



Universidade Federal da Paraíba
Centro de Tecnologia
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA MECÂNICA
- Mestrado - Doutorado

UMA ABORDAGEM DIRETA E DESACOPLADA DE SLAM VISUAL

Por

Vitor Meneghetti Ugulino de Araújo

*Tese de Doutorado apresentada à Universidade Federal da
Paraíba como parte dos requisitos para obtenção do grau de
Doutor*

VITOR MENEGHETTI UGULINO DE ARAÚJO

**UMA ABORDAGEM DIRETA E DESACOPLADA DE
SLAM VISUAL**

Tese de Doutorado apresentada ao
curso de Pós-Graduação em
Engenharia Mecânica da Universidade
Federal da Paraíba, em cumprimento às
exigências para obtenção do Grau de
Doutor.

Orientador: Prof. Dr. Francisco Antônio Belo

João Pessoa - Paraíba

abril, 2018

Catálogo na publicação
Seção de Catalogação e Classificação

A663a Araújo, Vitor Meneghetti Ugulino de.
Uma abordagem direta e desacoplada de SLAM visual /
Vitor Meneghetti Ugulino de Araújo. - João Pessoa,
2018.
112 f. : il.

Orientação: Francisco Antônio Belo.
Tese (Doutorado) - UFPB/CT.

1. Engenharia mecânica. 2. SLAM visual. 3. Robótica. 4.
Localização - robótica. 5. Mapeamento - robótico. I.
Belo, Francisco Antônio. II. Título.

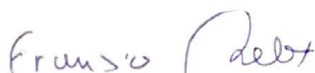
UFPB/BC

UMA ABORDAGEM DIRETA E DESACOPLADA DE SLAM VISUAL

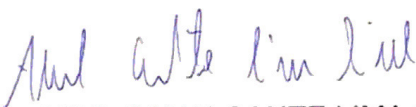
por

Vitor Meneghetti Ugulino de Araújo

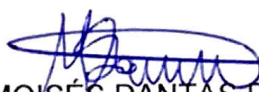
Tese aprovada em 04 de abril de 2018



Prof. Dr. FRANCISCO ANTÔNIO BELO
Orientador - UFPB



Prof. Dr. ABEL CAVALCANTE LIMA FILHO
Examinador Interno - UFPB



Prof. Dr. MOISÉS DANTAS DOS SANTOS
Examinador Interno - UFPB



Prof. Dr. ALBERDAN SANTIAGO DE AQUINO
Examinador Externo - IFPB



Prof. Dr. JOSÉ MIGUEL AROZTEGUI MASSERA
Examinador Externo - UFPB

*Aos meu pais, Leonardo e Marlene,
por todo apoio que recebi deles em
toda minha vida, aos meus irmãos,
Fábio e Viviane, por serem os
exemplos que venho seguindo e à
minha mulher, Talita, pela paciência
durante essa jornada.*

Agradecimentos

Agradeço a todos que fizeram parte dessa difícil jornada:

À minha família pelo apoio e pela educação que me deram;

Ao meu orientador, Francisco Antônio Belo, pela calma e sabedoria transmitida na etapa final dessa jornada;

Aos colegas de laboratório que me acolheram e me ajudaram nos momentos em que precisei

E aos demais colegas de pós-graduação, pelas críticas e sugestões.

Resumo

No SLAM (Simultaneous Localization and Mapping), um robô posicionado em uma localização desconhecida de um ambiente qualquer deve ser capaz de construir uma perspectiva deste ambiente (um mapa) e se localizar no mesmo simultaneamente, utilizando apenas informações captadas pelos sensores do robô e sinais de controle conhecidos.

Recentemente, impulsionados pelo avanço computacional, trabalhos nessa área propuseram usar câmera de vídeo como sensor principal e surgiu assim o SLAM Visual. Podemos observar duas linhas de pesquisas principais em SLAM Visual: a abordagem clássica, que utiliza características extraídas do ambiente e através das correspondências das mesmas e de filtros estatísticos, estimam os parâmetros necessários e a abordagem Direta, que não utiliza correspondência de características, porém é restrita à utilização de segmentos de planos e possui um sistema totalmente acoplado de registro direto de imagem e otimização, tornando difícil o acréscimo de outras formas geométricas e métodos de otimização mais modernos.

Nesse trabalho propõe-se uma nova abordagem de SLAM Visual Direto desacoplado, que utiliza métodos de otimização e registro direto de imagem independentes, tornando o sistema flexível o bastante para aceitar outras formas geométricas mais complexas. Também foi possível a utilização de diversos métodos de otimização afim de encontrar o mais adequado.

Palavras-chave: *SLAM Visual, Robótica, localização e Mapeamento.*

Abstract

In Simultaneous Localization and Mapping (SLAM - Simultaneous Localization and Mapping), a robot placed in an unknown location in any environment must be able to create a perspective of this environment (a map) and is situated in the same simultaneously, using only information captured by the robot's sensors and control signals known.

Recently, driven by the advance of computing power, work in this area have proposed to use video camera as a sensor and it came so Visual SLAM. This has several approaches and the vast majority of them work basically extracting features of the environment, calculating the necessary correspondence and through these estimate the required parameters.

In this work we propose a new approach of Direct decoupled SLAM, which uses methods of optimization and independent direct recording of images, making the system flexible enough to accept other more complex geometric forms. It was also possible to use several optimization methods in order to find the most appropriate one.

Worrying about the high computational cost of this approach have been tested several types of optimization methods in order to find a good balance between good estimates and processing time. The results presented in this work show the success of this system in different environments.

Palavras-chave: *Visual SLAM, Robotics, location and mapping*

Sumário

Sumário	i
Lista de Figuras	iv
Lista de Símbolos	vii
Lista de Tabelas	viii
1 Introdução	1
1.1 O SLAM e suas Abordagens	2
1.1.1 SLAM Visual Clássico	4
1.1.2 SLAM Visual Direto	6
1.2 Estado da Arte em SLAM Visual	7
1.2.1 Trabalhos de SLAM Visual no Brasil	9
1.2.2 Estado da Arte em SLAM Visual Direto	10
1.3 Motivação e Justificativa	12
1.4 Objetivos	14
1.4.1 Objetivos Gerais e Contribuições	14
1.4.2 Objetivos Específicos	15
1.5 Organização do Presente Documento	16
2 Fundamentação Teórica	17
2.1 Localização Robótica	17
2.1.1 Navegação Visual	19
2.1.2 Navegação Sem Mapa	20
2.2 Mapeamento Robótico	20
2.2.1 Mapas com informações espaciais	22
2.2.2 Mapas com Informações Semânticas	26
2.3 Localização e Mapeamento Simultâneo (SLAM)	26

2.3.1	Formulação Probabilística do SLAM Clássico e Filtro de Kalman	27
2.3.2	Sistemas de Visão computacional e SLAM Visual	29
2.4	Registro de Imagem e SFM	30
2.4.1	Registro de Imagem	30
2.4.2	Estrutura a partir do Movimento (<i>Struct from motion - SFM</i>) . . .	37
2.5	Métodos de Otimização	42
2.5.1	Métodos de Otimização Baseados em Gradientes	43
2.5.2	Métodos de Otimização Baseados em Meta-Heurísticas	44
2.5.3	Métodos de Otimização Livres de Derivadas	46
2.6	Conclusão do Capítulo	47
3	Desenvolvimento e Metodologia	48
3.1	Caracterização do Problema	48
3.2	Solução Desenvolvida	49
3.3	Metodologia	54
3.4	Conclusão do capítulo	63
4	Resultados Experimentais	64
4.1	Informações Preliminares	64
4.2	Experimentos	66
4.2.1	Experimento 1: Teste em Ambiente Interno	67
4.2.2	Experimento 2: Teste em Ambiente Interno 2	67
4.2.3	Experimento 3: Teste em Ambiente Externo	68
4.2.4	Experimento 4: Teste em ambiente Externo 2	68
4.2.5	Experimento 5: Teste Alternando o número de Regiões	69
4.2.6	Experimento 6: Segunda Versão do sistema	69
4.2.7	Experimento 7: Comparando com uma técnica consagrada	71
4.3	Resultados	72
4.3.1	Resultados Experimento 1	72
4.3.2	Resultados do Experimento 2	73
4.3.3	Resultado do Experimento 3	75
4.3.4	Resultados do Experimento 4	77
4.3.5	Resultados do Experimento 5	79
4.3.6	Resultados do Experimento 6	81
4.3.7	Resultados do Experimento 7	81
4.4	Conclusão do capítulo	83

5	Conclusões e perspectivas	84
5.1	Conclusão	84
5.2	Perspectivas	85

Lista de Figuras

1.1	Exemplificação de um problema de SLAM	3
1.2	Principais Sistemas visuais de percepção.	4
1.3	Exemplo de extração e correspondências entre características	5
1.4	Extração de características utilizando linhas no chão	8
1.5	Sensor Kinect	9
1.6	Regiões planares selecionadas para processamento	11
1.7	Motivo pelo qual não há acúmulo do erro	13
1.8	Seleção de regiões da imagem no trabalho de Silveira	13
1.9	Caso os parâmetros, no espaço 3D, dessas formas sejam conhecidos é possível a simulação dos mesmos em diferentes pontos de vistas.	14
2.1	Etapa de um processo de localização baseado em marcos ou características.	18
2.2	Exemplo de fusão sensorial com filtro de Kalman	19
2.3	Esquema de construção de um mapa topológico.	24
2.4	Principais elementos utilizados em um mapa de características	25
2.5	Sistemas de visão mais utilizados no SLAM	29
2.6	Exemplo de transformações que podem ser aplicadas em uma imagem	31
2.7	Exemplo de um alinhamento usando extração e correspondência de características	34
2.8	Formação das imagens após um deslocamento da câmera	37
2.9	Projeção no plano da câmera de um ponto X no espaço, R e T representam as transformações entre as coordenadas do mundo XYZ e as da câmera $X_cY_cZ_c$	38
2.10	Geometria epipolar de 2 pontos de vista: Dado a projeção $\tilde{x}$ em uma imagem de um ponto X no espaço, a projeção $\tilde{x}'$ na outra imagem está restrita a linha epipolar I' , onde C representa a origem do sistema de coordenada da câmera e e representa o ponto epipolar.	40

2.11	Esquematisação do método sequencial, onde E_{ij} representa a matriz essencial entre os pontos de vista i e j	41
2.12	Algoritmo padrão de métodos evolutivos	45
3.1	Ambiente de fácil extração e correspondência de características	49
3.2	Exemplo de ambientes com formas geométricas não planares	50
3.3	Esquema do sistema	51
3.4	Exemplo de uma cena gerada a partir de parâmetros equivocados	52
3.5	Exemplo de uma cena gerada a partir de parâmetros ótimos	52
3.6	Transformações (T_1, T_2 e T_3) que causam oclusão de pixels	53
3.7	Exemplo de pixels que desaparecem ou aparecem de acordo com o deslocamento da câmera	54
3.8	Parametrização do plano	54
3.9	Exemplo de péssimos parâmetros que geraram um grande erro.	59
3.10	Exemplpo de bons parâmetros que geraram um erro pequeno.	59
3.11	Evolução em termos de complexidade das diversas formas de representação do mundo.	60
3.12	Parametrização do esferóide	60
3.13	Exemplo de pixels que desaparecem ou aparecem de acordo com o deslocamento da câmera.	62
4.1	Alguns frames das sequências de imagens utilizada nos testes.	65
4.2	Sistema de referência da câmera.	66
4.3	Ambiente de teste do experimento 1.	67
4.4	Ambiente de Teste do Experimento 2.	67
4.5	Ambiente externo com as respectivas regiões de interesse.	68
4.6	Segundo ambiente externo e as regiões de interesse.	68
4.7	Imagem de referência	69
4.8	Imagem deslocada	69
4.9	Regiões de interesse em vermelho	70
4.10	Regiões selecionadas para processo de estimação	71
4.11	Resultados obtidos com Algoritmo genéticos.	73
4.12	Resultados obtidos com poliedros flexíveis.	73
4.13	Resultados obtidos com Levenberg-Marquadt.	73
4.14	Resultados obtidos com Algoritmo Genético.	74
4.15	Resultados obtidos com Poliedro Flexível.	74
4.16	Resultados obtidos com Levenberg-Marquadt.	75

4.17	Resultados obtidos com Levenberg-Marquadt.	76
4.18	Resultados obtidos com Algoritmo genéticos.	77
4.19	Resultados obtidos com poliedros flexíveis.	77
4.20	Resultados obtidos com Algoritmo Genético.	78
4.21	Resultados obtidos com Poliedro Flexível.	78
4.22	Resultados obtidos com Levenberg-Marquadt.	78
4.23	Resultados obtidos com diferentes números de regiões.	80
4.24	Resultados obtidos com diferentes números de regiões no mesmo gráfico.	80
4.25	Deslocamento estimado pelas diferentes técnicas.	82

Lista de Símbolos

ϵ_t, δ_t	ruídos gaussianos de média zero
$\mathbf{s}_t$	vetor de estados (ordem n) no instante t
$p(\cdot)$	modelo não linear do sistema
$\mathbf{u}_{t-1}$	sinais de entrada (ordem l), no instante $t-1$
ϵ_{t-1}	ruído do processo (ordem q), no instante $t-1$
$\mathbf{z}_t$	vetor de medições (ordem m) retornadas pelo sensor
$h(\cdot)$	modelo não linear do sensor
δ_t	ruído de medição
$\bar{\mu}_t, \mu_t$	média (ordem n) do vetor de estados $\mathbf{s}_t$, antes e depois da fase de atualização
$\bar{\Sigma}_t, \Sigma_t$	covariância ($n \times n$) do vetor de estados $\mathbf{s}_t$
$\mathbf{G}_t$	matriz Jacobiana ($n \times n$) que lineariza o modelo do processo $p(\cdot)$
$\mathbf{V}_t$	matriz Jacobiana ($n \times q$) que lineariza o ruído do processo ϵ_t
$\mathbf{M}_t$	covariância ($q \times q$) do ruído de processo ϵ_t
$\mathbf{K}_t$	matriz ganho de Kalman ($n \times m$)
$\mathbf{H}_t$	matriz Jacobiana ($m \times n$) que lineariza o modelo do sensor $h(\cdot)$
$\mathbf{Q}_t$	covariância ($m \times m$) do ruído de medição δ_t
f	distância focal
ρ	segmento de reta relativo à menor distância entre o plano e a origem
α	ângulo formado entre ρ e o plano XZ
β	ângulo formado entre a projeção de ρ no plano XZ e o eixo Z
ϕ, θ	ângulos de atuação de raio da esfera
i_c	vetor de características do i -ésimo marco
I_1, I_2	Intensidade de pixel na coordenada (x, y) de uma imagem
g	transformação radiométrica

Lista de Tabelas

2.1	Símbolos das equações (2.1) até (2.6)	28
4.1	Valores médios do número de iterações, tempo em milisegundos e da função minimizada.	66
4.2	Valores dos parâmetros de cada segmento de plano do experimento 1. . .	72
4.3	Valores dos parâmetros de cada segmento de plano do experimento 2. . .	75
4.4	Valores dos parâmetros de cada segmento de plano do experimento 3. . .	76
4.5	Valores dos parâmetros de cada segmento de plano do experimento 4. . .	79
4.6	Tempo médio de convergência do algoritmo em cada situação.	79
4.7	Valores dos parâmetros de cada região.	81
4.8	Valores dos parâmetros de cada região.	82

Capítulo 1

Introdução

Nas últimas décadas a busca por sistemas robóticos autônomos que possam auxiliar o homem nas tarefas e desafios do cotidiano recebeu uma grande atenção das engenharias e ciências modernas e proporcionou a criação de novas áreas de pesquisas.

Um fator essencial para um sistema robótico é torná-lo autônomo. Robôs autônomos devem apresentar reações às mudanças do ambiente, comportamento inteligente, integração entre vários sensores, resolução de múltiplas tarefas, robustez, capacidade de operar sem falha, programabilidade, modularidade, flexibilidade, expansibilidade, adaptabilidade e raciocínio global.

[Craig 2004] classifica os robôs em duas grandes classes: robôs manipuladores e robôs móveis. Os manipuladores robóticos têm forte atuação nas indústrias, sobretudo na automobilística. A classe de robôs móveis é constituída por robôs que tem a capacidade de locomoção no espaço tridimensional (robôs aeroespaciais e robôs subaquáticos) ou no espaço planar (robôs terrestres).

Inicialmente houve um grande esforço no desenvolvimento da robótica na área industrial, com a utilização de robôs manipuladores. Posteriormente as pesquisas em robótica se concentraram na construção de robôs móveis, introduzindo capacidades de mobilidade e autonomia para reagir adequadamente ao ambiente. Essa evolução proporcionou um vasto campo de novas aplicações e, conseqüentemente, grandes desafios.

As aplicações envolvendo robôs móveis vêm crescendo significativamente nos últimos anos devido a uma classe de tarefas que esses podem realizar, já que, ao contrário dos manipuladores, possuem capacidade de se locomover livremente pelo espaço de trabalho, limitados apenas por eventuais obstáculos.

Ao fornecer a capacidade de mobilidade à um sistema robótico, além de dispositivos que possibilitem essa locomoção, são necessários sensores que retornem informações so-

bre o ambiente no qual o mesmo está inserido e um sistema de processamento capaz de interpretar essas informações. Na robótica móvel grande parte das informações retornadas pelos sensores são utilizadas para tarefas de localização ou mapeamento, visto que as demais aplicações dependem fortemente dessas tarefas.

Até o final da década de 80, as tarefas de localização e mapeamento eram realizadas de forma isolada, ou seja, caso o mapa fosse conhecido a priori a tarefa seria de localização e caso a localização fosse conhecida a tarefa seria de mapeamento. Então surgiu a ideia de realizar o mapeamento e a localização de forma simultaneamente e iterativa. Esse problema ficou conhecido na robótica como SLAM (*Simultaneous Localization and Mapping*).

1.1 O SLAM e suas Abordagens

O SLAM consiste em estimar, de maneira incremental, um mapa do ambiente e simultaneamente determinar a posição do robô baseando-se no mapa gerado. A gênese desse problema surgiu em meados da década de 80 em São Francisco no IEEE Robotics and Automation Conference de 1986 [Durrant-Whyte e Bailey 2006]. Alguns pesquisadores como Peter Cheeseman, Jim Crowley, Hugh Durrant-Whyte, Raja Chatila, Oliver Faugeras e Randal Smith foram pioneiros com publicações como as de Smith et al. (1987) e Durrant-Whyte (1988) entre outras.

Segundo [Santana 2010], uma compreensão intuitiva do processo de SLAM pode ser conduzida por um exemplo hipotético. Considere um robô móvel simples, composto de um conjunto de rodas ligado a um motor, uma câmera e atuadores para controlar a velocidade e a direção da unidade; agora imagine o robô sendo utilizado remotamente por um operador para mapear locais inacessíveis; os acionadores permitem que o robô se mova, e a câmera fornece informação visual suficiente para o operador entender onde e como os objetos estão em torno do robô. O que acabamos de descrever é um exemplo de SLAM: determinar a localização de objetos no ambiente é um exemplo de mapeamento, e inferir a pose do robô com relação a esses objetos é um exemplo de localização.

O problema de SLAM pode ser pensado como um “problema do ovo e da galinha”, pois: para a localização é necessário o conhecimento do mapa enquanto que uma estimativa exata da localização (pose) do robô é necessária para construir esse mapa. A figura 1.1 ilustra o problema de SLAM.

Sabe-se que as informações obtidas a partir de sensores são, muitas vezes, imprecisas, devido a limitações físicas e a fenômenos que não podem ser previstos e, portanto, modelados computacionalmente. Dessa forma, as estimativas de localização e mapeamento

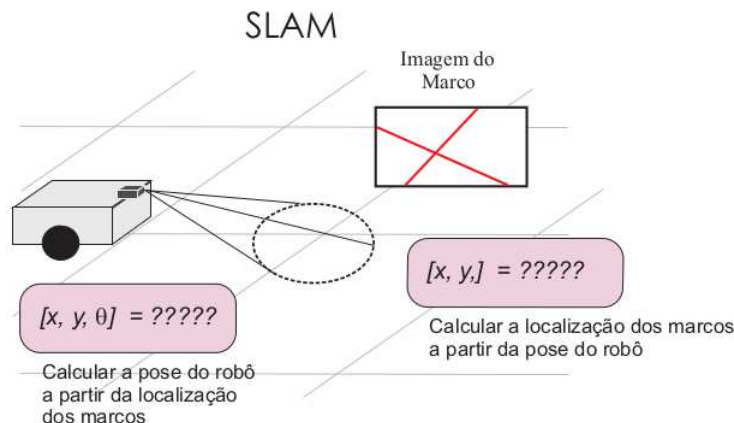


Figura 1.1: Exemplificação de um problema de SLAM

obtidas pelo robô a partir dessas informações possuirão erros que estabelecem uma diferença entre a realidade e como ela é percebida pelo robô. Se esses erros não forem tratados chegará o momento em que a imprecisão será tão grande que já não faz sentido utilizar essas estimativas, invalidando todo o processo.

Neste sentido, soluções probabilísticas lidam com o problema de SLAM através da minimização simultânea das incertezas envolvidas em cada uma das estimativas, reconhecendo que ambas são dependentes entre si. Resultados precisos da localização são necessários para o estabelecimento de um mapa preciso e vice-versa, e dessa forma não faz sentido abordar apenas um dos problemas, é necessário resolvê-los simultaneamente, eliminando sistematicamente os erros antes que eles se acumulem.

Nos primeiros trabalhos de SLAM foram utilizados sensores de alcance, como escâners lasers e sonares, e filtros estatísticos, como o filtro de Kalman [Kalman 1960], para tratar as incertezas das medidas retornadas pelos sensores e do modelo do sistema robótico. Trabalhos como [Clark e Dissanayake 1999], [Guivant e Nebot 2002], [Wang 2004] trataram o SLAM dessa forma.

Scanners laser são precisos, eficientes e não exigem muita computação para processar a informação por eles fornecida. As principais desvantagens na utilização destes sensores são que não funcionam bem em superfícies formadas por vidro e são muito caros. Já os sonares são sensores baratos, quando comparados aos scanners laser, porém, não apresentam leituras tão precisas e enfrentam grandes problemas de reflexão das ondas sonoras.

Em meados da década de 90 surgiu a ideia conhecida como SLAM Visual, que propõe o uso de câmera de vídeo como principal sensor do robô. A evolução dos hardwares e softwares nessa década foi um dos fatores que permitiram o surgimento dessa abordagem. A utilização de sensores visuais como uma forma de obtenção de informações do ambi-

ente parece ser mais intuitivo e pode ser mais atraente que laser ou sonar. O maior atrativo da utilização de imagens está na riqueza de informações que elas são capazes de fornecer, qualitativa e quantitativamente.

1.1.1 SLAM Visual Clássico

SLAM visual é, basicamente, o SLAM onde se usa câmeras de vídeo ao invés de sensores de alcance. Isso proporciona algumas vantagens como o fato de que uma câmera de vídeo fornece uma fonte mais rica de informações, proporcionando representações mais avançadas do mundo. Nessa técnica é construído um mapa visual, que consiste normalmente de um conjunto de marcos identificados pelo robô no seu caminho.

Em um ambiente desconhecido, o robô automaticamente identifica marcos, e quando revisita áreas mapeadas, ele utiliza os marcos conhecidos para estimar e melhorar sua pose e a localização dos marcos. Ao atualizar continuamente a posição de marcos e a pose do robô com base em novos dados, incrementalmente melhora-se o mapa calculado.

Os principais tipos de sistemas de percepção visual utilizados pelos robôs são: visão monocular, visão estéreo e visão omnidirecional (figura 1.2).



Figura 1.2: Principais Sistemas visuais de percepção.

Os sistemas de visão monocular enfrentam o problema de estimar corretamente as informações 3D dos marcos a partir de uma imagem. Os sistemas de visão estéreo não possuem este problema, mas possuem maior custo computacional. Já os sistemas de visão omnidirecional apresentam como vantagem a imagem panorâmica do ambiente e a desvantagem está associada à deformação que a imagem sofre quando o sistema utiliza um espelho ou o casamento das imagens quando são usados sistemas multi-câmera.

Nas pesquisas sobre SLAM Visual podemos observar um grande número de trabalhos em uma abordagem considerada clássica, onde se utiliza a correspondência entre primitivas geométricas extraídas das imagens para estimar o deslocamento da câmera e o mapa do ambiente. O objetivo é encontrar um conjunto coerente de primitivas geométricas correspondentes em dois ou mais pontos de vistas da mesma cena e com isso

estimar os parâmetros desejados através de um tratamento adequado das incertezas, presentes no sistema de aquisição e no próprio robô. Ou seja, baseando-se na similaridade entre as características extraídas, calculam-se os parâmetros que melhor representem a ligação entre pares de característica em diferentes imagens. Esses parâmetros representam o deslocamento da câmera e as coordenadas das características representam o mapa.

Para tanto são necessárias etapas iniciais de extração, descrição e correspondência de características da imagem e etapas de refinamento onde as falsas correspondências devem ser rejeitadas, posteriormente, utilizando-se técnicas robustas (e.g., RANSAC). Segundo Chen (2011) os métodos tradicionais costumam selecionar características baseadas em primitivas geométricas como pontos, cantos, ou retas.

A figura 1.3 mostra um exemplo de pares de característica correspondentes em dois pontos de vista da mesma cena.

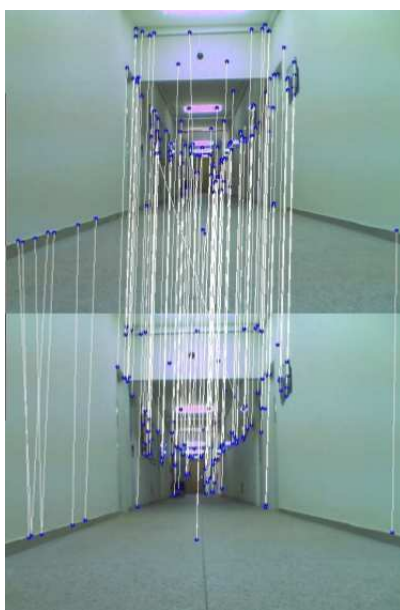


Figura 1.3: Exemplo de extração e correspondências entre características

Para se detectar essas característica (pontos, cantos, retas, etc), existem vários algoritmos consagrados na literatura, dos quais podemos destacar dois: Shi-Tomasi e o algoritmo SIFT (Scale Invariant Feature Transform). O detector Shi-Tomasi extrai informações de cantos na imagem e usa essas regiões para descrever as características. Shi-tomasi é parcialmente invariante à rotação, mas não é invariante à escala e iluminação [Jing Wu e Zhang 2007]. Já o algoritmo SIFT é invariante à escala e rotação e parcialmente invariante à iluminação. Embora o algoritmo SIFT seja mais robusto, ele também é computacionalmente custoso.

Em relação ao mapeamento, nas técnicas clássicas de SLAM Visual, também fazem-se necessárias algumas etapas a posteriori, onde o conjunto de pontos 3D extraídos devem ser refinados, agrupados, interpolados e/ou texturizados para se obter um mapa mais representativo do ambiente. Observa-se, nesse tipo de abordagem, uma grande quantidade de parâmetros gerados no processo de mapeamento. Essa quantidade está diretamente ligada ao número de características extraídas ao longo do tempo.

A maior dificuldade na tarefa de localização e mapeamento encontra-se na existência de incertezas inerentes aos sistemas robóticos e aos dispositivos sensoriais. Na abordagem clássica utilizam-se técnicas probabilísticas, como filtro de kalman para tratar essas incertezas. Essa técnica leva em consideração as propriedades estatísticas do ruído presente no sistema robótico e um esquema de realimentação que utiliza as medições retornadas pelos sensores.

1.1.2 SLAM Visual Direto

Como foi visto na seção anterior, existem inúmeros trabalhos consolidados que encontram a solução para o slam visual de forma clássica, ou seja, com etapas a priori de extração e correspondência de características das imagens, como pontos, linhas, cantos, etc. Uma questão relevante é: o que ocorre se o sistema robótico estiver inserido em um ambiente onde não é possível a extração dessas características ? Para tentar solucionar essa questão surgiu uma nova abordagem de SLAM Visual, também chamada de abordagem direta, onde a pose do robô e o mapa do ambiente são obtidos diretamente da observação da intensidade dos pixels nas sucessivas imagens, evitando assim as etapas de extração, descrição e correspondência de características e muitos problemas associados a essas etapas.

Na abordagem direta de SLAM Visual não se utilizam modelos probabilísticos, ao invés disso são utilizados métodos de otimização para tratar os erros de reprojeção obtidos através de técnicas de registro de imagem. Esse processo de otimização pode ser resumido em três etapas:

1. A transformação da imagem, tendo em conta o modelo e os parâmetros correntes;
2. O cálculo da medida de similaridade, ou seja, a função custo;
3. O cálculo do incremento nos parâmetros que otimiza a função custo.

