

UNIVERSIDADE FEDERAL DE MINAS GERAIS
Escola de Engenharia
Programa de Pós-graduação de Engenharia Elétrica

Rafael Fernandes Gonçalves da Silva

**Estudo de Métodos de Mapeamento Baseados em Informação
Visual Utilizando Robôs Móveis em Ambientes Confinados**

Belo Horizonte
2021

Rafael Fernandes Gonçalves da Silva

**Estudo de Métodos de Mapeamento Baseados em Informação
Visual Utilizando Robôs Móveis em Ambientes Confinados**

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 Título de Mestrado em Engenharia Elétrica.

Orientador: Gustavo Medeiros Freitas

Coorientador: Armando Alves Neto

Belo Horizonte
2021

R343p

Silva, Rafael Fernandes Gonçalves da.
Estudo de métodos de mapeamento baseados em informação visual utilizando robôs móveis em ambientes confinados [recurso eletrônico] / Rafael Fernandes Gonçalves da Silva. - 2021.
1 recurso online (92 f. : il., color.) : pdf.

Orientador: Gustavo Medeiros Freitas.
Coorientador: Armando Alves Neto.

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

Bibliografia: f. 83-92.

Exigências do sistema: Adobe Acrobat Reader.

1. Engenharia Elétrica - Teses. 2. Ambientes extremos – Teses. 3. Robôs móveis – Teses. 4. Visão de robô – Teses. I. Freitas, Gustavo Medeiros. II. Alves Neto, Armando. III. Universidade Federal de Minas Gerais. Escola de Engenharia. IV. Título.

CDU: 621.3(043)

"Estudo de Métodos de Mapeamento Baseados Em Informação Visual Utilizando Robôs Móveis Em Ambientes Confinados"

Rafael Fernandes Gonçalves da Silva

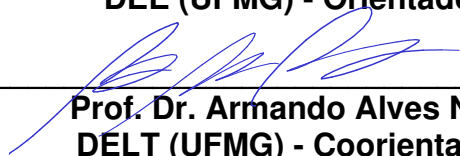
Dissertação de Mestrado submetida à Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação em Engenharia Elétrica da Escola de Engenharia da Universidade Federal de Minas Gerais, como requisito para obtenção do grau de Mestre em Engenharia Elétrica.

Aprovada em 24 de setembro de 2021.

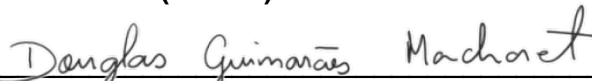
Por:



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



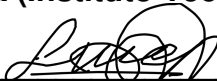
Prof. Dr. Armando Alves Neto
DELT (UFMG) - Coorientador



Prof. Dr. Douglas Guimarães Macharet
DCC (UFMG)



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



Prof. Dr. Luciano Cunha de Araújo Pimenta
DELT (UFMG)

Ao meu pai, à minha mãe e
à minha irmã por todo o apoio
e aos amigos e colegas pela
companhia nesses últimos anos.

Agradecimentos

O autor gostaria de agradecer pelo apoio financeiro e material à Vale S.A., ao Instituto Tecnológico Vale (ITV) e à Universidade Federal de Minas Gerais (UFMG), além dos principais órgãos de fomento, Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), Código de Financiamento 001, Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), e Fundação de Amparo à Pesquisa do Estado de Minas Gerais (FAPEMIG).

*“O verdadeiro valor de um homem não pode ser encontrado nele mesmo,
mas nas cores e texturas que faz surgir nos outros.”*
(Albert Schweitzer)

Resumo

A robótica é um dos campos mais importantes da tecnologia moderna e está cada vez mais presente no dia a dia das pessoas. A aplicação de robôs ocorre nas mais diversas áreas, sendo muitas vezes relacionada à preservação da saúde humana, em tarefas como inspeção, extração, manutenção, exploração e mapeamento. Essas atividades podem ser exaustivas e perigosas para seres humanos, principalmente em espaços que apresentam riscos, como é o caso dos ambientes confinados. Tais locais possuem entrada e saída limitadas e ventilação insuficiente para remoção de partículas, além de não serem projetados para ocupação humana contínua. Em áreas industriais e urbanas, há diversos ambientes contidos nessa categoria, como barragens, embarcações, túneis, cavernas, tubulações e galerias subterrâneas, os quais necessitam de constante inspeção e manutenção de forma a garantir o funcionamento correto. O Instituto Tecnológico Vale (ITV), em parceria com a Universidade Federal de Minas Gerais (UFMG), vem desenvolvendo o EspeleoRobô, cujo principal propósito consiste em reduzir eventuais riscos relacionados à presença humana em locais de difícil acesso. Uma das tarefas mais importantes do EspeleoRobô é a geração de mapas dos ambientes, essencial para o desenvolvimento de diversas aplicações na robótica móvel, como localização, navegação e planejamento de caminho, além de possíveis procedimentos de inspeções e manutenção. Dentre os sensores utilizados na reconstrução de ambientes, destacam-se as câmeras, capazes de adquirir grande quantidade de informação visual do seu entorno. Nesse contexto, o objetivo desta dissertação consiste na investigação métodos de reconstrução visual capazes de gerar mapas densos, coloridos e precisos em ambientes confinados, para fins posteriores de inspeção e análise estrutural por especialistas em espeleologia, por meio de modelos digitais e de realidade virtual. Para tal, são apresentados dois métodos: o Registro da Nuvem de Pontos, combinado com localização externa de um Filtro de Kalman Estendido (EKF), e o processo de Localização e Mapeamento Simultâneos (SLAM) com o *Real-Time Appearance-Based Mapping* (RTAB-Map), ambos utilizando câmeras RGB-D. Os resultados obtidos com os métodos em ambientes simulados e reais, tanto de localização quanto de mapeamento, são analisados e comparados, além dos seus respectivos tempos de processamento. Ao final, os resultados em simulação de ambos os métodos apresentaram valores muito próximos à referência utilizada e, para os ambientes reais, o RTAB-Map mostrou-se mais indicado, gerando reconstruções com menos distorções e mais próximas à referência.

Palavras-chave: Robôs Móveis, Ambientes Confinados, Reconstrução Visual, Registro da Nuvem de Pontos, RTAB-Map.

Abstract

Robotics is one of the most important fields of modern technology and is increasingly present in people's daily lives. The application of robots occurs in the most diverse areas, often related to the preservation of human health, in tasks such as inspection, extraction, maintenance, exploration, and mapping. These activities can be exhausting and dangerous for human beings, especially when the place presents risks, as is the case of confined environments. Such locations have limited entry and exit, insufficient ventilation to remove particles, and they were not even designed for continuous human occupation. In industrial and urban areas, there are several environments included in this category, such as dams, vessels, tunnels, caves, pipes, and underground galleries, which require constant inspection and maintenance to ensure correct operation. The Instituto Tecnológico Vale (ITV), in partnership with the Universidade Federal de Minas Gerais (UFMG), has been developing the EspeleoRobô, whose main purpose is to reduce risks related to human presence in hazardous areas. One of the most important tasks of EspeleoRobô is the map generation of the environments, which is essential for the development of several applications in mobile robotics, such as localization, navigation, path planning, and possible inspection and maintenance procedures. Cameras stand out among the sensors used in the reconstruction of environments, being able to acquire much visual information from the surroundings. In this context, the objective of this thesis is to investigate methods of visual reconstruction able to generate dense, colored, and accurate maps in confined environments, for further purposes of inspection and structural analysis by specialists in speleology, using digital and virtual reality models. For this aim, two methods are presented: the Point Cloud Registration, combined with an external localization with an Extended Kalman Filter (EKF), and the Simultaneous Localization and Mapping (SLAM) process with *Real-Time Appearance-Based Mapping* (RTAB-Map), both using RGB-D cameras. The localization and mapping results obtained with the methods are analyzed and compared in simulated and real environments, in addition to their respective processing times. In the end, the results in simulation of both methods present values very close to the reference, and, for real environments, the RTAB-Map shows to be more suitable, generating more adequate reconstructions and closer to the reference.

Keywords: Mobile Robots, Confined Environments, Visual Reconstruction, Point Cloud Registration, RTAB-Map.

Lista de Ilustrações

Figura 1 – Comparação da presença de humanos e robôs em ambientes confinados.	28
Figura 2 – EspeleoRobô em tarefas de inspeção e monitoramento em diferentes locais.	29
Figura 3 – Diagrama do EspeleoRobô e seus sensores.	30
Figura 4 – Exemplos de robôs aquáticos, aéreos e híbridos desenvolvidos para exploração de ambientes.	35
Figura 5 – Exemplos de robôs terrestres que utilizam esteiras, rodas articuladas e pernas, desenvolvidos para inspeção e exploração de ambientes.	36
Figura 6 – Exemplos de robôs com rodas utilizados em tarefas de exploração e mapeamento de espaços subterrâneos.	37
Figura 7 – Representação do modelo diferencial com indicação das variáveis e do sistemas de coordenadas.	46
Figura 8 – Representação do processo de captura de imagens por câmeras utilizando projeção em perspectiva.	51
Figura 9 – Representação do processo de triangulação utilizado para a obtenção da profundidade para câmeras estéreo.	53
Figura 10 – Representação do processo de projeção de luz estruturada utilizando listras verticais com diferentes espectros (Bhowmik, 2017).	53
Figura 11 – Diagrama representativo das variáveis envolvidas no processo de SLAM, sendo \mathbf{x}_s o estado do robô modificado pelas entradas relativas ao movimento \mathbf{u}_s e as medições \mathbf{z}_s observadas no ambiente \mathcal{M}_s (Stachniss et al., 2016).	55
Figura 12 – Ilustração representativa das incertezas envolvidas no EKF aplicado no processo de SLAM (Stachniss et al., 2016).	56
Figura 13 – Ilustração representativa funcionamento de um método de SLAM baseado em grafos (Stachniss et al., 2016).	57
Figura 14 – Diagrama de blocos do processo de localização com o EKF.	60
Figura 15 – Representação gráfica do ganho de escorregamento e atrito s_f em função do ângulo de arfagem, para b_f igual a -0,12 e c_f igual a 0,6, 0,8 e 1,0.	61
Figura 16 – Diagrama de blocos do processo de mapeamento do Registro de Pontos utilizando imagens e odometria externa.	62

Figura 17 – Representação dos principais sistemas de coordenadas utilizados: inercial (\mathcal{F}_I), do robô (\mathcal{F}_R) e da câmera (\mathcal{F}_C).	64
Figura 18 – Diagrama de blocos do processo de localização visual com o RTAB-Map (Labbé and Michaud, 2019).	65
Figura 19 – Diagrama de blocos do processo de mapeamento com o RTAB-Map (Labbé and Michaud, 2019).	67
Figura 20 – Digrama representativo dos principais tópicos e nós envolvidos no processo do <code>espeleo_pose_ekf</code>	69
Figura 21 – Digrama representativo dos principais tópicos e nós envolvidos no processo do <code>pcl_regristration</code>	70
Figura 22 – Digrama representativo dos principais tópicos e nós envolvidos no processo do <code>rtabmap_ros</code>	72
Figura 23 – Modelo do EspeleoRobô no simulador.	73
Figura 24 – Vistas externa e interna da Caverna do DARPA obtidas no simulador.	74
Figura 25 – Vistas externa e interna da Mina do DARPA obtidas no simulador.	75
Figura 26 – Imagem real do EspeleoRobô.	76
Figura 27 – Mapa gerado com o LeGO-LOAM e foto real da Mina do Veloso, na parte correspondente ao experimento teleoperado.	78
Figura 28 – Mapa gerado com o LeGO-LOAM e foto real da Mina do Veloso, na parte correspondente ao experimento semiautônomo.	79
Figura 29 – Representação gráfica do cálculo da distância entre um ponto da reconstrução de algum dos métodos para o <i>mesh</i> de referência.	83
Figura 30 – Representação dos intervalos de confiança para um desvio-padrão (1σ ou 68% dos pontos), dois desvios-padrão (2σ ou 95%) e três (3σ ou 99,7%).	84
Figura 31 – Resultados da posição dos métodos e da referência nos planos xy , xz e yz na Caverna do DARPA.	85
Figura 32 – Erros de posição e orientação dos métodos ao longo do tempo na Caverna do DARPA.	86
Figura 33 – Mapa obtido pelo <i>Registro-EKF</i> na Caverna do DARPA (vista em perspectiva acima e três vistas internas abaixo).	87
Figura 34 – Mapa obtido pelo <i>RTAB-Map</i> na Caverna do DARPA (vista em perspectiva acima e três vistas internas abaixo).	87
Figura 35 – Comparação dos mapas obtidos pelo <i>Registro-EKF</i> (à esquerda) e pelo <i>RTAB-Map</i> (à direita) com o <i>Ground-Truth</i> na Caverna do DARPA. Acima está representado o erro de cada ponto do mapa de forma colorida e, abaixo, o histograma do erro com a mesma referência de cores.	88
Figura 36 – Resultados da posição dos métodos e da referência nos planos xy , xz e yz na Mina do DARPA.	90

Figura 37 – Erros de posição e orientação dos métodos ao longo do tempo na Mina do DARPA.	91
Figura 38 – Mapa obtido pelo <i>Registro-EKF</i> na Mina do DARPA (vista em perspectiva acima e três vistas internas abaixo).	92
Figura 39 – Mapa obtido pelo <i>RTAB-Map</i> na Mina do DARPA (vista em perspectiva acima e três vistas internas abaixo).	92
Figura 40 – Comparação dos mapas obtido pelo <i>Registro-EKF</i> (à esquerda) e pelo <i>RTAB-Map</i> (à direita) com o <i>Ground-Truth</i> na Mina do DARPA. Acima é representado o erro de cada ponto do mapa de forma colorida e, abaixo, o histograma do erro com a mesma referência de cores.	93
Figura 41 – Resultados da posição dos métodos e da referência nos planos xy , xz e yz no experimento teleoperado na Mina do Veloso.	95
Figura 42 – Diferença de posição e orientação dos métodos em relação à referência ao longo do tempo no experimento teleoperado na Mina do Veloso.	96
Figura 43 – Mapa obtido pelo <i>Registro-EKF</i> no experimento teleoperado na Mina do Veloso (vista em perspectiva acima e três vistas internas abaixo).	97
Figura 44 – Mapa obtido pelo <i>RTAB-Map</i> no experimento teleoperado na Mina do Veloso (vista em perspectiva acima e três vistas internas abaixo).	97
Figura 45 – Comparação dos mapas obtido pelo <i>Registro-EKF</i> (à esquerda) e pelo <i>RTAB-Map</i> (à direita) com o <i>LiDAR-SLAM</i> no experimento teleoperado na Mina do Veloso. Acima é representada diferença entre os pontos do mapa de forma colorida e, abaixo, o histograma dessa diferença na mesma referência de cores.	98
Figura 46 – Resultados de localização nos planos xy , xz e yz no experimento semiautônomo na Mina do Veloso.	100
Figura 47 – Diferença de posição e orientação dos métodos em comparação à referência no experimento semiautônomo na Mina do Veloso.	101
Figura 48 – Mapa obtido pelo <i>Registro-EKF</i> no experimento semiautônomo na Mina do Veloso (vista em perspectiva acima e três vistas internas abaixo).	102
Figura 49 – Mapa obtido pelo <i>RTAB-Map</i> no experimento semiautônomo na Mina do Veloso (vista em perspectiva acima e três vistas internas abaixo).	102
Figura 50 – Comparação dos mapas obtido pelo <i>Registro-EKF</i> (à esquerda) e pelo <i>RTAB-Map</i> (à direita) com o <i>LiDAR-SLAM</i> no experimento semiautônomo na Mina do Veloso. Acima é representada diferença entre os pontos do mapa de forma colorida e, abaixo, o histograma dessa diferença na mesma referência de cores.	103

Lista de Tabelas

Tabela 1 – Tabela comparativa de métodos de localização e mapeamento que utilizam câmeras como sensores principais.	44
Tabela 2 – Erro médio de localização dos métodos na Caverna do DARPA.	84
Tabela 3 – Erro da posição final dos métodos na Caverna do DARPA.	84
Tabela 4 – Erro médio e intervalos confiança relativos aos pontos da nuvem obtida pelos métodos na Caverna do DARPA.	88
Tabela 5 – Informações relacionadas ao tempo de processamento da odometria e do mapeamento do <i>Registro-EKF</i> na Caverna do DARPA.	89
Tabela 6 – Informações relacionadas ao tempo de processamento da odometria e do mapeamento do <i>RTAB-Map</i> na Caverna do DARPA.	89
Tabela 7 – Erro médio de localização dos métodos na Mina do DARPA.	90
Tabela 8 – Erro da posição final dos métodos na Mina do DARPA.	91
Tabela 9 – Erro médio e intervalos confiança relativos aos pontos da nuvem obtida pelos métodos na Mina do DARPA.	93
Tabela 10 – Informações relacionadas ao tempo de processamento da odometria e do mapeamento do <i>Registro-EKF</i> na Mina do DARPA.	94
Tabela 11 – Informações relacionadas ao tempo de processamento da odometria e do mapeamento do <i>RTAB-Map</i> na Mina do DARPA.	94
Tabela 12 – Diferença média entre a localização dos métodos e a referência no experimento teleoperado na Mina do Veloso.	95
Tabela 13 – Diferença entre a posição final dos métodos e a referência no experimento teleoperado na Mina do Veloso.	95
Tabela 14 – Diferença média e intervalos confiança relativos aos pontos da nuvem obtida pelos métodos no experimento teleoperado na Mina do Veloso.	98
Tabela 15 – Informações relacionadas ao tempo de processamento da odometria e do mapeamento do <i>Registro-EKF</i> no experimento teleoperado na Mina do Veloso.	99
Tabela 16 – Informações relacionadas ao tempo de processamento da odometria e do mapeamento do <i>RTAB-Map</i> no experimento teleoperado na Mina do Veloso.	99

Tabela 17 – Diferença média entre a localização dos métodos e a referência no experimento semiautônomo na Mina do Veloso.	99
Tabela 18 – Diferença entre a posição final dos métodos e a referência no experimento semiautônomo na Mina do Veloso.	100
Tabela 19 – Diferença média e intervalos confiança relativos aos pontos da nuvem obtida pelos métodos no experimento semiautônomo na Mina do Veloso.	103
Tabela 20 – Informações relacionadas ao tempo de processamento da odometria e do mapeamento do <i>Registro-EKF</i> no experimento semiautônomo na Mina do Veloso.	103
Tabela 21 – Informações relacionadas ao tempo de processamento da odometria e do mapeamento do <i>RTAB-Map</i> no experimento semiautônomo na Mina do Veloso.	104

Lista de Abreviaturas e Siglas

CNN Redes Neurais Convolucionais

DARPA Agência de Projetos de Pesquisa Avançada de Defesa

EKF Filtro de Kalman Estendido

EKF Extended Kalman Filter

F2F Frame-To-Frame

F2M Frame-To-Map

GPS Sistema de Posicionamento Global

ICR Centro de Rotação Instantâneo

IMU Unidade de Medição Inercial

ITV Instituto Tecnológico Vale

LeGO-LOAM *Lightweight and Ground-Optimized LiDAR Odometry and Mapping*

LiDAR *Light Detection and Ranging*

LTM Memória de Termo Longo

PDF Função Densidade de Probabilidade

PnP *Perspective-n-Point*

RGB *Red, Blue, Green*

RGB-D *Red, Blue, Green and Depth*

ROS *Robot Operating System*

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

SE Especial Euclidiano

SO Especial Ortogonal

SfM *Structure from Motion*

SMC Monte Carlo Sequencial

SLAM Localização e Mapeamento Simultâneos

SLAM Simultaneous Localization and Mapping

STM Memória de Termo Curto

UFMG Universidade Federal de Minas Gerais

VIO Odometria Visual-Inercial

VO Odometria Visual

WM Memória de Trabalho

Lista de Símbolos

b_c Distância relativa entre as lentes da câmera estéreo no processamento de imagens

b_f Bias de escorregamento e atrito no EKF

$c_{c,x}$ Translação em x do centro óptico da câmera no processamento de imagens

$c_{c,y}$ Translação em y do centro óptico da câmera no processamento de imagens

c_f Coeficiente de escorregamento e atrito no EKF

$c_{m,x}$ Raio de busca no mapa obtido pelo método X na métrica de mapeamento

c_r Distância lateral do centro do robô ao Centro de Rotação Instantâneo

c_s Constante dos somatórios no processo de SLAM baseado em grafos

\mathbf{d}_f Vetor dos deslocamento lineares corrigidos no EKF

d_f Deslocamento linear no EKF

$d_{l,r}$ Distância percorrida pela referência na métrica de localização

$d_{m,x}$ Distância entre os pontos do método X e a referência na métrica de mapeamento

d_r Distância entre as rodas do robô

\mathcal{F}_C Sistemas de coordenadas da câmera utilizado no registro de pontos

\mathcal{F}_I Sistemas de coordenadas inercial utilizado no registro de pontos

\mathcal{F}_R Sistemas de coordenadas do robô utilizado no registro de pontos

F_f Matriz da dinâmica do sistema do EKF

$F_{\bar{f}}$ Matriz da dinâmica do sistema do Filtro de Kalman

f_f Função não linear da dinâmica do sistema do EKF

f_c Distância focal da câmera no processamento de imagens

f_c Distância focal da câmera RGB no registro de pontos

G_f Matriz da relação entre as entradas e os estados do sistema do Filtro de Kalman

g_s Função da relação entre entradas e estados no processo de SLAM baseado em grafos

H_f Matriz da relação entre os estados do sistema e a saída do EKF

$H_{\bar{f}}$ Matriz da relação entre os estados do sistema e a saída do Filtro de Kalman

$H_{f,i}$ Matriz da relação entre estados do sistemas e a medição da IMU no EKF

$H_{f,o}$ Matriz da relação entre os estados do sistemas e a medição da odometria no EKF

h_f Função não linear da relação entre os estados do sistema e a saída do EKF

h_s Função da relação entre estados e medições no processo de SLAM baseado em grafos

\mathcal{I}_c Imagem capturada pela câmera no processamento de imagens

K_c Matriz de parâmetros intrínsecos da câmera no processamento de imagens

k Posição qualquer em um elemento

\mathcal{M}_s Mapa do ambiente no processo de SLAM

\mathbf{m}_l Métrica relacionada a localização de um método

\mathbf{m}_m Métrica relacionada ao mapeamento de um método

m_f Métrica relacionada a posição final de um método

m_s Característica do mapa do ambiente no processo de SLAM

m_t Métrica relacionada ao custo computacional de um método

\mathcal{N} Distribuição Gaussiana para uma dada média e uma dada covariância

$\mathbf{n}_{m,r}$ Norma do plano do triângulo do *mesh* de referência na métrica de mapeamento

n Quantidade qualquer de elementos

n_l Quantidade de posições após a interpolação nas métricas

$n_{l,r}$ Quantidade de posições de referência na métrica de localização

$n_{l,x}$ Quantidade de posições obtidas pelo método X na métrica de localização

$n_{m,x}$ Quantidade de pontos na nuvem do método X na métrica de mapeamento

$\mathcal{O}_{l,r}$ Conjunto de orientações de referência na métrica de localização

$\mathcal{O}_{l,x}$ Conjunto de orientações obtidas pelo método X na métrica de localização
 $\overline{\mathcal{O}}_{l,r}$ Conjunto de orientações de referência após a interpolação na métrica de localização
 $\overline{\mathcal{O}}_{l,x}$ Conjunto de orientações do método X após a interpolação na métrica de localização
 $\mathbf{o}_{l,r}$ Vetor da orientação de referência na métrica de localização
 $\mathbf{o}_{l,x}$ Vetor da orientação obtida pelo método X na métrica de localização
 $\overline{\mathbf{o}}_{l,r}$ Vetor da orientação de referência após a interpolação na métrica de localização
 $\overline{\mathbf{o}}_{l,x}$ Vetor da orientação do método X após a interpolação na métrica de localização
 $\hat{\mathbf{o}}_f$ Vetor de orientação estimado no EKF
 $\mathcal{P}_{l,r}$ Conjunto de posições de referência na métrica de localização
 $\mathcal{P}_{l,x}$ Conjunto de posições obtidas pelo método X na métrica de localização
 $\mathcal{P}_{m,r}$ Parte da nuvem de pontos de referência na métrica de mapeamento
 $\mathcal{P}_{m,x}$ Nuvem de pontos obtida pelo método X na métrica de mapeamento
 $\overline{\mathcal{P}}_{l,r}$ Conjunto de posições de referência após a interpolação na métrica de localização
 $\overline{\mathcal{P}}_{l,x}$ Conjunto de posições do método X após a interpolação na métrica de localização
 P_s Probabilidade a posteriori da estimação dos estados na solução do processo de SLAM
 \mathbf{p} Vetor da posição de um ponto
 \mathbf{p}_c Vetor da posição de um ponto observado pela câmera no processamento de imagens
 $\mathbf{p}_{l,r}$ Vetor da posição de referência na métrica de localização
 $\mathbf{p}_{l,x}$ Vetor da posição obtida pelo método X na métrica de localização
 $\mathbf{p}_{m,r}$ Vetor de um ponto da nuvem de referência na métrica de mapeamento
 $\mathbf{p}_{m,x}$ Vetor de um ponto da nuvem obtida pelo método X na métrica de mapeamento
 $\overline{\mathbf{p}}$ Vetor homogeneizado da posição de um ponto
 $\overline{\mathbf{p}}_c$ Vetor homogeneizado da posição de um ponto no registro de pontos
 $\overline{\mathbf{p}}_c^C$ Vetor da posição de um ponto com respeito à camera no registro de pontos
 $\overline{\mathbf{p}}_c^I$ Vetor da posição de um ponto com respeito ao inercial no registro de pontos
 $\overline{\mathbf{p}}_c^R$ Vetor da posição de um ponto com respeito ao robô no registro de pontos

$\bar{\mathbf{p}}_f$ Vetor de posição final no EKF

$\bar{\mathbf{p}}_{l,r}$ Vetor da posição de referência após a interpolação na métrica de localização

$\bar{\mathbf{p}}_{l,x}$ Vetor da posição do método X após a interpolação na métrica de localização

$\bar{\mathbf{p}}_{m,x}$ Vetor da projeção de um ponto do método X na métrica de mapeamento

$\bar{\mathbf{p}}_s$ Vetor homogeneizado da posição de um ponto no processo de odometria visual

Q_s Matriz da relação das medições no processo de SLAM baseado em grafos

R Matriz de rotação

R_f Matriz do ganho do EKF

$R_{\bar{f}}$ Matriz do ganho do Filtro de Kalman

\mathbf{r}_c Vetor da posição de um pixel no processamento de imagens

$\mathbf{r}_{\bar{c}}$ Vetor da posição de um pixel na câmera RGB no registro de pontos

$\mathbf{r}_{c,d}$ Vetor da posição de um pixel na câmera direita no processamento de imagens

$\mathbf{r}_{c,e}$ Vetor da posição de um pixel na câmera esquerda no processamento de imagens

r_f Ganho de escorregamento e atrito relacionado ao θ no EKF

$\mathcal{S}_{m,r}$ Parte do *mesh* de referência na métrica de mapeamento

S_f Matriz auxiliar para cálculo do ganho do EKF

$S_{\bar{f}}$ Matriz auxiliar no cálculo do ganho do Filtro de Kalman

S_s Matriz da relação dos estados no processo de SLAM baseado em grafos

$\mathbf{s}_{m,r}$ Triângulo do *mesh* de referência na métrica de mapeamento

s_c Coeficiente de distorção da câmera no processamento de imagens

s_f Ganho final de escorregamento e atrito no EKF

T Matriz de transformação homogênea

T_C^R Matriz de transformação homogênea entre a câmera e o robô no registro de pontos

T_R^I Matriz de transformação homogênea entre o robô e o inercial no registro de pontos

T_s Matriz de transformação homogênea no processo de odometria visual

\mathbf{t} Vetor de translação

$\mathbf{t}_{l,x}$ Vetor do tempo de processamento de cada dado de localização

$\mathbf{t}_{m,x}$ Vetor do tempo de processamento de cada dado de mapeamento

t Instante qualquer de tempo

\mathbf{u}_f Vetor de entradas do sistema do EKF

$\mathbf{u}_{\bar{f}}$ Vetor de entradas do sistema do Filtro de Kalman

\mathbf{u}_s Vetor das entradas na solução do processo de SLAM

\mathbf{u}_s Vetor das entradas no processo de SLAM baseado em grafos

u_c Posição da linha de um pixel no processamento de imagens

$u_{\bar{c}}$ Posição horizontal de um pixel na câmera RGB no registro de pontos

$u_{c,d}$ Posição horizontal de um pixel na câmera direita no processamento de imagens

$u_{c,e}$ Posição horizontal de um pixel na câmera esquerda no processamento de imagens

V_f Matriz de covariância dos ruídos do sistema do EKF

$V_{\bar{f}}$ Matriz de covariância dos ruídos do sistema do Filtro de Kalman

\mathbf{v}_f Vetor de ruídos do sistema do EKF

$\mathbf{v}_{\bar{f}}$ Vetor de ruídos do sistema do Filtro de Kalman

$\mathbf{v}_{m,r}$ Vetor de um vértice do triângulo do *mesh* de referência na métrica de mapeamento

v_c Posição da coluna de um pixel no processamento de imagens

$v_{\bar{c}}$ Posição vertical de um pixel na câmera RGB no registro de pontos

v_r Velocidade linear do robô

$v_{r,d}$ Velocidade linear da roda direita do robô

$v_{r,e}$ Velocidade linear da roda esquerda do robô

W_f Matriz de covariância dos ruídos da medição do EKF

$W_{\bar{f}}$ Matriz de covariância dos ruídos da medição do Filtro de Kalman

$W_{f,i}$ Matriz de covariância dos ruídos da medição da IMU no EKF

$W_{f,o}$ Matriz de covariância dos ruídos da medição da odometria no EKF

\mathbf{w}_f Vetor de ruídos da medição do EKF

$\mathbf{w}_{\bar{f}}$ Vetor de ruídos da medição do Filtro de Kalman

\mathbf{x} Vetor qualquer de estados

\mathbf{x}_f Vetor de estados do EKF

$\mathbf{x}_{\bar{f}}$ Vetor de estados do Filtro de Kalman

\mathbf{x}_s Vetor dos estados do robô na solução do processo de SLAM

\mathbf{x}_s Vetor dos estados no processo de SLAM baseado em grafos

$\bar{\mathbf{x}}_s$ Vetor auxiliar no cálculo dos estados no processo de SLAM baseado em grafos

$\hat{\mathbf{x}}_f$ Vetor de estimação dos estados do EKF

$\hat{\mathbf{x}}_{\bar{f}}$ Vetor de estimação dos estados do Filtro de Kalman

x_c Posição em x de um ponto observado pela câmera no processamento de imagens

$x_{l,x}$ Posição em x obtida pelo método X na métrica de localização

x_r Posição do robô ao longo do eixo x dada pela odometria das rodas

$\bar{x}_{l,r}$ Posição em x do método X após a interpolação na métrica de localização

$\bar{x}_{l,x}$ Posição em x do método X após a interpolação na métrica de localização

\dot{x}_r Velocidade linear do robô ao longo do eixo x

\hat{x}_f Estimação da posição x no EKF

\mathbf{y}_f Vetor da saída do EKF

$\mathbf{y}_{\bar{f}}$ Vetor da saída do Filtro de Kalman

$\mathbf{y}_{f,i}$ Vetor de saída com a medição da IMU no EKF

$\mathbf{y}_{f,o}$ Vetor de saída com a medição da odometria no EKF

$\bar{\mathbf{y}}_f$ Vetor da diferença entre os estados e as medições do EKF

$\bar{\mathbf{y}}_{\bar{f}}$ Vetor da diferença entre os estados e as medições do Filtro de Kalman

y_c Posição em y de um ponto observado pela câmera no processamento de imagens

$y_{l,x}$ Posição em y obtida pelo método X na métrica de localização

y_r Posição do robô ao longo do eixo y dada pela odometria das rodas

$\bar{y}_{l,r}$ Posição em y do método X após a interpolação na métrica de localização

$\bar{y}_{l,x}$ Posição em y do método X após a interpolação na métrica de localização
 \dot{y}_r Velocidade linear do robô ao longo do eixo y
 \hat{y}_f Estimação da posição y no EKF
 \mathbf{z}_s Vetor das medições do ambiente na solução do processo de SLAM
 \mathbf{z}_s Vetor das medições no processo de SLAM baseado em grafos
 $\bar{\mathbf{z}}_s$ Vetor auxiliar no cálculo das medições no processo de SLAM baseado em grafos
 z_c Posição em z de um ponto observado pela câmera no processamento de imagens
 $z_{l,x}$ Posição em z obtida pelo método X na métrica de localização
 $\bar{z}_{l,r}$ Posição em z do método X após a interpolação na métrica de localização
 $\bar{z}_{l,x}$ Posição em z do método X após a interpolação na métrica de localização
 \hat{z}_f Estimação da posição z no EKF
 β_r Constante de escorregamento do robô
 $\theta_{l,x}$ Orientação θ obtida pelo método X na métrica de localização
 $\bar{\theta}_{l,r}$ Orientação θ do método X após a interpolação na métrica de localização
 $\bar{\theta}_{l,x}$ Orientação θ do método X após a interpolação na métrica de localização
 $\hat{\theta}_f$ Estimação da orientação θ no EKF
 ρ_c Profundidade de um ponto observado pela câmera no processamento de imagens
 Σ_f Matriz de covariância da estimacão dos estados do EKF
 $\Sigma_{\bar{f}}$ Matriz de covariância da estimacão dos estados do Filtro de Kalman
 $\phi_{l,x}$ Orientação ϕ obtida pelo método X na métrica de localização
 $\bar{\phi}_{l,r}$ Orientação ϕ do método X após a interpolação na métrica de localização
 $\bar{\phi}_{l,x}$ Orientação ϕ do método X após a interpolação na métrica de localização
 $\hat{\phi}_f$ Estimacão da orientacão ϕ no EKF
 $\psi_{l,x}$ Orientação ψ obtida pelo método X na métrica de localização
 ψ_r Orientação do robô ao redor do eixo z dada pela odometria das rodas
 $\bar{\psi}_{l,r}$ Orientação ψ do método X após a interpolação na métrica de localização

$\bar{\psi}_{l,x}$ Orientação ψ do método X após a interpolação na métrica de localização

$\dot{\psi}_r$ Velocidade angular do robô ao redor do eixo z

$\hat{\psi}_f$ Estimação da orientação ψ no EKF

ω_r Velocidade angular do robô

Sumário

1	Introdução	26
1.1	Motivação	27
1.1.1	Dificuldades Relacionadas a Ambientes Confinados	27
1.1.2	EspeleoRobô	29
1.2	Problema e Objetivos	31
1.2.1	Objetivo Geral	32
1.2.2	Objetivos Específicos	32
1.3	Contribuições da Dissertação	32
1.4	Organização da Dissertação	33
2	Revisão Bibliográfica	34
2.1	Plataformas Robóticas para Inspeção	34
2.2	Localização	37
2.2.1	Odometria	38
2.2.2	Fusão Sensorial	39
2.3	Mapeamento	41
2.4	Localização e Mapeamento Simultâneos (SLAM)	42
3	Fundamentos Teóricos	45
3.1	Modelos para Robôs Móveis	45
3.2	Filtro de Kalman Estendido	47
3.3	Processamento de Imagens	50
3.4	SLAM Visual	54
4	Metodologia	59
4.1	Localização com Filtro de Kalman Estendido	59
4.2	Mapeamento com Registro da Nuvem de Pontos	62
4.3	SLAM com RTAB-Map	64
5	Arcabouço Experimental	68
5.1	Pacotes de ROS Utilizados	68
5.1.1	Implementação do EKF	68
5.1.2	Implementação do Registro de Pontos	70
5.1.3	Implementação do RTAB-Map	71
5.2	Experimentos em Ambientes Simulados	72
5.2.1	Modelo Simulado do EspeleoRobô	73
5.2.2	Ambientes Simulados	74
5.3	Experimentos em Ambientes Reais	76
5.3.1	EspeleoRobô	76

5.3.2	Ambientes Reais	77
6	Resultados e Discussões	80
6.1	Técnicas de Avaliação Empregadas	80
6.2	Resultados na Caverna do DARPA	84
6.3	Resultados na Mina do DARPA	89
6.4	Resultados do Experimento Teleoperado na Mina du Veloso	94
6.5	Resultados do Experimento Semiautônomo na Mina du Veloso	99
7	Conclusões	105
7.1	Trabalhos Futuros	106
	Referências	108

Capítulo 1

Introdução

Robôs estão cada vez mais presentes no cotidiano das pessoas, desempenhando as mais diversas tarefas. Muitas delas são consideradas perigosas, repetitivas, cansativas para serem realizadas por pessoas e o uso do robôs nesses casos ocorre com o intuito de preservar a segurança e a saúde humana. Em alguns cenários, há ainda restrições relacionadas à presença e permanência de seres humanos em certos locais, como é o caso de ambientes confinados. Geralmente esses locais demandam frequentes inspeções para garantir seu funcionamento correto, ou inclusive de exploração para o caso de serem desconhecidos, e é comum a substituição de seres humanos por robôs para a realização de tais atividades.

Tarefas de inspeção e exploração necessitam certa interação com o ambiente, de forma a se localizar, detectar obstáculos e objetos, planejar rotas e navegar no local. Para tanto, existem diversos tipos de sensores que robôs podem utilizar de modo a adquirir informações de estados e condições atuais. Dentre eles estão os *encoders* de rodas e a Unidade de Medição Inercial (IMU), que são sensores mais simples capazes de obter dados locais do robô, sendo bastante empregados em atividades mais básicas de localização e navegação em cenários mais controlados. Por outro lado, as câmeras e os sensores *Light Detection and Ranging* (LiDAR) são capazes de obter grande quantidade de informação do ambiente à distância, algo imprescindível em tarefas de detecção de obstáculos e objetos e planejamento de rotas. Nesses casos, o uso de um modelo de ambiente é essencial, o qual pode ser utilizado como base em tarefas de monitoramento ou inclusive ser criado à medida em que o robô navega em atividades de exploração. Outra finalidade importante para os mapas consiste na visualização e inspeção humana, de modo a identificar pontos críticos em um dado local, como desmoronamentos ou identificação de seres humanos em uma caverna. Dessa forma, o processo geralmente envolve o uso de câmeras, onde o robô deve ser capaz de gerar um mapa preciso, denso (para o caso de nuvem de pontos) e, se possível, colorido, visto que a aparência é importante nesse contexto.

Dentre as diversas técnicas capazes de gerar mapas por meio de imagens, foram investigados nesta dissertação dois métodos de reconstrução visual de ambientes confina-

dos, utilizando câmeras embarcadas em robôs móveis terrestres. O primeiro corresponde no registro da nuvem de pontos com câmeras *Red, Blue, Green and Depth (RGB-D)*, combinado a uma localização obtida com os dados de odometria das rodas e de uma *IMU* aplicados em um Filtro de Kalman Estendido (*EKF*). O segundo método consiste no processo de Localização e Mapeamento Simultâneos (*SLAM*) com o *Real-Time Appearance-Based Mapping (RTAB-Map)*, também utilizando câmeras *RGB-D*. Foram realizados quatro experimentos em ambientes simulados e reais de cavernas e minas, onde os resultados de ambos os métodos são analisados e comparados.

1.1 Motivação

A utilização de dispositivos robóticos é bastante comum no âmbito industrial, justamente pelos riscos associados às atividades desenvolvidas nesses locais, como é o caso da exploração mineral. A mineração está entre as principais atividades econômicas brasileiras, movimentando mais de 100 bilhões de reais na extração de substâncias metálicas anualmente (*Agência Nacional de Mineração, 2019*). O país possui mais de 200 minas em operação, com os estados de Minas Gerais e do Pará mantendo 89% da produção nacional de metais. Uma das principais regiões de extração mineral do país é o quadrilátero ferrífero, localizado no centro-sul de Minas Gerais, sendo responsável por grande parte da produção nacional de ferro e ouro.

