UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

FERNANDA CAROLINE SILVEIRA RODRIGUES

Detecção de fechamento de *loop* utilizando sequências em uma estratégia hierárquica de três níveis

Dissertação apresentada como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação

Orientador: Profa. Dra. Mariana Luderitz Kolberg
Co-orientador: Prof. Dr. Edson Prestes e Silva Junior

Porto Alegre
2019

CIP — CATALOGAÇÃO NA PUBLICAÇÃO

Rodrigues, Fernanda Caroline Silveira

Detecção de fechamento de *loop* utilizando sequências em uma estratégia hierárquica de três níveis / Fernanda Caroline Silveira Rodrigues. – Porto Alegre: PPGC da UFRGS, 2019.

88 f.: il.

Dissertação (mestrado) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR–RS, 2019. Orientador: Mariana Luderitz Kolberg; Coorientador: Edson Prestes e Silva Junior.

1. Detecção de fechamento de *loop*. 2. Ambientes Dinâmicos. 3. Sequências. 4. Mapa hierárquico. I. Kolberg, Mariana Luderitz. II. Junior, Edson Prestes e Silva. III. Título.

“Nada nos limitava, nada nos definia, nada nos sujeitava;
nossas ligações com o mundo, nós é que as criávamos;
a liberdade era nossa própria substância.”

— SIMONE DE BEAUVOIR

AGRADECIMENTOS

Primeiramente agradeço aos meus pais que desde a infância me mostraram que o estudo sempre foi o melhor caminho, talvez a única trilha certeira na vida. À minha irmã que é tudo para mim, sempre esteve ao meu lado, divide minhas histórias e é, acima de tudo, minha amiga.

À professora Mariana por ter me dado a oportunidade de ingressar no mestrado, por ter me ajudado e especialmente por ter estado até o fim. Ao professor Renan que sempre esteve disponível e disposto a esclarecer minhas dúvidas. Ao professor Edson que respondeu meus e-mails sobre dúvidas de robótica ainda na faculdade, sem mesmo me conhecer, sendo portanto o responsável por eu querer entrar na UFRGS e especialmente nesse grupo. Aos meus amigos do φ-*Group* pelas incansáveis discussões e auxílios em vários momentos dessa trajetória, pelas risadas e pelo companheirismo. À Renata, que após tantas conversas sobre o trabalho e a vida, certamente é a melhor herança que o mestrado irá me deixar.

À minha amiga Érika que mesmo distante, sempre esteve comigo me apoiando e me sacudindo. Ao Augusto, que escolheu as minhas escolhas sem saber as proporções que elas alcançariam em nossas vidas, que me ajudou a chegar até aqui e resistiu as privações da minha vida acadêmica bem mais do que poderia.

RESUMO

Ao longo dos anos foi possível acompanhar a evolução das mais diversas aplicações robóticas. Robôs são utilizados em cirurgias e em linhas de montagem por conta de sua precisão e eficiência em trabalhos minuciosos e por vezes repetitivos. Quanto a robôs móveis, estes também podem ser aplicados em operações de busca e resgate, transporte, exploração subaquática, entre outras. Tais atribuições são possibilitadas se o robô for capaz de navegar autonomamente em ambientes desconhecidos. Para tanto, é importante que o robô tenha a capacidade de compreender a estrutura do ambiente em que está. Nesse sentido, foram desenvolvidos métodos de SLAM (Localização e Mapeamento Simultâneos) que proveem ao robô a habilidade de construir o mapa do ambiente enquanto se movimenta por ele. Para construir um mapa corretamente é importante que o robô consiga detectar que está revisitando um local de forma a compreender a topologia do ambiente. Os métodos utilizados para detecção de revisita, também conhecido como o problema de detecção de *loop*, são dependentes do sensor que o robô está utilizando. Recentemente, a câmera tem sido adotada nessa atividade por ser um sensor barato ao mesmo tempo em que é rico em informações. Por outro lado, traz desafios para o problema de detecção de *loop* como por exemplo: como saber que está revisitando o mesmo local quando se está em uma estação diferente, ou em um momento diferente do dia ou ainda, saber que é o mesmo local ainda que capturado sob um ponto de vista diferente? Nossa proposta tem o objetivo de responder essas perguntas através de uma estrutura que tira proveito da sequência de informações já coletadas pelo robô, junto a um mapa hierárquico que tem o papel de reduzir o espaço de busca sem interferir na precisão da tarefa da detecção de *loop*. Nossa abordagem é avaliada através de testes que expõem nossa solução a diferentes configurações de *datasets*. Esses testes demonstraram que o nosso método é capaz de reconhecer revisitas em locais com alteração de iluminação, movimentação de pessoas e diferenças de ponto de vista e apresenta resultados expressivos se comparado a um dos métodos estado da arte em LCD.

Palavras-chave: Detecção de fechamento de *loop*. Ambientes Dinâmicos. Sequências. Mapa hierárquico.

Three Level Sequence-Based Loop Closure Detection

ABSTRACT

Over the years it has been possible to observe the evolution of various robotic applications. Robots are used in surgeries and assembly lines because of their accuracy and efficiency in repetitive work. In the category of mobile robots, they can also be applied in search and rescue operations, transportation, underwater exploration, among others. To perform their duties, they must be able to navigate autonomously in unknown environments. Therefore, the robot must have the ability to understand the structure of these environments. In this way, SLAM (Simultaneous Location and Mapping) methods provide the robot the ability to construct the environment map while moving around it. To build a map correctly it is important that the robot can detect that it is revisiting a location to understand the topology of the environment in which it is located. The methods used for revisit detection, also known as the loop closure detection (LCD) methods, are dependent on the sensor the robot is using. Recently, the camera has been adopted in this activity because it is a cheap sensor while rich in information. On the other hand, it brings challenges to the LCD problem, e.g: how to know that you are revisiting the same location when you are at a different season, or at a different time of day, or knowing that it is the same location but captured from a different point of view? Our proposal aims to answer these questions through a method that takes advantage of the sequence of information already collected by the robot along with a hierarchical map that has the role of reducing the search space without interfering with the accuracy of the loop closure detection task. Our approach is assessed through tests that exposes our solution to different dataset configurations. These tests have shown that our method is capable of recognizing revisits in places with changing lighting, moving people, and differences of viewpoint and yields significant results compared to one of the state-of-the-art LCD methods.

Keywords: Loop Closure Detection, Dinamic Environments, Sequences, Hierarchical Map.

LISTA DE ABREVIATURAS E SIGLAS

LCD	*Loop Closure Detection*
BoW	*Bag of Words*
GPS	*Global Positioning System*
SLAM	*Simultaneous Localization and Mapping*
BLOB	*Binary Large OBject*
SIFT	*Scale-Invariant Feature Transform*
SURF	*Speeded Up Robust Features*
DoG	*Difference of Gaussians*
HOG	*Histogram of Oriented Gradients*
TF-IDF	*Term Frequency - Inverse Document Frequency*

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Ao longos dos anos foi possível acompanhar a evolução de diversas aplicações robóticas. As motivações são variadas, e envolvem desde redução de custos ao desenvolvimento de métodos que requisitam extrema precisão. Robôs podem auxilar na execução de cirurgias, sendo aplicados especialmente em procedimentos que necessitam de movimentos finos que eventualmente superam a habilidade humana. Por conta do grau de precisão, os robôs garantem a reprodutibilidade de técnicas cirúrgicas e consequentemente a melhoria dos resultados obtidos (BARGAR, 2007). Outro aspecto em que a robótica está sendo aplicada é no auxílio de pessoas com idade avançada. Projeções demográficas demonstram o aumento da população idosa e consequentemente o crescimento do número de acometidos por doenças que necessitam de acompanhamento e monitoramento constante. Nesse sentido, os robôs podem ser aplicados para reduzir a lacuna entre a escassez de recursos profissionais e a demanda crescente (BROADBENT; STAFFORD; MACDONALD, 2009).

A robótica também é aplicada em regiões atingidas por catástrofes naturais. O tempo para encontrar e auxiliar uma vítima é crucial. Por outro lado, as pessoas envolvidas no trabalho de resgate necessitam fazê-lo com cuidado uma vez que estão em áreas insalubres ou com risco de desmoronamento. Como alternativa é possível aplicar grupos de robôs heterogênios nessas situações, e.g., robôs terrestres junto a robôs aéreos, ou robôs de diferentes tamanhos (LIMA, 2012). Além de todas essas aplicações, robôs são utilizados há um longo tempo dentro da indústria, especialmente no formato de braços robóticos. A partir desses mecanismos todos os processos de uma linha de produção podem ser executados com precisão e agilidade. Isto se deve ao fato de que robôs conseguem trabalhar coordenadamente, nunca se cansam ou mesmo são passíveis de reparo ou substituição. Apesar de todos esses benefícios, a aplicabilidade de braços robóticos é limitada devido a falta de mobilidade (SIEGWART; NOURBAKHSH, 2004; FORD, 2015).

Com o propósito de diversificar o uso dos robôs, pesquisas atuais estão direcionadas ao desenvolvimento de robôs móveis capazes de se locomover em diferentes tipos de terrenos. Para que seja possível aplicar esses robôs em atividades de natureza repetitiva, perigosa ou insalubre os estudos são aplicados a criação de máquinas que interajam com o ambiente de forma autônoma. Essas pesquisas são auxiliadas pelo avanço tecnológico de sensores e atuadores que permitem o robô "sentir" o ambiente e criar representações mesmo de regiões nunca antes exploradas (SIEGWART; NOURBAKHSH, 2004; RO-

MERO et al., 2014).

Para que o robô consiga se movimentar e executar suas atividades de forma autônoma em uma região desconhecida é essencial que ele tenha a capacidade de construir o mapa da região corretamente. Esse mapa é estruturado de acordo com a interpretação do robô das leituras do ambiente obtidas através dos sensores. Para representar a topologia da região explorada é crucial que o robô saiba detectar que está revisitanto um local no mapa em construção para ser capaz de executar correções na estrutura criada. O grande desafio nesse problema é: considerando as diversas modificações que o ambiente pode ter sofrido ao longo da navegação robótica, como saber se o local já foi visitado, ou seja, como identificar uma revisita?

1.1 Motivação

Para que um robô consiga navegar é importante que ele tenha conhecimento do ambiente em que está. Nesse sentido, Makarenko et al. (2002) elencou as tarefas de mapeamento, localização e controle de movimento como essenciais para a movimentação robótica. No **mapeamento** o robô deve criar uma representação do mundo através de sensores que leem constantemente as características do ambiente e do próprio robô. Essas leituras proveem tanto a informação atualizada da pose do robô (posição e orientação), como também dos obstáculos percebidos. Essas informações são então interpretadas pelo robô e passam a integrar o mapa que é constantemente construído.

No caso da tarefa de localização o robô conhece previamente o mapa do local em que está e precisa estimar sua pose dentro dele. Esse problema pode ser subdividido em **localização global** em que o robô possui um estado inicial desconhecido e a **localização local** em que o robô conhece seu estado inicial. Em ambos os casos há a dependência da leitura dos sensores das informações do ambiente, e.g., vindas de lasers e câmeras; e de leituras de estruturas internas, como o caso de bússulas e odômetros. Assim como no problema do mapeamento, na localização é importante conhecer a imprecisão dos sensores e incorporar tal incerteza no cálculo da pose.

De acordo com Makarenko et al. (2002), a integração das tarefas de localizaçao e mapeamento junto ao controle do movimento de atuadores dão origem as diversas áreas de pesquisa na robótica. Na **Localização Ativa** são utilizados os controles do robô para auxiliar na estimação de sua pose no mapa. A tarefa de localização por padrão é passiva, mas unida à leitura de sensores e controles do robô que respondem às condições do am-

biente é possível otimizar a estimação da pose como pode ser visto no trabalho de Fox, Burgard e Thrun (1998). No caso da **Exploração** são utilizados os controles de movimento do robô a fim de permitir que ele conheça tanto quanto possível o ambiente em que está. Com informações mais completas da área explorada é possível gerar um mapa mais fiel ao ambiente real. Esse tipo de atividade é especialmente importante para compreender a estrutura de regiões insalubres, muito pequenas ou mesmo com risco de desastres naturais (THRUN; BURGARD; FOX, 2005).

No campo da robótica autônoma, é crucial construir um mapa que represente o ambiente fielmente. Conforme já mencionado, para construir esse mapa é necessário saber a localização do robô. No entanto, por vezes não se tem acesso a um mapa prévio ou mesmo a um sensor que responda a pose do robô. Para esses casos, o problema de **Localização e Mapeamento simultâneos** (SLAM) permite que o robô calcule sua própria pose enquanto realiza o mapeamento do ambiente. Esse problema é bastante complexo porque precisa lidar com as imprecisões dos sensores aliado a falta de conhecimento da estrutura do ambiente e a pose em que o robô está. Dentro da técnica de SLAM, uma das formas encontradas para compreender a topologia da região é a de garantir que o robô reconheça lugares já visitados. Na robótica, esse problema é conhecido como *place recognition* ou Detecção de Fechamento de *loop*, do inglês *Loop Closure Detection* (LCD) (TSINTOTAS; BAMPIS; GASTERATOS, 2018; LOWRY et al., 2016).