Em outras palavras, utiliza-se um modelo de transformação geométrica da cena e uma técnica que calcule a diferença entre o que o modelo retorna e o que realmente aparece na imagem após os deslocamentos do robô. Então, imagine um objeto na frente da câmera

de um robô. Ao enviar um comando para que o robô realize um deslocamento é possível calcular como esse objeto deverá aparecer no próximo ponto de vista através do modelo de transformação e os parâmetros que foram enviados ao robô. Acontece que o robô pode ter deslizado e não atingiu a posição especificada. Nesse caso a imagem do objeto capturada pela câmera (realimentação sensorial) será diferente da esperada, que foi calculada pelo modelo. Essa diferença é conhecida como erro de reprojeção e deve ser otimizada para estimar, tanto os parâmetros relativos a localização do robô, quanto aos que representam o forma geométrica e a posição do objeto (mapa).

1.2 Estado da Arte em SLAM Visual

Certamente os nomes mais citados, direta ou indiretamente, quando se pesquisa sobre SLAM Visual baseado em extração e correspondência de característica, são:

- **nos Estados Unidos:** Sebastian Thrun, Kurt Konolige e Dirk Haehnel da Universidade de Stanford; Dieter Fox da Universidade de Washington; John Leonard e Mathew Walter do Instituto Tecnológico de Massachusetts;
- **no Alemanha:** Wolfram Burgard da Universidade Bonn, Cyrill Stachniss da Universidade Freiburg e Udo Frese da Universidade de Bremen;
- **na Espanha:** Avier Civera, José Castellanos, Juan Tardós, José Neira, Carlos Estrada e José Martinez, todos da Universidade de Zaragoza;
- **na Inglaterra:** Andrew Davison, David Murray e Paul Newman, todos da Universidade de Oxford;
- **outros:** Hugh Durrant-Whyte da Universidade de Sidney-Austrália, Simon Lacroix do LAAS-França e Young-Ho Choi da Korea.

Podemos citar alguns trabalhos de grande relevância como de Davison e Murray (2002), que foi um dos primeiros a propor o uso de câmera de vídeo como sensor principal do robô. Posteriormente Davison (2003) propõe o uso de uma câmera monocular. Um grande problema de se usar um sistema monocular diz respeito à profundidade da cena, e o trabalho de Clemente et al. (2007) utiliza inversão de profundidade e mapas hierárquicos nos seus testes para contornar esse problema. Um trabalho mais recente nessa linha de SLAM Visual monocular é o de Kwon (2010) que usa planos e filtro de partículas para tratar as incertezas.

Outro sistema de visão muito utilizado na abordagem clássica de SLAM visual é a câmera estéreo, que tem como vantagem o fato de que profundidade da cena pode ser obtida facilmente através do mapa de disparidade de pixels das imagens direita e esquerda.

Nessa linha podemos citar alguns trabalhos mais recentes como os de Chatterjee (2011), Lategahn (2011), Gutierrez (2011). entre outros.

No trabalho de Santana (2010) encontra-se um estado da arte muito abrangente sobre essa abordagem de SLAM Visual, porém ele utiliza um sistema de visão monocular e extrai linhas presentes no chão, como pode ser visto na figura 1.4. Desse modo permite-se que o robô navegue no ambiente fundindo informações de odometria e de visão monocular.

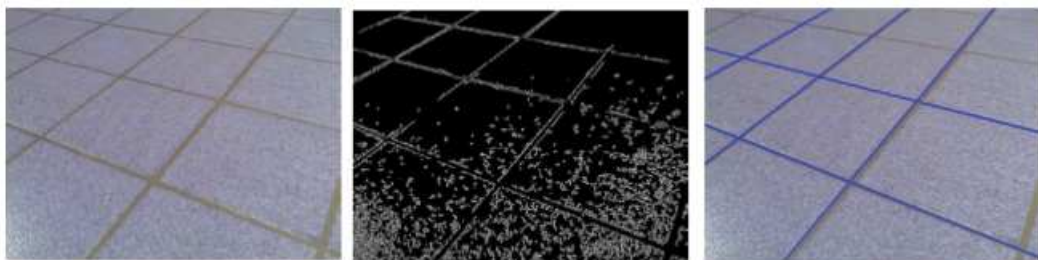


Figura 1.4: Extração de características utilizando linhas no chão

Já no trabalho Klippenstein e Zhang (2007) a preocupação foi a otimização do processo de SLAM Visual como um todo, independente do sistema de visão utilizado. Então foram comparadas diversas técnicas e algoritmos de extração e correspondência de características e concluiu-se que a combinação da técnica de detecção *Harris* com o descritor *SIFT* funciona muito bem para esse problema. Trabalhos como o de Angeli et al. (2008), Gee et al. (2008) e Civera et al. (2008) tratam o problema de fechamento de loop, que consiste em um robô ser capaz de detectar uma característica já visualizada anteriormente.

No trabalho de Ouellette e Hirasawa (2007) que apresenta um quadro comparativo entre diversas técnicas clássicas para SLAM Visual, mostrando as vantagens na utilização deste tipo de sensor para resolver o problema e classificando as duas abordagens principais em SLAM Visual: são os métodos baseados em características esparsas, utilizados mais recentemente em R. MurArtal, J. M. M. Montiel, and J. D. Tardos (2015), e os baseados em características densas, como em Jakob Engel, Thomas Schöps, and Daniel Cremers (2016).

Recentemente houve uma tendência por usar sensores RGB-D, como por exemplo o sistema de visão "Kinect" [Newcombe 2011], que é capaz de captar, além da imagem, uma informação extra sobre a profundidade da cena. Nessa linha de pesquisa podemos citar os trabalhos de HOGMAN (2013) e Henry P. e Fox (2014). O sensor "Kinect" (figura 1.5) é um dispositivo desenvolvido pela Primesense, inicialmente destinado ao Microsoft

Xbox 360, (lançado em novembro de 2010). É composto por uma câmera RGB, e sensores de profundidade 3D, um conjunto de microfones e um motor para dar inclinação.



Figura 1.5: Sensor Kinect

1.2.1 Trabalhos de SLAM Visual no Brasil

Na comunidade robótica brasileira é comum o uso de sensores visuais para a solução de inúmeros problemas de robótica. Alguns nomes consolidados de pesquisadores que utilizam este tipo de sensor nos seus trabalhos são: Adelardo Medeiros, Anna Reali, Denis Wolf, Fernando Osório, Geovany Borges, Geraldo Silveira, Guilherme Pereira, Jun Okamoto, Luiz Marcos, Mário Campos, Mário Sarcineli, Pablo Alsina, Raquel Vassallo, Roseli Romero, Silvia Botelho, Theodiano Bastso, Walter Lages, dentre outros.

Anna Reali, por exemplo, possui trabalhos relacionados a mapeamento visual monocular, reconstrução de planos para ambientes internos, mapas baseados em objetos, correspondência de linhas em imagens, dentre outros: [Werneck e Costa 2010b], [Werneck e Costa 2010a].

Silvia Botelho, por sua vez, publicou diversos trabalhos usando visão computacional para ambientes subaquáticos e também para monitoramento de linhas subterrâneas de distribuição de energia [Botelho et al. 2005], [Botelho et al. 2004] e [Botelho et al. 2003].

Já Denis Wolf e Fernando Osório, fogem um pouco da área de SLAM, porém possuem trabalhos importantes no campo da visão computacional. Usam redes neurais para classificação de terrenos a partir de imagens [Hata e Ososrio 2009], enquanto que Geovany Borges identifica linhas em imagens e as utiliza em servo-visão [Bernardes e Borges 2009]. Geraldo Silveira faz SLAM visual usando registro de imagem [Silveira et al. 2009] e Jun Okamoto resolve este problema usando visão omnidirecional.

Os trabalhos mais recentes no brasil são os de Guapacha (2017), Zacarkim (2017) e da Silva (2015) , ambos abordaram o método clássico de SLAM Visual.

1.2.2 Estado da Arte em SLAM Visual Direto

A utilização de métodos diretos em visão computacional começou a ser discutida na década de 90 com trabalhos pioneiros como o de Hanna (1991) e Szeliski e Kang (1995). No final da década Irani e Anandan (1999) publicaram um dos primeiros trabalhos consistentes sobre esse tema, até então aplicado apenas a visão computacional, pois a aplicação na robótica era proibitiva devido à complexidade computacional dessa classe de métodos. Porém, nos últimos anos, o aumento do poder computacional e a evolução de alguns algoritmos possibilitaram alguns trabalhos como o de Jin et al. (2003), ainda no campo de visão computacional, que trata o problema de estrutura a partir do movimento de forma direta e simultânea, mas requer movimentos, relativamente, lentos da câmera.

Outros trabalhos aplicados a estrutura a partir do movimento foram os de Davison et al. (2004), que usa estrutura a partir do movimento em tempo real e considera a imagem como um conjunto de planos locais usando métodos diretos para estimar o modelo de transformação desses planos. Essa técnica é sensível às mudanças de iluminação, não considera o forte acoplamento entre o movimento da câmera e a estrutura da cena que são estimados separadamente. Podemos falar também do trabalho de Murray e Klein (2008) que implementou o que eles chamaram de SLAM visual baseado em keyframes, onde tratam os movimentos rápidos da câmera e conseguem fazer o tracking usando técnicas diretas, mas deixam bem claro que a técnica é mais para tracking do que para mapeamento.

O primeiro trabalho que propôs usar métodos direto na robótica para fazer SLAM foi o de Silveira et al. (2007), onde foi proposto usar registro direto de imagem no SLAM. Para eles, o SLAM Visual podia ser visto como um problema de alinhamento não linear de imagens, onde os pixels nas imagens de referência seriam comparados e alinhados com os das imagens captadas, através de um modelo foto-geométrico de transformação. Foi utilizada uma técnica de otimização para minimizar erros de alinhamento, baseando-se no trabalho de Malis (2004), onde encontram-se comparações entre o uso de vários métodos de otimização de segunda ordem em visão computacional.

No trabalho de Silveira et al. (2007) existe uma forte restrição para que sistema funcione de forma adequada. São selecionadas apenas algumas regiões da imagem que representem segmentos de planos no mundo, como mostrado na figura 1.6. Estas foram automaticamente selecionadas baseando-se em um índice que reflete fortes gradientes em diferentes direções na imagem (texturas diferentes).

Nos anos seguintes, nos trabalhos de Silveira et al. (2008) e Silveira et al. (2009) a ideia foi consolidada, porém ainda restrita à utilização de segmentos de planos. Foi con-

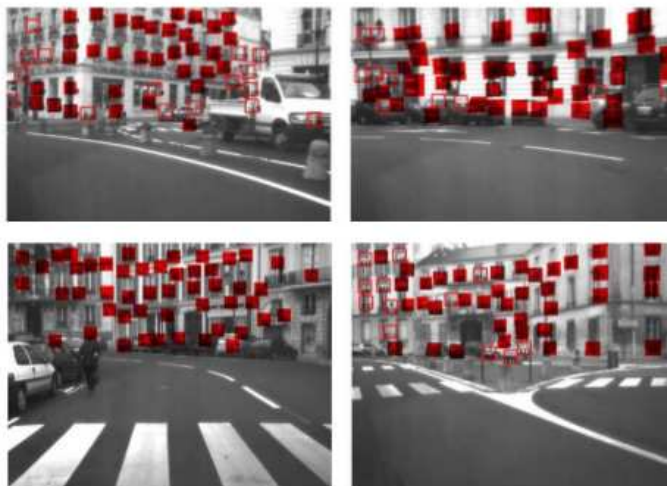


Figura 1.6: Regiões planares selecionadas para processamento

cluído que a exploração de mais informações, junto com algumas restrições estruturais aumentam a precisão e a robustez intrínseca do algoritmo com respeito às medidas aberrantes. Porém pouco se fala sobre mapa formado após a estimação dos parâmetros. Este se torna um pouco mais representativo que os gerados através de pontos, porém continua com um grande número de parâmetros e torna difícil o fechamento de loop, que é quando o robô passa por uma região já visitada anteriormente. Essa dificuldade ocorre devido a grande quantidade de informação que deve ser armazenada ao longo do tempo e o autor não deixa claro como ele realiza isto.

Recentemente apareceram alguns trabalhos como Concha et al. (2016), que propõe utilizar computação paralela para rodar um algoritmo de SLAM Visual direto denso em tempo real e assim como os já citados anteriormente está restrito a utilização de apenas uma primitiva geométrica. Temos também, o de Forster et al. (2014), que propõe um algoritmo de odometria visual monocular semi-direta, que não utiliza algoritmos de extração de características, porém utiliza apenas pontos 3D como representação do mundo. Já o de Lim et al. (2014) propõe utilizar métodos diretos para localização e mapeamento de Drones, nesse caso existe uma preocupação maior com os seis graus de liberdade do robô, visto que no céu o movimento nesses graus de liberdade são mais comuns. A utilização de regiões planares funciona bem nesse caso, pois visto de cima, o terreno abaixo parece ser plano.

Os trabalhos mais recentes são bem importantes, pois podem incorporar melhorias na técnica desenvolvida nesse trabalho. Eles focam na melhoria das estimativas: o trabalho

de Park et al. (2017) utiliza um algoritmo direto que é robusto à grandes diferenças na iluminação entre as tomadas das imagens. Isso é extremamente importante nos métodos diretos. E por fim, temos o trabalhos de Cremers (2017), propõe melhorar as estimativas dos métodos diretos incorporando modelos precisos no processo da formação de imagens, incluindo aspectos como a atenuação da lente (vignetting), tempo de exposição e correção gama. Como resultado, isso substitui a constância clássica da luminosidade e a constância de irradiação, por exemplo, a luz emitida a partir de um dado ponto 3D é suposto como sendo constante e independente de direção.

1.3 Motivação e Justificativa

Nesse trabalho é apresentado uma nova abordagem direta de SLAM Visual. Nesse tipo de abordagem existem muitas vantagens, das quais se destacam: a maior precisão nas estimativas, devido à utilização de uma maior número de informações contidas na imagem, e ao fato de evitar o acúmulo ou propagação do erro retornado por sensores. Esse problema de acúmulo de erro é muito comum nas abordagens tradicionais, Na abordagem direta esse problema não ocorre porque, dado uma imagem de referência, as estimativas são feitas comparando-se as várias imagens sucessivas com a imagem de referência, até que as regiões na cena que representam essa imagem de referência não apareça mais no campo de visão. A figura 1.7 exemplifica essa situação: note que a região em vermelho está aparecendo em várias imagens sucessivas após diversos deslocamentos da câmera. Todas essas imagens sucessivas estão sendo comparadas com uma única imagem de referência, até que a região de interesse não apareça mais no campo de visão.

Outras vantagens desse tipo de abordagem são obtidas através da utilização métodos de otimização ao invés de métodos probabilísticos. No trabalho de Davison et al. (2012) é realizado um estudo comparativo entre a utilização de métodos probabilísticos e de otimização no SLAM Visual, comprovando que a utilização de métodos de otimização no SLAM Visual retorna melhores resultados em diversos aspectos.

Porém nesse tipo de abordagem existem duas restrições importantes. A primeira está relacionada ao custo computacional de se usar grandes partes da imagem ou ela toda. Além do mais, é difícil parametrizar matematicamente as inúmeras formas geométricas que podem aparecer quando se usa uma grande parte da imagem do mundo 3D.

A segunda restrição está no fato de que existem inúmeras possibilidades que levam à oclusão de parte da imagem, tornando difícil a utilização de registro direto de imagem para retornar o erro de reprojeção, visto que é necessário que a região de interesse apareça por completo (todos os pixels) na imagem de referência e nas sucessivas imagens.

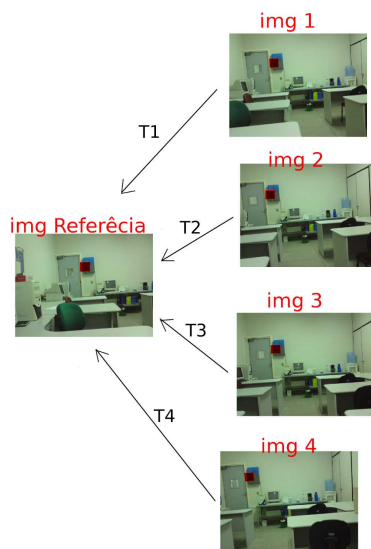


Figura 1.7: Motivo pelo qual não há acumulo do erro

Em [Davison et al. 2004] foi observado que essas restrições podem ser evitadas ao se utilizar regiões da imagem que representem planos no mundo real e que a seleção dessas regiões é uma tarefa considerada simples. Assim, observa-se que os trabalhos existentes na abordagem direta de SLAM Visual utilizam apenas regiões planares, como pode ser visto na imagem 1.8 retirada do trabalho de Silveira et al. (2008), que utiliza as regiões em vermelho para estimar os parâmetros necessários, regiões estas que representam pequenos segmentos de planos aproximados, com texturas não homogênea.



Figura 1.8: Seleção de regiões da imagem no trabalho de Silveira

Além de ser muito restritivo, tornar difícil o fechamento de loop e gerar um grande número de parâmetros (dependendo do número de regiões utilizadas), a utilização de apenas regiões planares pequenas retornam um mapa pouco representativo sobre o ambiente, assim como o mapa de características que utilizam pontos.

Então, a motivação para desenvolver esse trabalho é encontrar uma solução eficiente e precisa para o SLAM Visual sem a necessidade de extração e correspondência de carac-

terísticas em nenhuma etapa do processo. A abordagem desenvolvida deve utilizar partes maiores da imagem com formas geométricas mais complexas, e assim ficar livre da restrição de utilizar apenas regiões planares. Com isso é possível obter um mapa 3D com menos parâmetros que seja altamente manipulável através de técnicas de visão computacional. A figura 1.9 mostra um exemplo de uma mapa 3D baseados em formas geométricas.

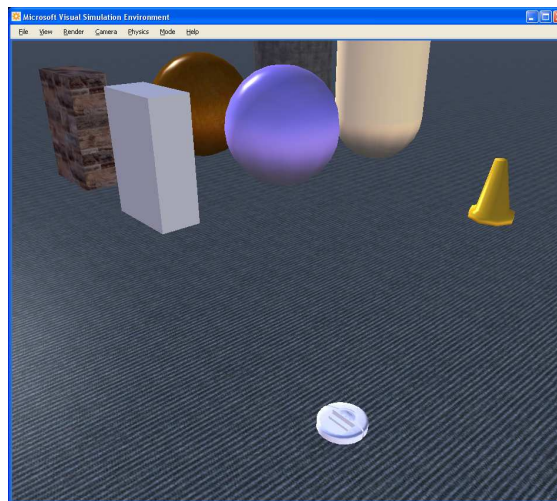


Figura 1.9: Caso os parâmetros, no espaço 3D, dessas formas sejam conhecidos é possível a simulação dos mesmos em diferentes pontos de vistas.

A justificativa para esse trabalho está no fato de que as tarefas destinadas aos sistemas robóticos móveis estão cada vez mais complexas e necessitam que os robôs possuam sistemas de localização mais precisos e que possam construir mapas do ambiente mais representativos e detalhados, contendo informações métricas e semânticas.

Por esse ponto de vista, nota-se que os trabalhos enquadrados na abordagem direta de SLAM Visual encontrados na literatura não estão totalmente livres de aproximações e restrições, funcionando de forma aceitável apenas em ambientes controlados. Na verdade observa-se que ambientes internos ou externos possuem diversas formas geométricas, além de pontos e planos, que podem ser utilizados para estimar a localização do robô e de mapas mais completos, detalhados e representativos.

1.4 Objetivos

1.4.1 Objetivos Gerais e Contribuições

A contribuição principal deste trabalho será a apresentação de uma abordagem de SLAM Visual Direto, cujo modelo de transformação e registro de imagem sejam imple-

mentados de forma desacoplada (independentes do resto do sistema), fazendo com que o sistema não seja restrito à utilização de apenas segmentos de planos.

Para isso foi necessário o desenvolvimento modular de uma nova técnica de registro de imagem, adequada aos diversos tipos de formas geométricas que serão utilizadas, e de um modelo de transformação geométrica menos restritivo. Esse modelo de transformação geométrica deverá ser capaz de calcular como certos pixels apareceram após determinados deslocamentos do robô, utilizando formas geométricas mais complexas que planos.

Uma outra contribuição importante será a implementação do modelo de transformação geométrica de forma que torne possível alterar as equações paramétrica que representem as formas geométricas utilizadas, sem muitas alterações no código ou na matemática envolvida. Isso será possível através da implementação modular e independente do registro de imagem e do modelo de transformação. Eles serão independentes do sistema de otimização, ou seja, a formulação matemática que permite usar registro de imagem para calcular o erro de reprojeção é independente do método de otimização.

Também será possível testar diferentes famílias de técnicas de otimização ou heurísticas afim de achar o que melhor se adequa ao problema. Por fim, espera-se obter um sistema mais abrangente, não restrito a primitivas geométrica como pontos, retas ou planos que retorne mapas em três dimensões mais representativos com um número otimizado de parâmetros necessários para tal representação.

1.4.2 Objetivos Específicos

- Estudar detalhadamente as técnicas de SLAM Visual que não extraem primitivas geométricas da imagem.
- Implementar um modelo de transformação geométrica que possa tratar o problema de oclusão de pixels e simular a aparência de diferentes formas geométricas dependendo do ponto de vista.
- Implementar esse modelo de forma modular para tornar possível a mudança dos parâmetros geométricos sem grandes alterações no sistema.
- Adaptar a técnica de registro direto de imagem para estimar o erro de reprojeção em forma geométricas mais complexas que planos.
- Implementar uma solução de SLAM visual, sem correspondência de características, baseada na técnica de registro de imagem desenvolvida.
- Abordar diferentes métodos de otimização para tratar o erro de reprojeção nas imagens.
- Testar a abordagem desenvolvida e comparar a mesma com técnicas já consagradas

na literatura.

1.5 Organização do Presente Documento

Este documento é uma tese de doutorado e será organizada da seguinte forma:.

- Capítulo 1: Encontra-se uma visão geral e introdutória do problema, classificando o SLAM Visual em duas abordagens, apresenta-se um estado da arte, a motivação, justificativa e os objetivos do presente trabalho;
- Capítulo 2: Encontraremos nesse capítulo uma fundamentação teóricas sobre temas e técnicas importantes para se entender o presente trabalho;
- Capítulo 3: Neste capítulo encontra-se a metodologia e desenvolvimento da técnica de SLAM desenvolvida;
- Capítulo 4: Neste capítulo encontra-se os resultados dos diversos experimentos realizados;
- Capítulo 5: Por fim encontra-se um capítulo de conclusão e perspectivas do presente trabalho.

Capítulo 2

Fundamentação Teórica

O objetivo deste Capítulo é apresentar a fundamentação teórica desse trabalho, ou seja, técnicas matemáticas e de visão computacional necessárias para desenvolvimento da técnica proposta, das quais se destacam: Localização e Mapeamento robótico, SLAM Visual, registro de imagem, estrutura a partir do movimento e métodos de otimização.

2.1 Localização Robótica

Uma importante etapa no processo de navegação de robôs móveis autônomos é a tarefa de localização. A partir de um conhecimento acumulado sobre o ambiente e utilizando as leituras atuais dos sensores, o robô deve ser capaz de determinar e manter atualizada a sua posição e orientação em relação a este ambiente, mesmo que os sensores apresentem erro e/ou ruído. Em outras palavras, localizar um robô consiste em determinar sua pose ou postura (posição e orientação) no espaço de trabalho a cada instante de tempo.

Os métodos de localização podem ser classificados duas grandes categorias: métodos de localização relativa, onde a pose do robô é estimada em relação a uma pose inicial e os métodos de localização absoluta, que dão a pose global do robô e não necessitam das posições calculadas anteriormente.

O modelo mais simples de localização envolve apenas parâmetros internos do robô, e por isso não depende da obtenção de informações do ambiente ao seu redor. Normalmente, as informações odométricas são obtidas através de sensores acoplados às rodas dos robôs, de forma a calcular o deslocamento dos mesmos a partir do movimento destas. Em se tratando de localização absoluta, Borenstein e Feng (1996) definem marcos como elementos distintos que o robô é capaz de reconhecer através de seus sensores. Geralmente os marcos estão em posições fixas e conhecidas em relação ao referencial do ambiente.

As informações dos sensores são utilizadas para reconhecer os marcos e determinar as suas posições com relação ao referencial do robô. Conhecendo a posição dos marcos com relação ao referencial do ambiente e com relação ao referencial do robô, é possível determinar a pose do robô com relação ao referencial do ambiente.

A figura 2.1 ilustra uma etapa do processo de localização. Nessa ilustração, o sistema de localização do robô móvel detecta o ponto de interseção entre as retas existentes no piso de um ambiente a partir de imagens providas por uma câmera. Sabendo a posição deste ponto no sistema de coordenadas global, o sistema calcula a pose do robô neste sistema de coordenadas.

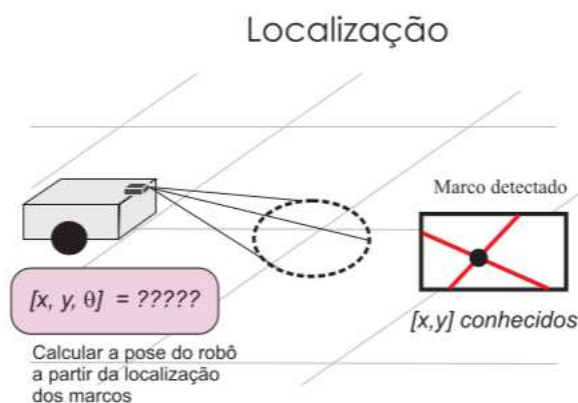


Figura 2.1: Etapa de um processo de localização baseado em marcos ou características.

Na prática, existem diversos fatores que causam erros nesse tipo de dispositivo como a imprecisão mecânica do sistema e irregularidades do terreno em que o robô atua. Outro fator que dificulta e pode até inviabilizar a localização do robô a partir da odometria é o fato de que os pequenos erros causados durante o cálculo da posição do robô vão se acumulando ao longo do tempo.

Para melhorar esse contexto pode se utilizar fusão sensorial, ou seja, sensores exteroceptivos, que retornam informações sobre o ambiente, juntamente com sensores proprioceptivos, que retornam dados internos como odometria, podem ser combinados para melhorar as estimativas (a figura 2.2 esquematiza esse processo). Nesse contexto, técnicas probabilísticas como filtro de Kalman têm sido amplamente utilizadas na solução desse problema [Choset et al. 2005].

A fusão sensorial melhora as estimativas, porém não é possível uma localização perfeita do robô devido a presença de ruídos nas medidas. Assim, o grande desafio na solução do problema de localização está no fato de que tanto as informações sobre o ambiente como os dados fornecidos pelos sensores são normalmente limitados e imprecisos.

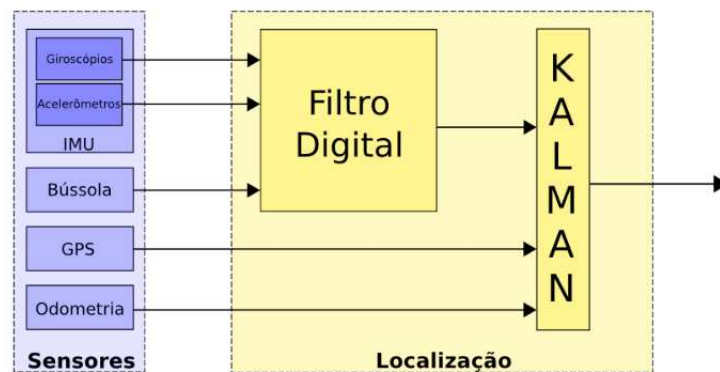


Figura 2.2: Exemplo de fusão sensorial com filtro de Kalman

2.1.1 Navegação Visual

A navegação, no contexto da robótica, pode ser basicamente descrita como o processo de determinação de um caminho adequado entre um ponto de partida e um ponto de destino para que um robô possa viajar entre eles de forma segura [Choset et al. 2005, Latombe 1991]. Diferentes sensores têm sido utilizados para esta finalidade e isto levou a um variado leque de soluções. Em particular, nas últimas três décadas, a navegação visual para robôs móveis tornou-se uma fonte de inúmeras contribuições para aumentar o âmbito de aplicação de veículos móveis autônomos, tais como: vigilância, patrulhamento, busca e salvamento, inspeção, missões aéreas, mapeamento, detecção de incêndio.

Os seres humanos confiam principalmente nos seus sensores externos, os olhos, para realizar tarefas de localização e navegação. Do mesmo modo em robótica, sensores externos (sensores que medem aspectos do mundo externo) podem render mais informações que os sensores internos. Um exemplo de sensor externo usado nos robôs são as câmeras.