A extração mineral apresenta desafios geralmente relacionados ao tipo de ambiente. Em muitos casos, este tipo de atividade ocorre em áreas subterrâneas, como minas e cavernas, onde a presença e a permanência de seres humanos apresentam uma série de riscos à saúde. Nesse contexto, o uso de robôs vem sendo cada vez mais comum, de forma a substituir seres humanos em tarefas como inspeção, extração de materiais, manutenção, exploração e mapeamento. Porém, apesar das vantagens associadas à segurança, a utilização de robôs nesses locais apresenta uma série de desafios.

1.1.1 Dificuldades Relacionadas a Ambientes Confinados

A Norma Regulatória N^o33 (NR-33) define ambientes confinados como espaços não projetados para ocupação humana contínua, possuindo meios limitados de entrada e saída, ventilação insuficiente para a remoção de partículas contaminantes ou ainda com deficiência (ou enriquecimento) de oxigênio (*Ministério do Trabalho e Emprego, 2006*). A norma estabelece ainda medidas administrativas e pessoais, técnicas de prevenção e capacitação, além das responsabilidades de empregadores e trabalhadores, de forma a garantir permanentemente a segurança e saúde de todos que interagem direta ou indiretamente com esses espaços.

Em áreas industriais e urbanas, diversos ambientes como barragens, embarcações, túneis, cavernas, tubulações e galerias subterrâneas necessitam de constante inspeção e manutenção, de forma a garantir o funcionamento correto do local. Essas atividades podem ser exaustivas, perigosas, apresentando riscos à saúde e à segurança de operadores humanos, inclusive para pessoal especializado. Dentre os perigos presentes, podem ser citados a presença de animais peçonhentos, gases nocivos, excrementos de animais e o risco de desmoronamento. Sendo assim, a redução do número de intervenções humanas nesses ambientes evita a exposição desnecessária de pessoas a riscos de acidente de trabalho (Rezende et al., 2020).

Uma possível forma de preservar a saúde humana em ambientes insalubres consiste na utilização de robôs móveis, algo que tem sido cada vez mais comum atualmente. Conforme apresentado na Figura 1, esses dispositivos podem substituir seres humanos em tarefas de inspeção, exploração, mapeamento ou mesmo manutenção de estruturas, evitando fadiga ou riscos à saúde. A utilização de robôs para a execução de tarefas em ambientes confinados apresenta uma série de restrições e desafios, demandando soluções de problemas ainda em aberto. As principais restrições encontradas nesses ambientes envolvem a interação com seres humanos e as condições do ambiente. Dentre estas, podem ser citadas: a falta de sinal de Sistema de Posicionamento Global (GPS), terrenos escorregadios e acidentados, baixa visibilidade, presença de partículas suspensas no ar e interferência de sensores magnéticos e de comunicação.



Figura 1 – Comparação da presença de humanos e robôs em ambientes confinados.

Essas restrições têm ganhado atenção em diversas áreas da robótica. Por exemplo, a Agência de Projetos de Pesquisa Avançada de Defesa (DARPA) tem organizado diversos desafios nessa área, entre eles o DARPA Subterranean (SubT) Challenge^{1,2}, iniciado em

¹ <https://www.subtchallenge.com/>

² <https://www.darpa.mil/program/darpa-subterranean-challenge>

2018 e com previsão de término para 2021 (Rouček et al., 2020). O principal objetivo desse desafio consiste em desenvolver novas tecnologias voltadas para operações subterrâneas com dispositivos robóticos móveis. Nesse cenário, diferentes equipes competem entre si em tarefas de mapeamento, navegação, inspeção e exploração de ambientes subterrâneos complexos, como cavernas naturais e túneis construídos por humanos.

1.1.2 EspeleoRobô

A segurança e a saúde do trabalhador são fatores de extrema importância no contexto comercial e industrial, principalmente tratando-se da presença humana em ambientes insalubres. Dessa forma, a Companhia de Mineração Vale S.A. tornou obrigatório em 2007 a formação em Requisitos de Atividades Críticas (RAC) para a realização de quaisquer atividades em áreas confinadas, onde operadores e monitores recebem treinamento teórico e prático na execução segura de tarefas em tais ambientes. Nesse contexto, o Instituto Tecnológico Vale (ITV), em parceria com a Universidade Federal de Minas Gerais (UFMG), vem desenvolvendo o projeto intitulado “Dispositivo Robótico para a Inspeção de Ambientes Restritos e Confinados”, que tem como objetivo a elaboração de um robô móvel para reduzir eventuais riscos relacionados à presença humana em locais de difícil acesso (Azpurua et al., 2019).

O dispositivo robótico, denominado EspeleoRobô, é baseado no RHex da Boston Dynamics (Saranli et al., 2001), com a proposta original de inspecionar cavernas naturais durante missões teleoperadas. Conforme ilustrado na Figura 2, o robô tem sido atualmente utilizado na inspeção e monitoramento de galerias, barragens, tubulações e espaços confinados industriais (Rezende et al., 2020). Além disso, é possível utilizar diferentes sensores acoplados ao dispositivo, dependendo da necessidade e da possibilidade de uso.

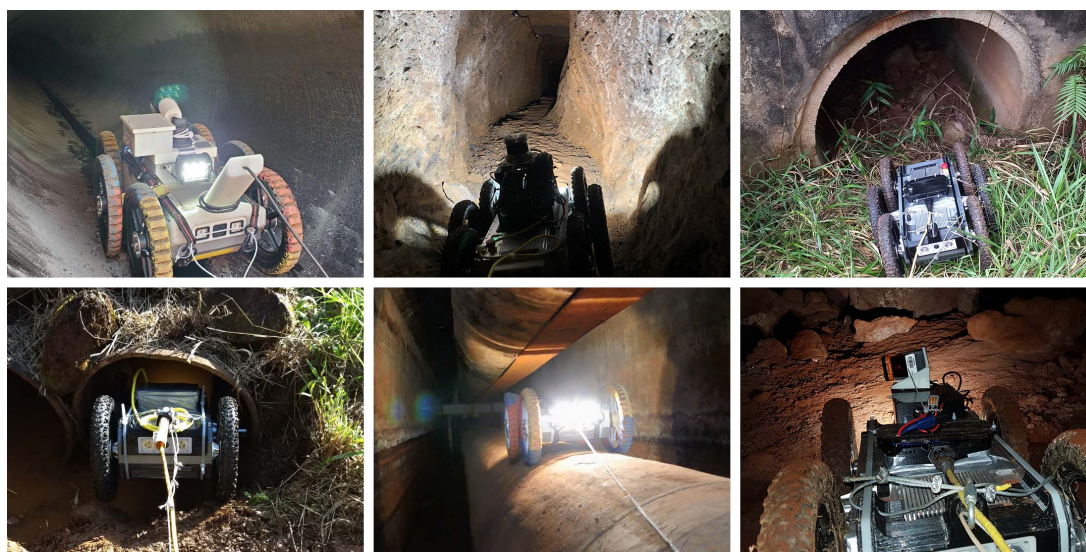


Figura 2 – EspeleoRobô em tarefas de inspeção e monitoramento em diferentes locais.

O robô, ilustrado em detalhes na Figura 3, foi projetado com dimensões reduzidas de 0,55m x 0,25m x 0,14m e peso de 22kg (com capacidade de carga extra de 10kg), permitindo sua entrada em ambientes relativamente estreitos. Ele também apresenta um encapsulamento mecânico com certificação IP67, que garante alta proteção contra partículas sólidas e água. Além disso, o robô possui pinos de liberação em seus mecanismos de locomoção, permitindo uma rápida troca na utilização de pernas, rodas, formas padrão (como estrelas) e configurações híbridas. Por padrão, as rodas são os mecanismos mais frequentemente utilizados, principalmente devido sua eficiência energética, estabilidade e simplicidade de operação (Azpurua et al., 2019). O fato do EspeleoRobô ser um dispositivo terrestre pode ser justificado por sua eficiência e autonomia de funcionamento, robustez e simplicidade de uso, possibilidade de entrar em locais estreitos e por ser capaz de carregar uma carga extra relativamente alta, incluindo os diversos sensores utilizados.

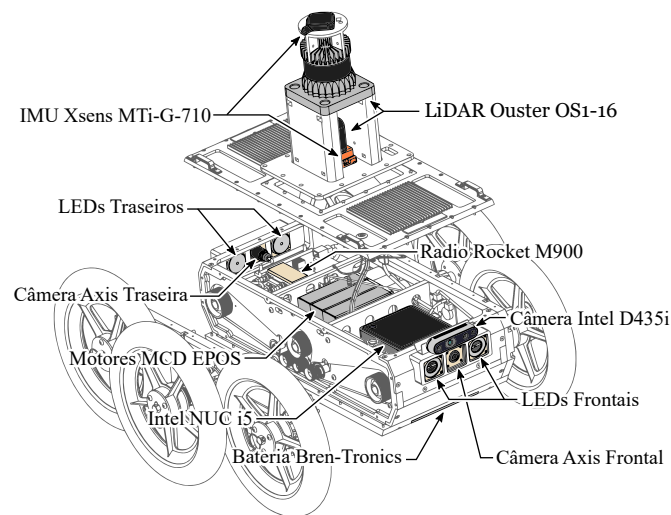


Figura 3 – Diagrama do EspeleoRobô e seus sensores.

Dentro de seu encapsulamento, há um conjunto de seis motores MCD EPOS da Maxon Motor de 60W³, sendo um para cada roda; duas baterias BB-2590U da Bren-Tronics⁴; uma câmera Axis P1224-E⁵ e um par de LEDs Cree XLamp XHP35⁶ posicionados na dianteira e na traseira do robô; um rádio Ubiquiti Rocket M900⁷ para comunicação sem fio; e um mini PC Intel NUC Core i5⁸, utilizando o *Robot Operating System (ROS)* (Quigley et al., 2009) com Ubuntu 16.04. O robô possui uma torre móvel na parte superior onde estão instalados diferentes sensores, entre eles: uma IMU Xsens MTi-G-710 GNSS/INS⁹ e um sensor LiDAR Ouster OS1-16¹⁰. Além disso, na parte frontal do robô

³ <https://www.maxongroup.com/maxon/view/product/326343>

⁴ <https://bren-tronics.com/batteries/rechargeable/bt-70791ck.html>

⁵ <https://axis.com/pt-br/products/axis-p1224-e>

⁶ <https://cree-led.com/products/xhp-family-leds/xhp-leds>

⁷ <https://ui.com/airmax/rocketm/>

⁸ <https://intel.com.br/content/www/br/pt/products/boards-kits/nuc.html>

⁹ <https://xsens.com/products/mti-100-series>

¹⁰ <https://ouster.com/products/os1-lidar-sensor/>

está posicionada uma câmera RGB-D Intel RealSense Depth D435i¹¹, conforme mostrado na Figura 3. Esta câmera corresponde ao principal sensor utilizado nesta dissertação devido ao objetivo geral de mapeamento explicado a seguir.

Mais recentemente, estão sendo realizados diversos estudos que visam dar autonomia nas missões do EspeleoRobô, dentre os quais: planejamento de caminhos (Souza Santos et al., 2019), métodos de localização e controle de navegação (Rezende et al., 2020), sistemas de alerta e navegação assistida (Amaral et al., 2020), geração de mapas do ambiente (da Cruz Júnior et al., 2020; Fernandes et al., 2020), métodos para exploração autônoma (Azpúrua et al., 2021) entre outros.

1.2 Problema e Objetivos

A utilização de mapas é essencial para o desenvolvimento de diversas aplicações na robótica móvel, como localização, navegação e planejamento de caminhos (Burgard et al., 2016). Eles podem ser definidos como um modelo de representação de um determinado ambiente, sendo conhecidos previamente ou construídos durante a exploração utilizando dados de diferentes tipos de sensores. Nesse contexto, tarefas como inspeções e coletas de informações de ambientes desconhecidos podem ser realizadas por meio dos mapas gerados. Assim, áreas perigosas podem ser exploradas por robôs e posteriormente analisadas por especialistas, evitando, assim, riscos à saúde humana.

O uso de câmeras é comum em aplicações de localização e mapeamento, pelo fato de serem dispositivos leves, passivos, geralmente com baixo consumo de energia, porém capazes de adquirir muita informação do ambiente (Fuentes-Pacheco et al., 2015). Além disso, o emprego desses dispositivos como sensores de mapeamento possibilita a construção de mapas coloridos e densos, onde é possível observar a cor, a textura e o formato dos objetos. Esses fatores, somados à precisão, contribuem na produção de mapas realistas e mais apropriados para a visualização humana, características essenciais em aplicações de análise e inspeção de ambientes.

Além dos problemas relacionados aos ambientes confinados citados anteriormente, como falta de posicionamento global e terrenos escorregadios, a realização de tarefas de mapeamento por meio de imagens em tais locais deve levar em consideração o custo computacional envolvido, a presença de partículas (como poeira ou neblina) e as condições de iluminação. Esta última corresponde a um dos problemas mais críticos em atividades que envolvem câmeras, geralmente ocasionando efeitos de desfoque (*motion blur*) nas imagens. Porém, não foi realizada uma análise direta de operações com baixa luminosidade, visto que o sistema de iluminação instalado no EspeleoRobô mostrou-se suficiente nos ambientes explorados.

¹¹ <https://intelrealsense.com/depth-camera-d435i/>

1.2.1 Objetivo Geral

O objetivo geral desta dissertação consiste em gerar reconstruções densas, coloridas e precisas de ambientes confinados para fins posteriores de inspeção e análise estrutural por especialistas em espeleologia, utilizando modelos digitais e realidade virtual. É importante evidenciar que a finalidade dos mapas consiste na inspeção visual, e não na navegação do dispositivo móvel.

1.2.2 Objetivos Específicos

Para a geração de reconstruções densas e coloridas de ambientes confinados, são apresentados e comparados dois métodos visuais de mapeamento, utilizando imagens de câmeras RGB-D. Dessa forma, os objetivos específicos desta dissertação são:

- Implementar um EKF para gerar uma localização 3D precisa com informações de odometria das rodas e IMU;
- Implementar um método de registro de nuvem de pontos utilizando dados externos de odometria para obter as transformadas entre as nuvens;
- Analisar o desempenho de localização e mapeamento do RTAB-Map em diferentes cenários;
- Realizar e processar experimentos simulados e reais similares às condições de trabalho do EspeleoRobô;
- Comparar e validar as reconstruções geradas por ambos os métodos nos ambientes utilizados para os experimentos.

1.3 Contribuições da Dissertação

Os estudos e experimentos realizados para a elaboração desta dissertação motivaram uma série de contribuições. Primeiramente, algumas técnicas de Odometria Visual (VO) e Odometria Visual-Inercial (VIO) foram estudadas a fim de determinar métodos com resultados precisos para tarefas de localização com o EspeleoRobô. A estimação da pose com a ferramenta OpenVINO, utilizando uma câmera estéreo, foi um dos métodos analisados que demonstrou bons resultados. Nesse contexto, Rezende et al. (2020) apresentam uma análise de diferentes métodos de odometria com o artigo *Indoor Localization and Navigation Control Strategies for a Mobile Robot Designed to Inspect Confined Environments*¹², publicado no *Conference on Automation Science and Engineering (CASE)*.

¹² <https://doi.org/10.1109/CASE48305.2020.9217005>

Na sequência, devido à necessidade de mapeamento dos ambientes trabalhados com o EspeleoRobô, foi implementado um algoritmo de estimação da profundidade para a câmera estéreo. Assim, utilizando informações de pose e de profundidade, foi desenvolvida uma técnica de registro de pontos, de forma a gerar mapas e complementar os resultados anteriores. Além disso, alguns métodos de **SLAM** Visual foram estudados, dentre os quais o **RTAB-Map** se destacou, demonstrando uma localização precisa e uma reconstrução densa e precisa. Nesse ponto, surgiu a oportunidade de se trabalhar com uma câmera **RGB-D**, o que aprimorou as informações de profundidade e, conseqüentemente, a qualidade dos mapas gerados. Alguns resultados com o **RTAB-Map** e com o registro de pontos, utilizando câmeras estéreo e **RGB-D**, são apresentados por [Fernandes et al. \(2020\)](#) com o artigo *Investigation of Visual Reconstruction Techniques for Mobile Robots in Confined Environments*¹³, o qual foi publicado no *Latin American Robotics Symposium (LARS)* e corresponde à principal publicação relacionada a esta dissertação.

As simulações também estiveram presentes durante o processo de elaboração desta dissertação, onde foi possível testar e avaliar as técnicas em diferentes cenários sob condições similares às enfrentadas pelo EspeleoRobô. Desse modo, foram avaliados alguns resultados de **SLAM** Visual com o **RTAB-Map** utilizando câmeras **RGB-D**, em experimentos envolvendo ambientes reais e simulados. Esses resultados são apresentados por [Azpúrua et al. \(2021\)](#) com o artigo *Towards Semi-Autonomous Robotic Inspection and Mapping in Confined Spaces with the EspeleoRobô*¹⁴, publicado no *Journal of Intelligent and Robotic Systems (JINT)*. Além disso, foi realizado um estudo à parte utilizando um **EKF** para gerar uma localização relativamente precisa com dados da odometria das rodas e de uma **IMU**, resultando no pacote de **ROS** chamado `espeleo_pose_ekf`.

1.4 Organização da Dissertação

A seguir, no Capítulo 2, é apresentada uma revisão bibliográfica sobre assuntos relacionados à reconstrução de ambientes, onde são abordados os temas de inspeção de áreas confinadas, localização, mapeamento e **SLAM**. Os fundamentos que envolvem a parte teórica necessária para o entendimento desta dissertação são apresentados no Capítulo 3. No Capítulo 4, é explicada a metodologia, onde é demonstrado o funcionamento dos métodos utilizados. A parte experimental é apresentada no Capítulo 5, contendo as implementações específicas dos métodos e os detalhes dos experimentos realizados em ambientes simulados e reais. Por fim, são apresentados os resultados e discussões no Capítulo 6 e as conclusões e trabalhos futuros no Capítulo 7.

¹³ <https://doi.org/10.1109/LARS/SBR/WRE51543.2020.9307018>

¹⁴ <https://doi.org/10.1007/s10846-021-01321-5>

Capítulo 2

Revisão Bibliográfica

Este Capítulo apresenta alguns trabalhos presentes na literatura, com o objetivo de explorar, conhecer e compreender técnicas recentes que estão de certa forma relacionadas com a reconstrução de ambientes confinados utilizando robôs móveis. Nesse contexto, são primeiramente apresentados diversos tipos de robôs móveis com foco em inspeção de ambientes. Em seguida, diversos exemplos de localização de robôs são investigados, incluindo métodos de odometria e de fusão sensorial de uma forma mais detalhada. Na sequência, são apresentadas algumas técnicas de mapeamento de ambientes e, por fim, alguns métodos recentes de [SLAM](#) presentes na literatura.

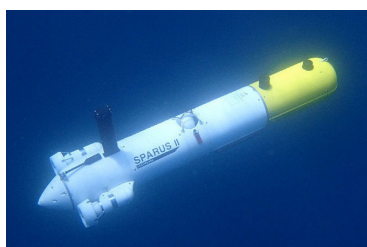
2.1 Plataformas Robóticas para Inspeção

O uso de robôs tem sido cada vez mais comum em tarefas de exploração e inspeção de ambientes confinados, de forma a preservar a segurança e a saúde humana. Nesse contexto, [Martz et al. \(2020\)](#) mostram uma investigação de dispositivos robóticos apresentados na literatura como solução para o problema de exploração e mapeamento de áreas subterrâneas, com um foco maior em túneis para mineração. Esses ambientes podem apresentar diversas peculiaridades, como: terrenos escorregadios, pedregosos e/ou desnivelados; caminhos com paredes ou muros, onde só é possível atravessar escalando ou voando; e áreas com presença de água ou inclusive inundadas. Essas adversidades implicam na utilização de diferentes tipos de robôs, os quais podem ser divididos em três principais grupos quanto ao método de locomoção: aquáticos, aéreos e terrestres. Há também a possibilidade de utilizar veículos híbridos, os quais possuem mais de um tipo de locomoção.

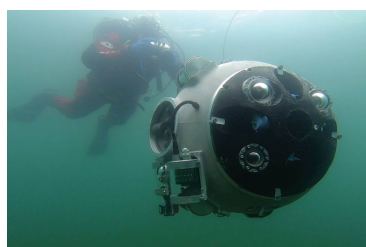
Tratando de espaços aquáticos, [Carreras et al. \(2018\)](#) apresentam um veículo subaquático autônomo denominado *Sparus II* (Figura 4-a), desenvolvido para a inspeção do solo oceânico. [Martins et al. \(2018\)](#) demonstram a implementação do *UX 1* (Figura 4-b), um sistema robótico para exploração de minas subterrâneas inundadas. Por outro

lado, referindo-se a espaços aéreos, [Balaram et al. \(2018\)](#) apresentam um helicóptero, posteriormente nomeado como *Ingenuity* (Figura 4-c), que está sendo atualmente utilizado para exploração aérea em Marte. [Alexis \(2019\)](#) exhibe resultados de exploração autônoma, mapeamento e planejamento de trajetórias em minas subterrâneas utilizando um robô aéreo (Figura 4-d).

Há também robôs híbridos, capazes de variar seu tipo de locomoção de acordo com o ambiente, algo que pode se tornar bastante útil devido a possíveis adversidades encontradas em espaços confinados. Nesse contexto, [Kalantari et al. \(2020\)](#) demonstram a implementação do *Drivocopter* (Figura 4-e), desenvolvido para a locomoção por terra ou pelo ar. Por outro lado, [Maity et al. \(2013\)](#) apresentam um robô anfíbio para a exploração de minas inundadas (Figura 4-f), capaz de se deslocar por terra e de nadar sobre água.



(a) *Sparus II*
([Carreras et al., 2018](#))



(b) *UX 1*
([Martins et al., 2018](#))



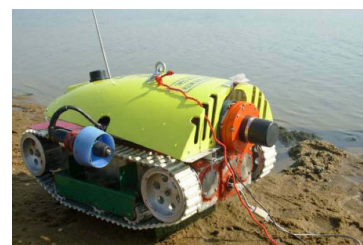
(c) *Ingenuity*
([Balaram et al., 2018](#))



(d)
([Alexis, 2019](#))



(e) *Drivocopter*
([Kalantari et al., 2020](#))



(f)
([Maity et al., 2013](#))

Figura 4 – Exemplos de robôs aquáticos, aéreos e híbridos desenvolvidos para exploração de ambientes.

Veículos terrestres são amplamente utilizados em tarefas de exploração e inspeção de ambientes. [Zhai et al. \(2020\)](#) apresentam robôs de resgate presentes na literatura, em particular para o caso terrestre utilizando visão binocular em minas de carvão. Comparado a outros métodos de locomoção, esses veículos podem transportar cargas relativamente pesadas, possuem alta autonomia de funcionamento e são capazes de entrar em locais estreitos. Os veículos terrestres podem ainda ser subdivididos quanto ao seu dispositivo de locomoção, sendo os mais utilizados: esteiras, rodas articuladas, pernas e rodas.

O uso de robôs com esteiras é bastante comum para a locomoção em ambientes terrestres. Nesse contexto, [Wang et al. \(2014\)](#) apresentam os dispositivos robóticos *MINBOT-I* e *MINBOT-II* (Figura 5-a), criados para aplicações de resgate em minas sub-

terrâneas de carvão. Com o mesmo propósito, o robô *CUMT-V* (Figura 5-b) é mostrado por Li et al. (2020b). Há também um grande uso de robôs com rodas articuladas, principalmente relacionado à exploração espacial. Desse modo, Cordes et al. (2018) exibem o *SherpaTT*, com rodas articuladas para a exploração de terrenos análogos a Marte (Figura 5-c), e Reid et al. (2020) vêm desenvolvendo o robô *RoboSimian* (Figura 5-d), para terrenos similares à Europa (lua de Júpiter). Outra forma de locomoção cada vez mais viável com o recente avanço computacional está relacionada ao uso de pernas, com os robôs bioinspirados. Nesse sentido, Hutter et al. (2016) e Bledt et al. (2018) apresentam respectivamente os robôs quadrúpedes *ANYmal* (Figura 5-e) e *MIT Cheetah 3* (Figura 5-f), ambos desenvolvidos para uma locomoção robusta em ambientes realmente desafiadores.



(a) *MINBOT-II*
(Wang et al., 2014)



(b) *CUMT-V*
(Li et al., 2020b)



(c) *SherpaTT*
(Cordes et al., 2018)



(d) *RoboSimian*
(Reid et al., 2020)



(e) *ANYmal*
(Hutter et al., 2016)



(f) *MIT Cheetah 3*
(Bledt et al., 2018)

Figura 5 – Exemplos de robôs terrestres que utilizam esteiras, rodas articuladas e pernas, desenvolvidos para inspeção e exploração de ambientes.

Robôs com rodas representam a classe mais comum de dispositivos robóticos terrestres, sendo utilizados já há algumas décadas, sobretudo em função da simplicidade de operação, eficiência energética e robustez. No início dos anos 2000, já era possível encontrar dispositivos robóticos realizando tarefas de exploração, como o *Groundhog* (Figura 6-a), apresentado por Morris et al. (2005), capaz de realizar mapeamento e planejamento de caminhos em minas abandonadas. Mais recentemente, Losch et al. (2018) demonstram um robô com rodas e um braço robótico, denominado *Julius* (Figura 6-b), para atividades de mapeamento, navegação e manipulação de minas subterrâneas. Ebadi et al. (2020) apresentam dois robôs terrestres denominados *Husky A200* (Figura 6-c), utilizados em tarefas de exploração envolvendo mapeamento e posicionamento autônomos, no contexto do DARPA Subterranean Challenge.



Figura 6 – Exemplos de robôs com rodas utilizados em tarefas de exploração e mapeamento de espaços subterrâneos.

Conforme apresentado anteriormente, os propósitos atuais do projeto do EspeleoRobô consistem em explorar e inspecionar espaços confinados. Dessa forma, o robô utilizado nesta dissertação consiste em um dispositivo robótico terrestre com rodas fixas, conforme ilustrado nas Figuras 2 e 3.

2.2 Localização

O processo de localização corresponde a estimação da posição, ou possivelmente da pose, de um dispositivo em um determinado ambiente. O termo pose é bastante utilizado no contexto de localização, representando de forma simultânea a posição e a orientação de um objeto. No âmbito da robótica, praticamente todas as aplicações necessitam que o robô se localize em um determinado local, incluindo tarefas de inspeção e exploração. Além disso, no processo de reconstrução de ambientes, a qualidade da localização possui uma grande influência na precisão e na deformação dos mapas gerados.

Uma das formas de realizar a localização de um dispositivo ocorre por meio de métodos diretos, onde a posição (ou a pose) é estimada de forma direta. Nesse contexto, o Sistema de Posicionamento Global (GPS) representa um dos métodos de localização em ambientes externos mais utilizados, sendo aplicado nas mais diversas áreas, inclusive em robôs móveis autônomos na agricultura conforme apresentado por [Thenmozhi et al. \(2019\)](#). Porém, para o caso de ambientes internos, outras abordagens devem ser empregadas devido a ausência de sinal de GPS. Uma das formas consiste em incluir alguns marcadores no local com posições (ou pose) conhecidas, como é demonstrado por [Nazemzadeh et al. \(2017\)](#), utilizando Código de Resposta Rápida (QR Code) para a estimação da pose de robôs móveis. De forma similar, [Retscher et al. \(2019\)](#) apresentam resultados de posicionamento em ambientes internos utilizando *Ultra-Wideband* (UWB) e Wi-Fi.

Por outro lado, em ambientes internos e de difícil acesso, não há sinal de GPS e a inclusão de objetos ou sensores com posições conhecidas pode se tornar impraticável. Dessa forma, existem métodos incrementais para a estimação relativa da pose, como a odometria, apresentada a seguir.

2.2.1 Odometria

O princípio básico da odometria consiste em estimar o movimento de um dispositivo entre dois intervalos sequenciais, o qual é integrado ao longo do tempo de forma a obter uma estimativa da pose relativa a um estado inicial. A estimativa desse movimento sequencial pode ser computada por meio de diversos métodos utilizando diferentes tipos de sensores, sendo os principais: *encoders*, IMU, sensores LiDAR e câmeras. Mohamed et al. (2019) apresentam um estudo geral sobre diversos métodos de localização, com um foco maior em odometria, além de processos que utilizam de fusão sensorial, que serão abordados a seguir.

A odometria de rodas (ou odometria por *encoders*) corresponde ao processo mais simples e mais antigo de localização incremental. A técnica é amplamente utilizada na robótica devido principalmente à sua simplicidade e ao seu baixo custo computacional. Conforme apresentado por Dudek and Jenkin (2016), o processo de odometria de rodas consiste em desenvolver um modelo matemático capaz de interpretar o movimento das rodas (ou juntas) de forma a estimar a pose relativa em função do tempo. Esse modelo varia de acordo com o projeto do veículo, podendo ser simples, como o diferencial para robôs de duas rodas, ou um pouco mais complexos, como o modelo *skid-steering*, atualmente projetado para o EspeleoRobô. Um exemplo de utilização desse último modelo é apresentado por Boas and Conceição (2020), exibindo resultados de odometria utilizando uma modelagem cinemática para o robôs do tipo *skid-steering*. De uma forma mais aprofundada, Yamauchi et al. (2017) apresentam um modelo de compensação para escorregamento das rodas, visto que este é um problema sempre presente em veículos desse tipo.

Sensores *Light Detection and Ranging* (LiDAR) têm sido muito utilizados no âmbito da robótica móvel, principalmente pelos recentes avanços tecnológicos que garantiram maior acessibilidade e redução em seu custo comercial. O termo odometria LiDAR é relativamente novo comparado aos outros métodos, sendo primeiramente apresentado por Zhang and Singh (2014) com o *LiDAR Odometry and Mapping* (LOAM), o qual define seu uso como um problema de estimativa do movimento por meio de nuvens de pontos obtidas em tempos diferentes. O uso de sensores desse tipo possibilita a redução da deriva (ou *drift*) nos resultados de odometria utilizando técnicas como o fechamento de laço. Tian et al. (2017) demonstram o uso de LiDAR para estimativa de odometria em uma plataforma robótica. Outro exemplo mais recente é apresentado por Li et al. (2020a), utilizando Redes Neurais Convolucionais (CNN) para estimativa de pose com nuvem de pontos.

A Odometria Visual (VO) é outro método de localização bastante aplicado na robótica, realizado por meio de imagens de câmeras. O primeiro exemplo clássico desse tipo de odometria é apresentado por Nister et al. (2004), onde o termo VO é descrito como uma estimativa do movimento utilizando apenas dados visuais como entrada. No

ano seguinte, [Cheng et al. \(2005\)](#) apresentam uma das primeiras aplicações significativas desse tipo de odometria, com robôs *Opportunity* e *Spirit* para a exploração do planeta Marte.

Uma revisão mais extensa de VO é apresentada por [Aqel et al. \(2016\)](#), demonstrando e explicando seus diversos tipos, além de alguns métodos recentes na literatura. Em comparação aos outros sensores utilizados nos processos de localização, as câmeras possuem um custo comercial relativamente baixo e podem ser aplicadas em ambientes fechado. Além disso, elas permitem capturar uma grande quantidade de informação do ambiente, possibilitando resultados satisfatórios de odometria e inclusive a aplicação de técnicas de otimização como o fechamento de laço. Por outro lado, o processo de VO é bastante susceptível às condições do ambiente, principalmente de iluminação, além de demandar alto custo computacional, envolvendo técnicas de processamento de imagens e extração de características.

Atualmente, os métodos de VO são divididos em três categorias: diretos, onde toda a imagem é utilizada no processo de estimação do movimento; baseados em características, onde são utilizados alguns pontos representativos da imagem; e os semi-diretos, que possuem aspectos híbridos dentre os outros dois anteriores. Dois exemplos de métodos diretos são o *Stereo Direct Sparse Odometry* (S-DSO), apresentado por [Wang et al. \(2017\)](#) utilizando câmeras estéreo, e o *Flowdometry*, apresentado por [Muller and Savakis \(2017\)](#) com câmera monocular e CNN. Para os métodos baseados em características, [Geiger et al. \(2011\)](#) e [Cvisic and Petrovic \(2015\)](#) demonstram respectivamente o *Visual Odometry 2* (VISO2) e o *Stereo Odometry based on Feature Selection and Tracking* (SOFT), ambos utilizando câmeras estéreo. Por fim, um exemplo de um método semi-direto com câmera monocular é o *Semi-direct Visual Odometry* (SVO), apresentado por [Forster et al. \(2014\)](#).

Nesta dissertação, para a localização do robô móvel, foram utilizados dados de odometria das rodas e de VO, combinados com outros métodos apresentados a seguir.

2.2.2 Fusão Sensorial

O processo de odometria utilizando apenas um único sensor pode acarretar em imprecisões nos resultados, principalmente pela falta de variedade das informações coletadas do ambiente. Desse modo, a fusão sensorial é um processo amplamente aplicado em diversas áreas da robótica que busca utilizar as vantagens de cada sensor de forma a alcançar melhores resultados finais. [Durrant-Whyte and Henderson \(2016\)](#) descrevem o processo de fusão de dados como a combinação de observações de diferentes sensores, visando obter uma descrição mais robusta e completa do ambiente.

Existem várias categorias diferentes de fusão sensorial, conforme apresentado por [Khaleghi et al. \(2013\)](#), incluindo processos baseados em filtros e em otimização. Dentre

os primeiros, os filtros probabilísticos são os mais utilizados em aplicações na área da robótica. Eles possuem como base principal a regra de *Bayes*, sendo os mais conhecidos: o Filtro Bayesiano, o Filtro de Kalman Estendido (**EKF**) e o método de Monte Carlo Sequencial (**SMC**). Os processos baseados em otimização consistem em resolver problemas de minimização de erro, podendo ser realizado de diversas formas, como o algoritmo de Levenberg-Marquardt. Além disso, outra categoria de fusão está relacionada ao modo de combinação entre os dados dos sensores, os quais podem estar fortemente ou fracamente acoplados.

Os sensores inerciais, como giroscópios e acelerômetros, são os principais dispositivos utilizados nos processos de fusão. Conforme apresentado em [Dudek and Jenkin \(2016\)](#), esses sensores são capazes de fornecer dados de velocidade angular e aceleração linear, informações úteis nos processos de localização. Embora sejam dispositivos muito comuns e de tamanho e custo reduzidos, os sensores inerciais são bastante propícios a ruídos, não sendo suficientes para alcançar bons resultados de forma isolada.

Os métodos mais simples e mais antigos de fusão de dados para odometria envolvem a combinação de informações de *encoders* e de sensores inerciais. Um exemplo disso é apresentado por [Barshan and Durrant-Whyte \(1995\)](#), realizando a fusão desses sensores por meio de um **EKF**. Mais recentemente, [Berntorp \(2015\)](#) demonstra a utilização de um método de **SMC** em um veículo automotor.

Na medida em que os sensores **LiDAR** foram ficando mais acessíveis, métodos de odometria que realizam fusão com seus dados tornaram-se mais comuns em tarefas de localização. Nesse contexto, [Zhou et al. \(2020\)](#) apresentam o *Region of Interest-cloud* (ROI-cloud), um método de odometria que utiliza dados de sensores **LiDAR** e inerciais combinados em um **EKF**. De forma similar, [Qin et al. \(2020\)](#) demonstram o *LiDAR-Inertial State Estimator* (LINS), que realiza a fusão entre os dados utilizando Filtro de Bayes e **SMC**.

Para a fusão de dados de sensores **LiDAR** com imagens de câmeras, [Zhang and Singh \(2015\)](#) apresentam o *Visual-LiDAR Odometry and Mapping* (V-LOAM), com uma odometria por fusão fortemente acoplada, aplicando um filtro próprio e específico para o caso. Outro exemplo é o *Tightly-coupled Vision-LiDAR Odometry* (TVLO), apresentado por [Seo and Chou \(2019\)](#), que utiliza um método de otimização fortemente acoplado para combinar os dados de sensores **LiDAR** e câmeras.

Por fim, um método de odometria com fusão sensorial frequentemente utilizado, e com resultados satisfatórios, corresponde a Odometria Visual-Inercial (**VIO**), a qual permite combinar dados de câmeras e sensores inerciais. Nesse contexto, [Huang \(2019\)](#) apresenta um estudo bastante completo do processo de **VIO**, explicando suas diferentes categorias, alguns dos métodos presentes na literatura e outros tópicos como o modelo

cinemático utilizado para a [IMU](#) e para a câmera. Em relação aos tipos de estimação para [VIO](#), os métodos baseados em características se destacam sobre os diretos. Dentre eles, podem ser citados: o *Multi-State Constraint Kalman Filter* (MSCKF), que utiliza câmeras monoculares e um [EKF](#) fortemente acoplado, apresentado por [Mourikis and Roumeliotis \(2007\)](#); o *Open Keyframe-based Visual* (OKVIS), com uma fusão baseada em otimização não linear fortemente acoplada para câmeras monoculares e estéreo, descrito por [Leutenegger et al. \(2015\)](#); o *Robust Visual Inertial Odometry* (ROVIO), utilizando imagens monoculares e um [EKF](#) intimamente acoplado, demonstrado por [Bloesch et al. \(2015\)](#); e o *Visual-Inertial System* (VINS) baseado em otimização com fusão fortemente acoplada para câmeras monoculares, apresentado por [Qin et al. \(2018\)](#).

Nesta dissertação, devido à simplicidade e ao baixo custo computacional, a odometria das rodas (citada anteriormente) foi combinada com dados inerciais de uma [IMU](#) por meio de um [EKF](#), de forma a gerar uma localização 3D. Essa informação foi utilizada como entrada para o método de registro da nuvem de pontos, apresentado na sequência.

2.3 Mapeamento

A tarefa de mapeamento é definida por [Burgard et al. \(2016\)](#) como a construção de um modelo de um determinado ambiente utilizando dados de sensores, algo que é crucial em diversas aplicações na robótica. Mais especificamente, a exploração de ambientes desconhecidos ou restritos possui grande motivação no âmbito militar, comercial, ou inclusive espacial (em explorações planetárias). Para tal tarefa, é necessária a utilização de sensores que permitam coletar dados do ambiente, os quais são chamados de exteroceptivos, sendo os mais utilizados as câmeras e os sensores [LiDAR](#).

O registro de nuvem de pontos utilizando sensores [LiDAR](#) corresponde a um processo bem comum em tarefas de mapeamento, o qual consiste em construir um mapa de pontos a partir de diversas nuvens locais. Nesse contexto, [Dong et al. \(2020\)](#) apresentam uma revisão sobre registro de pontos com [LiDAR](#) para veículos terrestres, explicitando alguns métodos recentes na literatura, além de alguns conceitos relacionados ao processo. De forma mais específica, [Magnusson et al. \(2007\)](#) demonstram um algoritmo de registro de pontos aplicados para robôs autônomos em tarefas de mapeamento. Mais recentemente, [Dong et al. \(2018\)](#) apresentam um método de registro [LiDAR](#) em veículos terrestres para nuvens de pontos que não são sequenciais.

O registro de pontos pode também ser realizado com câmeras (preferencialmente [RGB-D](#)) de forma a adicionar a informação das cores dos pontos, resultando em mapas coloridos e mais densos. [Morell-Gimenez et al. \(2018\)](#) apresentam um estudo de diversos métodos de registro de pontos utilizando câmeras [RGB-D](#) em ambientes estáticos, separando-os em diferentes categorias de forma a comparar seus resultados. Um método

de registro global de pontos robusto utilizando câmera RGB-D é apresentado por Halber and Funkhouser (2017), com o objetivo de reduzir problemas como deriva e perda de rastreamento em grandes ambientes. Em um contexto mais específico, Zeisl et al. (2013) demonstram resultados de um método de registro automático com câmera RGB-D, visando uma reconstrução densa e geometricamente bem estruturada de ambientes fechados.

