ordem de Euler da Equação 3.13, a estimativa da pose do robô é dada por:

$$\begin{bmatrix} p^\mathrm{O}_{\mathrm{R},x,k+1} \\ p^\mathrm{O}_{\mathrm{R},y,k+1} \\ \psi_{\mathrm{R},k+1} \end{bmatrix} = \begin{bmatrix} v_{\mathrm{R},k} \cos(\psi_{\mathrm{R},k}) + p^\mathrm{O}_{\mathrm{R},x,k} \\ v_{\mathrm{R},k} \sin(\psi_{\mathrm{R},k}) + p^\mathrm{O}_{\mathrm{R},y,k} \\ \omega_{\mathrm{R},k} + \psi_{\mathrm{R},k} \end{bmatrix}, \tag{3.14}$$

onde $v_{\mathrm{R},k}$ e $\omega_{\mathrm{R},k}$ são as velocidades linear e angular do robô no instante de tempo discreto k, $p^{\mathrm{O}}_{\mathrm{R},x,k}$, $p^{\mathrm{O}}_{\mathrm{R},y,k}$ e $\psi_{\mathrm{R},k}$ são as componentes da pose do robô dadas no instante de tempo k, $p^{\mathrm{O}}_{\mathrm{R},x,k+1}$, $p^{\mathrm{O}}_{\mathrm{R},y,k+1}$ e $\psi_{\mathrm{R},k+1}$ são as componentes da pose do robô dadas no instante de tempo $k+1$.

Robôs *skid steering* se movimentam por um conjunto de 4 ou mais rodas, como no exemplo apresentado na Figura 9, ou por meio de esteiras. Este modelo é baseado no controle das velocidades lineares $v_{\mathrm{R},d}$ e $v_{\mathrm{R},e}$ relativas aos lados direito e esquerdo do robô respectivamente, porém, devido ao alinhamento das rodas ao eixo longitudinal do dispositivo o escorregamento esta sempre presente ao realizar giros [Mandow et al., 2007]. Para a determinação do modelo, o sistema de coordenadas do robô é disposto ao centro da área definida pela região de contato no plano, onde o o eixo x está apontado para a frente do robô. Sendo assim, quando o robô gira o seu Centro Instantâneo de Rotação (CIR) pode ser expresso no sistema de coordenadas local pelo ponto $\mathbf{p}_{\mathrm{CIR}} = (x_{\mathrm{CIR}}, y_{\mathrm{CIR}})$. Quando o centro instantâneo de rotação $\mathbf{p}_{\mathrm{CIR}}$ do robô está sobreposto ao eixo y do sistema de coordenadas do robô, ou seja, $x_{\mathrm{CIR}} = 0$, a relação entre as velocidades v_{R} e ω_{R} são expressas da seguinte forma:

$$\begin{bmatrix} v_{\mathrm{R}} \\ \omega_{\mathrm{R}} \end{bmatrix} = \frac{\alpha}{2y_{\mathrm{CIR}}} \begin{bmatrix} y_{\mathrm{CIR}} & y_{\mathrm{CIR}} \\ 1 & -1 \end{bmatrix} \begin{bmatrix} v_{\mathrm{R},d} \\ v_{\mathrm{R},e} \end{bmatrix}, \tag{3.15}$$

onde α é um fator de correção relacionado às características mecânicas de transmissão do movimento para as rodas somados aos efeitos do contato com o terreno.

Portanto, o modelo cinemático *skid steering* para robôs móveis com rodas é dado substituindo a Equação 3.15 na Equação 3.13, resultando em:

$$\begin{bmatrix} \dot{p}^{\mathrm{O}}_{\mathrm{R},x} \\ \dot{p}^{\mathrm{O}}_{\mathrm{R},y} \\ \dot{\psi}_{\mathrm{R}} \end{bmatrix} = \begin{bmatrix} \frac{\cos(\psi_{\mathrm{R}})\ \alpha}{2} & \frac{\cos(\psi_{\mathrm{R}})\ \alpha}{2} \\ \frac{\sin(\psi_{\mathrm{R}})\ \alpha}{2} & \frac{\sin(\psi_{\mathrm{R}})\ \alpha}{2} \\ \frac{\alpha}{2y_{\mathrm{CIR}}} & -\frac{\alpha}{2y_{\mathrm{CIR}}} \end{bmatrix} \begin{bmatrix} v_{\mathrm{R},d} \\ v_{\mathrm{R},e} \end{bmatrix}. \tag{3.16}$$

O modelo completo apresentado requer a otimização dos parâmetros α e y_{CIR}. Contudo, um experimento simples pode ser utilizado para determinar um modelo cinemático simétrico que leva em consideração os escorregamentos das rodas.

Realizando uma rotação pura com o robô a componente y_{CIR} é determinada por:

$$y_{\mathrm{CIR}} = \frac{\int v_{\mathrm{R},d}(t)dt - \int v_{\mathrm{R},e}(t)dt}{2\psi_{max}}, \tag{3.17}$$

no qual ψ_{max} é ângulo total rotacionado.

Realizando um percurso reto com uma distância l_{max}, a componente α é determinada por:

$$\alpha = \frac{2l_{max}}{\int v_{\mathrm{R},d}(t)dt - \int v_{\mathrm{R},e}(t)dt}. \tag{3.18}$$

3.3 Filtros Bayesianos Aplicados à Localização

Um dos problemas na robótica móvel fortemente influenciado pelas incertezas é a determinação da pose do robô. Técnicas de odometrias podem ser utilizadas para estimar a localização de um dispositivo por meio de um modelo cinemático. Contudo, o cálculo de odometria pelo giro das rodas apresenta diversos fatores que geram incertezas na sua medição como o uso de encoders para realizar leituras das velocidades das rodas, modelos matemáticos simplificados, integração, escorregamento das rodas, dentre outros. Portanto, o objetivo desta Seção é descrever os filtros Bayesianos que são comumente utilizados na determinação da localização de um robô em função dos seus estados e sensores.

O filtro de Bayes é uma técnica recursiva utilizada para o cálculo da probabilidade posteriori de um estado por meio da probabilidade priori. A evolução dos estados é governada por leis probabilísticas que comumente são gerados de forma estocástica [Thrun et al., 2005]. A pose $\mathbf{x}_{\mathrm{R},k}$, por exemplo, é um estado do robô no instante de tempo k que pode ser calculado por meio das informações da história dos seus estados, da história dos estados do ambiente medido por meio dos sensores e das entradas de controle atuais. Portanto, a lei probabilística que descreve a evolução do estado $\mathbf{x}_{\mathrm{R},k}$ é dada pela distribuição:

$$\mathbb{P}(\mathbf{x}_{\mathrm{R},k}|\mathbf{x}_{\mathrm{R},0:k-1}, \mathbf{s}_{\mathrm{L},1:k-1}, \mathbf{u}_{1:k}), \tag{3.19}$$

onde $\mathbf{x}_{\mathrm{R},0:k-1}$ são as poses do robô no instante inicial (zero) até o instante de tempo $k-1$, $\mathbf{s}_{\mathrm{L},1:k-1}$ são as medidas do ambiente, assim como as medidas dadas pelo sensor *laser*, coletados no intervalo de tempo de 1 até $k-1$ e $\mathbf{u}_{1:k}$ são as entradas de controle dadas do instante 1 até k, tal como as velocidades, linear v_{R} e angular ω_{R} da Equação 3.13.

O filtro de Kalman é um filtro Bayesiano muito empregado em problemas de rastreamento na robótica, que busca estimar o estado do robô com uma margem de incerteza modelada por uma distribuição gaussiana multivariada com média zero e covariância finita [Choset et al., 2005]. Uma importante característica dessas distribuições é que elas são unimodais e possuem um único valor máximo. Este filtro corresponde a uma técnica iterativa utilizada para predição de sistemas lineares onde os estados $\mathbf{x}_{\mathrm{R}}$ são representados por uma distribuição normal $\mathcal{N}(\boldsymbol{\mu}_{\mathrm{R}}, \boldsymbol{\Sigma}_{\mathrm{R}})$ de média $\boldsymbol{\mu}_{\mathrm{R}}$ e covariância $\boldsymbol{\Sigma}_{\mathrm{R}}$. Estes estados devem ser estimados por um processo de Markov, em que a probabilidade condicional do estado futuro depende somente do estado presente, como mostra a Equação 3.20:

$$\mathbb{P}(\mathbf{x}_{\mathrm{R},k}|\mathbf{x}_{\mathrm{R},1:k-1}, \mathbf{u}_{1:k}) = \mathbb{P}(\mathbf{x}_{\mathrm{R},k}|\mathbf{u}_k, \mathbf{x}_{\mathrm{R},k-1}). \tag{3.20}$$

Portanto, os estados aplicados a um filtro de Kalman devem ser modelados por um sistema linear descrito por:

$$\mathbf{x}_{\mathrm{R},k} = \mathbf{A}_k\mathbf{x}_{\mathrm{R},k-1} + \mathbf{B}_k\mathbf{u}_k + \boldsymbol{\varepsilon}_k, \tag{3.21}$$

$$\mathbf{y}_{\mathrm{L},k} = \mathbf{C}_k\mathbf{x}_{\mathrm{R},k} + \boldsymbol{\delta}_k, \tag{3.22}$$

onde $\mathbf{A}_k$ é uma matriz de transição de estados $\mathbf{x}_{\mathrm{R},k-1}$ do sistema, $\mathbf{B}_k$ é uma matriz que mapeia as entradas de controle $\mathbf{u}_k$, $\mathbf{C}_k$ é uma matriz que relaciona os estados para a saída $\mathbf{y}_{\mathrm{L},k}$, e $\boldsymbol{\varepsilon}_k$ e $\boldsymbol{\delta}_k$ são vetores das variáveis gaussianas que modelam as aleatoriedades na transição dos estados e na medida respectivamente.

O filtro de Kalman é um algoritmo recursivo que pode ser descrito pelas etapas de predição e correção. Na etapa de predição são estimados os estados a *priori* $(\mathbf{x}_{\mathrm{R},k+1|k}, \boldsymbol{\Sigma}_{\mathrm{R},k+1|k})$ considerando somente as entradas de controle e o estado atual $(\mathbf{x}_{\mathrm{R},k|k}, \boldsymbol{\Sigma}_{\mathrm{R},k|k})$, como mostra as Equações 3.23 e 3:

$$\mathbf{x}_{\mathrm{R},k+1|k} = \mathbf{A}_k\mathbf{x}_{\mathrm{R},k|k} + \mathbf{B}_k\mathbf{u}_k, \tag{3.23}$$

$$\boldsymbol{\Sigma}_{\mathrm{R},k+1|k} = \mathbf{A}_k\boldsymbol{\Sigma}_{\mathrm{R},k|k}\mathbf{A}_k^T + \boldsymbol{\Sigma}_{\varepsilon,k}, \tag{3.24}$$

onde $\boldsymbol{\Sigma}_{\varepsilon,k}$ é a matriz de covariâncias referente às variáveis gaussianas que modelam a aleatoriedade na transição do estado.

Na etapa de correção são utilizadas as estimativas a *priori*, calculadas na etapa de predição, combinadas com a observação dos estados $\mathbf{y}_{\mathrm{L},k}$ dada por meio da medida de sensores. Desta forma, a etapa de correção é computada com:

$$\mathbf{x}_{\mathrm{R},k+1|k+1} = \mathbf{x}_{\mathrm{R},k+1|k} + \mathbf{K}_k(\mathbf{y}_{\mathrm{L},k} - \mathbf{C}_k\mathbf{x}_{\mathrm{R},k+1|k}) \tag{3.25}$$

$$\boldsymbol{\Sigma}_{\mathrm{R},k+1|k+1} = \boldsymbol{\Sigma}_{\mathrm{R},k+1|k} - \mathbf{K}_k\mathbf{C}_k\boldsymbol{\Sigma}_{\mathrm{R},k+1|k}, \tag{3.26}$$

onde $\mathbf{K}_k$ é a matriz de ganhos de Kalman definida como:

$$\mathbf{K}_k = \boldsymbol{\Sigma}_{\mathrm{R},k+1|k}\mathbf{C}_k^T(\mathbf{C}_k\boldsymbol{\Sigma}_{\mathrm{R},k+1|k}\mathbf{C}_k^T + \boldsymbol{\Sigma}_{\delta,k})^{-1}, \tag{3.27}$$

no qual $\boldsymbol{\Sigma}_{\delta,k}$ é a matriz de covariâncias das variáveis gaussianas referentes aos modelos de medição.

As considerações sobre a transição linear de estados e medições dadas pelas Equações 3.21 e 3.22 geralmente não são aplicadas a sistemas reais. Por exemplo, as Equações 3.13 e 3.16 são modelos de robôs com rodas descritos por sistemas não-lineares. O filtro de Kalman Estendido (EKF) é uma extensão do filtro de Kalman aplicado para predição de sistemas não-lineares expressos na forma:

$$\mathbf{x}_{\mathrm{R},k} = f(\mathbf{x}_{\mathrm{R},k-1}, \mathbf{u}_k, \boldsymbol{\varepsilon}_k), \tag{3.28}$$

$$\mathbf{y}_{\mathrm{L},k} = g(\mathbf{x}_{\mathrm{R},k}, \boldsymbol{\delta}_k), \tag{3.29}$$

onde a função f é equivalente à Equação 3.21, mapeando a transição de estados em função dos estados passados $\mathbf{x}_{\mathrm{R},k-1}$, das entradas de controle $\mathbf{u}_k$ somados aos ruídos $\boldsymbol{\varepsilon}_k$, a função g é equivalente à Equação 3.22 que mapeia os estados $\mathbf{x}_{\mathrm{R},k}$ para a saída $\mathbf{y}_{\mathrm{L},k}$ acrescido das variáveis aleatórias $\boldsymbol{\delta}_k$.

Considerando as não-linearidades, os sistemas compostos pelas funções f e g não possuem aleatoriedades dadas por uma distribuição gaussiana. Desta forma, o EKF calcula

a crença representada por uma aproximação da distribuição gaussiana. A ideia central está na linearização das funções f e g em torno da média $x_{\mathrm{R},k}$ utilizando a expansão por série de Taylor de primeira ordem. Sendo assim, pode ser assumido um gaussiano para esta função linearizada e utilizar a solução apresentada pelo filtro de Kalman. As linearizações das funções f e g são calculadas pelas derivadas parciais dadas pelas Jacobianas:

$$\mathbf{F}_k = \left.\frac{\partial f(\mathbf{x}_\mathrm{R}, \mathbf{u}_k)}{\partial \mathbf{x}_\mathrm{R}}\right|_{\mathbf{x}_\mathrm{R}=\mathbf{x}_{\mathrm{R},k}}, \tag{3.30}$$

$$\mathbf{G}_k = \left.\frac{\partial g(\mathbf{x}_\mathrm{R})}{\partial \mathbf{x}_\mathrm{R}}\right|_{\mathbf{x}_\mathrm{R}=\mathbf{x}_{\mathrm{R},k}}. \tag{3.31}$$

As matrizes $\mathbf{F}_k$ e $\mathbf{G}_k$ são análogas às matrizes $\mathbf{A}_k$ e $\mathbf{C}_k$ apresentadas nas Equações 3.21 e 3.22. Sendo assim, a estrutura do filtro de Kalman pode ser utilizada para estimar os estados $\mathbf{x}_{\mathrm{R},k}$ descritos por sistemas não-lineares da forma da Equação 3.28 e 3.29, como mostra o Algoritmo 1.

Algoritmo 1: Filtro de Kalman Estendido

Entrada : $\mathbf{x}_{\mathrm{R},k|k}$, $\mathbf{\Sigma}_{\mathrm{R},k|k}$, $\mathbf{u}_k$ e $\mathbf{y}_{\mathrm{L},k}$
Saída : $\mathbf{x}_{\mathrm{R},k+1|k+1}$, $\mathbf{\Sigma}_{\mathrm{R},k+1|k+1}$

Predição:
$\mathbf{x}_{\mathrm{R},k+1|k} = f(\mathbf{x}_{\mathrm{R},k|k}, \mathbf{u}_k)$
$\mathbf{\Sigma}_{\mathrm{R},k+1|k} = \mathbf{F}_k \mathbf{\Sigma}_{\mathrm{R},k|k} \mathbf{F}_k^T + \mathbf{\Sigma}_{\varepsilon,k}$
Correção:
$\mathbf{K}_k = \mathbf{\Sigma}_{\mathrm{R},k+1|k} \mathbf{G}_k^T (\mathbf{G}_k \mathbf{\Sigma}_{\mathrm{R},k+1|k} \mathbf{G}_k^T + \mathbf{\Sigma}_{\delta,k})^{-1}$
$\mathbf{x}_{\mathrm{R},k+1|k+1} = \mathbf{x}_{\mathrm{R},k+1|k} + \mathbf{K}_k (\mathbf{y}_{\mathrm{L},k} - g(\mathbf{x}_{\mathrm{R},k+1|k}))$
$\mathbf{\Sigma}_{\mathrm{R},k+1|k+1} = \mathbf{\Sigma}_{\mathrm{R},k+1|k} - \mathbf{K}_k \mathbf{G}_k \mathbf{\Sigma}_{\mathrm{R},k+1|k}$
return $\mathbf{x}_{\mathrm{R},k+1|k+1}$, $\mathbf{\Sigma}_{\mathrm{R},k+1|k+1}$

Outro filtro aplicado a sistemas não-lineares é o filtro de Kalman Unscented (UKF), proposto originalmente por Julier and Uhlmann [1997] como uma alternativa com melhor desempenho que o EKF. A diferença básica entre o EKF e o UKF está na forma de representar a distribuição das variáveis aleatórias. O EKF utiliza uma aproximação de uma variável gaussiana, no qual a propagação do estado ocorre de forma analítica por uma aproximação de primeira ordem do sistema, como descrito anteriormente. Já no UKF, a variável aleatória é representada por um conjunto de amostras selecionadas por meio de uma amostragem determinística, e a média e covariância a *posteriori* são determinadas pela segunda ordem da série de Taylor para qualquer sistema não-linear.

3.4 Filtros Probabilísticos Aplicados à Localização e Mapeamento Simultâneos

De maneira resumida, o SLAM busca construir um mapa do ambiente e localizar o dispositivo neste mapa de forma simultânea. Basicamente o problema pode ser separado em 3 etapas segundo Ben-Ari and Mondada [2017]. Na primeira etapa, que compõe o *Front-End*, é estimada uma nova pose do robô por odometria e é gerado um mapa local por meio da leitura dos sensores. Já na segunda etapa, é realizado o alinhamento entre o mapa local e global, e computanda a pose do dispositivo com respeito ao sistema de referência inercial. E na terceira etapa, a pose do robô é corrigida e o mapa global é atualizado incorporando o mapa local. Estas duas últimas etapas compõem o *Back-End*. Segundo Xu et al. [2018] o SLAM ainda possui uma última parte composta pelas técnicas de *loop closure*. Na Figura 10 pode ser visto o diagrama básico do SLAM com as relações entre as entradas e saídas.

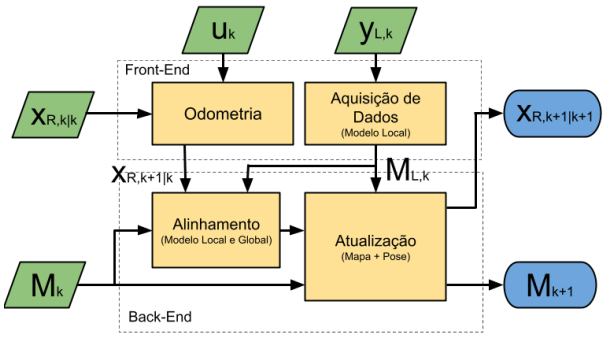


Figura 10 – Diagrama básico de funcionamento de uma estratégia de SLAM.

O processo do SLAM é iterativo e inicia a partir de uma localização conhecida do robô, podendo ou não utilizar um mapa pré-definido. A partir da pose $\mathbf{x}_{\mathrm{R},k|k}$ estimada no instante k, o robô evolui para o estado a *priori* $\mathbf{x}_{\mathrm{R},k+1|k}$ estimado por uma odometria, que pode ser obtida por meio das medidas dos *encoders* das rodas, da sequência de sinais de controle, varreduras consecutivas do *laser* e das combinações de diferentes métodos. A cada iteração uma correlação é estabelecida entre os recursos do mapa $\mathbf{M}_k$ identificados no ambiente para determinar a pose do robô $\mathbf{x}_{\mathrm{R},k+1|k+1}$. Após determinada a pose de melhor alinhamento entre o mapa local $\mathbf{M}_{\mathrm{L},k}$ e o mapa $\mathbf{M}_k$, estes são fundidos para formar o mapa $\mathbf{M}_{k+1}$.

Na literatura são distinguidos dois tipos de SLAM, o *full* SLAM e o SLAM *online* [Siciliano and Khatib, 2016]. O primeiro determina o problema do SLAM por meio de toda sequência de medições do caminho percorrido pelo robô e os dados do mapa, no qual a estimativa a *posteriori* é dada por $\mathbb{P}(\mathbf{x}_{\mathrm{R},0:k}, \mathbf{M}|\mathbf{s}_{\mathrm{L},1:k}, \mathbf{u}_{1:k})$. Já o segundo trata os dados das leituras atuais dadas por $\mathbb{P}(\mathbf{x}_{\mathrm{R},k}, \mathbf{M}|\mathbf{s}_{\mathrm{L},k}, \mathbf{u}_k)$.

Os algoritmos que descrevem o SLAM *online* são comumente chamados de filtros como os apresentados na Seção anterior. As principais abordagens para a solução do SLAM apresentadas nessa Seção são o EKF SLAM que calcula as estimativas sobre a localização e o mapa de forma conjunta, e o *Graph-Based* SLAM que é uma técnica baseada em grafos [Bailey and Durrant-Whyte, 2006].

EKF SLAM

A abordagem do EKF SLAM utiliza o modelo cinemático do dispositivo robótico como função de estado e, dadas as entradas de controle e os recursos do mapa identificados no ambiente, estima tanto a pose do robô quanto a posição dos novos recursos [Choset et al., 2005]. Para isso, o vetor de estados é definido de forma aumentada, composto pelo estado robô mais as posições dos pontos de referências, como mostra a Equação 3.32:

$$\mathbf{x}_{\mathrm{R}a,k} = \begin{bmatrix} \mathbf{x}_{\mathrm{R},k} \\ \mathbf{s}_{\mathrm{L},k}^{(1)} \\ \vdots \\ \mathbf{s}_{\mathrm{L},k}^{(m)} \end{bmatrix}, \tag{3.32}$$

onde $\mathbf{x}_{\mathrm{R}a,k}$ é o vetor de estados aumentado, $\mathbf{x}_{\mathrm{R},k}$ é vetor da posição do robô, e $\mathbf{s}_{\mathrm{L},k}^{(1)}$ a $\mathbf{s}_{\mathrm{L},k}^{(m)}$ são os vetores das posições dos pontos de referência observados. Desta forma, o modelo geral de evolução de estados aumentados é dado por:

$$\mathbf{x}_{\mathrm{R}a,k} = f_a(\mathbf{x}_{\mathrm{R}a,k-1}, \mathbf{u}_{k-1}, \boldsymbol{\varepsilon}_{k-1}) \tag{3.33}$$

$$\begin{bmatrix} \mathbf{x}_{\mathrm{R},k} \\ \mathbf{s}_{\mathrm{L},k}^{(1)} \\ \vdots \\ \mathbf{s}_{\mathrm{L},k}^{(m)} \end{bmatrix} = \begin{bmatrix} f(\mathbf{x}_{\mathrm{R},k-1}, \mathbf{u}_{k-1}, \boldsymbol{\varepsilon}_{k-1}) \\ \mathbf{s}_{\mathrm{L},k-1}^{(1)} \\ \vdots \\ \mathbf{s}_{\mathrm{L},k-1}^{(m)} \end{bmatrix}. \tag{3.34}$$

A saída do sistema de equações de estado do EKF SLAM, $\mathbf{y}_{\mathrm{R}a,k}$, é dada pelas posições dos pontos de referências $\mathbf{s}_{\mathrm{L}}^{(i)}$ com respeito à posição $\mathbf{p}_{\mathrm{R}}$ do dispositivo robótico, somados ao ruído Gaussiano $\boldsymbol{\delta}_{k-1}^{(i)}$. Desta forma, a função da medida do estado para o i-ésimo ponto de referência é expresso como:

$$\mathbf{y}_{\mathrm{L}a,k}^{(i)} = g_a^{(i)}(\mathbf{p}_{\mathrm{R}}, \mathbf{s}_{\mathrm{L}}^{(i)}) + \boldsymbol{\delta}_{k-1}^{(i)}, \tag{3.35}$$

onde $g_a^{(i)}(\mathbf{p}_{\mathrm{R}}, \mathbf{s}_{\mathrm{L}}^{(i)})$ é dada por:

$$g_a^{(i)}(\mathbf{p}_{\mathrm{R}}, \mathbf{s}_{\mathrm{L}}^{(i)}) = \begin{bmatrix} \sqrt{(p_{\mathrm{R},x} - s_{\mathrm{L},x}^{(i)})^2 + (p_{\mathrm{R},y} - s_{\mathrm{L},y}^{(i)})^2} \\ \mathrm{atan2}((p_{\mathrm{R},y} - s_{\mathrm{L},y}^{(i)}), (p_{\mathrm{R},x} - s_{\mathrm{L},x}^{(i)})) - \psi_{\mathrm{R}} \end{bmatrix}. \tag{3.36}$$

Portanto, dada uma distribuição Gaussiana inicial $\mathcal{N}(\mathbf{x}_{\mathrm{R}a,0}, \boldsymbol{\Sigma}_{\mathrm{R}a,0})$ para o estado aumentado, as entradas de controle $\mathbf{u}_k$ e as medidas do ambiente, o filtro EKF pode ser aplicado de forma iterativa para estimação de forma simultânea tanto da pose do robô quanto da posição dos pontos de referência $\mathbf{s}_{\mathrm{L}}$ observados.

Graph-Based SLAM

As técnicas de SLAM baseado em grafos, chamadas de *Graph-Based* SLAM ou somente Graph-SLAM, constroem um problema de estimação simples, abstraindo as informações cruas dos sensores, buscando resolver o problema da localização e mapeamento simultâneos utilizando otimização não linear esparsa [Grisetti et al., 2010]. Esse tipo de abordagem é focada na solução completa do SLAM, denominada de *Full* SLAM.

Os grafos são construídos representando os nós tanto pelos pontos de referência identificados do ambiente quanto as localizações do robô estimadas. Sendo assim, dois tipos de conexões são realizadas: uma entre pares de poses estimadas $\mathbf{x}_{\mathrm{R},k-1}$ e $\mathbf{x}_{\mathrm{R},k}$ dadas pelas entradas de controle $\mathbf{u}_{k-1}$, e outra dada pela observação dos pontos de referências $\mathbf{s}_{\mathrm{L},k}$ a partir da pose do robô $\mathbf{x}_{\mathrm{R},k}$. As bordas do grafo possuem restrições suaves, no qual o relaxamento dessas causam uma melhor estimativa para o mapa, considerando todo o caminho $\mathbf{x}_{\mathrm{R},0:k}$ [Siciliano and Khatib, 2016].

A Figura 11 exemplifica de forma simples a construção de um grafo enquanto o robô se desloca. O incremento desse grafo é expresso basicamente por duas etapas. No instante $k-1$, a partir da pose $\mathbf{x}_{\mathrm{R},k-1}$, são observados os pontos de referência $\mathbf{s}_{\mathrm{L}}^{(1:m)}$ adicionados ao grafo. Então, os arcos (ligações) entre a pose do robô e os pontos observados, que são dados por distribuições de probabilidades relativas entre os nós como restrições, são armazenados em uma matriz. Uma nova movimentação do robô é computada conforme as entradas de controle $\mathbf{u}_k$, quando um novo arco é adicionado no grafo entre as posições $\mathbf{x}_{\mathrm{R},k-1}$ e $\mathbf{x}_{\mathrm{R},k}$. O procedimento é repetido indefinidamente, realizando o incremento do mapa.

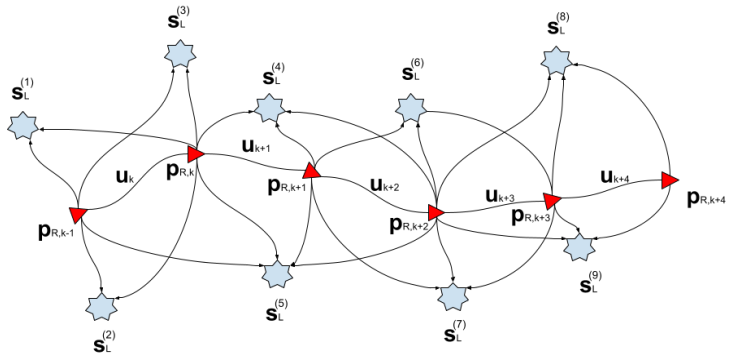


Figura 11 – Ilustração do funcionamento da técnica de Graph-SLAM.

3.5 Princípios de Funcionamento do LiDAR SLAM

Técnicas de LiDAR SLAM são abordagens para solução do problema do SLAM que utilizam as medidas de sensores lasers tanto para estimar a odometria por *ego-movimentação* entre duas varreduras do sensor, quanto para construir o modelo do ambiente por meio da

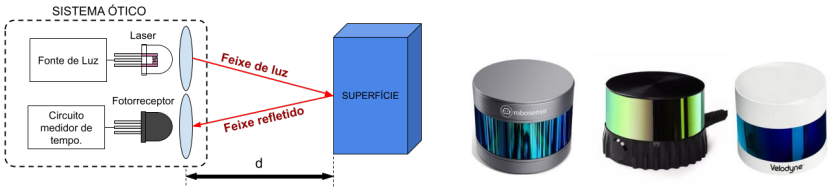
fusão dos pontos detectados [Shan and Englot, 2018]. Divido em *Front-End* e *Back-End*, as técnicas de LiDAR SLAM possuem subdivisões que formam o *pipeline* do SLAM onde são aplicadas diferentes técnicas. Portanto, esta Seção tem por finalidade a apresentação destas técnicas, descrevendo: os sistemas de aquisição de dados apresentando as diferentes tecnologias dos sensores a *laser* e a apresentação dos dados; as características extraídas do ambiente e as principais técnicas aplicadas; as abordagens para estimação da pose do dispositivo robótico dado entre duas varreduras do sensor; e por último algumas das formas de mapeamento comumente e adotadas por técnicas de LiDAR SLAM.

3.5.1 Sistema de Aquisição de Dados

A base de um sistema de SLAM está na aquisição de informações do ambiente por meio dos sensores que são utilizados para identificar características do meio, mapear o espaço e localizar o dispositivo. As técnicas de LiDAR SLAM utilizam os dados de sensores *lasers* na forma de nuvem de pontos, pré-processada por um sistema de aquisição de dados [Wong et al., 2011]. A seguir é descrita a categoria de sensor *laser* classificado como scanning LiDAR 3D, e as transformações realizadas para converter as informações capturadas dos ambientes em uma nuvem de pontos representativa.

Sensores *Scanning* LiDAR 3D

Sensores LiDAR, também conhecido como LADAR (*Laser Radar*), são dispositivos que capturam as características do ambiente por meio da detecção de feixes de luz medindo o tempo de voo da luz. Essa tecnologia é chamada de ToF (*Time of Flight*) e mede o tempo que um pulso de luz, emitido na forma de um feixe estreito, gasta para ser refletido por uma superfície e retornar. A distância percorrida pode ser calculada por: $d = \frac{1}{2}c\Delta t$, onde d é a distância do objeto, c é a velocidade média da luz e Δt é o tempo entre a emissão do feixe e a sua recepção pelo sensor. Um exemplo ilustrativo do sistema de ótico de um sensor ToF pode ser visto na Figura12(a).



(a) Sistema Ótico (b) Sistema Ótico

Figura 12 – (a) Ilustração do sistema ótico ilustrativo de um sensor ToF, e (b) Modelos de sensors LiDAR 3D da RoboSense RS-LiDAR-16, Ouster OS2, Velodyne Puck 32MR.

Os sensores *Scanning* LiDAR 3D escaneiam múltiplos pontos do ambiente. Devido a sua capacidade de gerar nuvens densas, estes são muito utilizados na robótica para

gerar modelos tridimensionais aplicando técnicas de SLAM 3D, como as estudadas nesta dissertação. Estes sensores possuem uma distribuição de *lasers* na vertical que varia de 16 a 128 feixes, como em modelos mais recentes como o Ouster OS2 e o Velodyne VLS-128, e realizam uma varredura de 360° com uma resolução de aproximadamente $0,1°$ com um alcance de 70 a 300. Na Figura 12(b) são apresentados três sensores *Scanning* LiDAR 3D: RoboSense RS-LiDAR-16, Ouster OS2 e Velodyne Puck 32MR.