Um atributo essencial para que um robô possa navegar de forma autônoma por um ambiente é que ele possua um sistema de percepção confiável. Além da confiabilidade da percepção, para a aceitação generalizada de aplicações, é necessário que as tecnologias utilizadas proporcionem uma solução com preço acessível, ou seja, que os componentes sejam de baixo custo. Uma solução interessante é a utilização dos sensores óticos. Devido à utilização maciça de câmeras digitais pessoais, câmeras em computadores e celulares, o preço do sensor de imagem diminuiu significativamente e tornaram-se muito atraentes. Além disso, as câmeras podem ser usadas para resolver uma série de problemas-chave na robótica e em outras operações automatizadas pois fornecem uma variedade de informações do ambiente, consomem pouca energia e são facilmente integradas ao hardware do robô.

Os principais desafios são tirar proveito deste poderoso e barato sensor e criar algorit-

mos confiáveis e eficazes que possam extrair as informações necessárias para a resolução de problemas na robótica.

2.1.2 Navegação Sem Mapa

No processo de navegação, um problema surge quando o mapa não está disponível (mapless navigation). A navegação sem mapa não necessita do conhecimento do ambiente a ser explorado e os movimentos do robô dependem dos elementos observados no ambiente (paredes, portas, mesas, etc.). Para a solução deste problema dois tipos principais de técnicas devem ser citadas: navegação baseada em aparência e navegação baseada em fluxo ótico.

Navegação baseada em aparência é uma técnica baseada no armazenamento de imagens que são usadas como modelos para o robô se auto-localizar e, além disso, são utilizadas para fazer correspondências de observações de marcos. Exemplos iniciais dessa abordagem são os trabalhos de Matsumoto et al. (1996) e Jones et al. (1997) onde uma sequência de imagens é armazenada para serem usadas como uma memória.

Soluções baseadas em fluxo ótico estimam movimento de objetos ou características dentro de uma sequência de imagens e os trabalhos iniciais nesta vertente são os de Horn e Schunck (1981) e Lucas e Kanade (1981). Uma abordagem interessante desenvolvida por Santos-Victor e Garibaldi (1993) emula o comportamento de abelhas voando. Neste trabalho um robô contendo duas câmeras apontadas para as paredes se desloca em um corredor e calcula as diferenças dos fluxos óticos a partir das imagens com o intuito de se mover sempre na direção do fluxo ótico de menor amplitude. O principal problema desta técnica é que as paredes precisam ser texturizadas para apresentar bons fluxos.

Com o avanço na utilização de câmeras, foram dadas outras soluções para o problema da navegação, dentre elas a odometria visual. Esta abordagem está sendo muito utilizada em ambientes outdoor onde a odometria mecânica falha muito devido à derrapagem das rodas. Um exemplo é o trabalho apresentado por Campbell et al. (2004) que calcula a odometria visual utilizando fluxo ótico para diversos ambientes como areia, cascalho e gelo e mostra a eficiência da técnica.

2.2 Mapeamento Robótico

A representação do ambiente ou mapeamento robótico consiste em obter informações sobre o ambiente ao redor de um robô mapeador, usando sensores, e processar essas informações a fim de achar uma representação desse ambiente. As primeiras investigações

mais aprofundadas na linha de mapeamento robótico surgem em meados da década de 80, em trabalhos como [Chatila e Laumond 1985, Elfes 1987, Elfes 1989]. Nessa mesma década surgiram várias formas de mapeamento usando diversas estruturas geométricas e topológicas, dos quais destacam-se o trabalho de Merat e Wu (1986), onde foi construído uma representação do ambiente por meio de características como pontos, arestas e cantos. No trabalho de Vincent (1987) é utilizada uma representação poligonal de espaços ocupados e visão estéreo. já em [Kriegman e Binford 1987] propõe-se a utilização modelos geométricos hierárquicos. Por fim podemos citar o trabalho de Kuipers e Byun (1988) que propõe uma representação com nível de abstração maior, baseada na topologia do ambiente.

Destaca-se também trabalhos mais recentes como [Civera et al. 2008, Fu e Yang 2009, Paz e Neira 2008] que constroem mapas baseados em características extraídas das imagens usando visão estéreo e monocular. Por fim destacamos o trabalho de Lacroix e Berger (2008) que defende a ideia de que as próximas investigações sobre mapas robóticos devem se concentrar em mapas semânticos, visto que a complexidade das tarefas designadas aos sistemas robóticos tendem a aumentar e assim as representações do ambiente devem ser mais avançadas.

É notável que a maioria das soluções de SLAM visual utilizam pontos como características do ambiente, logo o mapa é formado pelas coordenadas 3D desses pontos extraídos das imagens. Estes mapas, conhecido como mapa de características, normalmente, não fornecem informações tratáveis e úteis para os humanos. Já na abordagem Direta é comum a utilização de planos o que também deixa o mapa pouco representativo e ainda torna o problema restrito a ambiente que contenham planos.

Na abordagem direta, para se obter os parâmetros que representam o mapa, utiliza-se uma técnica conhecida como registro de imagem, que é basicamente o mapeamento entre imagens da mesma cena capturadas de pontos de vistas diferentes. Esse mapeamento representa a transformação geométrica ou fotométrica ocorrida entre a captura das duas imagens. No SLAM Visual essa técnica é utilizada para calcular o erro de reprojeção entre a imagem de referência e a imagem capturada após o deslocamento do robô, que normalmente apresentam discrepâncias.

Existem diversas formas de realizar o registro de imagem, das quais destaca-se registro direto de imagem, onde o erro de reprojeção é calculado utilizando apenas as intensidades quantizadas, ou discretizadas, dos pixels da imagem e o erro de reprojeção é igual ao quadrado da soma das diferenças entre os valores dos pixels correspondentes nas duas imagens.

Para melhor formalizar o problema de mapeamento robótico, devemos estabelecer al-

gumas hipóteses (ou restrições) básicas. A primeira hipótese é que o robô possui sensores proprioceptivos e exteroceptivos que o permitem coletar dados sobre si mesmo (posição e orientação) e sobre o ambiente, respectivamente. A segunda é que este sistema de percepção deve permitir, a todo momento, o conhecimento exato da posição e orientação do robô em relação a algum sistema de coordenadas ou referencial fixo global adotado.

As maiores dificuldades encontradas no desenvolvimento de algoritmos para esta e também outras tarefas autônomas, segundo Vasudevan et al. (2007) são:

- A incerteza nos dados dos sensores;
- Aquisição de informações parciais sobre o ambiente, a cada instante de tempo;
- A Produção de uma grande quantidade de dados;
- A Dinâmica do ambiente.

Normalmente são adotadas muitas simplificações, como por exemplo tratar o ambiente como estático.

Um outro problema que aparece no processo de mapeamento diz respeito à localização do robô mapeador que se desloca pelo ambiente. Nesse processo também encontra-se incertezas que devem ser tratadas. Por exemplo, erros acumulados de odometria.

Existem trabalhos na literatura propondo vários tipos de mapas de acordo com as diferentes situações, os quais podem ser classificados em duas categorias: mapas espaciais e mapas semânticos.

2.2.1 Mapas com informações espaciais

Segundo a definição dada em [Thrun 2002], o mapeamento com robôs móveis é o processo de obtenção de uma representação espacial do ambiente físico. Dentre os mapas espaciais destacamos o mapa métrico e o mapa topológico.

Os mapas métricos são representações mais exatas do ambiente em duas ou três dimensões, enquanto que os topológicos são abstrações da distribuição espacial representando locais distintos no ambiente e suas conexões físicas, muitas vezes através de um grafo de conectividade.

- Os mapas métricos são mais fáceis de se obter e atualizar e sua representação é mais intuitiva. Esse tipo de mapa é mais adequado para computação de trajetórias curtas e reconhecimentos de locais baseando-se na geometria do mesmo. Porém é necessário a localização precisa do robô no processo de mapeamento.

- Os mapas topológicos permitem um planejamento mais complexo de trajetória de forma mais eficiente, visto que o espaço de busca possui uma dimensão bem menor. Nesse tipo de mapeamento não é necessário ter a localização exata do robô mapeador. Porém o reconhecimento de locais é difícil devido à presença de ambiguidades no processo de mapeamento.

Mapas Topológicos

Os mapas topológicos são aqueles representados computacionalmente através de um grafo, que, por sua vez, descreve um arranjo de nós (ou vértices) conectados entre si por uma série de arestas (elos ou arcos). Normalmente, os grafos descrevem os espaços livres para a execução de tarefas. Os nós correspondem a regiões que possuem informações sensoriais homogêneas e os arcos refletem a conexão entre estas regiões.

A utilização de um grafo para descrever um ambiente parece ser, intuitivamente, uma boa opção pois além de ser uma estrutura compacta para armazenamento na memória do sistema computacional do robô podem ser usados para resolução de problemas de alto nível como o planejamento de tarefas.

O grande problema desta representação consiste na ausência de um padrão de definição de quais estruturas serão associadas aos vértices e quais relações serão descritas para serem utilizadas como elos. Ainda, a localização do robô usando mapas topológicos é abstrata, ou seja, não há como definir explicitamente a posição e orientação do robô. No entanto, pode-se afirmar em qual nó do grafo ou em qual região do ambiente ele se encontra. A figura 2.3 ilustra intuitivamente o processo de construção de um mapa topológico de um determinado ambiente representado por sua planta baixa.

Mapas Métricos e Mapas de Características

Por representar de maneira detalhada um ambiente com informações espacial, os mapas métricos são usados para navegações que exigem um posicionamento mais preciso do robô no meio, mas a obtenção de uma consistência global do mapa exige o tratamento de problemas como o de closed loops, que aparece quando o robô retorna a uma posição previamente visitada. Além disso, a grande quantidade de informação presente no mapa torna computacionalmente custosa qualquer manipulação do modelo, como o planejamento de trajetória.

Com a introdução dos algoritmos probabilísticos na década de 90, houve maior interesse no uso de mapas métricos e o foco principal deixou de ser as formas de representações e passou a ser os modelos matemáticos utilizados no processo de estimação dos

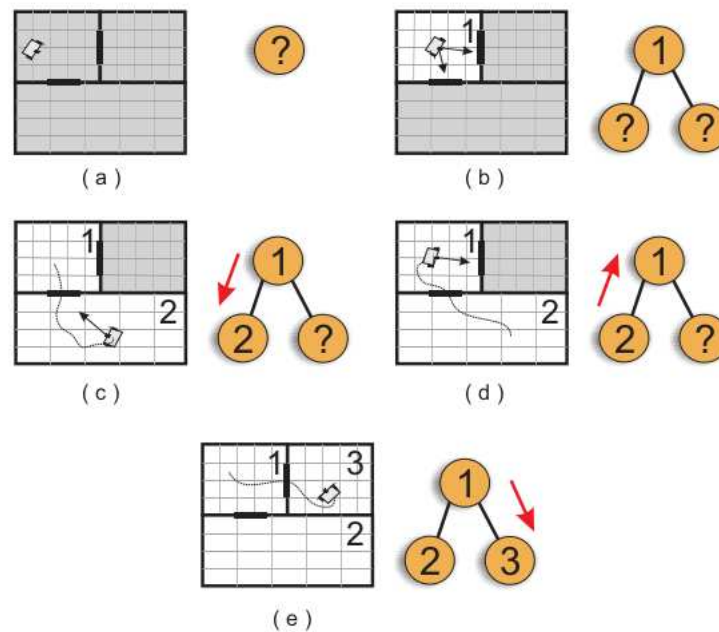


Figura 2.3: Esquema de construção de um mapa topológico.

parâmetros necessários. Surgiram então os Mapas de Características.

- Mapas de características

Os mapas de características armazenam informações de elementos importantes encontrados em localizações específicas do ambiente. Uma característica, ou feature, pode ser entendida como algo facilmente notável pelo sensores do robô. Além disso, as características devem ser distinguíveis entre si e devem poder ser reobservadas de diferentes posições do ambiente. Em outras palavras, se um robô detecta uma característica em um instante t e mais tarde, em um instante $t + k$, ele a reobserva de uma posição diferente, então esta característica tem que ser identificada de forma fácil e única. Na construção de mapas de características usando imagens é comum identificar partes com propriedades especiais, por exemplo, pontos, linhas, planos e formas (figura 2.4).

A maioria das soluções de SLAM visual (abordagens clássicas) utiliza pontos como características do ambiente. Este fato ocorre devido aos vários algoritmos existentes para detectar pontos notáveis em imagem, associado ao fato de pontos serem objetos geométricos simples.

Uma das técnicas mais adotadas para a obtenção de pontos de interesse é o detector Harris (Harris Corner Detector) devido à sua forte invariância à rotação, escala, iluminação e ruído de imagem [Harris e Stephens 1998]. Outra técnica bastante utilizada é o

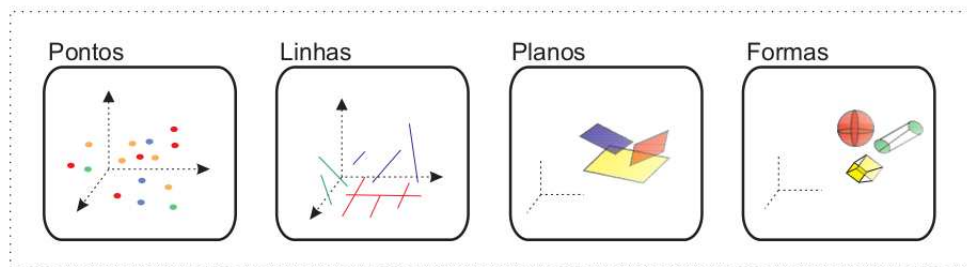


Figura 2.4: Principais elementos utilizados em um mapa de características

algoritmo SIFT (Scale-Invariant Feature Transform) que detecta pontos-chave distintivos na imagens e calcula um descritor para estes pontos [Lowe 1999].

Os pontos de interesse extraídos por este algoritmo são ditos invariante a escala, rotação e parcialmente às mudanças na iluminação. Uma variação do algoritmo SIFT, que também é bem utilizado, é chamado de SURF (Speeded Up Robust Features) e se baseia na matriz Hessiana devido à sua precisão e baixo custo computacional [Bay e Gool 2006]. Para maiores detalhes pode-se citar o trabalho de Gil A. e Reinoso (2006) que apresenta uma comparação entre os detectores e descritores de pontos mais utilizados no SLAM.

A construção de mapas de características baseados em pontos é bem aceita, porém estes mapas normalmente não fornecem informações tratáveis e úteis para os humanos. Assim, faz-se necessário a construção de mapas de características utilizando outras formas geométricas mais complexa e esta é uma das motivações deste trabalho. Planos são apenas o primeiro passo para a construção de um mapa com maior nível de representação do ambiente. Em ambientes estruturados muitos objetos podem ser descritos graças a esta primitiva.

Ultimamente as atenções se voltaram para os chamados mapas de objetos, que resgatam uma preocupação em representar conceitos abstratos sobre as informações espaciais em mapas métricos. Ou seja, existe uma transição de representações métricas puramente espaciais para representações que incorporam também informação semântica. O interesse pela incorporação da semântica das relações espaciais no ambiente não é recente [Galindo et al. 2005]. Porém as soluções anteriores nunca alcançaram muito destaque na literatura pela falta de um modelo adequado para implementar essa representação, e muitas vezes a informação semântica incorporada estava relacionada à manipulação simbólica de elementos do modelo.

O que está sendo proposto atualmente é trabalhar tanto com a simples classificação da informação espacial em categorias semânticas [Mozos et al. 2007], quanto com a estrutura relacional dos dados para inferências direcionadas a objetos [Limketkai et al. 2005].

2.2.2 Mapas com Informações Semânticas

Na navegação robótica e em muitas outras aplicações, os robôs móveis podem atingir seus objetivos com melhor desempenho se forem capazes de reconhecer lugares específicos e distinguir com maior exatidão um lugar do outro.

Segundo Tapus e Siegwart (2006), atuar em ambientes externos, não-estruturados e muito extensos exige uma abordagem distinta da que se popularizou na década passada. Propõe-se que o robô deva ser capaz de estruturar os dados espaciais por meio da categorização em informação semântica. A interação humano-robô também pode ser favorecida se os robôs possuírem representações de conhecimento similares às nossas, por meio da informação semântica.

Mesmo com a existência de alguns trabalhos recentes que se preocupam com a obtenção de informação semântica, pouco tem sido feito para analisar as propostas dessa área. Diversas representações têm sido propostas, sob diferentes denominações tais como mapas cognitivos [Tapus e Siegwart 2006, Vasudevan et al. 2007], mapas de objetos [Martin e Thrun 2002, Anguelov et al. 2004, Limketkai et al. 2005], mapas de atividades [Wolf e Sukhatme 2006, Lookingbill et al. 2005], entre outros.

Agora que temos uma certa noção sobre o que é localização e mapeamento robótico e suas vertentes podemos seguir em frente e falar sobre uma técnica que resolve esses dois problemas (localização e mapeamento) de forma simultânea.

2.3 Localização e Mapeamento Simultâneo (SLAM)

Um pré-requisito fundamental para um robô móvel é a autonomia, ou seja, a capacidade de navegação em ambientes desconhecidos sem ajuda humana. Para realizar tal tarefa, é necessário uma percepção sobre o ambiente no qual está inserido. Então o robô precisa de sensores que possam extrair o máximo de informações úteis sobre o ambiente ao seu redor e utilizar essas informações para se localizar no mesmo. O SLAM é aplicado quando não se tem nenhuma informação a priori do mapa ou da pose do robô e se dispõe apenas dos sinais de medida dos sensores e sinais de controle enviados ao robô, juntamente com seu modelo matemático. Nesse caso é preciso tratar alguns problemas, são eles:

- Como tratar as incertezas associadas ao movimento dos robôs e aos sensores de medição.
- Como tratar a questão da simultaneidade no mapeamento e localização.

- Como obter boas estimativas da pose do robô e do mapa do ambiente, visto que ambos estão fortemente acoplados.

No início da década 90, as abordagens probabilísticas dominavam as soluções de SLAM. A grande maioria dessas abordagens utilizaram Filtro de Kalman, que opera em modo de predição-atualização levando em consideração as propriedades estatísticas do ruído presente no sistema robótico. Usa-se um modelo interno desse sistema e um esquema de realimentação (sensores) realiza as medições. A principal razão para a popularidade destas técnicas probabilísticas reside no fato de que o mapeamento robótico é caracterizado por incertezas e ruídos dos sensores. Logo, algoritmos com base em probabilidade podem ser usados para modelar explicitamente as diferentes fontes de ruído e seus efeitos sobre as medidas [Thrun 2002].

Após as décadas 90 a ideia de utilizar câmeras de vídeo como principal sensor do robô se fortaleceu bastante e com isso surgiu uma nova abordagem denominada SLAM Visual. A utilização de câmeras de vídeo abriu muitas possibilidades para o tratamento das incertezas no SLAM.

2.3.1 Formulação Probabilística do SLAM Clássico e Filtro de Kalman

O filtro de Kalman estendido trabalha com um modelo segundo o Sistema 3.1, cujas variáveis são descritas na tabela 3.1. ϵ_t and δ_t e δ_t são ruídos gaussianos de média zero.

$$\begin{cases} s_t = p(s_{t-1}, u_{t-1}, \epsilon_{t-1}) \\ z_t = h(s_t) + \delta_t \end{cases} \quad (2.1)$$

A cada período de amostragem, o EKF calcula a melhor estimativa do vetor de estados em duas fases:

- a fase de **predição** usa o Sistema 3.2 para prever o estado corrente baseado no estado anterior e nos sinais de entrada aplicados;
- a fase de **atualização** usa o Sistema 3.3 para corrigir a predição do estado pela verificação de sua compatibilidade com as medidas atuais dos sensores.

Tabela 2.1: Símbolos das equações (2.1) até (2.6)

$\mathbf{s}_t$	vetor de estados (ordem n) no instante t
$p(\cdot)$	modelo não linear do sistema
$\mathbf{u}_{t-1}$	sinais de entrada (ordem l), no instante $t-1$
ε_{t-1}	ruído do processo (ordem q), no instante $t-1$
$\mathbf{z}_t$	vetor de medições (ordem m) retornadas pelo sensor
$h(\cdot)$	modelo não linear do sensor
δ_t	ruído de medição
$\bar{\mu}_t, \mu_t$	média (ordem n) do vetor de estados $\mathbf{s}_t$, antes e depois da fase de atualização
$\bar{\Sigma}_t, \Sigma_t$	covariância ($n \times n$) do vetor de estados $\mathbf{s}_t$
$\mathbf{G}_t$	matriz Jacobiana ($n \times n$) que lineariza o modelo do processo $p(\cdot)$
$\mathbf{V}_t$	matriz Jacobiana ($n \times q$) que lineariza o ruído do processo ε_t
$\mathbf{M}_t$	covariância ($q \times q$) do ruído de processo ε_t
$\mathbf{K}_t$	matriz ganho de Kalman ($n \times m$)
$\mathbf{H}_t$	matriz Jacobiana ($m \times n$) que lineariza o modelo do sensor $h(\cdot)$
$\mathbf{Q}_t$	covariância ($m \times m$) do ruído de medição δ_t

$$\begin{cases} \bar{\mu}_t = p(\mu_{t-1}, \mathbf{u}_{t-1}, 0) \\ \bar{\Sigma}_t = \mathbf{G}_t \Sigma_{t-1} \mathbf{G}_t^T + \mathbf{V}_t \mathbf{M}_t \mathbf{V}_t^T \end{cases} \quad (2.2)$$

$$\begin{cases} \mathbf{K}_t = \bar{\Sigma}_t \mathbf{H}_t^T (\mathbf{H}_t \bar{\Sigma}_t \mathbf{H}_t^T + \mathbf{Q}_t)^{-1} \\ \mu_t = \bar{\mu}_t + \mathbf{K}_t (\mathbf{z}_t - h(\bar{\mu}_t)) \\ \Sigma_t = (\mathbf{I} - \mathbf{K}_t \mathbf{H}_t) \bar{\Sigma}_t \end{cases} \quad (2.3)$$

onde:

$$\mathbf{G}_t = \left. \frac{\partial p(\mathbf{s}, \mathbf{u}, \varepsilon)}{\partial \mathbf{s}} \right|_{\mathbf{s}=\mu_{t-1}, \mathbf{u}=\mathbf{u}_{t-1}, \varepsilon=0} \quad (2.4)$$

$$\mathbf{V}_t = \left. \frac{\partial p(\mathbf{s}, \mathbf{u}, \varepsilon)}{\partial \mathbf{s}} \right|_{\mathbf{s}=\mu_{t-1}, \mathbf{u}=\mathbf{u}_{t-1}, \varepsilon=0} \quad (2.5)$$

$$\mathbf{H}_t = \left. \frac{\partial h(\mathbf{s})}{\partial \mathbf{s}} \right|_{\mathbf{s}=\mu_{t-1}} \quad (2.6)$$

Na tarefa de SLAM, além de estimar a pose do robô, também são estimadas as características de todos os marcos encontrados pelo caminho. Assim, torna-se necessário incluir as características dos marcos no vetor de estado. Se i_c é o vetor de características do i -ésimo marco e existem k marcos, então o vetor de estados é dado pela equação 2.7.

$$\mathbf{s}_t = \left[x_t y_t \theta_t^1 c_t^1 \dots^k c_t^k \right]^T \quad (2.7)$$

Quando o número de marcos (k) é conhecido *a priori*, a dimensão do vetor de estados é estática. Caso contrário, ele cresce quando um novo marco é encontrado.

2.3.2 Sistemas de Visão computacional e SLAM Visual

No capítulo anterior, nas seções 1.1.1 e 1.1.2, já vimos o que é SLAM Visual e suas abordagens, mas recapitulando: o que se quer resolver com o SLAM visual é como fazer a localização e mapeamento utilizando imagens capturadas por sistemas de percepção visual e técnicas de visão computacional que constroem representações do mundo a partir dos pixels dessas imagens. Os principais tipos de sistemas de percepção visual (ou sistemas de visão) utilizados pelos robôs estão esquematizados na figura 2.5.



Figura 2.5: Sistemas de visão mais utilizados no SLAM

O trabalho de Ayache e Sander (1991) foi um dos primeiros onde foi proposta a utilização de câmeras como sensores em robôs móveis. Depois surgiram trabalhos como o de Davison e Murray (2002), que utilizou técnicas de visão computacional no problema de SLAM e o trabalho de Davison et al. (2007), onde se destacou a ideia de SLAM Visual Monocular, ou seja, usar apenas uma câmera para fazer SLAM Visual e com isso tornar o problema de calibração de câmera mais simples, reduzir o custo computacional e material, tornar o robô mais leve e mais compacto (pré-requisito necessário em robôs aéreos). A desvantagem desse sistema reside no fato de que para se obter informações sobre a profundidade da cena é necessário a captura de imagens subsequentes após deslocamento do sistema.

Outro sistema de visão muito utilizado para realizar o SLAM Visual é a visão estéreo, que por sua vez proporciona informações sobre a profundidade da cena sem a necessidade de movimento do robô e de imagens subsequentes. Isso é possível através do mapa de disparidades entre os pixels das duas imagens (direita e esquerda) capturadas simultaneamente pelo sistema de visão.

Essas vantagens, em relação ao sistema de visão estéreo, fizeram com que muitas pesquisas fossem desenvolvidas para tratar o problema do SLAM Visual Monocular e que essa abordagem fosse escolhida para ser utilizada nesse trabalho.

Usar câmeras como sensor de percepção do ambiente nos retorna uma fonte de informação extremamente rica sobre o mesmo, em comparação com outros sensores [Artieda et al. 2009], além das diversas vantagens já discutidas no capítulo anterior. Porém a forma como é abordada a estimativa dos parâmetros é mais custosa, computacionalmente, que as abordagens clássicas de SLAM devido a utilização de técnicas de visão computacional que, normalmente, precisam de muito processamento. No presente trabalho foram utilizados conceitos de Registro de Imagem e Estrutura a Partir do Movimento.

2.4 Registro de Imagem e SFM

Dentre as várias técnicas de visão computacional que podem ser utilizadas na estimação dos parâmetros necessários na localização e mapeamento simultâneo, destacam-se o registro de imagem e a estrutura a partir do movimento ou apenas SFM - *Struct from motion*. No presente trabalho ambas as técnicas foram utilizadas. A técnica de registro de imagem implementada para calcular o erro de reprojeção entre as imagens, enquanto alguns conceitos do SFM foram utilizados no modelo de transformação geométrica.

2.4.1 Registro de Imagem

A idéia geral do registro de imagem é encontrar uma transformação que faça uma imagem ou uma região dessa imagem se alinhe corretamente com a região análoga em outra imagem. A figura 2.6 exemplifica algumas das possíveis transformações que podem representar o alinhamento de 2 ou mais imagens.

Registro de imagens pode ser visto como um processo de sobreposição de duas ou mais imagens da mesma cena, tomadas em momentos diferentes, a partir de diferentes pontos de vista, e/ou por sensores diferentes [Barbara e Jan 2003]. Segundo BROWN (1992) o registro de imagem pode ser definido como o mapeamento entre duas imagens no espaço no que diz respeito a intensidade, ou seja, se definimos essas imagens como

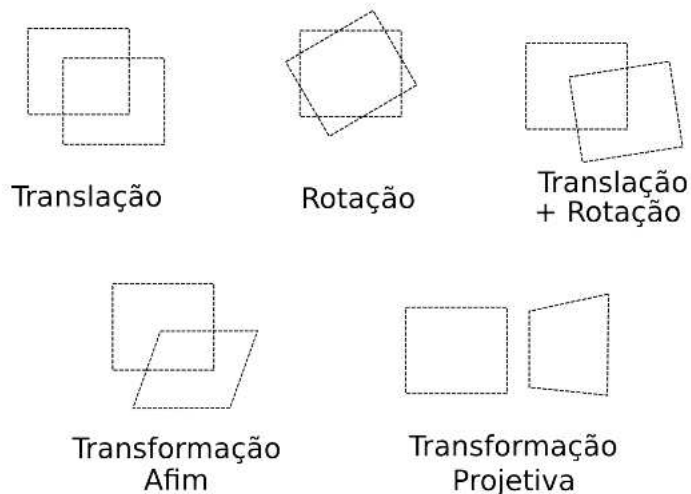


Figura 2.6: Exemplo de transformações que podem ser aplicadas em uma imagem

dois vetores 2D:

$$I_1(x,y) \quad (2.8)$$

$$I_2(x,y) \quad (2.9)$$

Então o mapeamento entre as imagens pode ser expresso da seguinte maneira:

$$I_2(x,y) = g(I_1(f(x,y))) \quad (2.10)$$

Onde $I_1(x,y)$ representa a intensidade do pixel na coordenada x,y , f é a transformação que mapeia as coordenadas espaciais e g é a transformação de intensidade ou radiométrica. Assim o problema do registro de imagem é encontrar as transformações f e g ideais para que as imagens sejam combinadas ou para efeito de determinação dos parâmetros da transformação ocorrida [BROWN 1992].