Outro processo utilizado em tarefas de reconstrução de ambientes é a fotogrametria, muito comum em áreas da geociência e com aplicações em robótica. Nesse processo, são capturadas imagens de um ambiente e algoritmos de *Structure from Motion* (SfM) são utilizados para determinar a pose da câmera, possibilitando a reconstrução do local imageado. Um estudo de SfM é realizado por Özyeşil et al. (2017), apresentando algumas técnicas atuais, assim como todo o procedimento teórico envolvido. Moulon et al. (2013) demonstram resultados precisos de reconstruções de monumentos utilizando um novo método de calibração entre pares. Em outro contexto, Jaud et al. (2020) apresentam resultados de monitoramento morfodinâmico de zonas costeiras utilizando fotogrametria.

Com o objetivo de obter um mapa denso e colorido, o primeiro método de reconstrução utilizado nesta dissertação corresponde ao registro da nuvem de pontos por meio de câmeras RGB-D, combinado a uma localização dada pelo EKF (citado anteriormente).

2.4 Localização e Mapeamento Simultâneos (SLAM)

Stachniss et al. (2016) definem o processo de SLAM como o problema de geração de um mapa espacial do ambiente por um dispositivo móvel, enquanto, ao mesmo tempo, este se localiza dentro desse mapa. Esse processo de localização e mapeamento simultâneos, é muito importante no âmbito da robótica, principalmente em cenários que envolvem exploração de ambientes restritos e confinados. Conforme explicado anteriormente, o processo de mapeamento envolve o uso de sensores exteroceptivos, sendo praticamente utilizados sensores LiDAR ou câmeras. Nesse contexto, Huang et al. (2019) apresentam um detalhamento amplo sobre os diversos métodos recentes de SLAM (LiDAR e Visual), além de alguns exemplos de sensores.

Há diversas pesquisas recentes envolvendo SLAM com sensores LiDAR. Um estudo comparativo de alguns métodos presentes na literatura é apresentado por da Cruz Júnior et al. (2020), utilizando um robô terrestre móvel em ambientes fechados e em uma mina subterrânea. Os dois métodos que alcançaram os melhores resultados nesse contexto foram o *Lightweight and Ground-Optimized LiDAR Odometry and Mapping* (LeGO-LOAM) e o HDL-Graph-SLAM, apresentados por Shan and Englot (2018) e Koide et al. (2019), respectivamente. Outros exemplos de resultados de SLAM utilizando sensores LiDAR com robôs móveis em minas subterrâneas são demonstrados por Papachristos et al. (2019) e Ren et al. (2019).

Com o recente avanço tecnológico, o desenvolvimento de processadores gráficos mais potentes e a comercialização de câmeras com menor custo, o **SLAM** Visual vem sendo um processo amplamente utilizado na robótica móvel, alcançando resultados bastante satisfatórios. [Fuentes-Pacheco et al. \(2015\)](#) realizam um estudo de métodos de **SLAM** que utilizam apenas câmeras como sensores exteroceptivos, abordando as principais etapas envolvidas no processo. Uma comparação entre métodos de **SLAM** Visual é apresentado por [Filipenko and Afanasyev \(2018\)](#), utilizando robôs móveis em ambientes fechados. No mesmo contexto, [Ibragimov and Afanasyev \(2018\)](#) demonstram resultados de uma comparação mais específica, utilizando métodos baseados em **ROS** em ambientes fechados homogêneos.

Dentre os métodos presentes na literatura, o *Large-Scale Direct monocular SLAM* (LSD-SLAM), apresentado por [Engel et al. \(2014\)](#), se destaca como uma abordagem direta que utiliza ajuste de imagens de câmeras monoculares para estimar a pose e construir um mapa denso. Em contraste, [Whelan et al. \(2016\)](#) exhibe o *ElasticFusion*, um método que utiliza câmeras **RGB-D** com estimação de pose baseada em características e mapeamento semidenso baseado em superfícies. O *Oriented FAST and Rotated BRIEF SLAM 2* (ORB-SLAM2), apresentado por [Mur-Artal and Tardos \(2017\)](#), possibilita o uso de câmeras monoculares, estéreo ou **RGB-D**, fornecendo uma odometria robusta e um mapa esparsos de características. [Pire et al. \(2017\)](#) apresenta o *Stereo Parallel Tracking and Mapping* (S-PTAM), o qual aceita abordagens distintas para detecção e descrição de recursos para gerar um mapa esparsos por meio de câmeras estéreo. Por fim, [Labbé and Michaud \(2019\)](#) apresenta o *Real-Time Appearance-Based Mapping* (RTAB-Map), um método de **SLAM** baseado em grafos que pode ser utilizado com dados de câmeras estéreo ou **RGB-D**, capaz de produzir um mapa semidenso.

Uma comparação entre os dois métodos recentes de **SLAM** Visual mais conhecidos, ORB-SLAM2 e RTAB-Map, é apresentado por [Ragot et al. \(2019\)](#). Alguns resultados em ambientes fechados, dessa vez com o *RGB-D Mapping*, são demonstrados por [Henry et al. \(2014\)](#). Também em ambientes fechados, [Kameyama and Hidaka \(2018\)](#) apresentam resultados de reconstrução visual utilizando o RTAB-Map. Por último, de uma forma mais específica, [Fernandes et al. \(2020\)](#) apresentam resultados com o RTAB-Map e o registro da nuvem de pontos em corredores fechados e em minas subterrâneas.

Dessa forma, nesta dissertação, foi utilizado o RTAB-Map como segundo método de reconstrução de ambientes, o qual realiza os processos de **VO** e de mapeamento por meio de câmeras **RGB-D**.

Por fim, para uma melhor visualização, a Tabela 1 apresenta uma comparação entre os métodos citados de localização e mapeamento que utilizam câmeras como sensores principais. Na Tabela, é possível observar o registro de pontos, apresentado como um método geral (sem um autor em particular), e o RTAB-Map.

Métodos	Autores	Sensores Principais	Método de Odometria	Tipo do Mapa
ElasticFusion	Whelan et al. (2016)	RGB-D	Baseado em características	Semidenso
Flowdometry	Muller and Savakis (2017)	Monocular	CNN	-
Fotogrametria	-	Monocular	SfM	Semidenso
LSD-SLAM	Engel et al. (2014)	Monocular	Direto	Denso
MSCKF	Mourikis and Roumeliotis (2007)	Monocular IMU	Baseado em características	Mapa de características
OKVIS	Leutenegger et al. (2015)	Monocular Estéreo IMU	Baseado em características	Mapa de características
ORB-SLAM2	Mur-Artal and Tardos (2017)	Monocular Estéreo RGB-D	Baseado em características	Mapa de características
Registro de Pontos	-	RGB-D	Variados	Denso
ROVIO	Bloesch et al. (2015)	Monocular IMU	Baseado em características	Mapa de características
RTAB-Map	Labbe and Michaud (2019)	Estéreo RGB-D	Baseado em características	Semidenso
S-DSO	Wang et al. (2017)	Estéreo	Direto	Semidenso
SOFT	Cvisic and Petrovic (2015)	Estéreo	Baseado em características	Mapa de características
S-PTAM	Pire et al. (2017)	Estéreo	Baseado em características	Esparso
SVO	Forster et al. (2014)	Monocular	Semi-direto	Mapa de características
VINS	Qin et al. (2018)	Monocular IMU	Baseado em características	Mapa de características
VISO2	Geiger et al. (2011)	Estéreo	Baseado em características	Mapa de características

Tabela 1 – Tabela comparativa de métodos de localização e mapeamento que utilizam câmeras como sensores principais.

Capítulo 3

Fundamentos Teóricos

Neste Capítulo, são apresentados os fundamentos teóricos necessários para o desenvolvimento das metodologias aplicadas. Primeiramente, os modelos para robôs móveis são apresentados e, na sequência, as equações que envolvem o [EKF](#) são demonstradas. Em seguida, são descritos alguns fundamentos envolvendo capturas de imagens e tipos de câmeras e, por fim, são apresentados algumas etapas que envolvem o processo de [SLAM](#).

3.1 Modelos para Robôs Móveis

Robôs com rodas têm sido amplamente usados em diversas aplicações na robótica, possuindo algumas vantagens como simplicidade na estrutura, baixo custo de fabricação, eficiência energética e possibilidade de movimentar-se em altas velocidades ([Chung and Iagnemma, 2016](#)). Há diversas configurações possíveis de robôs com rodas, sendo levados em consideração fatores como a quantidade, o tipo e a posição das rodas. O modelo mais simples e bastante utilizado em tarefas mais básicas de locomoção é o diferencial, o qual possui duas rodas ativas (ou controláveis) com eixos de giro coincidentes. Nesse caso, as rodas ativas devem estar fixas no robô e são frequentemente tratadas como roda direita e roda esquerda. Estas se movem em formas de arcos, cujos centros são coincidentes em um ponto denominado Centro de Rotação Instantâneo ([ICR](#)), o qual está alinhado com o eixo das rodas ([Dudek and Jenkin, 2016](#)).

Uma representação do robô diferencial é apresentado na [Figura 7](#), assim como as variáveis utilizadas e o sistema de coordenadas. Considerando as velocidades lineares $v_{r,d}$ e $v_{r,e}$ e a distância d_r entre as rodas, as respectivas velocidades linear v_r e angular ω_r podem ser obtidas por:

$$\begin{bmatrix} v_r \\ \omega_r \end{bmatrix} = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2d_r} & -\frac{1}{2d_r} \end{bmatrix} \begin{bmatrix} v_{r,d} \\ v_{r,e} \end{bmatrix}. \quad (3.1)$$

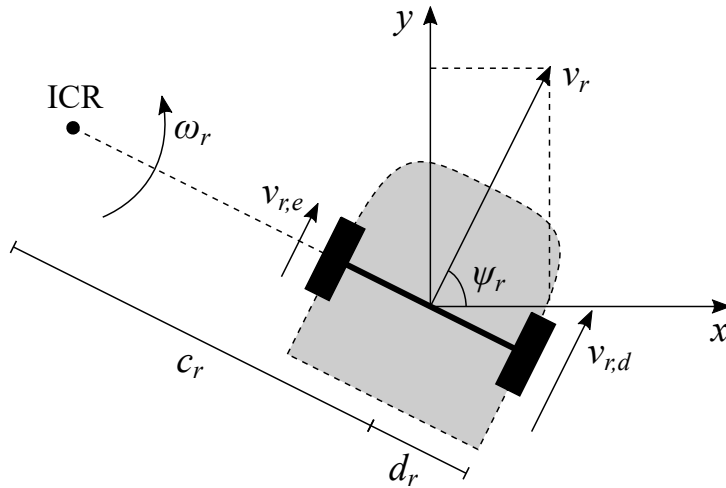


Figura 7 – Representação do modelo diferencial com indicação das variáveis e do sistemas de coordenadas.

Entretanto, o modelo diferencial não é indicado para robôs que possuem dois ou mais eixos fixos, como é o caso da maioria dos robôs com quatro (ou seis) rodas, visto que o ICR não necessariamente coincide com todos os eixos ao mesmo tempo. Dessa forma, durante movimentos de curva ou giro do robô, o escorregamento das rodas está sempre presente, o que atrapalha o desempenho e a precisão dos dados obtidos dos *encoders*. Para essa categoria de robôs, o modelo *skid-steering* é mais indicado, por levar em consideração dados da distância lateral do ICR (c_r) e de uma constante de escorregamento (β_r) (Mandow et al., 2007). Esse modelo é uma forma estendida do modelo diferencial, sendo computado por meio de:

$$\begin{bmatrix} v_r \\ \omega_r \end{bmatrix} = \begin{bmatrix} \frac{\beta_r}{2} & \frac{\beta_r}{2} \\ \frac{\beta_r}{2c_r} & -\frac{\beta_r}{2c_r} \end{bmatrix} \begin{bmatrix} v_{r,d} \\ v_{r,e} \end{bmatrix}, \quad (3.2)$$

onde $v_{r,d}$ e $v_{r,e}$ podem, por exemplo, ser obtidos utilizando a média das velocidades das rodas direita e esquerda.

Outra forma de representar o movimento de robôs consiste em expressar as velocidades de translação relacionadas aos eixos x e y e a de rotação em torno de z em função de v_r e ω_r . Para tal, o modelo cinemático diferencial corresponde a uma técnica simples e bastante empregada para diversos cenários. Nesse caso, são adotadas algumas simplificações como movimentos apenas planares do robô, com rodas perpendiculares ao terreno e sem a presença de escorregamento. Assim, analisando a Figura 7 com os respectivos dados de velocidades linear (v_r) e angular (ω_r) e da orientação atual do robô (ψ_r), as velocidades \dot{x}_r , \dot{y}_r e $\dot{\psi}_r$ nesse modelo podem ser computadas por:

$$\begin{bmatrix} \dot{x}_r \\ \dot{y}_r \\ \dot{\psi}_r \end{bmatrix} = \begin{bmatrix} \cos(\psi_r) & 0 \\ \sin(\psi_r) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_r \\ \omega_r \end{bmatrix}. \quad (3.3)$$

Como os valores de orientação e das velocidades lineares e angulares são funções do tempo, é então possível obter uma equação de movimento para o robô. Esse processo de estimação da posição e orientação planar do robô utilizando informações das velocidades ao longo do tempo é conhecido como odometria. Dessa forma, considerando um instante de tempo t e a pose planar inicial do robô em $t = 0$, as posições x_r e y_r e a orientação ψ_r pode ser obtida por meio de:

$$\begin{aligned} x_r[t] &= x_r[0] + \int \dot{x}_r[t] dt, \\ y_r[t] &= y_r[0] + \int \dot{y}_r[t] dt \quad \text{e} \\ \psi_r[t] &= \psi_r[0] + \int \dot{\psi}_r[t] dt. \end{aligned} \quad (3.4)$$

Em condições ditas perfeitas, esse modelo deveria ser o bastante para determinar precisamente a pose planar do robô. Porém, os cálculos de odometria são geralmente realizados em intervalos de tempo discretos, o que produz pequenos erros relacionados à integração numérica, proporcionais à janela de tempo entre as coletas dos dados. Além disso, pequenas incorreções no modelo, como nas medições do tamanho do eixo ou da roda, e incertezas relacionadas às medições de velocidade, como a ocorrência de escorregamento, podem gerar pequenos erros em cada instante de tempo (Dudek and Jenkin, 2016). Esses erros são integrados numericamente, resultando em uma pose final com um desvio que pode crescer de forma indefinida. E esse é um problema ainda maior nos modelos *skid-steering*, visto que o escorregamento está sempre presente.

3.2 Filtro de Kalman Estendido

No mundo real, processos de medição ou determinação de algum evento sempre envolve a presença de incertezas. Em diversos casos, essas incertezas podem gerar erros que, dependendo da precisão desejada, devem ser levados em consideração, principalmente em sistemas que envolvem cálculos integrativos ou derivativos. Dessa forma, é bastante comum a utilização de modelos que consideram as incertezas relacionadas às variáveis do sistema, como, por exemplo, os métodos de estimação probabilística. Nesses métodos, não é utilizado apenas um valor para descrever certo evento, mas sim uma distribuição de probabilidade.

No âmbito da robótica, métodos probabilísticos são geralmente utilizados para combinar dados de dois ou mais sensores, os quais são representados por uma Função Densidade de Probabilidade (PDF). De forma geral, a combinação entre duas ou mais distribuições corresponde ao principal tópico abordado pela estimação probabilística, consistindo em um processo de duas etapas: predição e correção. As estimativas relacionadas a um estado possuem a forma de uma PDF e são propagadas na etapa de predição à medida que o estado varia. Na etapa de correção, as informações coletadas pelos sensores são utilizadas para realizar uma tarefa de correção, combinando a PDF predita na etapa anterior com as distribuições obtidas na medição. Assim, a nova estimativa fornecida pela PDF corrigida é então utilizada pelo processo de estimação (Choset et al., 2005).

O Filtro de Kalman é um método recursivo de estimação probabilística, utilizado para sistemas dinâmicos com presença de ruído. Possui fácil implementação e é amplamente aplicado na robótica, especialmente na área de localização de robôs. Esse filtro é geralmente utilizado em problemas com tempo discreto de modo iterativo, onde os estados e as medições podem ser modelados por meio de um sistema linear dinâmico na forma:

$$\begin{cases} \mathbf{x}_{\bar{f}}[k+1] = F_{\bar{f}}[k]\mathbf{x}_{\bar{f}}[k] + G_{\bar{f}}[k]\mathbf{u}_{\bar{f}}[k] + \mathbf{v}_{\bar{f}}[k] \\ \mathbf{y}_{\bar{f}}[k] = H_{\bar{f}}[k]\mathbf{x}_{\bar{f}}[k] + \mathbf{w}_{\bar{f}}[k] \end{cases} . \quad (3.5)$$

O vetor $\mathbf{x}_{\bar{f}}[k] \in \mathbb{R}^n$ representa os estados do sistema em um instante de tempo k , enquanto que o vetor $\mathbf{y}_{\bar{f}}[k] \in \mathbb{R}^p$ consiste na saída do sistema, contendo os valores obtidos de cada sensor. Por sua vez, o vetor $\mathbf{u}_{\bar{f}}[k] \in \mathbb{R}^m$ corresponde às entradas que são diretamente aplicadas no sistema, como comandos de velocidade, torques ou forças. O vetor $\mathbf{v}_{\bar{f}}[k] \in \mathbb{R}^n$ representa os ruídos envolvidos no processo, definidos como gaussianos de média zero e com matriz de covariância $V_{\bar{f}}[k] \in \mathbb{R}^{n \times n}$. Por outro lado, o vetor $\mathbf{w}_{\bar{f}}[k] \in \mathbb{R}^p$ indica os ruídos relativos à medição dos sensores, também definidos como gaussianos de média zero, porém, com matriz de covariância $W_{\bar{f}}[k] \in \mathbb{R}^{p \times p}$. A matriz $F_{\bar{f}}[k] \in \mathbb{R}^{n \times n}$ corresponde à dinâmica do sistema, enquanto que $G_{\bar{f}}[k] \in \mathbb{R}^{n \times m}$ descreve como as entradas controlam o sistema. Por fim, a matriz $H_{\bar{f}}[k] \in \mathbb{R}^{p \times n}$ representa a influência do vetor de estados na saída do sistema.

O processo de estimação é composto pelas duas etapas de predição e correção, citadas anteriormente. Essas etapas são realizadas em sequência de forma iterativa, sendo utilizados os dados da iteração anterior para estimar o estado atual. Na primeira etapa, a estimação do estado atual $\mathbf{x}_{\bar{f}}$ em k é utilizada para gerar a predição $\mathbf{x}_{\bar{f}}$ em $k+1$ de acordo com a dinâmica do sistema. Nesse caso, realizando a substituição no modelo do sistema da Equação 3.5, a predição pode ser obtida por:

$$\hat{\mathbf{x}}_{\bar{f}}[k+1 | k] = F_{\bar{f}}[k]\hat{\mathbf{x}}_{\bar{f}}[k | k] + G_{\bar{f}}[k]\mathbf{u}_{\bar{f}}[k] . \quad (3.6)$$

Para a covariância relacionada à etapa de predição, é utilizada a definição de matriz de covariância $P = E[(X - E[X])(X - E[X])^T]$. Substituindo a estimação do vetor de estados $\hat{\mathbf{x}}_{\bar{f}}[k + 1 | k]$, pode-se obter:

$$\Sigma_{\bar{f}}[k + 1 | k] = F_{\bar{f}}[k]\Sigma_{\bar{f}}[k | k]F_{\bar{f}}[k]^T + V_{\bar{f}}[k]. \quad (3.7)$$

Na sequência, a predição obtida na etapa anterior é corrigida de acordo com a saída $\mathbf{y}_{\bar{f}}[k + 1]$, de forma a gerar uma nova estimação $\hat{\mathbf{x}}_{\bar{f}}[k + 1 | k + 1]$. Nessa etapa, é escolhido um valor de $\mathbf{x} \in \mathbb{R}^n$, com $\mathbf{y}_{\bar{f}}[k + 1] = H_{\bar{f}}[k + 1]\mathbf{x}$, que maximiza a distribuição definida por $\hat{\mathbf{x}}_{\bar{f}}[k + 1 | k]$ e $\Sigma_{\bar{f}}[k + 1 | k]$. Realizando as substituições necessárias, é possível obter a estimação corrigida:

$$\hat{\mathbf{x}}_{\bar{f}}[k + 1 | k + 1] = \hat{\mathbf{x}}_{\bar{f}}[k + 1 | k] + R_{\bar{f}}\bar{\mathbf{y}}_{\bar{f}}, \quad (3.8)$$

onde:

$$\begin{aligned} \bar{\mathbf{y}}_{\bar{f}} &= \mathbf{y}_{\bar{f}}[k + 1] - H_{\bar{f}}[k + 1]\hat{\mathbf{x}}_{\bar{f}}[k + 1 | k] \\ R_{\bar{f}} &= \Sigma_{\bar{f}}[k + 1 | k]H_{\bar{f}}[k + 1]^T S_{\bar{f}}^{-1} \\ S_{\bar{f}} &= H_{\bar{f}}[k + 1]\Sigma_{\bar{f}}[k + 1 | k]H_{\bar{f}}[k + 1]^T + W_{\bar{f}}[k + 1] \end{aligned} \quad (3.9)$$

Para determinar a covariância relacionada à etapa de correção, é novamente utilizada a definição de matriz de covariância e a estimação do vetor de estados $\hat{\mathbf{x}}_{\bar{f}}[k + 1 | k + 1]$, sendo possível obter:

$$\Sigma_{\bar{f}}[k + 1 | k + 1] = \Sigma_{\bar{f}}[k + 1 | k] - R_{\bar{f}}H_{\bar{f}}[k + 1]\Sigma_{\bar{f}}[k + 1 | k]. \quad (3.10)$$

O Filtro de Kalman se apresenta como uma ferramenta bastante robusta, porém, limitada para resolução de problemas lineares. Muitos dos processos encontrados no mundo real não são lineares, o que inviabiliza a aplicação desse método de filtragem. Nesse sentido, um sistema não linear pode ser dado da forma:

$$\begin{cases} \mathbf{x}_f[k + 1] = f_f(\mathbf{x}_f[k], \mathbf{u}_f[k], k) + \mathbf{v}_f[k] \\ \mathbf{y}_f[k] = h_f(\mathbf{x}_f[k], k) + \mathbf{w}_f[k] \end{cases}, \quad (3.11)$$

sendo os vetores \mathbf{x}_f , \mathbf{y}_f , \mathbf{u}_f , \mathbf{v}_f e \mathbf{w}_f conforme apresentados anteriormente na Equação 3.5 e as funções $f_f : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{Z}^+ \rightarrow \mathbb{R}^n$ e $h_f : \mathbb{R}^n \times \mathbb{Z}^+ \rightarrow \mathbb{R}^p$ continuamente diferenciáveis em $\mathbf{x}_f[k]$.

Para solucionar sistemas desse tipo, pode-se utilizar uma versão estendida do Filtro de Kalman denominada Filtro de Kalman Estendido (EKF). Esse método consiste na

linearização das equações com respeito à estimação atual, de modo a obter uma estimação resultante de forma aproximada. Assim, as equações envolvidas no processo de filtragem permanecem bastante similares às aquelas apresentadas no Filtro de Kalman, sendo para a etapa de predição:

$$\begin{aligned}\hat{\mathbf{x}}_f[k+1|k] &= f_f(\hat{\mathbf{x}}_f[k|k], \mathbf{u}_f[k], k) \\ \Sigma_f[k+1|k] &= F_f[k]\Sigma_f[k|k]F_f[k]^T + V_f[k]\end{aligned}\quad (3.12)$$

e para a correção:

$$\begin{aligned}\hat{\mathbf{x}}_f[k+1|k+1] &= \hat{\mathbf{x}}_f[k+1|k] + R_f\bar{\mathbf{y}}_f \\ \Sigma_f[k+1|k+1] &= \Sigma_f[k+1|k] - R_fH_f[k+1]\Sigma_f[k+1|k]\end{aligned}\quad (3.13)$$

Nesse caso, têm-se:

$$\begin{aligned}\bar{\mathbf{y}}_f &= \mathbf{y}_f[k+1] - h_f(\mathbf{x}_f[k+1|k], k+1) \\ R_f &= \Sigma_f[k+1|k]H_f[k+1]^T S_f^{-1} \\ S_f &= H_f[k+1]\Sigma_f[k+1|k]H_f[k+1]^T + W_f[k+1]\end{aligned}\quad (3.14)$$

onde as matrizes $F_f[k]$ e $H_f[k+1]$ podem ser computadas por:

$$F_f[k] = \left. \frac{\partial f_f}{\partial \mathbf{x}_f} \right|_{\mathbf{x}_f = \hat{\mathbf{x}}_f[k|k]} \quad \text{e} \quad H_f[k+1] = \left. \frac{\partial h_f}{\partial \mathbf{x}_f} \right|_{\mathbf{x}_f = \hat{\mathbf{x}}_f[k+1|k]} \quad (3.15)$$

3.3 Processamento de Imagens

A visão corresponde a um dos sentidos sensoriais mais importantes para os seres vivos. Nesse contexto, dispositivos capazes de simular (ou inclusive aprimorar) a visão humana são amplamente utilizados, envolvendo atividades que vão desde uma foto com a câmera simples até a exploração espacial com grandes telescópios. Em ambos os casos, são coletadas informações de um ambiente tridimensional e transformadas para um espaço bidimensional, o qual denominamos imagem.

Uma imagem pode ser definida como uma função bidimensional, $\mathcal{I}_c(u_c, v_c)$, onde u_c e v_c são coordenadas espaciais (Gonzalez et al., 2009). Em uma imagem digital, o número de elementos da função \mathcal{I}_c é finito, onde cada elemento, geralmente chamado de *pixel*, possui sua posição e seu valor na imagem. Nesse caso, a quantidade de *pixels* existentes na horizontal (colunas) e na vertical (linhas) de uma imagem define sua resolução. Esta é geralmente representada na forma “colunas x linhas”, como 640x480 e 1920x1080, ou com o sufixo “p”, como 480p e 1080p. Uma das formas mais simples para função

corresponde à intensidade (ou escala de cinza), no qual pontos mais brilhantes possuem maiores valores. Outra forma bastante comum é a utilização do sistema de cores *Red, Blue, Green* (RGB), onde as cores do ambiente são representadas por meio da combinação entre os três canais R, G e B. É possível ainda obter imagens fora do espectro visível para o ser humano, como é o caso das câmeras térmicas e de luz ultravioleta.

Os modelos de câmeras mais utilizados atualmente realizam o processo de captura de imagens por meio de projeção geométrica, onde a luz refletida por um objeto é projetada no plano da câmera através de sua lente (Solomon and Breckon, 2010). Dessa forma, um ponto $\mathbf{p}_c = [x_c, y_c, z_c]^T \in \mathbb{R}^3$ no ambiente é mapeado como $\mathbf{r}_c = [u_c, v_c]^T \in \mathbb{R}^2$ no plano de projeção da imagem, conforme ilustrado na Figura 8. Em uma forma mais simples, pode-se considerar a seguinte relação:

$$u_c = \frac{f_c x_c}{z_c} \quad \text{e} \quad v_c = \frac{f_c y_c}{z_c}, \quad (3.16)$$

onde f_c representa a distância entre a lente e o plano da câmera, normalmente chamado de distância focal (Zollhöfer, 2019). De uma maneira mais geral, considerando as translações $c_{c,x}$ e $c_{c,y}$ do centro óptico, a distância focal f_c e o coeficiente de distorção s_c da câmera, é possível determinar uma matriz $K_c \in \mathbb{R}^{3 \times 3}$, onde:

$$\begin{bmatrix} u_c \\ v_c \\ z_c \end{bmatrix} = \begin{bmatrix} f_c & s_c & c_{c,x} \\ 0 & f_c & c_{c,y} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix}. \quad (3.17)$$

Os valores contidos na matriz K_c são chamados parâmetros intrínsecos da câmera e podem ser obtidos por valores predefinidos do fabricante ou por processos de calibração (Bradski and Kaehler, 2008).

Um dos possíveis problemas que aparecem com os métodos de projeção em perspectiva está relacionado ao processo inverso, o qual consiste em obter a posição de um ponto no espaço tridimensional para uma dada posição em uma imagem. A projeção de um ponto 3D em um plano envolve perda de informação, o que pode tornar o processo

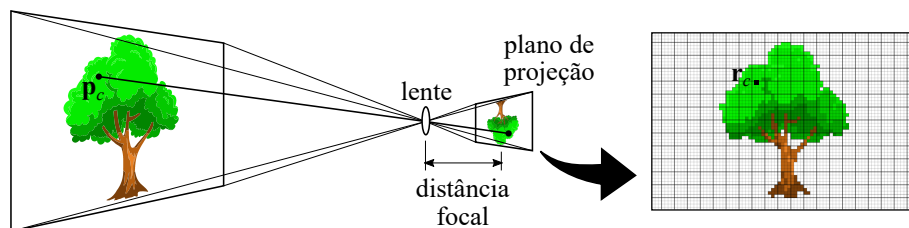


Figura 8 – Representação do processo de captura de imagens por câmeras utilizando projeção em perspectiva.

irreversível utilizando somente os dados da imagem e dos parâmetros da câmera. Uma informação essencial que é perdida com a projeção corresponde à distância entre a câmera e o ponto 3D no ambiente, a qual é normalmente chamada de profundidade. A seguir são apresentados os métodos mais clássicos de determinação ou estimação da profundidade de um ponto.

A estimação da profundidade dos pontos observados pelos sensores é um processo essencial para tarefas de localização e mapeamento. Dependendo do tipo de sensor escolhido, o desempenho do processo de estimação pode variar bastante, onde a melhor opção é dada de acordo com a necessidade. As câmeras monoculares correspondem ao tipo mais simples de sensor, onde é capturada apenas uma imagem por vez (colorida ou monocromática) utilizando sua única lente. A estimação de profundidade para esse modelo de câmera não será tratada nesta dissertação, por representar um processo relativamente complexo, necessitando de um movimento constante do dispositivo de forma a obter uma estimação contínua, a qual ainda sofre problemas de escala. Existem também alguns métodos recentes que envolvem a utilização de redes neurais específicas para a estimação de profundidade, porém, fogem do escopo principal desta dissertação. Os melhores resultados são apresentados pelo par de câmeras estéreo e pelas câmeras RGB-D, explicados na sequência. Há ainda alguns tipos de câmeras que utilizam o processo *Time-of-Flight* (ToF) para determinar a profundidade de um dado ponto, mas que também fogem do escopo principal desta dissertação.

Câmeras estéreo são dispositivos capazes de capturar duas imagens simultâneas do ambiente a partir de suas duas lentes, geralmente separadas de forma horizontal. A noção de profundidade com esse tipo de câmera ocorre de maneira similar à visão humana, podendo ser computado de forma inversa à projeção em perspectiva, conforme apresentado na Figura 9. Nesse caso, um ponto do ambiente $\mathbf{p}_c \in \mathbb{R}^3$ é projetado nos planos das imagens esquerda e direita em locais distintos, $\mathbf{r}_{c,e} \in \mathbb{R}^2$ e $\mathbf{r}_{c,d} \in \mathbb{R}^2$. Supondo uma distância horizontal entre as lentes b_c , a disparidade entre as projeções pode ser calculada com a diferença $u_{c,e} - u_{c,d}$. Assim, utilizando os parâmetros da distância focal f_c e a distância relativa entre o centro de suas lentes b_c , a profundidade ρ_c pode ser computada de acordo com a disparidade entre as imagens em um processo conhecido como triangulação, onde:

$$\rho_c = \frac{b_c f_c}{u_{c,e} - u_{c,d}} . \quad (3.18)$$

As maiores dificuldades enfrentadas com câmeras estéreo estão relacionadas ao cálculo da disparidade entre as imagens direita e esquerda. Nesse contexto, regiões com texturas de fácil detecção facilitam o processo de casamento entre os *pixels*, gerando resultados mais precisos na estimação da disparidade. Em contrapartida, locais muito

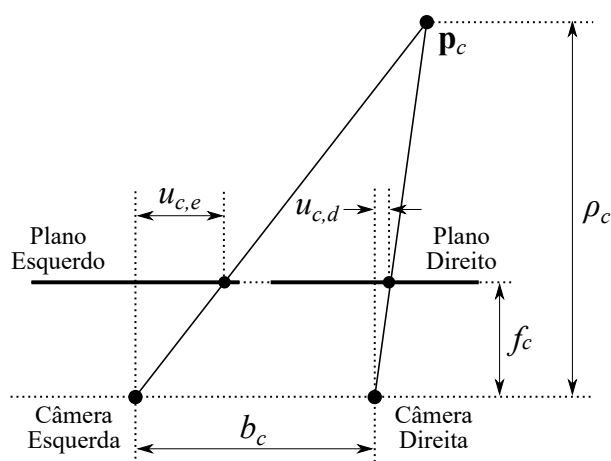


Figura 9 – Representação do processo de triangulação utilizado para a obtenção da profundidade para câmeras estéreo.

distantes, com pouca textura (como paredes lisas) ou iluminação inadequada (tanto pouca quanto em excesso) dificultam o casamento entre as imagens, reduzindo a precisão dos resultados.

O processo de estimação de profundidade em sensores **RGB-D** ocorre de maneira bem similar aos processos de triangulação utilizado em câmeras estéreo. A principal diferença consiste no projetor presente nas câmeras **RGB-D**, o qual é capaz de introduzir um padrão único de luz estruturada no ambiente, conforme ilustrado na Figura 10. Desse modo, são criadas características artificiais no ambiente, facilitando o cálculo da disparidade no processo de casamento entre as imagens (Zollhöfer, 2019). Além disso, a estimação de profundidade é realizada de uma maneira mais precisa comparando com o caso das câmeras estéreo, visto que esse método não é afetado caso haja pouca iluminação no ambiente ou pouca textura em uma superfície, como uma parede lisa, por exemplo. A iluminação projetada encontra-se geralmente no espectro infravermelho, onde o padrão utilizado depende de cada modelo, podendo ser formado por pontos, listras ou inclusive uma sequência temporal. Os maiores problemas enfrentados com esse tipo de câmera cor-

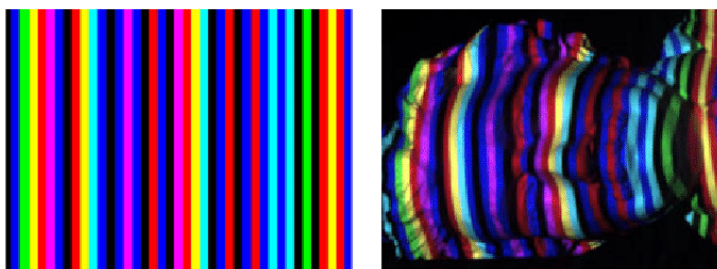


Figura 10 – Representação do processo de projeção de luz estruturada utilizando listras verticais com diferentes espectros (Bhowmik, 2017).

responde a locais muito distantes, com superfícies reflexivas ou com iluminação excessiva, os quais dificultam o processo de detecção dos padrões projetados. Além disso, esses sensores contam com uma lente RGB, capaz de capturar imagens coloridas do ambiente sem serem afetadas pela luz estruturada projetada em infravermelho.

3.4 SLAM Visual

Nas últimas décadas, tarefas de localização e mapeamento têm estado entre os assuntos mais abordados na robótica móvel. Há diversos estudos e aplicações envolvendo métodos de localização para diferentes tipos de robôs em diferentes cenários, além dos vários métodos de mapeamento capazes de gerar mapas de ambientes distintos.

O termo localização representa o processo de estimar os estados de um robô em relação a um referencial inercial do ambiente, também conhecido como posicionamento ou estimação de pose, representada pelas informações de posição e de orientação (Dudek and Jenkin, 2010). O processo de localização envolve dois casos: a estimação direta da pose, onde o dispositivo é capaz de se localizar utilizando marcações conhecidas no ambiente, e a estimação incremental, conhecida como odometria, onde os pequenos deslocamentos do robô são calculados e integrados ao longo do tempo. Por outro lado, o mapeamento consiste na construção de modelos do ambiente, sendo essencial para diversas tarefas associadas à robótica (Burgard et al., 2016). Dentre as possíveis formas de se representar o ambiente, é possível citar os mapas topológico e geométrico, podendo este último ser subdividido em três tipos: mapas de elevação 2,5D, onde cada ponto do terreno possui um valor de elevação; nuvens de pontos 3D, dadas por um conjunto de pontos tridimensionais; e malhas (*meshes*), compostas por superfícies geralmente em formas de triângulos. Em geral, a finalidade dos mapas envolve aplicações de planejamento e navegação, porém, em alguns casos a reconstrução do ambiente pode ser aplicada para inspeção visual humana.

É possível perceber que as tarefas de localização e mapeamento estão intrinsecamente relacionadas. Em muitos casos, enquanto navega pelo ambiente, o robô precisa ser capaz de estimar sua pose e mapear o local de forma simultânea, processo conhecido como Localização e Mapeamento Simultâneos (SLAM) (Durrant-Whyte and Bailey, 2006). Para realizar essa tarefa, o robô necessita diversos mecanismos capazes de obter informações geométricas do espaço a sua volta, onde sensores LiDAR e câmeras são os mais utilizados. Nesse contexto, métodos de fusão sensorial podem ainda ser aplicados com o objetivo de combinar dados entre dois ou mais sensores e a aprimorar os resultados. Além disso, os recentes avanços computacionais, principalmente relacionados aos processadores gráficos, permitiram a rápida manipulação de grande quantidade de dados, viabilizando a aplicação de métodos de SLAM de forma *online* em processos de larga escala.

De maneira geral, o SLAM é melhor definido como um processo probabilístico,

utilizando as variáveis da pose do robô \mathbf{x}_s , as entradas relativas ao movimento \mathbf{u}_s e as medições \mathbf{z}_s observadas no ambiente \mathcal{M}_s (Stachniss et al., 2016). A Figura 11 apresenta um diagrama do processo de SLAM, definido como a observação do ambiente local \mathcal{M}_s de forma a determinar a pose $\mathbf{x}_s[k]$ do robô, utilizando os valores de entrada $\mathbf{u}_s[k]$ e as medições $\mathbf{z}_s[k]$, para um dado instante k . Assim, a estimação pode ser dada na forma de probabilidade a posteriori, calculada por:

$$P_s(\mathbf{x}_s[k], \mathcal{M}_s \mid \mathbf{z}_s[k], \mathbf{u}_s[k]) . \quad (3.19)$$

Nesse caso, os valores de posição e as características do mapa não são conhecidas no SLAM e os métodos utilizados são frequentemente chamados de filtros, trabalhando com probabilidades e incertezas das medições.

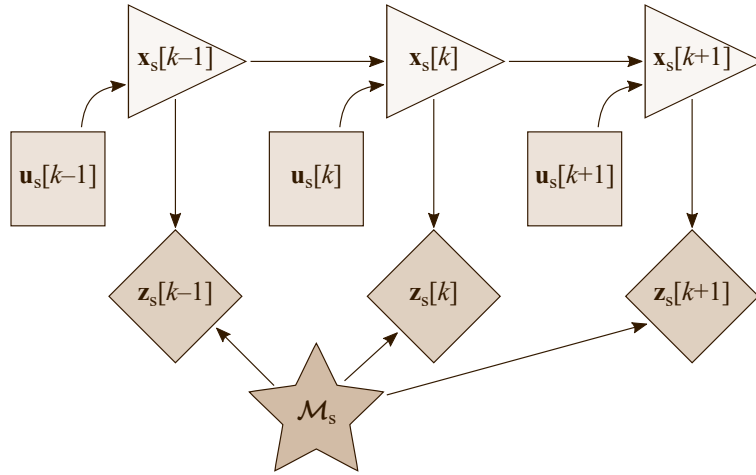


Figura 11 – Diagrama representativo das variáveis envolvidas no processo de SLAM, sendo \mathbf{x}_s o estado do robô modificado pelas entradas relativas ao movimento \mathbf{u}_s e as medições \mathbf{z}_s observadas no ambiente \mathcal{M}_s (Stachniss et al., 2016).