Há um interesse crescente nos métodos de LCD baseados em informação visual, principalmente devido ao baixo custo das câmeras e à riqueza de informações que uma imagem pode fornecer (LOWRY et al., 2016; GARCIA-FIDALGO; ORTIZ, 2015; SZELISKI, 2011). Recentemente, a tarefa de *visual place recognition* em cenários de longo prazo gerou um interesse crescente na comunidade de robótica. Essa tarefa é desafiadora devido ao requisito de que o robô deve reconhecer um local já visitado, tolerando possíveis mudanças visuais. Essas alterações podem ser de diferentes naturezas, e.g., variações de iluminação, transformações causadas por estações distintas, condições climáticas adversas ou alterações estruturais (ARROYO et al., 2015; LOWRY et al., 2016; MILFORD; WYETH, 2012). Ao executar corretamente esse reconhecimento, o método de LCD permite que o SLAM corrija o mapa, representando o ambiente com mais precisão.

Os métodos de LCD mantêm um mapa métrico ou topológico para representar o conhecimento de mundo do robô (LOWRY et al., 2016). O nível de detalhe dos mapas métricos os torna adequados para problemas que exigem precisão, apesar de serem caros para manutenção. Por outro lado, os mapas topológicos são um tipo mais compacto de

representação do ambiente e sua estrutura comumente é baseada em um grafo, na qual seus nós representam locais distintos e as arestas a conexão entre esses locais (GARCIA-FIDALGO; ORTIZ, 2015; GARCIA-FIDALGO; ORTIZ, 2017). O SeqSLAM (MILFORD; WYETH, 2012), considerado o estado da arte no problema de LCD (ARROYO et al., 2015; GARCIA-FIDALGO; ORTIZ, 2015; GARCIA-FIDALGO; ORTIZ, 2017), usa um mapa topológico denso para demonstrar que a busca pelo LCD através de uma sequência de imagens obtém melhores resultados se comparado ao uso de uma imagem de consulta exclusiva. Mostra bons resultados em datasets com diferenças de iluminação e com dinamicidade, mas falha quando há diferença de ponto de vista e tem um custo computacional elevado. Abordagens mais recentes de LCD (GARCIA-FIDALGO; ORTIZ, 2017; DONG et al., 2017) exploram o uso de mapas topológicos esparsos para lidar com o problema, reduzindo o custo computacional. Diferentemente do SeqSLAM, esse tipo de método executa o LCD usando apenas a imagem coletada, exigindo uma análise dispendiosa para fazer a correspondência correta com os locais já visitados.

Embora estes métodos tenham sido testados em datasets que apresentam mudanças ao longo do tempo, fica em aberto o questionamento: é possível criar uma estratégia que realize a detecção do fechamento de *loop* corretamente em imagens que apresentem dinamicidade, ponto de vista e iluminação diferentes? Além disso, é possível que esse método seja capaz de demonstrar precisão superior ao atual estado da arte apesar de utilizar menos informações no processo de detecção de fechamento de *loop*? Esta dissertação tem a proposta de responder tais questionamentos.

1.2 Objetivo

Propomos um método de LCD de três níveis baseado em sequência que usa um mapa topológico hierárquico no qual cada nó contém uma ou mais representações de locais. Cada representação de local consiste em uma sequência de imagens do respectivo local no mundo e um valor representativo para fornecer acesso rápido no mapa topológico. Nossa abordagem realiza o LCD em três níveis. No primeiro nível, o representante do último local visitado que compõe uma sequência de consulta (sequência de nós em processamento) é usado para procurar candidatos em nosso mapa topológico composto por nós já armazenados (locais visitados anteriormente). No segundo nível, a sequência de consulta é comparada a todas as seqüências candidatas criadas a partir dos nós selecionados no primeiro nível, com o objetivo de manter a que é mais semelhante. Além

disso, neste nível, também é verificado se a sequência candidata mais semelhante é temporalmente consistente. No terceiro nível, a sequência de imagens originada na consulta é comparada às imagens pertencentes à sequência resultante do segundo nível. Dessa forma, definimos o fechamento de *loop* de acordo com a similaridade entre as imagens comparadas.

A partir desssa estrutura nosso método tem a proposta de alcançar os seguintes objetivos:

- Criar um mapa que evidencie os pontos distintos de uma trajetória de forma a permitir que a busca por locais já visitados seja tratada sobre um espaço de busca reduzido porém com informações distinguíveis e relevantes;
- Apresentar uma estratégia que utiliza, além das informações visuais recentemente capturadas, as respostas já calculadas pelo método, a fim de definir a certeza de fechamento de *loop* em uma região. Portanto, deve melhorar a precisão dos resultados obtidos ao aproveitar as informações anteriormente providas pelo método sem necessidade de reprocessá-las durante a detecção do fechamento de *loop*;
- Apresentar um método que tenha uma estratégia hierárquica que reduza sucessivamente o espaço de busca por candidatos a fechamento de *loop*. Esta estratégia deve melhorar a precisão dos resultados obtidos apesar de ser executada apenas ao acumular a sequência das últimas informações coletadas e não a cada imagem capturada.

1.3 Organização

Este trabalho está organizado de acordo com a seguinte estrutura:

- Capítulo 2: apresenta os conceitos que foram relevantes para o entendimento e construção do nosso método;
- Capítulo 3: contém a estrutura e funcionamento do método de LCD proposto;
- Capítulo 4: apresenta a configuração e execução dos cenários de testes utilizados para verificar a aplicabilidade do método proposto;
- Capítulo 5: avaliação dos resultados obtidos com nosso método a partir dos cenários de testes apresentados no Capítulo anterior.

2 FUNDAMENTAÇÃO TEÓRICA

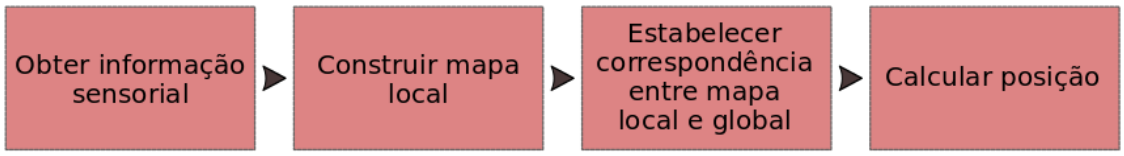
Para melhor compreender o problema de fechamento de *loop* utilizando informações visuais, esta seção introduz inicialmente as bases do problema de SLAM, iniciando pela abordagem do problema de Mapeamento, na Seç. 2.1, seguida da Seç. 2.2 que trata sobre a Localização. Na sequência, é abordado na Seç. 2.3 sobre o problema de SLAM com ênfase na detecção de fechamento de *loop* utilizando informação visual. Finalmente, na Seç. 2.4 são abordados os trabalhos correlatos que inspiraram as ideias desenvolvidas em nosso método.

2.1 Mapeamento

O mapeamento robótico tem por função representar estruturas físicas do ambiente ao gerar modelos computacionais através das informações coletadas pelos sensores do robô. A partir de um mapa confiável, o robô consegue se localizar dentro dele e consequentemente dentro do ambiente real. Dessa forma é possível executar tarefas de navegação e exploração robótica uma vez que o robô consegue desviar de obstáculos, realizar o planejamento de caminhos e traçar os pontos a serem explorados. Através dessas habilidades o robô passa a ter a capacidade de realizar atividades de forma autônoma (THRUN, 2003; ROMERO et al., 2014).

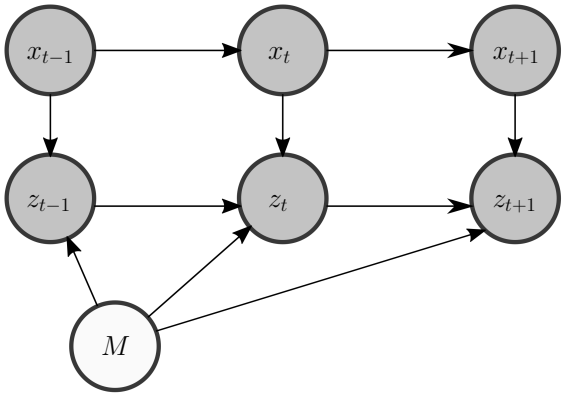
No problema de mapeamento é considerado que a pose inicial do robô é previamente conhecida. Então, a partir de seus sensores proprioceptivos e externoceptivos é possível criar um modelo computacional da estrutura do ambiente. Os sensores proprioceptivos medem valores internos do robô e podem ser exemplificados pela leitura de odometria, variação angular através da bússola ou medições de velocidade dos motores do robô. Já os sensores externoceptivos captam informações externas do ambiente a exemplo de medidas de distância, imagens, alterações na intensidade da luz e do som (SIEGWART; NOURBAKHSH, 2004; THRUN, 2003; ROMERO et al., 2014). A partir das medidas dos sensores, de acordo com Borenstein et al. (1996), a construção do mapa é realizada de forma iterativa conforme os processos demonstrados na imagem 2.1.

Figura 2.1: Ciclo de criação do mapa e atualização da posição do robô. Imagem adaptada de (BORENSTEIN et al., 1996)



No ciclo de criação do mapa, já se sabe a pose inicial do robô. A partir dela o mapa é contruído através da captura das informações dos sensores contidos neste robô. Essas informações são então transformadas no modelo computacional através da construção de um mapa local. Baseada nas características do mapa local é então verificado se há correspondência com o mapa global previamente contruído ou se o mapa local deve ser considerado uma nova área do mapa global. Como último processo do ciclo de mapeamento, caso o robô não possua sensores internos que forneçam sua pose, ela pode ser calculada baseada nas informações atualizadas do mapa. Portanto, de forma genérica, é possível representar o problema de mapeamento como demonstrado na Fig. 2.2.

Figura 2.2: Modelo de criação do mapa M. As informações sombreadas são conhecidas (nesse caso as posições do robô x e medidas dos sensores z). Imagem adaptada de Thrun, Burgard e Fox (2005)



Ao realizar a construção do mapa é importante considerar alguns pontos que são elencados por Siegwart e Nourbakhsh (2004), Thrun (2003) e Thrun, Burgard e Fox (2005):

- Alcance do sensor: analisar se a capacidade de adquirir informações do sensor disponível é compatível com o tamanho do ambiente;
- Considerar o ruído nas leituras: a tarefa de construção do mapa seria trivial se os sensores não possuíssem erros associados e em geral esse não é o caso. Por isso, ao obter a informação sensorial é necessário filtrá-la e modelar o erro do sensor se conhecido para então construir o mapa local;

- Erros na estrutura do mapa: na tarefa de mapeamento é importante detectar quando o robô está passando pelo mesmo ponto visitado anteriormente, i.e., quando há fechamento de *loops*. Quando as medidas para criação do mapa são baseadas em odometria, por exemplo, se o robô vai e volta pelo mesmo caminho é trivial realizar as correções nas medições. No entanto, quando ele retorna por um caminho diferente do inicial os erros de odometria se acumulam a ponto de ocasionar distorções na geração do mapa;
- Escolha do tipo de mapa a ser construído: o nível de detalhamento do mapa deve ser compatível com as necessidades da tarefa atribuída ao robô;
- Granularidade do mapa: as informações necessárias para construção do mapa devem estar conciliadas ao nível de detalhamento que o sensor consegue extrair do ambiente;
- Custo de construção e manutenção: a granularidade das características contidas no mapa é diretamente relacionada ao custo computacional para adquiri-las, compará-las as informações já existentes e armazená-las.

Dentre os itens citados a escolha do tipo do mapa tem grande impacto na execução da tarefa de mapeamento. Os mapas podem ser divididos em duas categorias principais: mapas métricos e mapas topológicos (THRUN, 2003).

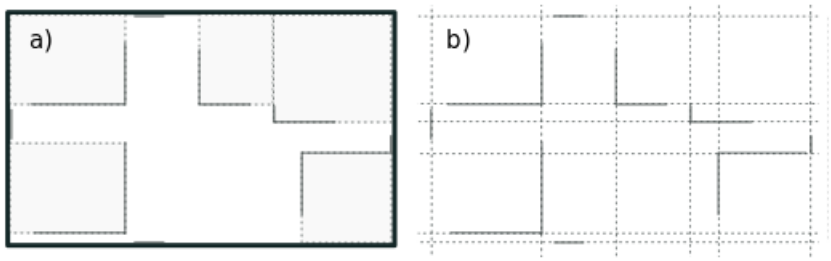
2.1.1 Mapas Métricos

Os **mapas métricos** representam o ambiente com grau variado de precisão. O nível de exatidão é respectivo a necessidade da tarefa a qual o mapa será aplicado. Com base na diversidade de aplicações para mapeamento robótico, foram desenvolvidos diferentes modelos de mapas métricos ao longo do tempo que são baseados em duas estratégias: representação contínua ou decomposição do espaço mapeado (SIEGWART; NOURBAKHSH, 2004; THRUN, 2003).