Registro de imagens é um passo crucial em todas as tarefas de análise de imagem em que a informação final é obtida a partir da combinação de várias fontes de dados, como na fusão de imagens, detecção de mudanças na cena, restauração de imagens em vários canais, entre outras. Suas aplicações podem ser divididas em quatro grupos principais de acordo com o modo de aquisição da imagem:

- **Pontos de vista diferentes:** As imagens da mesma cena são adquiridas de diferen-

tes pontos de vista. O objetivo é obter uma maior visualização 2D ou uma representação 3D da imagem digitalizada da cena.

- **Épocas diferentes:** As imagens da mesma cena são adquiridas em épocas diferentes e, possivelmente, sob diferentes condições. O objetivo é encontrar e avaliar as mudanças na cena que aparecem entre as aquisições das imagens consecutivas.
- **Sensores diferentes:** As imagens da mesma cena adquiridas por diferentes sensores. O objetivo é integrar as informações obtidas a partir de fontes diferentes para ganhar uma representação de cena mais complexa e detalhada.
- **Registro através de um modelo:** Imagens de uma cena e um modelo da mesma são registradas. O modelo pode ser uma representação da cena no computador. O objetivo é comparar a imagem capturada da cena com o modelo da mesma.

Existem duas metodologias principais de registro de imagem: a baseada em características e a baseada em intensidades da imagem.

Métodos Baseados em Características

Segundo Barbara e Jan (2003) os métodos de registro de imagem baseados em características funcionam, basicamente, em quatro etapas:

1. **Detecção de características:** Objetos distintos e salientes (fronteira fechada de regiões, bordas, contornos, cruzamentos de linhas, curvas, etc) são manualmente ou automaticamente detectados. Para posterior processamento, estas características podem ser representadas pelos seus pontos representativos (centros de gravidade, finais de linha, pontos distintos), que são chamados na literatura de pontos de controle.
2. **Correspondência de características:** Nesta etapa a correspondência entre características detectadas na imagem sensorial e os detectados na imagem de referência é estabelecida. Vários descritores de características e medidas de similaridade junto com as relações espaciais entre as características são utilizados para esse fim.
3. **Estimação do modelo de transformação:** Os tipos e os parâmetros das funções de mapeamento que alinham as imagens adquiridas com imagens de referência são estimados. Os parâmetros das funções de mapeamento são calculados por meio da correspondência de características estabelecida.
4. **Reamostragem e transformação:** A imagem sensorial é transformada por meio das funções de mapeamento. Os valores das imagens em coordenadas não inteiras, são calculadas pela técnica de interpolação adequada.

A implementação de cada etapa de registro tem seus problemas típicos. Em Primeiro lugar é preciso decidir que tipo de características é apropriado para uma determinada tarefa. Elas devem ser objetos distintos, frequentes, espalhados nas imagens e devem ser facilmente detectáveis. Os conjuntos de características detectadas nas imagens devem conter elementos suficientes em comum, mesmo nas situações em que as imagens não cubram exatamente a mesma cena, quando há oclusão ou mudanças inesperadas. Os métodos de detecção deverão ter boa precisão na localização das características e não devem ser sensíveis à degradações nas imagens. Em um caso ideal, o algoritmo deve ser capaz de detectar as mesmas características em todas as projeções da cena, independente da deformação de uma projeção em particular.

Na etapa de correspondência de características o objetivo é encontrar a correspondência entre as mesmas usando suas relações espaciais ou seus descritores. Os métodos baseados nas relações espaciais entre as características são normalmente aplicados se as características detectadas são ambíguas ou quando suas vizinhanças são localmente distorcidas. Na outra alternativa, as correspondências entre características podem ser estimadas utilizando os descritores. Características com descritores mais similares são consideradas correspondentes.

Problemas causados por uma detecção de características incorreta ou por degradações de imagem podem surgir: características diferentes podem ser consideradas correspondentes devido às condições diferentes da imagem e/ou devido às diferentes sensibilidades espectrais dos sensores. A escolha dos descritores de características e as medidas de similaridade têm que considerar estes fatores. Os descritores devem ser insensíveis às degradações assumidas e ao mesmo tempo devem ser suficientemente discriminantes para poderem distinguir entre as diferentes características bem como, suficientemente, estáveis para não serem influenciados por pequenas variações ou ruídos na imagem.

O tipo de função de mapeamento deve ser escolhido de acordo com as informações conhecidas a priori sobre o processo de aquisição e expectativa de degradações da imagem. Se não há informação a priori disponível, o modelo deverá ser flexível e geral o suficiente para lidar com todas as degradações eventuais que possam aparecer.

Na figura 2.7 retirada de Barbara e Jan (2003) podemos ver um exemplo de registro de imagem baseado em características, onde é feito o alinhamento das características correspondentes, representadas pelos pontos em vermelhos na imagem. Os pontos azuis representam características que não possuem correspondências na outra imagem.

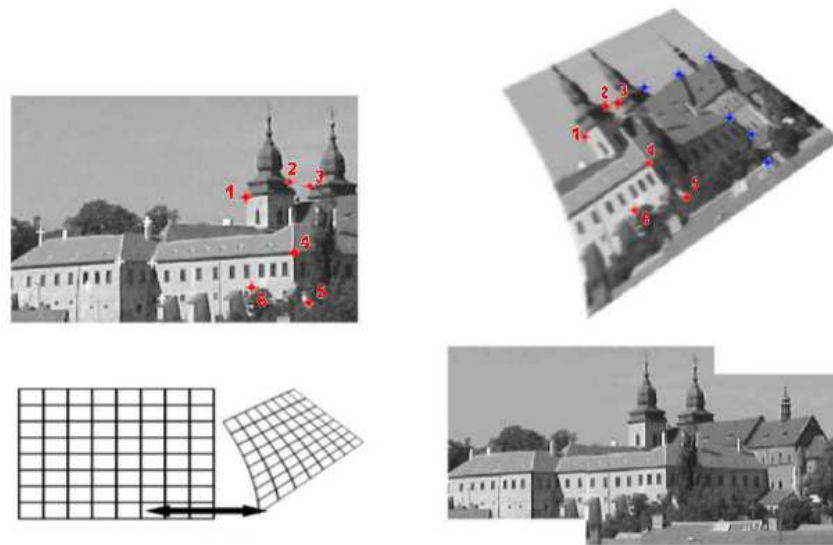


Figura 2.7: Exemplo de um alinhamento usando extração e correspondência de características

Métodos diretos ou baseados em intensidades

Nos métodos diretos não existem as etapas de detecção e correspondência de características. Nesse caso toda a imagem, ou parte dela, pode ser utilizada para estimar os parâmetros da função de mapeamento através da comparação das intensidades dos pixels, tornando possível uma maior exploração da informação contida em uma imagem ao contrário do método baseado em características que explora apenas alguns pixels da mesma. O fato de não precisar das duas primeiras etapas evita alguns problemas inerentes aos métodos de detecção e correspondência de características, como detecção incorreta de características, escolha equivocada dos descritores, falsas correspondências, entre outros. Porém aparecem outros problemas, como a seleção da parte da imagem que será utilizada para processamento, o cálculo do erro de reprojeção, entre outros.

Janelas de tamanho pré-definido ou, até mesmo, a imagem inteira são utilizadas para fazer as comparações entre as imagens. As limitações dos métodos baseados em janelas deslizantes se originam em sua idéia básica. Em primeiro lugar, a janela retangular, que é mais frequentemente utilizada, se adéqua ao registro de imagens que localmente se diferem apenas por uma translação. Se as imagens são deformadas por transformações mais complexas, este tipo de janela não é capaz de cobrir as mesmas partes da cena na imagem de referência e de sensoriamento. Então de alguma forma, essa janela deve ser dinâmica, mudando sua forma de acordo com as transformações esperadas, assim como foi implementada nesse trabalho.

Métodos clássicos baseados na intensidade, como os que usam correlação cruzada, exploram diretamente a intensidades da imagem, sem qualquer análise estrutural, consequentemente, eles são sensíveis a mudanças de intensidade introduzidas por ruídos, variação de iluminação ou quando se usam diferentes tipos de sensores.

Existem três métodos clássicos baseados em intensidades que merecem atenção: os que usam correlação, método de Fourier e otimização.

- **Métodos que usam correlação:** A medida de similaridade é calculada para pares de janelas da imagem de referência e sensorial e seu máximo é procurado. Os pares de janelas no qual o máximo é alcançado são definidos como os correspondentes. A função custo utiliza a diferença entre valores de pixels.
- **Métodos de Fourier:** Melhores do que os métodos que usam correlação. Eles exploram a representação de Fourier das imagens no domínio da frequência (correlação de fase). São aplicados caso seja necessária uma aceleração da velocidade computacional ou se as imagens foram obtidas sob condições variáveis.
- **Métodos de Otimização:** Encontrar o mínimo da medida de dissimilaridade ou o máximo da medida de similaridade entre imagens é um problema de otimização multidimensional, onde o número de dimensões corresponde aos graus de liberdade da transformação geométrica esperada. O único método que produz uma solução global extrema é uma pesquisa exaustiva sobre a imagem inteira. Embora seja computacionalmente custoso, muitas vezes é utilizado se apenas translações devem ser estimadas. Nos casos em que se esperam transformações com mais graus de liberdade ou medidas de similaridade mais complexas, são necessários algoritmos de otimização mais sofisticados. Os métodos de otimização mais utilizados são: minimização Gauss-Newton, métodos de descida do gradiente, Levenberg-Marquardt, entre outros. As aplicações do método de minimização numérica de Gauss-Newton para minimizar a soma do quadrado da diferença está descrito em Barnea e Silverman (1972) e do método de Levenberg-Marquardt é descrito em Sharma e Pavel (1997).

Deve-se notar uma coisa com relação a esses métodos de otimização. Às vezes, junto da medida de dissimilaridade, a fórmula a ser minimizada contém também o termo de penalidade, estes dois termos juntos formam a função de custo associada ao registro, e o objetivo dos métodos de otimização é minimizar isso. E similar aos métodos que usam correlação cruzada encontra-se o algoritmo de detecção de similaridade seqüencial [Barnea e Silverman 1972], onde é utilizado um método de pesquisa seqüencial. O algoritmo acumula a soma das diferenças absolutas dos valores de intensidade da imagem

e aplica um critério de corte. Se o valor acumulado exceder o limite indicado, o par de janelas de candidatos é rejeitado e o próximo par é testado. Esse método é muito utilizado devido a sua fácil implementação em hardware, o que o torna útil para aplicações em tempo real. Porém, funciona bem quando ocorrem apenas translações entre as imagens.

Então, observamos que em contraste com os métodos baseados em intensidades, os métodos baseados em características não trabalham diretamente com valores de intensidade da imagem. As características representam um nível mais elevado de informações. Esta propriedade faz com que os métodos baseados em características sejam mais adequados quando as imagens possuem objetos bastante distintos e facilmente detectáveis [Barbara e Jan 2003].

Já os métodos baseados em intensidade são recomendados quando as imagens não são tão ricas em detalhes, o que torna difícil a extração de características.

Um estudo mais detalhado sobre registro de imagem e suas aplicações podem ser encontrados em tutoriais como [BROWN 1992, Irani e Anandan 1999, Barbara e Jan 2003, Szeliski 2006]. Porém é importante salientar nenhum deles falam sobre registro de imagem aplicado em robótica.

Como Utilizar o Registro Direto de Imagem no SLAM Visual

Para explicar como estimar o mapa do ambiente e a pose do robô no SLAM visual usando registro direto de imagem, vamos usar as seguintes notações: Seja ζ^* a imagem capturada de uma cena e seja ζ outra imagem capturada após uma rotação $\mathbf{R} \in \mathcal{SO}(3)$ e uma translação $\mathbf{t} \in \mathbb{R}^3$ da câmera e esse deslocamento é expresso por uma matriz de transformação homogênea $\mathbf{T} \in \mathcal{SE}(3)$. Seja também $\mathbf{P} = [u, v, 1]^T \in \mathbb{P}^2$ o vetor homogêneo de coordenadas de um pixel e que $\zeta(P) \geq 0$ represente o seu valor de intensidade. A figura 2.8 esquematiza o que ocorre quando uma câmera captura imagens da mesma cena de pontos de vistas diferentes.

O problema de registro direto pode ser visto como a busca da matriz $\mathbf{T} \in \mathcal{SE}(3)$ que transforma todos os pixels da região da imagem corrente ζ tal que suas intensidades correspondem da melhor forma possível àquelas na imagem de referência ζ^* . Por sua vez um procedimento não linear de otimização local deve ser aplicado para resolver esse problema tendo em vista que métodos globais não respeitam as restrições de tempo real dos sistemas robóticos e a intensidade $\zeta(P)$ é não linear em $\mathbf{P}$. Em [Silveira et al. 2009] foi proposto usar a técnica clássica para resolver esse problema que consiste em desenvolver a função custo em série de Taylor, permitindo escrevê-lo como um problema de mínimos quadrados linear. Dado que essa solução corresponde apenas à uma aproximação,

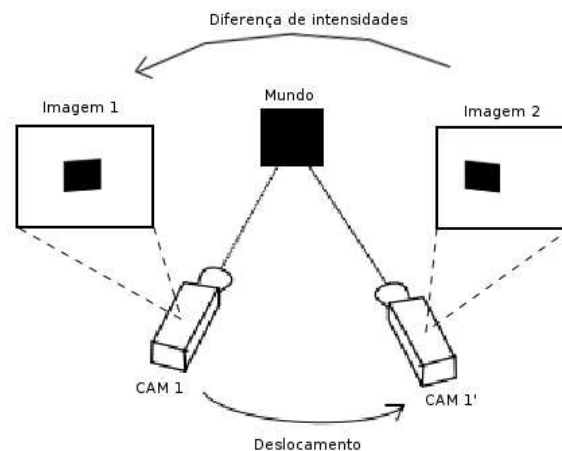


Figura 2.8: Formação das imagens após um deslocamento da câmera

iterações são conduzidas até a convergência dos parâmetros.

Até então o problema abordado foi o de localização, e no trabalho de Benhimane e Malis (2006) é apresentado uma solução eficiente para esse problema. Porém quando falamos de SLAM Visual o que buscamos, além da localização, é que o mapa do ambiente também seja estimado de forma simultânea. Então a solução da localização tem que ser entendida para incluir o modelo métrico $n_d \in \mathbb{R}^3$ como variável de otimização e assim estimar a pose da câmera e o mapa do ambiente de forma simultânea.

2.4.2 Estrutura a partir do Movimento (*Struct from motion - SFM*)

O principal objetivo da visão computacional é a reconstrução da geometria 3D da cena e do movimento de câmera a partir de um conjunto de imagens estáticas da mesma [Thrun et al. 2000]. É nesse contexto que se aplica o método conhecido como estrutura a partir do movimento ou apenas SFM (*Struct from motion*) que aborda a seguinte questão: Como obter informações sobre a geometria 3D da cena a partir de imagens 2D da mesma ?

Esta tarefa é um desafio porque o processo de formação da imagem não é inversível, ou seja, de sua posição projetada em um plano da imagem, um ponto da cena só pode ser recuperado com ambiguidade de parâmetros, correspondente à sua distância à câmera [Wiley 2009]. Uma possibilidade é a de explorar o conhecimento prévio sobre a cena (paralelismo e restrições de coplanaridade) para reduzir o número de graus de liberdade e assim as ambiguidades. Outra possibilidade é utilizar pontos correspondentes em múltiplas imagens para estimar, por triangulação, os pontos 3D correspondentes.

Um pré-requisito importante é a determinação da calibração e pose da câmera, que

pode ser expressa pela matriz de projeção, mas antes é importante explicar como funciona a relação entre um ponto 3D no espaço e um ponto 2D na imagem (plano da câmera). A figura 2.9 esquematiza essa relação.

Projeção na imagem de um ponto 3D no mundo

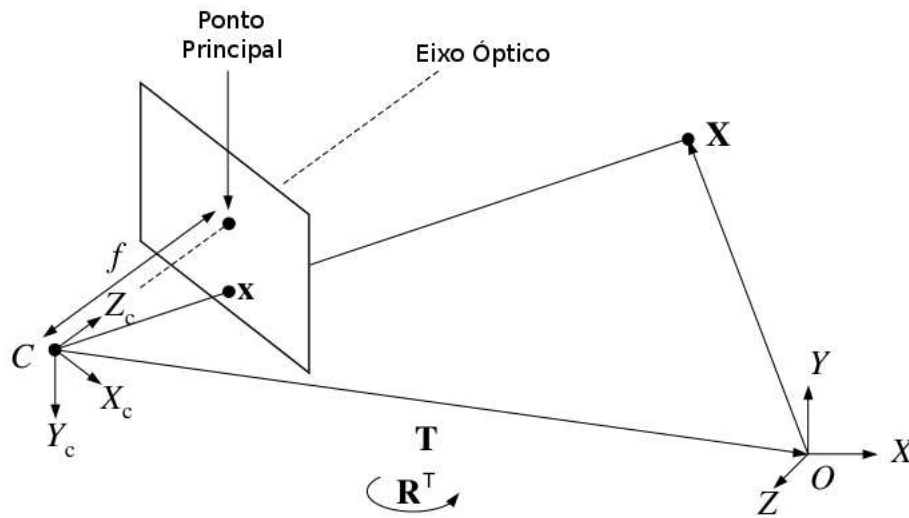


Figura 2.9: Projeção no plano da câmera de um ponto X no espaço, R e T representam as transformações entre as coordenadas do mundo XYZ e as da câmera $X_cY_cZ_c$

De acordo com o modelo de projeção pinhole, a relação entre um ponto 3D e seu ponto correspondente da imagem 2D tem três componentes:

Primeiro componente: Relacionar coordenadas de um ponto no mundo com as respectivas coordenadas na câmera através de uma transformação homogênea (parâmetros extrínsecos).

$$\begin{pmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{pmatrix} = \begin{pmatrix} R & T \\ 0 & 1 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \quad (2.11)$$

Segundo componente: Transformação 3D para 2D que relaciona pontos 3D em coordenadas da câmera com pontos 2D em coordenadas do plano da imagem. Usando semelhança de triângulos, obtemos a seguinte relação:

$$x = f \cdot (X_c / Z_c) \quad (2.12)$$

$$y = f \cdot (Y_c / Z_c) \quad (2.13)$$

Onde f é a distância focal. Como alterar o valor de f é o mesmo que mudar a escala da imagem, podemos definir $f = 1$.

Pode-se notar que o fator de escala some e a coordenada Z vira o novo fator de escala, isso porque a coordenada x só depende da direção do ponto de 3D em relação à câmera, e não quão longe ele está.

Terceiro componente: Transformação 2D para 2D que relaciona pontos em coordenadas da imagem ($\tilde{x} = [x \ y \ 1]$) em coordenadas de pixels $\tilde{u} = [u \ v \ 1]$:

$$\tilde{u} \sim K\tilde{x} \quad \text{Onde, } K = \begin{pmatrix} \alpha_u & S & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{pmatrix} \quad (2.14)$$

K é conhecida como matriz de calibração de câmera, onde seus parâmetros representam: α_u e α_v são fatores de escala conhecidos como distância focal, S representa a inclinação da malha óptica e u_0 e v_0 são as coordenadas da origem do sistema. Estes são os parâmetros intrínsecos da câmera.

Finalmente, é conveniente combinar todas essas equações em uma única equação linear, onde P é uma matriz projetiva que relaciona um ponto em coordenada de pixel e seu corresponde no mundo.

$$\tilde{u} \sim P\tilde{x} \quad (2.15)$$

$$P \sim K[RT] \quad (2.16)$$

Geometria Epipolar

Geometria Epipolar trata a seguinte questão: como um ponto no mundo aparece em duas imagens obtidas através de dois pontos de vista diferentes?

É possível resolver essa questão através da matriz essencial, que relaciona pontos da imagem que são correspondente em dois pontos de vista como mostra a equação 2.17, onde $\tilde{x}$ é a projeção do ponto 3D (do mundo) no plano do primeiro ponto de vista, $\tilde{x}'$ é a

projeção do mesmo ponto no segundo ponto de vista e $[E]$ é a matriz essencial. Essa equação também é conhecida como restrição epipolar. A figura 2.10 mostra, geometricamente, essa restrição.

$$\tilde{x}^T [E] \tilde{x}' = 0 \quad (2.17)$$

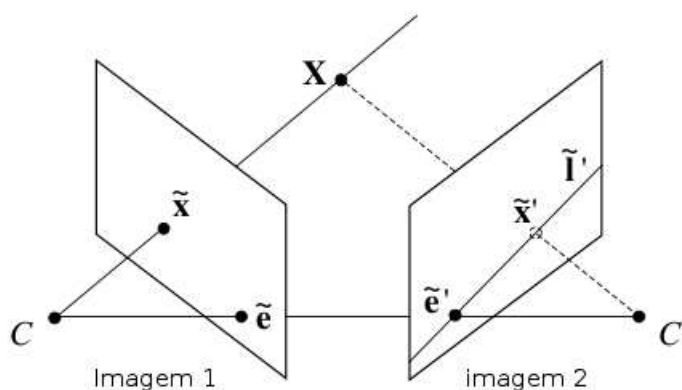


Figura 2.10: Geometria epipolar de 2 pontos de vista: Dado a projeção $\tilde{x}$ em uma imagem de um ponto X no espaço, a projeção $\tilde{x}'$ na outra imagem está restrita a linha epipolar I' , onde C representa a origem do sistema de coordenada da câmera e e representa o ponto epipolar.

O trabalho de Longuet-Higgins (1981) mostrou como uma matriz essencial relativa a um par de pontos de vista calibrados pode ser estimada a partir de oito ou mais pontos correspondentes. Já no trabalho de Wilely (2009) encontra-se como essa matriz pode ser decomposta para nos fornecer a orientação da câmera e a posição relativa dos pontos correspondentes.

A Geometria Epipolar nos fornece também a matriz fundamental que relaciona pontos da imagem, em coordenadas de pixels, que são correspondentes em dois pontos de vista. Ela pode ser estimada linearmente se forem dados oito ou mais pontos correspondentes e é uma matriz de rank dois. Decompondo a matriz fundamental, também, podemos recuperar a orientação da câmera e as coordenadas dos pontos correspondentes.

SFM a partir de vários pontos de vista

Visto que as matrizes essencial e fundamental contêm as restrições geométricas relacionadas a pares de pontos de vista, vamos voltar nossa atenção para resolver o problema da estrutura a partir do movimento para um número arbitrário de pontos de vista.

Entre os vários métodos propostos para resolverem esse problema, o método sequencial (figura 2.11) é o mais popular e funciona através da incorporação de sucessivos pontos de vista. Neste método, a inicialização adequada é normalmente obtida pela decomposição da matriz fundamental que relaciona os dois primeiros pontos de vista da sequência e então são feitas reconstruções parciais a partir de pontos 3D que aparecem em pares de pontos de vistas explorando a geometria epipolar que relaciona cada ponto de vista com seu antecessor.

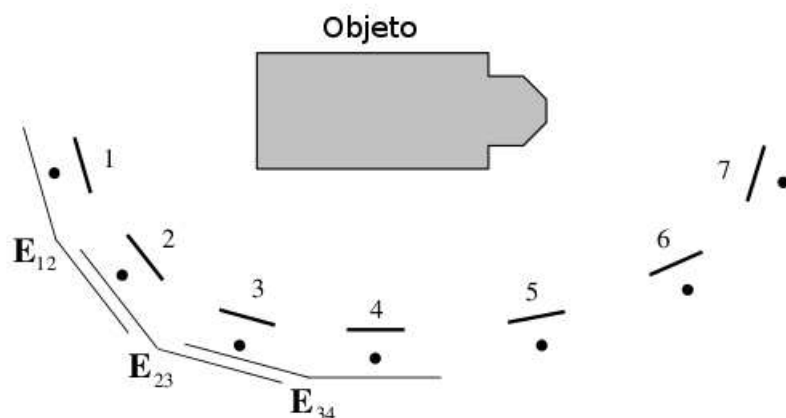


Figura 2.11: Esquematisação do método sequencial, onde E_{ij} representa a matriz essencial entre os pontos de vista i e j .

Uma desvantagem é que um grande número de pontos correspondentes deve ser definido em cada ponto de vista, e eles, normalmente, devem ser visíveis em três ou mais pontos de vista, isto significa que um tratamento, substancial, da sobreposição é necessário. Para longas sequências de pontos de vista (por exemplo, ao longo de uma rua da cidade), este requisito pode ser proibitivo.

Para finalizar o processo de estimação da pose da câmera e da estrutura da cena é necessário refinar as estimativas da matriz de projeção e dos pontos 3D usando otimização não-linear iterativa para minimizar uma função de custo adequada. Isso é conhecido como *bundle adjustment*, e funciona minimizando uma função de custo que está relacionada a uma soma ponderada dos erros de reprojeção ao quadrado. Normalmente iteração Gauss-Newton ou o método de Levenberg-Marquardt são os métodos de otimização utilizados devido à rápida convergência.

Existem duas classes de algoritmos que tratam o problema do SFM a partir de vários pontos de vista e a escolha entre essas classes de algoritmos depende fortemente da aplicação. São elas: Os algoritmos com *Narrow-Baseline* e os com *Wide-Baseline*.

O primeiro representa uma classe de algoritmos que parte do pressuposto de que a

mudança na posição e na orientação da câmera é pequena, assim a vizinhança local dos pontos de interesse é quase semelhante em dois pontos de vista próximos e podem ser caracterizados por um conjunto de valores das intensidades dos pixels da amostra de uma janela retangular centrada no ponto de interesse. Os valores das intensidades dos pixels são comparados por correlação cruzada normalizada ou pela soma da diferença de quadrados. Uma desvantagem desse método é na computação da profundidade que é muito sensível a ruídos na medida das coordenadas da imagem em pontos de vista muito próximos.

A outra classe de algoritmos, conhecida como *Wide-baseline*, é adequada quando a distância entre pontos de vista (baseline) é larga, nesse caso, superfícies nas duas imagens podem apresentar alterações substanciais de escala, diferentes graus de encurtamento, diferentes padrões de oclusão e grandes disparidades nas suas localidades. Essa classe de algoritmos deve ser invariante para diferentes transformações da imagem.

2.5 Métodos de Otimização

A essência da otimização está em melhorar algo em um conjunto de alternativas disponíveis. Trata-se de uma ferramenta de grande aplicabilidade que se constitui numa vasta e atraente área do conhecimento, sendo de grande importância para diversas áreas das ciências e engenharias. Um engenheiro, por exemplo, deseja projetar a melhor configuração de um equipamento com o menor custo possível.

A otimização está diretamente ligada a matemática e a computação. Antes de otimizar um problema é necessário elaborar um modelo, expresso por meio de uma função objetivo sujeita a variáveis de decisão, sendo que este pode ser considerado o passo principal da otimização.

O modelo não pode ser muito simples nem muito complexo, visto que, se muito simples poderá não representar o problema real, e se muito complexo dificultará a busca pela solução. Em muitos casos, a solução ideal não é encontrada de forma analítica, então é necessário procedimentos iterativos de cálculos que a cada passo procuram melhorar a solução atual, até que o extremo (mínimo ou máximo) da função objetivo seja encontrado, ou que algum critério de parada seja satisfeito. A construção de bons algoritmos, que resolvam problemas de otimização independentemente da dimensão e dos parâmetros envolvidos representa um passo importante nesse processo.

Podemos organizar as estratégias ou métodos de otimização em três grandes grupos: Métodos de Otimização Baseados em Gradientes, Métodos de Otimização Baseados em Meta-Heurísticas e Métodos de Otimização Livres de Derivadas.

2.5.1 Métodos de Otimização Baseados em Gradientes

A maioria dos métodos de otimização baseados em gradientes utilizam um procedimento iterativo para encontrar o menor valor da função objetivo. A ideia básica é: dado um ponto inicial começa-se uma busca arbitrária, onde a direção e o tamanho do passo são calculados no procedimento iterativo. As direções podem ser obtidas simplesmente pela direção contrária à direção do vetor gradiente da função objetivo. Em alguns casos são obtidas por meio da matriz Hessiana da função, e ainda, algumas direções utilizam combinações que levam em consideração essas duas últimas a fim de direcionar a busca para o mínimo da função.

Dentre os métodos de Otimização Baseados em Gradientes podemos destacar:

- **Método da Máxima Descida:** (steepest descent method) é um dos mais antigos e simples métodos de otimização. Consiste em uma estratégia de busca que se move na direção contrária à direção do gradiente da função objetivo no ponto atual da busca e com um tamanho de passo apropriado a cada iteração [Nocedal e Wright 2006].
- **Método de Newton:** a ideia principal do método de Newton é usar de forma iterativa uma aproximação quadrática da função objetivo e minimizar essa função. Este esquema iterativo é conhecido como método de Newton puro [Sun e Yuan 2006]. Para uma função quadrática, o método de Newton pode convergir para o mínimo local com apenas uma iteração. Porém, para uma função geral, não quadrática, não existem garantias de que o método de Newton convirja para o mínimo com iterações finitas. No entanto, uma vez que a função objetivo é aproximada por uma função quadrática, e se o ponto inicial estiver próximo do mínimo local, então o método de Newton irá convergir rapidamente.