Um dos primeiros e mais importantes métodos utilizados para a solução do problema de SLAM é o EKF. Com esse método, a localização do robô e o conjunto de características do ambiente são definidos como vetores de estados, além de matriz de covariância representando as incertezas associadas e a correlação entre os estados. À medida que o robô se move e obtém características do ambiente, os estados do sistema e as covariâncias são atualizadas no EKF. Dessa forma, para um dado instante k tem-se:

$$P_s(\mathbf{x}_s[k], \mathcal{M}_s \mid \mathbf{z}_s[k], \mathbf{u}_s[k]) = \mathcal{N}(\mathbf{x}_f[k], \Sigma_f[k]) , \quad (3.20)$$

onde \mathcal{N} representa a distribuição Gaussiana da estimação da localização do robô com média \mathbf{x}_f e covariância Σ_f .

A Figura 12 ilustra o funcionamento do **EKF** aplicado ao **SLAM** e as incertezas envolvidas no processo. A linha tracejada representa o caminho do robô e as elipses cinzas a sua estimativa, enquanto que os oito pontos indicam algumas marcações (ou características) no mapa e as elipses brandas suas estimações. Nos primeiros movimentos do robô (a), as incertezas relacionadas à pose do robô e das marcações são baixas, dado que a posição inicial é conhecida. Conforme o robô se desloca (b-c), as incertezas vão aumentando até ser observada uma marcação já conhecida (d). Nesse ponto, a incerteza da pose atual do robô é reduzida, assim como às incertezas relacionadas a outras marcações do mapa.

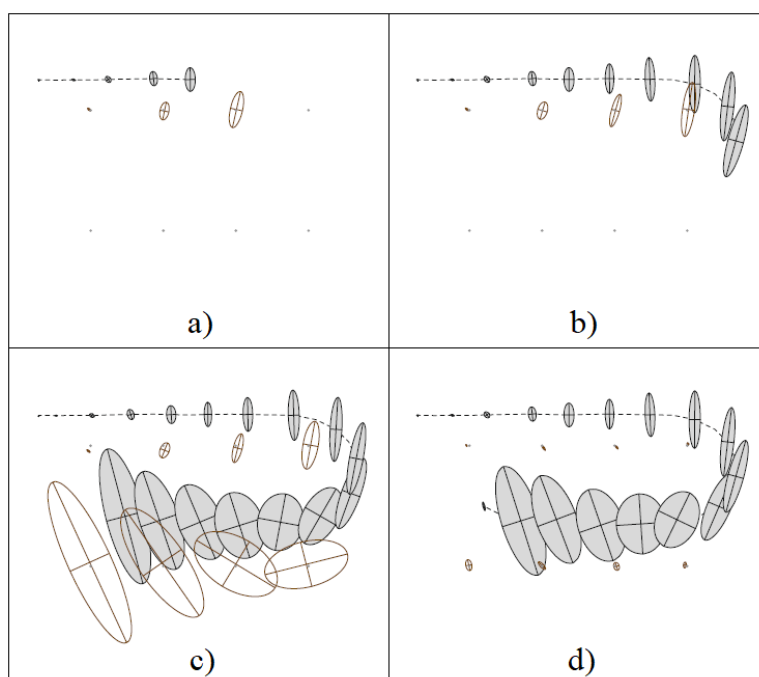


Figura 12 – Ilustração representativa das incertezas envolvidas no **EKF** aplicado no processo de **SLAM** (Stachniss et al., 2016).

Outra forma bastante comum em métodos mais recentes para a solução de problemas de **SLAM** consiste na representação baseada em grafos. Dessa forma, imaginando o grafo como um sistema massa-mola, a solução pode ser obtida de forma equivalente ao estado de mínima energia desse modelo. Considerando que o grafo corresponde ao log do problema, assumindo independência entre as observações de $\mathbf{z}_s[k]$ e $\mathbf{u}_s[k]$ e um ruído Gaussiano nas medições $\mathbf{z}_s[k]$, a probabilidade a posteriori pode ser expressa na forma quadrática:

$$P_s(\mathbf{x}_s[k], \mathcal{M}_s | \mathbf{z}_s[k], \mathbf{u}_s[k]) = c_s + \sum_k \bar{\mathbf{x}}_s[k]^T S_s[k]^{-1} \bar{\mathbf{x}}_s[k] + \sum_k \bar{\mathbf{z}}_s[k]^T Q_s[k]^{-1} \bar{\mathbf{z}}_s[k], \quad (3.21)$$

onde $\bar{\mathbf{x}}_s = \mathbf{x}_s[k] - g_s(\mathbf{x}_s[k-1], \mathbf{u}_s[k])$ e $\bar{\mathbf{z}}_s = \mathbf{z}_s[k] - h_s(\mathbf{x}_s[k], \mathcal{M}_s)$. As funções g_s e h_s são geralmente linearizadas de forma iterativa e a solução final da expressão pode ser obtida por métodos eficientes de otimização, como fatores de Cholesky, decomposição QR ou gradiente descendente.

Uma ideia básica do funcionamento de um método de SLAM baseado em grafos é apresentada na Figura 3.21. Em um primeiro instante $k = 1$, o robô se encontra em $\mathbf{x}_s[1]$ e observa uma característica $m_s[1]$ no mapa, indicado pela aresta 0. Nesse ponto, é marcado no grafo que $\mathbf{x}_s[1]$ está relacionado com $m_s[1]$ através de 0. Após a entrada dada pela aresta 1, o robô alcança a posição $\mathbf{x}_s[2]$ e essa relação é marcada no grafo. Nessa posição o robô observa $m_s[1]$, $m_s[2]$ e $m_s[3]$, através das respectivas arestas 2, 3 e 4, as quais são marcadas no grafo. Em seguida, o robô alcança a posição $\mathbf{x}_s[3]$ e visualiza o marcador $m_s[3]$, sendo marcado no grafo. O processo então continua seguindo esses passos até alcançar o último ponto. Com essa forma, o grafo é capaz de armazenar grande quantidade de informação sobre o ambiente, porém, possuindo uma estrutura relativamente simples e de fácil manipulação.

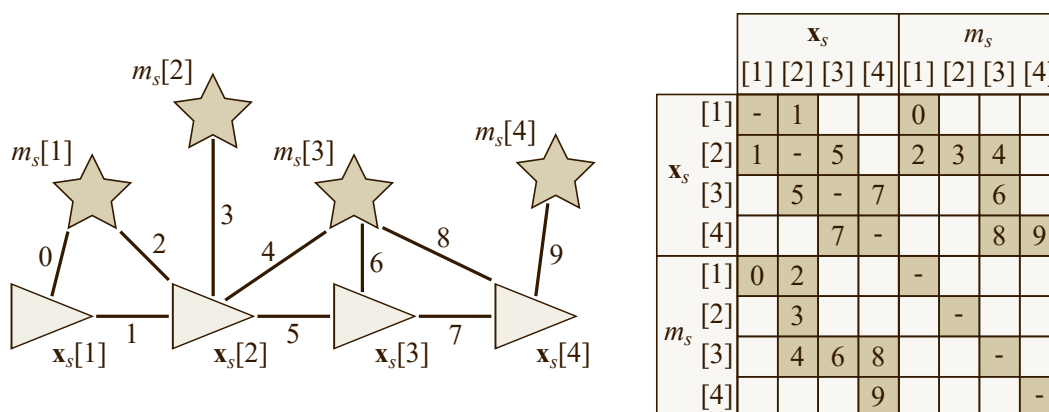


Figura 13 – Ilustração representativa funcionamento de um método de SLAM baseado em grafos (Stachniss et al., 2016).

Dentre as principais técnicas que envolvem localização e mapeamento, o SLAM Visual consiste na utilização de câmeras embarcadas para estimação da pose e reconstrução do ambiente. O uso desse tipo de sensor é bastante comum em aplicações de SLAM, pelo fato de serem dispositivos leves, passivos, que consomem pouca energia e adquirem muita informação do seu entorno. Além disso, o emprego de câmeras em tarefas relacionadas ao mapeamento é de grande importância por possibilitar a detecção da cor, da textura e do formato dos objetos, permitindo uma reconstrução densa e colorida do ambiente, sendo visivelmente mais fiel à visualização humana. Porém, o custo computacional e a susceptibilidade desses dispositivos a condições de iluminação e partículas do ambiente (como poeira ou neblina) são problemas a serem enfrentados, especialmente em ambientes confinados e não estruturados.

De forma mais específica, a Odometria Visual (**VO**) pode ser considerada como a etapa mais importante do processo de **SLAM** Visual, onde a pose do dispositivo é estimada incrementalmente utilizando imagens de uma ou mais câmeras embarcadas (**Scaramuzza and Fraundorfer, 2011**). Nesse caso, considerando duas imagens capturadas pela câmera em dois instantes consecutivos, o problema de **VO** consiste em achar uma transformação que relaciona as poses da câmera nesses instantes, a qual pode ser dada por uma matriz homogênea. Considerando os grupos Especial Euclidiano (**SE**) e Especial Ortogonal (**SO**), uma transformação homogênea $T \in \text{SE}(3)$ é composta por uma matriz de rotação $R \in \text{SO}(3)$ e um vetor de translação $\mathbf{t} \in \mathbb{R}^3$, na forma:

$$T = \begin{bmatrix} R & \mathbf{t} \\ 0 & 1 \end{bmatrix}. \quad (3.22)$$

É possível notar que T é uma matriz quadrada, possuindo um fator de homogeneização na última linha. Nesse caso, para desenvolvimento dos cálculos, um ponto qualquer $\mathbf{p} \in \mathbb{R}^3$ pode ser reescrito como $\bar{\mathbf{p}} = [\mathbf{p}, 1]^T \in \mathbb{R}^4$. Assim, supondo dois pontos $\bar{\mathbf{p}}[k]$ e $\bar{\mathbf{p}}[k+1]$, é possível determinar a rotação e a translação entre eles na forma:

$$\bar{\mathbf{p}}[k+1] = T \bar{\mathbf{p}}[k]. \quad (3.23)$$

Retornando ao processo de **VO**, pode-se considerar as poses consecutivas $\bar{\mathbf{p}}_s[k-1]$ e $\bar{\mathbf{p}}_s[k]$ da câmera, onde o problema consiste em determinar uma transformação $T_s[k] \in \text{SE}(3)$ que relaciona essas poses. Dessa forma, conhecendo a condição inicial $\bar{\mathbf{p}}_s[0]$, é possível determinar a pose $\bar{\mathbf{p}}_s$ em um instante k por meio de sucessivas transformações $\bar{\mathbf{p}}_s[k] = T_s[k]\bar{\mathbf{p}}_s[k-1]$, demonstrando que o processo é realizado de forma incremental. Além disso, um refinamento iterativo pode ser computado utilizando as últimas n imagens em um processo chamado *Windowed-Bundle Adjustment*, onde são executados métodos de minimização dos erros de projeção.

Existem três principais métodos para a estimação da transformação $T_s[k]$, variando com a dimensão utilizada para definir as características $m_s[k]$ (2D ou 3D). Para o caso *2D-to-2D*, o cálculo de $T_s[k]$ é realizado utilizando restrição epipolar e Decomposição de Valores Singulares (SVD) sobre o casamento entre $m_s[k-1]$ e $m_s[k]$, especificadas como coordenadas 2D da imagem. Já para o método *3D-to-3D*, a transformação obtida por um processo de minimização da distância entre $m_s[k-1]$ e $m_s[k]$, dadas como pontos tridimensionais para esse caso. Por fim, há também o método *3D-to-2D*, o qual utiliza minimização do erro entre a reprojeção de $m_s[k-1]$ e $m_s[k]$, um problema conhecido como *Perspective-n-Point* (**PnP**) (**Zheng et al., 2014**).

Capítulo 4

Metodologia

Conforme apresentado na Seção 1.2, o principal objetivo desta dissertação consiste na geração de reconstruções densas e coloridas de ambientes confinados para fins de inspeção por meio de modelos digitais e realidade virtual. Para tal fim, é proposta a análise e a comparação entre dois métodos distintos. O primeiro consiste em um mapeamento com registro da nuvem de pontos utilizando câmeras RGB-D, associado a uma localização obtida com a combinação de dados de odometria das rodas e de uma IMU. O segundo método corresponde a um processo de SLAM com o RTAB-Map, utilizando dados de câmeras RGB-D para a realização da VO, além de informações de IMU para o mapeamento.

Neste Capítulo são apresentados os procedimentos que envolvem a execução de cada método: EKF, registro de pontos e RTAB-Map; assim como as métricas empregadas para as avaliações dos resultados.

4.1 Localização com Filtro de Kalman Estendido

O primeiro método de localização utilizado nesta dissertação foi realizado com o Filtro de Kalman Estendido (EKF). Com esse método, é possível obter uma estimativa relativamente precisa da pose 6D do robô por meio de dados de odometria das rodas e de orientação de uma IMU. A Figura 14 ilustra o processo de localização com o EKF.

Primeiramente, é necessário sincronizar as informações de odometria das rodas e da IMU, que correspondem aos dados de entrada do processo. Para isso, foi utilizada uma estrutura para lidar com as informações obtidas com os sensores. Nesse caso, os valores das medições e suas respectivas covariâncias são adicionados na estrutura, além do sistema de coordenadas e do tempo de medição associados, visto que estes são diferentes para cada sensor. Assim, para obter uma informação em um tempo intermediário às medições, é possível acessar a estrutura e realizar uma interpolação dos valores armazenados, permitindo uma sincronização precisa entre os dados dos sensores.

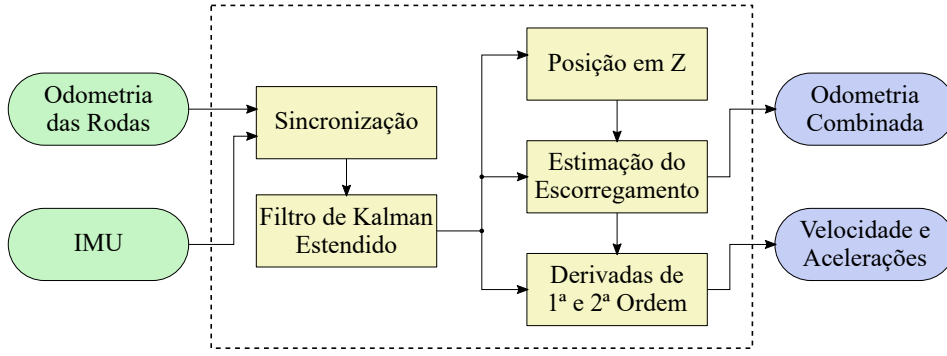


Figura 14 – Diagrama de blocos do processo de localização com o EKF.

As entradas do cálculo do EKF correspondem aos dados sincronizados de odometria das rodas e da IMU, ambos aplicados no processo de correção do filtro de acordo com a Equação 3.13. No processo de medição da odometria das rodas, foram coletados os dados da pose completa $\mathbf{y}_{f,o}[k] \in \mathbb{R}^6$ e sua respectiva covariância $\mathbf{W}_{f,o}[k] \in \mathbb{R}^{6 \times 6}$. Nesse caso, $\mathbf{H}_{f,o}[k] \in \mathbb{R}^{6 \times 6}$ possui valores não nulos somente nos locais referentes à pose planar. Para a IMU, foram utilizados os dados de orientação absoluta, os quais podem ser obtidos por meio de um filtro complementar com as informações do acelerômetro e do giroscópio. Dessa forma, foram obtidas a orientação completa $\mathbf{y}_{f,i}[k] \in \mathbb{R}^3$ e sua respectiva covariância $\mathbf{W}_{f,i}[k] \in \mathbb{R}^{3 \times 3}$. Além disso, $\mathbf{H}_{f,i}[k] \in \mathbb{R}^{3 \times 6}$ possui valores não nulos apenas nos locais relativos à orientação.

Na sequência, foram utilizadas as Equações 3.8 e 3.10 de forma a determinar uma predição para o estado do filtro. Nesse caso, a matriz $\mathbf{F}_f[k]$ foi utilizada como sendo uma matriz identidade e o vetor $\mathbf{u}_f[k]$ como um vetor nulo. Dessa forma, utilizando a predição e as duas etapas sucessivas de correção descritas anteriormente, pode ser obtida a estimação do estado $\hat{\mathbf{x}}_f[k+1] = [\hat{x}_f[k], \hat{y}_f[k], \hat{z}_f[k], \hat{\phi}_f[k], \hat{\theta}_f[k], \hat{\psi}_f[k]]^T \in \mathbb{R}^6$ e sua covariância $\Sigma_f[k] \in \mathbb{R}^{6 \times 6}$, em um dado instante k .

Os dados combinados da odometria das rodas e da IMU somente fornecem informações de posição planar, nos eixos x e y , e de orientação 3D, rolamento, arfagem e guinada. Logo, não é passada para o filtro nenhuma informação explícita da altura que o robô se encontra (posição em z), o que resulta em uma saída nula para esse estado. Levando em consideração o modelo utilizado pelo robô, esse valor pode ser calculado por meio das informações do ângulo de arfagem (em torno do eixo y) e de deslocamento linear. Utilizando os valores das estimações \hat{x}_f e \hat{y}_f nos instantes $k-1$ e k , esse deslocamento pode ser obtido por:

$$d_f = \sqrt{(\hat{x}_f[k] - \hat{x}_f[k-1])^2 + (\hat{y}_f[k] - \hat{y}_f[k-1])^2}. \quad (4.1)$$

Assim, dado o valor do ângulo de arfagem $\hat{\theta}_f[k-1]$ do robô, o vetor do deslocamento de posição $\mathbf{d}_f \in \mathbb{R}^3$ podem ser corrigidos com:

$$\mathbf{d}_f = \begin{bmatrix} (\hat{x}_f[k] - \hat{x}_f[k-1]) \cos(\hat{\theta}_f[k-1]) \\ (\hat{y}_f[k] - \hat{y}_f[k-1]) \cos(\hat{\theta}_f[k-1]) \\ -d_f \sin(\hat{\theta}_f[k-1]) \end{bmatrix}. \quad (4.2)$$

Um problema frequentemente observado na odometria de rodas está relacionado ao escorregamento ocasionado tanto por terrenos com baixo coeficiente de atrito, quanto pelo modelo *skid-steering* adotado para o EspeleoRobô. De forma a minimizar esse problema, foi adicionado ao processo de localização uma estimação do escorregamento, calculada por meio de um ganho relacionado tanto ao ângulo atual de arfagem do robô, quanto ao atrito do terreno. Para o primeiro caso, foi proposta uma função $r_f(\hat{\theta}_f)$, contínua e suave, que retorna um valor de ganho dependente da inclinação atual da subida (ou descida), calculada por:

$$r_f[k] = \begin{cases} \cos^2(\hat{\theta}_f[k-1]) & , \text{ caso } \hat{\theta}_f[k-1] \leq 0 \\ [\cos^2(\hat{\theta}_f[k-1])]^{-1} & , \text{ caso } \hat{\theta}_f[k-1] > 0 \end{cases}. \quad (4.3)$$

Para o atrito do terreno, foram utilizados dois parâmetros: b_f , correspondendo ao *bias* de escorregamento para o caso do robô estar em um local plano, e c_f , representando o coeficiente de atrito. O ganho final $s_f(\hat{\theta}_f)$ utilizado no método, ilustrado graficamente na Figura 15, é calculado na forma:

$$s_f[k] = [r_f[k] + b_f]^{c_f^{-1}-1}. \quad (4.4)$$

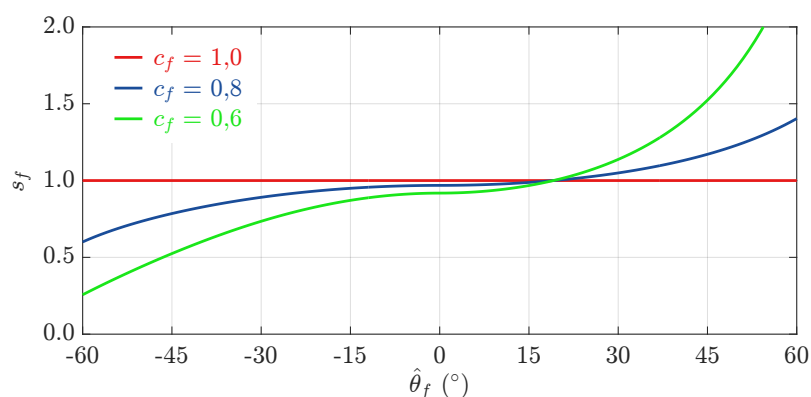


Figura 15 – Representação gráfica do ganho de escorregamento e atrito s_f em função do ângulo de arfagem, para b_f igual a -0,12 e c_f igual a 0,6, 0,8 e 1,0.

É importante enfatizar que os cálculos dos deslocamentos 3D e do ganho de escorregamento são realizados externamente ao **EKF**, conforme apresentado na Figura 14. Assim, os novos valores de posição $\bar{\mathbf{p}}_f[k] \in \mathbb{R}^3$ podem ser calculados de forma iterativa com os valores anteriores por meio de:

$$\bar{\mathbf{p}}_f[k] = \bar{\mathbf{p}}_f[k-1] + s_f[k]\mathbf{d}_f[k] . \quad (4.5)$$

Por fim, foram também calculados os valores de velocidade e acelerações com respeito às coordenadas inercial e corrente. Nesse caso, foram utilizados os valores de deslocamento das novas posições $\bar{\mathbf{p}}_f[k]$ e as orientações estimadas no filtro $\hat{\mathbf{o}}_f[k]$. Para isso, é necessário determinar o tempo decorrido desde a última atualização do filtro e realizar as derivadas discretas de primeira ordem para obter a velocidade e de segunda ordem para a aceleração.

4.2 Mapeamento com Registro da Nuvem de Pontos

O registro da nuvem de pontos, realizado por meio de dados de câmeras, consiste em gerar uma reconstrução de um ambiente utilizando de diversas nuvens locais determinadas a partir de imagens. Existem diversos algoritmos que realizam o alinhamento das nuvens locais, os quais geralmente demandam um custo computacional relativamente alto. De forma a minimizar o uso desses algoritmos, é possível utilizar dados já existentes da estimação de profundidade dos *pixels* e das transformações entre os sistemas de coordenadas de cada nuvem. Nesta dissertação, o processo utilizado para o registro da nuvem de pontos local corresponde a essa forma mais simples, que emprega dados externos de estimação de profundidade e de odometria, apresentando uma sequência de passos indicados pelo diagrama da Figura 16.

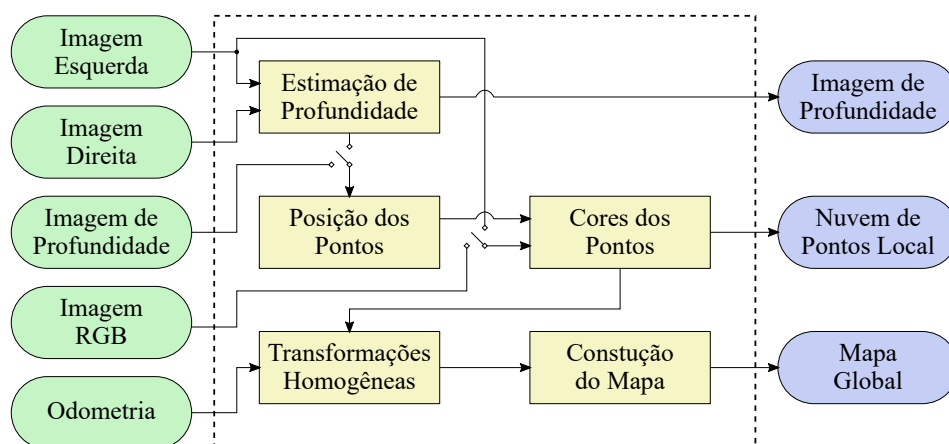


Figura 16 – Diagrama de blocos do processo de mapeamento do Registro de Pontos utilizando imagens e odometria externa.

Para a realização do registro de pontos, é necessária a informação da profundidade de cada *pixel* de forma a ser representado como um ponto tridimensional no mapa. Nesse caso, visando melhores resultados, são preferencialmente empregadas câmeras RGB-D. Porém, para o caso de câmeras estéreo, a profundidade pode ser obtida utilizando os parâmetros do sensor (b_c e f_c) e dos métodos de triangulação apresentados na Figura 9 e na Equação 3.18.

Dados os valores de profundidade ρ_c de uma imagem e a distância focal f_c da câmera, a posição tridimensional dos pontos da nuvem pode ser estimada utilizando novamente processos de triangulação. Considerando os sistemas de coordenadas padrão para dispositivos de imagens, indicado na Figura 17 (com x para baixo, y para a direita e z para frente), um *pixel* $\mathbf{r}_c = [u_c, v_c]^T$ pode ser representado como um ponto tridimensional $\mathbf{p}_c = [x_c, y_c, z_c]^T$ na forma:

$$x_c = \frac{u_c \rho_c}{f_c}, \quad y_c = \frac{v_c \rho_c}{f_c} \quad \text{e} \quad z_c = \rho_c. \quad (4.6)$$

Em seguida, para determinar a cor correspondente ao ponto \mathbf{p}_c , é necessário encontrar sua respectiva projeção na imagem RGB. Com as câmeras alinhadas no mesmo referencial e utilizando a distância focal do sensor RGB $f_{\bar{c}}$ e as coordenadas tridimensionais do ponto, a projeção $\mathbf{r}_{\bar{c}} = [u_{\bar{c}}, v_{\bar{c}}]^T$ pode ser obtida por:

$$u_{\bar{c}} = \frac{f_{\bar{c}} x_c}{z_c} \quad \text{e} \quad v_{\bar{c}} = \frac{f_{\bar{c}} y_c}{z_c}. \quad (4.7)$$

As posições dos pontos definidas anteriormente possuem como referência o sistema de coordenadas da câmera. Para realizar a construção do mapa, é necessário que essas posições estejam relacionadas a um referencial inercial, o qual pode ser alcançado por meio de sucessivas transformações homogêneas entre sistemas intermediários. Uma demonstração desses sistemas é apresentado pela Figura 17, onde estão indicadas as câmeras e os sistemas de coordenadas utilizados, de acordo com o padrão de cores RGB (eixo x em vermelho, y em verde e z em azul). Além disso, para desenvolvimento dos cálculos, é necessário adicionar um fator de homogeneização aos pontos, conforme apresentado na Equação 3.23, sendo reescritos na forma $\bar{\mathbf{p}}_c = [\bar{\mathbf{p}}_c, 1]^T \in \mathbb{R}^4$.

Nesse ponto, é possível considerar e uma transformação homogênea $T_C^R \in \text{SE}(3)$, que expressa a relação entre os sistemas de coordenadas da câmera (\mathcal{F}_C) e do robô (\mathcal{F}_R). Assim, um ponto do mapa $\bar{\mathbf{p}}_c^R$, representado em relação à câmera, pode ser expresso com respeito às coordenadas do robô por:

$$\bar{\mathbf{p}}_c^R = T_C^R \bar{\mathbf{p}}_c^C. \quad (4.8)$$

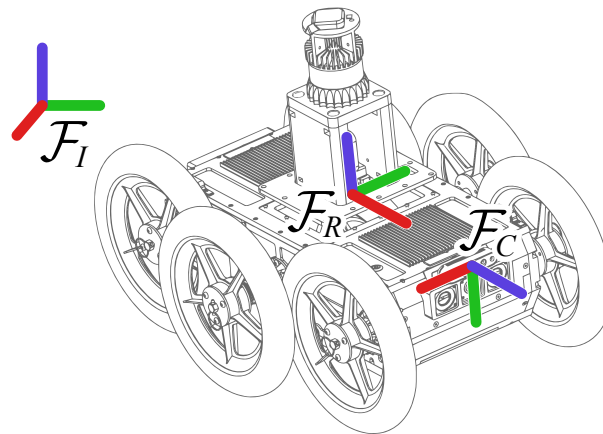


Figura 17 – Representação dos principais sistemas de coordenadas utilizados: inercial (\mathcal{F}_I), do robô (\mathcal{F}_R) e da câmera (\mathcal{F}_C).

Essa transformação T_C^R pode ser definida por valores predeterminados de construção do dispositivo, realizando medições físicas ou ainda por meio de métodos de estimação, utilizando pontos conhecidos no ambiente (Zheng et al., 2014).

Analisando agora a transformação $T_R^I \in \text{SE}(3)$, que relaciona os sistemas de coordenadas do robô (\mathcal{F}_R) com o sistema inercial (\mathcal{F}_I), o ponto $\bar{\mathbf{p}}_c^I$, pode então ser referenciado a este último sistema por:

$$\bar{\mathbf{p}}_c^I = T_R^I \bar{\mathbf{p}}_c^R = T_R^I T_C^R \bar{\mathbf{p}}_c^C . \quad (4.9)$$

Nesse caso, T_R^I pode ser obtida utilizando uma das odometrias computadas pelo robô.

Por fim, utilizando os pontos locais agora representados com respeito ao sistema de coordenadas inercial, juntamente com suas respectivas cores, é possível armazenar essas informações em memória, de forma a obter a reconstrução global do registro da nuvem de pontos.

4.3 SLAM com RTAB-Map

O *Real-Time Appearance-Based Mapping* (RTAB-Map) é um método de SLAM Visual baseado em grafos, atualmente disponível como uma biblioteca em C++ de código aberto (Labbé and Michaud, 2019). O algoritmo foi desenvolvido pelo Laboratório Interdisciplinar de robótica da *University of Sherbrooke*, no Canadá, sendo inicialmente proposto como um método de detecção de fechamento de laço para grandes ambientes utilizando dados visuais (Labbe and Michaud, 2013). O RTAB-Map permite o uso de imagens provenientes de câmeras estéreo ou RGB-D como informações principais de entrada. Além disso, é possível utilizar informações adicionais de IMU, de sensores LiDAR e de

odometria externa, de forma a complementar os processos de localização e o mapeamento. Dessa forma, o método é capaz de alcançar de maneira *online* uma localização precisa e uma reconstrução otimizada e relativamente densa, em forma de nuvem de pontos.

A localização e o mapeamento são executados por dois processos distintos, o que permite a utilização de outros tipos de odometria de acordo com a necessidade do usuário. Para a parte de localização, o **RTAB-Map** possui a implementação de duas abordagens, com os nomes Frame-To-Map (**F2M**) e Frame-To-Frame (**F2F**). No primeiro, o registro das transformações é processado no *frame* atual com base no mapa de características, enquanto que no segundo, o registro é realizado com base no último *keyframe*. Um diagrama com a sequência de passos que envolvem o processo de localização do **RTAB-Map** é apresentado na Figura 18.

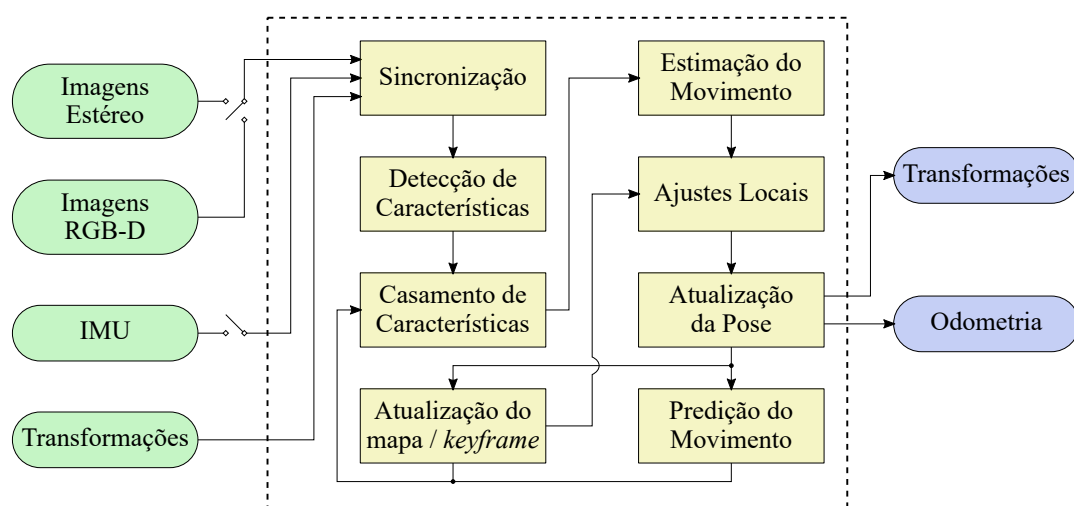


Figura 18 – Diagrama de blocos do processo de localização visual com o **RTAB-Map** (Labbé and Michaud, 2019).

Durante o processo de localização, as mensagens de entrada são inicialmente sincronizadas para, então, ser realizado o processo de detecção de características. O **RTAB-Map** aceita uma ampla gama de métodos para a realização desse processo, sendo utilizado como padrão extrator de características *Good Features to Track* (GFTT) (Jianbo Shi and Tomasi, 1994) devido ao seu bom desempenho, uniformidade nos resultados e facilidade de manuseio.

Nesse ponto do processo, um modelo de movimento é utilizado de forma a obter uma predição de onde deveriam estar as características de acordo com as transformações computadas nos tempos anteriores. É então realizado o casamento entre as características extraídas, o qual depende da abordagem utilizada. Para o caso da **F2M**, o casamento é processado com a busca do vizinho mais próximo em um dado raio de vizinhança, utilizando o descritor de características *Binary Robust Independent Elementary Features*

(BRIEF) (Calonder et al., 2010). Já para a abordagem F2F, o método de *optical flow* é utilizado diretamente nas características extraídas. Nesse contexto, caso o movimento computado com o *optical flow* seja muito diferente do movimento previsto, é utilizado o descritor BRIEF para o cálculo do casamento dos pontos.

A estimação do movimento é então computada utilizando uma implementação do *Random Sample Consensus* (RANSAC) em OpenCV (Bradski and Kaehler, 2008) para a resolução do problema de *Perspective-n-Point* (PnP). Em seguida, a transformação estimada é aprimorada por ajustes locais do mapa de características (F2M) ou do último *keyframe* (F2F). A pose final é então atualizada, assim como sua respectiva matriz covariância e transformações. Por fim, é realizado uma atualização do mapa (ou do *keyframe*), que é utilizado pelos processos de casamento de características e de ajustes locais.

Tratando agora do processo de mapeamento, a estrutura do mapa gerado pelo RTAB-Map consiste em um grafo, contendo vértices e arestas. Um vértice contém todas as informações locais necessárias para descrever um determinado ponto, incluindo: posição e orientação da odometria; dados dos sensores; palavras visuais (*visual words*), utilizadas no fechamento de laço e na detecção de proximidade; e *grid* de ocupação local, para a construção do mapa global. Por outro lado, as arestas correspondem a uma transformação rígida entre dois vértices e podem ser de três tipos: de vizinhança, de fechamento de laço ou de proximidade.

O gerenciamento de memória corresponde a um dos principais tópicos abordados pelo RTAB-Map na parte de mapeamento (Labbé and Michaud, 2018). Esse processo é tomado como prioridade pelo método, dividindo o armazenamento em três diferentes categorias: Memória de Trabalho (WM), Memória de Termo Curto (STM) e Memória de Termo Longo (LTM). A STM armazena os dados referentes ao vértice atual, apresentados no parágrafo anterior, e é responsável por gerar as arestas de vizinhança para conectar vértices consecutivos de odometria. Por sua vez, a WM abrange todo o processo corrente, armazenando todos os vértices e arestas necessárias para o mapeamento. A LTM é utilizada quando é ultrapassado um limiar (de memória ou de tempo) na WM. Nesse caso, os vértices mais antigos do grafo que possuem menor peso (computados com as palavras visuais da STM) são transferidas para a LTM, permanecendo inacessíveis pela WM. Dessa forma, o tamanho total do grafo de trabalho permanece limitado, o que auxilia na redução no tempo de execução de processo que envolvem a manipulação do grafo, possibilitando a utilização do RTAB-Map em ambientes consideravelmente grandes.

Conforme apresentado anteriormente, o processo de mapeamento do RTAB-Map é realizado de forma separada à localização. Desse modo, utilizando os dados de odometria e das transformações relacionadas, esse processo envolve a sequência de passos ilustrada pelo diagrama da Figura 19.

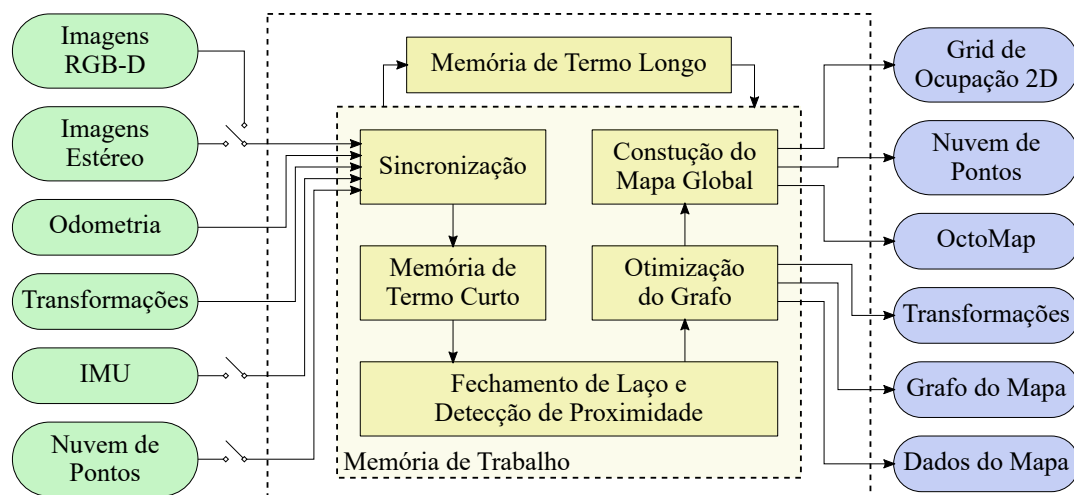


Figura 19 – Diagrama de blocos do processo de mapeamento com o [RTAB-Map](#) (Labbé and Michaud, 2019).

Dados de entrada fornecidos pelos sensores geralmente não são obtidas no mesmo tempo além de não possuírem a mesma frequência. Assim, é inicialmente realizada uma sincronização das mensagens, a qual é exata caso os dados dos sensores sejam fornecidos no mesmo tempo, ou aproximada caso contrário. Utilizando esses dados de entrada, a [STM](#) é então utilizada de forma a computar e armazenar as informações do vértice local.

A detecção de fechamento de laço e de proximidade é então processada utilizando o método saco-de-palavras (*bag-of-words*). Nesse ponto, é realizada uma comparação entre as palavras do vértice atual com aquelas armazenadas na [WM](#), utilizando um filtro de Bayes para determinar a probabilidade do ponto atual ser um local já visitado. Caso essa probabilidade seja maior que um dado limiar, o fechamento do laço é concretizado, sendo então computadas as transformadas associadas. Entretanto, para o caso do dispositivo estar em um local conhecido, mas no sentido contrário (retornando em um corredor, por exemplo), o fechamento de laço não pode ser aplicado caso forem utilizadas informações exclusivamente de imagens. Nesse caso, a detecção de proximidade permite contornar esse problema, utilizando dados de sensores [LiDAR](#), onde os vértices atuais criados são ligados aos vértices detectados como próximos.

Quando há algum gerenciamento de memória, fechamento de laço ou detecção de proximidade, o [RTAB-Map](#) realiza um cálculo do erro associado às possíveis alterações no grafo. Caso o erro esteja dentro de um limiar, a otimização é computada e os novos vértices e arestas são adicionadas ao grafo. Nesse processo, o [RTAB-Map](#) permite técnicas distintas de otimização, porém, adota como padrão uma implementação da técnica g2o (Kummerle et al., 2011), utilizando algoritmos de *Levenberg-Marquardt*. Por fim, os *grids* de ocupação local, criados e armazenados na [STM](#), são adicionados ao *grid* global, utilizando as poses e transformações otimizadas, de forma a gerar um mapa global do ambiente.