A partir da estratégia de representação contínua são gerados mapas semelhantes ao ambiente real e portanto com alto grau de exatidão. Nesse tipo de mapa a posição das características do espaço são associadas com precisão ao mapa. Por conta do nível de detalhamento e consequente custo computacional é comum que esses mapas sejam criados apenas para ambientes em duas dimensões. Pelo mesmo motivo, em geral se dá prioridade para que a descrição dos objetos em cena seja simplificada. Para tal é

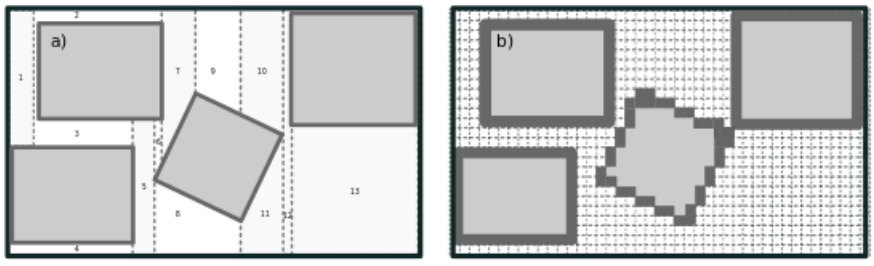
possível descrever apenas aspectos geométricos dos objetos em cena, representando-os como polígonos ou como um conjunto de linhas infinitas no ambiente, como pode ser visto na Fig. 2.3.

Figura 2.3: Um exemplo de mapa contínuo pode ser visto em (a) em que os objetos e obstáculos são representados através de polígonos nas posições e tamanho dos respectivos no mundo real. Em uma representação menos detalhada (b), é possível criar o mesmo mapa de (a) mas apenas com linhas infinitas que são posicionadas de acordo com as leituras do sensor do robô das extremidades dos objetos e obstáculos do ambiente real.



Na estratégia de decomposição o mapa é representado de forma mais abstrata do que no formato contínuo. Por esse motivo uma das desvantagens dessa abordagem é a perda de precisão do aspecto real do ambiente. Apesar disso, esse tipo de representação é útil a tarefas que exigem menor precisão, de forma que são mantidas apenas as características que são importantes para execução da tarefa robótica. Como exemplos desse tipo de mapa há o de decomposição exata de células e o de decomposição fixa. Um exemplo de mapa por decomposição exata de células pode ser visto na Fig. 2.4.a em que há células de espaço livre que são delimitadas a partir das extremidades dos polígonos que representam os obstáculos no ambiente. Esse tipo de mapa é utilizado para quando não é necessário estimar com precisão a posição do robô, apenas os espaços livres que conectam as regiões do mapa. Caso seja necessário estimar a posição é possível utilizar, por exemplo, o mapa de decomposição fixa. Nesse caso é estipulado um nível de discretização do ambiente em células de tamanho fixo como demonstrado na Fig. 2.4.b. A desvantagem desse mapa é que dependendo do tamanho das células, espaços que conectavam áreas livres deixam de existir, a exemplo das células 10, 12 e 13 do mapa na Fig. 2.4.a.

Figura 2.4: Exemplos de mapas métricos que utilizam a estratégia de decomposição: (a) decomposição exata e (b) decomposição fixa.



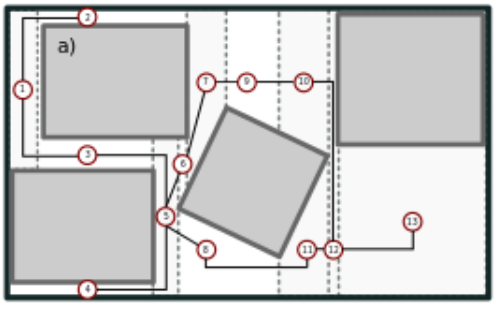
Outras representações de mapas de decomposição fixa tendem a resolver o problema de redução de espaço livre demonstrada na Fig. 2.4.b. Uma alternativa é criar células com o tamanho variado dependendo do detalhamento da área do mapa. Uma outra possibilidade é representar o mesmo mapa a partir de uma grid de ocupação. Cada célula dessa grid ganha o *status* de livre, ocupada ou não calculada. Essa categoria é definida mediante a, e.g., uma estratégia de votação que depende da incidência de leituras do sensor naquele ponto do mapa. Em geral, o tamanho das células da grid é fixo e reduzido, sendo portanto em grande quantidade. Adicionalmente, cada célula armazena um valor específico de ocupação. Dessa forma, esse tipo de mapa é custoso computacionalmente uma vez que tende a crescer conforme novas áreas são processadas.

2.1.2 Mapas Topológicos

Os **mapas topológicos** representam o ambiente de forma mais abstrata que os mapas métricos. No lugar de descreverem características geométricas, em geral exibem a conectividade entre regiões do ambiente (THRUN, 2003). Formalmente, um mapa topológico é representado através de um grafo composto de nodos e suas conexões, as arestas. A ligação entre dois nodos indica que o robô pode chegar de um ponto ao outro sem precisar passar por qualquer outro ponto do mapa. Esses nodos podem ser criados de forma a caracterizar informações em comum de uma região do ambiente real. Essas informações não precisam ser necessariamente geométricas, mas podem, por exemplo, ser extraídas de características de imagens capturadas da região mapeada (SIEGWART; NOURBAKHSH, 2004).

Na Fig. 2.5 é demonstrado a criação de um mapa topológico para representar de forma mais abstrata o mapa métrico da Fig. 2.4.a. No lugar de cada célula de espaço livre há um nodo, entre os nodos há arestas que correspondem as conexões entre as regiões. De forma simplificada, o mapa passa a ser representado com apenas treze nodos.

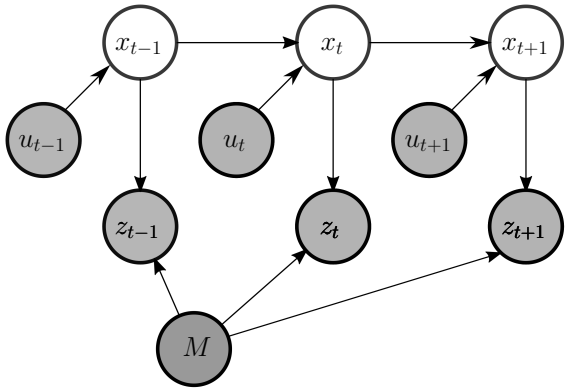
Figura 2.5: Mapa topológico



2.2 Localização

Ao contrário do problema de mapeamento em que a posição do robô é conhecida e a partir das medições adquiridas pelos sensores é construído o mapa, no problema de **localização** é necessário descobrir a pose do robô em um mapa previamente disponibilizado, como pode ser visto na Fig. 2.6 (THRUN; BURGARD; FOX, 2005).

Figura 2.6: Modelo genérico do problema de estimação da posição x de um robô ao longo do tempo. As informações sombreadas são conhecidas (nesse caso o mapa M, as medidas dos sensores z e as informações de controle u). Imagem adaptada de Thrun, Burgard e Fox (2005).



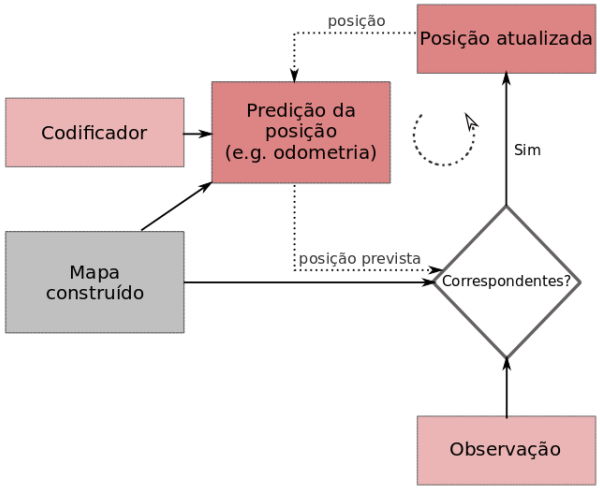
Existem duas formas de calcular a pose do robô no mapa: **localização local** e **localização global**. Na local, é considerado que a pose do robô tenha sido calculada no instante anterior, e assumindo que o erro de leitura dos sensores é pequeno, é possível tratar o problema de localização como o rastreamento da posição precedente. No caso da global, a posição anterior é completamente desconhecida. Portanto, métodos desse tipo precisam lidar com os erros dos sensores sem ter informação anterior que permita limitar a busca por uma área específica no mapa. Dentro do problema de localização global há o problema de rapto em que o robô é retirado da posição em que está no mapa e colocado em outro ponto arbitrário. Esta variação é similar ao problema de localização global clássica, porém mais desafiador. Enquanto na localização global o robô sabe que não tem conhecimento da própria pose, no problema de rapto o robô acredita saber onde está quando na verdade não está mais. Conseguir estimar a pose em qualquer uma dessas condições é essencial para tarefas que requerem autonomia robótica (THRUN; BURGARD; FOX, 2005).

De acordo com Siegwart e Nourbakhsh (2004), se fosse possível equipar um robô com um sensor de GPS (*Global Positioning System*) suficientemente preciso a tarefa de estimar a localização no mapa seria trivial. No entanto, esse tipo de sensor funciona

de forma mais acurada em ambientes abertos. Se estiver entre prédios, por exemplo, a precisão da estimação cai drasticamente uma vez que o sinal do sensor é refletido nas construções interferindo no cálculo da pose. Esse problema em específico pode ser suavizado a partir de técnicas que integram as leituras de vários sensores simultaneamente. No entanto, em ambientes fechados, o GPS ainda não é capaz de estimar a localização com precisão (SIEGWART; NOURBAKHSH, 2004; MAIER; KLEINER, 2010).

Como alternativa ao sensor de GPS, é possível utilizar as informações extraídas dos demais sensores exteroceptivos e proprioceptivos do robô. A partir das leituras, são detectadas características do ambiente e valores internos que permitem identificar a região próxima ao robô correspondente à uma previamente armazenada no mapa. Para reconhecer uma região no mapa é possível utilizar informações de *beacons* que emitem sua posição, *landmarks* distinguíveis, valores de odometria calculados a partir dos *encoders* dos motores do robô, ou ainda, informações visuais capturadas do ambiente. Ao definir a região do mapa respectiva as leituras dos sensores, é então determinada a pose do robô no mapa, como pode ser visto na Fig. 2.7. (BORENSTEIN et al., 1996).

Figura 2.7: Integração de leituras de sensores para estimação contínua da posição do robô.



Assim como na tarefa de mapeamento, um dos desafios para se estimar a localização está no fato de que os dados são capturados por sensores geralmente ruidosos. Além disso, por vezes o ambiente é dinâmico, i.e, há alterações dos componentes do mapa ao longo do tempo. Uma forma de melhorar a estimação da posição é integrar as leituras dos sensores como pode ser visto na Fig. 2.7, em que os valores observados do ambiente, juntamente com as informações extraídas de odometria possibilitam a constante atualização da posição do robô (THRUN; BURGARD; FOX, 2005).

2.3 SLAM

Nem sempre há um mapa disponível ou atualizado do ambiente, assim como a posição inicial do robô por vezes é desconhecida. A ausência dessas informações inviabiliza a execução de diversas tarefas a exemplo da realização de planejamento e exploração robóticas. Como alternativa, é possível construir o mapa ao mesmo tempo em que se realiza a estimação da pose do robô nesse mapa. Essa tarefa corresponde a um dos problemas recentemente mais abordados em robótica móvel denominado localização e mapeamento simultâneos, do inglês SLAM (Simultaneous Localization and Mapping). O SLAM não se trata de um problema trivial porque para se criar um mapa é necessário que o robô saiba com precisão sua localização, ao mesmo tempo que para conseguir estimar a localização com acurácia, é necessário dispor de um mapa que represente fielmente o ambiente real. Por esse motivo, o SLAM é conhecido como o problema do "ovo e galinha" (THRUN; BURGARD; FOX, 2005; DURRANT-WHYTE; BAILEY, 2006).

O SLAM é mais complexo que o problema de localização porque não possui um mapa preciso para calcular a pose com acurácia. Além disso, a estimação do posicionamento é realizada sobre um mapa que é construído durante a passagem do robô pelo ambiente. Sendo assim, o SLAM também se difere do problema de mapeamento, porque em nenhum momento há a informação da localização precisa do robô para auxiliar a construção do mapa. Todas essas incertezas são aliadas ao desafio de manter o mapa atualizado apesar da dinamicidade encontrada no ambiente. Dessa forma, o uso do SLAM é fundamentamental para considerar um robô verdadeiramente autônomo, uma vez que provê a capacidade de criar mapas e estimar a localização em ambientes ainda não explorados (BRESSON et al., 2017; THRUN; BURGARD; FOX, 2005; SIEGWART; NOURBAKHSH, 2004).