Algoritmos que usam a direção do método de Newton possuem uma taxa de convergência rápida, tipicamente quadrática, isso depois que a vizinhança da solução é alcançada por alguma iteração. Porém, a necessidade de calcular a inversa da matriz Hessiana a cada iteração, torna o método custoso do ponto de vista computacional.

- **Método dos Gradientes Conjugados:** esse método está entre o método da máxima descida e o método de Newton. Ele é motivado pelo desejo de acelerar a convergência tipicamente lenta do método da máxima descida, e ao mesmo tempo, evitar as informações associadas as derivadas de segunda ordem, as quais são necessárias pelo método de Newton. Dessa forma, uma vez que não precisa calcular a matriz Hessiana ou a sua aproximação, o método dos gradientes conjugados é amplamente utilizado na resolução de problemas de otimização de grande escala

[Luenberger 1984].

- **Métodos quase-Newton:** é sabido que o Método de Newton é um poderoso método, devido em sua direção de busca utilizar informações da matriz Hessiana, a qual fornece informações úteis sobre a curvatura de uma função objetivo. No entanto, em várias aplicações práticas, a avaliação computacional da matriz Hessiana é dispendiosa, ou ainda, a mesma não está disponível em sua forma analítica. Dessa forma, o método quase-Newton surgiu com a proposta de não calcular a matriz Hessiana, mas sim, realizar uma aproximação iterativa de sua inversa, reduzindo assim o custo computacional do método de Newton e ao mesmo tempo mantendo uma rápida taxa de convergência [Sun e Yuan 2006].

2.5.2 Métodos de Otimização Baseados em Meta-Heurísticas

Os métodos de otimização baseados em meta-heurísticas realizam a busca utilizando heurísticas tradicionais a fim de encontrar o mínimo global da função objetivo. Além disso, esses métodos possuem mecanismos que permitem a busca escapar de mínimos locais [Gendreau e Potvin 2010]. Um Exemplo de meta-heurísticas são os algoritmos evolutivos.

Algoritmos Evolutivos

Os algoritmos evolutivos (AEs) são algoritmos estocásticos, cuja a busca pelo melhor valor da função objetivo é baseada em fenômenos naturais propostos por Darwin. A ideia básica é começar com um conjunto de soluções, geralmente criados aleatoriamente, denominado população. Em cada iteração, novas populações são geradas a partir do conjunto existente, por meio da aplicação de operadores genéticos como reprodução, mutação e seleção. O melhor conjunto de população é mantido para a iteração seguinte, em que um novo ciclo de operações genéticas são realizadas. Assim, o algoritmo procede de uma maneira evolutiva, onde os indivíduos mais aptos sobrevivem. Embora simplista do ponto de vista de um biólogo, os algoritmos evolutivos são suficientemente complexos, no sentido de proporcionar mecanismos de busca de uma forma adaptativa e robusta [Spears 2000].

Três abordagens gerais foram desenvolvidas de forma independente: **Programação Evolutiva** por Fogel L. e J (1966) com o objetivo de utilizar os conceitos de evolução no desenvolvimento da inteligência artificial, **Estratégias Evolutivas** por Rechenberg (1973) com o objetivo de solucionar problemas de otimização de parâmetros e **Algoritmos Genéticos** por Goldberger (1989) e Holland (1992) com o objetivo de obter soluções computacionais eficientes para problemas de otimização.

A estrutura de qualquer método evolutivo é basicamente a mesma. As principais etapas de um algoritmo evolutivo pode ser visto na figura 2.12.

Algoritmo : Algoritmo Evolutivo Padrão

Entradas

(1) População inicial de indivíduos $\mathbf{P}^{(k)}$

Passo 1: **Faça** $k = 0$

Passo 2: Avalie os indivíduos de $\mathbf{P}^{(k)}$ e obtenha a aptidão para cada um

Passo 3: **Enquanto** algum critério de parada não for atingido, **faça**

(3a) Selecione os indivíduos de $\mathbf{P}^{(k)}$ com melhor aptidão

(3b) Recombine e gere novos indivíduos para $\mathbf{P}^{(k)}$

(3c) Modifique os novos indivíduos de $\mathbf{P}^{(k)}$

(3d) Avalie os novos indivíduos de $\mathbf{P}^{(k)}$ e obtenha a aptidão para cada um

(3e) Selecione uma nova população $\mathbf{P}^{(k+1)}$ a partir de $\mathbf{P}^{(k)}$

(3f) $k = k + 1$

Passo 4: **Retorna** indivíduo com melhor aptidão

Figura 2.12: Algoritmo padrão de métodos evolutivos

Geralmente, um algoritmo evolutivo inicializa a sua população aleatoriamente, embora o conhecimento específico do domínio também possa ser utilizado. A avaliação mede a aptidão de cada indivíduo, de acordo com o seu valor no ambiente em que se encontra, podendo ser simples como calcular uma função matemática, ou complexa, como executar uma simulação elaborada.

A seleção dos indivíduos no Passo (3a) decide quem serão os pais, isto é, os indivíduos com melhor aptidão são mais propensos a serem pais e terem mais filhos (indivíduos). Os filhos são criados através de recombinação e mutação. Na recombinação, os filhos trocam informações entre os pais, transmitindo para as novas gerações as boas propriedades das gerações anteriores.

Na mutação, os filhos são perturbados ainda mais, com o intuito de diversificar a população e assim terem mais chances de escapar dos mínimos locais. Os filhos são então avaliados e finalmente uma nova população é gerada (Passo (3e)).

O processo se repete até que um critério de parada seja satisfeito, podendo esse ser um número máximo de gerações ou qualquer outro critério que esteja intrínscio ao problema.

Existem algumas características nos algoritmos evolutivos no qual seu sucesso é atribuído. Primeiro, os AEs trabalham com uma população em vez de um único ponto. Assim, a probabilidade de ficar preso em um mínimo local é reduzida em comparação com os métodos que procedem a busca ponto a ponto. Em segundo, os AES são de fácil implementação, o que os torna muito atrativos no ponto de vista computacional.

Além disso, são adaptáveis a problemas com diversos tipos de variáveis e função objetivo [Rangaiah 2010].

Porém, como desvantagem, os AEs podem sofrer convergência prematura em algumas aplicações, além da existência de vários parâmetros de ajuste. Contudo, estas desvantagens podem ser atenuadas adicionando procedimentos específicos para manter a diversidade da população e utilizando a lista de referência dos parâmetros de ajuste disponível na literatura para serem empregados em problemas gerais [Lacerda e Carvalho 1999].

Nesse trabalho foi utilizado Algoritmos genéticos para se obter os parâmetros ótimos do mapa e da pose do sistema robótico. Veremos mais detalhes sobre a utilização desse algoritmo evolutivo no próximo capítulo.

2.5.3 Métodos de Otimização Livres de Derivadas

Os métodos de otimização livres de derivadas tem se tornado uma importante e desafiadora área. A necessidade cada vez maior em resolver problemas de otimização nas quais as derivadas estão indisponíveis ou disponíveis a um custo proibitivo e a crescente complexidade na modelagem matemática são algumas das razões pelas quais a otimização livre de derivadas é atualmente uma área de grande demanda.

O desenvolvimento de algoritmos livre de derivadas remonta aos trabalhos de Spendley W. e F. (1962) com seu algoritmo baseados em simplex. Trabalhos seguintes como os de Conn A. e P. (1996) e S. e M. (2002) tiveram um progresso significativo, fornecendo provas de convergência e incorporando o uso de modelos alternativos.

Pode-se afirmar que existem três características presentes em todos os algoritmos livres de derivadas que são globalmente convergentes:

1. Os métodos incorporam mecanismos para impor que a direção de busca seja sempre de descida, assim como nos métodos baseados em gradientes. Além disso, combinando esses mecanismos com uma possível redução do parâmetro do tamanho do passo, a direção de descida é garantida longe do ponto estacionário do problema;
2. Os métodos devem garantir alguma forma de controle da geometria do conjunto de pontos onde a função é avaliada. Essencialmente, essas operações asseguram que qualquer indício de estacionariedade é de fato verdadeira. Não impor uma boa geometria implica em falta de convergência do método;
3. Os métodos devem conduzir a sequência do parâmetro de tamanho do passo para zero. Neste caso, o tamanho do passo possui um duplo propósito, além de controlar o tamanho da área de onde a função é amostrada em torno da iteração atual, o tamanho do passo está intimamente ligado com o critério de parada do algoritmo.

Embora nos métodos baseados em gradientes esse tipo de terminação do algoritmo seja prematuro e uma indicação de falha do método, talvez porque as derivadas ou não são precisas o suficiente ou mal codificadas, nos métodos livres de derivadas esse tipo de terminação é uma consequência implícita dos mecanismos de algoritmos eficazes. Desse modo, um critério de parada com base no tamanho do passo é natural

O Método do Poliedro Flexível

Esse método faz parte da classe de otimização de funções numéricas multivariáveis, capaz de lidar com problemas de qualquer ordem. A ordem de um problema de otimização multivariável é o número de parâmetros da função a minimizar.

O método dos poliedros flexíveis cria um poliedro cujo número de vértices é igual à ordem do problema mais um e calcula o valor da função para cada um dos vértices. A cada passo, o pior vértice é refletido em relação à face formada pelos outros vértices (processo de REFLEXÃO) e o valor da função é calculado para este novo ponto. Se o valor for muito bom (ou seja, menor do que todos os anteriores), anda-se mais um pouco nesta direção e o poliedro aumenta de tamanho (processo de EXPANSÃO). Se o ponto for ruim, o poliedro diminui de tamanho (processos de REDUÇÃO ou CONTRAÇÃO). Se o ponto for, nem muito bom nem muito ruim, apenas a reflexão é feita

2.6 Conclusão do Capítulo

No presente capítulo foi apresentado os conceitos necessários para entender o problema do SLAM visual e a solução que está sendo proposta. Solução esta que utilizará métodos diretos e otimização baseada em meta-heurística livre de derivadas. Por esse motivo foi necessário uma breve explanação sobre registro direto de imagem e métodos de otimização. Para entender melhor o problema da localização e mapeamento simultâneo (SLAM) na robótica móvel e suas abordagens, esse problema foi tratado de forma separada, onde falamos sobre localização e depois sobre mapeamento. Também foram apresentadas duas técnicas de visão computacional cujos conceitos foram amplamente utilizados no trabalho proposto. O registro direto de imagem foi utilizado no cálculo do erro de reprojeção e a estrutura a partir do movimento foi usada para reconstrução 3D dos pontos da imagem. No próximo capítulo vamos falar sobre as abordagens do SLAM visual, explicando o motivo pelo qual essa abordagem está sendo muito pesquisada na área da robótica móvel e expor o estado da arte sobre esse tema.

Capítulo 3

Desenvolvimento e Metodologia

Nesse capítulo veremos como foi desenvolvida a técnica apresentada. Inicialmente será exposta uma visão geral do problema no qual será aplicado o sistema desenvolvido, depois veremos como foi formulada a solução para resolver esse problema.

3.1 Caracterização do Problema

Vimos nos capítulos anteriores que um pré-requisito fundamental para um robô móvel é a autonomia, ou seja, a capacidade de navegação em ambientes desconhecidos sem ajuda humana. Para realizar tal tarefa, é necessário uma percepção sobre o ambiente no qual está inserido. Então o robô precisa de sensores que possam extrair o máximo de informações úteis sobre o ambiente ao seu redor e utilizar essas informações para se localizar no mesmo. Acontece que a localização, mapeamento e até mesmo a locomoção robótica são caracterizados por incertezas e ruídos.

A abordagem clássica de SLAM Visual propõem estimar os parâmetros relativos ao mapa e à localização do robô usando extração e correspondência de pontos característicos das imagens. E utiliza filtros probabilísticos para tratar as incertezas presentes no sistema robótico como um todo. Elas funcionam bem quando o robô está inserido em um ambiente onde é possível a extração desses pontos. Na figura 3.1 retirada de [Aires 2009] está um exemplo desse tipo de ambiente.

Esse tipo de abordagem torna o sistema restrito a certos ambientes além de exigir etapas a priori de extração e correspondência de primitivas geométricas e etapas a posteriori para formular um mapa mais rico a partir dos pontos extraídos das imagens. Também faz-se necessário a modelagem completa do sistema robótico, incluindo a modelagem do erro sensorial, visto que as incertezas são tratadas com filtros probabilísticos que le-



Figura 3.1: Ambiente de fácil extração e correspondência de características

vam em consideração as propriedades estatísticas do ruído presente no sistema robótico e um esquema de realimentação que utiliza as medições retornadas pelos sensores. Outro problema da abordagem tradicional está relacionado ao número de parâmetros que são gerados pelo sistema, estes relativos à quantidade de primitivas geométricas extraídas da cena.

Pensando em contornar essas restrições, Silveira et al. (2008) propôs um sistema de SLAM Visual que utiliza registro direto de imagem e métodos de otimização para estimar os parâmetros relativos ao mapa e localização do robô. Dessa forma foi possível realizar o SLAM sem a necessidade de extração e correspondência de primitivas geométricas, porém o sistema fica restrito a ambientes que podem ser representados por planos ou segmentos de planos. Além do que ele utiliza um método de otimização baseado no gradiente, onde todo o sistema está acoplado para formar uma equação de segunda ordem derivável. Isso torna o sistema especialista em apenas um tipo de representação, que são os segmentos de planos.

Observa-se que ambientes externos ou internos são ricos em formas geométricas e obstáculos que são difíceis de serem modelados por planos, como mostra a figura 3.2. Nesses casos as abordagens existentes não funcionarão adequadamente.

Com intento de resolver esses problemas, foi desenvolvida a técnica apresentada nesta tese de doutorado e que será explanada a seguir.

3.2 Solução Desenvolvida

A proposta desse trabalho foi desenvolver uma abordagem desacoplada de SLAM Visual totalmente livre de etapas a priori de extração, descrição e correspondência de características, utilizando registro direto de imagem e métodos de otimização adequados para estimar os parâmetros necessários. Essa abordagem não será restrita a planos, podendo



Figura 3.2: Exemplo de ambientes com formas geométricas não planares

mapear adequadamente as diversas formas geométricas que podem aparecer na cena.

Desta forma, obteve-se um mapa com um número de parâmetros menor que os das abordagens tradicionais. Isso porque ao se utilizar formas geométricas mais complexas, os parâmetros necessários para representar o mapa são aqueles que representam uma forma geométrica específica e sua pose no espaço 3D. Já quando se utiliza um conjunto de primitivas geométricas, cada uma possui no mínimo três parâmetros a serem estimados (coordenadas espaciais).

Outra vantagem importante do sistema desenvolvido é que ao final do processo de estimação de parâmetros podemos ter, após um processo de texturização, um mapa 3D com textura, pronto para utilização ou manipulação, ao contrário das abordagens que utilizam pontos, onde são necessárias etapas a posteriores de agrupamento ou interpolação, refinamento e por fim a texturização, para se obter algo mais representativo.

Para ser totalmente livre da utilização de características, a ideia básica da abordagem desenvolvida é estimar os parâmetros necessários através da minimização do erro de reprojeção entre a imagem de referência transformada pelo modelo de transformação e a imagem capturada após deslocamento do robô. Esse erro será obtido através de uma técnica adequada de registro direto de imagem.

O sistema desenvolvido é formado por três componentes principais e independentes: o sistema de visão, responsável pela aquisição de imagem, que atualmente é um sistema monocular, porém de fácil adaptação para utilizar visão estéreo ou qualquer outro; um modelo de transformação e registro de imagem; e um componente responsável pela otimização ou estimação dos parâmetros ótimos. A figura 3.3 mostra um esquema do sistema,

onde se pode entender a interação entre os diversos componentes.

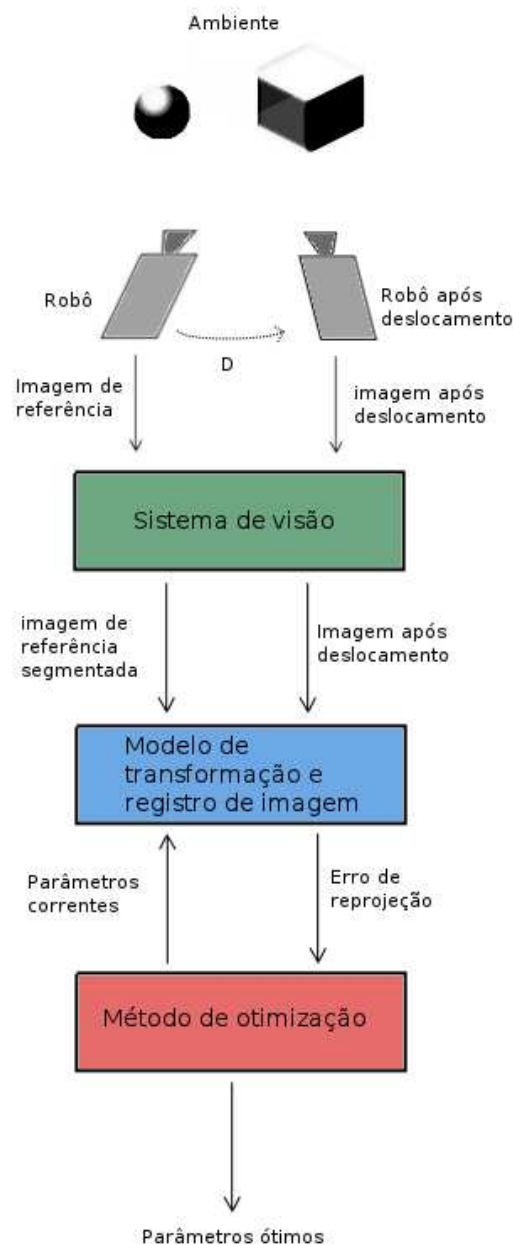


Figura 3.3: Esquema do sistema

O **modelo de transformação geométrica e registro de imagem** é o componente central do sistema. Ele receberá as imagens do sistema de visão computacional e os parâmetros correntes do mapa e da pose do robô, com isso ele simulará uma imagem de como deveria ser cena, de acordo com os parâmetros correntes. A comparação dessas

imagens retornará uma medida de erro que será passada para o componente responsável pela otimização dos parâmetros correntes.

Essa medida de erro é calculada através do método de **registro direto de imagem**. Este método utilizará apenas as intensidades de pixels para comparar a imagem real da cena e a imagem gerada pelo modelo de transformação, caso sejam iguais, significa que não temos nenhum erro na estimativa dos parâmetros relativos ao mapa e posição do robô. A figura 3.4 mostra um exemplo onde o modelo de transformação geométrica recebeu parâmetros ruins, que não condizem com os parâmetros que representam a cena real, e gerou uma cena diferente da realidade. Vale ressaltar que nesse caso, toda a cena foi considerada como um plano relativo ao solo. Já a figura 3.5 mostra um caso onde os parâmetros passados para o modelo de transformação são ótimos, logo o erro de reprojeção é mínimo.



Figura 3.4: Exemplo de uma cena gerada a partir de parâmetros equivocados



Figura 3.5: Exemplo de uma cena gerada a partir de parâmetros ótimos

Uma das contribuições desse trabalho é de que o sistema desenvolvido não ficou restrito a representações planares da cena, então o modelo de transformação geométrica foi

flexibilizado para funcionar bem com outras formas geométricas. Como já foi dito, ele terá como entrada a imagem de referência e os parâmetros de transformação e dependendo da forma geométrica que estará sendo utilizada, o modelo deve ser capaz de calcular como os pixels da imagem de referência devem ser projetados em outro ponto de vista de acordo com a forma geométrica que representa a região de interesse da cena. Por esse motivo ele foi implementado de forma modular para que seja possível testar várias parametrizações de formas geométricas sem necessidade de grandes mudanças no código ou na matemática envolvida.

O método de registro direto de imagem também deve ser modificado para se adequar à essa situação, pois o registro direto de imagem tradicional funciona muito bem quando se trata de planos, pois um plano está sujeito a oclusão de pixels apenas no caso de rotações muito agudas que transforme o plano em uma reta (figura 3.6). Essas rotações fogem totalmente às restrições físicas do problema.

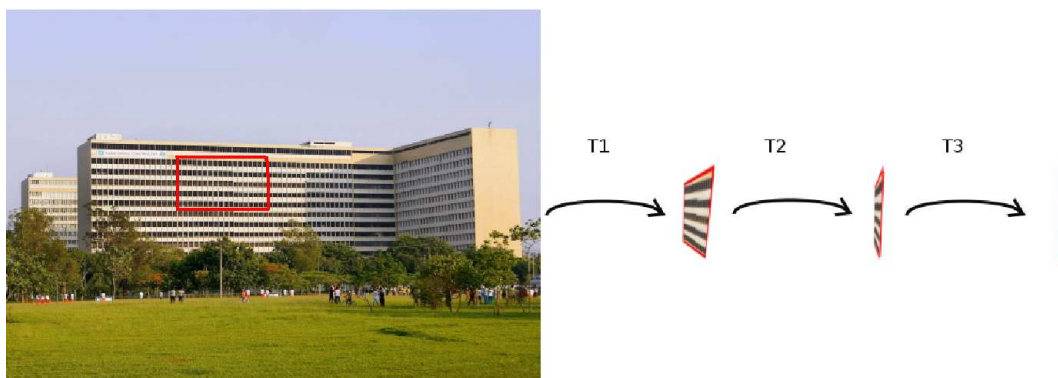


Figura 3.6: Transformações (T1,T2 e T3) que causam oclusão de pixels

Então foram necessárias algumas modificações para abordar os problemas que ocorrem ao se utilizar formas geométricas mais complexas, onde enfrentamos o problema de perda e de oclusão de pixels. Esse problema deve ser tratado e pode até ser utilizado no processo de otimização. Um exemplo é quando se trata de uma esfera, sabe-se que ao mudar o ponto de vista dessa forma geométrica ocorre uma perda gradual de pixels que se encontram nas extremidades da mesma. Isso pode ser utilizado para refinar os parâmetros estimados. A figura 3.7 exemplifica essa situação, onde a região em vermelho representa os pixels que deixaram de aparecer na imagem após o deslocamento e a região em verde representa os pixels que passaram a aparecer.

O fato do sistema ser formado por componentes independentes (como mostra a figura 3.3), ao contrário dos trabalhos encontrados na literatura, torna possível também a utilização de várias técnicas de otimização. Então foram testados várias técnicas baseadas



Figura 3.7: Exemplo de pixels que desaparecem ou aparecem de acordo com o deslocamento da câmera

em em Meta-Heurísticas (seção 2.5.2) e livres de derivadas (seção 2.5.3). Nesses testes a técnica que retornou melhores resultados foi um Algoritmos Evolutivo (seção 2.5.2) conhecido como Algoritmo Genético.

3.3 Metodologia

Nessa seção veremos como foi implementado o sistema, do ponto de vista matemático e as convenções utilizadas.

Na primeira versão do sistema foi considerado que o mundo é formado por planos. Assim garante-se que o mesmo pixel sempre aparecerá em dois pontos de vistas diferentes da mesma cena, ou seja não há perda de informação. Além de que é possível parametrizar planos com apenas três parâmetros. No sistema proposto foi utilizada a menor distância entre o plano e a origem do sistema e dois ângulos (ver figura 3.8).

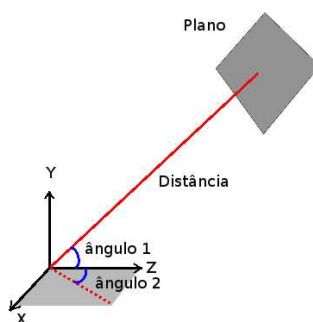


Figura 3.8: Parametrização do plano

O objetivo principal da técnica implementada é: a partir de imagens capturadas da cena, obter um vetor de parâmetros ótimos $[x \ y \ z \ R_x \ R_y \ R_z \ \rho \ \alpha \ \beta]^T$, onde os seis primeiros

parâmetros representam a localização do robô no espaço 3D, conseqüentemente, o deslocamentos do mesmo e os três últimos representam a forma geométrica em questão(o mapa).

Após o primeiro deslocamento do robô no espaço, teremos duas imagens obtidas pelo sistema de visão computacional: a primeira capturadas antes (imagem de referência) e a segunda depois deste certo deslocamento. na imagem de referência são selecionadas regiões de interesse que representam formas geométricas no mundo 3D (que nesta versão do sistema são os segmentos de planos). Já o método de otimização é responsável por gerar um vetor de parâmetros iniciais para limitar o espaço de busca. Para isso utiliza-se os sinais de controle enviados para o robô, ou seja, caso a ordem seja que o robô se desloque 2cm no eixo x, então o espaço de busca se restringe às proximidades daquele ponto. Esses parâmetros são chamados de parâmetros correntes.

Então, tendo as duas imagens, as regiões de interesse e um vetor de parâmetros correntes, o módulo responsável pela transformação geométrica e registro de imagem entra em ação com objetivo de testar os parâmetros e retornar um erro de reprojeção. Quando esse erro for mínimo significa que aqueles parâmetros correntes passados pelo método de otimização representam o vetor de parâmetros ótimos $[x \ y \ z \ R_x \ R_y \ R_z \ \rho \ \alpha \ \beta]^T$.

Então seja R uma região na imagem de referência que representa um segmento de plano no mundo, o modelo de transformação é capaz de calcular a projeção P^* de cada pixel P de acordo com certos parâmetros correntes, que possivelmente representam o plano e o deslocamento da câmera. Como se trata de regiões planares a matriz de Homografia soluciona esse problema, porém a proposta desse trabalho é utilizar um modelo de transformação que trate várias formas geométricas, por esse motivo esse módulo foi implementado em seis componentes separados:

O **primeiro componente** é responsável por uma transformação 2D para 2D, onde a inversa da matriz de parâmetros intrínsecos é aplicada em P , que está em coordenadas de pixel $[u \ v \ 1]^T$, para obter $[X_{img1} \ Y_{img1} \ 1]^T$ que está em coordenadas da imagem. Nota-se que todas as coordenadas serão expressas em coordenadas homogêneas.

$$\begin{pmatrix} X_{img} \\ Y_{img} \\ 1 \end{pmatrix} = (K^{-1}) \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} \quad (3.1)$$

Onde K^{-1} é a inversa da matriz de parâmetros intrínsecos.

O **segundo componente** é responsável por uma transformação 2D para 3D, onde calcula-se quais são as coordenadas 3D de P em relação à câmera. Em outras palavras:

qual a projeção no mundo 3D do ponto P . Esse é o componente principal do modelo. É importante salientar que um ponto em uma imagem 2D representa uma infinidade de pontos no mundo (uma reta). Então será necessário um cálculo de interseção entre essa reta, que representa as infinitas possibilidades de projeção e a forma geométrica que representa a cena. Como estamos considerando que o mundo é formado por planos, utilizaremos as equações paramétricas que representam segmentos de planos no espaço 3D. De acordo a convenção representada na figura 3.8 obtemos as seguintes equações:

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \rho \begin{pmatrix} \cos(\alpha) \cdot \sin(\beta) \\ -\sin(\alpha) \\ \cos(\alpha) \cdot \cos(\beta) \end{pmatrix} + \lambda_1 \begin{pmatrix} \sin(\alpha) \cdot \sin(\beta) \\ \cos(\alpha) \\ \sin(\alpha) \cdot \cos(\beta) \end{pmatrix} + \lambda_2 \begin{pmatrix} 1 - \cos^2(\alpha) \cdot (1 - \cos(\beta)) \\ \sin(\alpha) \cdot \cos(\alpha) \cdot \sin(\beta) \cdot (1 - \cos(\beta)) \\ -\sin(\beta) \end{pmatrix} \quad (3.2)$$

Onde ρ representa o segmento de reta relativo à menor distância entre o plano e a origem, α representa o ângulo formado entre ρ e o plano XZ e β representa o ângulo formado entre a projeção de ρ no plano XZ e o eixo Z.