Apresentação dos Dados 3D

Os dados coletados pelos sensores LiDAR 3D são apresentados em diferentes formatos que dependem do fabricante. Existem dois formatos padrões para representação das informações destes sensores que são imagens 3D representadas por $\mathbf{s}_{\mathrm{I}} = [i \ j \ d(i,j)]^T$ onde $d(i,j)$ é a distância de um ponto expresso nas coordenadas do *pixel* (i,j), e em coordenadas cartesianas dadas por $\mathbf{s}_{\mathrm{L}} = [s_{\mathrm{L},x} \ s_{\mathrm{L},y} \ s_{\mathrm{L},z}]^T$. Portanto, o conjunto de pontos de uma varredura de um sensor LiDAR pode ser organizado em uma lista denominada nuvem de pontos, descrita por:

$$\mathbf{s}_{\mathrm{L}}^{(1:m)} = \{\mathbf{s}_{\mathrm{L}}^{(1)}, \cdots, \mathbf{s}_{\mathrm{L}}^{(m)}\}, \tag{3.37}$$

onde m é igual a quantidades de pontos detectados.

As informações 3D coletadas dos ambientes pelos sensores LiDAR são mapeadas para o sistema de coordenadas do sensor segundo a geometria do dispositivo. Um sensor *scanning* LiDAR geralmente possui seu sistema de coordenadas com o eixo x apontando para a frente do sensor e o eixo z para cima, onde os dados coletados do ambiente são expressos em coordenadas esféricas $(\psi_{\mathrm{L}}, \theta_{\mathrm{L}}, d)$, como ilustra a Figura 13. Os pontos detectados por esse tipo de sensor podem ser convertidos como uma imagem de profundidade representando os ângulos ψ_{L} e θ_{L} como coordenadas longitudinais e latitudinais, e em coordenadas cartesianas dadas por:

$$\begin{bmatrix} s_{\mathrm{L},x} \\ s_{\mathrm{L},y} \\ s_{\mathrm{L},z} \end{bmatrix} = \begin{bmatrix} d\cos\psi_{\mathrm{L}}\cos\theta_{\mathrm{L}} \\ d\sin\psi_{\mathrm{L}}\cos\theta_{\mathrm{L}} \\ d\sin\theta_{\mathrm{L}} \end{bmatrix}. \tag{3.38}$$

3.5.2 Extração de Features

Os pontos detectados por um sensor LiDAR informam a dimensão do espaço mapeado a partir das distâncias mensuradas. Para que estas informações possam ser utilizadas na solução do SLAM, certas características do ambiente (*features*) devem ser extraídas. As abordagens de LiDAR SLAM comumente utilizam como *features* pontos 3D, planos, linhas 3D e segmentos de reta.

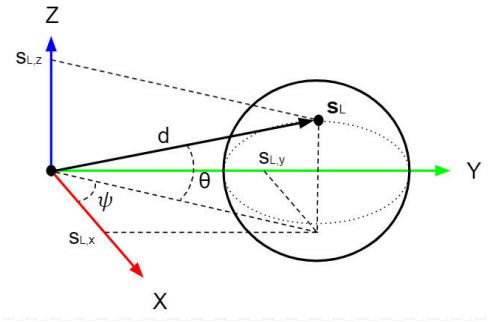


Figura 13 – Mapeamento esférico para sensor LiDAR 3D. Adaptado de [Siciliano et al., 2010].

Pontos 3D

As features representadas por pontos 3D descrevem saliências como quinas de portas, encontros entre duas ou três superfícies e protuberâncias, assim como podem ser usados também para identificar superfícies planas como pisos, paredes, teto e objetos. Uma metodologia aplicada à identificação destes pontos são os *Point Features Histograms* (PFH) que os classificam segundo as características geométricas da sua vizinhança [Rusu et al., 2008]. Desta forma os pontos pertencentes à mesma superfície podem ser agrupados. Uma segunda abordagem é a metodologia *Fast* Eigen-*Curvature Scale Space* (Eigne-CSS) apresentada em [Mokhtarian, 1995, Stiene et al., 2006], que identifica os contornos dos objetos detectados em uma imagem de profundidade interpretando os contornos como curvas.

Outra solução para a classificação de pontos em bordas ou planos é encontrada em [Zhang and Singh, 2017], que utilizam uma Equação de suavidade normalizada, definida como:

$$s = \frac{1}{|\mathrm{S_L}| \ \| \ \mathbf{s}^{(i)}_{\mathrm{L},k} \ \|} \ \| \sum_{j \in \mathrm{S_L}, j \neq i} \left(\mathbf{s}^{(i)}_{\mathrm{L},k} - \mathbf{s}^{(j)}_{\mathrm{L},k} \right) \|, \tag{3.39}$$

no qual $\mathbf{s}^{(i)}_{\mathrm{L},k}$ é o vetor da coordenada 3D de um ponto i na varredura k do sensor LiDAR, $\mathbf{s}^{(j)}_{\mathrm{L},k}$ é o vetor de um ponto j da varredura k e, $\mathrm{S_L}$ é o conjunto de pontos nas proximidades do ponto i. Desta forma, os pontos são classificados como bordas quando o valor s é maior que um limiar máximo, ou pertencentes a um plano quando o valor de s é menor que um limiar mínimo.

Planos

Uma superfície planar pode ser representada por uma Equação matemática de um plano infinito, dada por $ax + by + cz + d = 0$, incluindo a descrição do seu limite. As margens que definem uma região plana podem ser representadas por uma lista de pontos ou um conjunto de segmentos de retas. A extração desse recurso de uma nuvem de pontos é um problema de modelagem no qual se determina um conjunto de planos

que descrevem os pontos. Uma solução possível é o algoritmo RANSAC (*Random Sample Consensum*) adaptado para detecção de planos, proposto inicialmente por Fischler and Bolles [1981]. Os métodos padrões para detecção de planos são os baseados em Região *Growing* como apresentado em [Wan and Higgins, 2003]. Essa metodologia identifica uma região plana ao redor de um ponto 3D analisando a sua vizinhança utilizando o método de mínimos quadrados. Outras metodologias também são aplicadas, como as baseadas na adaptação da técnica *Hough Transform* (HT) para a identificação de planos, como descrito em [Borrmann et al., 2011].

Linhas 3D e Segmentos de Retas

As linhas 3D e segmentos de retas são comumente encontrados em diversos ambientes como quinas de portas e janelas, encontro de paredes com pisos e bordas de objetos. As linhas 3D são fáceis de se detectar utilizando um sensor LiDAR e são comumente representadas por um conjunto de pontos dados por $\mathbf{b} = \mathbf{s}_{\mathrm{L}} + \zeta\mathbf{c}$, onde $\mathbf{b}$ é equação da reta, $\mathbf{s}_{\mathrm{L}}$ é um ponto sobre a linha, $\mathbf{c}$ é um vetor unitário que dá a direção e ζ é um escalar [Gokturk et al., 2004]. Esse tipo de *feature* geralmente é extraída de forma indireta, como exemplo, após a extração de superfícies planares estando presentes nas interseções dos planos.

3.5.3 Estimação da Pose

A estimação da pose corresponde ao cálculo da posição e orientação de um corpo com respeito a um sistema de coordenadas inercial. Nas técnicas de LiDAR SLAM, a pose é calculada pelo alinhamento entre duas nuvens de pontos escaneadas pelo sensor ou entre uma nuvem de pontos e o mapa. Três soluções possíveis são a estimação da pose por um conjunto de pontos, por planos e por linhas 3D, ou por segmentos de reta. As três são resolvidas minimizando uma função de custo onde as *features* atuam como restrições.

Estimação da Pose Por um Conjunto de Pontos

Uma solução para estimação da pose dada por um conjunto de pontos é o algoritmo *Iteractive Closest Points* (ICP) [Besl and McKay, 1992]. Esse método é iterativo e usa uma distância máxima d_{max} para analisar os pontos mais próximos, determinando um conjunto de pontos parcialmente sobrepostos. Dados dois conjuntos de pontos 3D independentes a pose relativa $\mathbf{x}_{\mathrm{L}} = (\mathbf{p}_{\mathrm{L}}, \mathbf{R}_{\mathrm{L}})$ é determinada minimizando a função de custo:

$$e(\mathbf{p}_{\mathrm{L}}, \mathbf{R}_{\mathrm{L}}) = \sum_{j} \|\mathbf{s}^{(j)}_{\mathrm{L},k-1} - (\mathbf{R}_{\mathrm{L}}\ \mathbf{s}^{(j)}_{\mathrm{L},k} - \mathbf{p}_{\mathrm{L}})\|^2_{\mathbf{N}}, \tag{3.40}$$

no qual $e(\mathbf{p}_{\mathrm{L}}, \mathbf{R}_{\mathrm{L}})$ equivale ao erro a ser minimizado, $\mathbf{s}^{(j)}_{\mathrm{L},k}$ é a coordenada o ponto j para a varredura k, e $\mathbf{N}$ é uma matriz positiva definida de ganhos.

Algoritmo 2: *Iteractive Closest Points* - ICP

Entrada : $\mathbf{s}_{\mathrm{L},k}^{(1:m)}$ e $\mathbf{s}_{\mathrm{L},k-1}^{(1:m)}$
Saída : $\mathbf{x}_{\mathrm{L}} = (\mathbf{p}_{\mathrm{L}}, \mathbf{R}_{\mathrm{L}})$

for $i = 0$ *até* N **do**
  $N = \mathrm{n^o}$ máximo de iterações.
  for $\mathbf{s}_{\mathrm{L},k}^{(j)} \in \mathbf{s}_{\mathrm{L},k}^{(1:m)}$ **do**
    Encontrar o ponto mais próximo a um alcance d_{max} no conjunto $\mathbf{s}_{\mathrm{L},k-1}^{(1:m)}$ para o ponto $\mathbf{s}_{\mathrm{L},k}$.
  end
  Calcula a transformação $(\mathbf{p}_{\mathrm{L}}, \mathbf{R}_{\mathrm{L}})$ que minimiza o erro e da Equação 3.40.
  Aplica a transformação encontra no conjunto $\mathbf{s}_{\mathrm{L},k}^{(1:m)}$.
  Calcula a diferença do erro quadrático.
  $\|e_{i-1}(\mathbf{p}_{\mathrm{L}}, \mathbf{R}_{\mathrm{L}}) - e_i(\mathbf{p}_{\mathrm{L}}, \mathbf{R}_{\mathrm{L}})\|$.
  Se a diferença for menor que um limiar e_{min} inferior finaliza.
end

O Algorítimo 2 descreve de forma geral a metodologia do ICP.

As entradas do Algoritmo 2 são dois conjuntos de pontos que, para o caso da estimação entre dois escaneamentos, são os mapas locais gerados por varreduras consecutivas $\mathbf{s}_{\mathrm{L},k-1}^{(1:m)}$ e $\mathbf{s}_{\mathrm{L},k}^{(1:m)}$. A saída é a pose estimada entre os dois conjuntos, dada por uma matriz de rotação $\mathbf{R}_{\mathrm{L}}$ e um vetor de posição $\mathbf{p}_{\mathrm{L}}$. Para cada ponto $\mathbf{s}_{\mathrm{L},k}^{(j)}$ pertencente ao conjunto $\mathrm{S}_{\mathrm{L},k}$ o algoritmo encontra o ponto mais próximo dentre os pontos da sua vizinhança rastreados dentro de um raio d_{max}. Na sequência é calculada a transformação $(\mathbf{p}_{\mathrm{L}}, \mathbf{R}_{\mathrm{L}})$ que minimiza o erro dado pela função de custo conforme a Equação 3.40. Para isso as médias $\boldsymbol{\mu}_{\mathrm{L},k-1}$ e $\boldsymbol{\mu}_{\mathrm{L},k}$ são determinadas para cada conjunto. Sendo assim, são construídas então duas matrizes:

$$\mathbf{Q} = \left[\{\mathbf{s}_{\mathrm{L},k-1}^{(1)} - \boldsymbol{\mu}_{\mathrm{L},k-1}\} \quad \cdots \quad \{\mathbf{s}_{\mathrm{L},k-1}^{(m)} - \boldsymbol{\mu}_{\mathrm{L},k-1}\}\right] \tag{3.41}$$

$$\mathbf{D} = \left[\{\mathbf{s}_{\mathrm{L},k}^{(1)} - \boldsymbol{\mu}_{\mathrm{L},k}\} \quad \cdots \quad \{\mathbf{s}_{\mathrm{L},k}^{(m)} - \boldsymbol{\mu}_{\mathrm{L},k}\}\right], \tag{3.42}$$

no qual $\mathbf{Q}$ e $\mathbf{D}$ são formados pelos vetores dados pela diferença entre os pontos de cada varredura e sua respectiva média, gerando um conjunto centralizado. Desta forma é possível calcular a matriz de rotação $\mathbf{R}_{\mathrm{L}}$ por meio da decomposição de valores singulares dado por:

$$svd(\mathbf{D}\mathbf{Q}^T) = \mathbf{U}\mathbf{E}\mathbf{V}^*, \tag{3.43}$$

no qual $\mathbf{U}$ é uma matriz unitária, $\mathbf{E}$ é uma matriz diagonal retangular e $\mathbf{V}^*$ é a matriz conjugada transposta da matriz unitária $\mathbf{V}$. Portanto, a matriz de rotação $\mathbf{R}$ é calculada da seguinte forma:

$$\mathbf{R}_{\mathrm{L}} = \mathbf{U}\mathbf{V}^*. \tag{3.44}$$

Sendo assim, a translação $\mathbf{p}_{\mathrm{L}}$ pode ser calculada usando os valores das médias de

cada conjunto $\boldsymbol{\mu}_{\mathrm{L},k-1}$ e $\boldsymbol{\mu}_{\mathrm{L},k}$ com a matriz de rotação $\mathbf{R}_{\mathrm{L}}$ dada pela Equação 3.44, como:

$$\mathbf{p}_{\mathrm{L}} = \boldsymbol{\mu}_{\mathrm{L},k} - \mathbf{R}\boldsymbol{\mu}_{\mathrm{L},k-1}. \tag{3.45}$$

Vários métodos de mínimos quadrados podem ser aplicados para resolver o problema de otimização da Equação 3.40, como Levenberg-Marquadt, métodos de Gradientes Conjugados, e Gauss-Newton [Shirangi and Emerick, 2016].

Estimada a transformação $(\mathbf{p}_{\mathrm{L}}, \mathbf{R}_{\mathrm{L}})$, está é aplicada ao conjunto $\mathbf{s}_{\mathrm{L},k}^{(1:m)}$ e então é calculado o erro quadrático:

$$\|e_{k-1}(\mathbf{p}_{\mathrm{L}}, \mathbf{R}_{\mathrm{L}}) - e_k(\mathbf{p}_{\mathrm{L}}, \mathbf{R}_{\mathrm{L}})\| < e_{min}. \tag{3.46}$$

Quando essa diferença é menor que um certo limiar inferior pré-determinado, o Algoritmo 2 finaliza.

Outra abordagem para a estimação da pose por conjunto de pontos é o *Normal Distribution Transform* (NDT) por *scan matching*, onde os pontos escaneados pelo sensor LiDAR são modelados como uma distribuição gaussiana normal. Desenvolvida inicialmente para nuvens de pontos bidimensionais em [Biber and Straßer, 2003] e posteriormente adaptada para dados tridimensionais em [Magnusson et al., 2009], a técnica NDT *scan matching* possui eficácia melhor que o ICP tradicional.

A primeira etapa do algoritmo é a construção de um modelo NDT para descrever os pontos. O espaço então é divido em *voxels*, onde é calculada a distribuição normal D-dimensional dado pela média $\boldsymbol{\mu}_{\mathrm{L}}^{cj}$ e covariância $\boldsymbol{\Sigma}_{\mathrm{L}}^{cj}$ para célula cj, como:

$$\mathcal{N}(\boldsymbol{\mu}_{\mathrm{L}}, \boldsymbol{\Sigma}_{\mathrm{L}}) = \frac{1}{(2\pi)^{\frac{D}{2}}\sqrt{|\boldsymbol{\Sigma}_{\mathrm{L}}|}} \exp\left\{-\frac{1}{2}(\mathbf{s}_{\mathrm{L}} - \boldsymbol{\mu}_{\mathrm{L}})^T \boldsymbol{\Sigma}_{\mathrm{L}}^{-1}(\mathbf{s}_{\mathrm{L}} - \boldsymbol{\mu}_{\mathrm{L}})\right\}. \tag{3.47}$$

A segunda etapa é a determinação da pose entre duas nuvens de pontos, maximizando a função de custo:

$$e(\mathbf{p}_{\mathrm{L}}, \mathbf{R}_{\mathrm{L}}) = \sum_j \exp\left\{[\boldsymbol{\mu}_{\mathrm{L},k-1}^{cj} - (\mathbf{R}_{\mathrm{L}}\ \boldsymbol{\mu}_{\mathrm{L},k}^{cj} - \mathbf{p}_{\mathrm{L}})]^T (\boldsymbol{\Sigma}_{\mathrm{L},k}^{cj})^{-1} [\boldsymbol{\mu}_{\mathrm{L},k-1}^{cj} - (\mathbf{R}_{\mathrm{L}}\ \boldsymbol{\mu}_{\mathrm{L},k}^{cj} - \mathbf{p}_{\mathrm{L}})]\right\}. \tag{3.48}$$

Assim como apresentado anteriormente, diversos métodos de mínimos quadrados podem ser utilizado para maximizar a Equação 3.48.

Estimação da Pose Por Planos

Quando as *features* extraídas de um sensor LiDAR são planos a estimação da pose se dá pelo alinhamento dos vetores normais associados a N_n de pares de planos identificados de duas varreduras consecutivas. Desta forma, os planos da varredura $k-1$ são descritos por um conjunto de vetores $\mathbf{n}_{\mathrm{L},k-1}^{(i)}$ para $i = 1 \cdots N_n$, e por um conjunto de

pontos $\mathbf{s}_{\mathrm{L},k-1}^{(i)}$. De forma análoga os planos da varredura k são descritos por $\mathbf{n}_{\mathrm{L},k}^{(i)}$ e $\mathbf{s}_{\mathrm{L},k}^{(i)}$. Sendo assim, a matriz de rotação entre os conjuntos de planos é dada pela minimização da função de custo:

$$e(\mathbf{R}_{\mathrm{L}}) = \sum_i \|\mathbf{R}_{\mathrm{L}}\mathbf{n}_{\mathrm{L},k-1}^{(i)} - \mathbf{n}_{\mathrm{L},k}^{(i)}\|^2. \tag{3.49}$$

Da mesma forma que na estimação por conjunto de pontos são construídas duas matrizes de dimensões $3 \times N_n$, $\mathbf{Q}$ e $\mathbf{D}$, formadas pelos vetores $\mathbf{n}_{\mathrm{L},k-1}^{(i)}$ e $\mathbf{n}_{\mathrm{L},k}^{(i)}$ respectivamente. A matriz de rotação $\mathbf{R}_{\mathrm{L}}$ é então calculada pela decomposição de valores singulares de forma análoga ao apresentado nas Equações 3.43 e 3.44.

A translação entre os dois conjuntos de planos é calculada minimizando a soma do quadrado das distâncias entre os pontos $\mathbf{s}_{\mathrm{L},k-1}^{(i)}$ rotacionados por $\mathbf{R}_{\mathrm{L}}$ e os planos correspondentes definidos pelos vetores normais $\mathbf{n}_{\mathrm{L},k}^{(i)}$ e pelos pontos $\mathbf{s}_{\mathrm{L},k}^{(i)}$. Portanto, a translação $\mathbf{p}_{\mathrm{L}}$ é dada por:

$$\mathbf{p}_{\mathrm{L}} = -\mathbf{L}^{-1}\mathbf{n}, \tag{3.50}$$

no qual $\mathbf{L}$ é definido como:

$$\mathbf{L} = \sum_i \mathbf{n}_{\mathrm{L},k}^{(i)}(\mathbf{n}_{\mathrm{L},k}^{(i)})^T, \tag{3.51}$$

e $\mathbf{n}$ como:

$$\mathbf{n} = \sum_i \mathbf{n}_{\mathrm{L},k}^{(i)}(\mathbf{n}_{\mathrm{L},k}^{(i)})^T(\mathbf{R}_{\mathrm{L}}\mathbf{s}_{\mathrm{L},k-1}^{(i)} - \mathbf{s}_{\mathrm{L},k}^{(i)}). \tag{3.52}$$

Estimação da Pose por Linhas 3D ou Segmentos de Reta

Para a estimação da pose onde as *features* são linhas 3D ou segmentos de retas, o conjunto de pontos escaneados do sensor são descritos por um vetor diretor $\mathbf{c}$ e um ponto $\mathbf{s}_{\mathrm{L}}$. Para a varredura em $k-1$ um conjunto de N_l linhas são descritas por $(\mathbf{c}_{k-1}^{(i)}, \mathbf{s}_{\mathrm{L},k-1}^{(i)})$ e suas correspondentes na varredura k são descritas por $(\mathbf{c}_k^{(i)}, \mathbf{s}_{\mathrm{L},k}^{(i)})$ para $i = 1, \cdots, N_l$. Desta forma, a matriz de rotação entre às duas nuvens de pontos consecutivas é determinada minimizando a função de custo dada por:

$$e(\mathbf{R}_{\mathrm{L}}) = \sum_i \|\mathbf{R}_{\mathrm{L}}\mathbf{c}_{k-1}^{(i)} - \mathbf{c}_k^{(i)}\|^2. \tag{3.53}$$

Portanto, a matriz $\mathbf{R}_{\mathrm{L}}$ é calculada utilizando da decomposição de valores singulares, conforme as Equações 3.43 e 3.44, onde as matrizes $\mathbf{Q}$ e $\mathbf{D}$ são formadas pelos vetores $\mathbf{c}_{\mathrm{L},k-1}^{(i)}$ e $\mathbf{c}_{\mathrm{L},k}^{(i)}$ respectivamente.

Já a translação $\mathbf{p}_{\mathrm{L}}$ é determinada minimizando a soma do quadrado das distâncias entre os pontos $\mathbf{s}_{\mathrm{L},k-1}^{i}$ rotacionados por $\mathbf{R}_{\mathrm{L}}$ e linhas correspondentes definidas pelos vetores $\mathbf{c}_k^{(i)}$ e pelos pontos $\mathbf{s}_{\mathrm{L},k}^{(i)}$. Por fim $\mathbf{p}_{\mathrm{L}}$ é dada por:

$$\mathbf{p}_{\mathrm{L}} = -\mathbf{L}^{-1}\mathbf{n}, \tag{3.54}$$

no qual $\mathbf{L}$ é definido como:

$$\mathbf{L} = \sum_i (\mathbf{I} - \mathbf{c}_k^{(i)}(\mathbf{c}_k^{(i)})^T)^T(\mathbf{I} - \mathbf{c}_k^{(i)}(\mathbf{c}_k^{(i)})^T), \tag{3.55}$$

onde $\mathbf{I}$ é uma matriz identidade e $\mathbf{n}$ é dado por:

$$\mathbf{n} = \sum_i (\mathbf{I} - \mathbf{s}_{\mathrm{L},k}^{(i)}(\mathbf{I} - \mathbf{c}_k^{(i)}(\mathbf{c}_k^{(i)})^T)^T(\mathbf{I} - \mathbf{c}_k^{(i)}(\mathbf{c}_k^{(i)})^T)(\mathbf{R}_{\mathrm{L}}\mathbf{s}_{\mathrm{L},k-1}^{(i)} - \mathbf{s}_{\mathrm{L},k}^{(i)}). \tag{3.56}$$

3.5.4 Mapeamento LiDAR

O mapeamento de ambientes escaneados por um sensor LiDAR nada mais é que o registro de múltiplas varreduras ou, dependendo da *feature* extraída, de múltiplos conjuntos de pontos, linhas ou planos. Para realizar a construção do mapa, as nuvens de pontos escaneadas devem estar expressas no sistema de coordenadas inercial. Essa transformação é dada pela pose do dispositivo robótico que pode ser estimada pela odometria LiDAR, onde são utilizadas as técnicas apresentadas na Seção 3.5.3 determinando o deslocamento realizado pelo robô a cada instante de tempo a partir de uma pose conhecida.

Com o sensor LiDAR rigidamente acoplado ao dispositivo, os pontos $\mathbf{s}_{\mathrm{L}}$ podem ser representados com respeito ao sistema de coordenadas do robô segundo:

$$\mathbf{s}_{\mathrm{L}}^{\mathrm{R}} = \mathbf{H}_{\mathrm{L}}^{\mathrm{R}}\mathbf{s}_{\mathrm{L}}, \tag{3.57}$$

onde $\mathbf{H}_{\mathrm{L}}^{\mathrm{R}}$ é a matriz de transformação homogênea do sensor para o sistema de coordenadas do robô, e $\mathbf{s}_{\mathrm{L}}^{\mathrm{R}}$ são os pontos do LiDAR descritos com respeito ao sistema de coordenadas do robô. Desta forma, a partir da pose estimada com respeito ao sistema inercial O expressa por uma matriz de transformação homogênea $\mathbf{H}_{\mathrm{R}}^{\mathrm{O}}$ pode facilmente relacionar os pontos lidos pelo sensor ao mapa por:

$$\mathbf{s}_{\mathrm{L}}^{\mathrm{O}} = \mathbf{H}_{\mathrm{R}}^{\mathrm{O}}\mathbf{H}_{\mathrm{L}}^{\mathrm{R}}\mathbf{s}_{\mathrm{L}}. \tag{3.58}$$

Os pontos obtidos pelo escaneamento do ambiente, após transformados, são associados ao mapa aplicando técnicas de registros de nuvem de pontos como o algoritmo ICP apresentado na Seção 3.5.3. Porém, ao invés de utilizar duas varreduras consecutivas como entradas do algoritmo, é utilizado o modelo do ambiente e a varredura atual do sensor LiDAR. Desta forma a pose do dispositivo é determinada com respeito ao sistema de coordenadas inercial e os pontos escaneados podem ser registrados no mapa. Abordagens de SLAM convencionais como o GraphSLAM, apresentado na Seção 3.4, podem ser utilizadas para realizar o registro múltiplo de varreduras, como utilizado na técnica de LiDAR SLAM desenvolvida em [Koide et al., 2019]. Outra técnica também utilizada é o *Bayes Tree*, apresentado em [Kaess et al., 2012] e utilizado também em [Zhang and Singh, 2017, Shan and Englot, 2018].

Quando os ambientes de trabalho de um dispositivo robótico não são estruturados, por exemplo, ambientes naturais, mapas geométricos tridimensionais são utilizados. Na

robótica 3 modelos que recebem destaque são: Grade de Elevação que representa o ambiente de forma discretizada, onde a altura de um obstáculo h_{L} é registrada na localização (x_i, y_j); Grade 3D ou Nuvem de pontos 3D; e *Meshes* ou Malhas.

O modelo Grade de Elevação geralmente é utilizado por robôs móveis que operam em ambientes naturais sem superfícies verticais ou saliências. Contudo, essa característica impossibilita o uso em ambientes confinados ou subterrâneos como é o caso do robô utilizado nesta dissertação. Por outro lado, os modelos de Grade 3D e nuvem de pontos 3D são mais apropriados a robôs terrestres, devido à representação mais próxima do real dos ambientes. Outra vantagem é que os mapas podem ser gerados diretamente por meio da associação dos dados do sensor LiDAR 3D. Para isso, duas estruturas em forma de árvore são comumente utilizadas para registro de nuvem de pontos 3D: *Octrees* e *KD-trees* [Meagher, 1982, Ooi, 1987].

As estruturas como Grade 3D registram as informações dos sensores utilizando estimação probabilística da ocupação de um voxel, como em [Hornung et al., 2013], que integra estes dados utilizando a técnica *Occupancy Grid Mapping*. Nessa abordagem é utilizada uma estrutura do tipo *Octree*, onde a probabilidade da ocupação de uma folha da árvore n_{L} é dada em função das medidas do sensor LiDAR $\mathbf{s}_{\mathrm{L},1:k}$.

Assim como na Grade 3D, os modelos descritos por *meshes* ou malhas conseguem representar a geometria de um ambiente tridimensional, porém, por meio de polígonos conectados por seus vértices. A estrutura mais comumente utilizada para compor uma malha é o triângulo, devido a simples descrição e ao grande número de placas gráficas que processam estas estruturadas com uma alta velocidade [Shewchuk, 2002]. Nuvens de pontos não estruturadas calculadas utilizando medidas discretas e ruidosas do ambiente podem ser representadas por um conjunto de *meshes* utilizando métodos de reconstrução de superfícies como apresentado por Azpúrua et al. [2021]. Estas estratégias estimam uma variedade bidimensional que são aproximadas de superfícies 3D reais que compõem a cena. Porém, ambientes que possuem vegetação dispersa, por exemplo, não podem ser mapeados por superfícies, diferentemente dos modelos de Grade 3D ou nuvem de pontos 3D.

Capítulo 4

Análise, Adaptação e Fusão de Técnica de LiDAR SLAM

Este Capítulo descreve as metodologias aplicadas para a investigação das técnicas de LiDAR SLAM para implementação em um dispositivo de inspeção de ambientes confinados. Inicialmente são apresentadas as 3 técnicas de LiDAR SLAM analisadas, descrevendo as métricas utilizadas para a avaliação. Na sequência é apresentada a técnica de LiDAR SLAM que foi adaptada para uso no EspeloRobô, ressaltando as principais teorias aplicadas em cada etapa do algoritmo utilizado. Por fim, é descrito o EKF integrado ao SLAM para fundir a odometria LiDAR com odometria das rodas e IMU, considerando inspeções de ambientes com túneis e galerias com poucas *features* geométricas.

4.1 Análise Comparativa das Técnicas LiDAR SLAM

Com o objetivo de escolher uma técnica de LiDAR SLAM que atendesse os requisitos para aplicação no EspeleoRobô, inicialmente foi realizada uma análise de técnicas consideradas como estado da arte disponíveis em código aberto. A metodologia a ser utilizada deve fornecer estimativas de pose *online*, gerar mapas geometricamente representativos e leves, e ser executada em um computador embarcado no robô. A técnica irá fornecer dados para outros algoritmos do robô responsáveis pelo controle de navegação e planejamento de caminhos.

4.1.1 Técnicas LiDAR SLAM Investigadas

As técnicas analisadas nesta dissertação foram LOAM-Velodyne, LeGO-LOAM e HDL-Graph-SLAM, descritas a seguir.

4.1.1.1 LOAM-Velodyne

A técnica LOAM-Velodyne é uma versão adaptada para o sensor Velodyne da proposta desenvolvida em [Zhang and Singh, 2014], que possui baixo erro de desvio da odometria e dispensa o uso de técnicas de *loop closure*. O LOAM-Velodyne fornece as estimações de odometria LiDAR a cada 10 Hz, e o mapa é gerado a cada 1 Hz. A metodologia é subdivida em 4 etapas: Registro da Nuvem de Pontos, Odometria LiDAR, Mapeamento LiDAR e Integração das Transformações, comforme ilustrado na Figura 14.

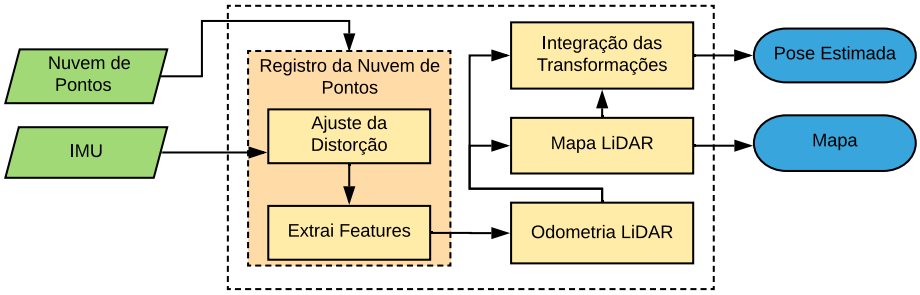


Figura 14 – Esquema geral de funcionamento da técnica LOAM-Velodyne, adaptada de [Zhang and Singh, 2014]

No módulo Registro da Nuvem de Pontos o algoritmo permite o ajuste da distorção da nuvem causada na ego-movimentação por meio dos dados fornecidos pela IMU, onde as *features* são extraídas destacando as bordas e os planos conforme a Equação de suavidade 3.39. No módulo de Odometria LiDAR, o método Levenberg-Marquardt é utilizado em uma única etapa para determinar a pose 6D do dispositivo, que é calculada em um sistema englobando tanto as restrições de borda quanto as de planos. Para o mapeamento LiDAR é utilizado o filtro *voxel-grid* em uma estrutura do tipo *KD-tree*, onde o alinhamento do mapa é feito utilizando um método similar ao ICP. A técnica de Levenberg-Marquardt é aplicada novamente para determinação da pose entre os conjuntos de pontos de bordas e planos com respeito a uma área de 10 m^3 do mapa, selecionada próximo da última posição estimada. No módulo de Integração das transformações, as poses retornadas pelos módulos de Odometria LiDAR e Mapeamento LiDAR são integradas gerando uma pose de saída.