De acordo com Durrant-Whyte e Bailey (2006), ao considerarmos que durante a movimentação do robô no ambiente, ele continuamente captura medições através dos seus sensores dos *landmarks* que estão dispostos em posições desconhecidas, o SLAM pode ser definido a partir das seguintes variáveis no instante t:

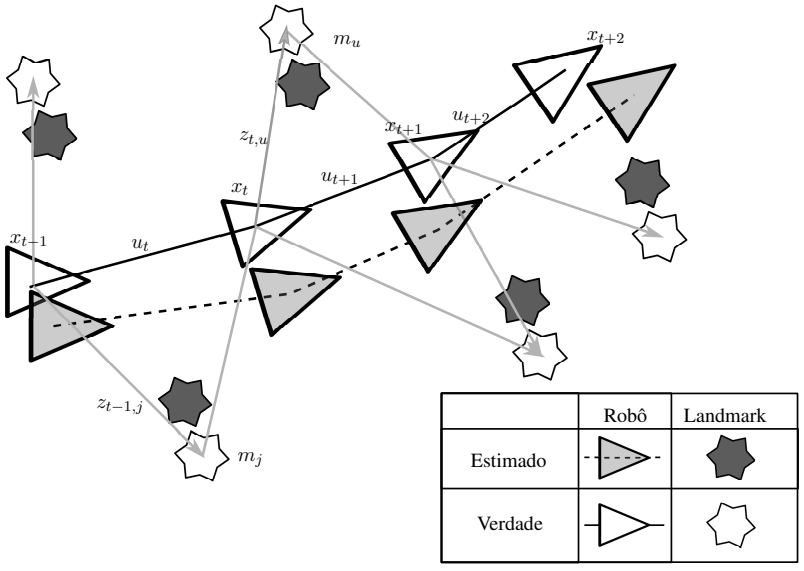
- Valores conhecidos:
    - u_t: valor de controle aplicado no tempo $t-1$ que direciona o robô ao estado x_t no tempo t;
    - $z_{t,j}$: uma observação coletada pelo sensor do robô no tempo t de um *landmark*

de índice j.

- Valores desejados:
  - x_t: a pose do robô no tempo t;
  - m_j: a localização de um *landmark* m de índice j. Os *landmarks* são representados por um conjunto M que corresponde ao mapa criado.

De acordo com Grisetti et al. (2010) e Bresson et al. (2017), para que seja possível estimar a trajetória e construir o mapa do ambiente é necessário considerar a existência de ruído inerente as medidas extraídas dos sensores. Por esse motivo, a formulação do SLAM é descrita probabilisticamente. Para tanto, o percurso da trajetória percorrida é representado por um conjunto $X = \{x_1, x_2, \cdots, x_t\}$, em que x_k indica o estado do robô no tempo k. O estado pode ser representado pela pose em 2D (posição e orientação), em 6D, aceleração, velocidade, entre outros tipos de informação. Enquanto o robô vai se conduzindo ele adquire valores de movimentação expressos pelo conjunto $U = \{u_1, u_2, \cdots, u_t\}$, em que u_k pode representar o valor de odometria ou de qualquer outro sensor que indique o deslocamento. Além disso, há o conjunto $Z = \{z_1, z_2, \cdots, z_t\}$ que contém as percepções do ambiente a exemplo de leituras de sonar ou imagens. O conjunto $M = \{m_1, m_2, \cdots, m_i\}$ representa o mapa do ambiente, em que cada m_j indica um *landmark* detectado pelos sensores do robô. A dinâmica dessas variáveis dentro do SLAM pode ser vista na Fig. 2.8.

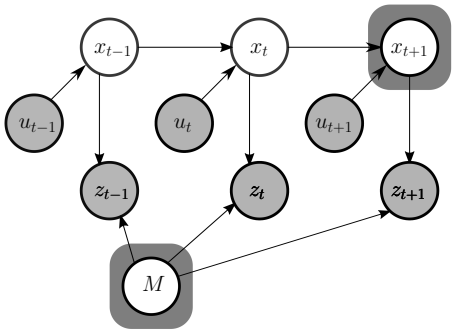
Figura 2.8: Funcionamento do SLAM. Imagem adaptada de (DURRANT-WHYTE; BAILEY, 2006)



De acordo com Thrun, Burgard e Fox (2005), Grisetti et al. (2010) e Bresson et al. (2017), dado o conjunto de variáveis que compõe o problema de SLAM, ele pode ser

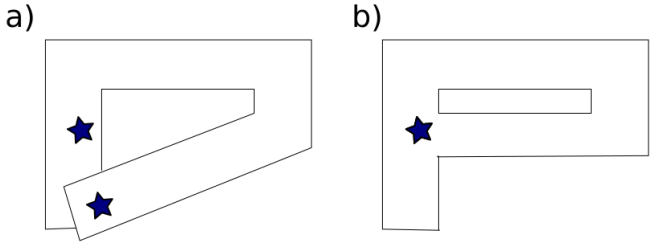
representado por uma Rede Bayesiana Dinâmica e pode ser solucionado a partir de duas abordagens igualmente importantes para a robótica móvel: *full* SLAM, e SLAM *on-line*. Na primeira, o objetivo do método é estimar a trajetória do robô assim como construir o mapa por completo. Para tanto são considerados os conjuntos de observações e controle do robô durante todo o percurso realizado. Já na *on-line*, as variáveis são estimadas no tempo t, como pode ser visto da Fig. 2.9. Em geral essa abordagem é composta por algoritmos que são incrementais e portanto discartam valores e medidas anteriores aos atuais, uma vez que estes já foram processados.

Figura 2.9: Modelo gráfico do SLAM *on-line*. Os valores sombreados são os observados pelo robô: no caso os de deslocamento compõe o conjunto U e o conjunto Z contém as observações extraídas de leituras do ambiente. A cada instante t são estimadas a posição x do robô e atualizado o mapa M (THRUN; BURGARD; FOX, 2005)



Considerando a existência de ruído no sensor que estima o deslocamento do robô entre duas poses, comumente calculado pelo valor de odometria, a geração do mapa pode ser distorcida ao longo do tempo. Como consequência o mapa aparenta ser um "corredor infinito", como pode ser visto na Fig. 2.10.a. A fim de expressar a correta topologia do ambiente, o sistema de SLAM realiza a detecção de pontos do mapa que já foram visitados. A partir da detecção da revisita, também conhecida como detecção de fechamento de *loop*, do inglês LCD (*Loop Closure Detection*), é possível corrigir o mapa distorcido retificando os erros de odometria, como pode ser visto na imagem Fig. 2.10.b (SÜNDERHAUF; PROTZEL, 2012; CADENA et al., 2016).

Figura 2.10: Mapa com efeito de "corredor infinito" por conta dos erros de estimação de deslocamento do robô em (a). A topologia do mapa é corrigida após a detecção do fechamento de *loop* em (b).



Para que toda essa estrutura seja mantida a arquitetura de um sistema de SLAM é subdividida em dois módulos: *front-end* e *back-end*. O *front-end* é a parte do SLAM que tem acesso aos sensores exteroceptivos e proprioceptivos do robô. Nesse módulo é gerada uma estrutura (podendo ser um grafo) que contém todas as informações obtidas pelos sensores (leituras do ambiente e deslocamentos do robô), assim como as estimações de posição do robô ao longo do tempo e o mapa gerado. Já o módulo de *back-end* é responsável por receber a estrutura gerada pelo *front-end*, interpretá-la e otimizá-la de forma que ela respeite as informações de restrições de deslocamento estimadas entre poses, assim como os fechamentos de *loop* detectados. Como exemplos de métodos que podem ser utilizados como *back-end* há o g^{2}o (KÜMMERLE et al., 2011) e o GTSAM (DELLAERT, 2012). Como o *back-end* depende das informações geradas pelo *front-end* para construir o mapa, é fundamental que não sejam detectados fechamentos de *loops* incorretamente, já que eles acarretariam na geração de um mapa inconsistente com a realidade (SUNDERHAUF; PROTZEL, 2011; SÜNDERHAUF; PROTZEL, 2012; KHAN; WOLLHERR, 2015; CADENA et al., 2016).

2.3.1 SLAM Topológico Visual

A partir da execução de um sistema de SLAM é gerado uma estrutura, geralmente representada por um grafo, em que cada nodo corresponde à uma posição alcançada pelo robô durante a trajetória percorrida. Além disso, são armazenados os valores de deslocamento entre poses e o posicionamento dos *landmarks* que compõe o mapa do ambiente. Baseados nessas informações, grande parte dos trabalhos na área de SLAM são desenvolvidos considerando a criação de mapas métricos em que as poses e os *landmarks* representam estruturas do ambiente real (GARCIA-FIDALGO; ORTIZ, 2015).

Além dos métodos de SLAM baseados em mapas métricos, há os que utilizam mapas topológicos. Assim como o primeiro tipo, também dependem da detecção do fechamento de *loop* pelo *front-end* para que o *back-end* consiga gerar o mapa corretamente. Nesse sentido, de acordo com Cadena et al. (2016), Garcia-Fidalgo e Ortiz (2015) e Lowry et al. (2016), sistemas de SLAM em que o *front-end* utiliza sensores ultra-sônicos ou a *laser* com robustez a falhas e suficientemente precisos já correspondem a um problema considerado largamente tratado. Por outro lado, a detecção de fechamento de *loop* em mapas baseados em imagens do ambiente ainda possui vários desafios em aberto e portanto é nosso objeto de estudo nesse trabalho.

2.3.2 Detecção de fechamento de *loop* utilizando informação visual

Recentemente houve um crescimento no número de trabalhos de detecção de fechamento de *loop* que utilizam informação visual como entrada de dados. Esses trabalhos são inspirados por aplicações em robôs domésticos, exploração de minas, mapeamento de áreas no espaço e, mais recentemente, no desenvolvimento de carros autônomos que se beneficiam de informações visuais. Esse tipo de dado passou a ser mais utilizado porque imagens são ricas em informações do ambiente se comparado a um *laser*, por exemplo. Além disso, câmeras são um sensor de baixo custo e passivos, i.e., são menos sujeitas a interferências nas leituras ocasionadas pela estrutura do ambiente ou pela presença de sinais emitidos por outros sensores (BORENSTEIN et al., 1996; SIEGWART; NOURBAKHSH, 2004; SZELISKI, 2011; GARCIA-FIDALGO; ORTIZ, 2017).

De acordo com Lowry et al. (2016), um sistema de LCD que utiliza informações visuais ainda é um problema em aberto. As diversas abordagens existentes tentam responder a seguinte pergunta: dadas as imagens de um local é possível saber se ele já foi visitado? Inicialmente, os métodos propostos consideravam que o ambiente era estático, isto é, não sofria modificações ao longo do dia ou alterações por conta das mudanças de estações. Essa suposição é verdadeira quanto se trata de um mapa pequeno, em um ambiente controlado. No entanto, quando é necessário executar o método durante um longo tempo, a ocorrência de alterações no ambiente é inevitável.

Por esse motivo, a tarefa de reconhecer um local já visitado em um ambiente dinâmico não é trivial, uma vez que sua aparência pode ter sofrido modificações. Dessa forma, para saber se as imagens capturadas pertencem à um local já visitado é importante considerar mudanças que ocorrem por conta da alteração natural de iluminação ou ponto de vista diferente, como pode ser visto nas Figs. 2.11 e 2.12.

Figura 2.11: Diferença de iluminação [1].

(a) Fig. 119 dia - GPW (b) Fig. 119 noite - GPW

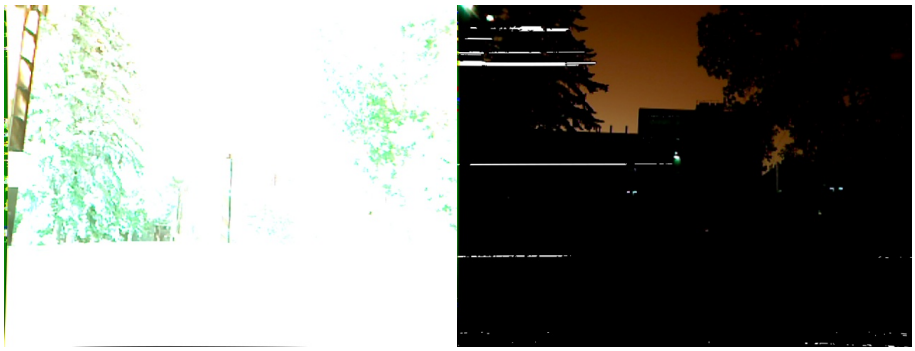
Figura 2.12: Diferença de ponto de vista [2].

(a) Fig. 210 dia - UofA (b) Fig. 210 tarde - UofA

Além disso, a qualidade da imagem pode ser afetada no processo de captura, por exemplo, por excesso ou ausência de iluminação, como pode ser visto na Fig. 2.13.

Figura 2.13: Falhas de captura no *dataset* UofA[2].

(a) Fig. 000 dia (falha) (b) Fig. 000 tarde (falha)



(c) Fig. 084 dia (d) Fig. 084 tarde (falha)

Ainda, há a movimentação de pessoas e objetos ou mesmo a oclusão parcial de uma cena por conta da alteração do ponto de vista, conforme demonstrado na Fig. 2.14.

Figura 2.14: Ambientes dinâmicos com oclusão parcial da cena (imagens 103), presença de pedestres (imagens 056) e mudança na posição de objetos (imagens 272)[2].