Para se obter as coordenadas 3D de P basta calcular a interseção entre reta que passa pela posição da câmera e pelo ponto $[X_{img1} \ Y_{img1} \ Z_{img1}]^T$ com o suposto plano no mundo. Essa interseção, descrita na equação abaixo, nos dará as coordenadas $[X_{c1} \ Y_{c1} \ Z_{c1}]^T$ que é a representação de P em coordenadas 3D da câmera.

$$\begin{pmatrix} PX_{cam} - \rho \cdot \cos(\alpha) \cdot \sin(\beta) \\ PY_{cam} + \rho \cdot \sin(\alpha) \\ PZ_{cam} - \rho \cdot \cos(\alpha) \cdot \cos(\beta) \end{pmatrix} = \begin{pmatrix} -X_{img1} & \sin(\alpha) \cdot \sin(\beta) & 1 - \cos^2(\alpha) \cdot (1 - \cos(\beta)) \\ -Y_{img1} & \cos(\alpha) & \sin(\alpha) \cdot \cos(\alpha) \cdot \sin(\beta) \cdot (1 - \cos(\beta)) \\ -Z_{img1} & \sin(\alpha) \cdot \cos(\beta) & -\sin(\beta) \end{pmatrix} \begin{pmatrix} \lambda_r \\ \lambda_1 \\ \lambda_2 \end{pmatrix} \quad (3.3)$$

Onde $[X_{img1} \ Y_{img1} \ Z_{img1}]^T$ e $[PX_{cam} \ PY_{cam} \ PZ_{cam}]^T$ representam os 2 pontos da reta, α , β , ρ , λ_1 e λ_2 são os parâmetros do plano e λ_r é um escalar. Após resolver este sistema de equações lineares, obteremos os valores de λ_1 e λ_2 , bastando substituí-los na equações paramétricas 3.2 para obter $[X_{c1} \ Y_{c1} \ Z_{c1}]^T$.

$$X_{c1} = \rho \cdot \cos(\alpha) \cdot \sin(\beta) + \lambda_1 \cdot \sin(\alpha) \sin(\beta) + \lambda_2 \cdot (1 - \cos^2(\alpha) \cdot (1 - \cos(\beta))) \quad (3.4)$$

$$Y_{c1} = -\rho \cdot \sin(\alpha) + \lambda_1 \cdot \cos(\alpha) + \lambda_2 \cdot \sin(\alpha) \cdot \cos(\alpha) \cdot \sin(\beta) \cdot (1 - \cos(\beta)) \quad (3.5)$$

$$Z_{c1} = \rho \cdot \cos(\alpha) \cdot \cos(\beta) + \lambda_1 \cdot \sin(\alpha) \cdot \cos(\beta) - \lambda_2 \cdot \sin(\beta) \quad (3.6)$$

É importante salientar que os parâmetros que representam o plano no mundo (o mapa) não são conhecidos nesse momento, utiliza-se os parâmetros correntes passados pelo método de otimização. Parâmetros esses que podem representar com exatidão a cena ou não.

O **terceiro componente** é uma transformação homogênea que representa os parâmetros extrínsecos da câmera. O objetivo é, obter $[X \ Y \ Z]^t$ que é a coordenada de P em relação ao referencial do mundo, pois até então a coordenada deste ponto estava representada em relação ao referencial da câmera.

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = \begin{pmatrix} R & T \\ 0 & 1 \end{pmatrix} \begin{pmatrix} X_{c1} \\ Y_{c1} \\ Z_{c1} \\ 1 \end{pmatrix} \quad (3.7)$$

Nota-se que o primeiro, segundo e terceiro componente, são responsáveis pela obtenção das coordenadas $[X \ Y \ Z]^T$, que é a projeção 3D do pixel P no plano (forma geométrica utilizada para representar a cena). Os próximos três componentes serão responsáveis por calcular como será a projeção desse ponto na nova imagem obtida após o deslocamento do robô.

Como podemos notar na figura 2.8, a forma como a cena será vista na próxima imagem depende totalmente de qual foi o deslocamento da câmera. Então precisamos dos parâmetros que representam esse deslocamento da câmera (deslocamento do robô). Caso esse deslocamento seja dado como conhecido o problema se torna o mapeamento do ambiente, caso contrário continua sendo um problema de SLAM, onde tanto os parâmetros do mapa como o do deslocamento (localização) devem ser estimados.

O **quarto componente** representa o deslocamento que ocorreu na câmera através de uma transformação homogênea com os seis parâmetros, onde três deles representam a rotação ocorrida e os outros três representam o deslocamento ocorrido. Partindo de $[X \ Y \ Z]^T$, obtemos $[X_{c2} \ Y_{c2} \ Z_{c2}]^T$ que é representação de P em coordenadas 3D da câmera

depois do deslocamento. Em outras palavras: voltamos a ter a coordenada do ponto em relação à câmera, mas agora a câmera sofreu um deslocamento, representando o deslocamento do robô no espaço 3D. Modelamos isso através de uma transformação homogênea.

$$\begin{pmatrix} X_{c2} \\ Y_{c2} \\ Z_{c2} \\ 1 \end{pmatrix} = \begin{pmatrix} R & T \\ 0 & 1 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \quad (3.8)$$

O **quinto componente** é responsável por uma transformação 3D para 2D, onde através do modelo pinhole de projeção obtemos $[X_{img2} \ Y_{img2} \ 1]^T$. A distância focal é representada por f .

$$X_{img2} = f \cdot (X_{c2}/Z_{c2}) \quad (3.9)$$

$$Y_{img2} = f \cdot (Y_{c2}/Z_{c2}) \quad (3.10)$$

No **sexto componente** é aplicada a matriz de parâmetros intrínsecos para obter as coordenadas $[u' \ v' \ 1]^t$ de P^* que é a projeção de P no outro ponto de vista de acordo com os supostos parâmetros do plano e do deslocamento da câmera.

$$\begin{pmatrix} u' \\ v' \\ 1 \end{pmatrix} = (K) \begin{pmatrix} X_{img2} \\ Y_{img2} \\ 1 \end{pmatrix} \quad (3.11)$$

Então ao final destas seis etapas, temos a coordenada de pixel $[u' \ v' \ 1]^t$ da projeção P^* que é onde o pixel P deverá aparecer no outro ponto de vista de acordo com os supostos parâmetros do plano e do deslocamento da câmera passados para o modelo de transformação. Note que foi calculado a projeção de apenas um pixel, então é necessário calcular essa projeção para todos os pixels da região de interesse.

Agora entra em ação o registro direto de imagem. Caso os parâmetros passados representem perfeitamente tanto o plano (mapa) como o deslocamento da câmera (localização), então a diferença entre a intensidade do pixel na coordenada de P^* da imagem capturada após o deslocamento real e a intensidade do pixel P da imagem de referência deve ser próxima a zero. Isso significa que o modelo de transformação, usando os parâmetros correntes, gerou a transformação que representa o movimento real do robô e os parâmetros corretos do mapa.

Quando essa diferença de intensidade for calculada para todos os pixels de uma certa região da imagem ela é conhecida como **erro de reprojeção**. Deve ser realizada uma varredura na região de interesse na imagem de referência, calculando o erro de reprojeção entre a projeção obtida pelo modelo de transformação e a imagem real capturada pelo robô após o deslocamento.

As figuras 3.10 e 3.9 mostram exemplos de parâmetros que geraram um grande erro e um pequeno erro de reprojeção, respectivamente. As regiões em vermelho representam as regiões de interesse.

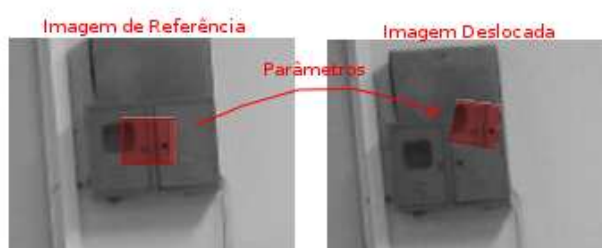


Figura 3.9: Exemplo de péssimos parâmetros que geraram um grande erro.

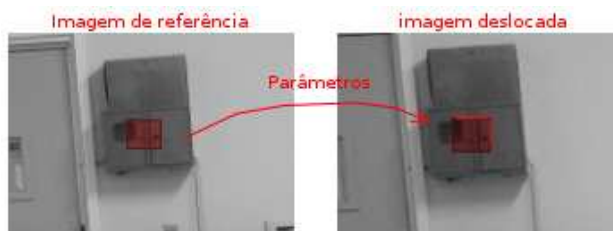


Figura 3.10: Exemplo de bons parâmetros que geraram um erro pequeno.

É importante salientar que podemos ter mais de uma região de interesse na imagem de referência, Então as etapas descritas acima devem ser repetidas para cada região.

Segunda versão do sistema

Na primeira versão do sistema proposto considera-se que o mundo é formado por segmentos de planos, o que representa uma das restrições à serem contornadas nesse trabalho. Com esse intuito o modelo de transformação geométrica foi implementado de tal forma que é possível mudar as equações paramétricas que representam o mapa sem grandes alterações no sistema.

Na segunda versão do sistema ocorreu a expansão desse modelo para tratar tanto planos como esferóides. Se classificarmos as possíveis formas de representação do mundo

por complexidade, como mostra a figura 3.11, partimos da forma mais simples: o ponto, depois temos a reta, depois o plano e a próxima forma em grau de complexidade é um esferóide.

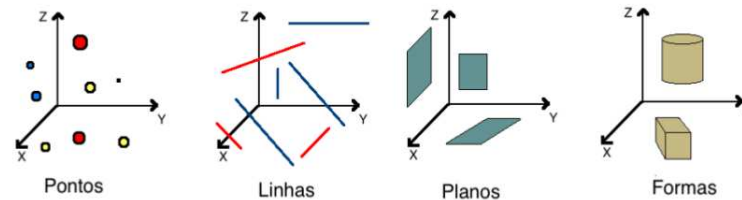


Figura 3.11: Evolução em termos de complexidade das diversas formas de representação do mundo.

A alteração principal encontra-se no **segundo componente** do modelo de transformação. Observa-se que na primeira versão foi calculada a interseção da reta que passava pela posição da câmera e pelo ponto $[X_{img1} \ Y_{img1} \ Z_{img1}]^T$ com o plano de parâmetros α, β, ρ . Então é necessário acrescentar as equações paramétricas que representem um esferóide no mundo 3D e alterar o cálculo da interseção. Também temos a adição de um parâmetro no vetor de parâmetros, que representa o raio do esferóide: $[x \ y \ z \ R_x \ R_y \ R_z \ \rho \ \alpha \ \beta \ R]^T$

O primeiro passo para fazer com que o sistema funcione com essa nova forma geométrica foi obter a parametrização geométrica da mesma. Ficou convencionado que o esferóide seria representado por uma distância e dois ângulo até o seu centro, um raio e dois ângulos de atuação como mostra a figura 3.12.

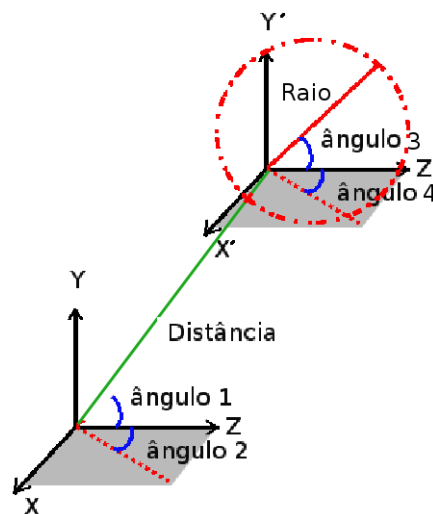


Figura 3.12: Parametrização do esferóide

De acordo com essa convenção obteve-se as seguintes equações paramétricas:

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = R \begin{pmatrix} \cos(\phi) \cdot \sin(\theta) \\ -\sin(\phi) \\ \cos(\phi) \cdot \cos(\theta) \end{pmatrix} + \rho \begin{pmatrix} \sin(\alpha) \cdot \sin(\beta) \\ \cos(\alpha) \\ \sin(\alpha) \cdot \cos(\beta) \end{pmatrix} \quad (3.12)$$

Onde R representa o raio da esfera; ϕ e θ representam os ângulos de atuação desse raio, ou seja, delimitam o pedaço da esfera que está sendo utilizado (esferóide). Já os parâmetros ρ , α e β representam o centro da esfera em coordenadas polares, ou seja, uma distância e dois ângulos.

O próximo passo é calcular a interseção entre a reta que passa pela posição da câmera e pelo ponto $[X_{img1} \ Y_{img1} \ Z_{img1}]^T$ com o esferóide parametrizado seguindo as convenções adotadas. Essa interseção é representada pelas equações abaixo:

$$X_{cam} - \rho \cdot \cos(\alpha) \cdot \sin(\beta) = -X_{img1} \cdot \lambda_r + R \cdot \cos(\phi) \cdot \sin(\theta) \quad (3.13)$$

$$Y_{cam} + \rho \cdot \sin(\alpha) = -Y_{img1} \cdot \lambda_r - R \cdot \sin(\phi) \quad (3.14)$$

$$Z_{cam} - \rho \cdot \cos(\alpha) \cdot \cos(\beta) = -Z_{img1} \cdot \lambda_r + R \cdot \cos(\phi) \cdot \cos(\theta) \quad (3.15)$$

Onde λ_r , ϕ e θ são as incógnitas a serem calculadas. Os demais parâmetros são conhecidos nessa etapa (parâmetros correntes passados pela otimização) e representam o centro da esfera (ρ , α e β) e o raio (R).

Tendo calculado ϕ e θ , basta substituir nas equações abaixo para obter as coordenadas 3D $[X \ Y \ Z \ 1]^t$ do ponto de interseção.

$$X = R \cdot \cos(\phi) \cdot \sin(\theta) - \rho \cdot \cos(\alpha) \cdot \sin(\beta) \quad (3.16)$$

$$Y = -R \cdot \sin(\phi) + \rho \cdot \sin(\alpha) \quad (3.17)$$

$$Z = R \cdot \cos(\phi) \cdot \cos(\theta) - \rho \cdot \cos(\alpha) \cdot \cos(\beta) \quad (3.18)$$

É importante notar que quando estamos tratando diferentes pontos de vistas de uma esfera ou esferóide, após deslocamentos de translação ou rotação da câmera, percebe-se que há uma perda natural de pixels na borda oposta ao movimento da câmera e o surgimento de pixels novos na outra borda. Em outras palavras, Imagine uma bola de futebol na frente de uma câmera. Caso a câmera se desloque para direita alguns pixels da parte esquerda dessa bola não aparecerão na próxima imagem, enquanto que alguns pixels da parte direita da bola, que não apareceram no primeiro ponto de vista, passaram a aparecer. A figura 3.13, repetida propositalmente, mostra um exemplo do que foi dito,

onde a região em vermelho representa os pixels que deixaram de aparecer na imagem após deslocamento e a região em verde mostra os pixels que passaram a aparecer.



Figura 3.13: Exemplo de pixels que desaparecem ou aparecem de acordo com o deslocamento da câmera.

Esse problema foi tratado pelo modelo de transformação e registro de imagem, pois os pixels que desaparecem são automaticamente excluídos da etapa de registro de imagem. Espera-se, futuramente, utilizar esses pixels para refinar as estimativas obtidas pelo sistema. Uma possível forma de conseguir isso é pensar que o número de pixels que deixaram de aparecer no próximo ponto de vista pode dizer algo sobre o tamanho do esferóide ou sobre o deslocamento da câmera.

Esses pixels que desaparecem são facilmente obtidos através do segundo componente do modelo de transformação, onde é realizado o cálculo da interseção: os casos onde não ocorrem a interseção podem representar um pixel que deixou de aparecer. O desafio é identificar o pixel que deixou de aparecer pela oclusão ou por alguma falha na captura da próxima imagem.

Outro problema a ser tratado futuramente é sobre a seleção automática de regiões de interesse na imagem. Selecionar qual parte da imagem representa um plano, uma esfera ou outra forma geométrica no mundo real, sem extração de características, não é um trabalho trivial. Uma possível solução é dividir a imagem em quadrículas e rejeitar aquelas que não possuem textura. Depois, sabendo-se o deslocamento da câmera, o modelo calcula o erro de reprojeção de cada quadrícula. Aquelas que tiverem um erro alto são rejeitadas. As que tiverem um erro aceitável são consideradas representações da forma geométrica utilizada no modelo. Por fim, as que tiverem um erro médio serão subdivididas em quadrículas menores e o cálculo é refeito. No final sobra apenas a parte da imagem que representa a forma geométrica em questão. É notável que quanto mais complexas as formas geométricas utilizadas mais problemas devem ser tratados.

3.4 Conclusão do capítulo

Vimos nesse capítulo qual o problema que o presente trabalho propõe resolver e como se pretende resolver esse problema. A solução encontrada foi implementar uma nova técnica de SLAM Visual que, além de ser livre das etapas de extração e correspondência de características, é extremamente versátil sendo capaz de se adequar à diferentes situações, tanto na questão da forma geométrica utilizada para representar a cena, como nos algoritmos de otimização que melhor se adequam em cada situação.

Foram apresentadas uma metodologia e as duas versões do sistema desenvolvido, que foram implementadas e testada com sucesso, provando que os principais objetivos dessa tese foram atingidos.

Os resultados dos testes realizados serão expostos no próximo capítulo.

Capítulo 4

Resultados Experimentais

O objetivo desse capítulo é comprovar, através da exposição dos resultados obtidos, a flexibilidade do sistema implementado em diversos ambientes e mostrar que, pelo fato do sistema ser desacoplado, foi possível testar vários métodos de otimização e alterá-lo para funcionar com várias formas geométricas.

4.1 Informações Preliminares

Os experimentos que veremos nesse capítulo foram realizados em ambientes controlados. Veremos os resultados das duas versões do sistema, com planos e esferoides, provando assim a versatilidade da técnica desenvolvida.

Todos os testes foram realizados com imagens capturadas manualmente, simulando um deslocamento de um robô no espaço 3D. Nos ambientes de teste se conhecem os parâmetros que representam o deslocamento da câmera e os parâmetros do mapa. Assim é possível ter um controle maior sobre o experimento e fazer um comparativo entre o que se sabe e o que o sistema nos retornou. Em alguns testes, para simular um caso real, essas informações foram desconsideradas. Ao invés disso foi utilizado um sinal de controle fictício com erros implícitos para simular um caso real de um robô móvel recebendo comandos de deslocamentos no ambiente desconhecido. Esses erros implícitos representam, por exemplo, um deslizamento da roda do robô.

Esses sinais de controles são utilizados pelo método de otimização para estabelecer a inicialização e os limites do espaço de busca dos parâmetros que representam a pose do robô. Em outras palavras, caso o comando seja para o robô se deslocar cinco centímetros para esquerda, o espaço de busca da otimização será nas redondezas daquele ponto.

Foram realizados vários tipos de experimentos: com imagens sintéticas para se obser-

var dados como tempo de processamento, outros com sequências capturadas em ambientes internos e depois com sequências de ambientes externos. Na figura 4.1 pode ser visto alguns frames dessas sequências. Também foram realizados experimentos para analisar o quanto de precisão se perde ao se reduzir o número de regiões de interesse selecionados na imagem de referência. Por fim um experimento compara os resultados obtidos com a técnica desenvolvida nesse trabalho e uma técnica já consagrada na literatura.



Figura 4.1: Alguns frames das sequências de imagens utilizada nos testes.

Em cada situação foram aplicados três métodos de otimização (Levenberg-Marquadt, Algoritmo Genético e Poliedro Flexíveis) e analisados onde cada um se adequa melhor, levando em conta a precisão e o tempo de processamento. Os parâmetros utilizados nos algoritmos de otimização foram:

- No algoritmo genético foi utilizada uma população de 50 indivíduos com uma taxa de mutação de 15% , de sobrevivência de 5% e o critério de parada foi de 100 gerações ou uma diferença de 0.9 entre o valor do melhor e pior indivíduo;
- No poliedro flexível, as taxas de reflexão, expansão, redução e contração foram de 1 , 1.5 , 0.5 e 0.5 respectivamente e o critério de parada foi quando a distância entre o melhor e pior vértice for menor que 0.1;
- No método de Levenberg-Marquadt foi fixado o critério de parada de quando a diferença entre o valor da função em uma iteração e o valor na iteração passada for menor que 0,1.

Em testes preliminares com imagens sintéticas monocromáticas, foram testados, inúmeras vezes, os algoritmos: Levenberg-Marquadt, algoritmo genético e poliedros flexíveis, com objetivo de obter as médias do tempo e do número de iterações que cada um leva para atingir o mínimo global. (tabela 4.1).

A partir desse teste preliminar constatou-se que a utilização de algoritmos genéticos nos fornece bons resultados, provavelmente, devido ao fato de ser robusto aos mínimos

Tabela 4.1: Valores médios do número de iterações, tempo em milissegundos e da função minimizada.

Métodos	Iterações	Tempo	Valor da função
<i>PoliedrosF.</i>	126,591	16,481	1,234
<i>A.Geneticos</i>	90,712	689,931	0,291
<i>LevenbergM.</i>	44,806	6,072	3,112

locais, porém seu tempo de processamento é muito alto. Os métodos do poliedro flexível e Levenberg-Marquadt se mostraram eficientes do ponto de vista computacional, mas não apresentaram resultados precisos devido ao fato de serem mais suscetíveis aos mínimos locais.

A seguir foram realizados testes com imagens reais de ambientes internos e externos e em ambos os testes é feita uma comparação entre o movimento estimado e o movimento conhecido da câmera (entre o mapa estimado e o mapa conhecido).

Ficou convencionado que o sistema de referência da câmera ficaria como exposto na figura 4.2, com eixo Z saindo da câmera para a cena e os respectivos eixos obedecendo a regra da mão direita.

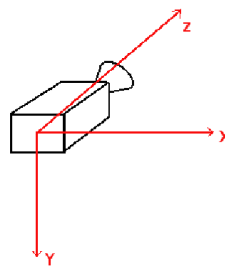


Figura 4.2: Sistema de referência da câmera.

4.2 Experimentos

Nesta seção encontraremos uma explicação sobre os experimentos realizados. Devido ao grande número de imagens, gráficos e tabelas, explicaremos cada experimento de forma separada e na próxima seção, serão expostos os resultados de cada um.

4.2.1 Experimento 1: Teste em Ambiente Interno

Nesse experimento os testes foram realizados em um ambiente interno e foi utilizada uma câmera não calibrada. É importante frisar que o fato da câmera não ser calibrada, nos retorna medidas de deslocamento em unidades de pixels. A calibração da câmera apenas converterá essas unidades para centímetros. A figura 4.3 mostra o ambiente onde foi realizado o teste. As regiões em vermelho representam as regiões de interesse.

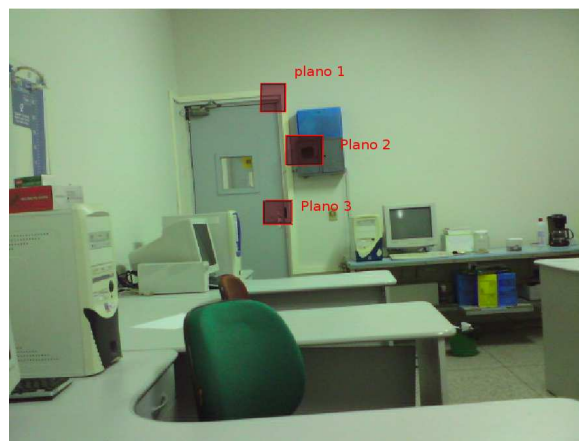


Figura 4.3: Ambiente de teste do experimento 1.

4.2.2 Experimento 2: Teste em Ambiente Interno 2



Figura 4.4: Ambiente de Teste do Experimento 2.

O experimento 2 foi realizado em outro ambiente interno (figura 4.4), diferente do primeiro experimento, onde o movimento predominante são translações no eixo x , o movimento predominante ocorreu ao longo do eixo z .

4.2.3 Experimento 3: Teste em Ambiente Externo



Figura 4.5: Ambiente externo com as respectivas regiões de interesse.

Agora o teste foi realizado em um ambiente externo (figura 4.5), onde as mudanças na iluminação podem atrapalhar a estimação dos parâmetros. Também foi utilizada uma câmera não calibrada e assim como no experimento 1, os ângulo de rolagem (em torno do eixo z), arfagem (em torno do eixo x) variaram de forma pouco significativa e por esse motivo não serão expostas.

4.2.4 Experimento 4: Teste em ambiente Externo 2



Figura 4.6: Segundo ambiente externo e as regiões de interesse.

Esse experimento ocorreu em outro ambiente externo e assim como no caso anterior temos o problema da mudança de iluminação. A diferença entre o experimento 3 e 4 está

na seleção de um segmento de plano que pertence à um plano diferente dos demais. O solo nesse experimento é mais homogêneo que no experimento anterior, provocando assim poucas variações nos ângulo de rolagem (em torno do eixo z), e arfagem (em torno do eixo x)

4.2.5 Experimento 5: Teste Alternando o número de Regiões

Notou-se nos experimentos anteriores uma carga computacional muito grande, então nesse experimento foi testado uma diminuição gradativa das regiões de interesse e os resultados foram comparados.

Neste experimento foram utilizados os dados do segundo ambiente externo (figura 4.6) e o método de otimização utilizado foi o Algoritmo genético com uma população de 50 indivíduos com uma taxa de mutação de 15%, de sobrevivência de 5% e um critério de parada foi de 100 gerações ou uma diferença de 0.9 entre o valor do melhor e pior indivíduo; O objetivo foi analisar como a precisão na estimativa de parâmetros é prejudicada com um número reduzido de regiões.

4.2.6 Experimento 6: Segunda Versão do sistema

Neste experimento foi testado a segunda versão do sistema, onde uma nova forma geométrica foi adicionada.

Nessa versão o sistema foi expandido para funcionar tanto com segmentos de planos, como com pedaços de esferas. Os testes foram realizados em um ambiente controlado, onde os parâmetros que representam o deslocamento da câmera são conhecidos e têm-se uma idéia sobre os parâmetros que representam o mapa. As figura 4.7 e 4.8 mostram o ambiente de teste.



Figura 4.7: Imagem de referência



Figura 4.8: Imagem deslocada

Nesse experimento utilizou-se as informações a priori sobre o deslocamento da câmera, ou seja, apenas os parâmetros relativos ao mapa foram estimados. Outra modificação em relação ao primeiro experimento está no fato de que foi utilizado apenas um deslocamento de câmera (duas imagens). O objetivo principal desse experimento foi testar se o sistema seria capaz de estimar os parâmetros da esfera presente na cena. A figura 4.9 mostra as regiões da imagem que foram utilizadas para estimar os parâmetros necessários.

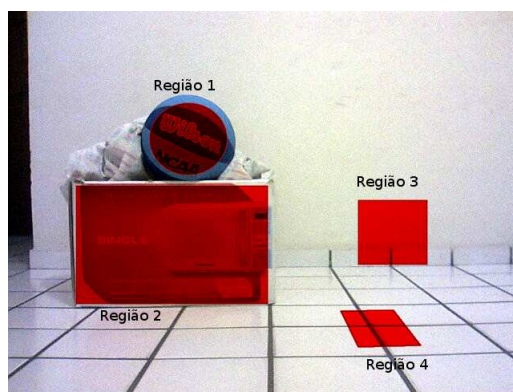


Figura 4.9: Regiões de interesse em vermelho

Como se observa na figura 4.9 foram selecionadas quatro regiões para registro. A região 1 representa um pedaço de esfera ou calota e deve ser descrito por seis parâmetros: três relativos ao centro (uma distância e dois ângulos) que descrevem a posição e orientação da esfera no espaço e três relativos ao formato e tamanho da calota (um raio e dois ângulos). As demais regiões selecionadas representam segmentos dos planos vistos na cena e devem ser descritos por cinco parâmetros: três descrevem o plano de forma única (uma distância e dois ângulos) e dois descrevem as dimensões do segmento de plano utilizado.

Nesse experimento a câmera não foi calibrada, retonado a medidas em pixels ao invés de centímetros e a origem do sistema de coordenada do mapa (mundo 3D) foi colocado na primeira posição de câmera, que capturou a imagem de referência. Também vale ressaltar que os dois parâmetros que representam o formato da calota (ϕ e θ) e os dois que representam o tamanho dos segmentos de plano (λ_1 e λ_2) não são estimados pelo algoritmo de otimização. Eles são calculados na etapa da interseção no segundo componente do modelo de transformação.

4.2.7 Experimento 7: Comparando com uma técnica consagrada

Nesse último experimento houve uma comparação de resultados. Os mesmo dados foram passados, tanto para a técnica desenvolvida no presente trabalho quanto para uma técnica já consagrada na literatura e implementada com objetivo de realizar esse experimento. Para isso o sistema de otimização foi implementado conforme foi apresentado no trabalho de Silveira et al. (2009), onde gerou-se uma equação derivável a partir dos parâmetros de transformação geométrica da cena, utilizando conceitos da álgebra de Lie [Warner 1987].

A maior diferença encontra-se na forma como os parâmetros ótimos são obtidos. A técnica já consagrada na literatura utiliza apenas segmentos de plano, tornando possíveis a modelagem do sistema em uma equação derivável e então aplica um procedimento não linear de otimização. Basicamente a função custo foi desenvolvida em série de Taylor, permitindo escrevê-la como um problema de mínimos quadrados linear.