4.1.1.2 LeGO-LOAM

A técnica LeGO-LOAM apresentada em [Shan and Englot, 2018] é uma versão mais leve do LOAM [Zhang and Singh, 2014], otimizada para veículos terrestres que obtém estimativas de pose *online*, podendo ser utilizada em um sistema embarcado de baixo consumo energético como em uma Jetson Nano. As principais diferenças entre as técnicas são a opção de uso de *loop closure*, e a estimação da pose do robô. A metodologia fornece as estimações de odometria LiDAR a cada 10 Hz, e o mapa é gerado a cada 2 Hz. A técnica LeGO-LOAM possui 5 módulos sendo eles: Segmentação, Extração de *Features*,

Odometria LiDAR, Mapeamento LiDAR e Integrações das Transformações. Seu esquema geral de funcionamento está ilustrado na Figura 15, apresentando as entradas e saídas, e as relações entre os módulos.

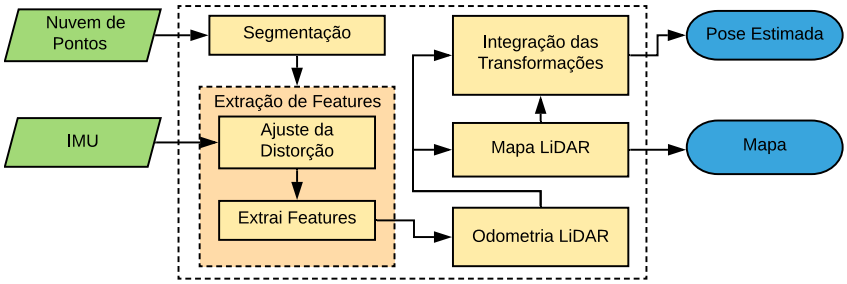


Figura 15 – Esquema geral de funcionamento da técnica LeGO-LOAM.

A nuvem de pontos do sensor LiDAR inicialmente é processada no módulo de Segmentação, onde são filtrados os *outliers*, remapeando os dados da nuvem em uma imagem de profundidade e segmentando. Em seguida, os dados são enviados para o módulo de Extração de *features*, onde são separados em dois conjuntos, um para pontos presentes em quinas e bordas e outros para pontos presentes em planos. Feito isso, estas *features* são utilizadas como restrições no cálculo da ego-movimentação pelo módulo de Odometria LiDAR utilizando a técnicas de Levemberg-Marquardt em duas etapas. Os dados referentes a pose, bem com os conjuntos de pontos selecionados, são enviados para o módulo de Mapeamento LiDAR, onde são registrados utilizando um grafo de poses e fundidos ao mapa pela técnica *Bayes-Tree*. Por fim, as poses geradas pela Odometria LiDAR e Mapeamento LiDAR são integradas no módulo de Integração das Transformações.

4.1.1.3 HDL-Graph-SLAM

O HDL-Graph-SLAM é uma técnica de LiDAR SLAM que faz parte de um sistema de rastreamento de pessoas [Koide et al., 2019]. A metodologia realiza a estimação da pose do dispositivo, mapeamento do ambiente e detecção da movimentação de pessoas próximas. Para auxílio da estimação da pose, a metodologia utiliza um sistema de identificação de *features* no plano do piso, semelhante ao LeGO-LOAM, com uma IMU para o uso em ambientes *indoor* ou GPS para ambientes *outdoor*. A Figura 16 mostra o esquema geral de funcionamento do sistema.

Para resolver o problema do SLAM, a técnica HDL-Graph-SLAM utiliza o algoritmo RANSAC para a extração de *features* da nuvem de pontos. Estas as informações são utilizadas como restrições na determinação da pose do dispositivo que é estimada utilizando o método NDT por *Scan Matching*. O mapeamento é gerado pelo algoritmo Graph-SLAM, no qual podem ser utilizados os métodos ICP, GICP, NDT para alinhamento dos pontos para registro e estimação da pose em relação ao mapa. Para a estimação da pose de saída, etapa equivalente à Integração de Transformações da técnica LOAM-Velodyne, o sistema

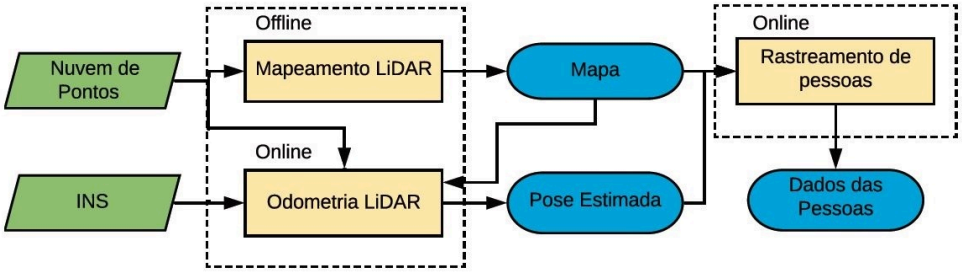


Figura 16 – Esquema geral de funcionamento da técnica HDL-Graph-SLAM, adaptada de [Koide et al., 2019].

utiliza um filtro UKF. A metodologia possui, assim como a LeGO-LOAM, a opção de uso da técnica de *loop closure*, corrigindo tanto o mapa como a pose gerada.

A técnica fornece as estimações de odometria LiDAR a cada 10 Hz, e o mapa é gerado conforme o deslocamento do robô; a configuração padrão do HDL-Graph-SLAM considera intervalos de 2 m entre publicações do mapa. Além disso, não há garantia de tempo máximo para a identificação de *loop closure*, de forma os próprios autores mencionam que a estratégia de mapeamento é executada de forma *offline* [Koide et al., 2019].

4.1.2 Métricas de Avaliação

Para a investigação das técnicas de LiDAR SLAM foram propostas 5 métricas para análise das odometrias estimadas e 1 métrica para a análise dos mapas gerados. Para isso, é considerado $\mathbf{p}_{\mathrm{R},k} \in \mathbb{R}^3$ como a posição estimada do robô por cada uma das técnicas no instante k com um total de N poses registradas.

A primeira métrica compara o deslocamento realizado pelo robô com uma distância de referência l_{max}. A métrica $\mathcal{M}_1$ é dada por:

$$\mathcal{M}_1 = \left| \sum_{k=1}^{N-1} \|\mathbf{p}_{\mathrm{R},k+1} - \mathbf{p}_{\mathrm{R},k}\| - l_{max} \right| {l_{max}}^{-1}. \tag{4.1}$$

A segunda métrica foi proposta para os experimentos onde o robô retorna à posição inicial. A métrica $\mathcal{M}_2$ é a medida do erro acumulado pela odometria dado por:

$$\mathcal{M}_2 = \|\mathbf{p}_{\mathrm{R},N} - \mathbf{p}_{\mathrm{R},1}\|. \tag{4.2}$$

A terceira métrica, $\mathcal{M}_3$, quantifica o quão ruidoso é o sinal da odometria, dado pela mínima distância da posição $\mathbf{p}_{\mathrm{R},k}$ no instante de tempo k em relação a um segmento de reta formado pelas posições $\mathbf{p}_{\mathrm{R},k-n}$ e $\mathbf{p}_{\mathrm{R},k+n}$. A métrica $\mathcal{M}_3$ corresponde ao ruído médio, calculado por:

$$\mathcal{M}_3 = \frac{1}{N-2n} \sum_{t=n+1}^{N-n} \frac{\|\mathbf{q}_{k-n,k+n} \times \mathbf{q}_{k,k-n}\|}{\|\mathbf{q}_{k-n,k+n}\|}, \tag{4.3}$$

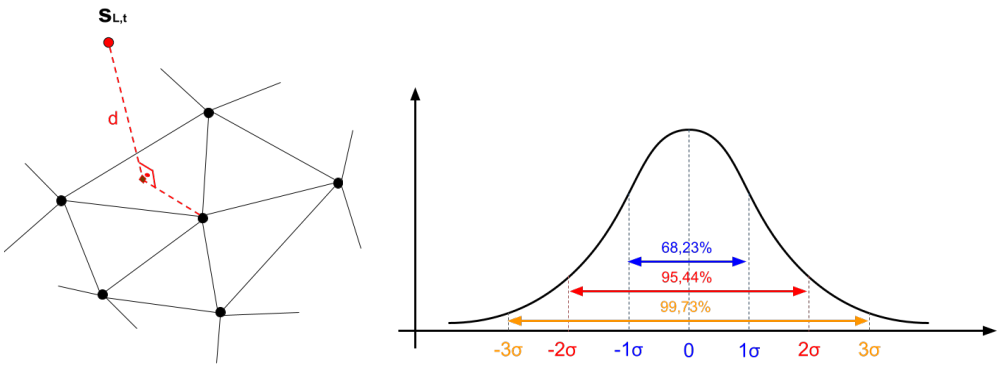
onde $\mathbf{q}_{k-n,k+n}$ é o vetor formado pelas posições $\mathbf{p}_{\mathrm{R},k-n}$ e $\mathbf{p}_{\mathrm{R},k+n}$, e $\mathbf{q}_{k,k-n}$ é o vetor formado pelas posições $\mathbf{p}_{\mathrm{R},k}$ e $\mathbf{p}_{\mathrm{R},k-n}$. Para os experimentos foi determinado o valor de n de forma que o segmento de reta $\mathbf{q}_{k-n,k+n}$ não fosse grande o suficiente para comprometer os resultados.

A quarta e quinta métricas fornecem uma estimativa da variação entre os resultados obtidos pelas diferentes técnicas de localização, calculando o erro médio e a variância com respeito às posições $\mathbf{p}_{g,k}$ fornecidas pela técnica de SLAM com melhor precisão. A escolha desta técnica utilizada como referência pode ser feita com base nos resultados obtidos por $\mathcal{M}_1$, $\mathcal{M}_2$ e $\mathcal{M}_3$. As métricas $\mathcal{M}_4$ e $\mathcal{M}_5$ são obtidas por:

$$\mathcal{M}_4 = \frac{1}{N}\sum_{k=1}^{N}(\mathbf{p}_{\mathrm{R},k} - \mathbf{p}_{g,k}), \tag{4.4}$$

$$\mathcal{M}_5 = \frac{1}{N-1}\sum_{k=1}^{N}(\mathbf{p}_{\mathrm{R},k} - \mathcal{M}_4)^2. \tag{4.5}$$

Para a investigação dos mapas é utilizado um algoritmo que calcula a distância *nearest neighbor* [Sankaranarayanan et al., 2007], por meio do software CloudCompare [CloudCompare, 2018]. A métrica compara duas nuvens de pontos analisando as distâncias de cada ponto e o modelo de superfície mais próximo determinado por meio dos pontos próximos, conforme ilustrado na Figura 17(a).



(a) *Nearest Neighbor Distance* (b) Intervalos de Confiança

Figura 17 – Métrica utilizada na análise do mapa onde: (a) ilustra a distância entre um ponto e um modelo de superfície triangulada, e (b) ilustra os intervalos de confiança para uma distribuição gaussiana utilizados na análise dos histogramas das distâncias computadas.

A análise dos mapas é feita por meio dos intervalos de confiança dos histogramas normalizados conforme as distâncias computadas entre os mapas. Para isso foi considerado um modelo gaussiano normal com média zero e variância finita, como mostra a Figura 17(b), onde os intervalos de confiança de 1σ, 2σ e 3σ são utilizados para a análise estatísticas dos erros associados.

4.2 Técnica LiDAR SLAM Adaptada para o EspeleoRobô

Esta Seção apresenta a técnica de LiDAR SLAM adaptada e utilizada nos experimentos realizados com EspeleoRobô, tanto em ambientes simulados quanto nos reais. Realizando uma pré-análise das metodologias apresentadas, a técnica LOAM-Velodyne é uma abordagem com capacidade de execução em tempo real que fornece as medidas para odometria LiDAR a cada 1 Hz e o mapa a cada 10 Hz. Porém, por não aplicar técnicas de *loop cloure* para reduzir os erros de estimação da pose ao revisitar lugares já mapeados, inviabiliza o seu uso em ambientes que possuem circuitos fechados, como uma rede de galerias. Por outro lado, a técnica HDL-Graph-SLAM possui a opção do uso de técnicas de *loop closure*, contudo, realiza o mapeamento de forma *offline*, o que pode impossibilitar a correção da odometria e fornecimento do mapa atualizado ao sistema de navegação autônoma do robô. Desta forma, foi optado pela técnica LeGO-LOAM, que assim como LOAM-Velodyne é uma técnica com capacidade de execução em tempo real, sendo otimizada para veículos terrestres como o EspeleoRobô, e utilizar a técnica de *loop cloure* para atualizar o mapa e odometria de forma *online*.

As adaptações realizadas no pacote LeGO-LOAM consistem na: alteração dos algoritmos para uso dos sensores embarcados no EspeleRobô, realização de transformações necessárias da nuvem de pontos, odometria e mapa com respeito ao sistema de coordenadas do robô; e parametrização de variáveis referentes a odometria LiDAR, mapeamento e *loop closure* para facilitar a inicialização e configuração dos algoritmos de SLAM. Desse modo, o restante dessa seção descreve em mais detalhes os aspectos teóricos da técnica de LiDAR SLAM, bem como as modificações realizadas, separados em 4 Subseções: Aquisição e Pré-Tratamento dos Dados, Associação de Features, Otimização do Mapa, e Integração das Transformações.

4.2.1 Aquisição e Pré-Tratamento dos Dados

Na etapa de aquisição e pré-tratamento de dados, a técnica considera como entrada uma nuvem de pontos 3D gerada pelo sensor LiDAR, que é descrita pela lista:

$$\mathbf{s}_{\mathrm{L}}^{(1:m)} = \{\mathbf{s}_{\mathrm{L}}^{(1)}, \cdots, \mathbf{s}_{\mathrm{L}}^{(m)}\}, \tag{4.6}$$

onde $\mathbf{s}_{\mathrm{L}} = [s_{\mathrm{L},x}\ s_{\mathrm{L},y}\ s_{\mathrm{L},z}]^T$. Sendo assim, estes pontos inicialmente são projetados em uma imagem de profundidade, conforme apresentado na Figura 18. Essa imagem é expressa pela tripla (i_m, j_m, d), na qual i_m e j_m são as coordenadas do ponto dadas pelos ângulos ψ_{L} e θ_{L}, e d é a distância do ponto em relação ao sistema de coordenadas do sensor. Para isso, uma matriz $\mathbf{M}_{\mathrm{s}}$ é dimensionada de modo que a quantidade de linhas é dada pela resolução vertical do sensor e a quantidade de colunas pela resolução horizontal.

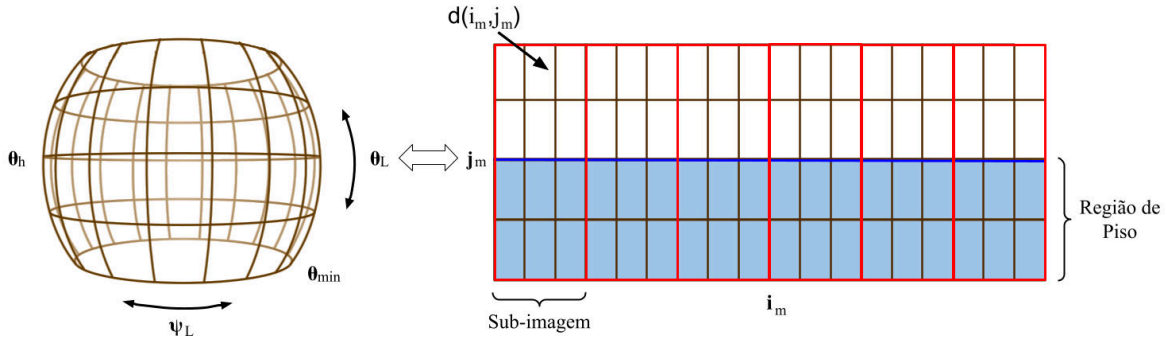


Figura 18 – Projeção retilínea de um sensor *Scanning* LiDAR 3D.

A posição vertical de um ponto $\mathbf{s}_\mathrm{L}$ na matriz $\mathbf{M}_\mathrm{s}$ é calculada por:

$$\mathrm{i_m} = \frac{\theta_\mathrm{L} + \theta_{min}}{\theta_{res}}, \tag{4.7}$$

onde $\mathrm{i_m}$ é o índice da linha, θ_{min} é o menor ângulo que o sensor consegue escanear, θ_{res} é o ângulo de resolução na vertical e θ_L é ângulo calculado por meio das coordenadas do ponto, dado por:

$$\theta_\mathrm{L} = \mathrm{atan2}\left(s_{\mathrm{L},z}, \sqrt{s_{\mathrm{L},x}^2 + s_{\mathrm{L},y}^2}\right). \tag{4.8}$$

A posição horizontal é determinada como:

$$\mathrm{j_m} = \frac{\psi_\mathrm{L} - \frac{i}{2}}{\psi_{res}} + \frac{h}{2}, \tag{4.9}$$

onde $\mathrm{j_m}$ é o índice da coluna, h é a quantidade de feixes horizontais do sensor LiDAR, ψ_{res} é o ângulo de resolução na horizontal e ψ_L é o ângulo formando pelas componentes x e y do ponto, calculado por:

$$\psi_\mathrm{L} = \mathrm{atan2}\left(s_{\mathrm{L},y}, s_{\mathrm{L},x}\right). \tag{4.10}$$

A distância Euclidiana d até o obstáculo é dada por:

$$d = \sqrt{s_{\mathrm{L},x}^2 + s_{\mathrm{L},y}^2 + s_{\mathrm{L},z}^2}\,. \tag{4.11}$$

Após a projeção na matriz $\mathbf{M}_\mathrm{s}$, os pontos que são detectados no piso são rotulados. Para isso, é realizado uma análise dos dados da parte inferior do sensor, como demonstra a região marcado em azul da Figura 18, delimitado pelo ângulo do plano horizontal θ_h. A seleção é feita analisando o ângulo θ_v formado entre dois pontos na mesma linha vertical, dado que:

$$|\theta_v - \theta_h| \leq \theta_g \tag{4.12}$$

onde θ_g é um valor de ângulo definido em [Shan and Englot, 2018] como 10° e θ_v é calculado da seguinte forma:

$$\theta_v = \mathrm{atan2}\left(s_{\mathrm{L},z}^{(i)} - s_{\mathrm{L},z}^{(i+1)}, \sqrt{(s_{\mathrm{L},x}^{(i)} - s_{\mathrm{L},x}^{(i+1)})^2 + (s_{\mathrm{L},y}^{(i)} - s_{\mathrm{L},y}^{(i+1)})^2}\right) \tag{4.13}$$

Depois da rotulação dos pontos do piso é realizado a segmentação da imagem de acordo com o ângulo de segmentação α_{seg} que é configurado originalmente em Shan and Englot [2018] como 10°. Portanto, para cada ponto de $\mathbf{M}_\mathrm{s}$ é calculado o ângulo α_L com respeito aos seus pontos vizinhos, conforme Equação 4.14:

$$\alpha_\mathrm{L} = \mathrm{atan2}\left(d_2 \sin(\alpha_{res}), (d_1 - d_2 \cos(\alpha_{res}))\right), \tag{4.14}$$

onde d_1 e d_2 são as distâncias euclidianas do ponto de referência e do ponto vizinho analisado, com d_1 é selecionado para ser maior do que d_2. O parâmetro α_{res} é o ângulo de resolução, que pode ser tanto θ_{res} quanto ψ_{res}, dependendo se o vizinho está na linha vertical ou horizontal. Posto isso, se α_L for maior que α_{seg} então o ponto analisado é marcado como segmentado. Este processo de segmentação é iterativo e se repete enquanto encontrar um ponto não segmentado ou percorrer todos os pontos de $\mathbf{M}_\mathrm{s}$.

Após a segmentação os pontos são enviados para realizar a extração de *features* e cálculo da odometria LiDAR.

4.2.2 Associação de *Features*

Com a nuvem de pontos segmentada, antes de realizar a correspondência entre *features* para estimar a ego-movimentação, é realizado o ajuste da distorção da nuvem por meio dos dados da IMU entre varreduras do sensor LiDAR. Para isso são integradas as acelerações lineares e velocidades angulares fornecidos para IMU determinando a pose $\mathbf{x}_{La}$ com respeito à última pose estimada da odometria LiDAR $\mathbf{x}_\mathrm{L}$. Sendo assim, a matriz de transformação homogênea $\mathbf{H}_\mathrm{L}^\mathrm{D}$ é gerada da pose $\mathbf{x}_\mathrm{La}$, no qual a nuvem de pontos $\mathbf{s}_\mathrm{L}$ é ajustada segundo a Equação 4.15:

$$\mathbf{s}_\mathrm{D} = \mathbf{H}_\mathrm{L}^\mathrm{D} \mathbf{s}_\mathrm{L}, \tag{4.15}$$

onde $\mathbf{s}_\mathrm{D}$ são os pontos ajustados.

Após o ajuste da distorção, a nuvem de pontos $\mathbf{s}_\mathrm{D}$ é utilizada para extrair as *features*. Os pontos de planos são analisados tanto no conjunto dos rotulados como piso quanto no dos segmentados, já os pontos de borda são analisados somente no conjunto dos segmentados. Estes pontos são classificados segundo a Equação 3.39, redefinida na Equação 4.16 como:

$$s = \frac{1}{|\mathrm{S_D}| \ \| \mathbf{s}_{\mathrm{D},k}^{(i)} \|} \ \| \sum_{j \in \mathrm{S_D}, j \neq i} \left(\mathbf{s}_{\mathrm{D},k}^{(i)} - \mathbf{s}_{\mathrm{D},k}^{(j)} \right) \|, \tag{4.16}$$

na qual $\mathrm{S_D}$ é um conjunto de pontos distorcidos sequenciais de uma mesma linha da imagem de profundidade. Para valores de s menores que um limiar s_{min} os pontos são classificados como presentes em um plano e, para valores de maior que um limiar s_{max} os pontos são classificados como bordas.

Os pontos são separados em 4 conjuntos analisados em 6 regiões (sub-imagens) iguais, como mostra a Figura 18, sendo eles $\mathcal{F}_b \subset \mathbb{F}_b$ para pontos de bordas e $\mathcal{F}_p \subset \mathbb{F}_p$ para pontos em planos. Os conjuntos $\mathbb{F}_b$ e $\mathbb{F}_p$ são os conjuntos de todos os pontos de bordas e planos identificados em uma varredura. Para cada região são selecionados um número máximo de *features* por linha, sendo propostos por Shan and Englot [2018] 40 pontos de bordas e 80 pontos de planos. Já para os subconjuntos $\mathcal{F}_b$ e $\mathcal{F}_p$ são selecionados 2 e 4 pontos, respectivamente, para cada linha de cada região.

A estimação da ego-movimentação entre duas varreduras do sensor é realizada pela correspondência do tipo "ponto-linha" e "ponto-plano". Contudo, basicamente é preciso determinar a correlação dos conjuntos de pontos $\mathcal{F}_{b,k}$ e $\mathcal{F}_{p,k}$, da varredura atual no instante de tempo k, com os conjuntos $\mathbb{F}_{b,k-1}$ e $\mathbb{F}_{p,k-1}$ do instante anterior $k-1$. A correlação "ponto-linha" é dada pelo cálculo da distância d_b de um ponto $\mathbf{s}_{\mathrm{D}}^{(i)} \in \mathcal{F}_{b,k}$ uma reta formado por dois pontos $\mathbf{s}_{\mathrm{D}}^{(j)}$ e $\mathbf{s}_{\mathrm{D}}^{(n)} \in \mathbb{F}_{b,k-1}$. Posto isso o cálculo de d_b é dado por:

$$d_b = \frac{\left|(\mathbf{s}_{\mathrm{D},k}^{(i)} - \mathbf{s}_{\mathrm{D},k-1}^{(j)}) \times (\mathbf{s}_{\mathrm{D},k}^{(i)} - \mathbf{s}_{\mathrm{D},k-1}^{(n)})\right|}{\left|\mathbf{s}_{\mathrm{D},k-1}^{(j)} - \mathbf{s}_{\mathrm{D},k-1}^{(n)}\right|}. \tag{4.17}$$

Para a correspondência do tipo "ponto-plano" é calculada a distância d_p entre um ponto $\mathbf{s}_{\mathrm{D}}^{(i)} \in \mathcal{F}_{p,k}$ e um plano formado pelos pontos $\mathbf{s}_{\mathrm{D}}^{(j)}$, $\mathbf{s}_{\mathrm{D}}^{(n)}$ e $\mathbf{s}_{\mathrm{D}}^{(l)} \in \mathbb{F}_{p,k-1}$. Então a distância d_p é calculada da seguinte forma:

$$d_p = \frac{\left|\begin{array}{c}(\mathbf{s}_{\mathrm{D},k}^{(i)} - \mathbf{s}_{\mathrm{D},k-1}^{(j)}) \\ ((\mathbf{s}_{\mathrm{D},k}^{(j)} - \mathbf{s}_{\mathrm{D},k-1}^{(n)}) \times (\mathbf{s}_{\mathrm{D},k}^{(j)} - \mathbf{s}_{\mathrm{D},k-1}^{(l)}))\end{array}\right|}{\left|(\mathbf{s}_{\mathrm{D},k}^{(j)} - \mathbf{s}_{\mathrm{D},k-1}^{(n)}) \times (\mathbf{s}_{\mathrm{D},k}^{(j)} - \mathbf{s}_{\mathrm{D},k-1}^{(l)})\right|}. \tag{4.18}$$

A relação geométrica entre os pontos de bordas $\mathbf{s}_{\mathrm{D}}^{(i)} \in \mathcal{F}_{p,k}$ com a linha correspondente em $\mathbb{F}_{b,k-1}$, bem como a relação entre os pontos de planos $\mathbf{s}_{\mathrm{D}}^{(i)} \in \mathcal{F}_{p,k}$ com os planos correspondentes em $\mathbb{F}_{p,k-1}$ é dada por uma função não-linear, como:

$$f_b(\mathbf{s}_{\mathrm{D}}^{(i)}, \mathbf{x}_{\mathrm{D}}) = d_b \quad , i \in \mathcal{F}_{b,k}, \tag{4.19}$$

$$f_p(\mathbf{s}_{\mathrm{D}}^{(i)}, \mathbf{x}_{\mathrm{D}}) = d_p \quad , i \in \mathcal{F}_{p,k}, \tag{4.20}$$

onde $\mathbf{x}_{\mathrm{D}}$ é pose relativa da odometria LiDAR.

Para solucionar o problema de otimização dado pelo sistema composto pelas funções f_b e f_p é utilizada a técnica de Levenberg-Marquadt (LM) em duas etapas para minimizar a distância (d_b, d_p) entre às duas varreduras do sensor [Zisserman, 2004]. Na primeira etapa é calculada a pose $\mathbf{x}_{\mathrm{D}}$ por meio da correlação dos pontos de planos $\mathcal{F}_{p,k}$ e $\mathbb{F}_{p,k-1}$ utilizando a Equação 4.20, onde são aproveitadas as componentes $p_{\mathrm{D},z}$, ϕ_{D} e θ_{D}, e descartadas as demais componentes devido à estimação pobre. Na sequência são calculadas as componentes $p_{\mathrm{D},x}$, $p_{\mathrm{D},y}$ e ψ_{D} por meio da correlação dos pontos de bordas $\mathcal{F}_{b,k}$ e $\mathbb{F}_{b,k-1}$

usando a Equação 4.19, utilizando os resultados anteriores $p_{\mathrm{D},z}$, ϕ_{D} e θ_{D} como restrições. Por fim a pose $\mathbf{x_D}$ é integrada com respeito ao sistema de coordenadas inercial resultando na Odometria LiDAR.

4.2.3 Otimização do Mapa

Na etapa de otimização do mapa, o conjunto de *features* dado pelas uniões dos conjuntos $\{\mathbb{F}_b, \mathbb{F}_p\}$ é utilizado para refinar o cálculo da pose do dispositivo robótico pela correspondência com o mapa de pontos acumulado $\mathbf{M}_{\mathrm{L}}^{\overline{\mathrm{O}}}$ representado com respeito ao sistema de coordenadas inercial $\overline{\mathrm{O}}$ do LiDAR SLAM, definido pelo primeiro sistema de coordenadas do sensor. Após determinado o alinhamento dos dados, os pontos são adicionados a esse mapa somado aos pontos de *outliers* que foram selecionados na etapa de segmentação.

Diferentemente da técnica LOAM que registra um único mapa de pontos, a técnica LeGO-LOAM salva múltiplos sub-mapas:

$$\mathbf{M}_{\mathrm{L},k-1}^{\overline{\mathrm{O}}} = \{\{\mathbb{F}_{b,1}^{\overline{\mathrm{O}}}, \mathbb{F}_{p,1}^{\overline{\mathrm{O}}}\}, \{\mathbb{F}_{b,2}^{\overline{\mathrm{O}}}, \mathbb{F}_{p,2}^{\overline{\mathrm{O}}}\}, \cdots, \{\mathbb{F}_{b,k-1}^{\overline{\mathrm{O}}}, \mathbb{F}_{p,k-1}^{\overline{\mathrm{O}}}\}\}. \tag{4.21}$$

no qual $\mathbf{M}_{\mathrm{L},k-1}^{\overline{\mathrm{O}}}$ é o mapa de pontos acumulado até o instante de tempo $k-1$, e $\{\mathbb{F}_{b,1}^{\overline{\mathrm{O}}}, \mathbb{F}_{p,1}^{\overline{\mathrm{O}}}\}$ a $\{\mathbb{F}_{b,k-1}^{\overline{\mathrm{O}}}, \mathbb{F}_{p,k-1}^{\overline{\mathrm{O}}}\}$ são os sub-mapas representados com respeito ao sistema inercial $\overline{\mathrm{O}}$.

Cada sub-mapa é gerado dado um tamanho mínimo pré-determinado. O procedimento para fusão de cada varredura é semelhante ao apresentado na técnica LOAM [Zhang and Singh, 2014], que utiliza um filtro *voxel-grid* para diminuir a resolução do mapa acumulado pela fusão dos pontos escaneados. A resolução desse mapa é definida pela variável *leafsize*, sendo originalmente configurada para fundir os pontos de bordas com uma resolução de 5 cm e pontos de planos com resolução de 10 cm. Esta variável foi transformada em um parâmetro configurável, assim como o tamanho máximo do mapa.

Para estimar a pose do dispositivo, o procedimento é similar ao utilizado na odometria LiDAR, porém, é realizado ao final da varredura do *laser* utilizando todos os pontos segmentados e alinhados. Para isso, os pontos $\mathbf{s}_{\mathrm{L}}$ inicialmente são transformados para o sistema de coordenadas inercial, utilizando os dados da odometria LiDAR. Em seguida, são extraídas as *features* determinadas as correspondências entre os conjuntos de pontos da varredura com os sub-mapas que compõem o mapa $\mathbf{M}_{\mathrm{L},k}^{\overline{\mathrm{O}}}$. Nesta etapa, também é utilizado o método de Levemberg-Marquardt para determinar a pose.

Outra diferença entre as técnicas LeGO-LOAM e a LOAM é que o LeGO utiliza de um grafo, chamado de *pose graph*, para relacionar cada um dos sub-mapas que são descritos como nós. As restrições espaciais entre os nós do grafo são descritas pelas translações calculados utilizando o método LM. Periodicamente é realizada uma análise do sub-mapa gerado pelo escaneamento com os demais por meio da técnica de *loop closure*. Identificado um *loop*, a transformação de correção é utilizada como uma restrição no

grafo e, tanto o caminho estimado quanto o mapa são corrigidos utilizando um método *Baes-Tree* apresentado em [Kaess et al., 2012].

A técnica LeGO-LOAM é executada com respeito ao sistema de coordenadas inercial $\overline{\mathrm{O}}$ do sensor LiDAR, sendo necessário realizar a transformação do mapa de pontos $\mathbf{M}_{\mathrm{L},k}^{\overline{\mathrm{O}}}$ para o sistema de coordenadas inercial O utilizado pelo dispositivo robótico, definido pelo sistema de coordenadas inicial do robô. Para isso, foram realizadas medições manuais da posição do sensor no robô, considerando os eixos entre os sistemas de coordenadas alinhados. Em trabalhos futuros estes valores serão ajustados por calibração, como apresentado em [Underwood et al., 2007]. Portanto, a transformação da nuvem de pontos é dada pela multiplicação da matriz de transformação homogênea $\mathbf{H}_{\overline{\mathrm{O}}}^{\mathrm{O}}$, definida pela translação e orientação entre sensor e robô, conforme Equação 4.22:

$$\mathbf{M}_{\mathrm{L},k}^{\mathrm{O}} = \mathbf{H}_{\overline{\mathrm{O}}}^{\mathrm{O}} \mathbf{M}_{\mathrm{L},k}^{\overline{\mathrm{O}}}. \tag{4.22}$$

4.2.4 Integração das Transformações

As posições e orientações geradas pela odometria e mapeamento LiDAR são combinadas para formar a pose integrada de saída do LiDAR SLAM na etapa de Integração das Transformações. Para tal, são considerados os sistemas de coordenadas L da odometria LiDAR, M do mapeamento LiDAR, I da pose integrada, e R do robô. Como referência, são utilizados os sistemas de coordenadas inerciais $\overline{\mathrm{O}}$ do LiDAR SLAM e O do robô.