(a) Fig. 103 dia - GPW (b) Fig. 103 noite - GPW

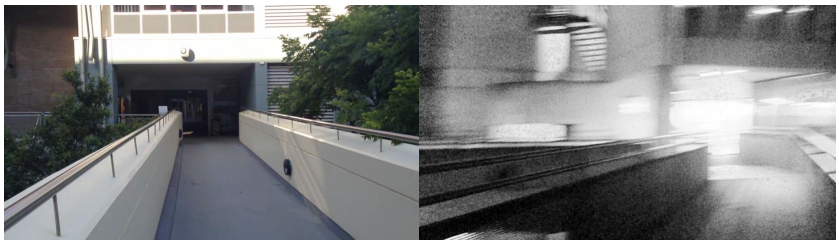
(c) Fig. 056 dia - GPW (d) Fig. 056 noite - GPW

(e) Fig. 272 dia - UofA (f) Fig. 272 tarde - UofA

Por fim, um dos maiores desafios a serem tratados no módulo de *front-end* do SLAM é definir quando as imagens capturadas, mesmo sendo similares às imagens de locais visitados anteriormente, na verdade não são correspondentes, como pode ser visto na imagem 2.15. Detectar a distinção entre locais é importante porque o fechamento de *loop* incorreto acarreta a geração de um mapa inconsistente pelo módulo de *back-end* do SLAM (KHAN; WOLLHERR, 2015; CADENA et al., 2016).

Figura 2.15: Imagens que são similares em relação as informações estruturais apresentadas mas não são fechamentos de *loop*[2].

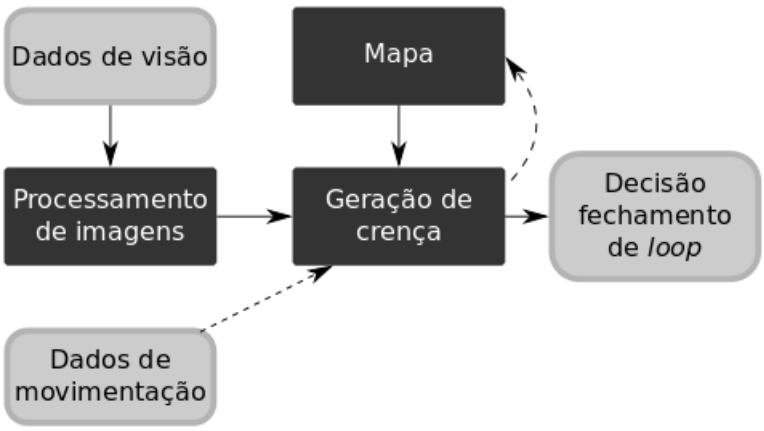
(a) Fig. 44 dia - GPW (b) Fig. 193 noite - GPW



(c) Fig. 099 dia - UofA (d) Fig. 132 tarde - GPW

A execução da detecção do fechamento de *loop* utilizando informações visuais é composta por três módulos principais: processamento da imagem que é capturada pelo robô, a construção do mapa a partir das imagens pré-processadas e a geração de crença que define o grau de certeza em que as informações recebidas em comparação com o mapa, correspondem a um fechamento de *loop*, como pode ser visto na Fig. 2.16.

Figura 2.16: Modelo genérico de um sistema de LCD que utiliza informações visuais. Imagem adaptada de Lowry et al. (2016).



[2]As imagens de mesmo índice representam fechamentos de *loop* em duas trajetórias percorridas em momentos diferentes do dia. Mais detalhes sobre os *datasets* na Tab. 4.2.

2.3.2.1 Processamento de imagens

De acordo com Garcia-Fidalgo e Ortiz (2015), para que seja possível criar um mapa e se localizar nele a partir de informações visuais é necessário primeiramente descrever as imagens de forma satisfatória. Pois é a partir dessa informação que se torna possível realizar a comparação com descritores de imagens já armazenadas no mapa. A escolha pelo método de descrição afeta diretamente a qualidade do mapa e consequentemente a execução do LCD. O processo de descrição pode ser dividido em três categorias principais: descrição de características locais, métodos baseados em *Bag of Words* (BoW) e métodos de descrição global.

Descritores Locais - Uma imagem pode ser descrita a partir de porções dela que contenham características que a tornem distinguíveis das demais. De acordo com Szeliski (2011), é possível diferenciar imagens através de seus *pontos de interesse*, ou *features*, como por exemplo, cantos de estruturas, regiões ou BLOBs (*Binary Large OBject*). Para que seja possível representar as *features*, os descritores locais necessitam de duas fases: a detecção e a descrição. Na fase de detecção, porções da imagem que não possuem textura em geral são ignoradas, e são escolhidas áreas que apresentam grande diferença de contraste. Então, é executada a fase de descrição, em que são selecionadas informações da vizinhança dos *pontos de interesse* para gerar o descritor. Para que imagens correspondentes possam ser encontradas é preferível que o descritor gerado seja invariável as transformações ocasionadas pela alteração do ponto de vista, além de possuir características de repetibilidade, precisão e distintividade (GARCIA-FIDALGO; ORTIZ, 2015; LOWRY et al., 2016; GARCIA-FIDALGO; ORTIZ, 2018).

O processo de detecção e descrição são independentes e por isso é possível combinar técnicas de acordo com as características de interesse nas imagens. Como exemplo de detectores de cantos há os já consolidados Harris (HARRIS; STEPHENS et al., 1988) e o Fast (ROSTEN; DRUMMOND, 2006) cujas ideias foram utilizadas em parte no desenvolvolvimento do método ORB (RUBLEE et al., 2011), que pode ser utilizado como detector e descritor combinados. Além do ORB, há o SIFT (*Scale-Invariant Feature Transform*) (LOWE, 2004) e o SURF (*Speeded Up Robust Features*) (BAY; TUYTELAARS; GOOL, 2006) que também possuem a função dupla de detector e descritor, mas são aplicados para informações do tipo BLOB. No caso do descritor SIFT as *features* são invariantes a escala e rotação assim como a distorções, ruído e a variações de iluminação. As *features* são calculadas a partir da técnica de DoG (*Difference of Gaussians*) em que uma versão da imagem original submetida ao filtro Gaussiano é subtraída de uma versão da mesma

imagem menos "borrada" de forma sucessiva. Através do DoG é possível evidenciar os contornos existentes na imagem a partir dos quais é extraído um conjunto de *features* de interesse. Por conta de suas características de robustez, esse descritor é largamente utilizado no problema de LCD. Caso também do descritor SURF, em que a proposta é de realizar a detecção e descrição das *features* em um tempo menor que o SIFT mantendo as características de invariância a escala e rotação (LOWRY et al., 2016).

Bag of Words - Cada imagem pode conter centenas de *features*, portanto para realizar o casamento entre duas imagens representadas por descritores locais o custo computacional pode ser elevado. Além disso, é possível identificar um padrão entre *features*, o que permite que estas sejam quantizadas e categorizadas. Essa é a ideia do *framework Bag of Words* (BoW) (SIVIC; ZISSERMAN, 2003), que foi originalmente desenvolvido para busca de imagens e atualmente é uma técnica amplamente utilizada no problema de LCD (CUMMINS; NEWMAN, 2008; ANGELI et al., 2008; GARCIA-FIDALGO; ORTIZ, 2014; KHAN; WOLLHERR, 2015). Cada imagem é representada por um vetor que indica quais palavras do BoW existem na imagem. Essas palavras são criadas mediante a um treinamento com imagens similares às do ambiente em que o *framework* será executado. O treinamento quantiza *features* comuns em palavras visuais que compõe o vocabulário visual do BoW.

Embora esse método consiga representar uma imagem de forma mais compacta e seja robusto a variação de ponto de vista, ele tem a desvantagem de exigir uma etapa de treinamento prévia. Além disso, o desempenho é relativo a similaridade das imagens do treinamento com o ambiente em que o método será executado posteriormente. Por esse motivo, o funcionamento do BoW é dependente do ambiente uma vez que o treinamento necessita ser refeito no caso de aplicar o método em uma área diferente (LOWRY et al., 2016). De acordo com Garcia-Fidalgo e Ortiz (2017) é possível tornar o *framework* adaptável ao ambiente já que foram desenvolvidas técnicas que permitem a criação do vocabulário durante a execução do método. No entanto, ao verificar continuamente a criação de novas palavras para o BoW, há um aumento de custo computacional. Outra desvantagem, é que ao representar a imagem como um vetor é perdido a informação de posicionamento espacial das *features*, como consequência, esse método sofre de *perceptual aliasing*: em que imagens diferentes são representadas da mesma forma. É possível contornar esse problema ao custo de adicionar etapas adicionais ao método, por exemplo, realizar verificação geométrica entre as *features* das imagens selecionadas, ou mesmo adicionar informação geométrica na geração do vocabulário visual (LIU; ZHANG, 2012;

KEJRIWAL; KUMAR; SHIBATA, 2016).

Descritores Globais - A partir dos descritores globais é possível generalizar a descrição de uma imagem inteira através de um vetor que contém informações de cor, formato ou textura. Geralmente são rápidos de computar e por terem uma estrutura compacta simplificam o processo para verificação de correspondência com os demais descritores. Ao contrário dos descritores locais, demonstram bons resultados quando são expostos a mudanças de luz e a ambientes dinâmicos, isto é, com pequenas oclusões ou movimentação de pessoas e objetos. Em contrapartida, geralmente sofrem com a diferença de ponto de vista (CHATZICHRISTOFIS; ZAGORIS; ARAMPATZIS, 2011; LOWRY et al., 2016; GARCIA-FIDALGO; ORTIZ, 2018).

Assim como os descritores locais, os descritores globais procuram registrar as informações que tornam a imagem discernível a partir de uma versão simplificada dos principais contornos e texturas. Como exemplos há os descritores Gist (OLIVA; TORRALBA, 2001) e HOG (*Histogram of Oriented Gradients*) (DALAL; TRIGGS, 2005) que representam as informações estruturais e disposição espacial dos itens de uma imagem. O descritor Gist, como o próprio nome indica, registra a "essência" de uma imagem, i.e., o arranjo espacial de objetos, regiões e BLOBs. Para tal, o descritor é criado a partir da resposta de filtros direcionados em diferentes orientações e escalas que são aplicados sobre a imagem. O Gist demonstrou ser acurado para reconhecimento de paisagens naturais, no entanto sua performance é reduzida em imagens de ambientes internos (WU; REHG, 2011). Já o descritor HOG, é gerado a partir das informações de orientações dos gradientes (direções das bordas). Para tanto, a imagem é dividida em células de tamanho fixo e para cada uma dessas regiões é calculado um histograma que contabiliza os gradientes existentes nos pixels da célula. Essas células são então agrupadas em blocos com sobreposição. Em cada bloco é aplicado uma normalização por contraste que tem o objetivo de melhorar a invariância à iluminação do descritor. Por fim, o valor dos blocos sobrepostos são combinados e é gerado o descritor HOG. Esse descritor é amplamente utilizado em sistemas de recuperação de imagem, além disso, por conta de sua capacidade descritiva, também é aplicado ao problema de LCD tanto na forma original, quanto no caso do PHOG (BOSCH; ZISSERMAN; MUNOZ, 2007), como descritor combinado de regiões da imagem (SAAVEDRA, 2015; SIAM; ZHANG, 2017; GARCIA-FIDALGO; ORTIZ, 2017).

Mais recentemente foi proposto o descritor global RST-SHelo (SAAVEDRA, 2015) que tem a proposta de sofrer pouca influência do ângulo de captura das imagens, diver-

gindo da limitação dos demais descritores globais. Originalmente, o RST-SHelo foi criado para o domínio de recuperação de esboços, i.e., é aplicado para descrever um esboço e possibilitar a recuperação de imagens regulares similares a ele. Esse descritor é baseado no HOG que também pode ser aplicado para descrição de esboços. Assim como o HOG, o RST-SHelo descreve uma imagem baseado nos valores de gradientes detectados. No RST-SHelo a imagem é dividida em um número fixo de células de tamanho variável. Para cada célula é calculado qual orientação é predominante, diferentemente do HOG que possui um histograma de orientações por célula. Então, para registrar não apenas a orientação mas também a distribuição espacial dessas orientações na imagem, as células são agrupadas em blocos sem sobreposição. Para cada bloco é calculado um histograma de orientações das células que o compõe. Por fim, esses histogramas são contatenados e normalizados. Como o SRT-SHelo foi criado originalmente para o contexto de recuperação de esboços, ao calcular uma única orientação por célula da imagem, ele se mostra mais adequado do que o HOG à esparsidade dos contornos existentes. Ao contrário do HOG, o RST-SHelo é invariável à escala e translação, o que o torna mais robusto à mudanças de ponto de vista. Assim como o HOG, o RST-SHelo é adequado tanto para aplicações em esboços como em imagens regulares. A superioridade dos resultados em relação ao HOG foi demonstrada no artigo que propõe o SRT-SHelo (SAAVEDRA, 2015), assim como nos testes realizados por Bhattacharjee et al. (2018).