O experimento ocorreu em um ambiente onde temos um terço das regiões de interesse representando esferóides. sabemos quais foram os deslocamentos ocorridos e também quais são os parâmetros que representam as regiões de interesse. A figura 4.10 mostra o ambiente de teste e as regiões de interesse:



Figura 4.10: Regiões selecionadas para processo de estimação

4.3 Resultados

4.3.1 Resultados Experimento 1

O experimento 1 foi realizado em um ambiente interno (Figura 4.3). Foram testados os 3 métodos de otimização propostos e os gráficos do deslocamento do robô no espaço e as variações do ângulo de guinada em radianos podem ser vistos nas figuras 4.11, 4.12 e 4.13.

Não houveram variações significativas nos ângulos de rolagem (em torno do eixo z), e arfagem (em torno do eixo x) e no deslocamento em y , visto que o piso do ambiente é plano.

Em relação ao mapa, foi calculado uma média entre os parâmetros das três regiões. Isso porque sabe-se que as três regiões fazem parte do mesmo plano e obteve-se os seguintes parâmetros: $\alpha = 0.0004$, $\beta = 0.0001$ e $\rho = 373$, onde α é o ângulo entre ρ e o eixo z da câmera; β é o ângulo entre a projeção ρ no plano xz e o eixo z da câmera e ρ é a menor distância entre o plano e a origem do sistema.

Os parâmetros que descrevem o tamanho do segmento de plano (λ_1 e λ_2) não são estimados. Eles são calculados no segundo componente do modelo de transformação, quando é feito o cálculo da interseção.

A tabela 4.2 mostra as melhores estimativas, obtidas quando se utilizou algoritmos genéticos, para cada segmento de plano. Os valores esperados foram obtidos manualmente usando o modelo de transformação geométrica.

Tabela 4.2: Valores dos parâmetros de cada segmento de plano do experimento 1.

Planos	α	β	ρ
<i>Regio1</i>	0.0009	0.0002	377
<i>Regio2</i>	0.0004	0.0005	370
<i>Regio3</i>	0.0001	-0.0003	376
<i>Valoresperado1</i>	0.0008	0.0000	373
<i>Valoresperado2</i>	0.0006	0.0001	373
<i>Valoresperado3</i>	0.0003	0.0000	373

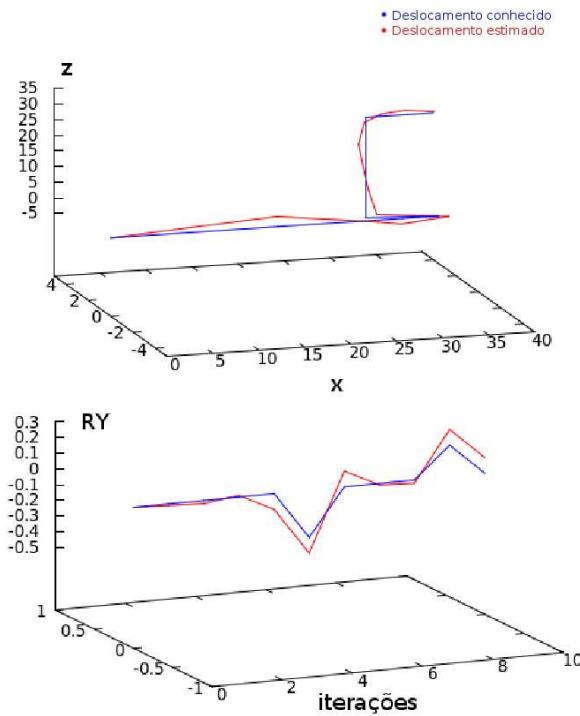


Figura 4.11: Resultados obtidos com Algoritmo genético.

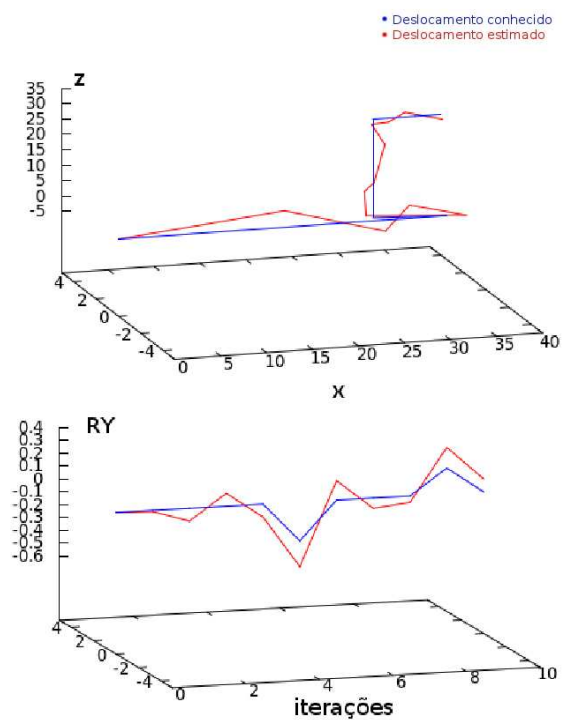


Figura 4.12: Resultados obtidos com poliedros flexíveis.

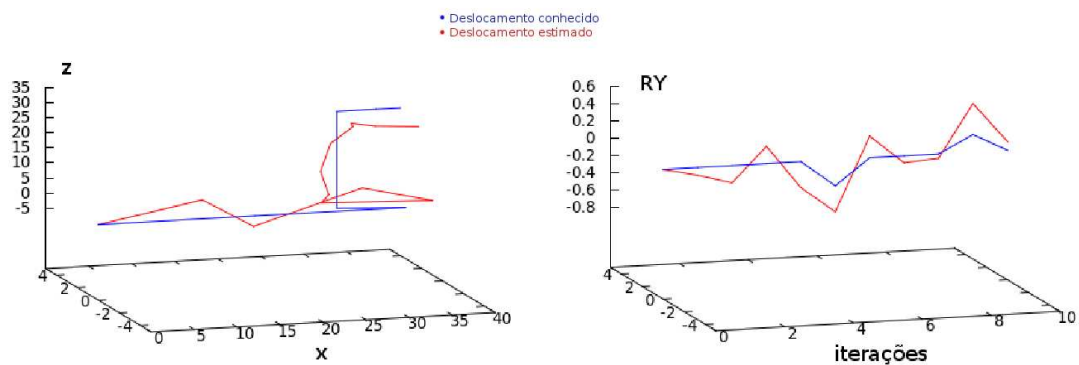


Figura 4.13: Resultados obtidos com Levenberg-Marquadt.

4.3.2 Resultados do Experimento 2

O experimento 2 também foi realizado em um ambiente interno (4.4) e o movimento predominante é ao longo do eixo z, porém nesse teste foi utilizada uma câmera calibrada. Isso nos retorna informações métricas sobre o deslocamento do robô. Os resultados estão apresentados nas figuras 4.14, 4.15 e 4.16, onde encontram-se os gráficos do deslocamento do robô no espaço e as variações do ângulo de guinada. Os melhores parâmetros

estimados do mapa encontram-se na tabela 4.7.

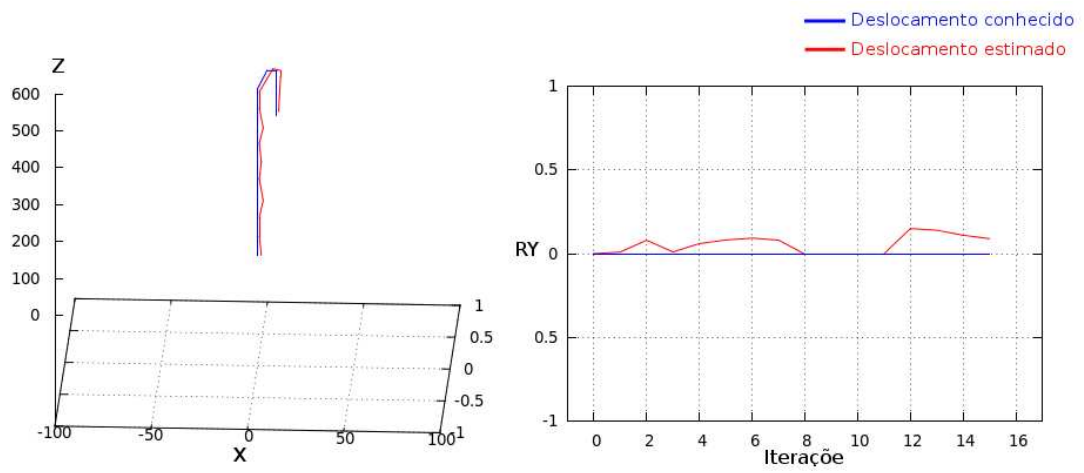


Figura 4.14: Resultados obtidos com Algoritmo Genético.

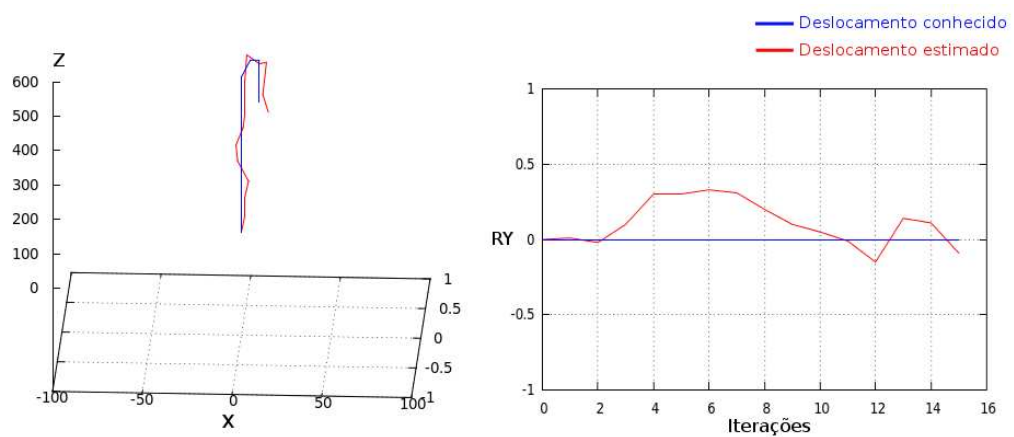


Figura 4.15: Resultados obtidos com Poliedro Flexível.

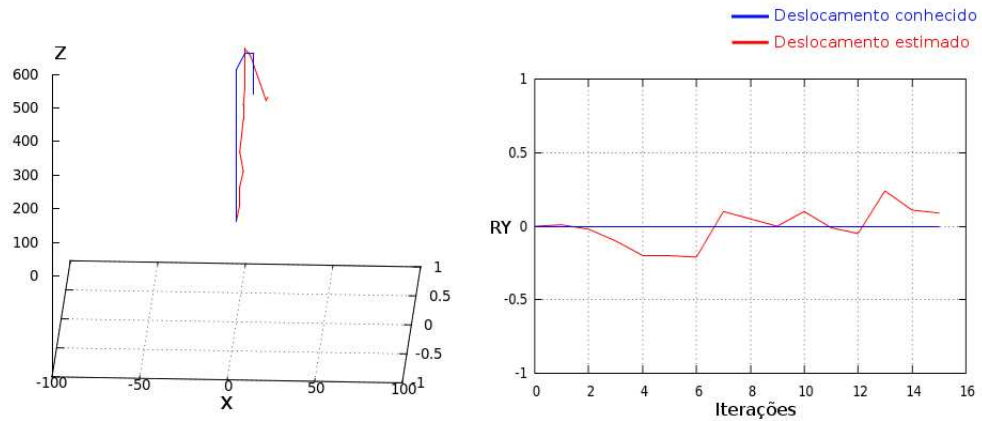


Figura 4.16: Resultados obtidos com Levenberg-Marquadt.

Tabela 4.3: Valores dos parâmetros de cada segmento de plano do experimento 2.

Planos	α	β	ρ
<i>Regio1</i>	0.0008	-0.0002	754
<i>Regio2</i>	0.0000	-0.0005	753
<i>Regio3</i>	-0.0001	0.0011	756
<i>Regio4</i>	-0.0002	0.0003	754
<i>Valoresperado1</i>	0.0008	-0.0001	745
<i>Valoresperado2</i>	0.0006	-0.0002	742
<i>Valoresperado3</i>	0.0000	0.0009	755
<i>Valoresperado4</i>	-0.0003	0.0004	747

4.3.3 Resultado do Experimento 3

O experimento 3 foi realizado em um ambiente externo (figura 4.5), sujeito à imperfeições do solo e mudanças de iluminação. Nas figuras 4.18, 4.19 e 4.17 encontram-se os gráficos do deslocamento do robô e na tabela 4.4 encontra-se os parâmetros do mapa estimados por cada método de otimização de forma separada.

Tabela 4.4: Valores dos parâmetros de cada segmento de plano do experimento 3.

Métodos	α	β	ρ
<i>PoliedrosF.plano1</i>	0.0091	-0.0126	822
<i>PoliedrosF.plano2</i>	0.0116	0.0127	758
<i>PoliedrosF.plano3</i>	-0.0012	0.0421	751
<i>A.Geneticosplano1</i>	0.0002	-0.06212	855
<i>A.Geneticosplano2</i>	-0.0001	0.0035	778
<i>A.Geneticosplano3</i>	0.0007	0.0002	779
<i>LevenbergM.plano1</i>	-0.0231	-0.4912	654
<i>LevenbergM.plano2</i>	0.0218	0.0522	738
<i>LevenbergM.plano3</i>	0.0126	0.0311	738
<i>Valoresperadodoplano1</i>	0.0002	-0.0600	850
<i>Valoresperadodoplano2</i>	0.0002	0.0030	785
<i>Valoresperadodoplano3</i>	0.0000	0.0003	785

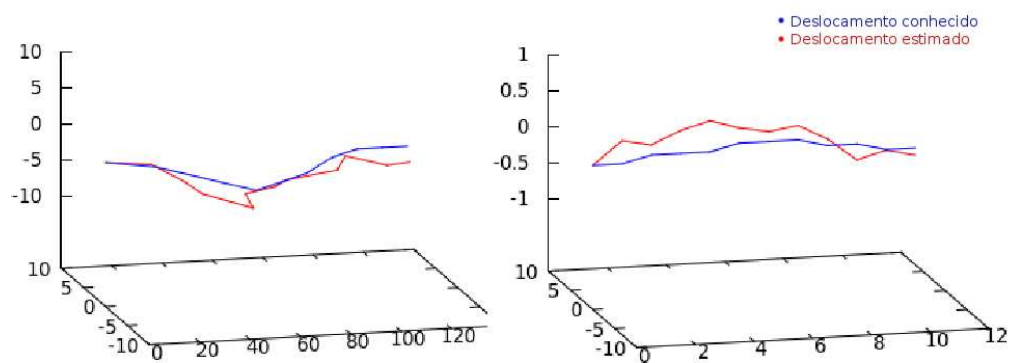


Figura 4.17: Resultados obtidos com Levenberg-Marquadt.

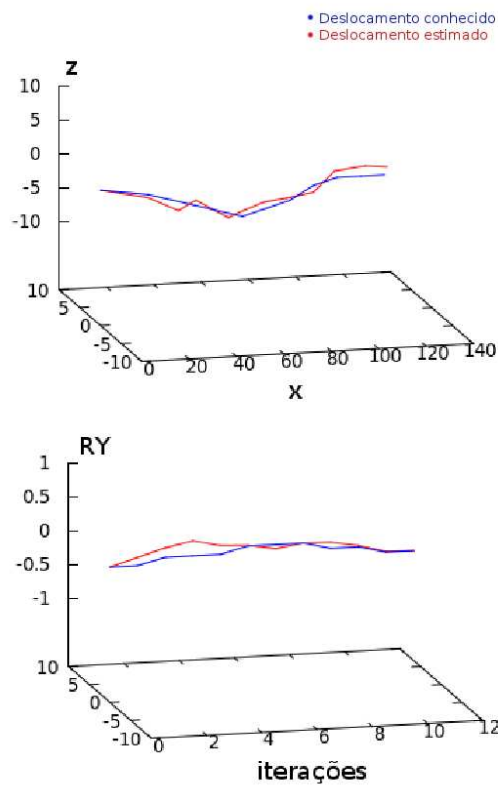


Figura 4.18: Resultados obtidos com Algoritmo genéticos.

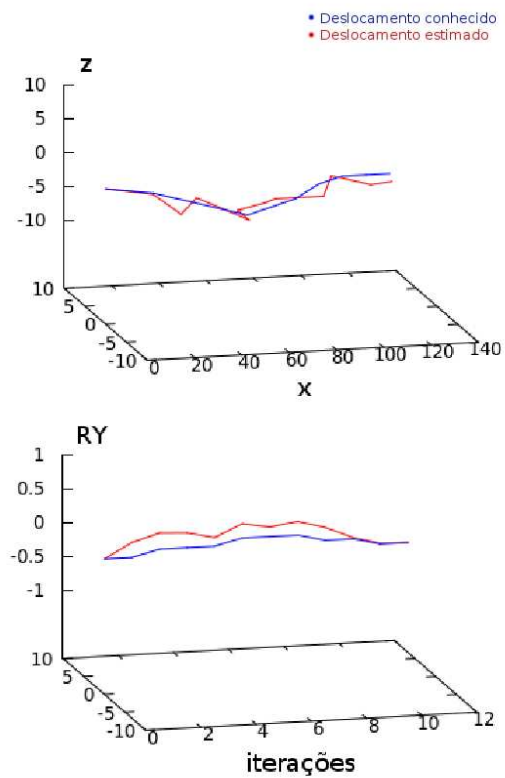


Figura 4.19: Resultados obtidos com poliedros flexíveis.

4.3.4 Resultados do Experimento 4

O experimento 4 também foi realizado em ambiente externo, com solo mais homogêneo e menos texturas na cena. Nas figuras 4.20, 4.21 e 4.22 encontram-se os gráficos do deslocamento do robô no espaço e as variações do ângulo de guinada em radianos. Na tabela 4.5 encontra-se os parâmetros estimados do mapa.

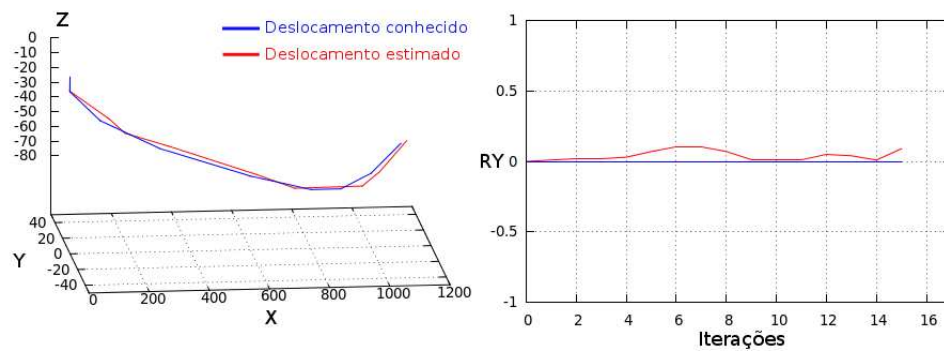


Figura 4.20: Resultados obtidos com Algoritmo Genético.

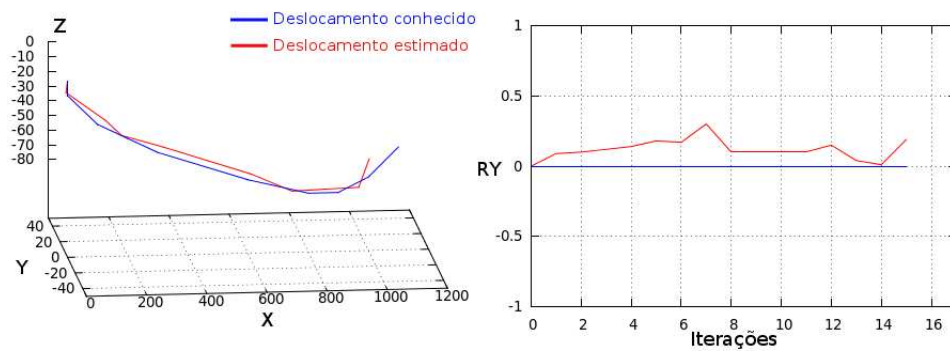


Figura 4.21: Resultados obtidos com Poliedro Flexível.

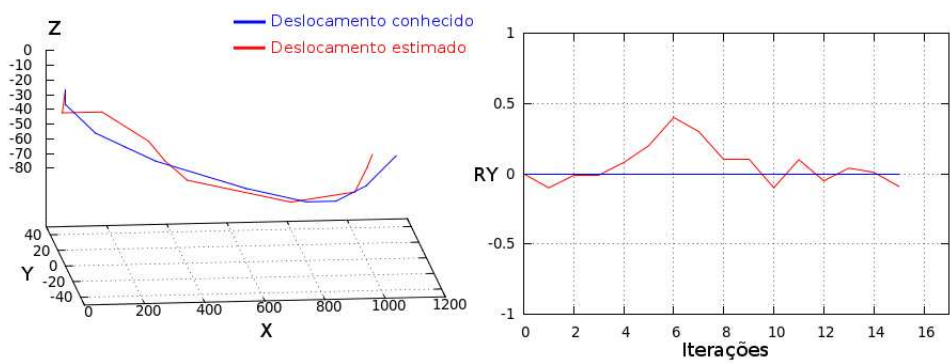


Figura 4.22: Resultados obtidos com Levenberg-Marquadt.

Tabela 4.5: Valores dos parâmetros de cada segmento de plano do experimento 4.

Métodos	α	β	ρ
<i>PoliedrosF.plano1</i>	0.0121	-0.0721	1222
<i>PoliedrosF.plano2</i>	0.0017	0.0241	722
<i>PoliedrosF.plano3</i>	0.0113	0.0282	721
<i>PoliedrosF.plano4</i>	0.0051	0.0421	732
<i>A.Geneticosplano1</i>	0.0034	-0.0522	1022
<i>A.Geneticosplano2</i>	0.0021	-0.0011	822
<i>A.Geneticosplano3</i>	0.0012	0.0301	823
<i>A.Geneticosplano4</i>	0.0032	0.0398	824
<i>Valoresperadodoplano1</i>	0.0000	-0.0020	1014
<i>Valoresperadodoplano2</i>	0.0000	0.0000	825
<i>Valoresperadodoplano3</i>	0.0001	0.0030	827
<i>Valoresperadodoplano4</i>	0.0000	0.0040	829

4.3.5 Resultados do Experimento 5

Neste experimento foi testado a precisão do sistema quando se altera o número de regiões de interesse. O resultado desse teste encontra-se nas figuras 4.23, 4.24, e na tabela 4.6 encontra-se um comparativo do tempo (em milissegundos) de convergência do algoritmo em cada teste com números diferentes de regiões.

Tabela 4.6: Tempo médio de convergência do algoritmo em cada situação.

N. de regiões	tempo médio de convergência
1Regiao	231.1
2Regioes	489.6
3Regioes	1052.3
4Regioes	2592.1

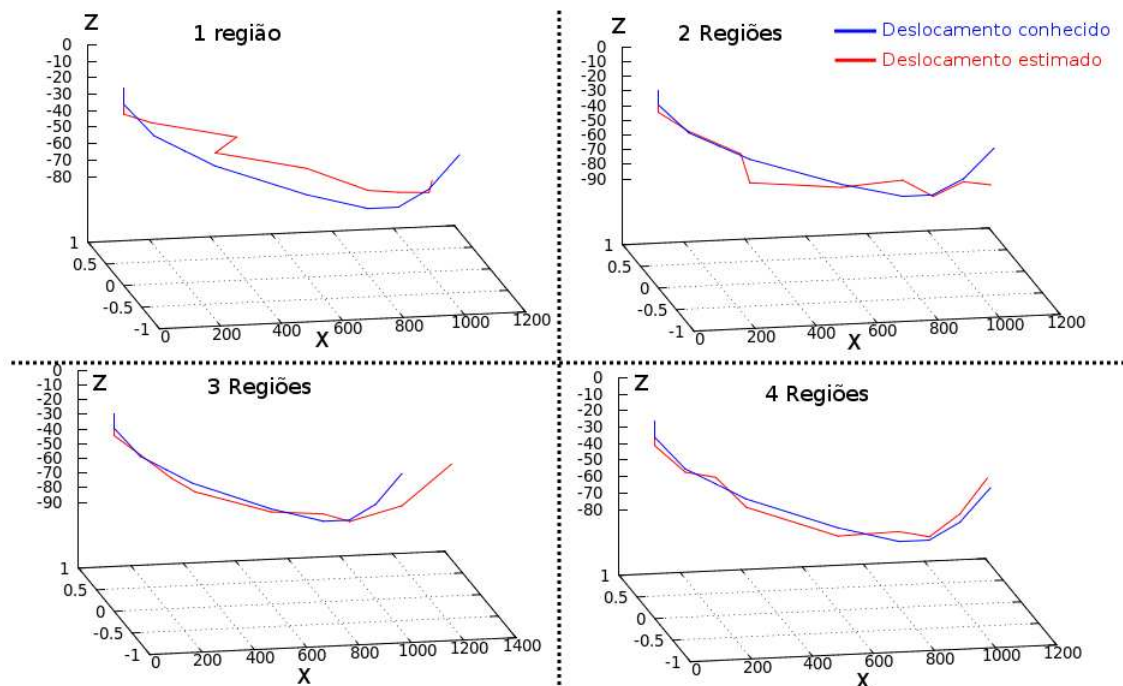


Figura 4.23: Resultados obtidos com diferentes números de regiões.

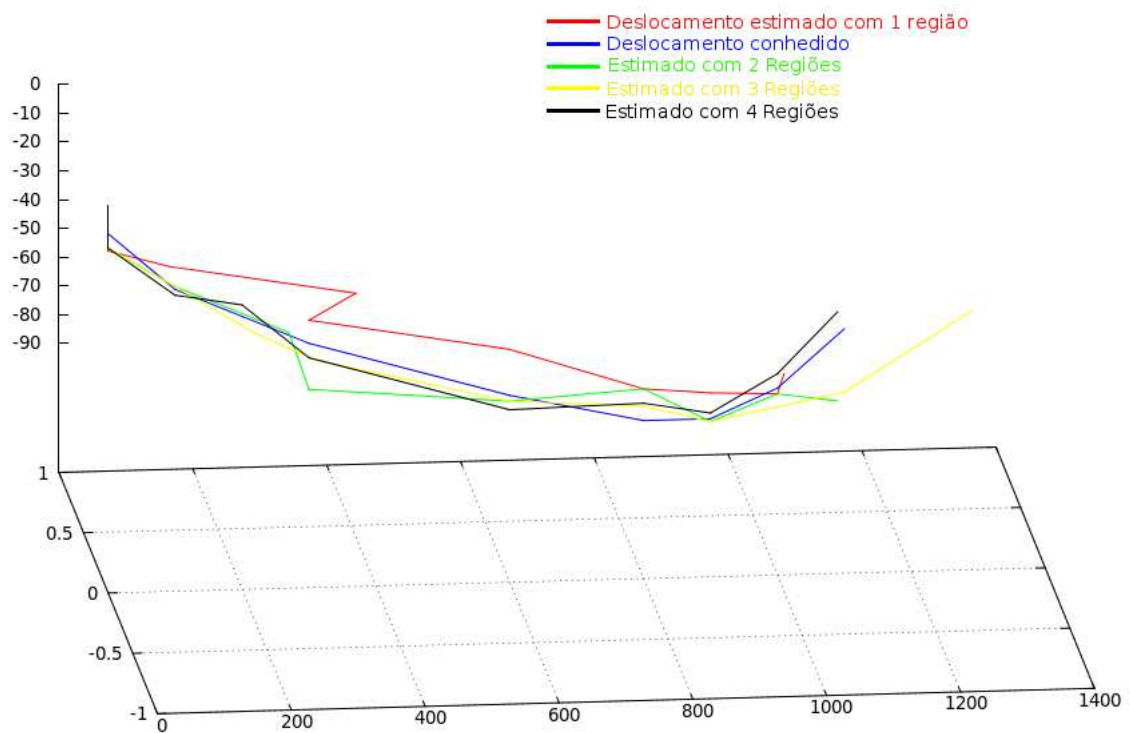


Figura 4.24: Resultados obtidos com diferentes números de regiões no mesmo gráfico.

4.3.6 Resultados do Experimento 6

Neste experimento, adicionou-se regiões de interesse que representam esferoides no mundo real. O objetivo foi validar a flexibilidade do sistema desenvolvido, ou seja, ele deve funcionar tanto com regiões planares como com regiões esféricas.

O sistema conseguiu estimar, tanto o deslocamento, quanto os parâmetros relativos às regiões de interesse. Os resultados obtidos foram os seguintes: Em relação à esfera obteve-se que o centro é representado pelos parâmetros: $\rho = 399$, $\alpha = 0.0001$ e $\beta = -0.0002$; e o raio foi estimado em $R = 65$.