Capítulo 5

Arcabouço Experimental

Neste Capítulo, são apresentados os tópicos relativos aos experimentos, sendo tratadas as implementações utilizadas, o dispositivo robótico e os sensores empregados nas simulações e nos experimentos reais, além dos ambientes escolhidos para a geração de resultados.

5.1 Pacotes de ROS Utilizados

De forma a facilitar e padronizar a comunicação entre os sensores e as plataformas robóticas, foi utilizado o *Robot Operating System* (ROS). Esse sistema permite trabalhar com o conceito de nós e tópicos, que simplificam o processo de comunicação entre os códigos e os dados, e ainda possui grande integração com diversos algoritmos e sensores utilizados no âmbito da robótica. Além disso, o ROS permite manipular os chamados *bags*, uma ferramenta bastante útil na gravação de experimentos. Com eles, é possível reproduzir dados de sensores e estados do robô da mesma maneira em que foram gravados, permitindo que o experimento seja repetido de forma fiel em qualquer momento. Os experimentos apresentados nesta dissertação foram gravados em *bags*, os quais foram reproduzidos a fim de gerar resultados utilizando as implementações apresentadas a seguir.

5.1.1 Implementação do EKF

Para realizar a fusão entre a odometria das rodas e os dados da IMU, foi utilizado o pacote de ROS `espeleo_pose_ekf`¹. Esse pacote consiste em uma versão modificada do `robot_pose_ekf`² (v1.14.5 2019-04-04), o qual é bastante comum em aplicações de robótica móvel. É importante ressaltar que os passos relacionados ao filtro original do pacote não foram alterados. Nesse ponto, o algoritmo utiliza uma Biblioteca de Filtro Bayesiano (BFL) para criação e manipulação de variáveis, além dos cálculos de predição e

¹ https://github.com/ITVRoC/espeleo_pose_ekf

² https://github.com/ros-planning/navigation/tree/kinetic-devel/robot_pose_ekf

correção dos estados do filtro. As principais modificações realizadas nesse pacote, citadas na Seção 4.1, estão relacionadas aos cálculos da posição em z , do ganho relativo ao atrito e escorregamento das rodas e das velocidades e acelerações inerciais e correntes. Além disso, o `espeleo_pose_ekf` inclui também implementações de outro pacote bastante comum chamado `imu_complementary_filter`³ (v1.1.8 2020-05-26), utilizado nos casos em que não há informações de orientação no tópico da `IMU`.

A Figura 20 ilustra um diagrama representativo dos tópicos e nós envolvidos no processo. Os dois principais dados de entrada do pacote são: `/odom` (mensagem de odometria das rodas do tipo `nav_msgs/Odometry`) e `/imu/data` (mensagem da `IMU` do tipo `sensor_msgs/Imu`). Inicialmente, os dados da `IMU` são republicados pelo nó `topic_repub`, de forma a serem incluídos na árvore de transformações do `espeleo_pose_ekf`. Em seguida, utilizando esse novo tópico como referência, o nó `ekf_broadcaster` publica as transformações entre os sistemas de coordenadas da odometria, da base do robô e da `IMU`. Os tópicos da odometria e da `IMU` e as transformações são subscritos pelo nó principal do `espeleo_pose_ekf`. As saídas desse nó correspondem ao resultado final do filtro (`/ekf/odom`), do tipo `nav_msgs/Odometry`, e as mensagens de velocidades e acelerações inerciais e correntes do tipo `geometry_msgs/TwistStamped`.

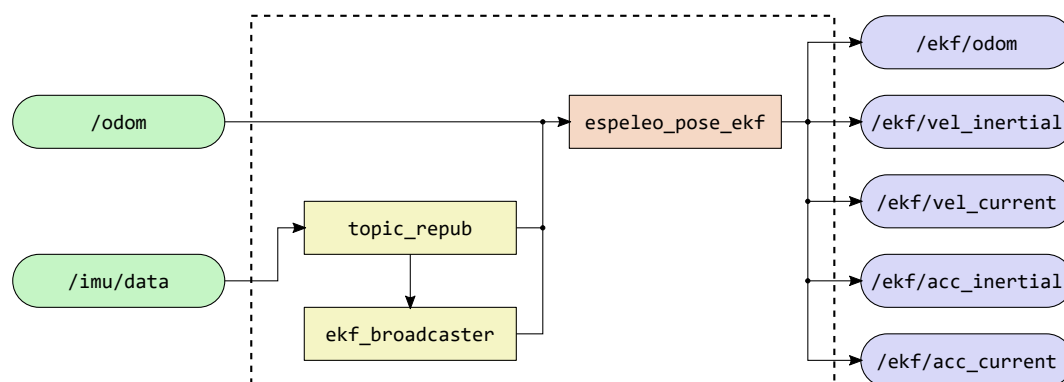


Figura 20 – Diagrama representativo dos principais tópicos e nós envolvidos no processo do `espeleo_pose_ekf`.

As matrizes de covariância relacionadas aos dados da odometria das rodas e da `IMU` devem ser passadas juntamente com cada mensagem. Em ambos os casos, foram utilizadas os valores já fornecidos pelo cálculo da odometria das rodas e pelo próprio dispositivo da `IMU`. Para o ganho relacionado ao atrito e escorregamento das rodas, apresentados pelas Equações 4.3 e 4.4, os valores dos parâmetros foram determinados após sucessivos testes em simulação. Foi assim estabelecido um valor de b_f igual a -0,12, enquanto que c_f foi deixado como parâmetro para o usuário (com o nome `friction_coeff` e valor padrão de 1,0), visto que este depende das condições de cada ambiente.

³ https://github.com/ccny-ros-pkg/imu_tools/tree/kinetic/imu_complementary_filter

É importante mencionar que o pacote também permite a utilização de dados de odometria visual e **GPS**. Contudo, essa informação não foi mencionada anteriormente já que esses dados não foram utilizados nos experimentos. Outra informação relevante corresponde à utilização de **IMU** que não fornece dados de orientação diretamente. Nesse caso, o nó `topic_repub` é substituído pelo `complementary_filter_node`, o qual subscreve os dados de acelerômetro e giroscópio do dispositivo de forma separada (`/accel/data` e `/gyro/data`) para gerar a orientação absoluta (`/imu/data`).

5.1.2 Implementação do Registro de Pontos

Os algoritmos utilizados pelo registro da nuvem de pontos foram implementados na linguagem Python, em uma versão compatível com a 2.7. O pacote de ROS, contendo toda parte de implementações desse método, está disponível em um repositório *online* com o nome `pcl_regristration`⁴. Para fins de compatibilidade, é importante mencionar que foram utilizadas as bibliotecas NumPy (1.11.0) e OpenCV (3.3.1-dev), para manipulação de vetores, matrizes e imagens.

A Figura 21 apresenta um diagrama com os tópicos e nós envolvidos no processo. Os tópicos de entrada do pacote são as informações e os dados das imagens colorida e de profundidade, sendo do tipo `sensor_msgs/CameraInfo` e `sensor_msgs/Image`, além da odometria do robô do tipo `nav_msgs/Odometry`. O primeiro nó `depth_to_cloud` subscreve as imagens coloridas e de profundidade e algumas de suas informações (como a distância focal de cada sensor) e publica uma nuvem de pontos local no tópico `/cloud`. Em seguida, o nó `cloud_to_map` sincroniza e combina os dados dessa nuvem local e das informações da odometria, de forma a montar e publicar um mapa global do ambiente. Por fim, o nó `map_to_file` é executado após o término do experimento de forma a salvar esse mapa em um arquivo PCD.

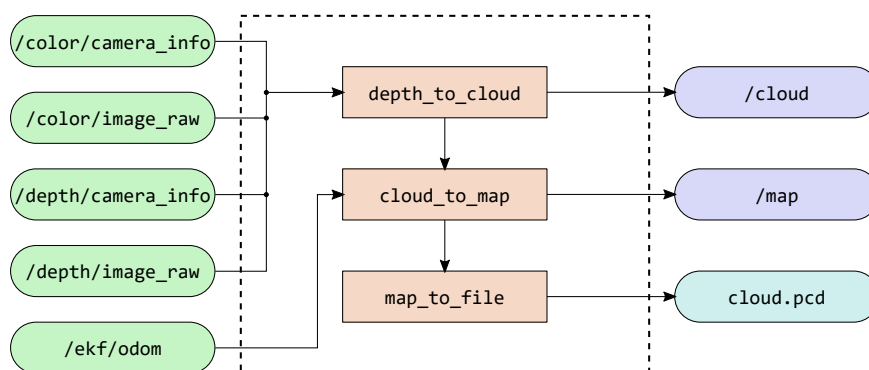


Figura 21 – Diagrama representativo dos principais tópicos e nós envolvidos no processo do `pcl_regristration`.

⁴ https://github.com/rafaelfgs/pcl_regristration

O pacote tem um total de seis parâmetros que podem ser alterados pelo usuário de acordo com a necessidade de cada experimento. O tamanho da imagem, ou `image_size`, varia com o tipo de ambiente e possui um valor padrão de 240p. Os alcances mínimo e máximo, ou `min_range` e `max_range`, também variam com o tipo do ambiente e possuem como padrão os respectivos valores de 1m e 6m. Para a frequência de publicação da nuvem, ou `cloud_freq`, foi utilizado um valor igual a 0,6Hz em todos os ambientes, enquanto que a frequência do mapa, ou `map_freq`, foi igual a 0,4Hz. Por fim, a velocidade do *bag*, ou `bag_rate`, foi igual a 1,0 em todos os experimentos.

É importante citar que, para o caso de se utilizar uma câmera estéreo, o processo para a retificação e cálculo da profundidade das imagens foram realizados com a biblioteca OpenCV, utilizando uma implementação em Python baseada no algoritmo `t265_stereo.py`⁵. Como esta dissertação não apresenta resultados com esse tipo de câmera, o algoritmo correspondente não foi detalhado.

5.1.3 Implementação do RTAB-Map

O **RTAB-Map** está atualmente disponível como uma biblioteca independente *open-source* em C++. Para a aplicação desse método, foi utilizado o pacote `rtabmap`⁶ e seu complemento para comunicação com o ROS `rtabmap_ros`⁷ (na versão Kinetic). Nesse pacote, não foram realizadas modificações internas na implementação, somente no acréscimo de alguns nós e na alteração dos valores de alguns parâmetros.

A Figura 20 apresenta um diagrama simplificado dos tópicos e nós do **RTAB-Map**. Os tópicos de entrada correspondem às informações e os dados das imagens colorida e de profundidade, sendo do tipo `sensor_msgs/CameraInfo` ou `sensor_msgs/Image`, além dos dados da IMU do tipo `sensor_msgs/Imu`. Assim como no `espeleo_pose_ekf`, foram adicionados dois nós iniciais: o `topic_repub` para a republicação dos tópicos, de forma a serem incluídos na árvore de transformações do `rtabmap_ros`, e o `rtab_broadcaster` para a publicação das transformações dos sistemas de coordenadas das imagens e da IMU. O primeiro nó específico do **RTAB-Map** é o `rgbd_odometry`, que utiliza as imagens e suas informações para realizar o processo de odometria visual, publicada no tópico `/rtabmap/odom`. O nó principal para o mapeamento é o `rtabmap`, o qual realiza processos de otimização de grafos para computar e publicar um caminho otimizado e o mapa global. Por último, o nó `rtabmapviz` é utilizado para a visualização do processo de **SLAM** e, ao final, processar e exportar a reconstrução em um arquivo PCD.

Dentre os vários parâmetros que o **RTAB-Map** possui, seis foram modificados visando obter melhores resultados com os experimentos, sendo:

⁵ https://github.com/IntelRealSense/librealsense/blob/master/wrappers/python/examples/t265_stereo.py

⁶ <https://github.com/introlab/rtabmap>

⁷ https://github.com/introlab/rtabmap_ros/tree/kinetic-devel

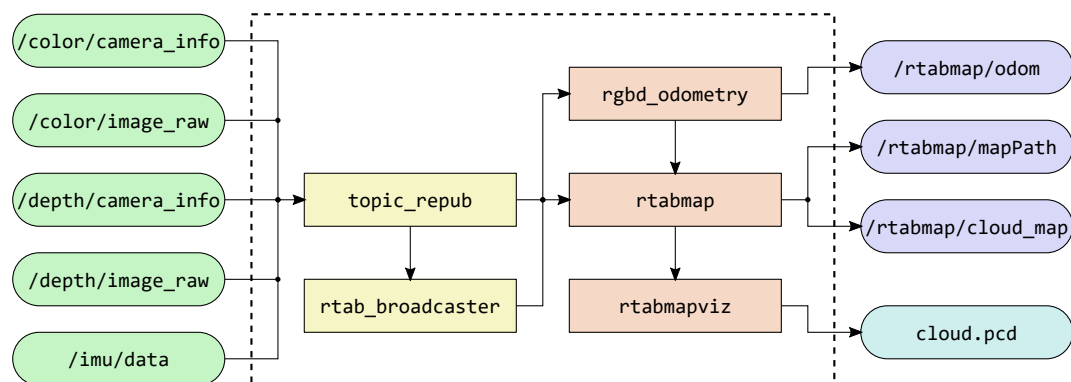


Figura 22 – Digrama representativo dos principais tópicos e nós envolvidos no processo do rtabmap_ros.

- estratégia de odometria *Frame-to-Frame* (`Odom/Strategy=1`);
- utilização de um robô com restrições holonômicas (`Odom/Holonomic=false`);
- distância mínima de 11 *pixels* entre as características (`GFTT/MinDistance=11`);
- erro máximo de 6,0 permitido no grafo após otimização de um fechamento de laço ou detecção de proximidade (`RGBD/OptimizeMaxError=6.0`);
- reinicialização da odometria quando se perder (`Odom/ResetCountdown=1`); e
- restrição da criação de um novo mapa quando a odometria for reiniciada, sem a espera por um fechamento de laço (`Rtabmap/StartNewMapOnLoopClosure=true`).

Outro parâmetro, dessa vez mais geral, que também foi modificado é o de sincronização dos tópicos de entrada (com o nome de `approx_sync`). Este teve de ser configurado como verdadeiro pelo fato das imagens da câmera não possuírem exatamente o mesmo tempo de publicação, sendo então realizado um processo de sincronização aproximada entre os tempos das mensagens.

5.2 Experimentos em Ambientes Simulados

O uso de simuladores representa um processo importante e às vezes necessário em diversas áreas da tecnologia. Com eles, é possível criar cenários específicos e realizar testes de forma controlada e segura. Dessa forma, é possível observar o funcionamento de uma técnica sem a necessidade de se manipular um dispositivo real em seu ambiente físico de trabalho, evitando assim possíveis perdas e danos. Alguns simuladores mais completos permitem algumas interações bem específicas e importantes em experimentos reais, como peso de um objeto, atrito ou o esforço com que ele é submetido, entre outras características de objetos estáticos e dinâmicos.

No âmbito da robótica, os simuladores estão relacionados ao desenvolvimento e utilização de modelos de um robô e da interação de seus sensores com um ambiente virtual. Para a realização das simulações com o EspeleoRobô, é utilizado o CoppeliaSIM, um simulador de robótica bastante versátil com grande integração com o ROS. Nesse simulador, é possível utilizar diversos tipos de sensores, como de visão, de medição inercial, entre outros, além de possibilitar a interação entre os objetos dinâmicos e estáticos presentes no ambiente. Mais especificamente para a parte de localização, é possível determinar o valor exato da posição e orientação atual do robô, permitindo analisar e comparar o desempenho de diferentes métodos.

5.2.1 Modelo Simulado do EspeleoRobô

O modelo simulado do EspeleoRobô, apresentado na Figura 23, foi desenvolvido utilizando processos computacionais como *Computer Aided Engineering* (CAE) e *Computer Aided Design* (CAD) (Cid et al., 2020). Para a construção e o desenvolvimento de peças mecânicas, optou-se pela utilização do *SolidWorks* devido à quantidade de recursos disponíveis em modelagem e simulação. Além disso, possui um banco de dados de materiais e suas propriedades, permitindo a simulação das características visuais e mecânicas dos elementos. Os componentes finais foram importados para o ambiente de simulação do CoppeliaSIM utilizando arquivos STL e URDF.

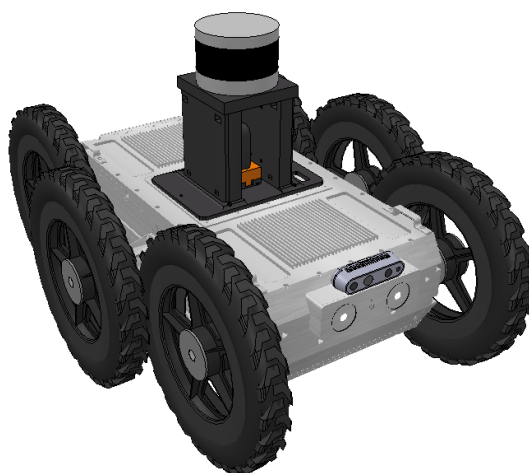


Figura 23 – Modelo do EspeleoRobô no simulador.

Além das peças e componentes mecânicos, o próprio simulador disponibiliza o uso de diversos sensores e equipamentos genéricos para aplicações de robótica. Alguns desses componentes foram utilizados e adaptados de forma a representar mais precisamente os sensores reais, dentre os quais pode-se citar: os motores, as câmeras internas, o sistema de iluminação, a IMU, o sensor LiDAR e a câmera RGB-D.

5.2.2 Ambientes Simulados

Para a realização dos experimentos em ambiente simulado, foram utilizados dois cenários disponíveis em um repositório⁸ do DARPA Subterranean (SubT) Challenge. É importante mencionar que, para serem importados, os arquivos tiveram que ser convertidos para um formato compatível com o CoppeliaSIM. Os modelos utilizados correspondem a representações similares aos ambientes típicos de trabalho do EspeleoRobô, como cavernas naturais e túneis subterrâneos construídos por humanos. Conforme apresentado na Seção 1.1, esse desafio envolve desenvolver novas tecnologias voltadas para tarefas de mapeamento, navegação, inspeção e exploração de espaços subterrâneos complexos.

Para a comparação dos resultados obtidos pelos métodos nas simulações, foram utilizados como referência as informações do próprio simulador, sendo estas chamadas de *ground truth*. Nesse caso, a localização foi obtida diretamente do simulador durante os experimentos e, para o mapeamento, foram utilizados os mapas das cavernas do simulador exportados como *meshes*.

• Simulação na Caverna do DARPA

O primeiro ambiente simulado utilizado para a realização de experimentos foi a `simple_cave_02`⁹, ilustrada na Figura 24. Esse ambiente corresponde a um cenário com geometria e texturas condizentes com uma caverna, além de possuir obstáculos como pedras e locais inacessíveis. O cenário conta com um terreno bastante irregular, com inclinações laterais variando até 10°, além de pequenas subidas e descidas com cerca de 20° e uma grande subida de 25 metros de comprimento com 23° de inclinação.

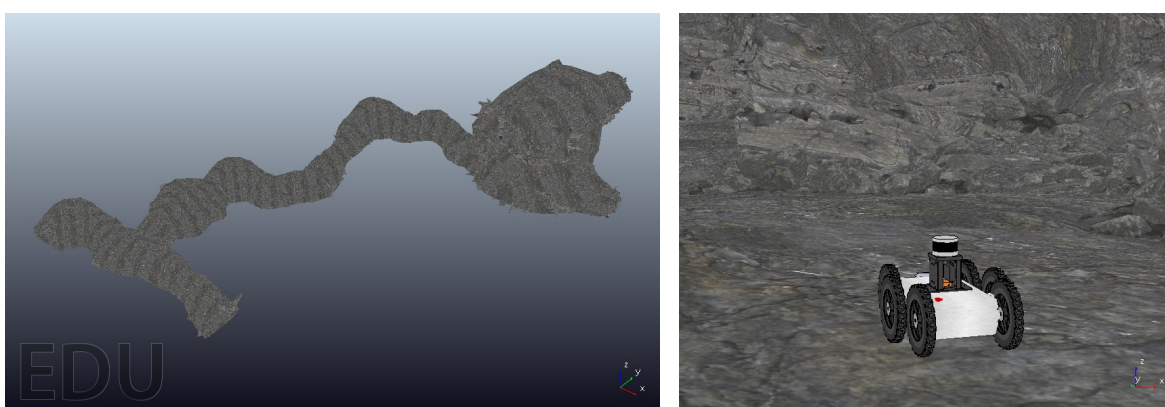


Figura 24 – Vistas externa e interna da Caverna do DARPA obtidas no simulador.

O experimento na caverna foi realizado no dia 19 de novembro de 2020, onde os dados dos diversos sensores presentes no modelo do robô foram gravados em *bags*, de forma a serem reproduzidos posteriormente para a geração dos resultados. Nesse experi-

⁸ <https://www.subtchallenge.world/models>

⁹ https://github.com/osrf/subt/blob/master/subt_ign/worlds/simple_cave_02.sdf

mento, o robô foi teleoperado por toda a caverna apresentada, realizando uma trajetória relativamente complexa, composta predominantemente por curvas, incluindo 5 locais com fechamento de laço. O tempo total do experimento foi de 15:03 (ou 903 segundos), com uma velocidade média de 0,5m/s e uma distância percorrida de 440,95 metros.

Para a geração dos resultados relativos a esse experimento foram utilizados os seguintes parâmetros: `friction_coeff = 0,64`; `image_size = 180p`; `min_range = 2m`; e `max_range = 10m`.

• Simulação na Mina do DARPA

O segundo ambiente utilizado nas simulações foi a `virtual_stix`¹⁰, apresentado na Figura 25. Esse ambiente corresponde a um cenário similar a uma mina, com geometria e texturas condizentes, além de alguns objetos como trilhos e um sistema de cabos e de iluminação. O terreno apresentado pelo cenário é um pouco irregular, com inclinações máximas de 10°, porém, sem locais de subidas e descidas consideráveis.

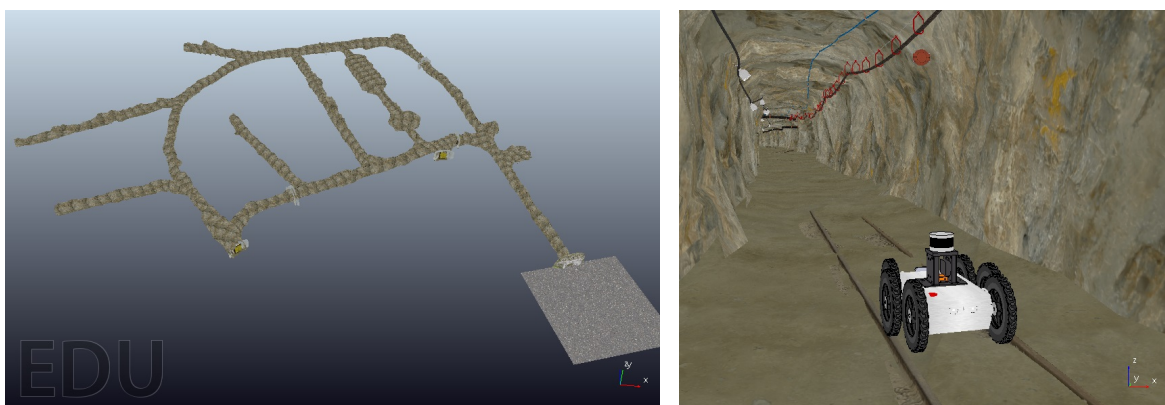


Figura 25 – Vistas externa e interna da Mina do DARPA obtidas no simulador.

O experimento na mina foi realizado no dia 26 de fevereiro de 2021, onde foram utilizados *bags* para salvar os dados dos sensores presentes no modelo do robô, de forma a serem posteriormente reproduzidos para a geração dos resultados. Nesse experimento, o robô foi novamente teleoperado em apenas parte da mina apresentada com uma trajetória bem mais simples que a anterior, movimentando-se ao longo de três retas de cerca de 50 metros cada e realizando duas curvas fechadas entre elas com mais de 90°. O tempo total do experimento foi de 5:44 (ou 344 segundos), com o robô a uma velocidade média de 0,5m/s e percorrendo uma distância de 144,18 metros.

Para a geração dos resultados relativos a esse experimento foram utilizados os seguintes parâmetros: `friction_coeff = 0,82`; `image_size = 240p`; `min_range = 2m`; e `max_range = 8m`.

¹⁰ https://github.com/osrf/subt/blob/master/subt_ign/worlds/virtual_stix.sdf

5.3 Experimentos em Ambientes Reais

A utilização de um robô físico em um ambiente real apresenta grandes desafios. Diferentemente das simulações, esses locais apresentam situações não controláveis, como condições adversas do ambiente e do terreno, falha ou mau funcionamento de sensores e atuadores, presença de incertezas nos dados e outros fatores que podem afetar a qualidade dos resultados. Contudo, os experimentos em ambientes reais são a melhor forma de analisar os resultados de um método, lidando com condições adversas.

5.3.1 EspeleoRobô

A Figura 26 apresenta uma foto do EspeleoRobô, o qual já foi apresentado na Subseção 1.1.2. Dentre os sensores presentes no robô, os principais utilizados nesta dissertação são os *encoders*, a *IMU* e a câmera da Intel.



Figura 26 – Imagem real do EspeleoRobô.

Tratando mais especificamente dos sensores, os *encoders* utilizados no processo de odometria das rodas corresponde a um dos componentes eletrônicos presentes nos motores Maxon MCD EPOS de 60W, localizados no encapsulamento do robô. Eles possuem 3 canais (A, B e I) e uma resolução de 1000 pulsos por volta, possibilitando uma resolução de posição de 4000 contagens de quadratura.

A *IMU* Xsens MTi-G-710 GNSS/INS possui um giroscópio, com alcance máximo de $1000^\circ/\text{s}$ e estabilidade de $10^\circ/\text{h}$, e um acelerômetro, com alcance máximo de $200\text{m}/\text{s}^2$ e estabilidade de $15\mu\text{g}$, ambos podendo ser atualizados a uma frequência máxima de 2000Hz. Além disso, o modelo utilizado contém um filtro interno capaz de fornecer informações de orientação com erros inferiores a $0,3^\circ$ para rolamento (x) e arfagem (y) e a $0,8^\circ$ para guinada (z). Para esse sensor, os dados de magnetômetro e de *GPS* são geral-

mente descartados devido às interferências magnéticas presentes nas cavernas e pelo fato do ambiente ser fechado.

A Intel RealSense Depth D435i possui tamanhos reduzidos de 90mm x 25mm x 25mm e fica localizada na parte frontal do robô. O dispositivo possui uma IMU, uma lente RGB e um módulo para estimação da profundidade. A lente RGB tem uma resolução máxima de 1920x1080, com campos de visão horizontal e vertical de 64° e 41°, respectivamente, e uma frequência máxima de captura de 30fps. Por sua vez, o módulo de profundidade é composto por um par estéreo de sensores infravermelho com resolução máxima de 1280x720, possui um campo de visão de 86° x 57°, uma frequência máxima de captura de 90fps e um alcance de 0,2m a 10m (variando de acordo com as condições do ambiente).

Outro dispositivo particularmente importante para os experimentos em ambientes confinados é o sistema de iluminação do robô. A falta de iluminação adequada nesses locais é compensada por um par de LEDs Cree XLamp XHP35 de alto fluxo luminoso, localizados na parte frontal e na traseira do robô.

5.3.2 Ambientes Reais

Para os resultados em ambientes reais, foram realizados dois experimentos distintos na Mina do Veloso, um ambiente típico para aplicações com o EspeleoRobô. É importante ressaltar que nesta dissertação somente estão apresentados resultados de experimentos em ambientes representativos, ou seja, resultados em locais abertos ou estruturados não são abordados.

A Mina do Veloso é uma galeria subterrânea localizada em Ouro Preto-MG, criada manualmente por mineiros nos séculos XVIII e XIX para a extração de ouro. Possui um complexo de galerias e de corredores subterrâneos de aproximadamente 200 metros de extensão, sem luz natural. A mina apresenta características desafiadoras para robôs autônomos, como terrenos irregulares e escorregadios, degraus, áreas com risco de queda, interferência magnética pela presença de ferro no solo e falta de sinal de GPS. Além disso, a iluminação artificial presente no local não é suficiente para a obtenção de imagens apropriadas para realizar um mapeamento do ambiente, sendo então necessário a utilização dos LEDs do robô.

Visto que não há valores de *ground truth* nos experimentos realizados nesse ambiente, foram então utilizados como referência os resultados de localização e mapeamento provenientes do método de LiDAR SLAM chamado LeGO-LOAM (Shan and Englot, 2018). Esse método apresentou bons resultados nos ambientes apresentados nesta dissertação, principalmente para a parte de mapeamento (da Cruz Júnior et al., 2020; Azpúrua et al., 2021).

• Experimento Teleoperado na Mina do Veloso

A parte da mina relacionada ao primeiro experimento possui um espaço bastante restrito, com paredes muito próximas (cerca de 1,4 metros de distância) e um teto baixo com alturas variando entre 1,3 e 1,8 metros, conforme mostrado na Figura 27. Além disso, possui um sistema de iluminação no teto, uma canaleta para a passagem de rejeitos à direita e uma textura característica nas paredes e no chão. O terreno é pouco irregular, com inclinações máximas de 5° , sem locais de subidas e descidas consideráveis.

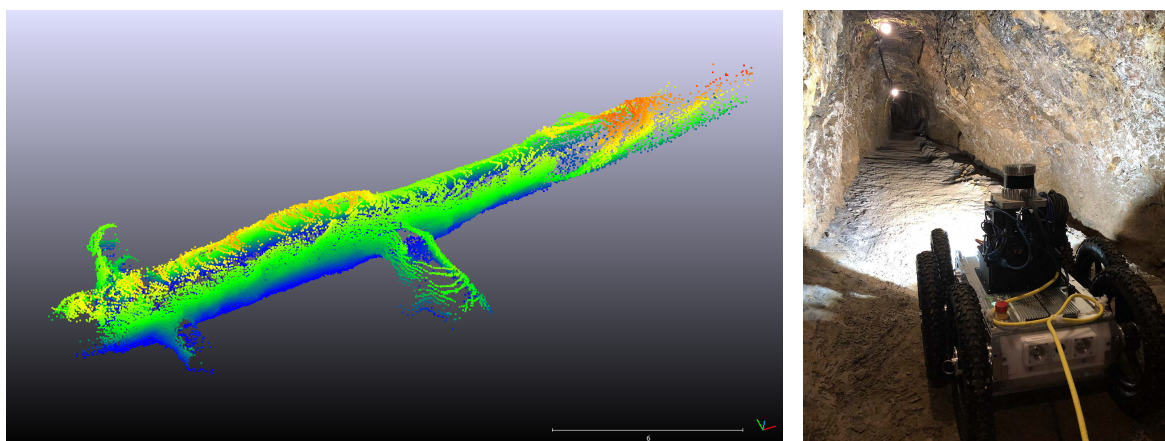


Figura 27 – Mapa gerado com o [LeGO-LOAM](#) e foto real da Mina do Veloso, na parte correspondente ao experimento teleoperado.

Esse primeiro experimento foi realizado no dia 21 de fevereiro de 2020, onde os dados dos sensores presentes no robô foram gravados em *bags*, de forma a serem reproduzidos posteriormente para a geração dos resultados. É importante destacar que, nesse experimento, devido ao grande volume de informações, os dados dos sensores apresentaram certa instabilidade. Além disso, a Xsens apresentou uma falha no funcionamento e os dados de IMU foram obtidos da Realsense D435i. O experimento foi realizado de forma teleoperada em um percurso relativamente simples, composta por trechos em linha reta, curvas e giros. O tempo total do experimento foi de 1:05 (ou 65 segundos), com uma velocidade média de 0,25m/s e uma distância percorrida de pouco mais de 10 metros.

Para a geração dos resultados relativos a esse experimento foram utilizados os seguintes parâmetros: `friction_coeff = 1,0`; `image_size = 360p`; `min_range = 1m`; e `max_range = 5m`.

• Experimento Semiautônomo na Mina do Veloso

O segundo experimento na Mina do Veloso também foi realizado em um espaço restrito, com paredes muito próximas (cerca de 1,2 metros de distância), porém, com um teto relativamente alto, variando entre 2 e 2,5 metros de altura, conforme mostrado na Figura 28. Essa parte da mina também possui um sistema de iluminação no teto e

uma textura característica nas paredes e no chão, porém, com uma menor saturação de cor. Nesse caso, o terreno mostrou uma irregularidade muito superior ao anterior, com inclinações variando cerca de 20° , além de apresentar uma subida de aproximadamente 8° na parte inicial e de 15° na parte final.

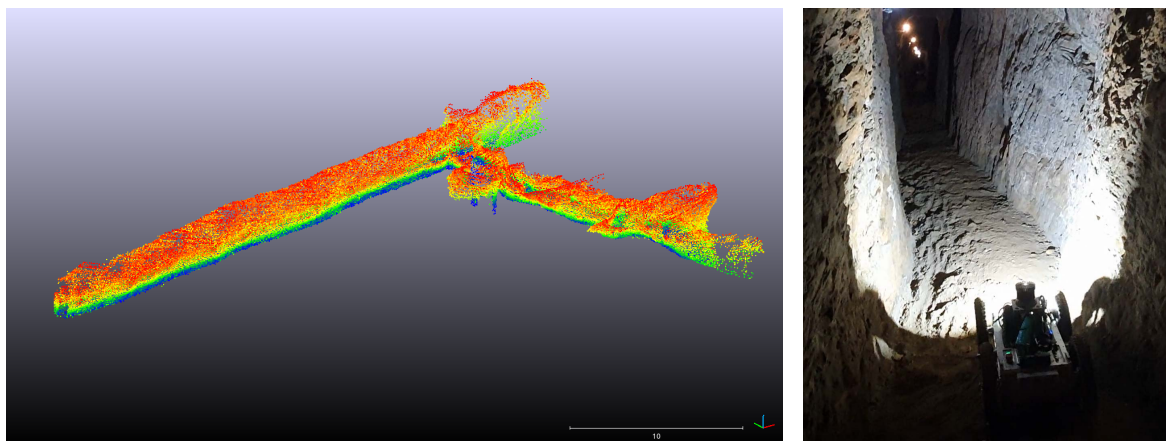


Figura 28 – Mapa gerado com o [LeGO-LOAM](#) e foto real da Mina do Veloso, na parte correspondente ao experimento semiautônomo.

Esse experimento foi realizado no dia 18 de dezembro de 2020, com parte dos dados dos sensores gravados em *bags*, para serem posteriormente reproduzidos na geração dos resultados. O maior problema enfrentado nesse experimento foi relacionado à movimentação do robô, o qual encontrou dificuldades em alguns trechos, realizando alguns movimentos bruscos de rotação. De forma geral, o percurso foi composto por duas partes relativamente retas separadas por uma curva de 90° , porém, o robô realizou curvas e giros em vários locais. Nesse experimento, foram manualmente determinados pontos no mapa, com o robô se deslocando de forma autônoma entre eles. O tempo foi de 3:52 (ou 232 segundos), com uma velocidade média de $0,35\text{m/s}$ e uma distância percorrida de mais de 50 metros.

Para a geração dos resultados relativos a esse experimento foram utilizados os seguintes parâmetros: `friction_coeff = 0,64`; `image_size = 240p`; `min_range = 1m`; e `max_range = 6m`.

Capítulo 6

Resultados e Discussões

Neste Capítulo, são apresentados os resultados de localização e o mapeamento do registro da nuvem de pontos, combinado com o [EKF](#), e do [RTAB-Map](#). Ambos os métodos são comparados com os valores de referência, dados pelo simulador ou pelo [LeGO-LOAM](#), conforme explicado anteriormente. É importante mencionar que, embora o objetivo principal desta dissertação seja a reconstrução do ambiente, a localização dos métodos é também analisada por estar diretamente relacionada à deformação dos mapas.

De forma a facilitar a referência aos métodos, foram utilizadas neste Capítulo as seguintes nomenclaturas:

- *Registro-EKF*: para os resultados de localização com o [EKF](#) e de mapeamento com o registro da nuvem de pontos;
- *RTAB-Map*: para os resultados de [SLAM](#) com o [RTAB-Map](#);
- *Ground-Truth*: para a localização e os mapas obtidos com o simulador; e
- *LiDAR-SLAM*: para os resultados com o [LeGO-LOAM](#) nos ambientes reais.

O computador utilizado na geração dos resultados conta com o [ROS](#) na versão *kinetic*, rodando no Ubuntu 16.04, possui uma CPU Intel[®] Core[™] i7-3632QM 2.20GHz, com 8GB de RAM e um processador gráfico integrado Intel[®] HD Graphics 4000.

6.1 Técnicas de Avaliação Empregadas

A avaliação dos resultados de cada método foi realizada por meio de quatro métricas distintas. Para isso, foram utilizados dados confiáveis como referência de modo a comparar quantitativamente os resultados de localização e de mapeamento. De forma a padronizar a simbologia, foram utilizados os índices l e m como referência à localização e ao mapeamento, e x e r para os resultados de algum dos métodos e para a referência.

Primeiramente foi analisado o erro relativo à trajetória ao longo do tempo obtida pelos métodos. De forma a representar os valores de posição da odometria de algum método X, tem-se o conjunto de pontos $\mathcal{P}_{l,x} = \{\mathbf{p}_{l,x}[1], \dots, \mathbf{p}_{l,x}[k], \dots, \mathbf{p}_{l,x}[n_{l,x}]\}$, onde $\mathbf{p}_{l,x}[k] = [x_{l,x}[k], y_{l,x}[k], z_{l,x}[k]]^T \in \mathbb{R}^3$. De maneira similar, os valores de orientação são representados pelo conjunto $\mathcal{O}_{l,x} = \{\mathbf{o}_{l,x}[1], \dots, \mathbf{o}_{l,x}[k], \dots, \mathbf{o}_{l,x}[n_{l,x}]\}$, onde $\mathbf{o}_{l,x} \in \mathbb{R}^3$ é dado pelos ângulos de rolamento, arfagem e guinada obtidos com a odometria no formato $[\phi_{l,x}[k], \theta_{l,x}[k], \psi_{l,x}[k]]^T$. É importante ressaltar que ambos os conjuntos foram obtidos no tempo próprio do método, possuindo um tamanho $n_{l,x}$.