2.3.2.2 Mapeamento topológico utilizando imagens

Existem diferentes estratégias para criação de um mapa topológico baseado em imagens para que possa ser utilizado por um sistema de LCD. A forma mais simplificada consiste no armazenamento das imagens capturadas sem qualquer tipo de indexação ou estrutura. Para esse tipo de mapa, a partir da última imagem capturada é selecionada a imagem correspondente mais similar no mapa. Baseado em um *threshold* de similaridade é definido se as imagens correspondem a um fechamento de *loop*. O problema desse tipo de mapa para LCD é que mesmo que as imagens sejam armazenadas em uma estrutura de acesso rápido, o volume de dados tende a crescer rapidamente dependendo do tamanho do trajeto e frequência de captura. Além disso, a cada nova imagem, todas as já armazenadas tem a mesma chance de serem candidatas a fechamento de *loop*, i.e., não é considerada a estimação do local em que o robô estava na iteração anterior, o único valor avaliado é a similaridade entre imagens. (LOWRY et al., 2016).

De acordo com Cadena et al. (2016) a estrutura de um sistema de LCD compre-

ende um bloco que realiza a associação dos dados recebidos a curto prazo e um segundo bloco para dados a longo prazo. O primeiro bloco analisa a relação existente entre dados coletados consecutivamente pelos sensores, por exemplo, verifica que duas imagens capturadas consecutivamente são similares e portanto podem representar um mesmo local no ambiente. Já na associação de dados a longo prazo, ou módulo de fechamento de *loop*, é verificado a similaridade das últimas informações coletadas do ambiente com os locais visitados anteriormente. Para que esse processo seja eficiente, é interessante que o mapa criado permita verificar, por exemplo, o local em que o robô estava na iteração anterior a fim de reduzir o espaço de busca na iteração atual. Nesse sentido, é possível representar o mapa de um sistema de LCD a partir de uma estrutura topológica. Considerando que o sistema tem como entrada informações visuais, o mapa é composto por nós que representam os locais, enquanto que as arestas caracterizam os caminhos entre esses locais físicos. Dentro desse contexto, um local topológico não é necessariamente igual a um local de acordo com o entendimento humano de porção ou região de um ambiente. Em um sistema de LCD, a divisão entre dois locais representa uma discretização do ambiente criada a partir, por exemplo, da diferença perceptual entre duas imagens capturadas consecutivamente (WU; REHG, 2011; LOWRY et al., 2016).

Ainda que problemas de SLAM geralmente sejam aplicados a mapas métricos é possível executá-lo em um mapa topológico. Assim como descrito no capítulo 2.1, os mapas topológicos possuem um custo de criação e manutenção reduzido se comparado a mapas métricos. Por não disporem de informações precisas, devem ser aplicados a missões que não requisitem a estimação da localização do robô, embora seja possível adicionar dados de deslocamento que auxiliem nessa tarefa (ARROYO et al., 2015; LOWRY et al., 2016).

Em relação ao tamanho do mapa, há estratégias que discretizam o ambiente representando um local, ou seja, um nodo no mapa, para cada imagem que é capturada, gerando mapas topológicos densos. Uma alternativa para reduzir o tamanho do mapa e permitir que o SLAM execute por mais tempo, dadas as restrições de processamento e espaço, é discretizar o ambiente gerando um nodo para um conjunto de imagens sequenciais que sejam similares entre si (GARCIA-FIDALGO; ORTIZ, 2017).

2.3.2.3 Geração de crença

A principal função de um sistema de LCD é a capacidade de determinar se um local do mapa já foi visitado anteriormente. Dessa forma, deve verificar se as observa-

ções mais recentes são respectivas a um local já armazenado no mapa. Uma das premissas é a de que se dois locais são similares visualmente eles devem representar o mesmo ponto na trajetória. No entanto, a confiança dessa correspondência varia em condições de mudanças de iluminação e ambientes dinâmicos. Portanto, não basta indicar se o local corresponde a um ponto já visitado no mapa, mas sim o percentual de confiança dessa revisita (LOWRY et al., 2016).

O grau de certeza de uma revisita pode ser calculado a partir de diferentes informações extraídas do sistema de LCD. A exemplo do trabalho de Milford e Wyeth (2012), o percentual de confiança é baseado no quão uma correspondência é significativa. Para tanto, dado as últimas leituras do ambiente é calculado uma lista dos candidatos a fechamento de *loop* ordenados por similaridade visual. Considerando que o primeiro candidato da lista seja o mais provável de representar o fechamento de *loop*, então o percentual de confiança é calculado a partir da razão do valor de similaridade do primeiro candidato em relação ao valor da similaridade do primeiro candidado fora de uma janela estipulada. No caso de trabalhos que utilizam BoW, por exemplo, é possível calcular a confiança baseado no valor de TF-IDF (Term Frequency - Inverse Document Frequency). Nessa técnica a confiança é calculada a partir de duas métricas: a primeira verifica a frequência da ocorrência de uma *feature* visual na imagem, denominada *Term Frequency* e a segunda calcula o quão comum essa *feature* aparece nas imagens, no caso *Inverse Document Frequency*. O valor final é o produto entre as duas métricas (KHAN; WOLLHERR, 2015; LOWRY et al., 2016).

Se tratando do SLAM *on-line* geralmente os módulos de *front-end* e *back-end* se comunicam a cada imagem capturada que corresponda a um fechamento de *loop*. Dessa forma, a confiança associada à um fechamento de *loop* é baseada puramente na leitura atual dos sensores do robô. No entanto, a análise da captura isolada pode não ser suficiente para representar um local já visitado. Esse tipo de situação pode ocorrer em ambientes com poucas *features* ou por regiões com *features* que se repetem em vários pontos do trajeto. Nesse sentido, é interessante acumular as leituras capturadas, para então verificar o fechamento de *loop* uma vez que mais informações tendem a tornar a busca mais distintiva. Uma alternativa é unir as imagens capturadas no passado junto com a mais recente pra verificar se esta corresponde a um fechamento de *loop*. Em contrapartida, essa estratégia exige o reprocessamento das informações que são passadas para responder ao estado atual. Outra alternativa é acumular novas leituras para só então analisar o fechamento de *loop*. Apesar de possivelmente detectar o *loop* com atraso, essa estratégia auxilia em

uma busca acurada ao mesmo tempo que reduz a quantidade de processamento do sistema de SLAM (NEWMAN; HO, 2005; BAILEY; DURRANT-WHYTE, 2006; KONOLIGE; AGRAWAL, 2008; MILFORD; WYETH, 2012).

2.4 Trabalhos relacionados

Trabalhos recentes demonstram um aumento do interesse pelo desenvolvimento de abordagens de LCD que realizam a busca da imagem capturada pelo robô, que denominamos imagem de busca, em um mapa topológico baseado em informações visuais. Esses trabalhos estão especialmente focados em métodos que permitem realizar a deteção do fechamento de *loop* consideranto mudanças visuais ocorridas ao longo do tempo. Tais abordagens podem ser agrupadas em duas categorias principais: a primeira corresponde a métodos tradicionais de LCD que utilizam isoladamente a imagem de busca em um mapa que contém as imagens dos locais já visitados. A segunda categoria, representada por trabalhos mais recentes, é composta de abortagens baseadas no conceito do uso de sequências, i.e., o método considera uma sequência de imagens de busca para verificar se um local já foi visitado no mapa, ao invés de utilizar isoladamente a última imagem para tal tarefa (LOWRY et al., 2016; ARROYO et al., 2015).

O FAB-MAP (CUMMINS; NEWMAN, 2008) é um trabalho largamente referenciado para o tratamento do problema de LCD (LOWRY et al., 2016; GARCIA-FIDALGO; ORTIZ, 2017; ARROYO et al., 2015). De forma geral, o mundo é modelado como um conjunto de locais discretos. Cada imagem capturada corresponde a um local que é descrito por uma distribuição de palavras visuais. Para tanto, sempre que uma nova imagem é capturada ela é convertida em uma representação de BoW. Para definir se há um fechamento de *loop*, é verificado a probabilidade de que a observação atual tenha vindo dos locais já armazenados ou se corresponde a um novo local do mapa.

Uma das principais inovações desse método é em relação a forma de aplicar o BoW. Este *framework* era originalmente usado em recuperação de imagens, mas o FAB-MAP passou a aplicá-lo na análise de co-ocorrência de palavras visuais em uma imagem. Para tanto, foi utilizado a árvore Chow and Liu (CHOW; LIU, 1968) construída a partir de um grafo de informação mútua. Cada nodo corresponde a uma palavra visual e o peso das arestas entre os nodos contém a quantidade de vezes em que as respectivas palavras visuais co-ocorrem. Para que essa árvore seja corretamente criada, é necessário que as verificações de co-ocorrência sejam geradas a partir de um conjunto de treinamento

extenso de imagens similares aos locais que serão analisados posteriormente. Dessas imagens de treinamento são extraídas e clusterizadas as features SURF que formam o vocabulário do BoW.

Além das informações contidas na árvore foi necessário implementar um fator de suavização para inibir casos de *perceptual aliasing*. Dessa forma, a probabilidade de uma imagem vir do mesmo lugar é a combinação devolvida pela árvore Chow and Liu junto com o fator de suavização que define se as palavras detectadas correspondem a características repetitivas.

Inspirados pelo FAB-MAP, trabalhos como o de Garcia-Fidalgo e Ortiz (2017) e Dong et al. (2017) também usam BoW, no entanto diferem do FAB-MAP por aplicarem o *framework* em um mapa topológico hierárquico de dois níveis. Ambos os métodos utilizam a imagem de busca para selecionar os locais do mapa que são mais similares a ela. Cada local contém uma sequência de imagens que são semelhantes umas as outras e um valor que as representa. Esse valor é usado no primeiro nível do mapa para verificar os lugares que são mais similares a imagem de busca. Apesar da abordagem de ambos os métodos no primeiro nível serem parecidas, elas possuem algumas diferenças em relação ao processamento. Por exemplo, no primeiro nível do trabalho de Garcia-Fidalgo e Ortiz (2017), um descritor global é extraído da imagem de busca que então é utilizado para buscar pelos representantes de locais mais similares. Cada um desses locais contém um índice com palavras que representam as *features* locais das imagens que compõe o local. Essas palavras são definidas pelo *framework* BoW em uma estratégia de treinamento realizada durante a execução do método. No segundo nível, é verificada qual imagem que pertence aos locais selecionados é mais provável de fechar *loop* com a imagem de busca. Para tanto, é calculado o *score* de similaridade do descritor global da imagem de busca com o representante de cada local selecionado (nesse caso também um descritor global). Esse score é combinado com a similaridade entre as *features* locais da imagem de busca em relação as *features* locais de cada imagem que compõe cada local selecionado no primeiro nível. Ao combinar o score entre descritores globais e o índice de similaridade entre *features* locais o método seleciona a imagem que provavelmente representa um fechamento de *loop*. Para definir se há o fechamento de *loop* com a imagem selecionada, como último passo, o método realiza uma verificação geométrica entre as *features* locais da imagem de busca e as da imagem selecionada como a mais similar. Caso sejam suficientemente parecidas, isto é, a quantidade de *features* locais comuns esteja acima de determinado limiar, é definido o fechamento de *loop*.

Já no caso do trabalho de Dong et al. (2017), ainda no primeiro nível são extraídas *features* locais da imagem de busca que são utilizadas pelo *framework* BoW para definir as palavras que representam esta imagem. Esse conjunto de palavras é utilizado para selecionar um local já armazenado no mapa que seja o mais similar. Já no segundo nível, é realizada uma validação geométrica entre as *features* locais da imagem de busca e as *features* locais das imagens pertencentes ao local selecionado no primeiro nível a fim de verificar a ocorrência de fechamento de *loop*.

O SeqSLAM (MILFORD; WYETH, 2012) também é uma referência como método de LCD (MILFORD; WYETH, 2012; GARCIA-FIDALGO; ORTIZ, 2017; ARROYO et al., 2015). Sua contribuição é a habilidade de detectar fechamento de *loop* para a última imagem tomada utilizando a sequência das imagens capturadas recentemente. Além disso, esse trabalho demonstrou melhores resultados que o FAB-MAP em cenários em que há mudanças visuais perceptíveis. Por outro lado, o SeqSLAM apresenta um custo computacional elevado devido a sua rotina de verificação de correspondências. Isto porque o método calcula a similaridade entre a sequência das últimas imagens capturadas em relação as sequências de todas as imagens armazenadas, sendo que esse custo é proporcional ao tamanho da sequência e ao número de imagens do mapa.

Além de influenciar no custo computacional, o tamanho da sequência representa o fator de maior impacto na capacidade do SeqSLAM em detectar *loop*. Ao escolher um valor elevado para esse parâmetro há uma melhora no número correto de correspondências, no entanto sequências grandes não favorecem os resultados do método quando aplicadas sobre *datasets* que possuem muitas curvas (ARROYO et al., 2015). Outro fator que restringe a aplibilidade do SeqSLAM é o fato de que a verificação da similaridade entre a sequência das imagens de busca com as sequências de imagens armazenadas é realizada a partir da subtração entre as imagens correspondentes. Para tanto, com as sequências alinhadas, é calculada a soma das diferenças absolutas entre as imagens. A soma dessas diferenças deve ser o menor possível quanto mais similares as sequências forem entre si. Dessa forma, é necessário que as imagens estejam sob um mesmo ponto de vista. De acordo com Sünderhauf, Neubert e Protzel (2013), uma pequena alteração no ponto de vista é suficiente para degradar expressivamente os resultados do método.