Assim como na primeira versão, os dois parâmetros que representam o formato da calota (ϕ e θ) e os dois que representam o tamanho dos segmentos de plano (λ_1 e λ_2) não são estimados pelo algoritmo de otimização. Eles são calculados na etapa da interseção no segundo componente do modelo de transformação.

Em relação ao plano representado pela região 2 obteve-se os seguintes parâmetros: $\alpha = 0.0002$, $\beta = -0.0001$ e $\rho = 332$. Já o plano representado pela região 3 não teve resultado convincente, pois o algoritmo de otimização não convergiu e muitas vezes retornava o valor inicial de busca. Isso pode ser explicado pela falta de textura presente na mesma. O plano representado pela região 4 também teve problemas de convergência, porém em algumas ocasiões o algoritmo retornou os seguintes parâmetros (em média): $\alpha = -0.0020$, $\beta = 0.0014$ e $\rho = 298$. A tabela 4.7 resume os parâmetros estimados pelo sistema proposto. Os valores esperados foram obtidos manualmente usando o modelo de transformação.

Tabela 4.7: Valores dos parâmetros de cada região.

Regiões	α	β	ρ	R
<i>Regio1</i>	0.0002	-0.0002	399	62.5
<i>Regio2</i>	0.0001	-0.0001	332	— — —
<i>Regio4</i>	-0.0020	0.0014	298	— — —
<i>Valoresperado1</i>	0.0001	0.0000	400	60
<i>Valoresperado2</i>	0.0000	0.0000	350	— — —
<i>Valoresperado4</i>	0.0001	0.0009	300	— — —

4.3.7 Resultados do Experimento 7

O objetivo principal desse experimento foi comparar as estimativas obtidas com a técnica desenvolvida nesse trabalho em relação às estimativas obtidas com uma técnica já

consagrada na literatura. A figura 4.25 mostra o gráfico do deslocamento real e estimado. A linha azul representa o deslocamento real conhecido. A linha verde representa o deslocamento estimado pela técnica desenvolvida e a linha vermelha representa o deslocamento estimado pela técnica consagrada na literatura. É importante destacar que o deslocamento ocorreu em uma região plana e por esse motivo temos deslocamentos apenas no plano XZ.

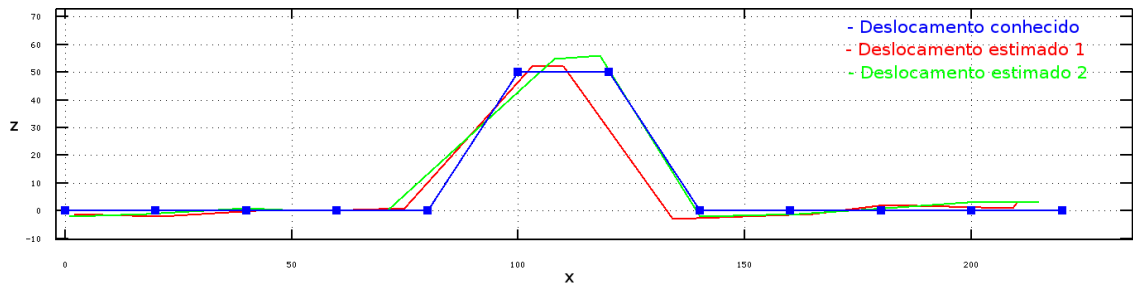


Figura 4.25: Deslocamento estimado pelas diferentes técnicas.

Em relação as estimativas obtidas do deslocamento, notamos que houve uma diferença muito sutil entre as duas técnicas, apesar de que a técnica consagrada na literatura utilizou, aproximadamente, 30% a menos de informação disponível para estimação dos parâmetros. Isso porque a mesma não foi capaz de estimar os parâmetros da região 1 que representa um esferoide.

Em relação ao mapa, podemos observar na tabela abaixo os parâmetros estimados pela técnica desenvolvida neste trabalho. Os parâmetros estão em unidades de pixels, pois foi utilizada uma câmera não calibrada.

Tabela 4.8: Valores dos parâmetros de cada região.

Regiões	α	β	ρ	R
<i>Regio1</i>	0.0003	-0.0002	205	45
<i>Regio2</i>	-0.0001	0.0001	193	— — —
<i>Regio3</i>	-0.0002	0.0004	197	— — —
<i>Valoresperado1</i>	0.0001	-0.0001	202	50
<i>Valoresperado2</i>	0.0000	0.0000	200	— — —
<i>Valoresperado3</i>	0.0000	0.0001	203	— — —

4.4 Conclusão do capítulo

Neste capítulo foram apresentados os resultados de vários testes realizados com a técnica desenvolvida, onde se tentou abordar várias e possíveis situações em que o sistema seria capaz de ser aplicado. As duas versões do sistema foram testadas com sucesso, provando assim a versatilidade, eficácia e robustez da técnica. Com isso os objetivos deste trabalho foram atingidos, gerando contribuições importantes para literatura. No próximo capítulo veremos a conclusão e perspectivas para trabalhos futuros.

Capítulo 5

Conclusões e perspectivas

5.1 Conclusão

Nas últimas duas décadas o problema conhecido como SLAM (localização e mapeamento simultâneos) foi alvo de intensa pesquisa e podemos encontrar diversos trabalhos com diferentes soluções para esse problema. Recentemente, impulsionados pelo avanço computacional, trabalhos nessa área propuseram usar câmera de vídeo como sensor e surgiu assim o SLAM Visual. Este possui várias abordagens e a grande maioria delas funcionam basicamente extraindo características do ambiente, calculando as devidas correspondências e através destas e de filtros estatísticos estimam os parâmetros necessários.

Neste trabalho foi apresentado um sistema de SLAM Visual que utiliza registro direto de imagem e métodos de otimização, de forma desacoplada, para estimar os parâmetros relativos à pose do robô e ao mapa do ambiente e dessa forma dispensa as fases de extração e correspondências de características, o que abre a possibilidade do sistema funcionar bem em ambientes nos quais as abordagens tradicionais teriam dificuldades. O fato de se usar registro direto nos permite explorar mais informações da imagem ao invés de reduzi-la a alguns pixels, como fazem os métodos clássicos.

O sistema se mostrou versátil, podendo utilizar vários métodos de otimização sem a necessidade de alterações na formulação matemática ou implementação do mesmo. Bastando apenas uma etapa de integração do método de otimização com o modelo de transformação geométrica e registro de imagem (ver 3.3). Com isso uma das contribuições esperadas foi atingida: implementar o SLAM Visual de forma desacoplada.

Outra vantagem obtida ao se abordar o problema do SLAM da forma proposta nesse trabalho é o fato de minimizar um problema muito comum nas abordagens tradicionais, conhecido como acúmulo ou propagação do erro. No nosso caso, esse problema não

ocorre porque, dado uma imagem de referência, todo cálculo do deslocamento é feito comparando as várias imagens sucessivas com a imagem de referência, até que as regiões da mesma não apareçam mais nas imagens sucessivas. A figura 1.7 exemplifica esta situação.

Nos sistemas robóticos há uma exigência de que os algoritmos funcionem em tempo real, por isso é importante alcançar um equilíbrio entre custo computacional e algoritmos robustos. Afim de procurar o melhor equilíbrio, o presente trabalho testou vários métodos de otimização, dos quais podemos tirar as seguintes conclusões:

- O uso de algoritmos genéticos retornam bons resultados devido a robustez à mínimos locais, porém o tempo de processamento é muito alto o que nos obriga a usar uma população reduzida e isso pode ocasionar uma perda de precisão.
- Usar poliedros flexíveis nos retorna resultados medianos em um tempo de processamento bem menor, porém esse método é, razoavelmente, suscetível à mínimos locais.
- O método de Levenberg-Marquadt é o que exige menos poder computacional, porém os resultados não foram tão bons.

Também foi possível utilizar outra forma geométrica para representação do mundo 3D sem a necessidade de grandes alterações na formulação matemática e na implementação do sistema, atingindo assim outra contribuição esperada. Essa contribuição não foi encontrada em nenhuma proposta de SLAM Visual direto pesquisada na literatura.

E para finalizar, no último experimento, onde ocorreu uma comparação com a técnica consagrada na literatura, podemos observar, que o fato de que o sistema desenvolvido pode funcionar com formas geométricas diferentes de planos pode trazer vantagens, caso a cena seja rica nessas formas. Por fim, todos os objetivos propostos no capítulo 1 deste documento foram atingidos.

5.2 Perspectivas

O sistema desenvolvido retornou bons resultados, porém, após alguns ajustes e pesquisas mais detalhadas o sistema pode se tornar mais eficiente e completo. Para conseguir isso é preciso:

- Estudar mais detalhadamente o método de Levenberg - Marquadt para tentar achar um ajuste de parâmetros que melhore os resultados.

- Testar exaustivamente algoritmos genéticos para chegar em uma população mínima que pode ser usada para obter bons resultados em tempo real, visto que o tempo de processamento está diretamente ligado ao número de indivíduos.
- Testar estratégias de combinação dos métodos testados para chegar à solução do problema de forma mais eficiente.
- Implementar, se necessário, um método de otimização próprio para o sistema.
- Melhorar a questão da seleção das regiões da imagem, que deve ser automatizada, se possível, por técnicas que não usam extração de características.
- Também é possível a adição de novas formas geométricas ao sistema, necessitando de pouco esforço, visto que o sistema é desacoplado.

Referências Bibliográficas

- Aires, Kelson Rômulo Teixeira (2009), *Segmentação de Planos Baseada em Homografia Afim, Fluxo Óptico e Reconstrução Métrica*, Universidade Federal do Rio Grande do Norte.
- Angeli, A., D. Filliat, S. Doncieux e J. Meyer (2008), ‘Fast and incremental method for loop-closure detection using bags of visual words’, *IEEE Transaction on Robotics*, Vol. 24, No. 5, pp. 1027-1037 .
- Anguelov, D., D. Koller, E. Parker e S. Thrun (2004), ‘Detecting and modeling doors with mobile robots’, *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pg: 3777-3784 .
- Artieda, Jorge, José M. Sebastian, Pascual Campoy, Juan F. Correa, Iván F. Mondragón, Carol Martínez e Miguel Olivares (2009), ‘Visual 3-d slam from uavs’, *J. Intell. Robotics Syst.* **55**, 299–321.
<http://portal.acm.org/citation.cfm?id=1569372.1569375>
- Ayache, Nicholas e Peter T. Sander (1991), *Artificial Vision for Mobile Robots: Stereo Vision and Multisensory Perception*, MIT Press, Cambridge, MA, USA.
- Barbara, Zitova e Flusser Jan (2003), ‘Image registration methods: a survey’, *Image and Vision Computing* **21**.
- Barnea, D.I. e H.F. Silverman (1972), ‘A class of algorithms for fast digital image registration’, *IEEE Transactions on Computing* .
- Bay, H., T. Tuytelaars e L. Van Gool (2006), ‘Object recognition from local scale-invariant features’, *European Conference on Computer Vision* .
- Benhimane, S. e E. Malis (2006), ‘Integration of euclidean constraints in template based visual tracking of piecewise-planar scenes’, *Proc. of the IEEE/RSJ IROS* .

- Bernardes, M. C. e G. A. Borges (2009), '3d line estimation for mobile robotics visual servoing', *Congresso Brasileiro de Automática (CBA)* .
- Borenstein, J. e L. Feng (1996), 'Measurement and correction of systematic odometry errors in mobile robots', *IEEE Transactions on Robotics and Automation* .
- Botelho, S. S. C., R. Mendes e L. Tadei (2005), 'Localization of a fleet of auvs using visual maps', *Oceans 2005*, pp. 1320- 1325 .
- Botelho, S. S. C., R. Mendes, L. Tadei e M. M. Teixeira (2003), 'Lambdair um robô subaquático autônomo', *Simpósio Brasileiro de Automação Inteligente* .
- Botelho, S. S. C., R. Neves e V. Oliveira (2004), 'Multi-auvs for mosaicking tasks', *IEEE OES Aiatolmnous Underwater Vehicles* .
- BROWN, LISA GOTTFELD (1992), 'A survey of image registration techniques', *ACM Computing Surveys* **24**.
- Campbell, J., R. Sukhankar e I. Nourbakhsh (2004), 'Techniques for evaluating optical flow for visual odometry in extreme terrain', *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3704-3711 .
- Chatila, R. e J. P. Laumond (1985), 'Position referencing and consistent world modeling for mobile robots', *IEEE International Conference on Robotics and Automation (ICRA)* .
- Chatterjee, Avishek Ray, Olive Chatterjee Amitava Rakshit Anjan (2011), 'Development of a real-life ekf based slam system for mobile robots employing vision sensing', *Expert Systems with Applications* .
- Chen, S. (2011), 'Kalman filter for robot vision: a survey', *IEEE Transactions on Industrial Electronics* .
- Choset, H., K. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. Kavraki e S. Thrun (2005), *Principles of Robot Motion*, The MIT Press.
- Civera, J., A. J. Davison e J. M. Montiel (2008), 'Inverse depth parametrization for monocular SLAM', *IEEE Transaction on Robotics*, Vol. 24, No. 5, pp. 932-945 .
- Clark, S. e G. Dissanayake (1999), 'Simultaneous localisation and map building using millimetre wave radar to extract natural features', *IEEE International Conference on Robotics and Automation* .

- Clemente, L., A. Davison, I. Reid, J. Neira e J. Tardos (2007), ‘Mapping large loops with a single hand-held camera’, *Proceedings of Robotics: Science and Systems* .
- Concha, Alejo, Giuseppe Loianno, Vijay Kumar e Javier Civera (2016), Visual-inertial direct slam, *em* ‘Robotics and Automation (ICRA), 2016 IEEE International Conference on’, IEEE, pp. 1331–1338.
- Conn A., Scheinberg K. e Toint P. (1996), ‘On the convergence of derivative-free methods for unconstrained optimization’, *Cambridge University Press* .
- Craig, J. J. (2004), *Introduction to Robotics: Mechanics and Control*, Pearson Education.
- Cremers, Daniel (2017), Direct methods for 3d reconstruction and visual slam, *em* ‘Machine Vision Applications (MVA), 2017 Fifteenth IAPR International Conference on’, IEEE, pp. 34–38.
- da Silva, Bruno Marques Ferreira (2015), ‘Registro global de nuvens de pontos rgbd em tempo real usando fluxo Óptico e marcadores’, *Universidade Federal do Rio Grande do Norte* .
- Davison, A. J., N. D. Molton e I. D. Reid (2004), ‘Locally planar patch features for real-time structure from motion’, *Proc. of the BMVC* .
- Davison, Andrew J. (2003), ‘Real-time simultaneous localisation and mapping with a single camera’, *International Conference on Computer Vision (ICCV)* .
- Davison, Andrew J. e David W. Murray (2002), ‘Simultaneous localization and map-building using active vision’, *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE* **24**(7).
- Davison, Andrew J., Hauke Strasdat e J. M. M. Montiel (2012), ‘Real-time monocular slam: Why filter?’, *Image and Vision Computing* .
- Davison, Andrew J., Ian D. Reid, Nicholas D. Molton e Olivier Stasse (2007), ‘Monoslam: Real-time single camera slam’, *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE* **29**(6).
- Durrant-Whyte, H. (1988), ‘Uncertain geometry in robotics’, *IEEE TRANSACTIONS ON ROBOTICS* **4**(1).
- Durrant-Whyte, H. e T. Bailey (2006), ‘Simultaneous localization and mapping: Part i’, *IEEE Transactions on Robotics and Automation*, pp. 99-108 .

- Elfes, A. (1987), ‘Sonar-based real-world mapping and navigation’, *IEEE Journal of Robotics and Automation*, pp. 249-265 .
- Elfes, A. (1989), Occupancy Grids: A Probabilistic Framework for Robot Perception and Navigation, Tese de doutorado, Electrical and Computer Engineering, Carnegie Mellon University.
- Fogel L., Owens A. J. e Walsh M. J (1966), ‘Artificial intelligence through simulated evolution’, *New York: John Wiley and Sons* .
- Forster, Christian, Matia Pizzoli e Davide Scaramuzza (2014), Svo: Fast semi-direct monocular visual odometry, em ‘Robotics and Automation (ICRA), 2014 IEEE International Conference on’, IEEE, pp. 15–22.
- Fu, S. e G. Yang (2009), ‘Uncalibrated monocular based slam for indoor autonomous mobile robot navigation’, *International Conference on Networking, Sensing and Control*, pp. 663-668 .
- Galindo, C., A. Saffiotto, S. Coradeschi, P. Buschka, J. A. Fernández-Madriral e J. González (2005), ‘Multi-hierarchical semantic maps for mobile robotics’, *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pg: 3492-3497 .
- Gee, A., D. Chekhlov, A. Calway e W. Cuevas (2008), ‘Discovering higher level structure in visual SLAM’, *IEEE Transaction on Robotics*, Vol. 24, No. 5, pp. 980-990 .
- Gendreau, M. e J. Potvin (2010), ‘Handbook of metaheuristics’, *International Series in Operations Research and Management Science* .
- Gil A., O. Mozos, M. Ballesta e O. Reinoso (2006), ‘A comparative evaluation of interest point detectors and local descriptors for visual slam’, *Machine Vision and Applications* .
- Goldberger, D. E. (1989), ‘Genetic algorithms in search, optimization, and machine learning’, *New York: Addison-Wesley Publishing* .
- Guapacha, Jovanny Bedoya (2017), ‘Localização e mapeamento simultâneos (slam) visual usando sensor rgb-d para ambientes internos e representação de características’, *UNIVERSIDADE ESTADUAL PAULISTA* .
- Guivant, J. e E. Nebot (2002), ‘Simultaneous localization and map building: Test case for outdoor applications’, *IEEE Int. Conference on Robotics and Automation* .

- Gutierrez, Rituerto A., Montiel J. Guerrero (2011), 'Adapting a real-time monocular visual slam from conventional to omnidirectional cameras', *Computer Vision Workshops, IEEE International Conference on* .
- Hanna, K.J. (1991), 'Direct multi-resolution estimation of egomotion and structure from motion', *Workshop on Visual Motion* .
- Harris, C. e M. Stephens (1998), 'A combined corner and edge detector', *Alvey Vision Conference* .
- Hata, A. Y., D. F. Wolf e F. S. Ososrio (2009), 'Terrain mapping and classification in outdoor environments using neural networks', *International Journal of u- and e-Service, Science and Technology* .
- Henry P., Kraing M. e D. Fox (2014), 'Rgb-d mapping: using depth cameras for dense 3d modeling of indoor environments.', *Berlin: Springer* .
- HOGMAN, V (2013), 'Build a 3d map from rgb-d sensors.', *Royal Institute of Tecnology, KTH, School of Computer Science and Communication (CSC)* .
- Holland, J. H. (1992), 'Adaptation in natural and artificial systems', *MIT Press* .
- Horn, B. K. P. e B. G. Schunck (1981), 'Determining optical flow', *Artif. Intell.*, pp. 185-203 .
- Irani, M. e P. Anandan (1999), 'About direct methods', in *Proc. Workshop Vis. Algorithms: Theory Pract.*, Corfu, Greece .
- Jakob Engel, Thomas Schöps, and Daniel Cremers (2016), 'Lsd-slam: Large-scale direct monocular slam', *Springer International Publishing* .
- Jin, H., P Favaro e S. Soatto (2003), 'A semi-direct approach to structure from motion', *The Visual Computer* 6 .
- Jing Wu, J. e H. Zhang (2007), 'Camera sensor model for visual SLAM', *IEEE Canadian Conference on Computer and Robot Vision* .
- Jones, S.D., C. Andresen e J.L. Crowley (1997), 'Appearance based processes for visual navigation', *IEEE International Conference. on Robotics and Automation*, pp. 551-557 .

- Kalman, R. E. (1960), 'A new approach to linear filtering and predictive problems', *Transactions ASME, Journal of basic engineering* .
- Klippenstein, J. e H. Zhang (2007), 'Quantitative evaluation of feature extractors for visual SLAM', *conference on Computer and Robot Vision* .
- Kriegman, D., E. Triendl e T. Biford (1987), 'Generation of object descriptions from range data using feature extraction by demands', *IEEE International Conference on Robotics and Automation (ICRA)*, Vol. 4, pp. 402-408 .
- Kuipers, B. e Y. Byun (1988), 'A robust, qualitative method for spatial learning in unknown environments', *National Conference on Artificial Intelligence* , pp. 1-12 .
- Kwon, Junghyun Kyoungh Mu Lee (2010), 'Monocular slam with locally planar landmarks via geometric rao-blackwellized particle filtering on lie groups', *The Twenty-Third IEEE Conference on Computer Vision and Pattern Recognition* .
- Lacerda, E. e A. Carvalho (1999), 'Introdução aos algoritmos genéticos', *Congresso Nacional da Sociedade Brasileira de Computação* .
- Lacroix, S., T. Lemaire e C. Berger (2008), 'More vision for slam', *Unifying Perspectives in Computational and Robot Vision*, Vol. 8, pp. 129-147 .
- Lategahn, Henning Geiger, Andreas Kitt Bernd (2011), 'Visual slam for autonomous ground vehicles', *IEEE International Conference on Robotics and Automation* .
- Latombe, J. (1991), *Robot Motion Planning*, Kluwer Academic Publishers.
- Lim, Hyon, Jongwoo Lim e H Jin Kim (2014), Real-time 6-dof monocular visual slam in a large-scale environment, *em 'Robotics and Automation (ICRA), 2014 IEEE International Conference on'*, IEEE, pp. 1532–1539.
- Limketkai, B., L. Liao e D. Fox (2005), 'Relational object maps for mobile robots', *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pg: 1471-1476 .
- Longuet-Higgins, H. C. (1981), 'A computer algorithm for reconstructing a scene from two projections', *Nature* **293**.
- Lookingbill, A., D. Lieb, D. Stavens e S. Thrun (2005), 'Learning activity-based ground models from a moving helicopter platform', *IEEE International Conference on Robotics and Automation (ICRA)*, pg: 3948-3953 .

- Lowe, D. (1999), 'Object recognition from local scale-invariant features', *International Conference on Computer Vision (ICCV)* .
- Lucas, B. D. e T. Kanade (1981), 'An iterative image registration technique with an application to stereo vision', *Proc. of DARPA Image Understanding Workshop*, pp. 121-130 .
- Luenberger, D. G. (1984), 'Linear and nonlinear programming', *Massachusetts Addison-Wesley* .
- Malis, E. (2004), 'Improving vision-based control using efficient second-order minimization techniques', *Proc. of the IEEE ICRA* .
- Martin, C. e S. Thrun (2002), 'Real-time acquisition of compact volumetric maps with mobile robots', *Proceedings of the International Conference on Robotics and Automation (ICRA)*, volume 1, pg: 311-316 .
- Matsumoto, Y., M. Inaba e H. Inoue (1996), 'Visual navigation using view sequenced route representation', *IEEE International Conference on Robotics and Automation*, pp. 83-88 .
- Merat, F. e F. Wu (1986), 'Generation of object descriptions from range data using feature extraction by demands', *IEEE International Conference on Robotics and Automation (ICRA)*, Vol. 4, pp. 941-946 .
- Mozos, O. M., R. Triebel, P. Jensfelt, A. Rottmann e W. Burgard (2007), 'Supervised semantic labeling of places using information extracted from sensor data', *Robotics and Autonomous Systems*, volume 55, no. 5, *From Sensors to Human Spatial Concepts*, pg: 391-402 .
- Murray, D. e G. Klein (2008), 'Improving the agility of keyframe based slam', *In Proceedings of the European Conference on Computer Vision* .
- Newcombe, Richard A., Andrew J. Davison (2011), 'Kinectfusion: Real-time dense surface mapping and tracking', *Proceedings of the 2011 10th IEEE International Symposium on Mixed and Augmented Reality* .
- Nocedal, J. e S. J Wright (2006), 'Numerical optimization', *Springer-Verlag New York* .
- Ouellette, R. e K. Hirasawa (2007), 'A comparison of SLAM implementations for indoor mobile robots', *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1479 - 1484 .

- Park, Seonwook, Thomas Schöps e Marc Pollefeys (2017), Illumination change robustness in direct visual slam, *em* 'Robotics and Automation (ICRA), 2017 IEEE International Conference on', IEEE, pp. 4523–4530.
- Paz, P. Pinies, J. Tardos e J. Neira (2008), 'Large-scale 6-dof slam with stereo in hand', *IEEE Transaction on Robotics*, Vol. 24, No. 5, pp. 946-957 .
- R. MurArtal, J. M. M. Montiel, and J. D. Tardos (2015), 'Versatile and accurate monocular slam system', *IEEE Transactions on Robotics* .
- Rangaiah, G. P. (2010), 'Stochastic global optimization - techniques and applications in chemical engineering', *Singapore: World Scientific Publishing* .
- Rechenberg, I. (1973), 'Evolutions strategie: optimierung technischer systeme nach prinzipien der biologischen evolution.', *Frommann-Holzboog* .
- S., Lucidi e Sciandrone M. (2002), 'On the global convergence of derivative-free methods for unconstrained minimization', *SIAM Journal on Optimization* .
- Santana, André Macêdo (2010), *Localização e Mapeamento Simultâneos de Ambientes Planos Usando Visão Monocular e Representação Híbrida do Ambiente*, Universidade Federal do Rio Grande do Norte.
- Santos-Victor, J. Sandini, G. Curotto F. e S. Garibaldi (1993), 'Divergence stereo for robot navigation: learning from bees', *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)* .
- Sharma, R.K. e M. Pavel (1997), 'Multisensor image registration', *Proceedings of the Society for Information Display XXVIII* .
- Silveira, Geraldo, Ezio Malis e Patrick Rives (2007), 'An efficient direct method for improving visual slam', *IEEE International Conference on Robotics and Automation* .
- Silveira, Geraldo, Ezio Malis e Patrick Rives (2008), 'An efficient direct approach to visual slam', *IEEE TRANSACTIONS ON ROBOTICS* 24(5).
- Silveira, Geraldo, Ezio Malis e Rives Patrick (2009), 'Registro direto de imagens para slam visual', *Simpósio Brasileiro de Automação Inteligente* .
- Smith, R., M. Self e P. Cheeseman (1987), 'On the representation of spatial uncertainty', *Int J Robot Res* 5.

- Spears, W. M. (2000), 'Evolutionary algorithms : the role of mutation and recombination', *Natural computing series ISBN* .
- Spendley W., Hext G. e Himsworth F. (1962), 'Sequential application for simplex designs in optimisation and evolutionary operation', *Technometrics* .
- Sun, W. e Y. Yuan (2006), 'Optimization theory and methods: nonlinear programming', *Springer Optimization and Its Applications* .
- Szeliski e Kang (1995), 'Direct methods for visual scene reconstruction.', *IEEE Workshop on Representation of Visual Scenes* .
- Szeliski, Richard (2006), 'Image alignment and stitching: A tutorial', *Foundations and Trends in Computer Graphics and Vision* 2, 1-104 .
- Tapus, A. e R. Siegwart (2006), 'A cognitive modeling of space using fingerprints of places for mobile robot navigation', *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pg: 1188-1193 .
- Thrun, S. (2002), 'Robotic mapping: A survey', *Academic Publish, Carnegie Mellon University - Pittsburgh* .
- Thrun, Sebastian, Frank Dellaert, Steven M. Seitz e Charles E. Thorpe (2000), 'Structure from motion without correspondence', *Computer Vision and Pattern Recognition Proceedings. IEEE Conference on* .
- Vasudevan, S., S. Gächter, V. Nguyen e R. Siegwart (2007), 'Cognitive maps for mobile robots - an object based approach', *Robotics and Autonomous Systems*, no. 55, pg: 359-371 .
- Vincent, A. (1987), 'Visual navigation for a mobile robot building a map of the occupied space from sparse 3d stereo data', *IEEE International Conference on Robotics and Automation (ICRA)*, Vol. 4, pp. 1429-1435 .
- Wang, C. (2004), Simultaneous Localization, Mapping and Moving Object Tracking, Tese de doutorado, Carnegie Mellon University - Pittsburgh.
- Warner, F. W. (1987), 'Foundations of differential manifolds and lie groups', *Springer Verlag* .

- Werneck, Nicolau Leal e Anna Helena Reali Costa (2010a), ‘Monocular visual mapping with the fast hough transform’, *Workshop on MSc Dissertation and PhD Thesis in Artificial Intelligence* .
- Werneck, Nicolau Leal e Anna Helena Reali Costa (2010b), ‘Slam monocular com reconstrução de planos para ambientes internos’, *WTDIA2010* .
- Wiley, John (2009), *Practical Image Processing and Computer Vision*, D.P. Robertson and R. Cipolla, capítulo 13.
- Wolf, D. F. e G. S. Sukhatme (2006), ‘Activity-based semantic mapping of an urban environment’, *International Symposium on Experimental Robotics (ISER)* .
- Zacarkim, Valber Lemes (2017), ‘Avaliação do detector de pontos de interesse igfft em visual slam’, *Universidade Federal do Paraná* .