As poses $\mathbf{x}_{\mathrm{L}}^{\overline{\mathrm{O}}}$ da odometria LiDAR, $\mathbf{x}_{\mathrm{M}}^{\overline{\mathrm{O}}}$ do mapeamento LiDAR, e $\mathbf{x}_{\mathrm{I}}^{\overline{\mathrm{O}}}$ integrada são descritas respectivamente por:

$$\mathbf{x}_{\mathrm{L}}^{\overline{\mathrm{O}}} = (\mathbf{p}_{\mathrm{L}}^{\overline{\mathrm{O}}}, \mathbf{R}_{\mathrm{L}}^{\overline{\mathrm{O}}}), \tag{4.23}$$

$$\mathbf{x}_{\mathrm{M}}^{\overline{\mathrm{O}}} = (\mathbf{p}_{\mathrm{M}}^{\overline{\mathrm{O}}}, \mathbf{R}_{\mathrm{M}}^{\overline{\mathrm{O}}}), \tag{4.24}$$

$$\mathbf{x}_{\mathrm{I}}^{\overline{\mathrm{O}}} = (\mathbf{p}_{\mathrm{I}}^{\overline{\mathrm{O}}}, \mathbf{R}_{\mathrm{I}}^{\overline{\mathrm{O}}}), \tag{4.25}$$

representadas com respeito ao sistema inercial $\overline{\mathrm{O}}$. Já a pose do robô $\mathbf{x}_{\mathrm{R}}^{\mathrm{O}}$ é descrita como:

$$\mathbf{x}_{\mathrm{R}}^{\mathrm{O}} = (\mathbf{p}_{\mathrm{R}}^{\mathrm{O}}, \mathbf{R}_{\mathrm{R}}^{\mathrm{O}}), \tag{4.26}$$

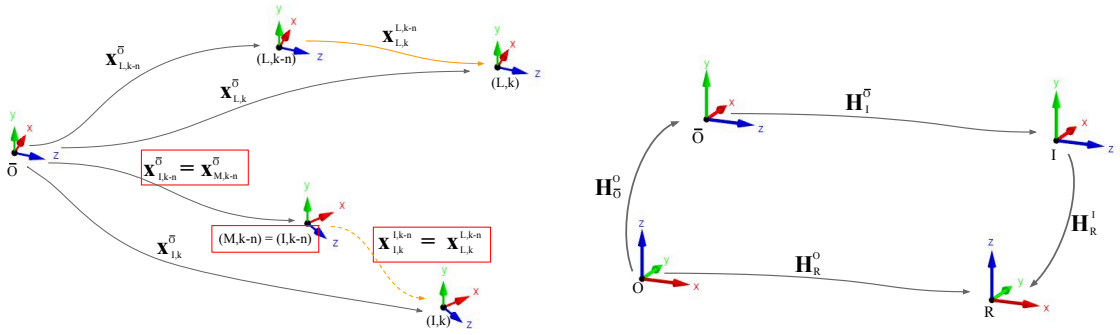
representada com respeito ao sistema inercial O.

Os pares formados pelos vetores de posições $\mathbf{p}$ e pelas matrizes de rotações $\mathbf{R}$ são utilizados para gerar as respectivas matrizes de transformações homogêneas $\mathbf{H}$ conforme a Equação 3.9. Desta forma, a partir das poses $\mathbf{x}_{\mathrm{L}}^{\overline{\mathrm{O}}}$, $\mathbf{x}_{\mathrm{M}}^{\overline{\mathrm{O}}}$, $\mathbf{x}_{\mathrm{I}}^{\overline{\mathrm{O}}}$ e $\mathbf{x}_{\mathrm{R}}^{\mathrm{O}}$, são determinadas as respectivas matrizes de transformações homogêneas $\mathbf{H}_{\mathrm{L}}^{\overline{\mathrm{O}}}$, $\mathbf{H}_{\mathrm{M}}^{\overline{\mathrm{O}}}$, $\mathbf{H}_{\mathrm{I}}^{\overline{\mathrm{O}}}$ e $\mathbf{H}_{\mathrm{R}}^{\mathrm{O}}$.

A odometria LiDAR, calculada entre duas varreduras do sensor, disponibiliza a pose $\mathbf{x}_{\mathrm{L}}^{\overline{\mathrm{O}}}$ a uma frequência de 10 Hz, porém, apresentando baixa fidelidade quanto à precisão. Já o mapeamento LiDAR computa a pose $\mathbf{x}_{\mathrm{M}}^{\overline{\mathrm{O}}}$ diretamente com respeito ao

inercial $\overline{\mathrm{O}}$ de forma mais precisa, contudo fornecida a uma frequência de 2 Hz de forma sincronizada com a odometria LiDAR. Portanto, o procedimento de integração consiste em gerar uma pose integrada $\mathbf{x}_{\mathrm{I}}^{\overline{\mathrm{O}}}$ buscando combinar a alta precisão de $\mathbf{x}_{\mathrm{M}}^{\overline{\mathrm{O}}}$ com a alta frequência de estimação de $\mathbf{x}_{\mathrm{L}}^{\overline{\mathrm{O}}}$.

Para este processo iterativo as poses são descritas com respeito ao instante de tempo em que são fornecidas. Por exemplo, a pose $\mathbf{x}_{\mathrm{L},k}^{\overline{\mathrm{O}}}$ corresponde a pose da odometria LiDAR no instante de tempo k, e $\mathbf{x}_{\mathrm{L},k-n}^{\overline{\mathrm{O}}}$ corresponde à pose dada no instante de tempo $k-n$, no qual n representa decrementos de tempo, conforme ilustrado na Figura 19(a).



(a) Integração das Transformações (b) Transformação Final

Figura 19 – (a) Relação entre os sistemas de coordenadas das poses da odometria LiDAR, do mapeamento LiDAR e da pose integrada, utilizadas na Integração das Transformações; e (b) relação entre os sistemas de coordenadas da pose integrada e a pose correspondente do robô.

Para a integração da odometria LiDAR é considerada a pose relativa $\mathbf{x}_{\mathrm{L},k}^{\mathrm{L},k-n}$, determinada pela pose $\mathbf{x}_{\mathrm{L},k}^{\overline{\mathrm{O}}}$ representada com respeito à pose $\mathbf{x}_{\mathrm{L},k-n}^{\overline{\mathrm{O}}}$, calculada segundo as relações de transformações apresentadas pela parte superior da Figura 19(a) da seguinte forma:

$$\mathbf{p}_{\mathrm{L},k}^{\mathrm{L},k-n} = \mathbf{R}_{\overline{\mathrm{O}}}^{\mathrm{L},k-n}\left(\mathbf{p}_{\mathrm{L},k}^{\overline{\mathrm{O}}} - \mathbf{p}_{\mathrm{L},k-n}^{\overline{\mathrm{O}}}\right), \tag{4.27}$$

$$\mathbf{R}_{\mathrm{L},k}^{\mathrm{L},k-n} = \mathbf{R}_{\overline{\mathrm{O}}}^{\mathrm{L},k-n}\mathbf{R}_{\mathrm{L},k}^{\overline{\mathrm{O}}}. \tag{4.28}$$

onde $\mathbf{R}_{\overline{\mathrm{O}}}^{\mathrm{L},k-n}$ é a matriz transposta de $\mathbf{R}_{\mathrm{L},k-n}^{\overline{\mathrm{O}}}$.

Toda vez que uma pose do mapeamento LiDAR é fornecida, por exemplo no instante de tempo $k-n$, a pose integrada é corrigida tal que $\mathbf{x}_{\mathrm{I},k-n}^{\overline{\mathrm{O}}} = \mathbf{x}_{\mathrm{M},k-n}^{\overline{\mathrm{O}}}$, conforme apresentado na parte inferior da Figura 19(a). Enquanto o algoritmo espera por uma nova pose do mapeamento, a pose integrada atual $\mathbf{x}_{\mathrm{I},k}^{\overline{\mathrm{O}}}$ é computada pela pose $\mathbf{x}_{\mathrm{I},k-n}^{\overline{\mathrm{O}}}$ somada à pose relativa $\mathbf{x}_{\mathrm{L},k}^{\mathrm{L},k-n}$ da odometria LiDAR, assumido que $\mathbf{x}_{\mathrm{L},k}^{\mathrm{L},k-n} = \mathbf{x}_{\mathrm{I},k}^{\mathrm{I},k-n}$, como ilustrado pelos seguimentos em laranja na Figura 19(a). Portanto, a pose integrada $\mathbf{x}_{\mathrm{I},k}^{\overline{\mathrm{O}}}$ é calculada por:

$$\mathbf{p}_{\mathrm{I},k}^{\overline{\mathrm{O}}} = \mathbf{p}_{\mathrm{I},k-n}^{\overline{\mathrm{O}}} + \mathbf{R}_{\mathrm{I},k-n}^{\overline{\mathrm{O}}}\mathbf{p}_{\mathrm{I},k}^{\mathrm{I},k-n}, \tag{4.29}$$

$$\mathbf{R}_{\mathrm{I},k}^{\overline{\mathrm{O}}} = \mathbf{R}_{\mathrm{I},k-n}^{\overline{\mathrm{O}}}\mathbf{R}_{\mathrm{I},k}^{\mathrm{I},k-n}. \tag{4.30}$$

Cabe destacar que a pose integrada $\mathbf{x}_{\mathrm{I}}^{\overline{\mathrm{O}}}$ é calculada com respeito ao sistema de coordenadas inercial $\overline{\mathrm{O}}$ do LiDAR SLAM, sendo ainda necessário determinar a pose correspondente do robô $\mathbf{x}_{\mathrm{R}}^{\mathrm{O}}$ com respeito ao sistema de coordenadas O. Para isso, as poses são representadas pelas suas respectivas matrizes de transformações homogêneas, como mostra a relação apresentada na Figura 19(b). Portanto, a pose do robô $\mathbf{x}_{\mathrm{R}}^{\mathrm{O}}$ descrita por $\mathbf{H}_{\mathrm{R}}^{\mathrm{O}}$ é determinada como:

$$\mathbf{H}_{\mathrm{R}}^{\mathrm{O}} = \mathbf{H}_{\overline{\mathrm{O}}}^{\mathrm{O}}\mathbf{H}_{\mathrm{I}}^{\overline{\mathrm{O}}}\mathbf{H}_{\mathrm{R}}^{\mathrm{I}}, \tag{4.31}$$

onde $\mathbf{H}_{\mathrm{I}}^{\overline{\mathrm{O}}}$ é a matriz de transformação homogênea da pose integrada $\mathbf{x}_{\mathrm{I}}^{\overline{\mathrm{O}}}$, e $\mathbf{H}_{\mathrm{R}}^{\mathrm{I}}$ é a matriz de transformação entre os sistemas de coordenadas do robô e da pose integrada. Vale ressaltar que $\mathbf{H}_{\mathrm{R}}^{\mathrm{I}}$ é igual à matriz inversa de $\mathbf{H}_{\overline{\mathrm{O}}}^{\mathrm{O}}$, dado que os sistemas de coordenadas inerciais são determinados pelas poses iniciais do robô e do sensor LiDAR.

4.3 Filtro EKF Integrado ao LiDAR SLAM

As técnicas LiDAR SLAM acumulam erros na estimação da pose do dispositivo em ambientes que possuem poucas *features* geométricas como túneis, galerias, ou mesmo corredores muito extensos e homogêneos. Esses ambientes caracterizam partes dos locais de trabalhos do EspeleoRobô. Visando corrigir possíveis erros no cálculo da odometria, bem como o mapa, uma solução é a aplicação de um filtro entre as etapas de *Back-end* e *Front-end* da técnica de SLAM. Uma solução consiste em aplicar um filtro à odometria LiDAR, fornecendo uma pose corrigida para a estimativa inicial do mapeamento LiDAR e para integração das transformações, como mostra a Figura 20. Para isso foi proposto um filtro EKF para fundir a odometria LiDAR com a odometria das rodas e os dados da IMU.

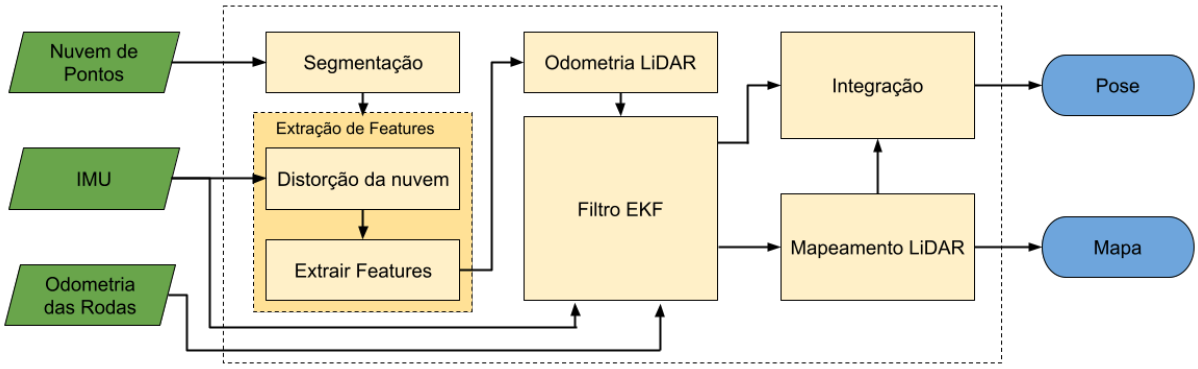


Figura 20 – Filtro de Kalman Estendido aplicado à técnica LiDAR SLAM.

O filtro EKF foi desenvolvido inicialmente como uma extensão do trabalho realizado por Rafael Fernandes em [Fernandes et al., 2020], adaptando o algoritmo robot-

pose-ekf[1] para usar um modelo 6D para fundir as medidas dos sensores. As correções utilizam as medições relativas da odometria das rodas, IMU e odometria LiDAR. Portanto, conforme que uma nova informação é disponibilizada, é realizada uma correção descrita pela diferença entres medidas consecutivas. Então, o algoritmo usa interpolação linear para determinar a pose a *posteriori* estimada.

Na etapa de correção da IMU a medida corresponde à orientação $\boldsymbol{\varphi}_u = [\phi_u \ \theta_u \ \psi_u]^T$ fornecida pelo sensor. Portanto, o modelo de medição e covariância da medida são dados por:

$$g_u(\boldsymbol{\varphi}_{\mathrm{E},k+1|k}) = \begin{bmatrix} \boldsymbol{\varphi}_{\mathrm{E},k+1|k} \end{bmatrix} , \tag{4.32}$$

$$\boldsymbol{\Sigma}_\delta = \begin{bmatrix} \sigma_{\phi_u} & 0 & 0 \\ 0 & \sigma_{\theta_u} & 0 \\ 0 & 0 & \sigma_{\psi_u} \end{bmatrix} , \tag{4.33}$$

onde σ_{ϕ_u}, σ_{θ_u} e σ_{ψ_u} são as variâncias referentes às medidas $\boldsymbol{\varphi}_u$ dadas pelo fabricante.

Devido a odometria das rodas ser definida pela pose planar $\mathbf{p}_\mathrm{R} = [p_{\mathrm{R},x} \ p_{\mathrm{R},y} \ \psi_\mathrm{R}]^T$, calculada pelo do modelo *Skid-Steering* apresentado na Seção 3.2, de covariância $\boldsymbol{\Sigma}_\mathrm{R}$, e a etapa de correção da IMU utilizar somente a orientação $\boldsymbol{\varphi}_u = [\phi_u \ \theta_u \ \psi_u]^T$, é necessário calcular a componente z. Sendo assim, uma solução possível é utilizar o deslocamento em x e y da odometria das rodas combinados com o ângulo de *pitch* da IMU. esta forma, o modelo de medição é dado por:

$$g_\mathrm{R}(\mathbf{p}_{\mathrm{E},k+1|k}, \psi_{\mathrm{E},k+1|k}) = \begin{bmatrix} v_{\mathrm{R},k} \cos(\theta_{\mathrm{E},k+1|k}) \cos(\psi_{\mathrm{E},k+1|k}) + p_{\mathrm{E},x,k|k} \\ v_{\mathrm{R},k} \cos(\theta_{\mathrm{E},k+1|k}) \sin(\psi_{\mathrm{E},k+1|k}) + p_{\mathrm{E},y,k|k} \\ -v_{\mathrm{R},k} \sin(\theta_{\mathrm{E},k+1|k}) + p_{\mathrm{E},z,k|k} \\ \omega_{\mathrm{R},k} + \psi_{\mathrm{E},k|k} \end{bmatrix} , \tag{4.34}$$

onde as velocidades $v_{\mathrm{R},k}$ e $\omega_{\mathrm{R},k}$ são calculadas segundo o modelo *Skid-Steering* apresentado na Equação 3.15.

A covariância da medida é definida como:

$$\boldsymbol{\Sigma}_\delta = \mathbf{G}_\mathrm{R} \begin{bmatrix} \sigma_{\mathrm{R},x} & 0 & 0 & 0 \\ 0 & \sigma_{\mathrm{R},y} & 0 & 0 \\ 0 & 0 & \sigma_{\theta_u} & 0 \\ 0 & 0 & 0 & \sigma_{\psi_\mathrm{R}} \end{bmatrix} \mathbf{G}_\mathrm{R}^T , \tag{4.35}$$

onde $\sigma_{\mathrm{R},x}$, $\sigma_{\mathrm{R},y}$ e σ_{ψ_R} são as variâncias referentes componentes x, y e ψ das poses $\mathbf{x}_\mathrm{R}$ computadas pela odometria das rodas pela Equação 3.16. Estas variâncias dependem da velocidade angular $\dot{\psi}_\mathrm{R}$ assumindo valores diferentes para movimentos em linha reta

[1] O robot-pose-ekf é um algoritmo desenvolvido como pacote do meta-sistema operacional ROS disponível em https://github.com/ros-planning/robot_pose_ekf.

e em rotações. Já σ_{θ_u} é a variância da componente θ_u da medida $\boldsymbol{\varphi}_u$ da IMU, e $\mathbf{G}_{\mathrm{R}}$ é a Jacobiana da função $g_{\mathrm{R}}(\mathbf{x}_{\mathrm{E},k+1|k})$.

Na etapa de correção da odometria LiDAR a medida corresponde à pose $\mathbf{x}_{\mathrm{L}} = [\mathbf{p}_{\mathrm{L}}{}^T \ \boldsymbol{\varphi}_{\mathrm{L}}^T]^T$, onde o modelo de medição e a covariância da medida são dados por:

$$g_{\mathrm{L}}(\mathbf{x}_{\mathrm{E},k+1|k}) = \begin{bmatrix} \mathbf{p}_{\mathrm{E},k+1|k} \\ \boldsymbol{\varphi}_{\mathrm{E},k+1|k} \end{bmatrix}, \tag{4.36}$$

$$\boldsymbol{\Sigma}_{\delta} = \begin{bmatrix} \mathrm{g}_x g_b(n_b) & 0 & 0 & 0 & 0 & 0 \\ 0 & \mathrm{g}_y g_b(n_b) & 0 & 0 & 0 & 0 \\ 0 & 0 & \mathrm{g}_z g_p(n_p) & 0 & 0 & 0 \\ 0 & 0 & 0 & \mathrm{g}_\phi g_p(n_p) & 0 & 0 \\ 0 & 0 & 0 & 0 & \mathrm{g}_\theta g_p(n_p) & 0 \\ 0 & 0 & 0 & 0 & 0 & \mathrm{g}_\psi g_b(n_b) \end{bmatrix}, \tag{4.37}$$

onde g_x, g_y, g_z, g_ϕ, g_θ e g_ψ são ganhos e, g_b e g_p são funções dadas pela relação do número de *features* de bordas e planos identificados do ambiente, definidas como:

$$g_b(n_b) = \frac{\mathbb{n}_b - n_b}{\mathbb{n}_b}, \tag{4.38}$$

$$g_p(n_p) = \frac{\mathbb{n}_p - n_p}{\mathbb{n}_p}, \tag{4.39}$$

onde n_b e n_p são o número de pontos de borda de planos identificados do ambiente, e $\mathbb{n}_b$ e $\mathbb{n}_p$ os valores ótimos para a quantidade de *features* de borda e planos.

De maneira geral, a medida que o número de *features* diminuí, o erro da estimativa da pose fornecida por uma técnica de LiDAR SLAM aumenta [Wisth et al., 2021]. Desta forma, utilizando um número máximo de *features* de bordas $\mathbb{n}_b$ e planos $\mathbb{n}_p$, para uma estimação ótima, são calculadas as covariâncias referente à pose gerada pela odometria LiDAR pelas funções $g_b(n_b)$ e $g_p(n_p)$ segundo o número de *features* identificadas do ambiente.

Capítulo 5

Arcabouço Experimental

Este Capítulo descreve o arcabouço experimental utilizado nas simulações e experimentos com o EspeleoRobô para a investigação das técnicas de LiDAR SLAM, bem como para a análise preliminar do filtro EKF integrado ao LiDAR SLAM. Inicialmente são descritas as implementações dos algoritmos encapsulados em pacotes ROS desenvolvidos e/ou adaptados das técnicas LOAM-Velodyne, HDL-Graph-SLAM e LeGO-LOAM, e do filtro Espeleo-Pose-EKF. Em seguida é exibida a plataforma robótica, destacando os modelos simulado e real, e por último são apresentados os ambientes virtuais utilizados nas simulações e os ambientes reais utilizados nos experimentos em laboratório e campo.

5.1 Implementações de Algoritmos por meio de Pacotes do ROS

O *Robot Operating System* é um meta-sistema operacional de código aberto desenvolvido para aplicações de projetos robóticos complexos [Quigley et al., 2009]. O ROS disponibiliza um conjunto de bibliotecas e ferramentas para os desenvolvedores, onde é possível realizar abstrações de *hardware*, controle de dispositivos de baixo nível, troca de mensagens entre processos, gerenciamento de pacotes e implementações de diversas funcionalidades, como é esperado de um sistema operacional. O conceito básico do ROS consiste em uma rede ponto-a-ponto de aplicações executadas em paralelo de forma síncrona ou assíncrona, composta por Nós, Mensagens, Tópicos e Serviços, como mostra a Figura 21, além do Master, Servidor de Parâmetros e Bags. Outro conceito são os pacotes que são a unidade principal para organizar um *software* em ROS, podendo conter Nós, bibliotecas dependentes do ROS, arquivos de configuração, conjuntos de dados, dentre outros.

Os Nós são processos, por exemplo, controle dos atuadores dos sensores de um robô, controle de navegação, planejamento de caminhos, dentre muitos outros, gerenciados pelo Master e as mensagens são simples estruturas de dados composta por campos editáveis.

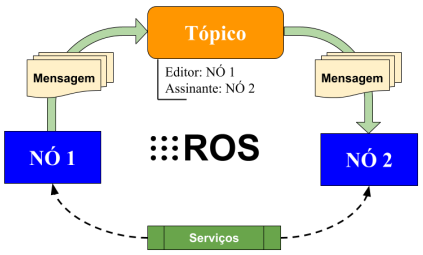


Figura 21 – Ilustração do conceito básico do meta-sistema operacional ROS.

Um Nó pode trocar informações com outros Nós por meio dos Tópicos, que são sistemas de troca de mensagens, ou por meio dos Serviços que são compostos por um par de estruturas, sendo um para realizar uma solicitação de uma informação e outro para obter uma resposta. Já o Servidor de Parâmetros permite salvar dados por uma chave em uma localização central, podendo ser facilmente resgatados pelos Nós. Já os Bags são arquivos utilizados para salvar e reproduzir mensagens do ROS, muito útil na realização de testes de algoritmos.

Para realização dos experimentos desta dissertação, todos os algoritmos utilizados foram implementados e desenvolvidos como pacotes do ROS. Alguns dos pacotes já estavam disponíveis à comunidade, sendo adaptados para uso no EspeloRobô, e outros foram desenvolvidos para permitir a análise comparativa das técnicas LiDAR e a adaptação do filtro EKF aplicado ao LiDAR SLAM. Portanto, esta seção descreve os algoritmos implementados em ROS utilizados e os seus princípios de funcionamento.

5.1.1 LOAM-Velodyne

O pacote LOAM-Velodyne[1] é uma cópia da versão original desenvolvida em [Zhang and Singh, 2014], disponibilizada na plataforma github. A abordagem descrita na Seção 4.1.1 é desenvolvida em 4 algoritmos (Nós): `scanRegistration`, `lidarOdometry`, `lidarMapping` e `transformMaintenance`, como mostra a Figura 22, que ilustra as relações e os tópicos de comunicação entre os nós. As entradas do pacote são: a nuvem de pontos do sensor LiDAR dada pelo Tópico `/os1_cloud_node/points` e os dados da IMU pelo Tópico `/imu/data`, e as saídas são: o mapa de pontos `/laser_cloud_surround` e a odometria integrada dada por `/integrated_to_init`.

A nuvem de pontos e as informações da IMU são processadas primeiramente no algoritmo `scanRegistration`, onde os dados são registrados e separados de acordo com as *features* de bordas e planos, e divididos em 4 grupos, os quais são enviados para o nó `laserOdometry` por meio dos tópicos `/laser_cloud_sharp`, `/laser_cloud_less_sharp`, `/laser_cloud_flat` e `/laser_cloud_less_flat`. São enviados também a nuvem de pon-

[1] https://github.com/daobilige-su/loam_velodyne

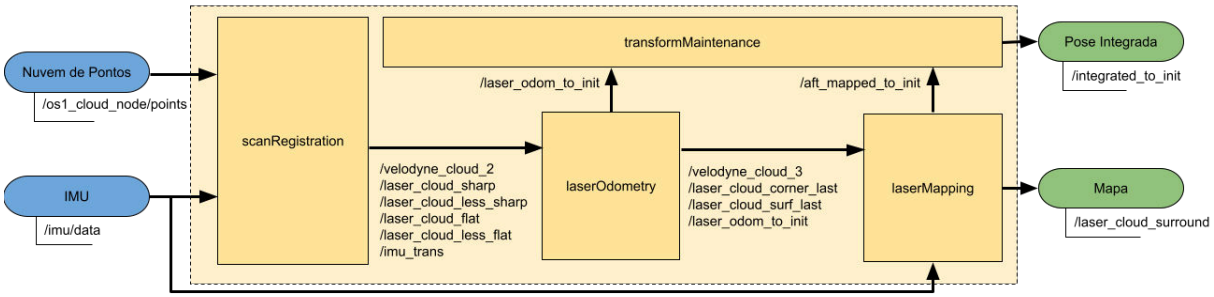


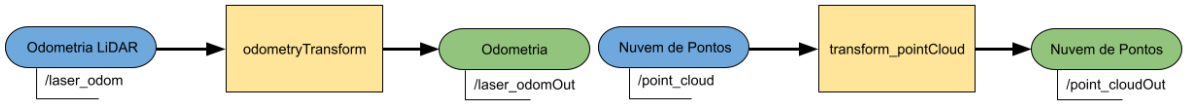
Figura 22 – Esquema geral de funcionamento do pacote LOAM-Velodyne.

tos completa e ajustada, dada pelo tópico `/velodyne_cloud_2`, e os deslocamentos computados da IMU por meio do tópico `/imu_trans`.

O algoritmo `laserOdometry` realiza a estimação da pose do dispositivo entre varreduras do sensor LiDAR utilizando os conjuntos de pontos processados em uma única etapa pela técnica Levemberg-Marquardt. Feito isso, as poses são transformadas para o sistema de coordenadas inercial formando a odometria LiDAR que é enviada para os Nós `laserMapping` e `transformMaintenance` por meio do tópico `/laser_odom_to_init`. O Nó `laserOdometry` também reenvia a nuvem completa pelo tópico `/velodyne_cloud_3`.

O processamento do mapa é executado pelo Nó `laserMapping`. Contudo, primeiramente é realizada a estimação da pose por meio das *features* de bordas e planos em comparação com o mapa completo. Então, os dados da pose são enviados para o nó `transformManteinance` por meio do tópico `/aft_mapped_to_init`. Feito isso, a nuvem de pontos completa, recebida do Tópico `/velodyne_cloud_3`, é representada com respeito ao sistema de coordenadas inercial e fundida ao mapa utilizando um filtro *voxel-grid* em uma estrutura KD-*tree*. Então, esse mapa de pontos é publicado a cada 1Hz no tópico `/laser_cloud_surround`. Por fim, o Nó `transformManteinance` realiza a integração das odometrias LiDAR e do mapa publicando no tópico `/integrated_to_init`.

As principais modificações realizadas no pacote foram em relação aos parâmetros do sensor LiDAR, como resolução horizontal, vertical e frequência de escaneamento, que foram configurados diretamente via linha de código adequando ao LiDAR 3D embarcado no EspeleoRobô. Outra adaptação foi em relação à orientação da nuvem de pontos de entrada, que no algoritmo é considerada com o eixo z apontado para a frente do robô e o eixo y para cima. Sendo assim, é necessário realizar essa transformação da nuvem de pontos tanto na entrada do pacote quanto na saída, além da transformação da odometria de saída com respeito à orientação do robô. Para isso foram criados 2 Nós simples, um chamado de `transform_pointCloud` baseado na biblioteca PCL e outro denominado `odometryTransform` baseado na biblioteca Eigen, como mostra a Figura 23.



(a) Odometria (b) Nuvem de Pontos

Figura 23 – Algoritmos desenvolvidos para transformação da: (a) odometria e (b) nuvem de pontos.

5.1.2 Espeleo-LeGO-LOAM

O pacote Espeleo-LeGO-LOAM é uma versão adaptada do pacote LeGO-LOAM[2] que foi originalmente desenvolvido em [Shan and Englot, 2018], como uma otimização da técnica LOAM para veículos terrestres. As modificações foram realizadas para a implementação no EspeleoRobô, considerando a categoria de sensor, as orientações dos sistemas de coordenadas e as exigências dos demais pacotes presentes no robô. As entradas do pacote são a nuvem de pontos e as leituras da IMU, subscritas dos tópicos `/os1_cloud_node/points` e `/imu/data`, e as saídas correspondem à odometria integrada e um mapa de pontos, publicados nos tópicos `/integrated_to_inti2` e `/laser_cloud_surround2`, conforme ilustrado na Figura 24.

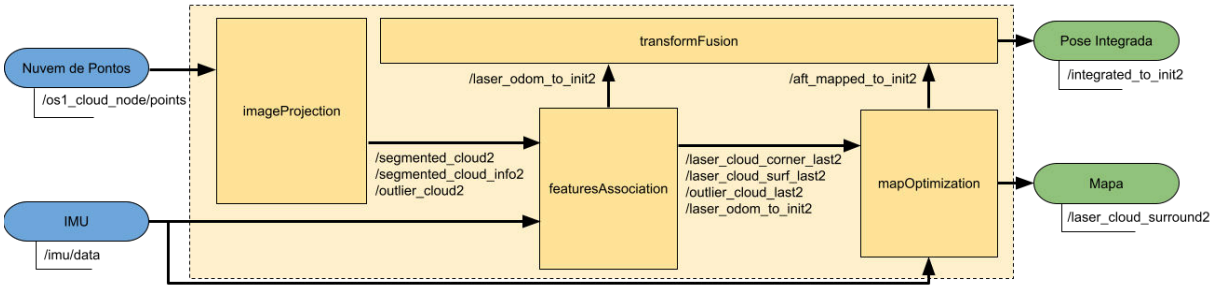


Figura 24 – Esquema geral de funcionamento do pacote LeGO-LOAM adaptado para o EspeleoRobô.

O sensor utilizado no EspeleoRobô é um Ouster OS1-16[3], e possui dois sistemas de coordenadas, um denominado "lidar" com o eixo x apontando para a parte traseira do sensor, e outro para o sensor com o eixo x apontando para frente. Portanto, no algoritmo `imageProjection` a nuvem de pontos é primeiramente representada com respeito ao sistema de coordenadas do sensor (com o eixo x apontado para a frente do robô), utilizando um a transformada adicionado ao algoritmo. Logo após os dados são registrados em uma imagem de profundidade, dimensionada pelas configurações do sensor, e em seguida são segmentados e rotulados quanto ao piso, separando-os em 2 grupos que são enviados para o Nó `featuresAssociation` por meio dos Tópicos `/segmented_cloud2` e

[2] https://github.com/RobustFieldAutonomyLab/LeGO-LOAM
[3] https://ouster.com/

`/outlier_cloud`, além das informações de rotulagem dos pontos enviados pelos tópicos `/segmented_cloud_info2`.