Nesse mesmo sentido, os conjuntos de pontos $\mathcal{P}_{l,r}$ e $\mathcal{O}_{l,r}$ representam respectivamente as posições e orientações de referência, obtidas em seu próprio tempo e com tamanho $n_{l,r}$. Para realizar uma comparação entre os conjuntos, foi primeiramente computada uma interpolação linear utilizando uma sequência de tempos preestabelecidos e de tamanho n_l . Dessa forma, foram definidos os conjuntos $\bar{\mathcal{P}}_{l,x}$, $\bar{\mathcal{O}}_{l,x}$, $\bar{\mathcal{P}}_{l,r}$ e $\bar{\mathcal{O}}_{l,r}$, representando a localização de um dos métodos e da referência, possuindo mesmo tamanho, com valores de tempos iguais e passíveis de comparação. Assim, é possível definir uma métrica $\mathbf{m}_l \in \mathbb{R}^6$, representando o erro médio de posição e orientação em cada eixo, na forma:

$$\mathbf{m}_l = \begin{bmatrix} n_l^{-1} \sum_k |\bar{x}_{l,x}[k] - \bar{x}_{l,r}[k]| \\ n_l^{-1} \sum_k |\bar{y}_{l,x}[k] - \bar{y}_{l,r}[k]| \\ n_l^{-1} \sum_k |\bar{z}_{l,x}[k] - \bar{z}_{l,r}[k]| \\ n_l^{-1} \sum_k |\bar{\phi}_{l,x}[k] - \bar{\phi}_{l,r}[k]| \\ n_l^{-1} \sum_k |\bar{\theta}_{l,x}[k] - \bar{\theta}_{l,r}[k]| \\ n_l^{-1} \sum_k |\bar{\psi}_{l,x}[k] - \bar{\psi}_{l,r}[k]| \end{bmatrix}. \quad (6.1)$$

Outra avaliação de localização mais simples e comumente utilizada corresponde ao erro da posição final. Esta pode ser facilmente calculada pela distância euclidiana entre as posições finais do método e da referência por meio de:

$$m_f = \|\bar{\mathbf{p}}_{l,x}[n_l] - \bar{\mathbf{p}}_{l,r}[n_l]\|. \quad (6.2)$$

Nesse caso, o resultado também pode ser expressado em forma de porcentagem, levando em consideração a distância percorrida pelo robô, calculada por:

$$d_{l,r} = \sum_k \|\mathbf{p}_{l,r}[k] - \mathbf{p}_{l,r}[k-1]\|. \quad (6.3)$$

Desse modo, a métrica da pose final pode ser obtida na forma porcentual por meio de:

$$m_f(\%) = 100 \frac{\|\bar{\mathbf{p}}_{l,x}[n_l] - \bar{\mathbf{p}}_{l,r}[n_l]\|}{d_{l,r}}. \quad (6.4)$$

Para realizar a comparação entre as reconstruções dos métodos, foi utilizado o *software* CloudCompare¹. Inicialmente, foram realizadas duas etapas de pré-processamento das nuvens de pontos, de forma a garantir melhores resultados. A primeira etapa consiste no alinhamento fino entre a reconstrução obtida pelo método e a referência, utilizando a ferramenta *Fine Registration* do próprio *software*. Em seguida, foi definido um padrão para a distância mínima entre os pontos, de modo a garantir uma nuvem mais homogênea entre os métodos. Para tal, foi realizada uma subamostragem espacial com a ferramenta *Subsample*, utilizando uma distância mínima entre os pontos de 0,01m.

Para a comparação das reconstruções, devem ser levados em consideração duas situações: o mapa de referência na forma de nuvem de pontos ou na forma de *mesh*. Em ambos os casos, foram calculadas as distâncias entre as reconstruções dos métodos e o mapa de referência, por meio das ferramentas *Cloud/Cloud dist* e *Cloud/Mesh dist* do *software* CloudCompare. Para a comparação entre nuvem de pontos, dado um ponto $\mathbf{p}_{m,x}[k] \in \mathbb{R}^3$ da nuvem de reconstrução de algum dos métodos, é definido um raio $c_{m,x}$ ao seu redor de forma a englobar parte da nuvem de referência $\mathcal{P}_{m,r}$. Em seguida, são calculadas as distâncias de $\mathbf{p}_{m,x}[k]$ a todos os pontos $\mathbf{p}_{m,r} \in \mathcal{P}_{m,r}$. A menor dessas distâncias é definida como a distância do ponto $\mathbf{p}_{m,x}[k]$ à nuvem de referência, sendo expressa como:

$$d_{m,x}[k] = \min_{\mathbf{p}_{m,r} \in \mathcal{P}_{m,r}} \|\mathbf{p}_{m,x}[k] - \mathbf{p}_{m,r}\|. \quad (6.5)$$

Um formato *mesh* é geralmente composto por triângulos resultantes da ligação entre seus vértices. Dessa forma, dado um ponto $\mathbf{p}_{m,x}[k]$ da reconstrução de algum dos métodos e um raio $c_{m,x}$ ao seu redor, é determinado um conjunto de triângulos $\mathcal{S}_{m,r}$ contidos dentro da esfera resultante. Em seguida, é calculada a distância do ponto relativa a cada triângulo $\mathbf{s}_{m,r} \in \mathcal{S}_{m,r}$, a qual é dada pela distância do ponto ao plano do triângulo, caso sua projeção ortogonal $\bar{\mathbf{p}}_{m,x}[k]$ neste plano esteja dentro do triângulo, ou pela distância do ponto à aresta mais próxima, caso contrário. Para melhor visualização, a Figura 29 apresenta uma representação gráfica do processo. Considerando então os três vértices $\{\mathbf{v}_{m,r}[1], \mathbf{v}_{m,r}[2], \mathbf{v}_{m,r}[3]\}$ do triângulo $\mathbf{s}_{m,r}$, a menor distância entre o ponto $\mathbf{p}_{m,x}[k]$ ao *mesh* de referência pode ser calculada por:

$$d_{m,x}[k] = \min_{\mathbf{s}_{m,r} \in \mathcal{S}_{m,r}} \begin{cases} \frac{|\mathbf{n}_{m,r} \cdot (\mathbf{p}_{m,x}[k] - \mathbf{v}_{m,r}[1])|}{\|\mathbf{n}_{m,r}\|} & , \text{ caso } \bar{\mathbf{p}}_{m,x}[k] \in \mathbf{s}_{m,r} \\ \frac{\|(\mathbf{v}_{m,r}[2] - \mathbf{v}_{m,r}[1]) \times (\mathbf{v}_{m,r}[1] - \mathbf{p}_{m,x}[k])\|}{\|\mathbf{v}_{m,r}[2] - \mathbf{v}_{m,r}[1]\|} & , \text{ caso } \bar{\mathbf{p}}_{m,x}[k] \notin \mathbf{s}_{m,r} \end{cases} \quad (6.6)$$

¹ <http://www.cloudcompare.org/>

onde $\mathbf{n}_{m,r}$ corresponde ao vetor normal do triângulo $\mathcal{S}_{m,r}$, que pode ser obtido com o produto vetorial $\|(\mathbf{v}_{m,r}[3] - \mathbf{v}_{m,r}[1]) \times (\mathbf{v}_{m,r}[3] - \mathbf{v}_{m,r}[2])\|$. Nesse caso, foi utilizado o vértice $\mathbf{v}_{m,r}[1]$ como referência para o cálculo da distância do ponto ao plano e, por simplificação, $(\mathbf{v}_{m,r}[2] - \mathbf{v}_{m,r}[1])$ foi tomada como a aresta mais próxima do ponto $\mathbf{p}_{m,x}[k]$.

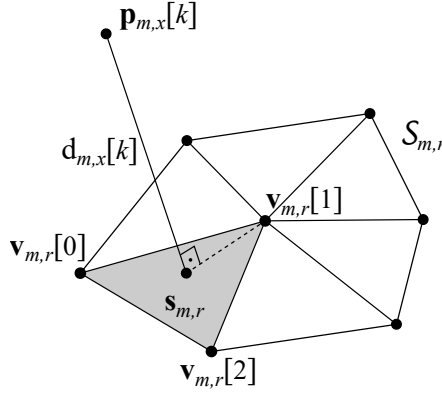


Figura 29 – Representação gráfica do cálculo da distância entre um ponto da reconstrução de algum dos métodos para o *mesh* de referência.

Por fim, a métrica utilizada para o mapeamento corresponde a um vetor com as menores distâncias de todos os pontos em relação à referência, independentemente de seu tipo (nuvem ou *mesh*). Assim, para uma nuvem de pontos $\mathcal{P}_{m,x}$ de tamanho $n_{m,x}$ obtida por algum dos métodos, o vetor contendo todas as distâncias relacionadas a reconstrução pode ser dado por:

$$\mathbf{m}_m = [d_{m,x}[0] \ d_{m,x}[1] \ \dots \ d_{m,x}[k] \ \dots \ d_{m,x}[n_{m,x}]]^T . \quad (6.7)$$

Devido a grande quantidade de pontos obtidas nas reconstruções, geralmente na ordem de milhões, foram utilizadas duas técnicas para demonstrar os resultados da métrica \mathbf{m}_m . Primeiramente, foi apresentada a própria a nuvem de pontos obtida pelos métodos, porém recolorida de modo a retratar a distância de cada ponto ao valor de referência. Nesse caso, pontos azuis possuem erros mais próximos a zero, verdes possuem erros intermediários e vermelhos estão mais distantes da referência. Além disso, foi utilizado um histograma para indicar de forma normalizada a quantidade de pontos dentro de cada intervalo de erro, com a mesma referência de cores da nuvem de pontos, incluindo uma representação da quantidade de pontos acumulada.

A segunda técnica consiste em apresentar dados estatísticos da distância dos pontos à referência. Para tal, foram utilizados os valores médios e o desvio-padrão, além dos intervalos de confiança mostrados na Figura 30. Nesse caso, é computada a quantidade de pontos contida em um desvio-padrão (1σ ou 68% dos pontos), dois desvios-padrão (2σ ou 95%) e três (3σ ou 99,7%).

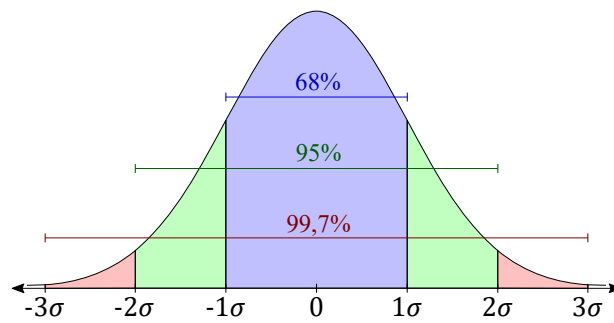


Figura 30 – Representação dos intervalos de confiança para um desvio-padrão (1σ ou 68% dos pontos), dois desvios-padrão (2σ ou 95%) e três (3σ ou 99,7%).

Em uma abordagem diferente, a última métrica consiste na análise do custo computacional de cada método. Nesse caso, foram coletados o tempo gasto no processamento de cada dado transmitido, tanto para a parte de localização quanto para o mapeamento. Assim, foram obtidos os vetores $\mathbf{t}_{l,x}$ e $\mathbf{t}_{m,x}$, possivelmente com quantidade distinta de elementos. O custo computacional total de cada método pode então ser computado por:

$$m_t = \sum_k \mathbf{t}_{l,x}[k] + \sum_k \mathbf{t}_{m,x}[k]. \quad (6.8)$$

6.2 Resultados na Caverna do DARPA

Os resultados relativos à localização no primeiro experimento em ambiente simulado estão apresentados a seguir. A Figura 31 mostra os resultados de localização obtidos com o *Registro-EKF* e o *RTAB-Map* nos planos xy , xz e yz , além da referência de posição apresentada pelo simulador (*Ground-Truth*). A Figura 32 ilustra os gráficos dos erros de posição (x , y e z) e de orientação (ϕ , θ e ψ) de cada método ao longo do tempo. Por sua vez, as Tabelas 2 e 3 apresenta esses valores de erro de forma quantitativa, calculados por meio das métricas m_l e m_f .

Tabela 2 – Erro médio de localização dos métodos na Caverna do DARPA.

Métrica m_l	x	y	z	ϕ_x	θ_y	ψ_z
<i>Registro-EKF</i>	0,562m	0,620m	0,367m	0,061°	0,073°	0,173°
<i>RTAB-Map</i>	1,115m	0,648m	0,168m	0,248°	0,325°	2,177°

Tabela 3 – Erro da posição final dos métodos na Caverna do DARPA.

Métrica m_f	Valor absoluto	Valor relativo
<i>Registro-EKF</i>	2,20m	0,5%
<i>RTAB-Map</i>	3,09m	0,7%

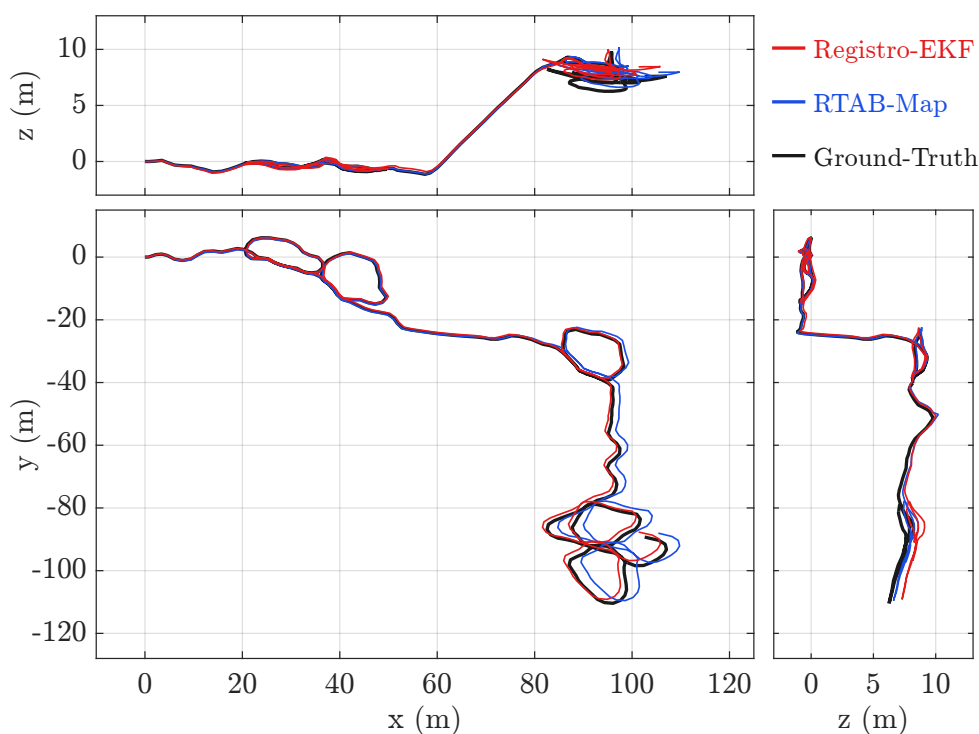


Figura 31 – Resultados da posição dos métodos e da referência nos planos xy , xz e yz na Caverna do [DARPA](#).

Neste experimento, o *Registro-EKF* apresentou resultados de localização ligeiramente melhores, com um erro de posição final de 2,20m (ou 0,5%), enquanto que o *RTAB-Map* atingiu um erro de 3,09m (ou 0,7%). Os maiores problemas observados nesse cenário foram relacionados a variações bruscas de inclinação nos momentos iniciais e finais de subidas e descidas, que ocorreram nos tempos de 360 e 540 segundos de experimento. Nesse caso, onde o *RTAB-Map* acumulou erros de orientação ψ , causando um maior erro na posição em x , e o *Registro-EKF* apresentou maior erro em z . Contudo, levando em consideração a distância percorrida de 440m, é possível observar que os resultados de localização nesse primeiro ambiente permaneceram muito próximos ao valor de referência durante todo o percurso.

Na sequência, as Figuras 33 e 34 apresentam os resultados de mapeamento do *Registro-EKF* e do *RTAB-Map* respectivamente. A imagem na parte superior das figuras corresponde a uma vista em perspectiva da reconstrução completa da caverna obtida por cada método e, na parte inferior, são ilustradas três vistas internas em locais específicos do mapa.

Observando as imagens, pode-se perceber que ambos os mapas apresentaram baixos níveis de ruído e distorção, sendo possível notar o formato da caverna. É ainda possível visualizar uma riqueza de detalhes do ambiente, como a cor e a textura do chão e das

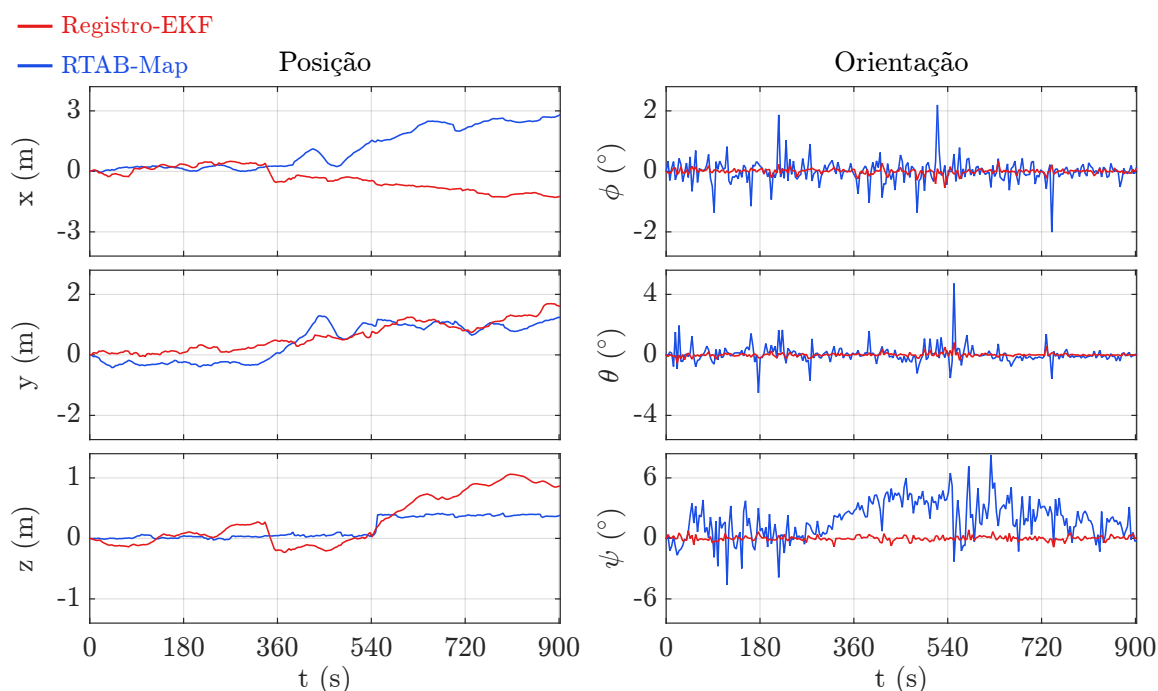


Figura 32 – Erros de posição e orientação dos métodos ao longo do tempo na Caverna do DARPA.

paredes da caverna, além da presença de algumas pedras. Comparando visualmente os dois mapas resultantes, não há muita diferença entre eles, apenas algumas regiões mais esparsas para o caso do *Registro-EKF*. O principal problema enfrentado nesse cenário consiste nas características físicas da caverna, onde o teto e algumas paredes permaneceram fora do alcance máximo da câmera (10m), além das alterações abruptas de inclinação nas subidas e descidas, que limitaram a observação do cenário. Nesse caso, vários pontos do ambiente não puderam ser capturados pela câmera, o que contribuiu com a presença de buracos nas reconstruções finais.

De modo a comparar quantitativamente os mapas gerados pelos métodos, os resultados a seguir apresentam os erros das reconstruções comparadas ao *Ground-Truth*, de acordo com os cálculos da métrica \mathbf{m}_m . Na parte superior da Figura 35, são apresentados os mapas gerados com os valores de erro de cada ponto indicados em cores (com azul para erros mais próximos de zero e vermelho para erros maiores que 0,5m), sendo o *Registro-EKF* à esquerda e o *RTAB-Map* à direita. De forma similar, os gráficos exibidos na parte inferior da figura correspondem ao histograma normalizado dos erros com a mesma referência de cores, além de uma linha indicando a quantidade acumulada de pontos. Para uma melhor visualização dos dados, a Tabela 4 apresenta o valor médio e o desvio-padrão dos erros, assim como os respectivos intervalos de confiança de 1σ , 2σ e 3σ , explicados anteriormente.

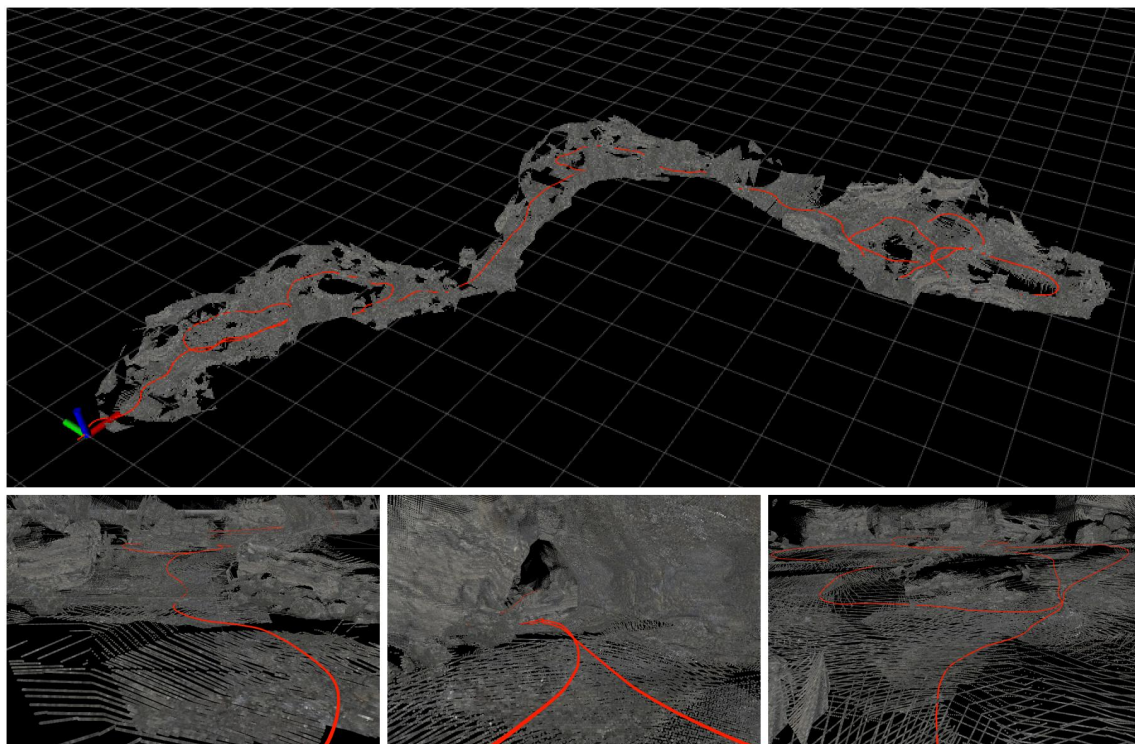


Figura 33 – Mapa obtido pelo *Registro-EKF* na Caverna do [DARPA](#) (vista em perspectiva acima e três vistas internas abaixo).

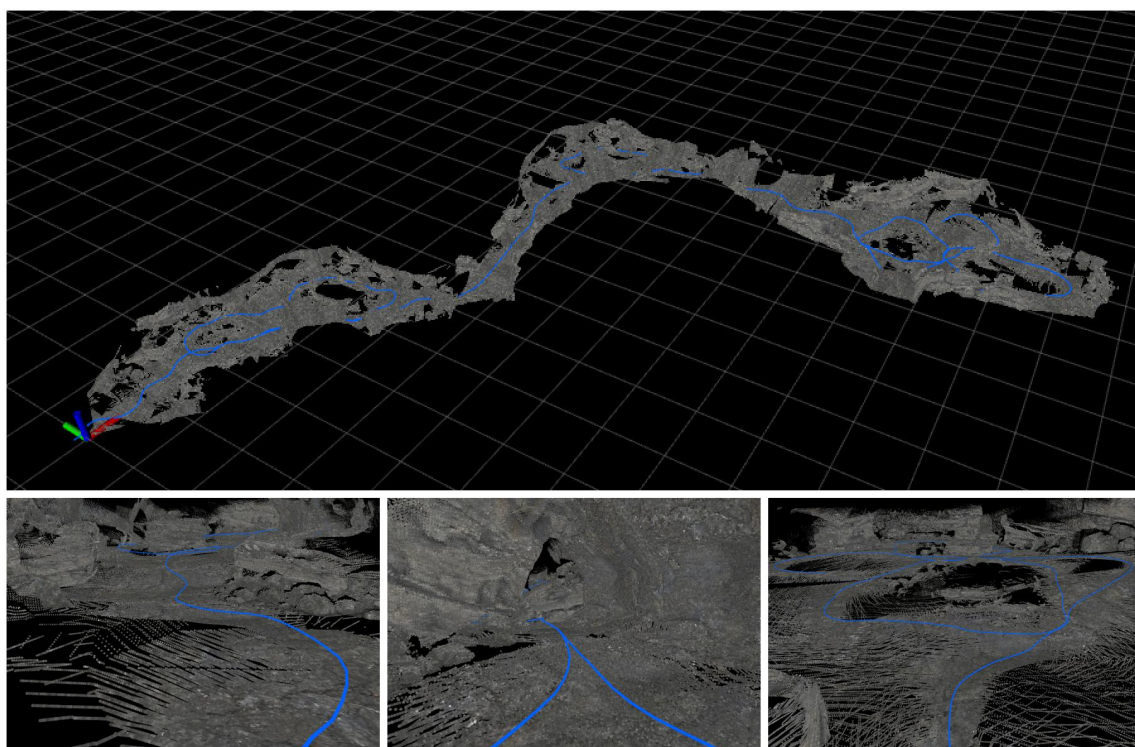


Figura 34 – Mapa obtido pelo *RTAB-Map* na Caverna do [DARPA](#) (vista em perspectiva acima e três vistas internas abaixo).

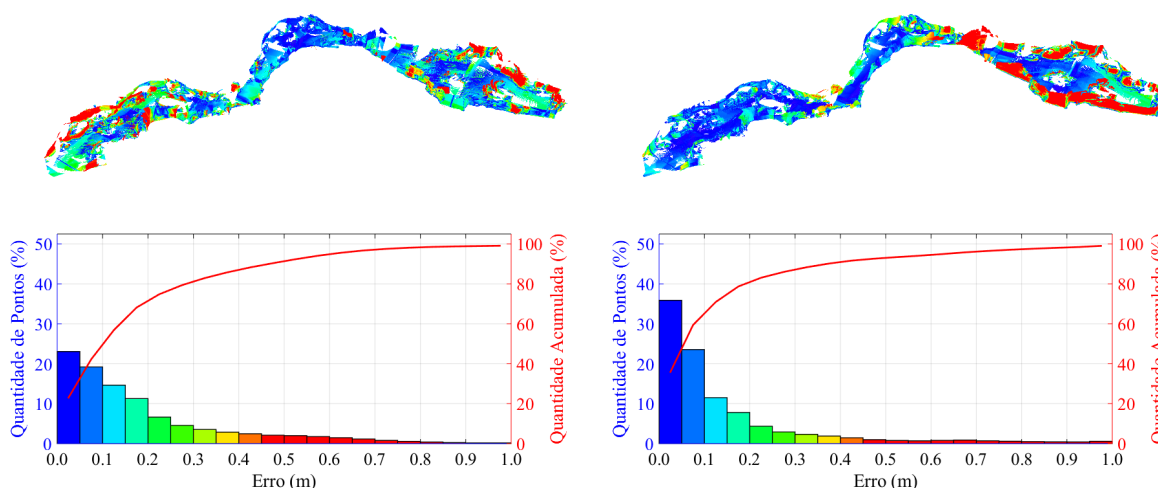


Figura 35 – Comparação dos mapas obtidos pelo *Registro-EKF* (à esquerda) e pelo *RTAB-Map* (à direita) com o *Ground-Truth* na Caverna do [DARPA](#). Acima está representado o erro de cada ponto do mapa de forma colorida e, abaixo, o histograma do erro com a mesma referência de cores.

Tabela 4 – Erro médio e intervalos confiança relativos aos pontos da nuvem obtida pelos métodos na Caverna do [DARPA](#).

Métrica m_m	Nº de pontos	Erro médio	1σ (68%)	2σ (95%)	3σ (99.7%)
<i>Registro-EKF</i>	4.843.252	$0,20 \pm 0,21m$	$\leq 0,17m$	$\leq 0,63m$	$\leq 1,31m$
<i>RTAB-Map</i>	6.789.432	$0,15 \pm 0,21m$	$\leq 0,11m$	$\leq 0,65m$	$\leq 1,08m$

De forma mais objetiva, as informações relacionadas aos erros das reconstruções demonstraram certa vantagem para o *RTAB-Map*, mesmo com resultados menos precisos na localização. Isso ocorreu pelo fato do método realizar processos de otimização do mapa, diferentemente do *Registro-EKF*, o qual apresentou deformações durante a maior parte do percurso. Este obteve um erro médio 33% maior e um mapa menos denso (quase 5 milhões de pontos), com 68% dos pontos com erro menor que 0,17m, 95% menor que 0,63m e 99,7% menor que 1,31m. Por outro lado, o *RTAB-Map* apresentou um mapa de quase 7 milhões de pontos, sendo os erros para 68%, 95% e 99,7% menores que 0,11m, 0,65m e 1,08m, respectivamente.

Por fim, foram também verificadas algumas informações relacionadas ao tempo de processamento dos métodos, conforme apresentado nas etapas da métrica m_t . A Tabela 5 demonstra os resultados para o *Registro-EKF*, indicando o número total de mensagens publicadas e o tempo de cada mensagem do [EKF](#), incluindo as modificações realizadas para o cálculo do eixo z , e do processamento do mapa, que engloba o cálculo da nuvem local e a atualização para o sistema de coordenadas global. Já para os resultados do *RTAB-Map*, a Tabela 6 apresenta o número total de mensagens e o tempo de processamento referentes às partes de odometria e mapeamento separadamente.

Tabela 5 – Informações relacionadas ao tempo de processamento da odometria e do mapeamento do *Registro-EKF* na Caverna do **DARPA**.

Número total de mensagens de odometria	9015
Número total de mensagens de mapeamento	361
Tempo da odometria por mensagem	$22 \pm 6 \mu\text{s}$
Tempo do mapeamento por mensagem	$210 \pm 79 \text{ ms}$
Métrica m_t (tempo total)	75.9 s

Tabela 6 – Informações relacionadas ao tempo de processamento da odometria e do mapeamento do *RTAB-Map* na Caverna do **DARPA**.

Número total de mensagens de odometria	14204
Número total de mensagens de mapeamento	902
Tempo da odometria por mensagem	$45 \pm 7 \text{ ms}$
Tempo do mapeamento por mensagem	$248 \pm 247 \text{ ms}$
Métrica m_t (tempo total)	857,8 s

Conforme explicado na Subseção 5.2.2, devido ao tamanho da caverna, a imagem utilizada no *Registro-EKF* foi convertida para uma resolução menor, garantindo um mapa resultante mais compacto. Isso provocou uma redução considerável na quantidade de pontos das nuvens locais, com um valor médio de 14651 ± 6801 , e consequentemente no tempo de processamento no mapeamento. Para a parte de localização, o **EKF** apresentou um tempo de processamento quase insignificante, na ordem de microssegundos. Em contrapartida, por se tratar de um método que utiliza dados mais complexos (imagens) para o processo de odometria, o *RTAB-Map* mostrou-se mais lento, principalmente na parte de localização. Contudo, o método obteve um tempo total de processamento menor que o tempo gasto no experimento, indicando a possibilidade de ser utilizado de forma *online*. Outra informação importante a ser analisada é o número de *inliers* por *frame*, onde *RTAB-Map* foi capaz de obter uma média de 284 ± 81 . Esse valor foi suficiente para o método apresentar bons resultados, e o alto desvio-padrão está de acordo com os problemas relacionados ao baixo campo de visão da câmera. Por fim, comparando ambos os métodos, o *Registro-EKF* apresentou um tempo total de processamento cerca de 11 vezes menor que o *RTAB-Map*.

6.3 Resultados na Mina do **DARPA**

Os resultados de localização relativos ao segundo experimento em simulador estão apresentados a seguir. A Figura 36 mostra os resultados de odometria obtidos com o *Registro-EKF* e o *RTAB-Map* nos planos xy , xz e yz , além do *Ground-Truth* do simulador utilizado como referência. Os gráficos dos erros de posição e de orientação de cada método

ao longo do tempo são ilustrados na Figura 37. As Tabelas 7 e 8 apresenta os valores de erro de uma forma quantitativa, de acordo com as métricas m_l e m_f .

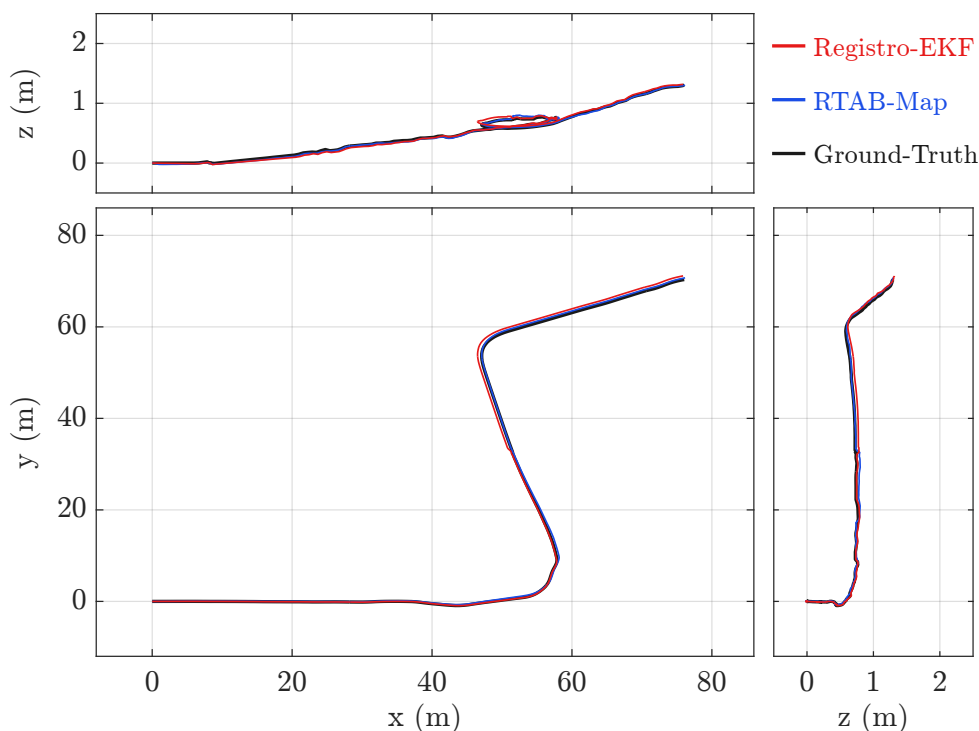


Figura 36 – Resultados da posição dos métodos e da referência nos planos xy , xz e yz na Mina do DARPA.

Tabela 7 – Erro médio de localização dos métodos na Mina do DARPA.

Métrica m_l	x	y	z	ϕ_x	θ_y	ψ_z
<i>Registro-EKF</i>	0,242m	0,339m	0,023m	0,083°	0,072°	0,063°
<i>RTAB-Map</i>	0,131m	0,315m	0,013m	0,330°	0,319°	0,502°

Os resultados de localização de ambos os métodos foram bastante precisos, apresentando valores muito próximos ao esperado durante todo o percurso. Nesse cenário, o *RTAB-Map* apresentou resultados de posição ligeiramente melhores, enquanto que o *Registro-EKF* foi melhor na parte de orientação. Além disso, o *RTAB-Map* apresentou melhor resultado em relação à posição final, com um erro de apenas 0,40m, contra 0,82m do *Registro-EKF*. O fator que mais contribuiu com o erro deste segundo método ocorreu no tempo de 200s (ou 46m em x e 33m em y), quando o robô enfrentou problemas ao tentar atravessar um trilho presente no cenário, provocando um escorregamento das rodas e, conseqüentemente, um erro de posição no *EKF*. Contudo, os erros ao longo do percurso foram relativamente baixos, mantendo menor que 1m para a posição (5cm para o eixo z) e menor que 2° para a orientação (com apenas picos de 3°).

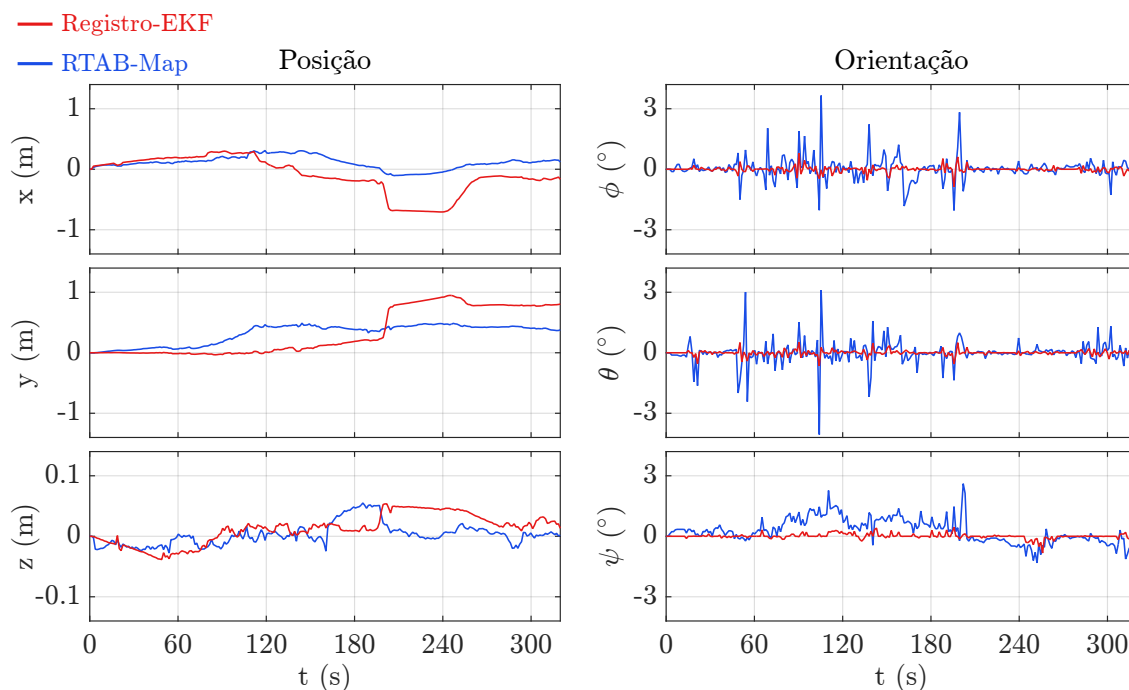


Figura 37 – Erros de posição e orientação dos métodos ao longo do tempo na Mina do [DARPA](#).

Tabela 8 – Erro da posição final dos métodos na Mina do [DARPA](#).

Métrica m_f	Valor absoluto	Valor relativo
<i>Registro-EKF</i>	0,82m	0,6%
<i>RTAB-Map</i>	0,40m	0,3%

Os resultados de mapeamento do *Registro-EKF* e do *RTAB-Map* estão apresentados nas Figuras 38 e 39, respectivamente, com uma vista em perspectiva na parte superior e três vistas internas na parte inferior. Os mapas obtidos pelos métodos nesse ambiente foram novamente muito similares entre si. É possível observar nas imagens que ambas as reconstruções mostraram ser bem fiéis ao mapa original da caverna, preservando a textura, a geometria, a cor e os objetos no cenário, como sistemas de iluminação e trilhos. A maior diferença entre os mapas está relacionada a densidade de pontos, com o *RTAB-Map* apresentando um mapa mais homogêneo e com menor quantidade de buracos, porém, menos denso que a reconstrução obtida com o *Registro-EKF*.

Em seguida, os resultados relacionados à comparação das reconstruções dos métodos com o *Ground-Truth* são apresentados na Figura 40, sendo o *Registro-EKF* à esquerda e o *RTAB-Map* à direita. Na parte superior estão os mapas gerados com os valores de erro de cada ponto indicados em cores, e abaixo são exibidos os histogramas normalizados dos erros com a mesma referência de cores e a quantidade acumulada de pontos. Conforme apresentado pela métrica m_m , para uma visualização mais objetiva dos dados, a Tabela 9 apresenta as informações do erro médio e dos intervalos de confiança.