Outros trabalhos também foram propostos com base no conceito do uso de sequência e são comparáveis aos resultados apresentados pelo SeqSLAM. O Fast-SeqSLAM (SIAM; ZHANG, 2017) se difere do SeqSLAM porque no lugar de procurar um local no mapa usando uma sequência das imagens mais recentes, usa uma sequência de descritores

globais extraídos das últimas imagens capturadas. Assim como o SeqSLAM, quando a sequência de consulta é significativamente semelhante a uma sequência de imagens armazenadas no mapa, o *loop* da última imagem capturada é definido. Este método possui um custo computacional reduzido se comparado ao SeqSLAM porque processa a detecção de *loop* apenas em um conjunto reduzido de sequências de descritores armazenados, que ele acredita ter maior probabilidade de conter o *loop* pesquisado. Embora o custo de processamento seja reduzido, os resultados em relação a quantidade de detecção de *loops* em um *dataset* não são expressivamente melhoradas em relação ao SeqSLAM.

Além do Fast-SeqSLAM, há o trabalho de Arroyo et al. (2015), em que é apresentada uma abordagem que busca por locais visitados considerando uma seqüência concatenada de descritores binários globais de imagens. Diferentemente do Fast-SeqSLAM, a sequência de consulta é comparada a sequências de descritores binários globais concatenados que são armazenadas como nodos de uma árvore para rápido acesso. De acordo com a similaridade encontrada entre as sequências concatenadas, o LCD é definido para a última imagem capturada. Este trabalho consegue detectar mais *loops* corretamente que o SeqSLAM para os *datasets* abordados no trabalho. Por outro lado, necessita de sequências de imagens extensas para conseguir tais resultados.

Em uma análise geral, alguns trabalhos, como os de Cummins e Newman (2008), Dong et al. (2017) e Garcia-Fidalgo e Ortiz (2017), usam descritores de *features* locais, que são representados por meio do *framework* BoW, para descrever as imagens. Conforme já mencionado, os descritores locais são conhecidos por não serem robustos a mudanças de iluminação e requerem um método de detecção. Além disso, às vezes o método de verificação de correspondências entre descritores é ineficiente, a exemplo dos trabalhos de Garcia-Fidalgo e Ortiz (2017) e Dong et al. (2017) que necessitam de validação geométrica entre as *features* das imagens de busca em relação as *features* das imagens candidatas a fechamento de *loop*. Quanto ao uso do *framework* BoW, que é aplicado para representar os descritores das *features* locais, há a exigência de uma etapa de treinamento prévia com imagens similares ao ambiente em que o método será executado posteriormente. Como alternativa, no trabalho de Garcia-Fidalgo e Ortiz (2017), o vocabulário visual foi criado durante a execução do método de LCD, no entanto incrementando o custo computacional da abordagem. No caso de Arroyo et al. (2015) e Siam e Zhang (2017), são utilizados descritores globais para representar as imagens e demonstram bons resultados quando executados sob mudanças de luz e ambientes dinâmicos.

Todos os trabalhos apresentados até o momento contém vantagens e desvantagens

que foram consideradas durante o desenvolvimento do nosso método. Portanto, propomos uma abordagem que une os conceitos de execução do método de LCD através de uma estratégia hierárquica, similar aos trabalhos de Garcia-Fidalgo e Ortiz (2017) e Dong et al. (2017), juntamente ao conceito do uso de sequências inspirado nos trabalhos do SeqSLAM, Fast-SeqSLAM e Arroyo et al. (2015). Ao unir tais abordagens, contornando suas limitações, propomos resolver o problema de detecção de fechamento de *loop* que seja robusto à informações visuais que possam vir a sofrer alterações ao longo do tempo.

3 ESTRATÉGIA PARA DETECÇÃO DE FECHAMENTO DE *LOOP* UTILIZANDO SEQUÊNCIAS EM UMA ESTRUTURA HIERÁRQUICA DE TRÊS NÍVEIS

Esse capítulo tem o propósito de demonstrar a estratégia proposta para o método de LCD. Primeiramente, na Seção 3.1, apresentamos as técnicas utilizadas para pré-processar as imagens que são obtidas durante a execução de uma travessia. Primeiramente, na Seção 3.1.1, demonstramos o método adotado para reduzir as imperfeições do processo de captura das imagens e salientar as características robustas à dinamicidade do ambiente. Em seguida, na Seção 3.1.2, apresentamos o método de criação dos locais que são particionados de acordo com a similaridade das imagens pré-processadas capturadas sequencialmente.

O entendimento do processo de criação de locais é fundamental para compreender o funcionamento do método proposto de LCD, já que cada iteração recebe como entrada uma sequência de locais criados que são processados em uma estratégia hierárquica de três níveis, conforme abordado na Seção 3.2. No primeiro nível, a sequência dos locais visitados mais recentemente é usada como consulta para procurar sequências candidatas em nosso mapa topológico composto por locais visitados anteriormente, conforme Seção 3.2.1. No Segundo Nível, abordado na Seção 3.2.2, o método seleciona a sequência mais semelhante à consulta entre todas as sequências candidatas, que é temporalmente consistente com a resposta anterior do nosso método de LCD. No terceiro nível, abordado na Seção 3.2.3, combinamos as sequências de imagens pertencentes à sequência de consulta e à sequência candidata selecionada no segundo nível. A partir dessa combinação, nossa estratégia define o % de certeza de fechamento de loop entre as imagens processadas.

3.1 Pré-processamento

Esta seção descreve as etapas necessárias para configurar nosso sistema para detecção de fechamentos de *loop*. Para tanto é explicado como as imagens capturadas são pré-processadas e agrupadas em locais. Além disto, é demonstrado como os locais são unidos em uma sequência de consulta e como esta sequência é incluída no mapa. Todas essas etapas da fase de pré-processamento são demonstradas no Alg. 1 e, durante essa seção, é utilizada a notação apresentada na Tabela 3.1.

Tabela 3.1: Notação do método.

Notação	Definição
g_t	Descritor global RST-SHelo no intante t
G	Conjunto de descritores RST-SHelo
P_l	O último local criado
$\mathbf{P}$	Conjunto de locais
Q	Sequência de locais de consulta de tamanho ρ
F	Sequência de locais candidata de tamanho ρ
$\mathbf{F}$	Conjunto de sequências de locais candidatas
F^*	Sequência de local candidata $F \in \mathbf{F}$ mais similar a Q
M	Mapa topológico composto de V e E
V	Conjunto de vértices v
E	Conjunto de bordas que conectam dois vértices em V
$\bar{Q}$	Conjunto de descritores de $\forall\ P \in Q$ gerados pela função $\mathcal{G}(Q)$
$\bar{F}$	Conjunto de descritores de $\forall\ P \in F^*$ gerados pela função $\mathcal{G}(F^*)$
$\bar{F}^*$	Conjunto de descritores $\subseteq \bar{F}$ mais similares a $\bar{Q}$
$\mathcal{R}(P_j)$	Função que calcula o descritor g_i representante de P_j
$\mathcal{P}(v)$	Função que retorna o conjunto de locais do vértice v
$\mathcal{P}(V)$	Função que retorna o conjunto de locais do mapa, i.e., locais de $\forall\ v \in V$
$\mathcal{S}(g_u, g_j)$	Função que calcula a similaridade entre os descritores g_u e g_j
$\mathcal{I}(P_j)$	Função que calcula o índice j
$\mathcal{D}(g_u, g_j)$	Função que calcula a distância entre os descritores g_u e g_j (L1-*norm*)

3.1.1 Pré-processamento de imagens

Os desafios mais comuns em tarefas de reconhecimento visual de locais estão relacionados a mudanças de iluminação e ambientes dinâmicos. Uma alternativa robusta para superá-los é considerar padrões do ambiente que sejam mais estáveis ao longo do tempo. Como exemplo há as informações estruturais presentes em construções, estradas e objetos fixados no ambiente como lixeiras e placas de sinalização. Em geral esses objetos ou cenas são caracterizados e reconhecidos especialmente pelas bordas que apresentam (SZELISKI, 2011; SHAPLEY; TOLHURST, 1973).

A informação de borda mostrou ser útil ao problema de LCD por ser robusta a

alterações ao longo do tempo e a modificações naturais de iluminação (LEE et al., 2013; LEE et al., 2014; NUSKE; ROBERTS; WYETH, 2009; SZELISKI, 2011; SHAPLEY; TOLHURST, 1973). Apesar disso, esse tipo de informação pode ser afetado pelo ruído e iluminação incorreta resultantes de problemas gerados, por exemplo, por uma câmera mal configurada, uso inadequado de lentes e tempo de exposição incorreto durante a captura da imagem.

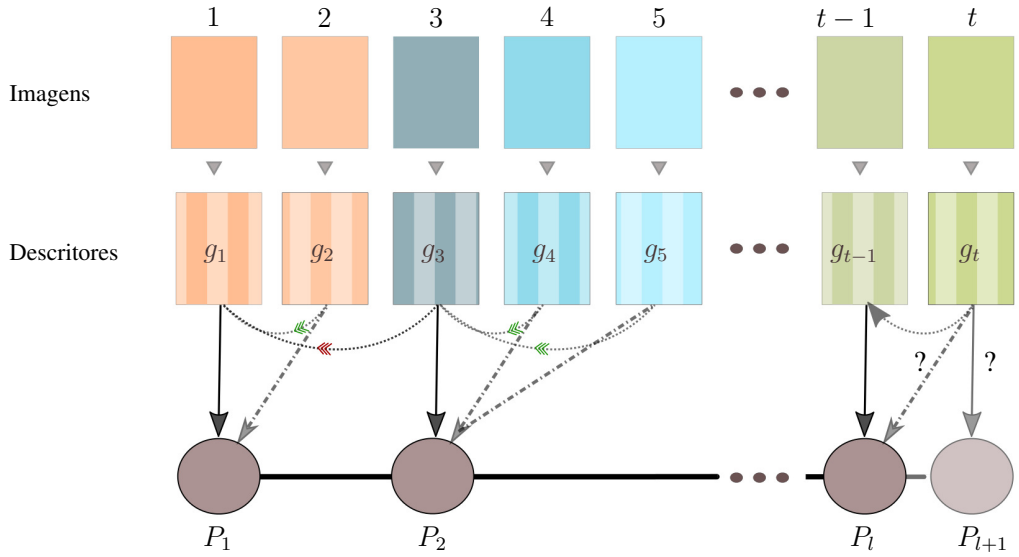
No caso específico da presença de ruído, é possível que sejam geradas bordas na imagem onde elas não deveriam existir, comprometendo a qualidade dos descritores que usam esse tipo de informação. Para superar esse problema, uma solução adequada é aplicar um filtro de suavização antes de detectar as bordas (GONZALEZ; WOODS; MASTERS, 2009; SZELISKI, 2011). Em nosso sistema, aplicamos um Filtro Gaussiano em todas as imagens, devido ao fato de que ele é usado na maioria dos algoritmos de detecção de bordas (SZELISKI, 2011).

Além disso, nos casos em que há excesso ou pouca iluminação na imagem, as bordas não ficam tão evidentes, uma vez que o contorno não está bem definido. Portanto, além do filtro gaussiano, nosso método também aplica equalização de histograma em cada imagem para destacar suas bordas, melhorando a invariância a iluminação (GONZALEZ; WOODS; MASTERS, 2009; DALAL; TRIGGS, 2005; SZELISKI, 2011).

3.1.2 Criação do local

Com base na ideia de que imagens sequenciais estão associadas ao mesmo local no ambiente (GARCIA-FIDALGO; ORTIZ, 2017; DONG et al., 2017; TSINTOTAS; BAMPIS; GASTERATOS, 2018), agrupamo-las de acordo com a sua similaridade. Uma vez que uma imagem é pré-processada no instante t, calculamos seu descritor global RST-SHelo g_t e o incluímos em um conjunto G que contém todos os descritores das imagens. Em seguida, o método verifica se g_t deve ser incluído em um grupo de descritores semelhantes gerados anteriormente ou se g_t deve pertencer a um novo grupo de descritores. A este grupo de descritores é dado o nome de *local* que é definido como um subconjunto de G, onde todos os descritores são semelhantes e foram extraídos de imagens sequenciais.

Figura 3.1: Quando uma nova imagem é capturada, ela é pré-processada e seu descritor g_t é extraído. Em seguida, o método verifica se esse descritor é semelhante ao último local criado, P_l, ou se deve pertencer a um novo local, i.e., P_{l+1}.