No algoritmo `featuresAssociation`, os pontos são ajustados pelos dados da IMU e classificados como de bordas e planos, onde são utilizados para o cálculo da odometria LiDAR por meio da técnica de Levenberg-Marquardt em duas etapas. Feito isso, os dados da odometria LiDAR são enviados por meio do Tópico `/laser_ odom_to_init2` para os Nós `transformFusion` e `mapOptimization`, junto com os conjuntos de pontos ajustados e enviados pelos Tópicos `/laser_cloud_corner_last2`, `/laser_cloud_surf_last2` e `/outiler_clous_last2`.

No algoritmo `mapOptimization` os dados são orientados inicialmente com respeito ao sistema inercial pela odometria LiDAR integrada ao deslocamento dado pela IMU. Assim, a pose do dispositivo é calculada por meio da correspondência dos pontos com um segmento do mapa principal. Na sequência os conjuntos de pontos de bordas, planos e *outliers* são agrupados e fundidos a um sub-mapa por meio de um filtro *voxel-grid*. Então, o mapa completo é publicado por meio do tópico `/laser_cloud_surround2` transformado para o sistema de coordenadas inercial do robô com o eixo x apontado para frente. A pose gerada é enviada para o nó `transformFusion` por meio do Tópico `/aft_mapped_to_init2`, no qual as odometrias são integradas. Por fim, a odometria é transformada por similaridade para o sistema de coordenadas do robô, como mostra a Seção 4.2.4.

As modificações realizadas para a integração do pacote ao EspeleoRobô foram realizadas de modo a criar uma versão adaptada do algoritmo para o robô. Desta forma, as principais alterações são relacionadas às transformações da nuvem de pontos, identificação do número de *features*, além do cálculo da covariância para uso no filtro da odometria LiDAR e a parametrização de variáveis referentes a odometria LiDAR, mapeamento e *loop closure* para facilitar a inicialização e configuração do algoritmo de SLAM, incluindo o tamanho do mapa global, resolução do mapa dado pelo *leafsize* da estrutura da KD-*tree* utilizada, o ângulo de segmentação, tamanho do sub-mapa, a habilitação do *loop closure*, dentre outros. Portanto, foi gerada a versão `espeleo_lego_loam` que está disponível em https://github.com/ITVRoC/espeleo_lego_loam.

5.1.3 HDL-Graph-SLAM

O pacote HDL-Graph-SLAM[4] possui basicamente 4 algorítimos principais, sendo eles `prefiltering`, `scan_matching_odometry`, `floor_detection` e `hdl_graph_slam`, conforme ilustrado na Figura 25. As entradas do pacote são a nuvem de pontos subscrita no tópico `/os1_cloud_node/points`, e os dados da IMU subscritos no tópico `/imu/data`.

A nuvem de pontos é inicialmente tratada pelo Nó `prefiltering`, separando os

[4] https://github.com/koide3/hdl_graph_slam

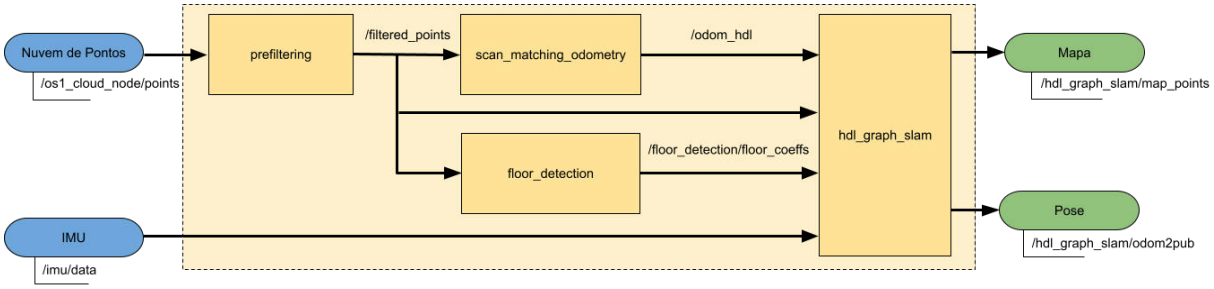


Figura 25 – Esquema geral de funcionamento do pacote HDL-Graph-SLAM.

outliers e registrando os dados que são enviados por meio do tópico `/filtered_points` para os demais nós do pacote. O Nó `scan_matching_odometry` calcula a odometria LiDAR entre duas nuvens de pontos utilizando um método de correspondência, podendo ser ICP, GICP, GICP-OMP, NTD, NDT-OMP. Para os experimentos foi utilizado a configuração NDT, conforme utilizado originalmente pelos autores em [Koide et al., 2019]. A odometria gerada é então enviada para o nó `hdl_graph_slam` por meio do tópico `/odom_hdl`. De forma concomitante é executado o nó `floor_detection`, que realiza a identificação do plano do piso e gera coeficientes de restrições que são enviadas pelo tópico `/floor_detection/floor_coeffs` para a estimação da pose final. Por fim, o mapa é gerado pelo Nó `hdl_graph_slam` utilizando uma abordagem de Graph-SLAM, publicado no Tópico `/hdl_graph_slam/map_points`, e a pose final é fundida com os dados da IMU e a odometria LiDAR usando um UKF, publicadas como uma transformação no Tópico `/hdl_graph_slam/odom2pub`.

Neste pacote, não foram necessárias modificações via código ou de pacotes auxiliares para ser utilizado no EspeleoRobô. Para isso, foram realizadas configurações no pacote original, como nomes de tópicos, sensores utilizados, metodologia aplicada para correspondência entre nuvens de pontos, habilitação da detecção do plano do piso, *loop closure*, tamanho de mapa, dentre outras configurações presentes no arquivo de inicialização do pacote.

5.1.4 Filtro Espeleo-Pose-EKF Adaptado para o LiDAR SLAM

O pacote utilizado para o investigar o uso de filtros integrados à técnica LiDAR SLAM, com o intuito de melhorar a eficiência quando aplicado dentro de dutos e túneis, foi o Espeleo-Pose-EKF[5] desenvolvido como uma extensão do trabalho realizado em [Fernandes et al., 2020]. Este pacote é uma adaptação do robot-pose-ekf[6] para a estimação da pose calculada com a odometria das rodas considerando o modelo tridimensional do robô, fundidos com os dados da IMU. A versão original robot-pose-ekf foi projetada para

[5] https://github.com/ITVRoC/espeleo_pose_ekf
[6] https://github.com/ros-planning/robot_pose_ekf

fusão de uma pose planar ($\mathbf{x}_\mathrm{R} = [p_{\mathrm{R},x}\ p_{\mathrm{R},y}\ \psi_\mathrm{R}]^T$) com os dados da orientação da IMU ($\boldsymbol{\varphi}_u = [\phi_u\ \theta_u\ \psi_u]$) e uma segunda odometria ($\mathbf{x} = [p_x\ p_y\ p_z\ \phi\ \theta\ \psi]^T$) que foi utilizada para integrar a odometria LiDAR.

Os sistemas de coordenadas das poses gerados pelos pacotes Espeleo-LeGO-LOAM e Espeleo-Pose-EKF possuem referenciais inerciais diferentes. Portanto, foi criado um Nó simples, denominado de `odometryTransform` (Figura 26.), para realizar a transformação da odometria LiDAR para o sistema de coordenadas do filtro EKF, e em seguida transformar a odometria corrigida do filtro para o sistema de coordenadas do pacote do LiDAR SLAM.

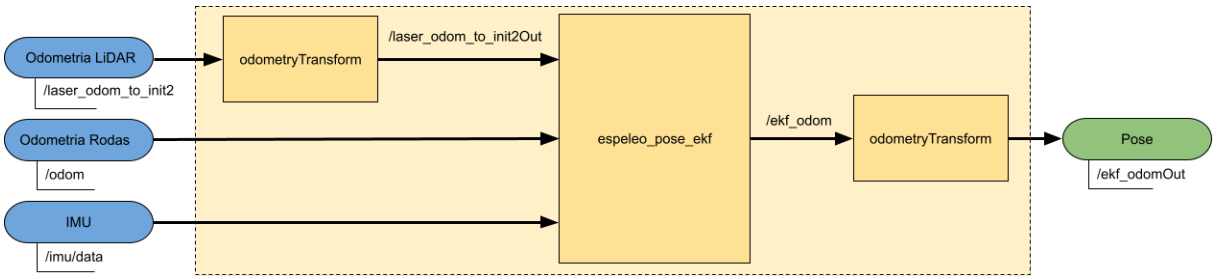


Figura 26 – Esquema geral de funcionamento da implementação pacote espeleo_pose_ekf para adaptado para aplicação no LiDAR SLAM.

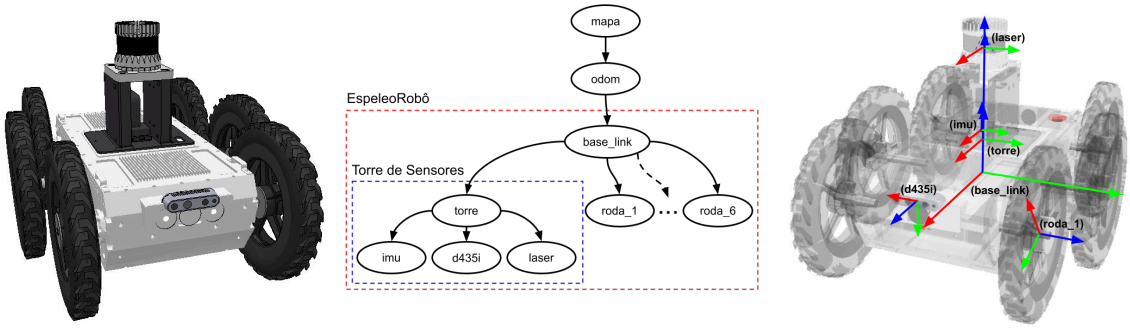
A odometria LiDAR, estimada pelo Nó `featureAssociation` do pacote Espeleo-LeGO-LOAM, primeiramente é representada com respeito ao sistema de coordenadas de referência do pacote Espeleo-Pose-EKF, utilizando o Nó `odometryTransform`. Feito isso, é realizada a fusão da pose LiDAR com a odometria das rodas e a IMU, conforme descrito na seção 4.3. Em seguida, a pose corrigida é representada com respeito ao sistema de coordenadas do pacote Espeleo-LeGO-LOAM utilizando o Nó `odometryTransform` configurado agora para transformação inversa, e é enviada para os Nós `transformFusion` e `mapOptimization`.

5.2 Plataforma Robótica

A plataforma robótica utilizada nos experimentos é o EspeleoRobô, que vem sendo desenvolvido em parceria entre a mineradora Vale S.A, o Instituto Tecnológico Vale e a Universidade Federal de Minas Gerais. Para realização de testes e experimentos dos algoritmos desenvolvidos pela equipe de pesquisa, foi desenvolvida também uma versão virtual da plataforma robótica. Ambas as versões utilizadas neste trabalho de dissertação serão descritas nesta seção.

5.2.1 Modelo Virtual do Robô

Os modelos virtuais do EspeleoRobô foram desenvolvidos para serem utilizados no simulador de robótica CoppeliaSim[7] da Coppelia Robótics, integrado ao ROS [Cid et al., 2020]. A representação física do robô foi virtualizada por meio dos recursos computacionais *Computer-Aided Engineering* (CAE) e *Computer-Aided Design* (CAD) utilizando o *software* SoliWorks[8], como pode ser visto na Figura 27(a).



(a) Modelo Simulado em CAD (b) Árvore de TF's (c) Sistemas de Coordenadas

Figura 27 – Modelo simulado do EspeleoRobô ilustrado em (a) por uma imagem CAD, (b) pela árvore de TF's e em (c) pelas posições dos sistemas de coordenadas robô.

A estrutura física do robô foi descrita utilizando um código *Unifield Robot Description Format* (URDF) [Kunze et al., 2011], implementado num arquivo XML que contém toda a especificação dos *links*, juntas, bem como as propriedades dos atuadores e sensores. Cada componente descrito possui um sistema de coordenadas definido com respeito ao sistema de coordenadas principal do robô, denominado de `base_link`, como pode ser visto na Figura 27(c). A relação entre cada uma dessas partes do robô é representada por uma árvore de transformações, como ilustra a Figura 27(b). Essa árvore foi implementada por meio da ferramenta TF do meta-sistema operacional ROS, que fornece a transformação entre pontos, vetores e planos entre quaisquer sistemas de coordenadas. Para o sensoriamento o modelo simulado foi equipado com um sensor LiDAR do tipo Velodyne VLP 16 com uma resolução horizontal de $0,52°$ e uma frequência de 5 Hz, mais uma IMU composta por 3 acelerômetros e 3 giroscópios, além de uma câmera RGB-D não utilizada no escopo dessa dissertação. Para medir o giro das rodas, foi utilizado um recurso do simulador, medindo diretamente a velocidade dos objetos roda.

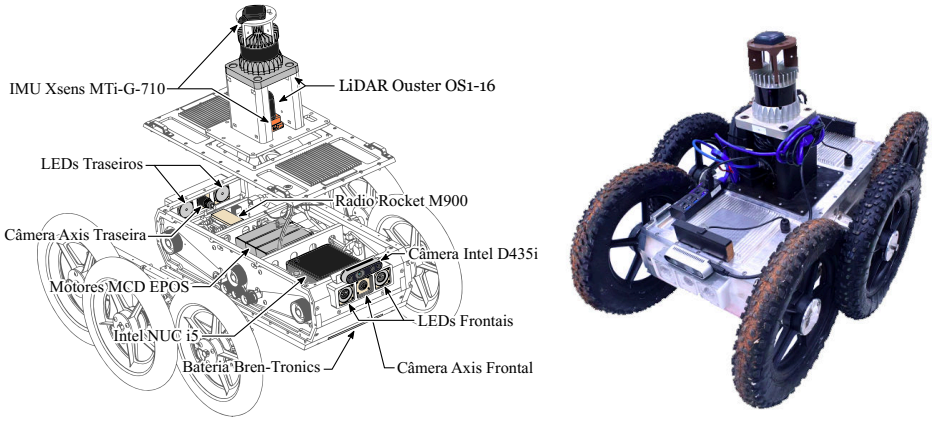
[7] https://www.coppeliarobotics.com/
[8] https://www.solidworks.com/pt-br

5.2.2 Modelo Real do Robô

O modelo real do EspeleoRobô utilizado nesta dissertação está equipado com 6 conjuntos de caixas de redução, motores, encoders e Power Drivers MCP EPOS da Maxon Motors; duas baterias militares Bren-Tronics; um par de câmeras Axis P12 HD com LEDs e sistemas de iluminação IR; e um mini PC Intel NUC Core I5 com 16Gb de memória RAM rodando o meta-sistema operacional ROS, no Ubuntu 16.04. Para comunicação o robô dispõe de um sistema de rádio Ubiquinti Rocket M900.

O dispositivo possui também um sensor Ouster OS1-16 multi-flash LiDAR[9], configurado com uma resolução horizontal de 1024 pontos e uma frequência de 10 Hz, e uma Xsense MTi-G-710 GNSS/INS[10], além de uma câmera Intel RealSense Depth D435i[11] e um módulo UWB Decawave DWM1001[12].

A configuração utilizada nos experimentos possui 6 rodas, como mostra a Figura 28, que apresenta uma visão geral do EspeleoRobô com uma descrição dos seus equipamentos e sensores, e uma imagem real do robô.



(a) Visão Geral (b) EspeleoRobô

Figura 28 – Modelo real do EspeleoRobô descrito em (a) por uma visão geral dos seus equipamentos e sensores e em (b) por uma imagem da versão utilizada nos experimentos.

5.3 Ambientes Utilizados nos Experimentos e Simulações

Os experimentos realizados nesta dissertação foram executados tanto no CoppeliaSim integrado ao ROS, onde foi utilizada a pose dada pelo simulador como *ground truth*,

[9] https://ouster.com/products/os1-LiDAR-sensor/
[10] https://www.xsens.com/products/mti-100-series
[11] https://www.intelrealsense.com/depth-camera-d435i/
[12] https://www.decawave.com/product/dwm1001-module/

quanto em ambientes reais. Esta seção descreve os ambientes virtuais e reais utilizados na realização de simulações e experimentos, coletando dados para realizar a comparação das técnicas de LiDAR SLAM investigadas, os experimentos com o pacote Espeleo-LeGO-LOAM sendo executado *online* embarcado no EspeleoRobô, e os experimentos de testes da implementação do filtro EKF integrado ao LiDAR SLAM.

5.3.1 Ambientes Virtuais Utilizados nas Simulações

As simulações foram realizadas com o *software* CoppeliaSim integrado ao ROS por meio do *plugin* `sim_ros_interface`[13]. Foram utilizados 3 cenários: um segmento de uma das cavernas da competição da DARPA *Subterranean Challenge* empregado para comparação das técnicas LiDAR SLAM em um ambiente representativo ao real de operação do EspeleoRobô; uma galeria quadrada; e um cenário de um duto longo para testar a aplicabilidade do filtro EKF ao LiDAR SLAM.

A caverna da DARPA utilizada é ilustrada na Figura 29, apresentando uma vista externa e uma vista interna com o robô. Este cenário é um ambiente fechado com grandes galerias, solos irregulares e muitos obstáculos rochosos, como pode ser visto na Figura 29(b), além de subidas e descidas.

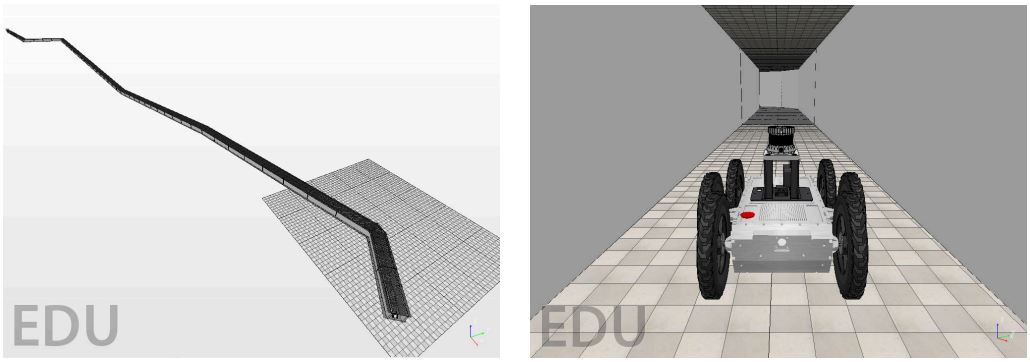
(a) Caverna DARPA (b) Visão Interna

Figura 29 – Ambiente virtual da caverna da DARPA ilustrado em (a) por uma visão em perspectiva do segmento utilizado nos experimentos e em (b) por uma visão interna de uma das galerias do cenário.

O cenário da galeria quadrada, ilustrada Figura 30 com uma vista externa e uma interna da entrada, possui cerca de 300 m de comprimento, 3 curvas abertas, descida e subida. A dificuldade neste cenário está na homogeneidade da geometria do espaço, porém, apresentando planos de referência e linhas longitudinais nos encontros entre paredes, piso e teto.

Outro cenário também utilizado para testes do filtro EKF aplicado ao LiDAR SLAM é um duto longo, que assim como a galeria quadrada, possui cerca de 300 m de

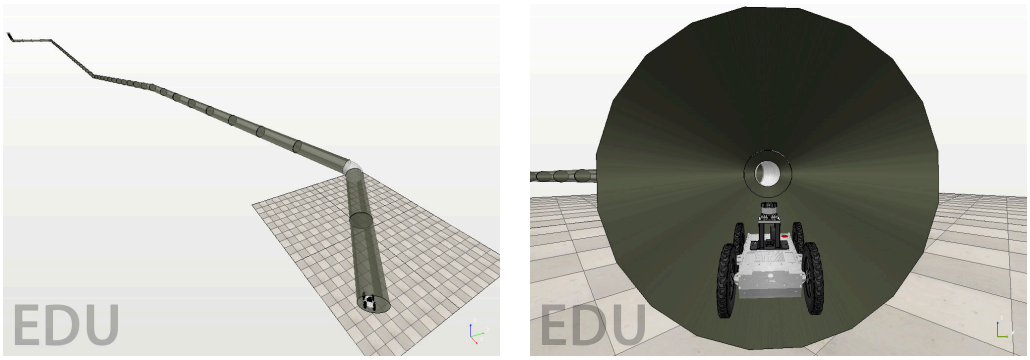
[13] https://github.com/CoppeliaRobotics/simExtROS



(a) Galeria Quadrada (b) Visão Interna

Figura 30 – Ambiente virtual de um galeria quadrada ilustrada em (a) por uma visão em perspectiva do segmento utilizado nos experimentos e em (b) por uma visão interna da entrada da galeria.

comprimento, 3 curvas abertas, descida e subida, porém foi construído com emendas a cada 5 m, como ilustrado na Figura 31. Este ambiente foi criado para testes da técnica LiDAR SLAM em um possível ambiente de operação do EspeleoRobô. Este tipo de ambiente é desafiador para execução de técnicas de LiDAR SLAM por conta da baixa presença *features* geométricas devido à homogeneidade da geometria do espaço e ausência de planos de referência.



(a) Duto Longo (b) Visão Interna

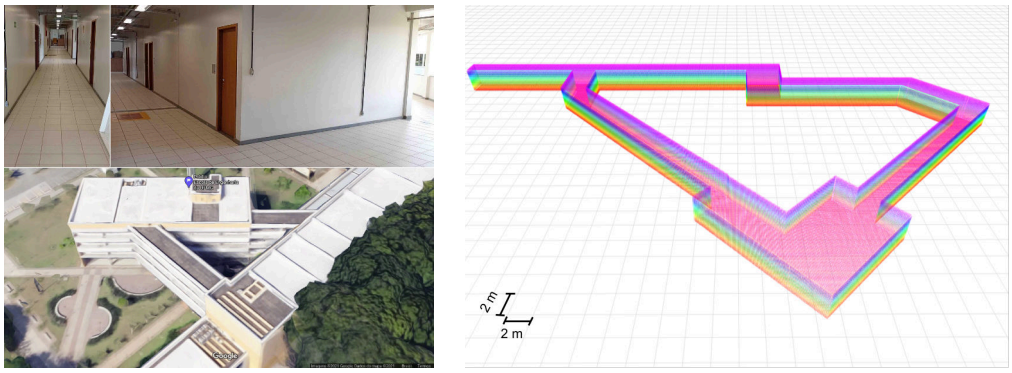
Figura 31 – Ambiente virtual de um duto longo ilustrado em (a) por uma visão em perspectiva do segmento utilizado nos experimentos e em (b) por uma visão interna da entrada do duto.

5.3.2 Ambientes Reais Utilizados nos Experimentos

Os ambientes reais utilizados nos experimentos têm por finalidade investigar as técnicas LiDAR SLAM verificando suas aplicabilidades em diferentes ambientes, validar a técnica escolhida para implementação no EspeleoRobô , e verificar a implementação do EKF integrado ao LiDAR SLAM. Para isso, forma utilizados ambientes *indoors* executando os experimentos nas instalações do prédio da Escola de Engenharia da UFMG, e em

ambientes *outdoors* nas imediações do Restaurante Universitário da UFMG, além de um ambiente subterrâneo na Mina du Veloso localizada na Cidade de Ouro Preto-MG. Os locais foram separados conforme as categorias dos experimentos executados, incluindo testes em laboratório, em ambientes *indoor* e *outdoor*, e subterrâneo.

Os experimentos em laboratório foram realizados nas instalações da Escola de Engenharia da UFMG em um corredor estreito, e num conjunto de corredores formando um circuito fechado. Utilizados para comparar as técnicas LiDAR SLAM investigadas, estes ambientes possuem superfícies planas com um alto coeficiente de atrito e muitas *features* como quinas de portas, janelas e pilastras, além de paredes longas em alguns pontos do trajeto. A Figura 32 mostra uma imagem dada pela reconstrução 3D do mapa da Google, duas vistas internas dos locais utilizados e um mapa de pontos 3D utilizado como referência para a comparação dos mapas gerados pelas técnicas LiDAR SLAM.



(a) Imagens (b) Nuvem de Pontos 3D

Figura 32 – Localização dos experimentos realizados em laboratório apresentado em (a) por duas vistas internas e via reconstrução 3D do mapa Google e em (b) pelo mapa 3D utilizado como referência.

Para os experimentos *indoors* foram utilizadas as instalações da Escola de Engenharia da UFMG, próximo ao auditório e no pátio interno do prédio principal. Assim como nos corredores, estes ambientes possuem muitas *features* geométricas como quinas de portas, janelas, além de bancos, vegetação, pilastras, dentre outros. A Figura 33 mostra uma imagem da reconstrução 3D do mapa Google e 3 vistas internas da câmera do robô.

Os experimentos realizados em ambiente *outdoor* foram executados nas proximidades do restaurante da UFMG, como mostra a Figura 34, que apresenta uma imagem via satélite do mapa Google e 4 vistas do solo retiradas da câmera embarcada do robô. Este ambiente é repleto de *features* como árvores, arbustos e algumas instalações prediais próximas, porém é desafiador no sentido de possuir geometrias muito semelhantes que podem causar erros de correspondência entre a nuvem escaneada e o mapa, podendo gerar, por exemplo, um fechamento de *loop closure* de forma indesejada.

Considerando ambientes subterrâneos, foram executados dois experimentos na

Figura 33 – Localização do experimento realizado no prédio da Escola de Engenharia com uma imagem via reconstrução 3D do mapa Google e 3 vistas internas.

Figura 34 – Localização do experimento externo nas proximidades do refeitório da UFMG com uma imagem via satélite e 4 vistas do solo.

Mina du Veloso, conforme ilustrado na Figura 35, que apresenta imagens de uma galeria aberta, um corredor estreito e uma bifurcação. Um dos experimentos foi realizado em um percurso curto, para testar a técnicas LiDAR SLAM investigadas, e o outro em um percurso um pouco mais longo entrando dentro de uma galeria da mina. O ambiente da mina, possui também muitas *features* geométricas presentes em rochas, nas imperfeições do piso, no teto e paredes. Porém, possui um solo escorregadio e irregular, fazendo com que o robô execute o percurso com muita trepidação e escorregamento das rodas.

Figura 35 – Localização do experimento realizado na Mina du Veloso com 3 vistas internas.

Capítulo 6

Experimentos e Resultados

Este Capitulo está dividido em 3 Seções que descrevem as simulações e os experimentos *online* e *offline* realizados para esta dissertação. A primeira Seção apresenta a análise comparativa das técnicas de LiDAR SLAM LOAM-Velodyne, LeGO-LOAM e HDL-Graph-SLAM. Para isso foram realizadas simulações em ambientes virtuais e experimentos em laboratório e campo. Na segunda Seção são apresentados os resultados da execução *online* da técnica Espeleo-LeGO-LOAM fornecendo a estimativa da pose e o mapa para os algoritmos de planejamento de caminhos e controle de navegação. Por último, a terceira Seção apresenta os resultados da aplicação do filtro EKF integrado ao SLAM executado em dois ambientes virtuais, um duto e uma galeria, e um ambiente real.

Nas simulações e nos experimentos de investigação das técnicas de LiDAR SLAM, assim como para a análise do filtro EKF integrado ao SLAM, as informações dos sensores foram coletadas em arquivos Bags do ROS. Estes dados foram processados *offline*, porém, em tempo real de execução em um computador Intel(R) Core i7 com 8Gb de memória RAM. Já os experimentos *online* foram processados por um computador Intel(R) Corei i5 com 16Gb de memória RAM embarcado no EspeleoRobô.

6.1 Comparação das Técnicas de LiDAR SLAM

A comparação das técnicas de LiDAR SLAM visa analisar os desempenhos das localizações e mapeamentos, de modo a determinar qual das estratégias é mais adequada para implementação no EspeleoRobô. Inicialmente foram realizadas simulações em uma caverna da DARPA com o objetivo de determinar a técnica que apresenta as estimativas de poses mais precisas e o mapa com menor erro, comparados com o *ground truth* fornecido pelo simulador. Em seguida, foram realizados experimentos em laboratório e campo utilizando os dado reais dos sensores do robô para avaliar a precisão da localização e do mapa por meio de métricas pré-determinadas na Seção 1. Por último, as técnicas LiDAR SLAM foram testadas em um ambiente subterrâneo mapeando uma das entradas da Mina

du Veloso com o objetivo de avaliar o desempenho em um ambiente mais representativo do real de operação do EspeleoRobô.

A técnica de melhor desempenho nas simulações da caverna DARPA foi utilizada como referência para comparação das demais técnicas nos experimentos em laboratório e campo, e em todas as análises foram comparadas também a pose 6D estimada pela odometria das rodas fundida com os dados da IMU por um filtro EKF.

6.1.1 Simulações em uma Caverna Virtual

As simulações utilizadas para a investigação das técnicas LiDAR SLAM foram realizadas em duas etapas, em um segmento curto e em um segmento longo, dentro de uma caverna utilizada na competição da DARPA *Subterranean Challenge* apresentada na Figura 29. Para ambas as etapas foi usado o modelo virtual do EspeleoRobô retratado na Seção 5.2.1 teleoperado desviando de obstáculos rochosos e mapeando o cenário. No segmento curto o robô percorreu um total de $21,48$ m em um trecho de subida, e no segmento longo $185,63$ m passando por pisos irregulares, subidas e descidas.

Para a análise das odometrias foram utilizadas as métricas $\mathcal{M}_1$ comparando o deslocamento total estimado por cada uma das técnicas LiDAR SLAM com o deslocamento dado pelo simulador, $\mathcal{M}_3$ mensurando o ruído da estimação, e $\mathcal{M}_4$ e $\mathcal{M}_5$ medindo o erro médio e a variância em relação ao *ground truth* fornecido pelo simulador. As análises dos mapas visam medir as suas precisões pelos histogramas de erros considerando a distância de cada ponto do mapa gerado com respeito a superfície mais próxima no modelo do cenário, conforme descrito na Seção 4.1.2.

6.1.1.1 Segmento Curto da Caverna

Os resultados obtidos no segmento curto da caverna virtual são apresentados na Figura 36, mostrando as posições 3D do robô em diferentes arranjos, onde as posições iniciais estão indicadas com um círculo preenchido e as finais com um círculo vazio, e na Tabela 1 destacando os melhores resultados marcados de azul.

Tabela 1 – Métricas para a simulação do segmento curto da caverna da DARPA.

SLAM/Métricas	$\mathcal{M}_1$[%]	$\mathcal{M}_3$[m]	$\mathcal{M}_4$[m]	$\mathcal{M}_5$[m]
LOAM-Velodyne	6,13	0,130	1,042	0,1723
LeGO-LOAM	0,4	0,027	0,072	0,002
HDL-Graph-SLAM	0,04	0,006	0,220	0,056
Rodas+IMU	7,22	0,014	2,762	3,884

Nesta simulação a técnica HDL-Graph-SLAM obteve melhores resultados para as métrica $\mathcal{M}_1$ e $\mathcal{M}_3$ com um erro de $0,04\%$ em relação ao deslocamento estimado por sua odometria e um ruído na casa dos milímetros. Porém a técnica LeGO-LOAM obteve um

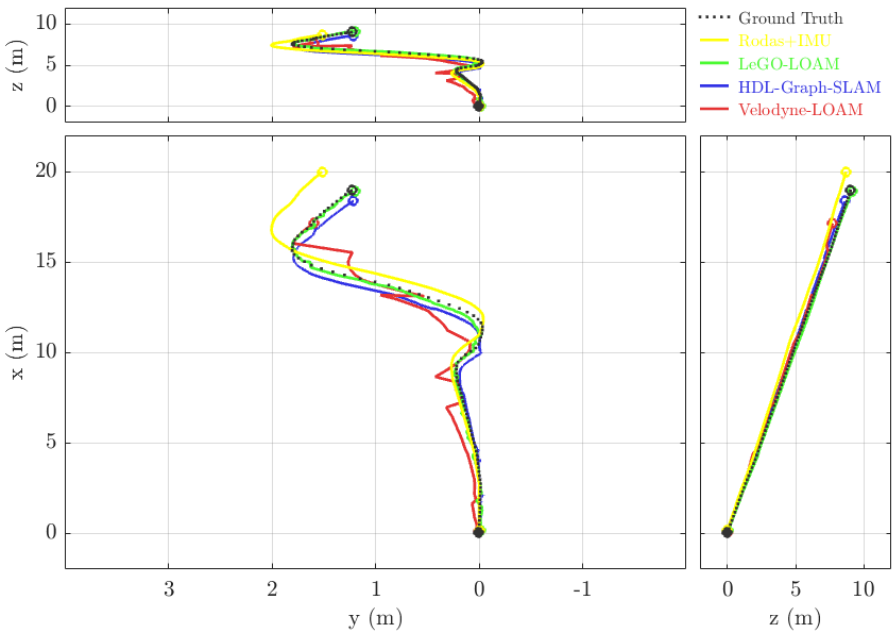


Figura 36 – Resultados para a simulação do segmento curto da caverna da Darpa.

melhor estimativa em comparação com o *ground truth* do simulador, com um erro médio de $7,2$ cm e variância de $0,2$ cm dados pelas métricas $\mathcal{M}_4$ e $\mathcal{M}_5$. Já o LOAM-Velodyne apresentou os piores resultados nesta simulação, com um erro de $6,13\%$ na estimação do deslocamento, erro médio de $1,04$ m e variância de $0,17$ m. Devido ao trecho ser de um subida, a odometria das rodas com IMU, descrita como Rodas+IMU, apresentou erro de escorregamento estimando um deslocamento $7,22\%$ superior que o fornecido pelo simulador. com erro médio de $2,76$ m e variância $3,88$ m.