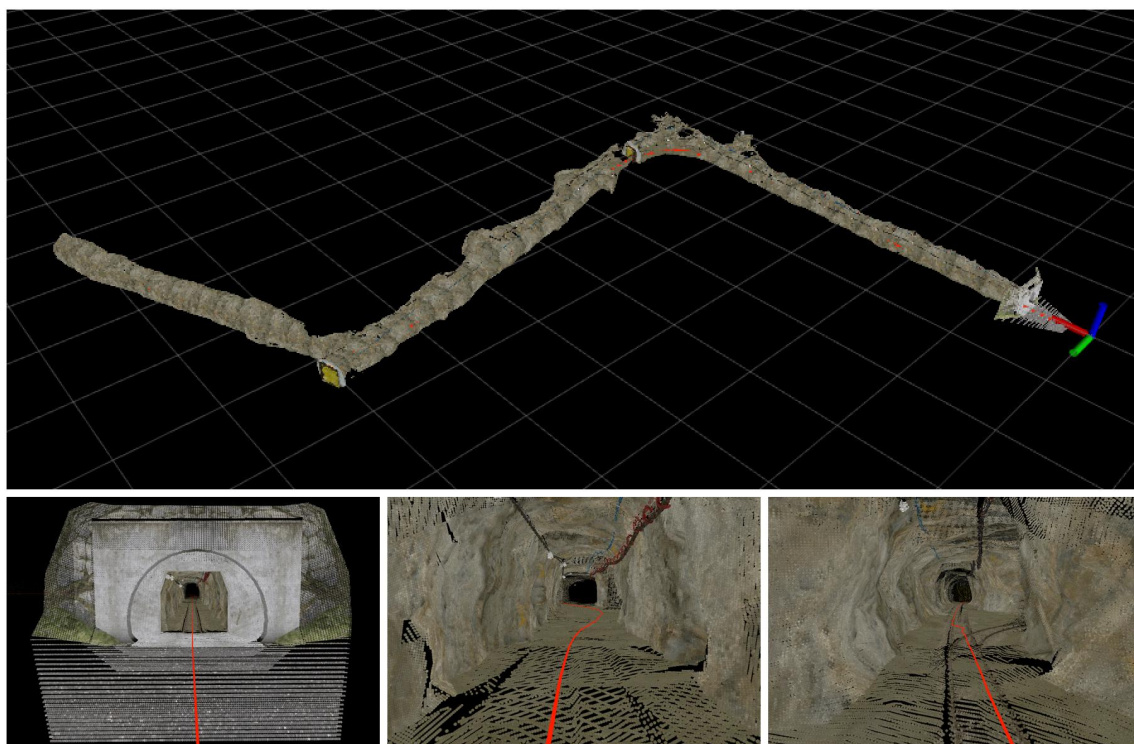


Figura 38 – Mapa obtido pelo *Registro-EKF* na Mina do [DARPA](#) (vista em perspectiva acima e três vistas internas abaixo).

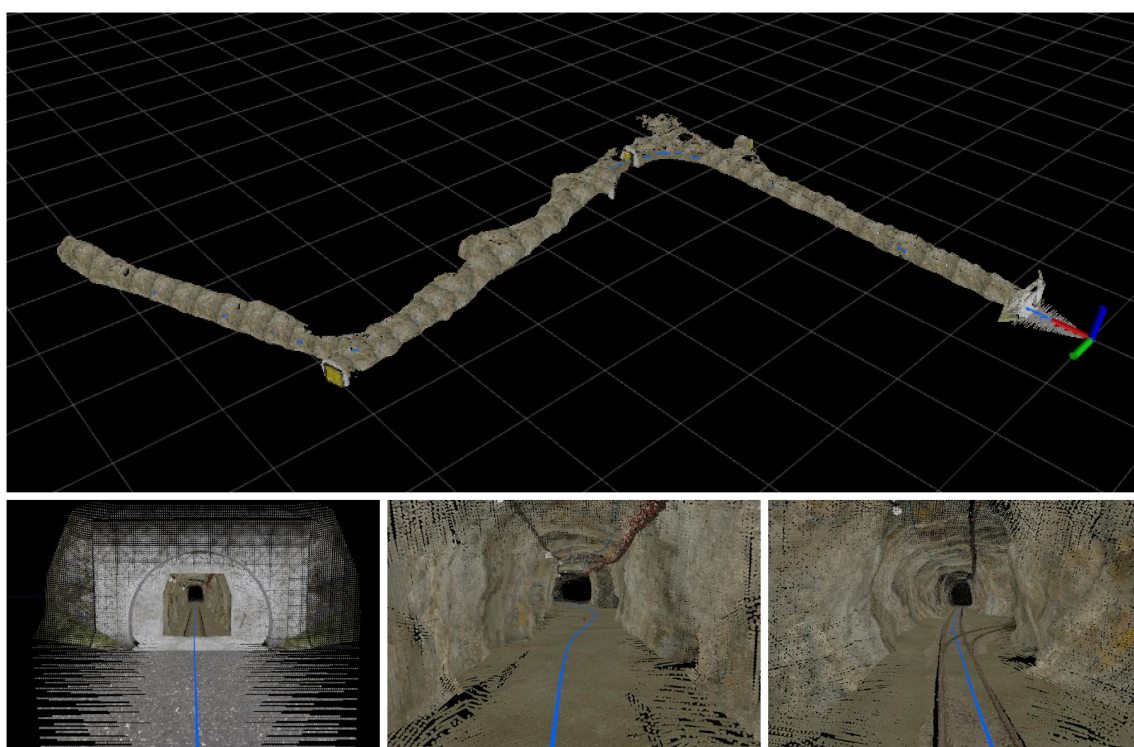


Figura 39 – Mapa obtido pelo *RTAB-Map* na Mina do [DARPA](#) (vista em perspectiva acima e três vistas internas abaixo).

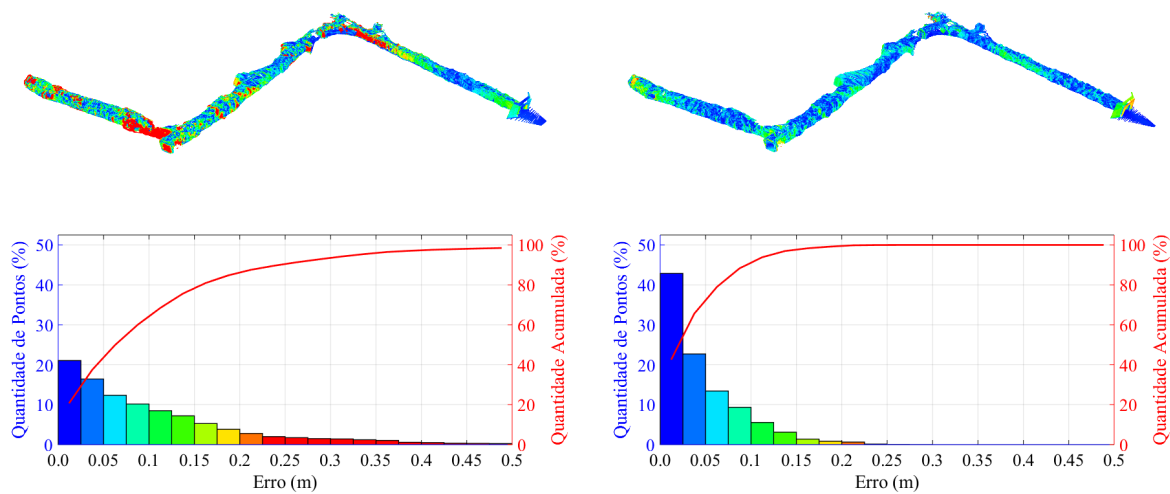


Figura 40 – Comparação dos mapas obtido pelo *Registro-EKF* (à esquerda) e pelo *RTAB-Map* (à direita) com o *Ground-Truth* na Mina do [DARPA](#). Acima é representado o erro de cada ponto do mapa de forma colorida e, abaixo, o histograma do erro com a mesma referência de cores.

Tabela 9 – Erro médio e intervalos confiança relativos aos pontos da nuvem obtida pelos métodos na Mina do [DARPA](#).

Métrica \mathbf{m}_m	Nº de pontos	Erro médio	1σ (68%)	2σ (95%)	3σ (99.7%)
<i>Registro-EKF</i>	3.924.313	$0,11 \pm 0,11\text{m}$	$\leq 0,11\text{m}$	$\leq 0,34\text{m}$	$\leq 0,62\text{m}$
<i>RTAB-Map</i>	3.309.290	$0,05 \pm 0,04\text{m}$	$\leq 0,05\text{m}$	$\leq 0,13\text{m}$	$\leq 0,22\text{m}$

Nesse cenário, o *RTAB-Map* apresentou um mapa mais próximo da referência de *Ground-Truth* da caverna, alcançando valores de erros bastante reduzidos. O método obteve uma reconstrução com cerca de 3,3 milhões de pontos, sendo 68% com erros menores que 0,05m, 95% menores que 0,13m e 99,7% menores que 0,22m. Enquanto isso, o *Registro-EKF* apresentou valores de erro cerca de duas vezes maior que o *RTAB-Map*, onde dos quase 4 milhões de pontos, 68% tiveram erros menores que 0,11m, 95% menores que 0,34m e 99,7% menores que 0,62m. Contudo, levando em consideração a distância percorrida de quase 150m, os pontos de ambos os mapas estiveram muito próximos ao *Ground-Truth*.

Por último, as Tabelas 10 e 11 exibem algumas informações relacionadas ao tempo de processamento dos métodos, de acordo com a métrica m_t . A primeira demonstra os tempos gastos no *EKF* e no cálculo do mapa para o *Registro-EKF*. E, para os resultados do *RTAB-Map*, a segunda tabela apresenta os tempos de processamento referentes às partes de odometria e de mapeamento. Assim como no experimento anterior, a parte de localização do *Registro-EKF* teve pouca influência no tempo de processamento do método, com uma média de $22\mu\text{s}$. Nesse caso, como foi utilizado uma resolução maior na imagem

Tabela 10 – Informações relacionadas ao tempo de processamento da odometria e do mapeamento do *Registro-EKF* na Mina do [DARPA](#).

Número total de mensagens de odometria	3139
Número total de mensagens de mapeamento	128
Tempo da odometria por mensagem	$22 \pm 7 \mu\text{s}$
Tempo do mapeamento por mensagem	$233 \pm 67 \text{ ms}$
Métrica m_t (tempo total)	29,9 s

Tabela 11 – Informações relacionadas ao tempo de processamento da odometria e do mapeamento do *RTAB-Map* na Mina do [DARPA](#).

Número total de mensagens de odometria	4000
Número total de mensagens de mapeamento	321
Tempo da odometria por mensagem	$52 \pm 6 \text{ ms}$
Tempo do mapeamento por mensagem	$187 \pm 110 \text{ ms}$
Métrica m_t (tempo total)	266,2 s

de entrada, o tempo de processamento do método foi um pouco maior que o anterior, com uma média de 16817 ± 3933 pontos por nuvem local. Entretanto, o tempo total de processamento do método ainda foi cerca de 9 vezes menor que o *RTAB-Map*. Este último, por sua vez, obteve resultados mais homogêneos (com menores valores de desvio-padrão) que no experimento anterior, apresentando um número médio de *inliers* iguais à 294 ± 54 . Um dos motivos para essa ocorrência é o fato da parede e do teto estarem dentro do alcance da câmera, provocando uma maior uniformidade nas imagens de entrada do método. Além disso, o *RTAB-Map* necessitou de um tempo menor de processamento por *keyframe*, atingindo um tempo total de processamento de cerca de 80% do tempo do experimento, reforçando a possibilidade de ser utilizado de forma *online*.

6.4 Resultados do Experimento Teleoperado na Mina du Veloso

Os resultados do *Registro-EKF* e do *RTAB-Map* no experimento teleoperado na Mina du Veloso estão apresentados a seguir. Conforme explicado anteriormente, nesse ambiente foram utilizados os dados do *LiDAR-SLAM* como referência. Sendo assim, os dados tratados como erros nos resultados das simulações são tratados como diferenças nos experimentos em ambientes reais. Os resultados de localização nos planos xy , xz e yz são apresentados na Figura 41. A Figura 42 exibe e os gráficos da diferença dos métodos em comparação com a referência ao longo do tempo. Os valores da diferença entre a localização dos métodos e a referência, indicados pelas métricas m_l e m_f , são demonstrados nas Tabelas 12 e 13.

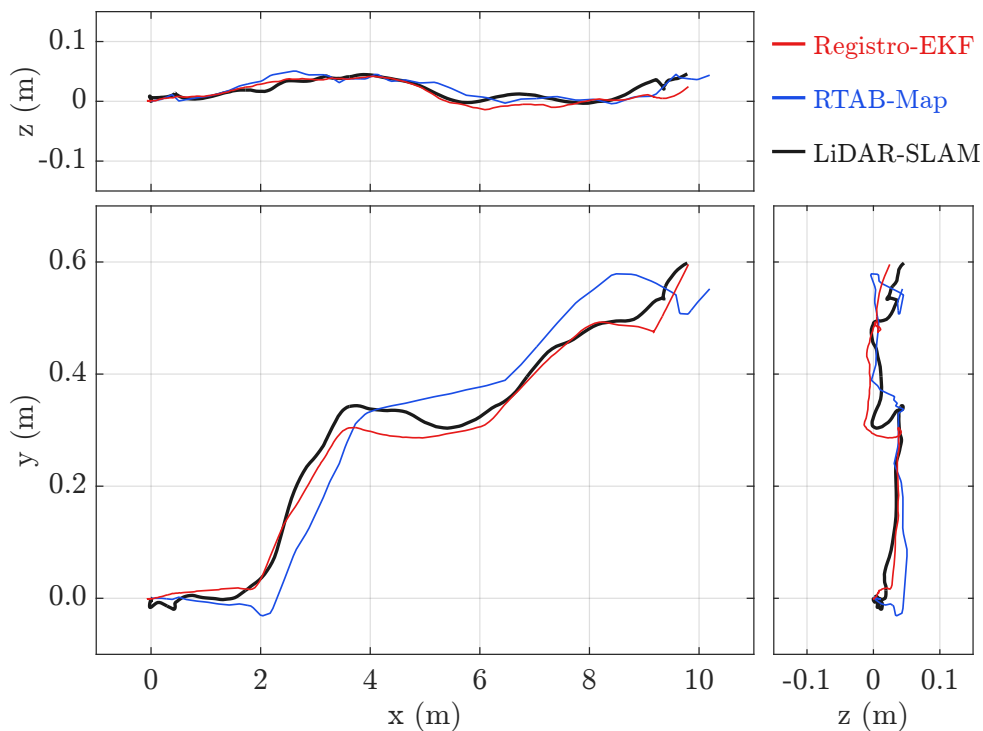


Figura 41 – Resultados da posição dos métodos e da referência nos planos xy , xz e yz no experimento teleoperado na Mina do Veloso.

Tabela 12 – Diferença média entre a localização dos métodos e a referência no experimento teleoperado na Mina do Veloso.

Métrica m_l	x	y	z	ϕ_x	θ_y	ψ_z
<i>Registro-EKF</i>	0,050m	0,021m	0,007m	1,061°	0,952°	1,984°
<i>RTAB-Map</i>	0,128m	0,035m	0,006m	0,704°	0,967°	1,033°

É possível perceber que os resultados em um ambiente real não foram tão precisos quanto os resultados de simulação, principalmente devido a incertezas e ruídos das medições. Nesse experimento, foi possível observar que o sistema de iluminação do robô foi suficiente para a VO, sendo as maiores dificuldades relacionadas à instabilidade de conexão com os sensores, que provocou uma menor precisão nos dados obtidos pelos métodos. Nesse caso, as imagens (colorida e de profundidade) da câmera não ficaram sincronizadas, o que atrapalhou consideravelmente o desempenho do *RTAB-Map*. Contudo, as poses obtidas pelos métodos foram relativamente próximas ao valor de referência com diferenças menores que 0,5m na posição e que 8° na orientação.

Tabela 13 – Diferença entre a posição final dos métodos e a referência no experimento teleoperado na Mina do Veloso.

Métrica m_f	Valor absoluto	Valor relativo
<i>Registro-EKF</i>	0,03m	0,3%
<i>RTAB-Map</i>	0,41m	3,3%

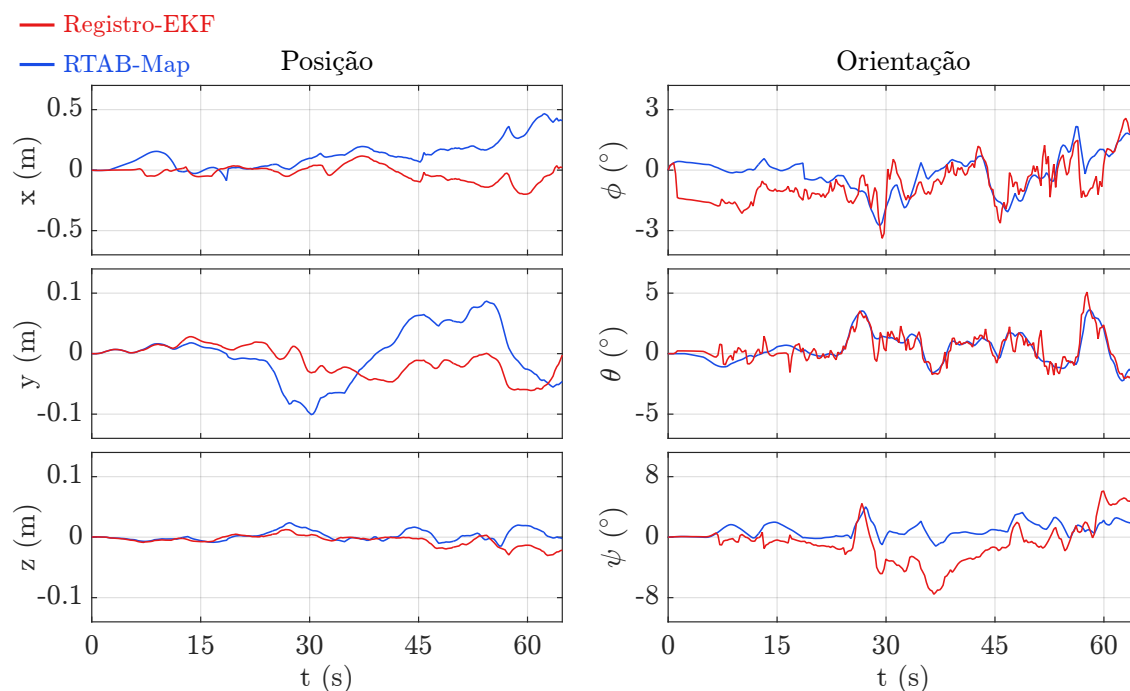


Figura 42 – Diferença de posição e orientação dos métodos em relação à referência ao longo do tempo no experimento teleoperado na Mina do Veloso.

Os resultados de mapeamento do *Registro-EKF* e do *RTAB-Map* estão apresentados nas Figuras 43 e 44. Embora tenha apresentado piores resultados na localização, o *RTAB-Map* com gerou um mapa com menos distorções, onde é possível visualizar a parte final da caverna desde o ponto inicial (imagem inferior esquerda). O *Registro-EKF* apresentou uma reconstrução mais densa, porém, com uma visualização limitada de toda o percurso, indicando a presença de deformações no mapa. Além disso, em ambos os casos, pode-se observar a presença de ruídos nos mapas. Contudo, ainda assim é possível perceber a cor e a textura da caverna, além da presença de alguns pontos luminosos.

Para uma análise quantitativa dos resultados de mapeamento, a Figura 45 apresenta os mapas obtidos por cada método, com a diferença entre eles e a referência indicada de forma colorida, assim como o seu respectivo histograma, de acordo com a métrica \mathbf{m}_m . Os valores relativos a essa comparação são apresentados na Tabela 14.

Com a comparação, é possível notar que o *RTAB-Map* apresentou resultados mais precisos na parte de mapeamento para esse ambiente, com uma reconstrução relativamente próxima a do *LiDAR-SLAM*. Embora o método tenha apresentado uma localização menos precisa, os processos de otimização do *RTAB-Map* foram fundamentais nesse cenário, reduzindo as deformações na reconstrução final. O método obteve um mapa com quase 500 mil pontos, onde 68% tiveram uma diferença menor que 0,05m, 95% menores que 0,11m e 99,7% menores que 0,25m. Por outro lado, o *Registro-EKF* apresentou um mapa com quase 1 milhão de pontos e uma diferença média mais de duas vezes maior, com valores inferiores a 0,10m, 0,29m e 0,49m para 68%, 95% e 99,7% dos pontos.

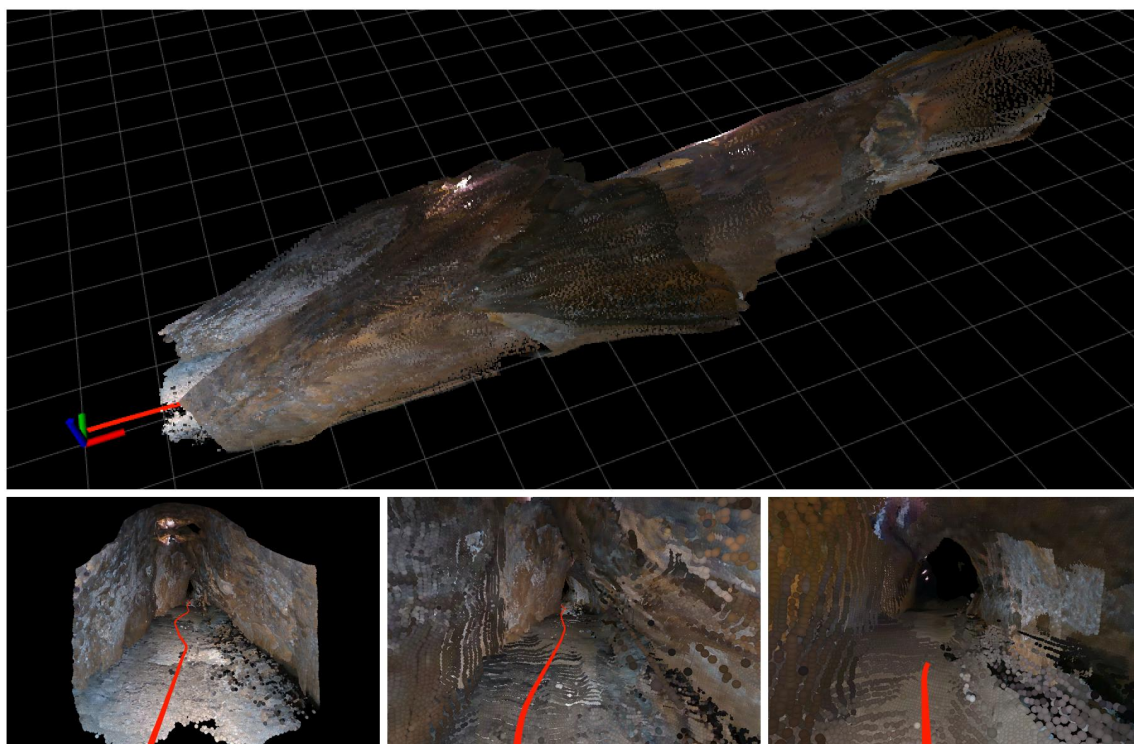


Figura 43 – Mapa obtido pelo *Registro-EKF* no experimento teleoperado na Mina do Veloso (vista em perspectiva acima e três vistas internas abaixo).

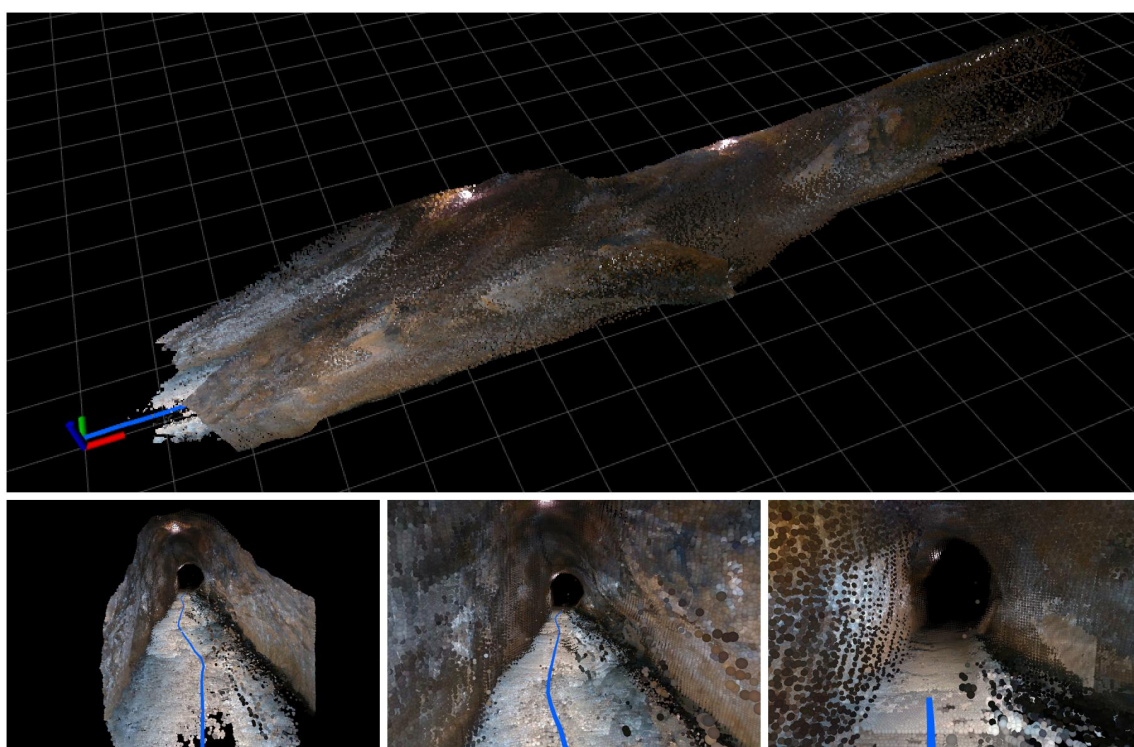


Figura 44 – Mapa obtido pelo *RTAB-Map* no experimento teleoperado na Mina do Veloso (vista em perspectiva acima e três vistas internas abaixo).

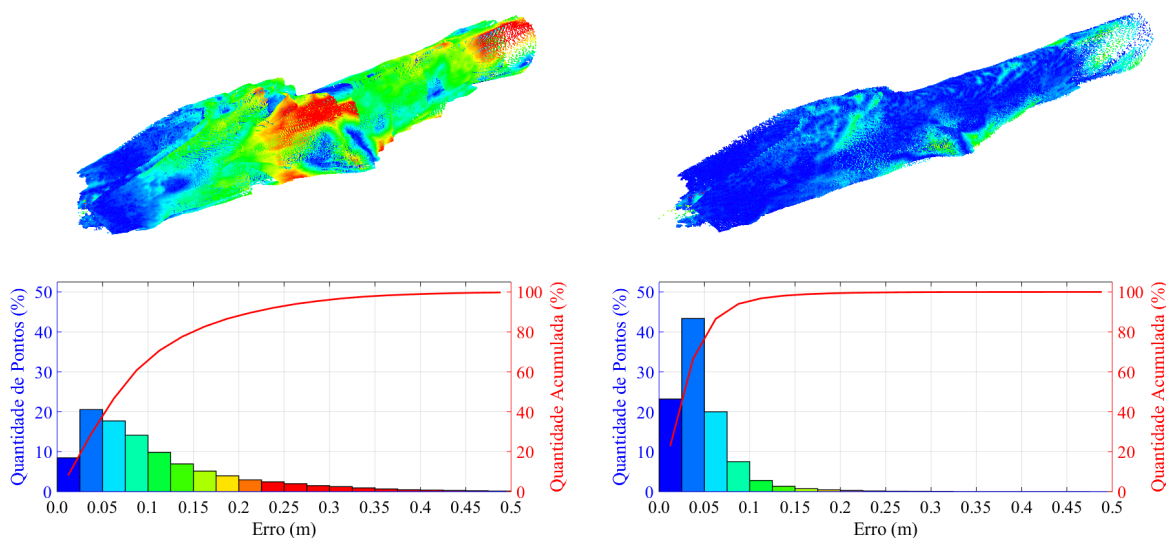


Figura 45 – Comparação dos mapas obtido pelo *Registro-EKF* (à esquerda) e pelo *RTAB-Map* (à direita) com o *LiDAR-SLAM* no experimento teleoperado na Mina do Veloso. Acima é representada diferença entre os pontos do mapa de forma colorida e, abaixo, o histograma dessa diferença na mesma referência de cores.

Tabela 14 – Diferença média e intervalos confiança relativos aos pontos da nuvem obtida pelos métodos no experimento teleoperado na Mina do Veloso.

Métrica m_m	Nº de pontos	Erro médio	1σ (68%)	2σ (95%)	3σ (99.7%)
<i>Registro-EKF</i>	951.603	$0,11 \pm 0,09m$	$\leq 0,10m$	$\leq 0,29m$	$\leq 0,49m$
<i>RTAB-Map</i>	447.809	$0,05 \pm 0,03m$	$\leq 0,05m$	$\leq 0,11m$	$\leq 0,25m$

Por fim, as Tabelas 15 e 16 apresentam as informações relacionadas ao tempo de processamento dos métodos, de acordo com a métrica m_t . A primeira demonstra os tempos gastos no *EKF* e do processamento do mapa para o *Registro-EKF*, enquanto que a segunda apresenta os tempos de processamento referentes às partes de odometria e de mapeamento do *RTAB-Map*.

Devido ao tamanho reduzido desse experimento, conforme explicado na Subseção 5.3.2, a resolução da imagem utilizada no *Registro-EKF* foi maior em comparação aos experimentos anteriores. Esse fator, somado à proximidade entre as paredes da caverna, provocou um aumento considerável o tamanho da nuvem local, obtendo uma média de 115280 ± 6710 de pontos. Dessa forma, o tempo de processamento do mapa foi muito superior ao último experimento, cerca de oito vezes maior, enquanto que o *EKF* apresentou novamente um tempo de execução muito rápido, similar às simulações. Porém, o tempo total do método ainda foi menor que o tempo do experimento indicando a possibilidade de ser utilizado *online*. Para o *RTAB-Map*, o tempo de processamento também foi similar aos resultados de simulação, principalmente na etapa de localização. Por outro lado, o número médio de *inliers* neste experimento foi de 374 ± 94 , consideravelmente maior que

Tabela 15 – Informações relacionadas ao tempo de processamento da odometria e do mapeamento do *Registro-EKF* no experimento teleoperado na Mina du Veloso.

Número total de mensagens de odometria	264
Número total de mensagens de mapeamento	27
Tempo da odometria por mensagem	$20 \pm 7 \mu\text{s}$
Tempo do mapeamento por mensagem	$1910 \pm 444 \text{ ms}$
Métrica m_t (tempo total)	51,6 s

Tabela 16 – Informações relacionadas ao tempo de processamento da odometria e do mapeamento do *RTAB-Map* no experimento teleoperado na Mina du Veloso.

Número total de mensagens de odometria	524
Número total de mensagens de mapeamento	59
Tempo da odometria por mensagem	$53 \pm 11 \text{ ms}$
Tempo do mapeamento por mensagem	$115 \pm 51 \text{ ms}$
Métrica m_t (tempo total)	34,6 s

os experimentos anteriores, indicando que mais pontos estavam no alcance da câmera. Porém, a instabilidade no recebimento das imagens na câmera causou uma maior variação no número de *inliers*, conforme apresentado pelo alto valor de desvio-padrão. Contudo, comparando o tempo total entre os métodos, o *RTAB-Map* foi cerca de 33% mais rápido que o *Registro-EKF*.

6.5 Resultados do Experimento Semiautônomo na Mina du Veloso

A seguir, estão apresentados os resultados do experimento semiautônomo na Mina du Veloso. A Figura 46 demonstra os resultados de localização nos planos xy , xz e yz obtidos pelo *Registro-EKF* e pelo *RTAB-Map*, além da localização de referência dada pelo *LiDAR-SLAM*. A Figura 47 exibe e os gráficos da diferença dos métodos em comparação com a referência ao longo do tempo. Utilizando os valores calculados nas métricas m_l e m_f , as Tabelas 17 e 18.

Tabela 17 – Diferença média entre a localização dos métodos e a referência no experimento semiautônomo na Mina du Veloso.

Métrica m_l	x	y	z	ϕ_x	θ_y	ψ_z
<i>Registro-EKF</i>	0,620m	0,739m	0,096m	$1,623^\circ$	$1,223^\circ$	$4,347^\circ$
<i>RTAB-Map</i>	0,487m	1,002m	0,100m	$1,828^\circ$	$1,359^\circ$	$4,824^\circ$

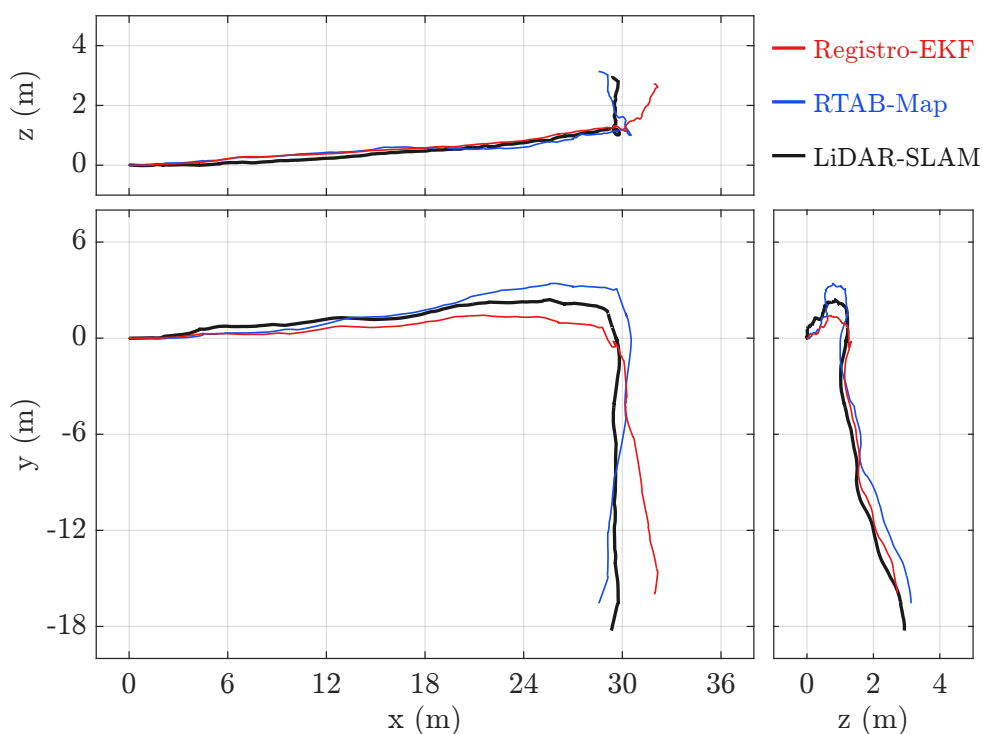


Figura 46 – Resultados de localização nos planos xy , xz e yz no experimento semi-autônomo na Mina do Veloso.

Tabela 18 – Diferença entre a posição final dos métodos e a referência no experimento semiautônomo na Mina do Veloso.

Métrica m_f	Valor absoluto	Valor relativo
<i>Registro-EKF</i>	3,47m	6,7%
<i>RTAB-Map</i>	1,88m	3,6%

Nesse experimento, é novamente possível observar os desafios apresentados pelos ambientes reais, onde a irregularidade do terreno ocasionou em maior escorregamento das rodas, exigindo movimentos mais bruscos com o robô. Com uma distância percorrida cerca de cinco vezes maior que a anterior, a diferença entre a localização dos métodos e a referência foi consideravelmente maior, principalmente para o *Registro-EKF*, com mais de 3m de diferença na posição final, contra pouco menos de 2m para o *RTAB-Map*. Neste último, os principais problemas foram relacionados aos movimentos mais bruscos de giro realizados pelo robô, que provocaram efeitos de desfoque (*motion blur*) nas imagens devido à baixa luminosidade do ambiente, comprometendo o número momentâneo de *inliers* e a qualidade dos resultados. Para o *Registro-EKF*, o escorregamento das rodas foi o principal problema, em especial na curva de 90° realizada em 140s, onde o erro ψ ocasionou um crescente erro de posição em x e y . Durante todo o percurso, ambos os métodos apresentaram erros máximos de posição de cerca de 2m em x e y (0,2m para z) e de orientação de 5° em ϕ e θ e 10° em ψ .

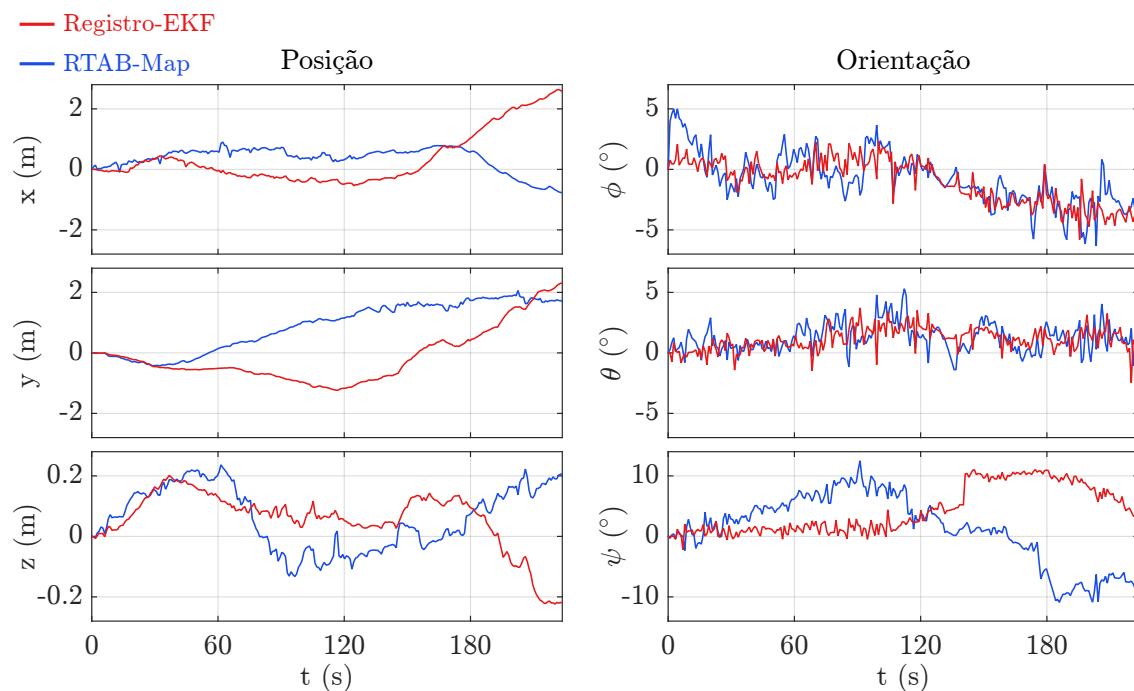


Figura 47 – Diferença de posição e orientação dos métodos em comparação à referência no experimento semiautônomo na Mina do Veloso.

As Figuras 48 e 49 apresentam os resultados de mapeamento do *Registro-EKF* e do *RTAB-Map* respectivamente, com uma vista em perspectiva acima e três vistas interna abaixo. É possível notar que o ambiente utilizado é mais escuro e possui com um teto mais alto, dificultando a visualização de toda a caverna nos resultados de reconstrução. Ambos os resultados apresentaram um nível relevante de ruído, onde é possível observar a diferença de intensidade de brilho das paredes à medida que o robô se aproxima delas. Entretanto, pode-se perceber em ambos os mapas a cor, a textura e o formato da caverna, principalmente o chão e as paredes. Nesse caso, o *RTAB-Map* apresentou uma reconstrução com menos distorções, porém um pouco menos densa que o *Registro-EKF*.

Na sequência, é realizada uma análise quantitativa dos resultados de mapeamento, apresentando a diferença entre os mapas dos métodos e a referência do *LiDAR-SLAM*, de acordo com a métrica m_m . A Figura 50 exibe essa diferença em forma de um mapa de pontos coloridos, além de seu histograma, e a Tabela 19 demonstra os valores estatísticos da diferença. Observando a comparação dos mapas, é possível notar que o *RTAB-Map* novamente obteve melhores resultados de mapeamento, apresentando um mapa mais próximo ao do *LiDAR-SLAM*. O método obteve uma reconstrução com 1,6 milhões de pontos, onde 68% tiveram uma diferença inferior a 0,17m, 95% inferior a 0,40m e 99,7% inferior a 0,69m. Por sua vez, o *Registro-EKF* novamente apresentou uma diferença cerca de duas vezes maior que o *RTAB-Map*, obtendo um mapa com 2,3 milhões de pontos, sendo 68% menor que 0,26m de diferença, 95% menor que 1,18m e 99,7% menor que 2,36m.