A avaliação dos mapas é realizada com base nos histogramas de erros entre as nuvens de pontos e o cenário, calculados com base nas distâncias entre cada ponto e o plano do *mesh* mais próximo. Na Figura 39 são apresentados os mapas gerados por cada uma das técnicas LiDAR SLAM sobrepostos ao cenário, onde a escala de cores dos pontos é equivalente às cores apresentadas nos respectivos histogramas abaixo de cada mapa. Na Tabela 2 são apresentados os erros máximos computadas para cada técnica de LiDAR SLAM destacando os intervalos de confiança de 1σ, 2σ e 3σ, o erro máximo e o total de pontos dos mapas.

Nessa primeira análise o LeGO-LOAM apresentou os melhores resultados com um mapa composto por 31.239 pontos, onde $68,23\%$ (intervalo de 1σ) dos pontos da nuvem apresentam um erro inferior $2,1$ cm, $95,44\%$ (intervalo de 2σ) dos pontos possui um erro menor que $8,5$ cm, e $99,73\%$ (intervalo de 3σ) apresentam um erro menor que $23,5$ cm (intervalo de 3σ); o erro máximo foi de $27,8$ cm. A técnica HDL-Graph-SLAM apresentou resultantes intermediários estimando um mapa mais denso com 103.309 pontos, com um erro de $19,6$ cm para o intervalo de 1σ, $47,7$ cm para o intervalo de 2σ, e $72,9$ cm

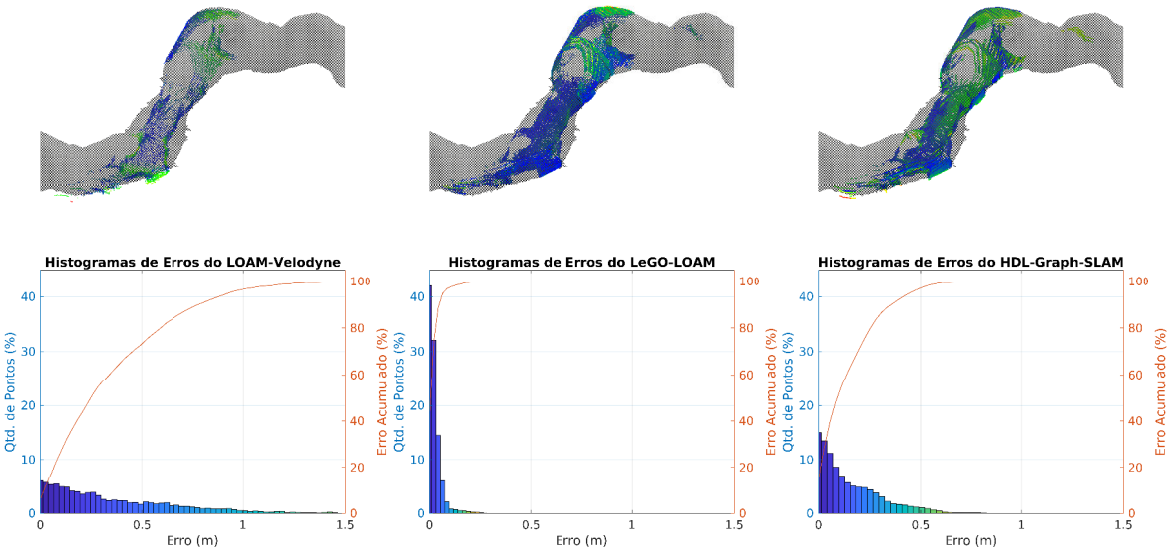


Figura 37 – Mapas gerados pelas técnicas LiDAR SLAM para o segmento curto da caverna Darpa em comparação com o *Mesh* do cenário.

para o intervalo 3σ, com erro máximo de $86,9$ cm. Por último, a técnica LOAM-Velodyne apresentou os piores resultados para o mapeamento do segmento curto da caverna DARPA, com um mapa composto por 8.115 pontos, com um erro de 44 cm para o intervalo de 1σ, $95,8$ cm para o intervalo de 2σ, e $1,42$ m para o intervalo 3σ, com erro máximo de $2,55$ m.

Tabela 2 – Erros dos mapas para segmento curto da caverna da DARPA.

SLAM/Métricas	1σ[m]	2σ[m]	3σ[m]	Erro Máximo[m]	Total de Pontos
LOAM-Velodyne	0,440	0,958	1,424	2,553	8.115
LeGO-LOAM	0,021	0,085	0,235	0,278	31.239
HDL-Graph-SLAM	0,196	0,477	0,729	0,869	103.309

Assim como na avaliação da odometria a técnica LeGO-LOAM obteve também os melhores resultados para o mapeamento. Nessa simulação do segmento curto da caverna da DARPA, a técnica LeGO-LOAM levou um menor tempo para processar a nuvem de pontos, avaliado desde a entrada da nuvem gerada a cada medição do sensor LiDAR até a construção do mapa, equivalente a $1,239 \pm 226$ s contra $1,708 \pm 836$ s do LOAM-Velodyne. O tempo médio gasto pelo HDL-Graph-SLAM para processar a nuvem de pontos e gerar o mapa foi de $1,593 \pm 833$ s; entretanto, a publicação do mapa só é realizada entre intervalos de deslocamento de 2 m do robô, sendo considerado um algoritmo *offline* de mapeamento.

6.1.1.2 Segmento Longo da Caverna

Os resultados das estimações das odometrias das técnicas de LiDAR SLAM para a simulação no segmento longo da caverna da DARPA são apresentados na Figura 38, ilustrando as localizações estimadas por cada um das técnica de LiDAR SLAM mais a odometria das rodas com IMU. Na Tabela 3 são descritos os resultados das métricas utilizadas na avaliação, onde os melhores resultados entre as técnicas de LiDAR SLAM estão marcados em azul e, quando a odometria das rodas mais IMU apresenta melhor precisão dentre todas a técnicas analisadas, o resultado está marcado em roxo.

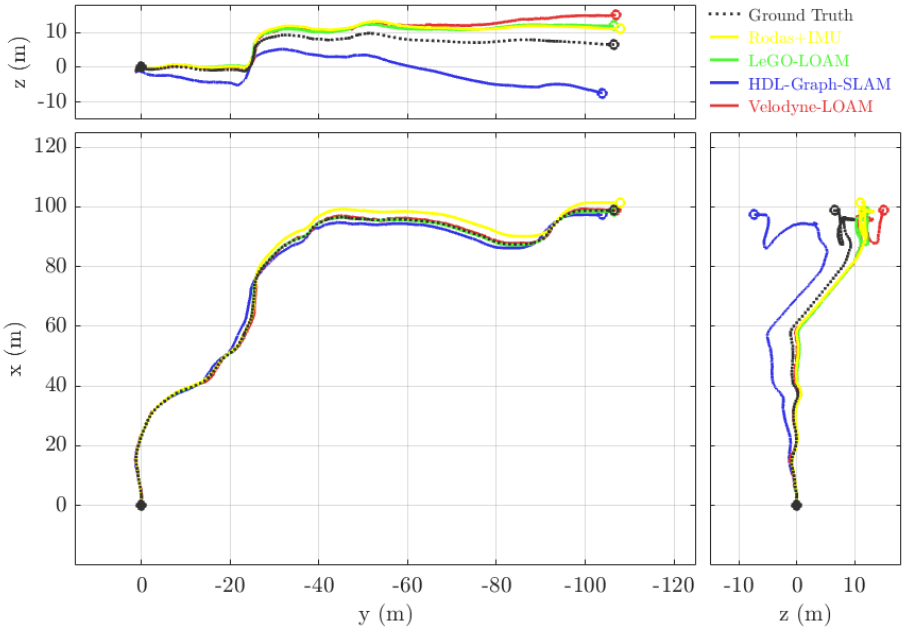


Figura 38 – Resultados para a simulação do segmento longo da caverna da Darpa.

Tabela 3 – Métricas para a simulação do segmento longo da caverna da DARPA.

SLAM/Métricas	$\mathcal{M}_1$[%]	$\mathcal{M}_3$[m]	$\mathcal{M}_4$[m]	$\mathcal{M}_5$[m]
LOAM-Velodyne	12,4	0,006	3,781	15,293
LeGO-LOAM	23,6	0,02	1,902	2,354
HDL-Graph-SLAM	11,6	0,001	7,294	30,450
Rodas+IMU	2,2	2,35e-03	2,762	3,884

A odometria das rodas, devido ao processo de integração, possui baixo nível de ruído, na casa dos milímetros como mostra a métrica $\mathcal{M}_3$, e possui uma boa estimação do deslocamento do robô dado pela métrica $\mathcal{M}_1$ com um erro acumulado de $2,2\%$. Diferentemente, as técnicas LiDAR SLAM apresentaram um nível de ruídos na faixa de centímetros, ainda baixo, mas o suficiente para obterem uma estimação do deslocamento com erro acumulado de $11,6\%$, $12,4\%$ e $23,6\%$ e para as técnicas HDL-Graph-SLAM, LOAM-Velodyne e LeGO-LOAM respectivamente. Este é um dos piores resultados da

técnica LeGO-LOAM, porém, é possível verificar a relação direta entre o ruído e o erro acumulado dado pelas métricas $\mathcal{M}_1$ e $\mathcal{M}_3$, como pode ser visto na Tabela 3, onde quanto maior o nível de ruído maior o erro na estimação do deslocamento do robô. Contudo, esse tipo de erro pode ser facilmente resolvido com a aplicação de um filtro.

Das 3 técnicas LiDAR SLAM analisadas, a que apresentou uma precisão melhor em relação ao *ground truth* foi a LeGO-LOAM com um erro médio de posição 3D de aproximadamente $1,90$ m com variância de $2,35$ m, dados pelas métricas $\mathcal{M}_4$ e $\mathcal{M}_5$, seguida da odometria as rodas, fundida com a IMU, com um erro médio de $2,76 \pm 1,97$ m. As técnicas LOAM-Velodyne e HDL-Graph-SLAM apresentaram maiores erros, com valores de $3,78 \pm 3,91$ m e $7,29 \pm 5,52$ m, respectivamente.

Para uma análise mais detalhada da precisão são apresentados na Tabela 4 os erros e desvios padrões de cada componente da pose estimada pelas técnicas LiDAR SLAM e o Filtro EKF (rodas+IMU). Os resultados apresentados confirmam a melhor precisão das posições e orientações estimadas pelo LeGO-LOAM em comparação com as demais técnicas de LiDAR SLAM. As componentes da pose ϕ_L, θ_L e $p_{\mathrm{L},z}$, que são estimadas por meio dos pontos presentes em planos como mostra a Subseção 4.2.2, obtiveram um erro maior que as componentes $p_{\mathrm{L},x}$, $p_{\mathrm{L},y}$ e ψ_L, devido o cenário da cavernas possuir superfícies completamente irregulares em algumas partes do trajeto realizado pelo robô, dificultando a identificação de pontos em planos. Contudo, a pose gerada pelo EKF (rodas+IMU) obteve uma melhor estimação para a orientação com erros de menos de 1°, devido a fusão direta da odometria das rodas com a IMU pelo EKF. Diferentemente, a técnica LeGO-LOAM utiliza os deslocamentos dados pela IMU para ajustar as nuvens de pontos antes de realizar as correspondências entre *features* utilizados na estimação da pose.

Tabela 4 – Média e desvio padrão dos erros das posições e orientações da técnicas LiDAR SLAM em relação ao *ground truth* para a simulação do segmento longo da caverna da DARPA.

Valores/SLAM	LOAM-Velodyne	LeGO-LOAM	HDL-Graph-SLAM	Rodas+IMU
$\mu_x \pm \sigma_x$[m]	$1,522 \pm 1.771$	$0,096 \pm 0,089$	$2,124 \pm 1.895$	$1,605 \pm 1.179$
$\mu_y \pm \sigma_y$[m]	$2,554 \pm 3.075$	$0,099 \pm 0.084$	$3,259 \pm 3.303$	$0,525 \pm 0.517$
$\mu_z \pm \sigma_z$[m]	$2,097 \pm 2.283$	$1,921 \pm 1.564$	$5,609 \pm 4.750$	$2,155 \pm 1.537$
$\mu_\phi \pm \sigma_\phi$[°]	$92,010 \pm 0.725$	$1,620 \pm 0.110$	$10,691 \pm 1.136$	$0,657 \pm 0.032$
$\mu_\theta \pm \sigma_\theta$[°]	$13,639 \pm 1.912$	$1,737 \pm 0.148$	$8,146 \pm 1.108$	$0,851 \pm 0.032$
$\mu_\psi \pm \sigma_\psi$[°]	$89,054 \pm 2.698$	$0,412 \pm 0.045$	$14,124 \pm 2.391$	$0,124 \pm 0.018$

Os resultados referentes aos mapas gerados no segmento longo da caverna da DARPA podem ser vistos na Figura 39, e os erros estimados na Tabela 3. Novamente a técnica LeGO-LOAM apresenta um melhor resultado para o mapeamento. Para um total de 117.241 pontos estimados no mapa, os resultados obtidos pelo LeGO-LOAM papresentam um erro menor que $75,1$ cm para $68,23\%$ dos pontos da nuvem, $2,63$ m para $95,44\%$ dos pontos, e $3,5$ m para $99,73\%$ dos pontos, com um erro máximo de $3,89$ m.

Diferentemente da simulação do segmento curto, o LOAM-Velodyne apresentou resultados medianos dentre as 3 técnicas LiDAR SLAM investigadas. Com um mapa composto por 247.480 pontos o LOAM possui um erro de $81,6$ cm para o intervalo de 1σ, $3,95$ m para o intervalo de 2σ, e $5,85$ m para o intervalo de 3σ, com um erro máximo de $6,81$ m. Já a técnica HDL-Graph-SLAM apresentou os piores resultados, num mapa formado por 342.056 pontos, com erros de $1,64$ m para o intervalo de 1σ, $4,78$ m para 2σ e $17,31$ para 3σ, com erro máximo de $18,96$ m.

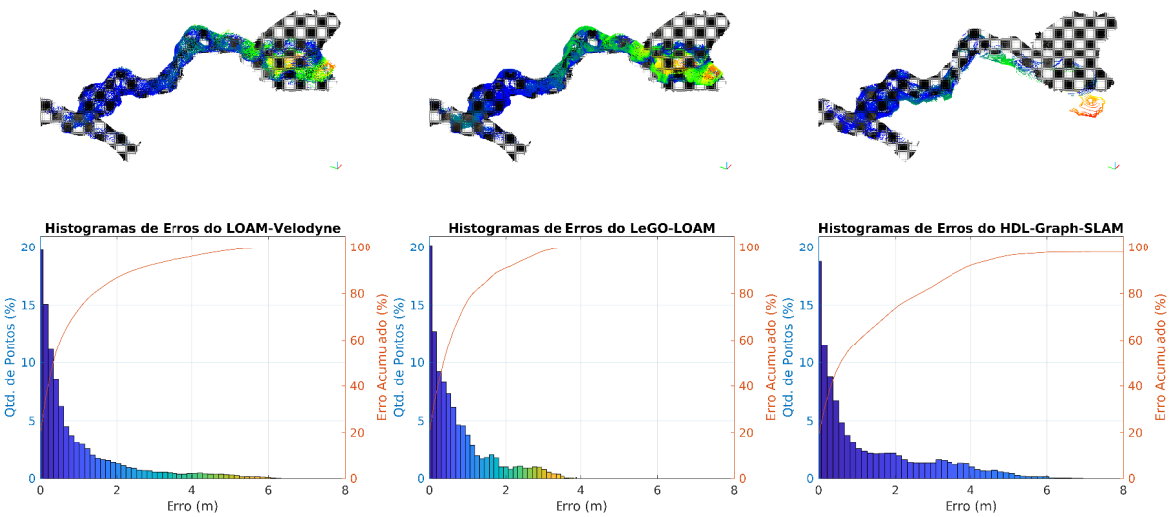


Figura 39 – Mapas gerados pelas técnicas LiDAR SLAM no experimento da caverna Darpa em comparação com o *Mesh* do cenário.

Tabela 5 – Erros dos mapas completos da Caverna da DARPA.

SLAM/Métricas	1σ[m]	2σ[m]	3σ[m]	Erro Máximo[m]	Total de Pontos
LOAM-Velodyne	0,816	3,946	5,851	6,803	247.480
LeGO-LOAM	0,751	2,627	3,503	3,878	177.241
HDL-Graph-SLAM	1,642	4,776	17,314	18,956	343.056

A análise realizada com o segmento longo da caverna da DARPA ilustra como os problemas da localização e mapeamentos estão intrinsecamente ligados. Todas as técnicas de LiDAR SLAM apresentaram erros maiores na estimativa das poses, como mostram as Tabelas 1 e 3, e consequentemente na construção dos mapas. Isso fica bem evidenciado pela técnica HDL-Graph-SLAM que apresentou maiores erros em relação *ground truth*, acarretando em erros maiores na construção do mapa.

Os resultados obtidos pelo LeGO-LOAM mostram que esta técnica possui um melhor desempenho com boa precisão para a estimação da odometria e para a geração dos mapas, além de um menor tempo de processamento. Sendo assim, a técnica foi utilizada

como referência nos experimentos em laboratório para comparação das diferenças entre as estimações das odometrias, e na Mina du veloso para comparação também das diferenças entre os mapas, devido a falta de uma informação válida de *ground truth*.

6.1.2 Experimentos em Laboratório

Após a análise por simulação, as técnicas LiDAR foram analisadas em laboratório, sendo executadas com os dados coletados da plataforma robótica real descrita na Seção 5.2.2. Os ambientes utilizados foram corredores da Escola de Engenharia da UFMG que podem ser vistos na Figura 32. Os experimentos foram separados em duas etapas, uma com o EspeleoRobô percorrendo um corredor reto, e outra representada por um circuito fechado formado por corredores. Devido à interferência nos magnetômetros na maioria dos ambientes de trabalho do EspeleoRobô, os experimentos com a plataforma real foram executados calculando a orientação do robô considerando somente as informações dos acelerômetros e giroscópios disponíveis na IMU.

6.1.2.1 Corredor Reto

Este experimento avalia a estimação das posições computadas enquanto o robô se move dentro de um corredor reto. Controlado remotamente por um operador, o robô realiza um percurso aproximadamente em linha reta de $22,3$ m, vira 180° e retorna para o ponto de partida executando um percurso com perfil senoidal. A métrica $\mathcal{M}_1$ com $l_{max} = 22,3$ m e a métrica $\mathcal{M}_2$ são utilizadas para avaliar a precisão das posições estimadas. O nível de ruído associado é quantificado pela métrica $\mathcal{M}_3$. As métricas $\mathcal{M}_4$ e $\mathcal{M}_5$ fornecem uma estimativa do erro médio e variância entre os resultados obtidos pelas diferentes estratégias utilizando como referência a técnica LeGO-LOAM. A Figura 40 e Tabela 6 apresentam os resultados relativos às localizações do robô computadas por cada técnica.

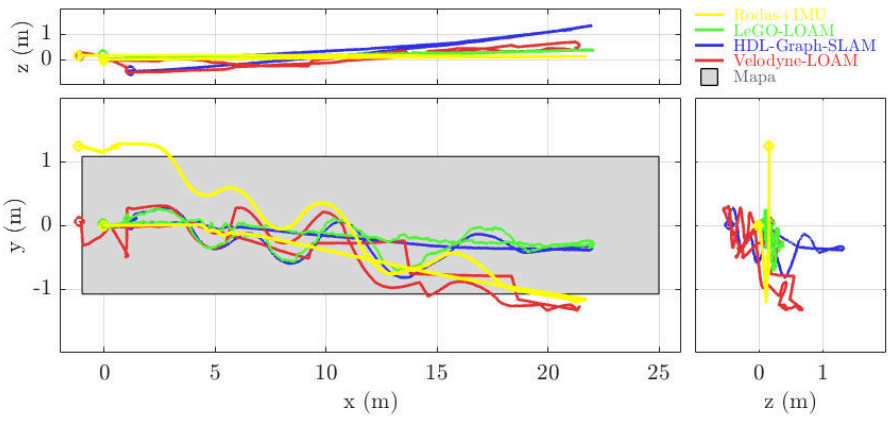


Figura 40 – Resultados das posições estimadas no experimento onde o robô percorre um corredor reto.

Tabela 6 – Resultado das métricas para o experimento onde o robô percorre um corredor reto.

SLAM/Métricas	$\mathcal{M}_1$[%]	$\mathcal{M}_2$[m]	$\mathcal{M}_3$[m]	$\mathcal{M}_4$[m]	$\mathcal{M}_5$[m]
LOAM-Velodyne	45,3	0,798	0,027	0,587	0,099
LeGO-LOAM	5,9	0,185	0,012	-	-
HDL-Graph-SLAM	0,9	1,272	0,007	0,460	0,089
Rodas+IMU	2,9	1,721	0,001	0,634	0,186

A odometria das rodas com IMU novamente obteve um menor nível de ruído dado pela métrica $\mathcal{M}_3$, porém, devido ao escorregamento das rodas ao realizar curvas esta métrica acumula erros maiores, conforme destacado pela métrica $\mathcal{M}_1$ que dá a porcentagem da diferença do caminho estimado no percurso de ida, e pela métrica $\mathcal{M}_2$ que mensura o acúmulo de erro dado pela diferença da posição final e inicial. Das técnicas LiDAR SLAM, o LeGO-LOAM obteve um menor erro ao retornar à posição de partida, medindo em $18,5$ cm de diferença. Em contrapartida, a técnica HDL-Graph-SLAM obteve uma estimativa do caminho de ida melhor com erro de $0,9\%$ do percurso medido, justificado pelo baixo nível de ruído da odometria. Para este experimento as técnicas LeGO-LOAM e HDL-Graph-SLAM obtiveram resultados muito próximos durante todo o percurso, conforme indicado pelas métricas $\mathcal{M}_4$ e $\mathcal{M}_5$, que mensuram a média e a variância da diferença medindo $46,00 \pm 8,90$ cm.

Os modelos gerados do ambiente são apresentados na Tabela 7 e Figura 41, que ilustra os mapas sobrepostos ao modelo 3D real do corredor (Figura 32(b)). Nesse experimento, o mapa gerado pela técnica LeGO-LOAM é composto por 14.153 pontos e possui um erro menor que $24,8$ cm para $68,23\%$ (1σ) dos pontos, $74,4$ cm para $95,44\%$ (2σ) dos pontos, e $1,99$ m para $99,73\%$ (3σ) dos pontos, com erro máximo de $7,13$ m. O mapa gerado pela técnica HDL-Graph-SLAM com 31.271 pontos apresentou resultados muito próximos para os intervalos de confiança de 1σ com um erro de $29,7$ cm e 2σ com erro de $81,7$ cm. Porém, a técnica apresentou melhores resultados para o intervalo de 3σ com erro de $1,78$ m, além de um menor erro máximo com valor de $4,23$ m. Por último a técnica LOAM-Velodyne gerou um mapa composto por 3.798 pontos apresentando um erro de $66,6$ cm para o intervalo de 1σ, $1,83$ m para 2σ, e $2,91$ m para 3σ, com erro máximo de $4,91$ m.

Tabela 7 – Erros dos mapas gerados pelas técnicas LiDAR SLAM com respeito ao mapa real para o experimento onde o robô percorre um corredor reto.

SLAM/Métricas	1σ[m]	2σ[m]	3σ[m]	Erro Máximo[m]	Total de Pontos
LOAM-Velodyne	0,666	1,830	2,911	4,907	3.798
LeGO-LOAM	0,248	0,744	1,985	7,132	14.153
HDL-Graph-SLAM	0,2976	0,817	1,781	4,229	31.271

Apesar da pouca diferença em relação ao HDL-Graph-SLAM, o mapa do LeGO-

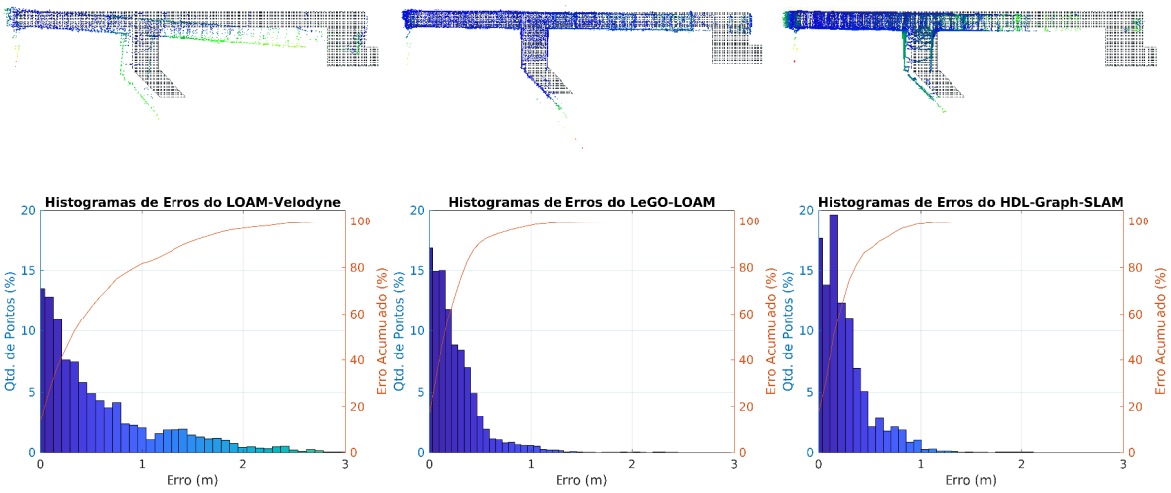


Figura 41 – Comparação dos mapas gerados pelas técnicas LiDAR SLAM com respeito ao mapa real para o experimento onde o robô percorre um corredor reto.

LOAM possui pontos uniformemente espalhados pelo ambiente mapeado devido ao uso do filtro *voxel grid* implementado por meio da técnica *Baes-Tree*, além de utilizar todos os pontos, incluindo os *outliers*, integrados ao mapa. Desta forma, o seu mapa possui pontos mapeados fora dos limites do *ground truth*. Diferentemente, a técnica HDL-Graph-SLAM, mesmo possuindo um mapa mais denso, limita os pontos mapeados nas proximidades da pose estimada do robô ao eliminar os pontos mais distantes. Portanto, os pontos que representam o mapa estão concentrados no piso e a parte inferior das paredes do ambiente percorrido, limitando os erros estimados.

6.1.2.2 Corredores Formando um Circuito Fechado

Neste experimento o robô, também teleoperado, executa duas voltas em um caminho fechado nos corredores do prédio da da Escola de Engenharia da UFMG, percorrendo uma distância total de 270 m e parando na mesma posição de início. Os resultados obtidos podem ser vistos na Figura 42 e Tabela 8, que ilustram as odometrias estimadas e as respectivas métricas calculadas.

Com respeito ao cálculo da distância percorrida pelo robô expressa pela métrica $\mathcal{M}_1$, agora com $l_{max} = 270$ m, a odometria das rodas obteve neste experimento a pior precisão, estimando uma distância $9,2\%$ superior ao caminho executado, justificado pelo escorregamento das rodas ao realizar curvas. A técnica LeGO-LOAM obteve uma melhor precisão nestes resultados com um erro de $1,8\%$ ($\mathcal{M}_1$) do caminho total percorrido e uma diferença de $37,1$ cm entre posição final e inicial ($\mathcal{M}_2$). Novamente a técnica HDL-Graph-SLAM possui resultados próximos do LeGO-LOAM, como pode ser observado pelas métricas $\mathcal{M}_4$ e $\mathcal{M}_5$, porém obteve a pior precisão referente ao acúmulo de erro dado pela métrica $\mathcal{M}_2$, com uma diferença de $4,98$ m quando o robô retorna para o ponto de partida. Já a técnica LOAM-Velodyne obteve novamente resultados mais distantes da

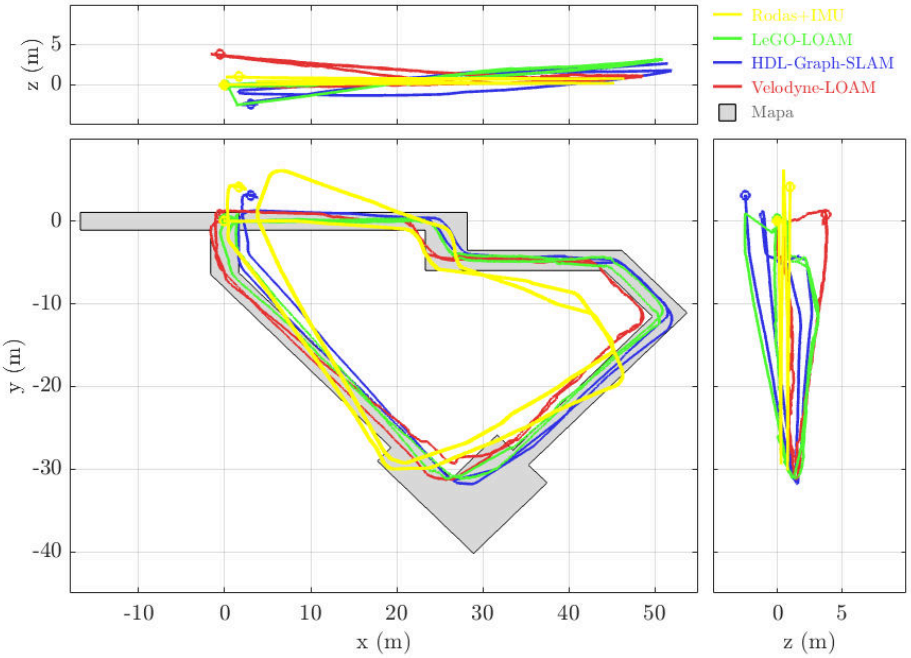


Figura 42 – Resultados das posições estimadas no experimento onde o robô percorre corredores em um circuito fechado.

técnica LeGO-LOAM com erro de $2,09 \pm 1,45$ m de diferença, dado pelas métricas $\mathcal{M}_4$ e $\mathcal{M}_5$, e um erro acumulado de $3,91$ m entre a posição final e inicial (métrica $\mathcal{M}_2$). Os resultados inferiores são devido a não aplicação da técnica de *loop closure*.

Tabela 8 – Métricas para o experimento onde o robô percorre corredores em circuito fechado.

SLAM/Métricas	$\mathcal{M}_1$[%]	$\mathcal{M}_2$[m]	$\mathcal{M}_3$[m]	$\mathcal{M}_4$[m]	$\mathcal{M}_5$[m]
LOAM-Velodyne	2,5	3,908	0,049	2,096	2,093
LeGO-LOAM	1,8	0,371	0,046	-	-
HDL-Graph-SLAM	0,8	4,986	0,019	1,226	0,802
Rodas+IMU	9,2	4,605	0,003	2,362	2,023

Os resultados dos mapeamentos podem ser vistos na Figura 43, que mostra os mapas das técnicas LiDAR SLAM sobrepostos ao mapa real na parte superior e os histogramas de erros na parte inferior, e na Tabela 9. Para a avaliação dos mapas foi determinado uma região próxima ao mapa real dos corredores limitando a distância de análise do mesmo modo para as 3 técnicas LiDAR SLAM, eliminando pontos mapeados fora dos limites do prédio.

Para o experimento dos corredores em circuito fechado o HDL-Graph-SLAM apresentou melhores resultados para o mapa do que a técnica LeGO-LOAM. Com um mapa composto por 250.503 pontos a técnica HDL-Graph-SLAM possui um erro de $70,7$ cm para um intervalo de 1σ, $1,62$ m para o intervalo de 2σ, e $3,33$ m para o intervalo de

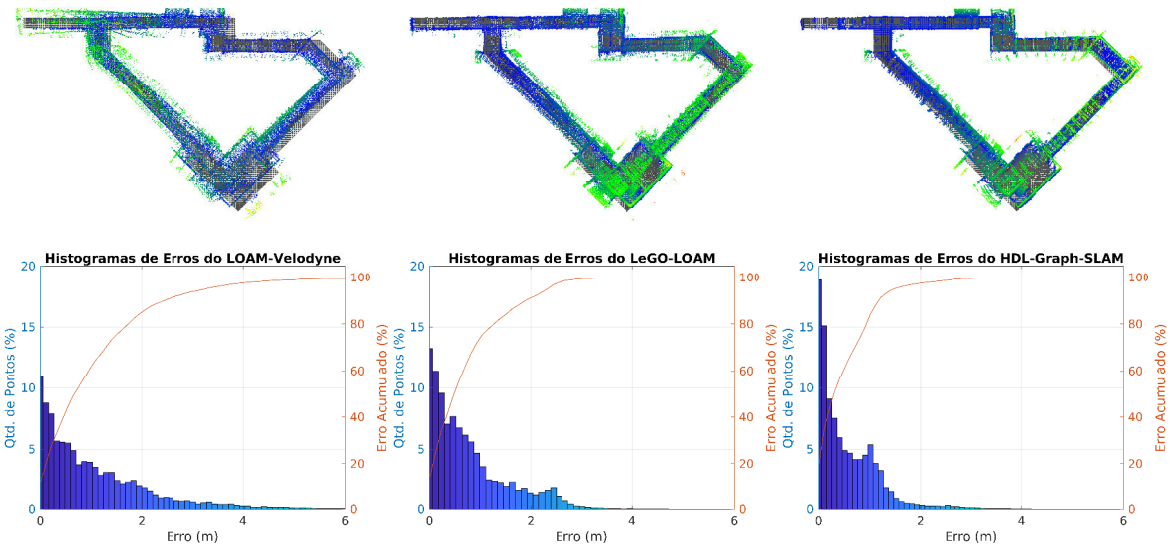


Figura 43 – Comparação dos mapas gerados pelas técnicas de LiDAR SLAM com respeito ao mapa real para o experimento onde o robô percorre corredores formando um circuito fechado.

Tabela 9 – Erros dos mapas gerados pelas técnicas de LiDAR SLAM com respeito ao mapa real para o experimento onde o robô percorre corredores em circuito fechado.

SLAM/Métricas	1σ[m]	2σ[m]	3σ[m]	Erro Máximo[m]	Total de Pontos
LOAM-Velodyne	1,292	3,446	6,892	11,091	44.827
LeGO-LOAM	0,931	2,444	3,608	7,333	83.669
HDL-Graph-SLAM	0,707	1,616	3,333	6,364	250.503

3σ, com erro máximo de $6,36$ m. Já a técnica LeGO-LOAM forneceu um mapa composto por de um total de 83.669 de pontos com resultados muito próximos com um erro de $93,1$ cm para o intervalo de 1σ, $2,44$ m para o intervalo de 2σ, e $3,61$ m para 3σ, com erro máximo de $7,33$ m. Novamente a técnica LOAM-Velodyne apresentou os piores resultados para o mapeando, justificado pela não aplicação de técnicas de *loop closure*. Com um mapa composto por 44.827 pontos, a técnica LOAM-Velodyne obteve um erro menor que $1,29$ m para o intervalo de 1σ, $3,45$ m para 2σ, e $6,89$ m para 3σ, com erro máximo de $11,1$ m.

De forma geral, a técnica LeGO-LOAM obteve um conjunto de resultados mais consistentes dentre os dois experimentos realizados em laboratório com relação às métricas utilizadas. A técnica HDL-Graph-SLAM obteve resultados próximos, porém, devido à construção do mapa ser *offline* e, nos casos em que o mapa é muito denso, exige mais recursos computacionais o que acaba afetando a precisa da odometria. Em razão da não aplicação de qualquer metodologia de *loop closure*, a técnica LOAM-Velodyne resume o problema do SLAM ao da odometria LiDAR, segundo [Cadena et al., 2016], e consequentemente causa deformações no mapa e acúmulos de erros na estimação da pose.

6.1.3 Experimento em Ambiente Subterrâneo

Para testar as técnicas de LiDAR SLAM em um ambiente subterrâneo foi realizado um experimento na Mina du Veloso, representando um ambiente real de operação do EspeleoRobô no escopo da espeleologia. Pertencente ao complexo de mineração do Veloso na cidade de Ouro Preto, esta foi uma mina de extração de ouro durante os séculos XVIII e XIX. Este ambiente confinado apresenta estruturas irregulares, pisos escorregadios e ausência de sinal de GPS. A Figura 35 da Seção 5.3.2 apresenta vista internas das galerias e corredores mapeados pelo EspeleoRobô.

Neste experimento o robô executa um caminho curto composto por segmentos retos com pequenas defasagens de orientação, de forma a mapear uma das galerias da mina. Para avaliar o desempenho das técnicas de LiDAR SLAM foram consideradas as métricas $\mathcal{M}_3$, mensurando o ruído nas odometrias, e $\mathcal{M}_4$ e $\mathcal{M}_5$ para quantificar a variação da localização do robô estimada pelas técnicas com respeito ao LeGO-LOAM. A Figura 44 ilustra o caminho estimado por cada técnica, e a Tabela 10 apresenta as respectivas métricas calculadas.

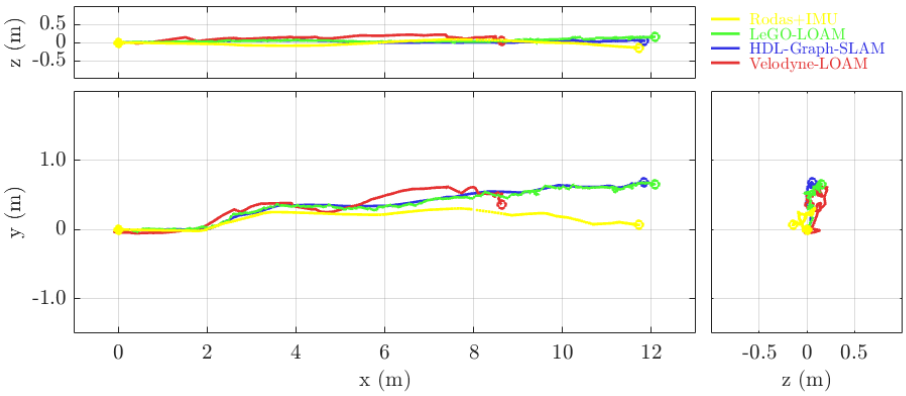


Figura 44 – Resultados das posições estimadas no experimento da Mina du Veloso.

Tabela 10 – Métricas para o experimento na Mina Du Veloso.

SLAM/Métricas	$\mathcal{M}_3$[m]	$\mathcal{M}_4$[m]	$\mathcal{M}_5$[m]
LOAM-Velodyne	0,012	0,838	1,361
LeGO-LOAM	0,010	-	-
HDL-Graph-SLAM	0,008	0,072	0,009
Rodas+IMU	9e-4	0,214	0,039

A odometria das rodas com IMU resultou em pequeno erro médio e variância com respeito ao LeGO-LOAM, diferentemente dos resultados obtidos em laboratório. Isto ocorreu devido ao percurso realizado pelo robô não possuir curvas acentuadas que aumentam o escorregamento das rodas e acarretam maiores erros de orientação. A técnica LOAM-Velodyne obteve um erro médio ($\mathcal{M}_4$) de $0,83$ m, porém estimou um percurso

menor com uma diferença de aproximadamente 3 m em relação ao LeGO-LOAM, conforme ilustrado na Figura 44. As técnicas HDL-Graph SLAM e LeGO-LOAM obtiveram novamente resultados muito próximos, com níveis de ruído ($\mathcal{M}_3$) na casa dos centímetros, erro médio ($\mathcal{M}_4$) de $21,4$ cm e variância ($\mathcal{M}_5$) de $3,9$ cm.

A Figura 45 ilustra os mapas gerados pelas técnicas de LiDAR SLAM LOAM-Velodyne e HDL-Graph-SLAM sobrepostos ao mapa do LeGO-LOAM, apresentado na cor cinza. Nesse experimento a técnica HDL-Graph-SLAM gerou um mapa com 7.460 pontos, semelhante ao fornecido pelo LeGO-LOAM, com erros menores que $4,7$ cm para o intervalo de 1σ, $13,4$ cm para o intervalo de 2σ, e $32,2$ cm para o intervalo de 3σ, com erro máximo de $43,2$ cm. Mais uma vez a técnica LOAM-Velodyne apresentou os piores resultados para o mapeando. Com um mapa composto por 3.595 pontos, a técnica LOAM-Velodyne obteve um erro menor que $13,9$ cm para o intervalo de 1σ, $60,4$ cm m para 2σ, e $1,35$ m para 3σ, com erro máximo de $2,55$ m.

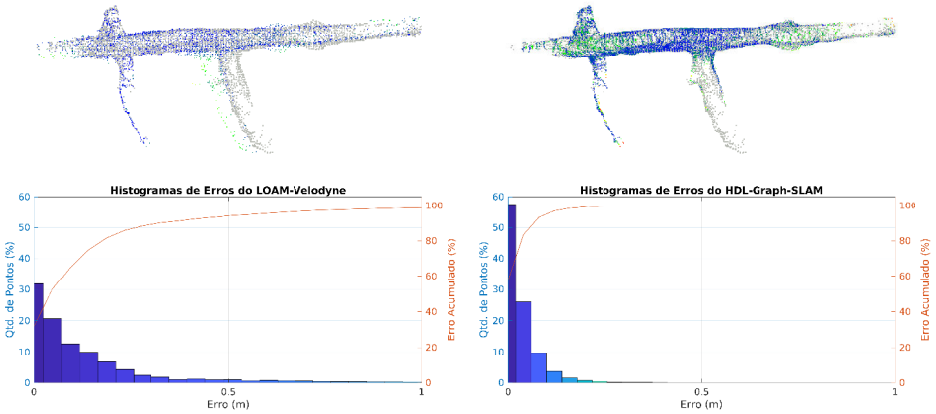


Figura 45 – Comparação dos mapas para o experimento da Mina du Veloso com respeito à técnica LeGO-LOAM.

Tabela 11 – Erros dos mapas para o experimento da Mina du Veloso com respeito ao mapa gerado pela técnica LeGO-LOAM.

SLAM/Métricas	1σ[m]	2σ[m]	3σ[m]	erro Máximo[m]	Total de Pontos
LOAM-Velodyne	0,139	0,604	1,347	2,554	3.595
HDL-Graph-SLAM	0,047	0,134	0,322	0,432	14.920

6.2 Experimentos *Online* com o Espeleo-LeGO-LOAM

Após realizada a investigação das técnicas de LiDAR SLAM por meio dos experimentos descritos na Seção 6.1, foi verificado que a técnica LeGO-LOAM obteve melhores

resultados com boas precisões para a odometria e o mapa do ambiente. Portanto, foram realizadas modificações no algoritmo original para implementação e uso no EspeleoRobô, conforme descrito na Seção 4.2, sendo este renomeado para Espeleo-LeGO-LOAM. Alguns experimentos foram realizados com o LiDAR SLAM embarcado no EspeleoRobô, de modo a validar a aplicação da técnica embarcada no robô fornecendo dados de forma *online* para os demais algoritmos do sistema de navegação autônoma. Esta seção apresenta os resultados da execução do pacote Espeleo-LeGO-LOAM mapeando um ambiente *indoor*, nos saguões superior e inferior do auditório da Escola de Engenharia da UFMG, como mostra a Figura 33, percorrendo os dois andares passando por uma rampa que liga ao pátio superior; um ambiente *outdoor* em uma área arborizada nas proximidades do restaurante da UFMG, exibido na Figura 34; e em um ambiente subterrâneo na Mina du Veloso (Figura 35), desta vez percorrendo mais galerias do que no experimento descrito da Seção 6.1.3.

6.2.1 Experimento *Indoor*

Para o experimento realizado no saguão do auditório da Escola de Engenharia da UFMG o robô foi operado no modo autônomo, como apresentado em [Azpúrua et al., 2021]. Para isso, as informações da pose e do mapa estimados pelo Espeleo-LeGO-LOAM foram utilizadas pelos algoritmos de controle de navegação e planejamento de caminhos.

O robô inicia o percurso do saguão inferior do auditório, passa pela rampa lateral até chegar no pátio superior interno. O trajeto é determinado selecionando pontos a serem explorados dentro do mapa gerado pelo LiDAR SLAM, em seguida planejando o caminho a ser seguido, e por fim comandando o robô utilizando um controle de navegação por campos vetoriais, como apresentado em [Rezende et al., 2020]. A Figura 46 apresenta a mapa final do experimento com as vistas superiores e em perspectiva, localizadas na parte superior da imagem, e 3 vistas internas, uma do saguão inferior, uma da rampa e outra da parte superior do saguão. O mapa gerado possui 299.515 pontos, e o caminho realizado pelo robô está marcado em branco.

Nesse experimento é possível observar pela vista superior do mapa, apresentada no canto superior esquerdo da Figura 46, que durante o trajeto do robô pela rampa lateral, devido a homogeneidade da estrutura com poucas *features* geométricas, houve uma subestimação da odometria. Isso é evidenciado pelo desalinhamento da parede superior e inferior dos saguões, marcados na figura por duas linhas amarelas dentro da região do círculo vermelho. Para esse problema de subestimação será apresentado na Seção 6.3 a aplicação de um filtro EKF descrito na Seção 4.3, corrigindo tanto a estimativa da odometria quanto a deformação do mapa.

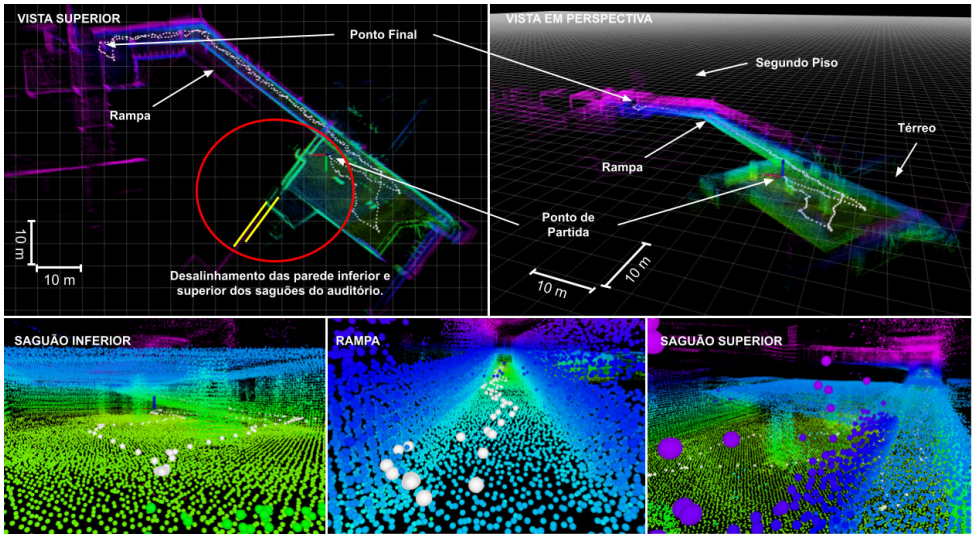


Figura 46 – Aplicação do LiDAR SLAM para navegação autônoma no prédio da Escola de Engenharia da UFMG.

6.2.2 Experimentos *Outdoor*

Os experimentos *outdoors* foram realizados nas proximidades do restaurante da UFMG, onde o robô executou o algoritmo de navegação por campos vetoriais utilizando os dados de posição e orientação do LiDAR SLAM. Para isso, foram utilizadas 3 curvas de referência para a navegação: uma retangular de bordas arredondadas, uma lemniscata de Bernoulli e uma elipse. A Figura 47 mostra os mapas resultantes onde é possível observar uma vista em perspectiva e uma vista superior para cada curva. No experimento do retângulo o mapa gerado possui 821.552 pontos, no lemniscata de Bernoulli o mapa possui 583.123 pontos, e no da elipse possui 590.202 pontos.

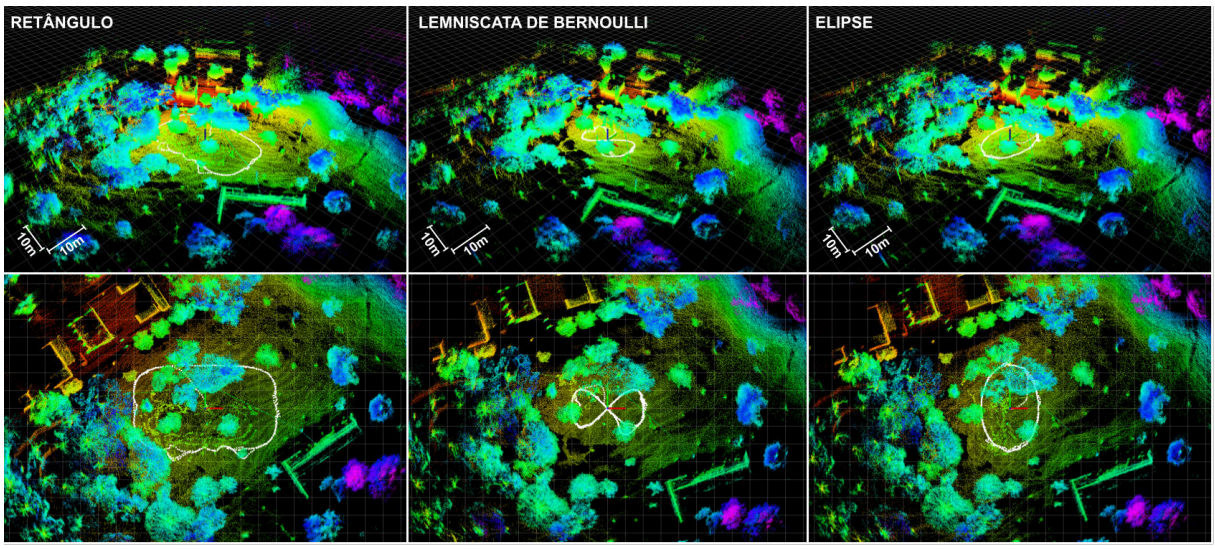


Figura 47 – Aplicação do LiDAR SLAM para o controle de navegação por campos vetoriais nas proximidades do restaurante da UFMG.

Uma observação importante sobre estes experimentos é que o ambiente utilizado possui muitas árvores estreitas e próximas umas das outras em algumas partes do percurso do robô. Isso pode gerar erros de duplicidade nas correspondências entre as *features* pela odometria LiDAR, diminuindo sua eficiência como descrito em [Sun et al., 2018], assim como pode causar erros na identificação de *loop closure*, provocando deformações no mapa. Contudo, a técnica LiDAR SLAM adaptada apresentou um bom desempenho ao realizar os experimentos *outdoors*, permitindo que o robô executasse o percurso desejado de forma correta.

6.2.3 Experimento em Ambiente Subterrâneo

Em um experimento diferente do realizado na Seção 6.1.3, o EspeleoRobô percorreu aproximadamente 55 m de túneis conectados dentro das galerias da Mina du Veloso, sendo operado no modo autônomo utilizando a pose e o mapa da técnica LiDAR SLAM, da mesma forma que no experimento *indoor* descrito na Subseção 6.2.1. Este cenário possui muitos obstáculos como rochas e buracos, além de solos escorregadios, irregulares e inclinados, fazendo com que o robô percorra o trajeto com muitos solavancos e trepidação, o que pode oferecer dificuldades na estimação da odometria. Contudo, a técnica LiDAR SLAM foi executada de forma correta, permitindo a operação autônoma, como apresentado em [Azpúrua et al., 2021]. O mapa final gerado possui 84.644 pontos e é apresentado na Figura 48 com uma vista superior, uma em perspectiva e 3 vistas internas.

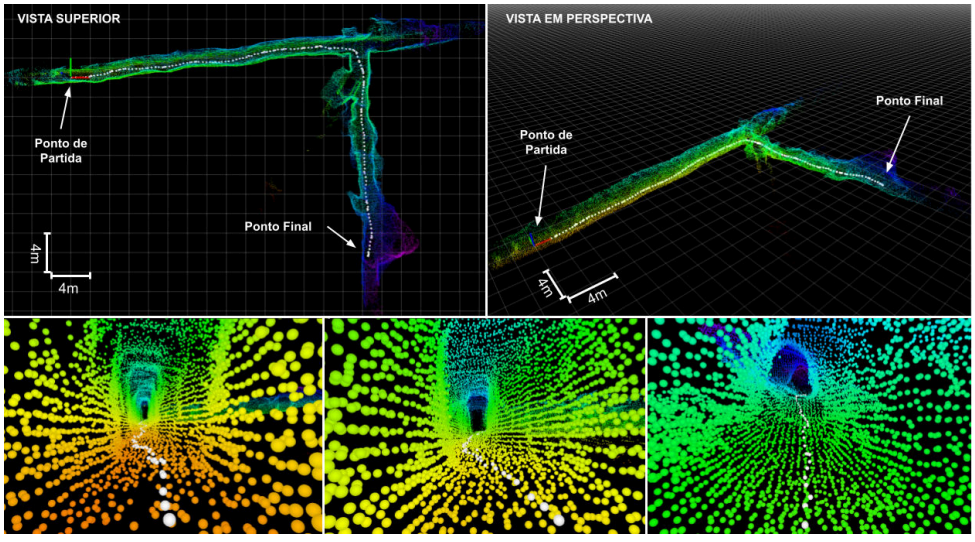


Figura 48 – Resultados na Mina du Veloso com o LeGO-LOAM.

6.3 Análise do Filtro EKF Integrado ao LiDAR SLAM

A técnica LiDAR SLAM adaptada e implementada no EspeleoRobô, assim como as demais técnicas baseadas em sensores LiDAR, apresentam erros na estimação da pose em ambientes com poucas *features* geométricas como dutos longos, galerias e corredores extensos e lisos, como foi observado no experimento autônomo apresentado na Subseção 6.2.1. Por exemplo, nos experimentos realizados nos corredores da Escola de Engenharia foram identificados em média 200 *features* de bordas e 1750 *features* de planos, nos experimentos realizados próximo ao auditório foram identificados 430 *features* de bordas e 3132 *features* de plano, e na simulação da caverna da DARPA foram identificados 380 *features* de bordas e 3600 *features* de planos. Já durante as simulações em dutos e galerias, esses valores foram em média 80% menores para *features* de bordas e 40% menores para *features* de planos.

Para lidar com esse problema, foi proposto e implementado um filtro EKF para fundir os dados da odometria das rodas com os dados da IMU e a odometria LiDAR, aplicado entre as etapas de *Back-End* e *Front-End* do LiDAR SLAM utilizando covariâncias adaptativas, conforme descrito na Seção 4.3. Os resultados do filtro EKF integrando a odometria das rodas com a IMU apresentados na Seção 6.1 mostraram uma boa estimação dos deslocamentos lineares e angulares, apesar do acúmulo de erros causados por escorregamento de rodas quando o robô realiza curvas. Contudo, esta fusão pode auxiliar a odometria LiDAR em situações onde esta é prejudicada pelo baixo número de *features* identificadas.

Para a aplicação do filtro EKF proposto, as covariâncias da odometria LiDAR foram configuradas com os ganhos $\mathrm{g}_x = 4,8\text{e-}3$, $\mathrm{g}_y = 2,2\text{e-}3$, $\mathrm{g}_z = 1,6\text{e-}3$, $\mathrm{g}_\phi = 4,4\text{e-}3$, $\mathrm{g}_\theta = 5,2\text{e-}3$ e $\mathrm{g}_\psi = 5,0\text{e-}3$, e o número de máximo de *features* de bordas e planos configurados como $\mathbb{n}_b = 700$ e $\mathbb{n}_p = 5400$ respectivamente. Estes valores foram determinados a partir de experimentos realizados previamente em ambientes virtuais com o Espeleo-LeGO-LOAM, comparando a odometria LiDAR com o *ground truth* do simulador. Para isso foram calculados os erros de cada componente das poses relativas, assim como as suas variâncias. Posteriormente foram realizados ajustes manuais dos valores em simulações com o filtro integrado ao LiDAR SLAM.

Com o objetivo de avaliar o desempenho do filtro EKF foram realizados duas simulações, uma em uma galeria quadrada como mostra a Figura 30, e outra em um duto longo como apresentado na Figura 31. Nestes dois cenários o EspeleoRobô virtual foi operado utilizando um algoritmo de controle de navegação em dutos e um seguidor de parede fazendo com que o robô se desloque sempre no centro do duto ou galeria [Amaral et al., 2020]. No total foram percorridos 209 m dentro do duto e 207 m na galeria, onde

o robô passa por um curva a esquerda, uma descida e uma subida antes de finalizar o percurso. Para a análise foram utilizadas as métricas $\mathcal{M}_1$ medindo o percurso estimado pela técnica LiDAR com e sem o filtro em relação ao *ground truth* do simulador, a métrica $\mathcal{M}_2$ analisando a diferença entre as posições finais do LiDAR SLAM com respeito ao *ground truth*, e o erro médio e desvio padrão para cada componentes da pose estimada. Com a finalidade de validar a aplicação do filtro em um ambiente real com o EspeleoRobô, foi executado um experimento nas proximidades do saguão do auditório da Escola de Engenharia da UFMG.

6.3.1 Simulações em Galeria e Duto

Os resultados da simulação com o robô virtual percorrendo uma galeria quadrada podem ser vistos nas Figuras 49 e 50 que apresentam as posições estimadas pelo LiDAR SLAM com e sem o filtro EKF assim como os mapas gerados, e na Tabela 12 que apresenta os resultados das métricas. Nesse cenário a técnica de LiDAR SLAM identificou em média 25 *features* de bordas e 1414 *features* de planos, onde a sua execução sem o filtro obteve um erro de $51,78\%$ na estimação do percurso realizado pelo robô, com uma diferença de $191,20$ m da posição final comparada com o *ground truth*. Entre as componentes da pose, as que obtiveram maiores erros foram $p_{\mathrm{L},x}$, $p_{\mathrm{L},y}$, com erro médio de $60,29 \pm 33,77$ m e $76,60 \pm 47,81$ m respectivamente, e ψ_{L} com erro médio de $14,48 \pm 0,60°$, devido à identificação de poucas *features* de bordas que são utilizadas para a estimação destas componentes. Os erros de estimação são grades, chegando ao ponto da odometria estimar uma direção contrária de movimento que a executada pelo robô, como pode ser visto na Figura 49 pelo gráfico em vermelho e na Figura 50(a) pela construção do mapa. A aplicação do filtro integrado ao LiDAR SLAM obteve resultados significativamente melhores, com um erro de $0,38\%$ do caminho percorrido e uma diferença de $2,60$ m da posição final.

Tabela 12 – Resultados do experimento Galeria Quadrada com e sem o filtro aplicado ao LiDAR SLAM.

Valores/SLAM	Sem Filtro	Com Filtro
$\mathcal{M}_1$ [%]	$51,78\%$	$0,38\%$
$\mathcal{M}_2$ [m]	$191,196$	$2,603$
$\mu_x \pm \sigma_x$[m]	$60,291 \pm 33,768$	$0,604 \pm 0,274$
$\mu_y \pm \sigma_y$[m]	$76,596 \pm 47,806$	$0,753 \pm 0,347$
$\mu_z \pm \sigma_z$[m]	$1,724 \pm 1,549$	0.804 ± 0.708
$\mu_\phi \pm \sigma_\phi$[°]	$1,555 \pm 0,286$	$0,389 \pm 0,029$
$\mu_\theta \pm \sigma_\theta$[°]	$4,194 \pm 0,450$	$0.655 \pm 0,043$
$\mu_\psi \pm \sigma_\psi$[°]	$14,484 \pm 0,604$	$0,504 \pm 0,047$

Os mapas gerados podem ser vistos na Figura 50 onde se observa a subestimação da odometria, incluindo uma estimação de movimentação no sentido contrário, como se o robô começasse a andar de ré, deformando o mapa e indicando uma bifurcação no lugar da curva. Na Figura 50(b) é possível verificar a correção do mapa pelo filtro implementado.

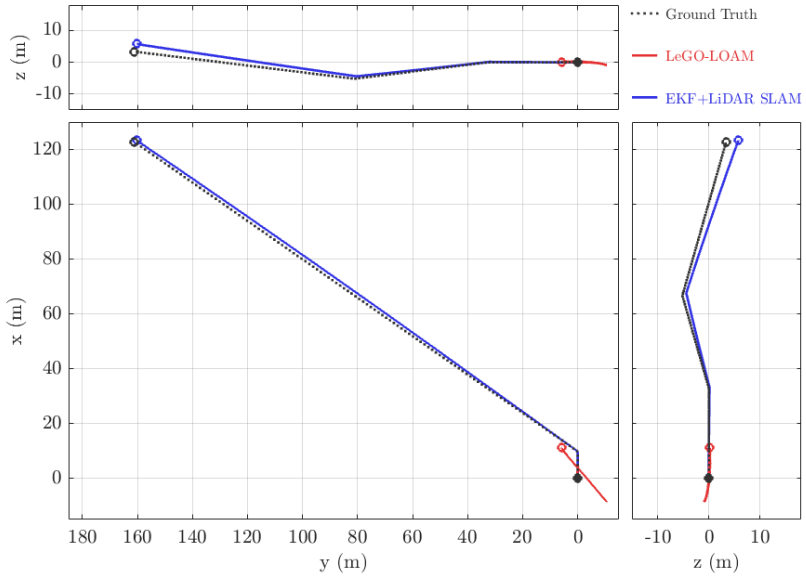
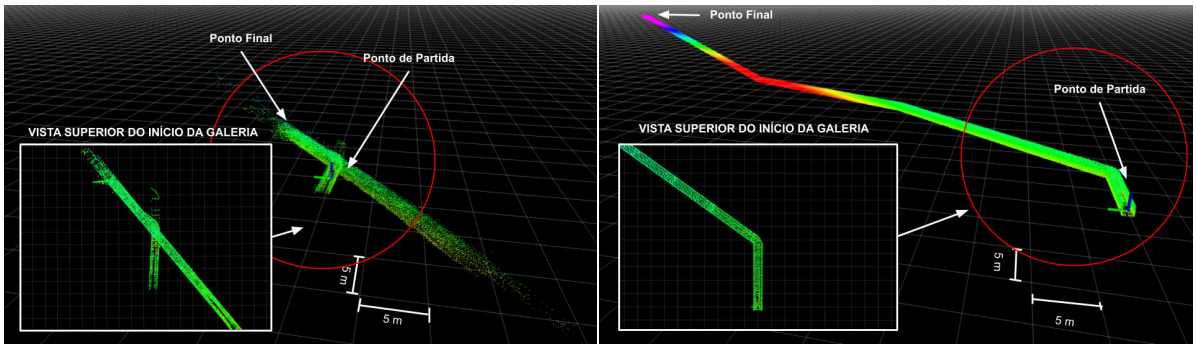


Figura 49 – Resultados das posições estimadas no experimento da Galeria Quadrada.



(a) Sem Filtro (b) Com Filtro

Figura 50 – Mapas gerados na simulação dentro da Galeria Quadrada: (a) mapa resultante sem o filtro, e (b) mapa resultante com o filtro.

Para a simulação do robô se locomovendo em um duto, onde os resultados podem ser observados nas Figuras 51 e 52, a técnica LiDAR SLAM identificou no cenário em média 36 *features* de bordas e 1279 *features* de planos. Com isso a metodologia sem o filtro apresentou um erro na estimação do caminho percorrido de $55,37\%$ ($\mathcal{M}_1$) e uma diferença de $154,35$ m de distância entre as posições finais, dada pela métrica $\mathcal{M}_2$. Novamente houve uma subestimação das componentes $p_{\mathrm{L},x}$, $p_{\mathrm{L},y}$ e ψ_{L}. Em contrapartida, o experimento realizado com o filtro obteve erro de $0,03\%$ na estimação do caminho total percorrido pelo robô e uma diferença de $4,02$ m entre as posições finais.

Nos mapas apresentados na Figura 52 é possível verificar o erro na construção do mapa gerado em consequência do erro da odometria no o experimento sem o filtro. As imagens apresentam o mapa completo e o segmento inicial quando o robô realiza a primeira curva. Na Figura 52(a) é possível observar que houve subestimação da odometria

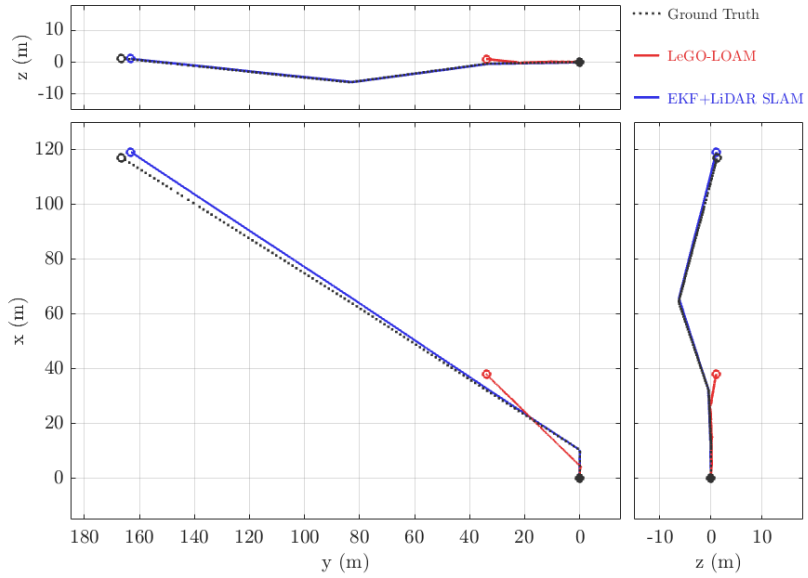


Figura 51 – Resultados das posições estimadas no experimento do Duto Longo.