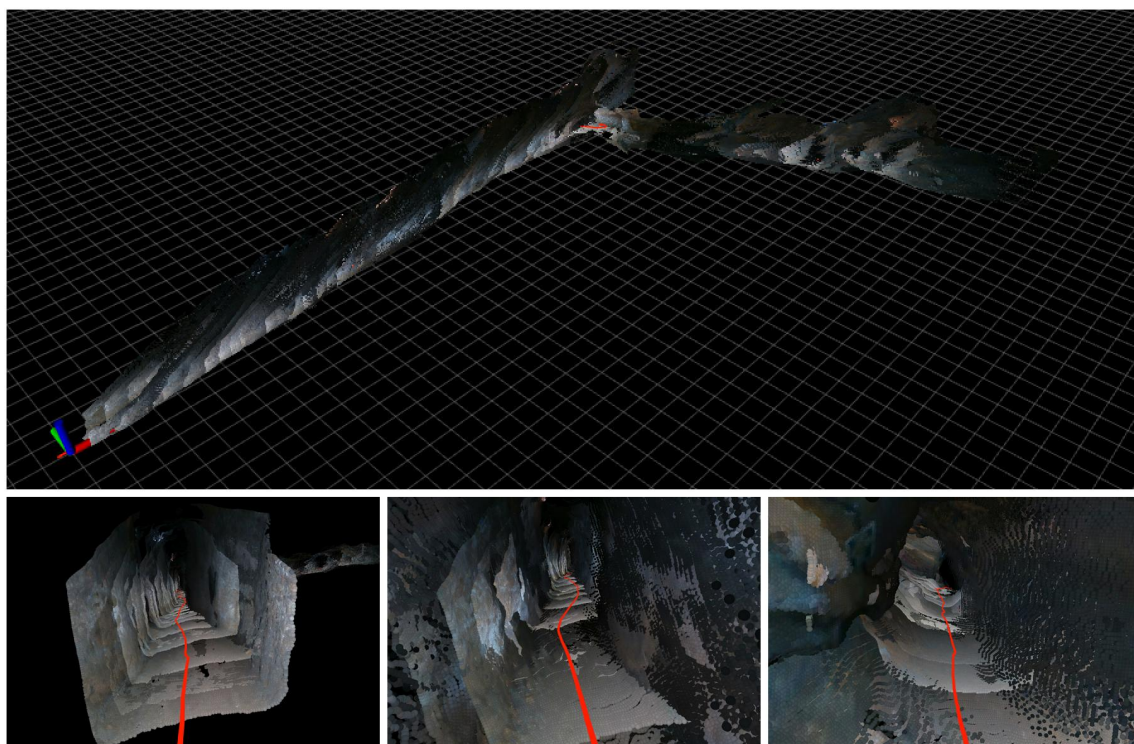


Figura 48 – Mapa obtido pelo *Registro-EKF* no experimento semiautônomo na Mina do Veloso (vista em perspectiva acima e três vistas internas abaixo).

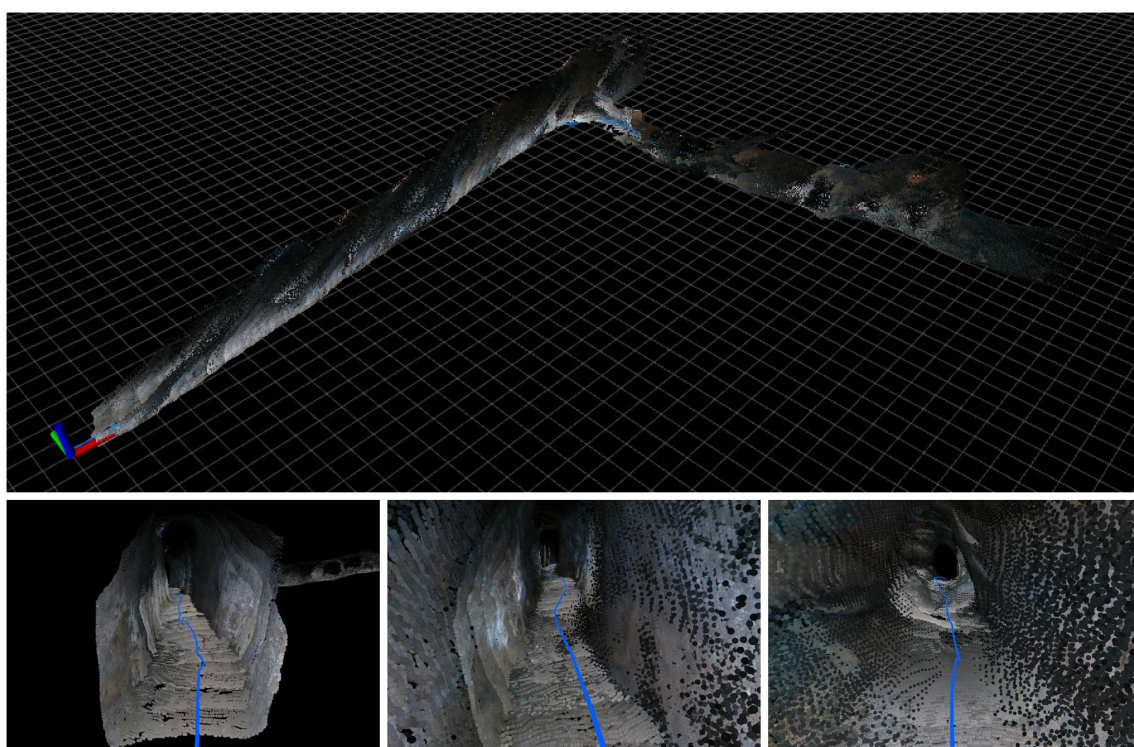


Figura 49 – Mapa obtido pelo *RTAB-Map* no experimento semiautônomo na Mina do Veloso (vista em perspectiva acima e três vistas internas abaixo).

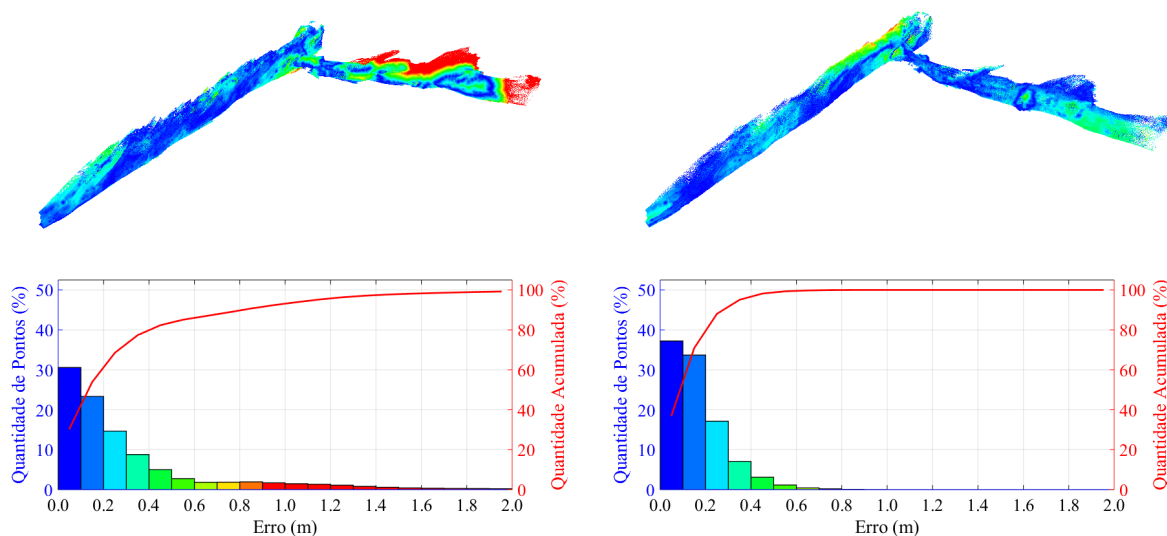


Figura 50 – Comparação dos mapas obtido pelo *Registro-EKF* (à esquerda) e pelo *RTAB-Map* (à direita) com o *LiDAR-SLAM* no experimento semiautônomo na Mina du Veloso. Acima é representada diferença entre os pontos do mapa de forma colorida e, abaixo, o histograma dessa diferença na mesma referência de cores.

Tabela 19 – Diferença média e intervalos confiança relativos aos pontos da nuvem obtida pelos métodos no experimento semiautônomo na Mina du Veloso.

Métrica m_m	Nº de pontos	Erro médio	1σ (68%)	2σ (95%)	3σ (99.7%)
<i>Registro-EKF</i>	2.325.811	$0,32 \pm 0,39m$	$\leq 0,26m$	$\leq 1,18m$	$\leq 2,36m$
<i>RTAB-Map</i>	1.609.283	$0,16 \pm 0,12m$	$\leq 0,17m$	$\leq 0,40m$	$\leq 0,69m$

Por último, de acordo com a métrica m_t , as Tabelas 20 e 21 apresentam as informações relacionadas ao tempo de processamento de odometria e de mapeamento do *Registro-EKF* e do *RTAB-Map*, respectivamente. Novamente devido ao espaço estreito, o número de pontos dentro do alcance da câmera foi maior nesse ambiente em relação aos experimentos no simulador. A nuvem local apresentou uma média de 47430 ± 7631 pontos (cerca de 3x mais que nas simulações), o que contribuiu no aumento do tempo de processamento do mapa no *Registro-EKF*. Já a odometria realizada com o *EKF* gastou novamente um tempo de execução praticamente insignificante, com uma média de $21\mu s$

Tabela 20 – Informações relacionadas ao tempo de processamento da odometria e do mapeamento do *Registro-EKF* no experimento semiautônomo na Mina du Veloso.

Número total de mensagens de odometria	1908
Número total de mensagens de mapeamento	90
Tempo da odometria por mensagem	$20 \pm 7 \mu s$
Tempo do mapeamento por mensagem	$686 \pm 162 ms$
Métrica m_t (tempo total)	61,8 s

Tabela 21 – Informações relacionadas ao tempo de processamento da odometria e do mapeamento do *RTAB-Map* no experimento semiautônomo na Mina do Veloso.

Número total de mensagens de odometria	3103
Número total de mensagens de mapeamento	231
Tempo da odometria por mensagem	50 ± 28 ms
Tempo do mapeamento por mensagem	168 ± 168 ms
Métrica m_t (tempo total)	194,0 s

por mensagem. O *RTAB-Map* apresentou tempos de processamento para a localização e mapeamento similares aos experimentos anteriores, porém, obteve um menor número de *inliers* por *frame*, com uma média de 224 ± 81 . Isso ocorreu devido ao efeitos de desfoque, causados pelos movimentos bruscos do robô, o que também contribuiu com o alto valor de desvio-padrão. O tempo total gasto por esse método foi cerca de 3 vezes maior que o do *Registro-EKF*, porém, com um valor ainda abaixo do tempo total de experimento, reforçando a possibilidade de ser utilizado de forma *online*.

Capítulo 7

Conclusões

Esta dissertação apresentou, de forma comparativa, dois métodos de reconstrução visual aplicados em ambientes confinados. O primeiro método corresponde ao registro da nuvem de pontos integrado com um [EKF](#) para a parte de localização, enquanto que o segundo se refere ao processo de [SLAM](#) com o [RTAB-Map](#). Utilizando dados de câmeras [RGB-D](#), ambos os métodos foram capazes de gerar reconstruções densas, coloridas e relativamente precisas dos ambientes analisados, alcançando, assim, o objetivo proposto na [Seção 1.2](#).

Primeiramente, para a parte de localização nos ambientes simulados, os erros associados a ambos os métodos foram consideravelmente baixos, apresentando para a posição final um valor médio de apenas 0,5%. Dessa forma, é possível afirmar que as reconstruções geradas pelos métodos apresentaram baixa distorção em relação ao esperado, algo que é comprovado pela análise comparativa dos mapas. Nesse caso, o erro foi menor que 65 centímetros para 95% dos pontos no primeiro experimento (um percurso de 440m), enquanto que, para o segundo (de quase 150m), foi menor que 34 centímetros. Nesse contexto, os métodos apresentaram resultados bem próximos entre si, porém, o [RTAB-Map](#) foi capaz de gerar uma reconstrução com menores erros e ligeiramente mais homogênea.

Nos experimentos em ambientes reais, os resultados foram satisfatórios, porém, com erros e deformações maiores que os apresentados nas simulações. Nesse cenário, não foi possível a utilização de um *ground truth* e a comparação foi realizada com o método de [LiDAR SLAM](#) utilizado no EspeleoRobô. Para os resultados de localização, os métodos apresentaram uma proximidade razoável entre si, com uma diferença média na posição final de 3,5%. A maior disparidade entre os métodos ocorreu na reconstrução de ambientes reais, onde o [RTAB-Map](#) mostrou-se mais adequado para uso, apresentando um mapa consideravelmente mais próximo ao [LiDAR SLAM](#) em ambos os experimentos. Para os respectivos percursos de 10m e 50m, o método apresentou uma diferença menor que 11 e 40 centímetros para 95% dos pontos, enquanto que o registro de pontos obteve uma diferença quase três vezes maior, com 29 centímetros e 1,18 metros.

Os principais problemas observados na geração dos resultados estiveram relacionados à qualidade dos dados obtidos dos sensores. Para o caso dos experimentos realizados em simulador, não houve presença de ruídos e de deriva nas informações dos sensores, o que viabilizou a geração de resultados muito próximos ao valor de referência. Por outro lado, os principais obstáculos enfrentados nos experimentos reais estiveram relacionados às informações obtidas com a câmera RGB-D. Nesse caso, a baixa iluminação dos ambientes contribuiu nos efeitos de desfoque (*motion blur*) ocasionados em movimentos bruscos do robô e, inclusive ocasionou uma variação na intensidade das cores à medida que o robô se deslocava, como é possível observar nos mapas gerados. Além disso, os valores de profundidade mostraram-se ruidosos para grandes distâncias, visto que o alcance ideal da câmera é de apenas 3 metros obtidos. De maneira complementar, os ruídos e a deriva da IMU e o escorregamento imprevisível das rodas comprometeram a qualidade dos resultados de localização do registro de pontos, implicando em deformações na reconstrução final.

Contudo, as reconstruções obtidas com os ambos os métodos na Mina do Veloso permitem observar alguns detalhes específicos do ambiente de forma densa e colorida. Caso houvesse algum ponto de desmoronamento, a presença de algum objeto ou inclusive de pessoas no local, seria possível visualizar por meio dos mapas gerados de uma forma relativamente detalhada. Assim, dado o objetivo principal da dissertação, pode-se afirmar que alguns procedimentos de inspeções e análises estruturais podem ser realizadas de forma limitada por especialistas em espeleologia, utilizando modelos digitais e realidade virtual.

7.1 Trabalhos Futuros

Dentre os problemas apontados anteriormente, uma possível forma de avaliar o desempenho dos métodos em condições de baixa luminosidade (ou inclusive variável) de modo mais controlado consiste na utilização do simulador. Foram realizados alguns testes iniciais utilizando objetos de luz direcionada, para atuar como os LEDs do robô real, porém, não foi abordado nesta dissertação. Outro problema citado é a presença de ruído na profundidade obtida com a câmera. Nesse caso, alguns métodos de filtragem, tanto espacial, quanto temporal, poderiam ser testados, de forma a obter uma imagem mais adequada para a aplicação dos métodos.

Na tentativa de obter resultados mais precisos, podem ser realizadas algumas modificações nos métodos utilizados, bem como a investigação de novas técnicas de mapeamento. Para o caso do EKF, novas estratégias nas etapas de predição e correção poderiam ser investigadas com o objetivo de obter um modelo mais adequado para o robô utilizado. No registro da nuvem de pontos, é válida a realização de testes com alguns algoritmos para

o casamento entre as nuvens, visto que o método foi capaz de gerar os resultados em tempo reduzido. Para o [RTAB-Map](#), a utilização de outros tipos de detectores e descritores pode ser analisada, assim como um estudo maior dos resultados com a abordagem [F2M](#). Nesse caso, seria interessante a investigação de outras técnicas de [SLAM](#), como o *ElasticFusion* e o *KinectFusion*, capazes de gerar uma nuvem de pontos densa e colorida por meio de câmeras [RGB-D](#). Além disso, a fusão dos dados utilizados com informações de sensores [LiDAR](#) representa um ponto importante a ser analisado, com o objetivo de aprimorar o mapa final utilizando dados de profundidade mais precisos. Outro ponto possível de ser estudado consiste na inclusão de ruídos nos dados das câmeras e da [IMU](#) nas simulações de forma a obter uma melhor representação dos experimentos em ambientes reais.

Outra possível forma de variar os resultados obtidos consiste na alteração da execução dos experimentos e da exibição das reconstruções. Para ambientes mais amplos, como a Caverna do [DARPA](#) apresentada, o mapeamento por meio de câmeras ocorre de maneira restrita, visto que esses sensores possuem campo de visão limitado. Nesse caso, foi inclusive verificada a realização de novos modos de exploração do ambiente, porém, não foram gerados resultados nesta dissertação. Além disso, alguns testes outros tipos de ambientes, como dutos e galerias, usualmente utilizados com o *EspeleoRobô*, então sendo realizados. Por fim, outro ponto que começou a ser analisado corresponde à conversão das reconstruções para *meshes*, porém, sem resultados para a presente dissertação.

Referências

- Agência Nacional de Mineração. *Anuário Mineral Brasileiro: Principais Substâncias Metálicas*. Brasília, 2019.
- K. Alexis. Resilient Autonomous Exploration and Mapping of Underground Mines using Aerial Robots. In *2019 19th International Conference on Advanced Robotics (ICAR)*, pages 1–8. IEEE, dec 2019.
- 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 Inspeção de Ambientes Confinados - EspeleoRobô. In *Anais do Congresso Brasileiro de Automática 2020*, dec 2020.
- M. O. A. Aqel, M. H. Marhaban, M. I. Saripan, and N. B. Ismail. Review of visual odometry: types, approaches, challenges, and applications. *SpringerPlus*, 5(1):1897, dec 2016.
- H. Azpúrua, 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 International Conference on Advanced Robotics (ICAR)*, pages 17–23. IEEE, dec 2019.
- H. Azpúrua, A. Rezende, G. Potje, G. P. d. C. Júnior, R. Fernandes, V. Miranda, L. W. d. R. Filho, J. Domingues, F. Rocha, F. L. M. de Sousa, L. G. D. de Barros, E. R. Nascimento, D. G. Macharet, G. Pessin, and G. M. Freitas. Towards Semi-autonomous Robotic Inspection and Mapping in Confined Spaces with the EspeleoRobô. *Journal of Intelligent and Robotic Systems*, 101(4):69, apr 2021.
- B. Balaram, T. Canham, C. Duncan, H. F. Grip, W. Johnson, J. Maki, A. Quon, R. Stern, and D. Zhu. Mars Helicopter Technology Demonstrator. In *2018 AIAA Atmospheric Flight Mechanics Conference*, Reston, Virginia, jan 2018. American Institute of Aeronautics and Astronautics.
- B. Barshan and H. F. Durrant-Whyte. Inertial navigation systems for mobile robots. *IEEE Transactions on Robotics and Automation*, 11(3):328–342, jun 1995.

- K. Berntorp. Particle filter for combined wheel-slip and vehicle-motion estimation. In *2015 American Control Conference (ACC)*, pages 5414–5419. IEEE, jul 2015.
- A. K. Bhowmik. Sensification of computing: adding natural sensing and perception capabilities to machines. *APSIPA Transactions on Signal and Information Processing*, 6, jan 2017.
- G. Bleedt, M. J. Powell, B. Katz, J. Di Carlo, P. M. Wensing, and S. Kim. MIT Cheetah 3: Design and Control of a Robust, Dynamic Quadruped Robot. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2245–2252. IEEE, oct 2018.
- 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, sep 2015.
- L. S. V. Boas and A. G. S. Conceição. Modelagem Cinemática de Robôs Móveis da Classe Skid-Steer. In *Anais do Congresso Brasileiro de Automática 2020*, dec 2020.
- G. Bradski and A. Kaehler. *Learning OpenCV: Computer vision with the OpenCV library*. O’Reilly Media, Inc., Sebastopol, 2008.
- W. Burgard, M. Hebert, and M. Bennewitz. World Modeling. In B. Siciliano and O. Khatib, editors, *Springer Handbook of Robotics*, chapter 45, pages 1135–1152. Springer International Publishing, Cham, 2016.
- M. Calonder, V. Lepetit, C. Strecha, and P. Fua. BRIEF: Binary Robust Independent Elementary Features. In *European Conference on Computer Vision*, volume 6314, pages 778–792. Springer, 2010.
- M. Carreras, J. D. Hernandez, E. Vidal, N. Palomeras, D. Ribas, and P. Ridao. Sparus II AUV-A Hovering Vehicle for Seabed Inspection. *IEEE Journal of Oceanic Engineering*, 43(2):344–355, apr 2018.
- Y. Cheng, M. Maimone, and L. Matthies. Visual Odometry on the Mars Exploration Rovers. In *2005 IEEE International Conference on Systems, Man and Cybernetics*, pages 903–910. IEEE, 2005.
- H. Choset, K. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. Kavraki, and S. Thrun. *Principles of Robot Motion: Theory, Algorithms, and Implementations*. MIT press, Cambridge, 2005.
- W. Chung and K. Iagnemma. Wheeled Robots. In B. Siciliano and O. Khatib, editors, *Springer Handbook of Robotics*, chapter 24, pages 575–594. Springer International Publishing, Cham, 2016.

- A. Cid, M. Nazario, M. Sathler, F. Martins, J. Domingues, M. Delunardo, P. Alves, R. Teotonio, L. G. Barros, A. Rezende, V. Miranda, G. Freitas, G. Pessin, and H. Azpúrua. 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, nov 2020.
- F. Cordes, F. Kirchner, and A. Babu. Design and field testing of a rover with an actively articulated suspension system in a Mars analog terrain. *Journal of Field Robotics*, 35(7):1149–1181, oct 2018.
- I. Cvisic and I. Petrovic. Stereo odometry based on careful feature selection and tracking. In *2015 European Conference on Mobile Robots (ECMR)*, pages 1–6. IEEE, sep 2015.
- G. P. da 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.
- Z. Dong, B. Yang, F. Liang, R. Huang, and S. Scherer. Hierarchical registration of unordered TLS point clouds based on binary shape context descriptor. *ISPRS Journal of Photogrammetry and Remote Sensing*, 144:61–79, oct 2018.
- Z. Dong, F. Liang, B. Yang, Y. Xu, Y. Zang, J. Li, Y. Wang, W. Dai, H. Fan, J. Hyypä, and U. Stilla. Registration of large-scale terrestrial laser scanner point clouds: A review and benchmark. *ISPRS Journal of Photogrammetry and Remote Sensing*, 163:327–342, may 2020.
- G. Dudek and M. Jenkin. Pose Maintenance and Localization. In *Computational Principles of Mobile Robotics*, chapter 8, pages 240–275. Cambridge University Press, Cambridge, 2010.
- G. Dudek and M. Jenkin. Inertial Sensing, GPS and Odometry. In B. Siciliano and O. Khatib, editors, *Springer Handbook of Robotics*, chapter 29, pages 737–752. Springer International Publishing, Cham, 2016.
- H. Durrant-Whyte and T. Bailey. Simultaneous localization and mapping: part I. *IEEE Robotics and Automation Magazine*, 13(2):99–110, jun 2006.
- H. Durrant-Whyte and T. C. Henderson. Multisensor Data Fusion. In B. Siciliano and O. Khatib, editors, *Springer Handbook of Robotics*, chapter 35, pages 867–896. Springer International Publishing, Cham, 2016.

- K. Ebadi, Y. Chang, M. Palieri, A. Stephens, A. Hatteland, E. Heiden, A. Thakur, N. Funabiki, B. Morrell, S. Wood, L. Carlone, and A.-a. Agha-mohammadi. 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, may 2020.
- J. Engel, T. Schöps, and D. Cremers. LSD-SLAM: Large-Scale Direct Monocular SLAM. In *European Conference on Computer Vision (ECCV 2014)*, pages 834–849, Cham, 2014. Springer.
- R. Fernandes, T. L. Rocha, H. Azpurua, 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, nov 2020.
- M. Filipenko and I. Afanasyev. Comparison of Various SLAM Systems for Mobile Robot in an Indoor Environment. In *2018 International Conference on Intelligent Systems (IS)*, pages 400–407. IEEE, sep 2018.
- C. Forster, M. Pizzoli, and D. Scaramuzza. SVO: Fast semi-direct monocular visual odometry. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 15–22. IEEE, may 2014.
- J. Fuentes-Pacheco, J. Ruiz-Ascencio, and J. M. Rendón-Mancha. Visual simultaneous localization and mapping: a survey. *Artificial Intelligence Review*, 43(1):55–81, jan 2015.
- A. Geiger, J. Ziegler, and C. Stiller. StereoScan: Dense 3d reconstruction in real-time. In *2011 IEEE Intelligent Vehicles Symposium (IV)*, pages 963–968. IEEE, jun 2011.
- R. C. Gonzalez, R. E. Woods, and S. L. Eddins. *Digital Image Processing Using MATLAB*. Gatesmark Publishing, 2 edition, 2009.
- M. Halber and T. Funkhouser. Fine-to-Coarse Global Registration of RGB-D Scans. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6660–6669. IEEE, jul 2017.
- P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox. RGB-D Mapping: Using Depth Cameras for Dense 3D Modeling of Indoor Environments. In *Experimental Robotics*, pages 477–491. Springer, Berlin, Heidelberg, 2014.
- B. Huang, J. Zhao, and J. Liu. A Survey of Simultaneous Localization and Mapping with an Envision in 6G Wireless Networks, 2019.

- G. Huang. Visual-Inertial Navigation: A Concise Review. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 9572–9582. IEEE, may 2019.
- M. Hutter, C. Gehring, D. Jud, A. Lauber, C. D. Bellicoso, V. Tsounis, J. Hwangbo, K. Bodie, P. Fankhauser, M. Bloesch, R. Diethelm, S. Bachmann, A. Melzer, and M. Hoepflinger. ANYmal - a highly mobile and dynamic quadrupedal robot. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 38–44. IEEE, oct 2016.
- I. Z. Ibragimov and I. M. Afanasyev. Comparison of ROS-based visual SLAM methods in homogeneous indoor environment. In *2017 14th Workshop on Positioning, Navigation and Communications, WPNC 2017*, pages 1–6. IEEE, jan 2018.
- M. Jaud, S. Bertin, M. Beauverger, E. Augereau, and C. Delacourt. RTK GNSS-Assisted Terrestrial SfM Photogrammetry without GCP: Application to Coastal Morphodynamics Monitoring. *Remote Sensing*, 12(11):1889, jun 2020.
- Jianbo Shi and Tomasi. Good features to track. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition CVPR-94*, pages 593–600. IEEE Comput. Soc. Press, 1994.
- A. Kalantari, T. Touma, L. Kim, R. Jitosh, K. Strickland, B. T. Lopez, and A.-A. Agha-Mohammadi. Drivocopter: A concept Hybrid Aerial/Ground vehicle for long-endurance mobility. In *2020 IEEE Aerospace Conference*, pages 1–10. IEEE, mar 2020.
- N. Kameyama and K. Hidaka. A sensor-based exploration algorithm for autonomous map generation on mobile robot using kinect. In *2017 Asian Control Conference, ASCC 2017*, pages 459–464. IEEE, jan 2018.
- B. Khaleghi, A. Khamis, F. O. Karray, and S. N. Razavi. Multisensor data fusion: A review of the state-of-the-art. *Information Fusion*, 14(1):28–44, jan 2013.
- K. Koide, J. Miura, and E. Menegatti. A portable three-dimensional LIDAR-based system for long-term and wide-area people behavior measurement. *International Journal of Advanced Robotic Systems*, 16(2):172988141984153, mar 2019.
- R. Kummerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard. G2o: A general framework for graph optimization. In *2011 IEEE International Conference on Robotics and Automation*, pages 3607–3613. IEEE, may 2011.
- M. Labbe and F. Michaud. Appearance-Based Loop Closure Detection for Online Large-Scale and Long-Term Operation. *IEEE Transactions on Robotics*, 29(3):734–745, jun 2013.

- M. Labbé and F. Michaud. Long-term online multi-session graph-based SPLAM with memory management. *Autonomous Robots*, 42(6):1133–1150, aug 2018.
- M. Labbé and F. Michaud. RTAB-Map as an open-source lidar and visual simultaneous localization and mapping library for large-scale and long-term online operation. *Journal of Field Robotics*, 36(2):416–446, mar 2019.
- S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale. Keyframe-based visual-inertial odometry using nonlinear optimization. *The International Journal of Robotics Research*, 34(3):314–334, mar 2015.
- C. Li, Y. Liu, F. Yan, and Y. Zhuang. Deep Point Cloud Odometry: A Deep Learning Based Odometry with 3D Laser Point Clouds. In *International Symposium on Neural Networks*, pages 154–163. Springer, 2020a.
- Y. Li, M. Li, H. Zhu, E. Hu, C. Tang, P. Li, and S. You. Development and applications of rescue robots for explosion accidents in coal mines. *Journal of Field Robotics*, 37(3): 466–489, apr 2020b.
- R. Losch, S. Grehl, M. Donner, C. Buhl, and B. Jung. Design of an Autonomous Robot for Mapping, Navigation, and Manipulation in Underground Mines. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1407–1412. IEEE, oct 2018.
- M. Magnusson, A. Lilienthal, and T. Duckett. Scan registration for autonomous mining vehicles using 3D-NDT. *Journal of Field Robotics*, 24(10):803–827, oct 2007.
- A. Maity, S. Majumder, and D. N. Ray. Amphibian subterranean robot for mine exploration. In *2013 International Conference on Robotics, Biomimetics, Intelligent Computational Systems*, pages 242–246. IEEE, nov 2013.
- 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 International Conference on Intelligent Robots and Systems*, pages 1222–1227. IEEE, oct 2007.
- A. Martins, J. Almeida, C. Almeida, A. Dias, N. Dias, J. Aaltonen, A. Heininen, K. T. Koskinen, C. Rossi, S. Dominguez, C. Voros, S. Henley, M. McLoughlin, H. van Moerkerk, J. Tweedie, B. Bodo, N. Zajzon, and E. Silva. UX 1 system design - A robotic system for underwater mining exploration. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1494–1500. IEEE, oct 2018.
- J. Martz, W. Al-Sabban, and R. N. Smith. Survey of unmanned subterranean exploration, navigation, and localisation. *IET Cyber-Systems and Robotics*, 2(1):1–13, mar 2020.

- Ministério do Trabalho e Emprego. *NR-33 Segurança e Saúde nos Trabalhos em Espaços Confinados*. Brasília, 2006.
- S. A. S. Mohamed, M.-H. Haghbayan, T. Westerlund, J. Heikkonen, H. Tenhunen, and J. Plosila. A Survey on Odometry for Autonomous Navigation Systems. *IEEE Access*, 7:97466–97486, 2019.
- V. Morell-Gimenez, M. Saval-Calvo, V. Villena-Martinez, J. Azorin-Lopez, J. Garcia-Rodriguez, M. Cazorla, S. Orts-Escolano, and A. Fuster-Guillo. A Survey of 3D Rigid Registration Methods for RGB-D Cameras. In J. Garcia-Rodriguez, editor, *Advancements in Computer Vision and Image Processing*, pages 74–98. IGI Global, 2018.
- A. C. Morris, D. Silver, D. Ferguson, and S. Thayer. Towards Topological Exploration of Abandoned Mines. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2117–2123. IEEE, 2005.
- P. Moulon, P. Monasse, and R. Marlet. Global Fusion of Relative Motions for Robust, Accurate and Scalable Structure from Motion. In *2013 IEEE International Conference on Computer Vision*, pages 3248–3255. IEEE, dec 2013.
- A. I. Mourikis and S. I. Roumeliotis. A Multi-State Constraint Kalman Filter for Vision-aided Inertial Navigation. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pages 3565–3572. IEEE, apr 2007.
- P. Muller and A. Savakis. Flowdometry: An Optical Flow and Deep Learning Based Approach to Visual Odometry. In *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 624–631. IEEE, mar 2017.
- R. Mur-Artal and J. D. Tardos. ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras. *IEEE Transactions on Robotics*, 33(5):1255–1262, oct 2017.
- P. Nazemzadeh, D. Fontanelli, D. Macii, and L. Palopoli. Indoor Localization of Mobile Robots Through QR Code Detection and Dead Reckoning Data Fusion. *IEEE/ASME Transactions on Mechatronics*, 22(6):2588–2599, dec 2017.
- D. Nister, O. Naroditsky, and J. Bergen. Visual odometry. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, pages 652–659. IEEE, 2004.
- O. Özyeşil, V. Voroninski, R. Basri, and A. Singer. A survey of structure from motion. *Acta Numerica*, 26:305–364, may 2017.

- C. Papachristos, S. Khattak, F. Mascarich, and K. Alexis. Autonomous Navigation and Mapping in Underground Mines Using Aerial Robots. In *2019 IEEE Aerospace Conference*, pages 1–8. IEEE, mar 2019.
- T. Pire, T. Fischer, G. Castro, P. De Cristóforis, J. Civera, and J. Jacobo Berles. S-PTAM: Stereo Parallel Tracking and Mapping. *Robotics and Autonomous Systems*, 93: 27–42, jul 2017.
- C. Qin, H. Ye, C. E. Pranata, J. Han, S. Zhang, and M. Liu. LINS: A Lidar-Inertial State Estimator for Robust and Efficient Navigation. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 8899–8906. IEEE, may 2020.
- T. Qin, P. Li, and S. Shen. VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator. *IEEE Transactions on Robotics*, 34(4):1004–1020, aug 2018.
- 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 on open source software*, jan 2009.
- N. Ragot, R. Khemmar, A. Pokala, R. Rossi, and J. Y. Ertaud. Benchmark of visual SLAM algorithms: ORB-SLAM2 vs RTAB-map. In *2019 8th International Conference on Emerging Security Technologies, EST 2019*, pages 1–6. IEEE, jul 2019.
- W. Reid, B. Emanuel, B. Chamberlain-Simon, S. Karumanchi, and G. Meirion-Griffith. Mobility Mode Evaluation of a Wheel-on-Limb Rover on Glacial Ice Analogous to Europa Terrain. In *2020 IEEE Aerospace Conference*, pages 1–9. IEEE, mar 2020.
- Z. Ren, L. Wang, and L. Bi. Robust GICP-Based 3D LiDAR SLAM for Underground Mining Environment. *Sensors*, 19(13):2915, jul 2019.
- G. Retscher, V. Gikas, H. Hofer, H. Perakis, and A. Kealy. Range validation of UWB and Wi-Fi for integrated indoor positioning. *Applied Geomatics*, 11(2):187–195, jun 2019.
- 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. Šalanský, D. Heřt, M. Petrlík, T. Báča, V. Spurný, F. Pomerleau, V. Kubelka, J. Faigl, K. Zimmermann, M. Saska, T. Svoboda, and T. Krajník. DARPA Subterranean Challenge: Multi-robotic Exploration of Underground Environments. In *Modelling and Simulation for Autonomous Systems (MESAS 2019)*, pages 274–290. Springer International Publishing, 2020.

- U. Saranli, M. Buehler, and D. E. Koditschek. RHex: A Simple and Highly Mobile Hexapod Robot. *The International Journal of Robotics Research*, 20(7):616–631, jul 2001.
- D. Scaramuzza and F. Fraundorfer. Visual Odometry [Tutorial]. *IEEE Robotics and Automation Magazine*, 18(4):80–92, dec 2011.
- Y. Seo and C.-C. Chou. A Tight Coupling of Vision-Lidar Measurements for an Effective Odometry. In *2019 IEEE Intelligent Vehicles Symposium (IV)*, pages 1118–1123. IEEE, jun 2019.
- T. Shan and B. Englot. LeGO-LOAM: Lightweight and Ground-Optimized Lidar Odometry and Mapping on Variable Terrain. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4758–4765. IEEE, oct 2018.
- C. Solomon and T. Breckon. *Fundamentals of Digital Image Processing*. John Wiley and Sons, Chichester, UK, dec 2010.
- A. Souza Santos, H. Azpúrua, G. Pessin, and G. Freitas. Planejamento de caminhos para robôs móveis em ambientes acidentados. In *Anais do 14^o Simpósio Brasileiro de Automação Inteligente*, 2019.
- C. Stachniss, J. J. Leonard, and S. Thrun. Simultaneous Localization and Mapping. In B. Siciliano and O. Khatib, editors, *Springer Handbook of Robotics*, chapter 46, pages 1153–1176. Springer International Publishing, Cham, 2016.
- S. Thenmozhi, V. Mahima, and R. Maheswar. GPS Based Autonomous Ground Vehicle for Agricultural Utility. In *Innovations in Electronics and Communication Engineering*, pages 143–150. Springer, 2019.
- D. Tian, F. Zou, F. Xu, and P. Di. 3D Laser Odometry for a Mobile Robot Platform. In *2017 IEEE 7th Annual International Conference on CYBER Technology in Automation, Control, and Intelligent Systems (CYBER)*, pages 423–426. IEEE, jul 2017.
- R. Wang, M. Schworer, and D. Cremers. Stereo DSO: Large-Scale Direct Sparse Visual Odometry with Stereo Cameras. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 3923–3931. IEEE, oct 2017.
- W. Wang, W. Dong, Y. Su, D. Wu, and Z. Du. Development of Search-and-rescue Robots for Underground Coal Mine Applications. *Journal of Field Robotics*, 31(3):386–407, may 2014.
- T. Whelan, R. F. Salas-Moreno, B. Glocker, A. J. Davison, and S. Leutenegger. Elastic-Fusion: Real-time dense SLAM and light source estimation. *The International Journal of Robotics Research*, 35(14):1697–1716, dec 2016.

- G. Yamauchi, K. Nagatani, T. Hashimoto, and K. Fujino. Slip-compensated odometry for tracked vehicle on loose and weak slope. *ROBOMECH Journal*, 4(1):27, dec 2017.
- B. Zeisl, K. Koser, and M. Pollefeys. Automatic Registration of RGB-D Scans via Salient Directions. In *2013 IEEE International Conference on Computer Vision*, pages 2808–2815. IEEE, dec 2013.
- G. Zhai, W. Zhang, W. Hu, and Z. Ji. Coal Mine Rescue Robots Based on Binocular Vision: A Review of the State of the Art. *IEEE Access*, 8:130561–130575, 2020.
- J. Zhang and S. Singh. LOAM: Lidar Odometry and Mapping in Real-time. In *Robotics: Science and Systems X*. Robotics: Science and Systems Foundation, jul 2014.
- J. Zhang and S. Singh. Visual-lidar odometry and mapping: low-drift, robust, and fast. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2174–2181. IEEE, may 2015.
- Y. Zheng, S. Sugimoto, I. Sato, and M. Okutomi. A General and Simple Method for Camera Pose and Focal Length Determination. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 430–437. IEEE, jun 2014.
- Z. Zhou, M. Yang, C. Wang, and B. Wang. ROI-cloud: A Key Region Extraction Method for LiDAR Odometry and Localization. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3312–3318. IEEE, may 2020.
- M. Zollhöfer. Commodity RGB-D Sensors: Data Acquisition. In P. L. Rosin, Y.-K. Lai, L. Shao, and Y. Liu, editors, *RGB-D Image Analysis and Processing*, chapter 1, pages 3–13. Springer International Publishing, Cham, 2019.