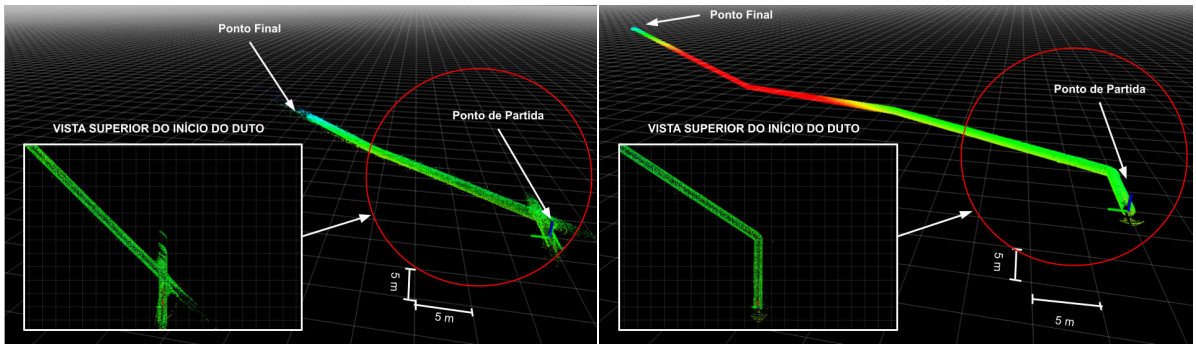
Tabela 13 – Resultados do experimento do Duto Longo com e sem o filtro aplicado ao LiDAR SLAM.

Valores/SLAM	Sem Filtro	Com Filtro
$\mathcal{M}_1$ [%]	$55,37\%$	$0,03\%$
$\mathcal{M}_2$ [m]	$154,353$	$4,024$
$\mu_x \pm \sigma_x$[m]	$39,479 \pm 24,095$	$1,181 \pm 0,775$
$\mu_y \pm \sigma_y$[m]	$62,460 \pm 42,280$	$1,121 \pm 0,923$
$\mu_z \pm \sigma_z$[m]	$2,327 \pm 2,007$	$0,052 \pm 0,040$
$\mu_\phi \pm \sigma_\phi$[°]	$3,171 \pm 0,242$	$0,510 \pm 0,040$
$\mu_\theta \pm \sigma_\theta$[°]	$2,085 \pm 0,257$	$0,296 \pm 0,068$
$\mu_\psi \pm \sigma_\psi$[°]	$11,482 \pm 0,360$	$1,033 \pm 0,067$

e deformação do mapa já no primeiro segmento percorrido pelo robô. A Figura 52(b) ilustra que esse erro foi reparado com a aplicação do filtro, que corrigiu tanto a odometria quando o mapa.

6.3.2 Experimento *Indoor* com o Filtro EKF Integrado ao SLAM

Para uma validação inicial do filtro EKF aplicado ao LiDAR SLAM em um ambiente real, foi realizado um experimento com o EspeleoRobô próximo ao auditório da Escola de Engenharia, diferente do apresentado na Subseção 6.2.1. Saindo do saguão inferior, o robô foi teleoperado subindo pela rampa lateral, passando pelo pátio interno e retornando pela passarela até o saguão superior do auditório. Os resultados das posições estimadas são apresentados na Figura 53, onde é possível observar que a técnica de LiDAR SLAM, com e sem o filtro, estima basicamente o mesmo caminho até a posição $x = 9,71$ m e $y = 25,9$ m, onde inicia a subestimação da odometria LiDAR quando o robô está



(a) Sem Filtro (b) Com Filtro

Figura 52 – Mapas gerados na simulação dentro do Duto Longo: (a) mapa resultante sem o filtro, e (b) mapa resultante com o filtro.

subindo a rampa de acesso ao pátio superior. A partir desse ponto ocorre uma defasagem no caminho estimado do robô, no qual a estimativa da localização sem filtro finaliza o percurso com uma diferença de $8,82$ m da estimativa da técnica LiDAR com filtro.

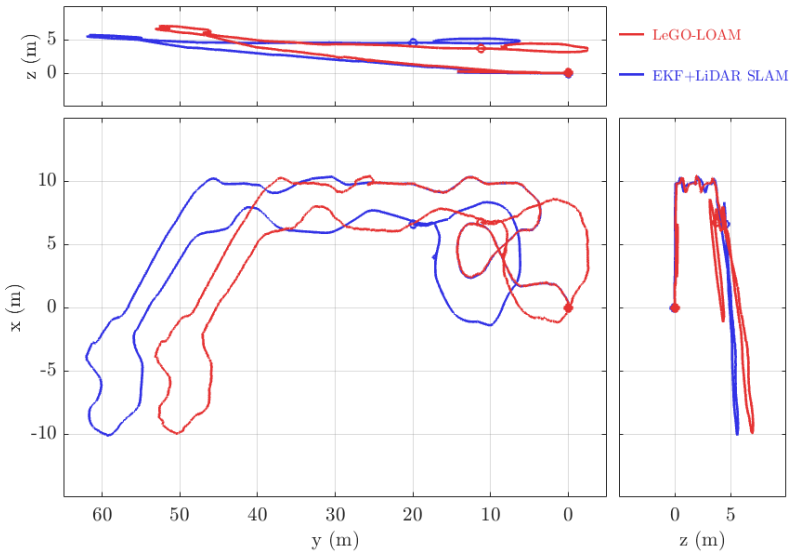
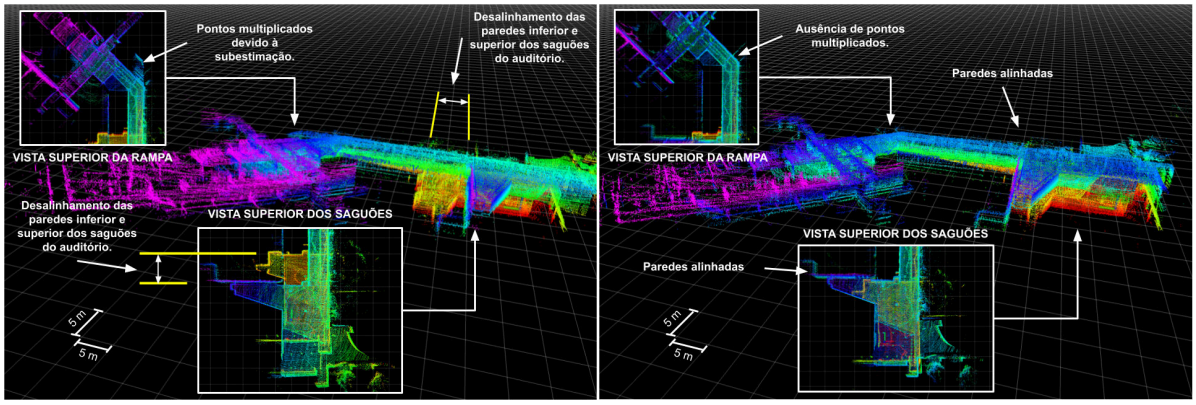


Figura 53 – Resultados das posições estimadas no experimento da Escola de Engenharia pelo LiDAR SLAM com e sem filtro EKF integrado.

A aplicação do filtro permitiu corrigir a odometria e realizar a distorção do mapa, alinhando os saguões inferior e superior do auditório. A correção realizada é vista de um melhor ângulo na Figura 54. Pode ser observado na Figura 54(a) que houve um desalinhamento de aproximadamente 9 m, conforme indicado pelo erro da posição final estimada, entre as parede dos saguões inferior e superior do auditório da Escola de Engenharia. No meio da rampa, indicado pela imagem no canto superior esquerdo da Figura 54(a), onde ocorreu a subestimação do odometria LiDAR, pode ser observado o erro na construção do mapa devido a multiplicidade do pontos pertencentes à parede.

Estes erros foram corrigidos com a aplicação do filtro, como mostra a Figura 54(b) pelo casamento do saguão inferior e superior, e a distorção do mapa do corredor. Os efeitos podem ser comparados com a estimação da odometria em azul no Figura 53 que aparenta ser um caminho mais conciso com o percurso realizado pelo robô, principalmente na estimação do z, que mostra que o robô subiu a rampa e retornou para o saguão superior executando um caminho no mesmo nível, a 5 m de altura da posição inicial.



(a) Sem Filtro (b) Com Filtro

Figura 54 – Mapas gerados no experimento realizado nos saguões do auditório da Escola de Engenharia da UFMG, onde (a) mostra o mapa resultante sem o filtro e (b) mostra o mapa resultante com o filtro.

Capítulo 7

Conclusões e Trabalhos Futuros

Esta dissertação apresentou um estudo e investigação de 3 técnicas LiDAR SLAM consideradas como estado da arte, LOAM-Velodyne, LeGO-LOAM e HDL-Graph-SLAM, para implementação no EspeleoRobô, um dispositivo robótico de inspeção de ambientes confinados. Este robô está sendo desenvolvido pelo Instituto Tecnológico Vale (ITV) juntamente com a equipe de espeleologia da mineradora Vale S.A. e a Universidade Federal de Minas Gerais (UFMG). Como parte de um complexo sistema de navegação, a localização e mapeamento são informações essenciais para a operação autônoma, pois fornecem dados para os algoritmos de controle de navegação e planejamento de caminhos. Porém, espaços confinados oferecem diversos desafios à estimação da posição e orientação do robô, como ausência de sinal de GPS, interferências magnéticas, solos escorregadios e falta de iluminação, o que faz com que a aplicação de técnicas de SLAM baseadas em sensores LiDAR, chamadas de LiDAR SLAM, seja uma solução viável. Contudo, para o funcionamento adequado de um robô autônomo, a sua localização deve ser estimada d eforma *online* e o mapa gerado do ambiente deve ser representativo e leve do ponto de vista computacional. Portanto, esta dissertação de mestrado teve como objetivo inicial a identificação de uma técnica LiDAR SLAM capaz de ser embarcada e executada *online* EspeleoRobô de forma a proporcionar sua operação em modo autônomo.

A análise das técnicas de LiDAR SLAM foi realizada inicialmente por meio de uma simulação em uma das cavernas da DARPA *Subterranean challenge*, possibilitando utilizar informações de *ground truth* fornecidas pelo simulador para avaliar as precisões das estimações das poses e dos mapas gerados. Em seguida, foram realizados experimentos em laboratório nas instalações da Escola de Engenharia, empregando métricas para avaliar a precisão das odometrias e dos mapas com respeito ao ambiente real de operação. Por fim, as técnicas foram verificadas em um ambiente representativo de operação do EspeleoRobô, mapeando uma das galerias da Mina du Veloso na cidade de Ouro Preto.

Apesar de muito utilizada, a técnica LOAM-Velodyne apresentou menor precisão e maior nível de ruído com respeito às demais estratégias, além de ter o desempenho

prejudicado por não aplicar técnicas de identificação de *loop clousure*, acarretando erros estimação da odometria e deformação do mapa. A técnica HDL-Graph-SLAM obteve resultados muito próximos aos do LeGO-LOAM, porém o acúmulo de erros na odometria acarretaram em menor precisão, além da construção *offline* do mapa que pode prejudicar a correção da pose gerada, bem como causar deformações no mapa gerado devido ao elevado consumo computacional exigido. Já o LeGO-LOAM obteve um bom desempenho em todos os experimentos, com boa precisão da estimação da pose do robô e mapas leves e representativos do ambientes. Sendo assim, essa técnica foi adaptada para a implementação no EspeleoRobô, incluindo modificações do código para facilitar a configuração dos parâmetros do SLAM como as variáveis do mapeamento, *loop clousure* e da odometria LiDAR, além de realizar as transformações de coordenadas necessárias e o cálculo da covariância em função da quantidade de *features* identificadas do ambiente.

A técnica LeGO-LOAM, assim como as demais metodologias LiDAR SLAM possuem uma boa precisão na geração de modelos tridimensionais, identificação de obstáculos, além de serem inerentes à iluminação do ambiente e mapear grandes distâncias. Porém, ambientes como dutos e galerias extensas ou mesmo corredores longos e homogêneos apresentam poucas *features* geométricas que podem causar erros de subestimação da pose. Desta forma, foi proposto e implementado um EKF fundindo os dados da odometria LiDAR, usando uma covariância adaptativa, com a odometria das rodas e os dados a IMU, aplicado entre as etapas de *Back-End* e *Front-End* da técnica de LiDAR SLAM de forma corrigir tanto a estimação da odometria quanto o mapa gerado. Durante simulações em um cenário de uma galeria quadrada e um duto longo, a aplicação do filtro reduziu os erros da estimação da odometria em mais de 50% e permitiu um mapeamento mais conciso e representativo do ambiente. Para validar a aplicação do filtro aplicado ao LiDAR SLAM, foi realizado um experimento próximo ao auditório da Escola de Engenharia da UFMG, onde a técnica LiDAR SLAM sem o filtro subestimou o caminho percorrido em aproximadamente 9 metros. A utilização do filtro permitiu corrigir a odometria e realizou a distorção do mapa, alinhando os saguões inferior e superior do auditório.

Trabalhos Futuros

Esta dissertação apresentou resultados satisfatórios em relação à aplicação do filtro de Kalman Estendido ao *pipeline* do LiDAR SLAM. Este tipo de aplicação tem sido desenvolvida e utilizada com o objetivo de melhorar o desempenho das estratégias de SLAM em diferentes ambientes de operação. Desta forma, um dos trabalhos futuros consiste na implementação de uma nova versão do filtro EKF considerando novos modelos de predição e correção adaptados para diferentes cenários.

A fusão da odometria LiDAR, IMU e odometria das rodas, como apresentado nesta

dissertação, é baseada na translação e rotação dos sensores com respeito ao sistema de coordenadas do robô. Portanto, outro trabalho futuro consiste em implementar algoritmos para calibração de forma a melhorar a precisão na estimação da pose calculada pelo filtro.

Com o objetivo de aprimorar o desempenho da técnica de LiDAR SLAM, outro trabalho futuro se concentra no levantamento preciso das covariâncias associadas à odometria LiDAR em função do número de *features* geométricas identificadas no ambiente, gerando funções adaptativas mais fiéis ao comportamento do sistema, além da integração da odometria visual ao sistema de localização e mapeamento do EspeloRobô. Da mesma forma que na odometria LiDAR, seria necessária a determinação de um modelo para definir as covariâncias em função das *features* visuais identificadas.

Buscando melhorar o desempenho do sistema conforme as condições de operação, é proposto também como um trabalho futuro a implementação de uma máquina de estados para a identificação de características do ambiente e ajuste dos parâmetros da estratégia de SLAM, buscando uma estimação de poses mais precisa associada a mapas com baixo erro computados num curto intervalo de tempo.

Referências

I. Amaral, C. Fany, D. Simon, L. Matos, Á. Araújo, L. Leão, A. Rezende, H. Azpúrua, G. Pessin, and G. Freitas. Sistema de alertas e operação assistida de um robô para a inspeçãao de ambientes confinados-espeleorobô. *Anais da Sociedade Brasileira de Automática*, 2(1), 2020.

H. Azpurua, F. Rocha, G. Garcia, A. S. Santos, E. Cota, L. G. Barros, A. S. Thiago, G. Pessin, and G. M. Freitas. EspeleoRobô - a robotic device to inspect confined environments. In *2019 19th Int. Conf. on Advanced Robotics (ICAR)*. IEEE, Dec. 2019.

H. Azpúrua, A. Rezende, G. Potje, G. P. da Cruz Júnior, R. Fernandes, V. Miranda, L. W. de Resende Filho, J. Domingues, F. Rocha, F. L. M. de Sousa, et al. Towards semi-autonomous robotic inspection and mapping in confined spaces with the espeleorobô. *Journal of Intelligent & Robotic Systems*, 101(4):1–27, 2021.

T. Bailey and H. Durrant-Whyte. Simultaneous localization and mapping (slam): Part ii. *IEEE robotics & automation magazine*, 13(3):108–117, 2006.

M. Ben-Ari and F. Mondada. *Elements of robotics*. Springer Nature, 2017.

P. J. Besl and N. D. McKay. Method for registration of 3-d shapes. In *Sensor fusion IV: control paradigms and data structures*, volume 1611, pages 586–606. International Society for Optics and Photonics, 1992.

P. Biber and W. Straßer. The normal distributions transform: A new approach to laser scan matching. In *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)(Cat. No. 03CH37453)*, volume 3, pages 2743–2748. IEEE, 2003.

M. Bloesch, S. Omari, M. Hutter, and R. Siegwart. Robust visual inertial odometry using a direct ekf-based approach. In *2015 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 298–304. IEEE, 2015.

A. Bögli. *Karst hydrology and physical speleology*. Springer Science & Business Media, 2012.

D. Borrmann, J. Elseberg, K. Lingemann, and A. Nüchter. The 3d hough transform for plane detection in point clouds: A review and a new accumulator design. *3D Research*, 2(2):3, 2011.

BRASIL. Nr 33 - segurança e saúde nos trabalhos em espaços confinados, 2006.

C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Transactions on robotics*, 32(6):1309–1332, 2016.

L. Chang, X. Niu, and T. Liu. Gnss/imu/odo/lidar-slam integrated navigation system using imu/odo pre-integration. *Sensors*, 20(17):4702, 2020.

H. M. Choset, S. Hutchinson, K. M. Lynch, G. Kantor, W. Burgard, L. E. Kavraki, and S. Thrun. *Principles of robot motion: theory, algorithms, and implementation*. MIT press, 2005.

A. Cid, M. Nazário, M. Sathler, F. Martins, J. Domingues, M. Delunardo, P. Alves, R. Teotônio, L. G. Barros, A. Rezende, et al. A simulated environment for the development and validation of an inspection robot for confined spaces. In *2020 Latin American Robotics Symposium (LARS), 2020 Brazilian Symposium on Robotics (SBR) and 2020 Workshop on Robotics in Education (WRE)*, pages 1–6. IEEE, 2020.

CloudCompare. Cloudcompare (version 2.9. 1) gpl software. 2018.

W. F. Costa, J. P. Matsuura, F. S. Santana, and A. M. Saraiva. Evaluation of an icp based algorithm for simultaneous localization and mapping using a 3d simulated p3dx robot. In *2010 Latin American Robotics Symposium and Intelligent Robotics Meeting*, pages 103–108. IEEE, 2010.

J. L. Crowley. World modeling and position estimation for a mobile robot using ultrasonic ranging. In *Proceedings, 1989 International Conference on Robotics and Automation*, pages 674–680. IEEE, 1989.

G. P. d. Cruz Júnior, L. V. d. C. Matos, H. Azpúrua, G. Pessin, and G. M. Freitas. Investigação de Técnicas LiDAR SLAM para um Dispositivo Robótico de Inspeção de Ambientes Confinados. In *Anais do Congresso Brasileiro de Automática 2020*, dec 2020.

T. Dang, F. Mascarich, S. Khattak, H. Nguyen, N. Khedekar, C. Papachristos, and K. Alexis. Field-hardened robotic autonomy for subterranean exploration. *Field and Service Robotics (FSR)*, 2019.

X. Dang, Z. Rong, and X. Liang. Sensor fusion-based approach to eliminating moving objects for slam in dynamic environments. *Sensors*, 21(1):230, 2021.

C. Debeunne and D. Vivet. A review of visual-lidar fusion based simultaneous localization and mapping. *Sensors*, 20(7):2068, 2020.

J.-E. Deschaud. Imls-slam: scan-to-model matching based on 3d data. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2480–2485. IEEE, 2018.

G. Dissanayake, S. Huang, Z. Wang, and R. Ranasinghe. A review of recent developments in simultaneous localization and mapping. In *2011 6th International Conference on Industrial and Information Systems*, pages 477–482. IEEE, 2011.

C. F. A. Duarte, I. F. S. Amaral, L. P. V. Dutra, G. P. d. Cruz Júnior, H. Azpúrua, and G. M. Freitas. Utilizaçao do robot operating system (ros) em conjunto com o kit didático lego mindstorms no ensino de robótica móvel. In *Anais da Sociedade Brasileira de Automática*, volume 1. Simpósio Brasileiro de Automaçao Inteligente, 2019.

G. Dudek and M. Jenkin. *Computational principles of mobile robotics*. Cambridge university press, 2010.

H. F. Durrant-Whyte. Uncertain geometry in robotics. *IEEE Journal on Robotics and Automation*, 4(1):23–31, 1988.

K. Ebadi, Y. Chang, M. Palieri, A. Stephens, A. Hatteland, E. Heiden, A. Thakur, N. Funabiki, B. Morrell, S. Wood, et al. Lamp: Large-scale autonomous mapping and positioning for exploration of perceptually-degraded subterranean environments. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 80–86. IEEE, 2020.

R. Fernandes, T. L. Rocha, H. Azpúrua, G. Pessin, A. A. Neto, and G. Freitas. Investigation of visual reconstruction techniques using mobile robots in confined environments. In *2020 Latin American Robotics Symposium (LARS), 2020 Brazilian Symposium on Robotics (SBR) and 2020 Workshop on Robotics in Education (WRE)*, pages 1–6. IEEE, 2020.

M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

G. M. Freitas, A. M. d. C. Rezende, V. R. F. Miranda, G. P. d. C. Júnior, R. F. G. da Silva, C. F. A. Duarte, I. F. S. Amaral, D. S. Marques, L. V. d. C. Matos, Á. R. Araújo, H. I. A. Perez-Imaz, F. A. S. Rocha, G. C. Garcia, A. S. Santos, L. W. d. R. Filho, J. D. Domingues, A. M. d. S. T. Filho, F. L. M. de Souza, and G. Pessin. Terceiro Relatório Técnico de Acompanhamento do Projeto "Dispositivo Robótico para Inspeção de Ambientes Restritos e Confinados". Technical report, Instituto Tecnológico Vale, Belo Horizonte, jul 2020.

A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3354–3361. IEEE, 2012.

S. B. Gokturk, H. Yalcin, and C. Bamji. A time-of-flight depth sensor-system description, issues and solutions. In *2004 conference on computer vision and pattern recognition workshop*, pages 35–35. IEEE, 2004.

J. Graeter, A. Wilczynski, and M. Lauer. Limo: Lidar-monocular visual odometry. In *2018 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 7872–7879. IEEE, 2018.

G. Grisetti, C. Stachniss, and W. Burgard. Improved techniques for grid mapping with rao-blackwellized particle filters. *IEEE transactions on Robotics*, 23(1):34–46, 2007.

G. Grisetti, R. Kümmerle, C. Stachniss, and W. Burgard. A tutorial on graph-based slam. *IEEE Intelligent Transportation Systems Magazine*, 2(4):31–43, 2010.

S. Hening, C. A. Ippolito, K. S. Krishnakumar, V. Stepanyan, and M. Teodorescu. 3d lidar slam integration with gps/ins for uavs in urban gps-degraded environments. In *AIAA Information Systems-AIAA Infotech@ Aerospace*, page 0448. 2017.

A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard. Octomap: An efficient probabilistic 3d mapping framework based on octrees. *Autonomous robots*, 34 (3):189–206, 2013.

S. J. Julier and J. K. Uhlmann. New extension of the kalman filter to nonlinear systems. In *Signal processing, sensor fusion, and target recognition VI*, volume 3068, pages 182–193. International Society for Optics and Photonics, 1997.

M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert. isam2: Incremental smoothing and mapping using the bayes tree. *The Int. Journal of Robotics Research*, 31(2):216–235, 2012.

S. Khattak, C. Papachristos, and K. Alexis. Keyframe-based direct thermal–inertial odometry. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 3563–3569. IEEE, 2019.

S. Khattak, H. Nguyen, F. Mascarich, T. Dang, and K. Alexis. Complementary multi–modal sensor fusion for resilient robot pose estimation in subterranean environments. In *2020 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 1024–1029. IEEE, 2020.

K. Koide, J. Miura, and E. Menegatti. A portable three-dimensional lidar-based system for long-term and wide-area people behavior measurement. *Int. Journal of Advanced Robotic Systems*, 16(2):1729881419841532, 2019.

H. Kolvenbach, D. Wisth, R. Buchanan, G. Valsecchi, R. Grandia, M. Fallon, and M. Hutter. Towards autonomous inspection of concrete deterioration in sewers with legged robots. *Journal of Field Robotics*, 37(8):1314–1327, 2020.

L. Kunze, T. Roehm, and M. Beetz. Towards semantic robot description languages. In *2011 IEEE International Conference on Robotics and Automation*, pages 5589–5595. IEEE, 2011.

Z. Kurt-Yavuz and S. Yavuz. A comparison of ekf, ukf, fastslam2. 0, and ukf-based fastslam algorithms. In *2012 IEEE 16th International Conference on Intelligent Engineering Systems (INES)*, pages 37–43. IEEE, 2012.

J. Larson, B. Okorn, T. Pastore, D. Hooper, and J. Edwards. Counter tunnel exploration, mapping, and localization with an unmanned ground vehicle. In *Unmanned Systems Technology XVI*, volume 9084, page 90840Q. International Society for Optics and Photonics, 2014.

M. Magnusson, A. Nuchter, C. Lorken, A. J. Lilienthal, and J. Hertzberg. Evaluation of 3d registration reliability and speed-a comparison of icp and ndt. In *2009 IEEE International Conference on Robotics and Automation*, pages 3907–3912. IEEE, 2009.

A. Mandow, J. L. Martinez, J. Morales, J. L. Blanco, A. Garcia-Cerezo, and J. Gonzalez. Experimental kinematics for wheeled skid-steer mobile robots. In *2007 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 1222–1227. IEEE, 2007.

D. Meagher. Geometric modeling using octree encoding. *Computer graphics and image processing*, 19(2):129–147, 1982.

F. Mokhtarian. Silhouette-based isolated object recognition through curvature scale space. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(5):539–544, 1995.

F. Neuhaus, T. Koß, R. Kohnen, and D. Paulus. Mc2slam: Real-time inertial lidar odometry using two-scan motion compensation. In *German Conference on Pattern Recognition*, pages 60–72. Springer, 2018.

B. C. Ooi. Spatial kd-tree: A data structure for geographic database. In *Datenbanksysteme in Büro, Technik und Wissenschaft*, pages 247–258. Springer, 1987.

M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng. Ros: an open-source robot operating system. In *ICRA workshop*, volume 3, page 5. Kobe, 2009.

A. M. C. Rezende, G. P. C. Junior, R. Fernandes, V. R. F. Miranda, H. Azpurua, G. Pessin, and G. M. Freitas. Indoor Localization and Navigation Control Strategies for a Mobile Robot Designed to Inspect Confined Environments. In *2020 IEEE 16th International Conference on Automation Science and Engineering (CASE)*, pages 1427–1433. IEEE, aug 2020.

T. Rouček, M. Pecka, P. Čížek, T. Petříček, J. Bayer, V. Šalanskỳ, D. Heřt, M. Petrlík, T. Báča, V. Spurnỳ, et al. Darpa subterranean challenge: Multi-robotic exploration of underground environments. In *International Conference on Modelling and Simulation for Autonomous Systesm*, pages 274–290. Springer, 2019.

R. B. Rusu, Z. C. Marton, N. Blodow, and M. Beetz. Learning informative point classes for the acquisition of object model maps. In *2008 10th International Conference on Control, Automation, Robotics and Vision*, pages 643–650. IEEE, 2008.

J. Sankaranarayanan, H. Samet, and A. Varshney. A fast all nearest neighbor algorithm for applications involving large point-clouds. *Computers & Graphics*, 31(2):157–174, 2007.

T. Shan and B. Englot. Lego-loam: Lightweight and ground-optimized lidar odometry and mapping on variable terrain. In *2018 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pages 4758–4765. IEEE, 2018.

J. R. Shewchuk. Delaunay refinement algorithms for triangular mesh generation. *Computational geometry*, 22(1-3):21–74, 2002.

M. G. Shirangi and A. A. Emerick. An improved tsvd-based levenberg–marquardt algorithm for history matching and comparison with gauss–newton. *Journal of Petroleum Science and Engineering*, 143:258–271, 2016.

B. Siciliano and O. Khatib. *Springer Handbook of Robotics*. 2016.

B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo. *Robotics: modelling, planning and control*. Springer Science & Business Media, 2010.

R. C. Smith and P. Cheeseman. On the representation and estimation of spatial uncertainty. *The international journal of Robotics Research*, 5(4):56–68, 1986.

S. Stiene, K. Lingemann, A. Nuchter, and J. Hertzberg. Contour-based object detection in range images. In *Third International Symposium on 3D Data Processing, Visualization, and Transmission (3DPVT'06)*, pages 168–175. IEEE, 2006.

Y. Su, T. Wang, S. Shao, C. Yao, and Z. Wang. Gr-loam: Lidar-based sensor fusion slam for ground robots on complex terrain. *Robotics and Autonomous Systems*, page 103759, 2021.

H. Sun, D. Y. Kim, J.-H. Hwang, and C.-W. Park. Mobile robot navigation under duct environment. In *2011 IEEE Applied Imagery Pattern Recognition Workshop (AIPR)*, pages 1–3. IEEE, 2011.

L. Sun, J. Zhao, X. He, and C. Ye. Dlo: Direct lidar odometry for 2.5 d outdoor environment. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 1–5. IEEE, 2018.

F. Tâche, F. Pomerleau, G. Caprari, R. Siegwart, M. Bosse, and R. Moser. Three-dimensional localization for the magnebike inspection robot. *Journal of Field Robotics*, 28(2):180–203, 2011.

D. Tardioli, D. Sicignano, L. Riazuelo, A. Romeo, J. L. Villarroel, and L. Montano. Robot teams for intervention in confined and structured environments. *Journal of Field Robotics*, 33(6):765–801, 2016.

S. Thrun, W. Burgard, and D. Fox. A real-time algorithm for mobile robot mapping with applications to multi-robot and 3d mapping. In *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, volume 1, pages 321–328. IEEE, 2000.

S. Thrun, W. Burgard, and D. Fox. Probabilistic robotics. 2005.

J. Underwood, A. Hill, and S. Scheding. Calibration of range sensor pose on mobile platforms. In *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3866–3871. IEEE, 2007.

R. Vincent, B. Limketkai, and M. Eriksen. Comparison of indoor robot localization techniques in the absence of gps. In *Detection and Sensing of Mines, Explosive Objects, and Obscured Targets XV*, volume 7664, page 76641Z. International Society for Optics and Photonics, 2010.

S.-Y. Wan and W. E. Higgins. Symmetric region growing. *IEEE Transactions on Image processing*, 12(9):1007–1015, 2003.

C. Wang, W. Sun, D. X. Bu, and Z. W. Zhou. Ekf-based 6d slam for air-duct cleaning robot using inertial sensor and stereo vision. In *Applied Mechanics and Materials*, volume 313, pages 941–945. Trans Tech Publ, 2013.

J. Wirtz, P. G. Patterson, W. H. Kunz, T. Gruber, V. N. Lu, S. Paluch, and A. Martins. Brave new world: service robots in the frontline. *Journal of Service Management*, 2018.

D. Wisth, M. Camurri, S. Das, and M. Fallon. Unified multi-modal landmark tracking for tightly coupled lidar-visual-inertial odometry. *IEEE Robotics and Automation Letters*, 6 (2):1004–1011, 2021.

U. Wong, A. Morris, C. Lea, J. Lee, C. Whittaker, B. Garney, and R. Whittaker. Comparative evaluation of range sensing technologies for underground void modeling. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3816–3823. IEEE, 2011.

Y. Xu, Y. Ou, and T. Xu. Slam of robot based on the fusion of vision and lidar. In *2018 IEEE International Conference on Cyborg and Bionic Systems (CBS)*, pages 121–126. IEEE, 2018.

J. Zhang and S. Singh. Loam: Lidar odometry and mapping in real-time. In *Robotics: Science and Systems*, volume 2, 2014.

J. Zhang and S. Singh. Low-drift and real-time lidar odometry and mapping. *Autonomous Robots*, 41(2):401–416, 2017.

J. Zhang and S. Singh. Laser–visual–inertial odometry and mapping with high robustness and low drift. *Journal of Field Robotics*, 35(8):1242–1264, 2018.

R. H. A. Zisserman. Multiple view geometry in computer vision. 2004.