Em mais detalhes, vamos supor que já identificamos um conjunto de locais $\mathbf{P} = \{P_1, P_2, \ldots, P_l\}$ e um novo descritor g_t é gerado como mostrado na Fig. 3.1. O descritor g_t será incluído no ultimo local P_l se

$$\mathcal{S}(\mathcal{R}(P_l), g_t) \geq \nu, \tag{3.1}$$

onde $0 \leq \mathcal{S}(.,.) \leq 1$ é uma função que retorna 0 quando há alta dissimilaridade e 1 para alta similaridade; $\mathcal{R}(\cdot)$ é uma função que retorna o representante de P_l, em nosso método expresso pelo primeiro descritor adicionado a P_l, e ν é o limiar de similaridade. Caso o último descritor g_t não seja similar ao representante do último local P_l, um novo local P_{l+1} é criado e g_t é associado a ele. Esse processo de criação de um local é demonstrado no Alg. 1, onde em primeiro lugar nosso método carrega o descritor g_t nas linhas 5 até a 7. Então, a similaridade entre este descritor e o representante do último local criado P_l é verificada nas linhas 8 e 9. Se não for semelhante, g_t dá origem a um novo local P_{l+1} nas linhas 16 a 19.

3.1.3 Construção do mapa

Em nossa abordagem, descrevemos o ambiente como um mapa topológico. Ele é representado por um grafo não direcionado $M = (V, E)$, onde $V = \{v_1, v_2, \cdots, v_m\}$ é um conjunto de vértices e E é o conjunto de arestas entre pares de vértices. Cada v está

relacionado a um ou mais locais de $\mathbf{P}$, i.e., $|\mathbf{P}| \geq |V|$, em que $\mathcal{P}(v)$ é o conjunto de locais associados a v. A função $\mathcal{P}(\cdot)$ também pode ser aplicada ao conjunto V da seguinte forma

$$\mathcal{P}(V) = \bigcup_{v \in V} \mathcal{P}(v), \tag{3.2}$$

portanto retornando todos os locais existentes no mapa.

Os locais são agrupados em uma sequência de consulta Q com ρ locais. Uma nova sequência de consulta Q é criada com base no último índice de local criado, de forma que o índice l de P_l seja múltiplo de ρ, ou seja, $l = k \cdot \rho \mid k \in \mathbb{N}$. Assim, a sequência de consulta Q terá os seguintes elementos

$$Q = \{q_1, q_2, \cdots, q_\rho\}, \tag{3.3}$$

onde q_j representa $P_{l-\rho+j}$.

De acordo com o Alg. 1, a sequência de consulta Q é processada pelo nosso sistema para detectar um *loop*, como pode ser visto nas linhas 11 a 15. Quando a iteração do LCD terminar, todos os $P \in Q$ e suas respectivas arestas, isto é, $(P_{l-\rho+j}, P_{l-\rho+j-1})$ onde $1 < j \leq |\mathcal{P}(V)|$ são incluídas no nosso mapa topológico. Então, Q é esvaziada e está pronta para receber os ρ próximos novos locais criados. Assim, nosso mapa é construído iterativamente, no qual apenas as sequências de consultas processadas são incluídas.

Algorithm 1 Construção do mapa

procedure CONSTRUIRMAPA
  $l \leftarrow 1$
  Criar $P_l \leftarrow \varnothing$
  while câmera está capturando **do**
    Ler e pré-processar a imagem I_t
    Computar o descritor g_t de I_t ▷ RST-SHelo
    $G \leftarrow G \cup g_t$
    if P_l está $\varnothing$ **or** $\mathcal{S}\big(\mathcal{R}(P_l), g_t\big) \geq \nu$ **then**
      $P_l \leftarrow P_l \cup g_t$
    else
      if $\left\lceil \frac{l}{\rho} \right\rceil = \left\lfloor \frac{l}{\rho} \right\rfloor$ **then**
        Criar Q de acordo com a Eq. 3.3
        $\mathbf{result} \leftarrow$ LCD_PrimeiroNivel(Q, M)
        Atualizar M de acordo com Q
      end if
      $l = l + 1$
      Criar novo $P_l \leftarrow \varnothing$
      $P_l \leftarrow P_l \cup g_t$
      $\mathbf{P} \leftarrow \mathbf{P} \cup P_l$
    end if
  end while
end procedure

Composição do último local criado

Construção do mapa

Criação de um local

3.2 DETECÇÃO DE FECHAMENTO DE *LOOP*

O sistema de LCD proposto possui uma estratégia que vai de um espaço de busca denso a um reduzido, no qual o processo de detecção de *loop* é dividido em três níveis. O primeiro nível começa com uma pesquisa aproximada da sequência de consulta Q no mapa M. Esta pesquisa retorna um conjunto de sequências de locais candidatas que são semelhantes aos locais de Q. O segundo nível recebe a saída do primeiro nível e, em seguida, ele escolhe a seqüência de locais candidata que é mais provável que contenha um fechamento de *loop*. Quando o segundo nível decide sobre a melhor sequência can-

didata, o terceiro nível realiza uma verificação mais precisa através de um processo de comparação dos descritores das imagens dos locais de Q com os da sequência candidata escolhida. Nós detalhamos cada nível abaixo e usamos a notação apresentada na Tab. 3.1.

3.2.1 Primeiro nível: procurando por sequências de locais candidatas no mapa M

No primeiro nível, para cada nova sequência de consulta Q, nosso sistema de LCD encontra sequências de locais candidatas associadas aos vértices de V que são similares a Q através de uma comparação simplificada. Esta comparação considera apenas o descritor da imagem representante do último local inserido em Q, portanto, garantindo que o primeiro nível realize uma comparação simples para encontrar sequências promissoras, já que é processado sobre um espaço de busca reduzido, em vez de um processo mais refinado que é mais dispendioso em termos computacionais. Os passos executados durante este nível estão apresentados resumidamente no Alg. 2 e posteriormente descritos em mais detalhes.

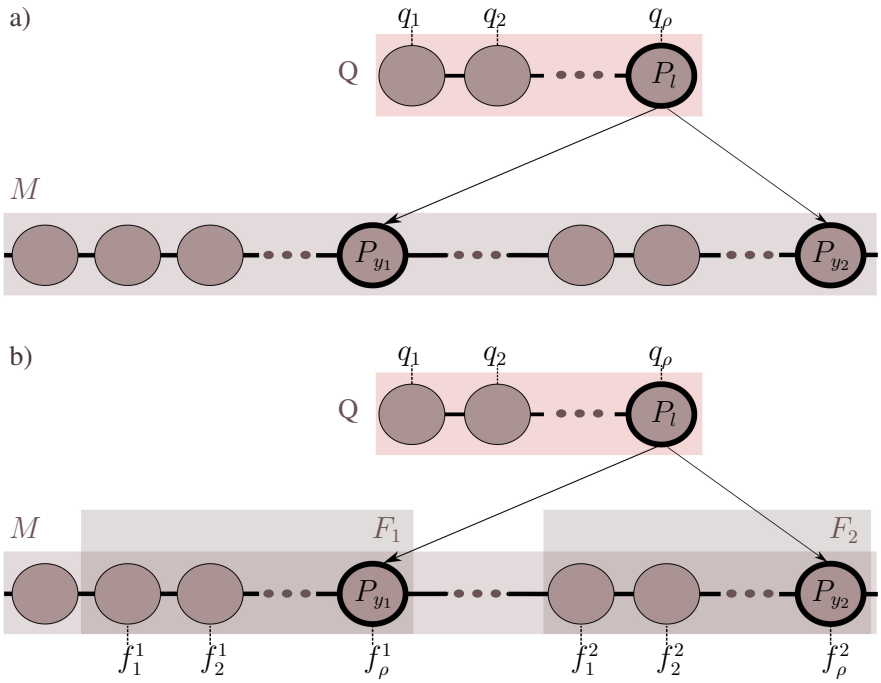
Algorithm 2 Primeiro nível do método de LCD

1: **procedure** LCD_PRIMEIRONIVEL(Q, M)
2: Encontrar os ψ locais mais similares ao último local visitado $q_\rho \in Q$ em $\mathcal{P}(V)$
3: Computar o conjunto de sequências de locais candidatas **F** baseado nos locais selecionados da linha 2
4: LCD_SegundoNivel(**F**, M)
5: **end procedure**

Para que seja possível encontrar as sequências canditadas que são mais similares a sequência de consulta Q realizamos a pesquisa a partir do local atual em que o robô se encontra. Dado que Q contém uma sequência de ρ locais, usamos o representante do último local visitado $\mathcal{R}(q_\rho)$, onde $q_\rho \in Q$ de acordo com a Eq. 3.3, e compararamos ele com os representantes de todos os locais já visitados até então, i.e., $\mathcal{R}(P)\ \forall\ P \in \mathcal{P}(V)$ como mencionado na linha 2 no Alg. 2.

Aceleramos a execução da busca pelos vizinhos mais próximos através da VP-Tree (YIANILOS, 1993). Esta árvore é usada para encontrar um conjunto Y contendo os índices dos ψ locais que têm o representante $\mathcal{R}(P)$ mais parecido com o representante do último local visitado $\mathcal{R}(q_\rho)$. Cada um destes lugares selecionados são denominamos como locais candidatos e esta seleção é demonstrada na Fig. 3.2.a.

Figura 3.2: Em *a)*, com base no valor do representante do último local visitado $\mathcal{R}(q_\rho) \in Q$ são escolhidos os ψ vizinhos mais próximos. Neste exemplo, demonstrado em *b)*, o valor de $\psi = 2$ e, portanto são selecionados os vizinhos mais próximos P_{y_1} e P_{y_2} que dão origem às sequências de locais candidatas F_1 e F_2, respectivamente.



Para continuar a comparação nos próximos níveis, para cada local candidato correspondente a cada índice $y_k \in Y$, onde $1 \leq k \leq \psi$, criamos uma sequência de locais candidata chamada F_k, como mostrado em Fig. 3.2.b que consiste em

$$F_k = \left\{ f_1^k, f_2^k, \ldots, f_\rho^k \right\} \tag{3.4}$$

onde f_j^k representa $P_{y_k - \rho + j}$. Dessa forma, temos um conjunto $\mathbf{F} = \{F_1, F_2, \cdots, F_\psi\}$ de todas as sequências de locais candidatas, em que $|F_k| = |Q|$ assim como demonstrado na linha 3 do Alg. 2. Vale ressaltar que para um local ser considerado candidato, ele deve ter $\rho - 1$ locais antes dele para que seja possível construir F_k. Desta forma, os locais pertencentes ao conjunto $\{P_1, P_2, \cdots, P_{\rho-1}\}$ não podem ser considerados como locais candidatos, uma vez que não há a quantidade mínima de locais que os antecedem para que seja possível executar o passo demonstrado na Fig. 3.2.a. Além disso, quando a sequência de consulta Q é criada, de acordo com a Eq. 3.3, se a quantidade de locais existentes no mapa for menor que o tamanho de uma sequência de consulta, i. e., $|\mathcal{P}(V)| \leq \rho$, Q não pode ser processado. Nesse caso, Q é incluído no mapa M e nosso sistema LCD continua na próxima iteração quando uma nova sequência de consulta Q é criada, ou seja, para processar a Q, $|\mathbf{P}| = 2 \cdot \rho$.

3.2.2 Segundo nível: escolhendo a melhor sequência

Após o conjunto $\mathbf{F}$ ter sido definido no primeiro nível, no segundo nível analisamos apenas as sequências que pertencem a este conjunto. Considerando que as sequências contidas em $\mathbf{F}$ são as mais similares a Q, o segundo nível tem o propósito de selecionar qual delas é a mais similar.

A comparação deste nível é mais refinada do que a do nível anterior, uma vez que comparamos os representantes de locais de Q aos representantes de locais de cada sequência candidata $F \in \mathbf{F}$. Procuramos a melhor F^* que possua a menor distância para Q, dado por

$$k^* = \arg\min_k \Big(\sum_{j=1}^{\rho} \mathcal{D}\big(\mathcal{R}(q_j), \mathcal{R}(f_j^k)\big) \Big), \tag{3.5}$$

onde $\mathcal{D}(\cdot,\cdot)$ é a distância entre os respectivos descritores representantes de $q_j \in Q$ e $f_j^k \in F_k$ e o índice k^* define a melhor sequência candidata, ou seja, $F^* = F_{k^*}$.

Vale ressaltar que o processo apresentado nesta seção acontece em iterações, nas quais uma nova iteração começa quando nosso sistema de LCD recebe uma nova sequência de consulta Q. Além de selecionar a sequência canditata F^* na iteração i, temos que verificar se ela é temporalmente consistente com relação à melhor sequência candidata F^* encontrada nas iterações $i-1$ ou $i+1$, ou seja, se elas estão próximas o suficiente em nosso mapa M. Ao selecionar uma sequência no segundo nível, nosso método está indicando o ponto do mapa em que há maior chances do robô estar revisitando. No sentido de refinar a busca por essa região, verificamos se ela está próxima a pelo menos uma das duas regiões do mapa calculadas nas iterações visinhas. Essa ideia origina-se no fato de que, na iteração atual, o robõ provavelmente estará próximo a região em que estava no momento anterior. Baseado nessas suposições a consistência temporal verifica se o robô está próximo da região indicada nas iterações vizinhas, i.e., $F^{*,i-1}$ e/ou $F^{*,i+1}$. Para esse fim, ele valida se a região do mapa em que o método acredita ter fechamento de loop com Q, no caso, $F^{*,i}$ é posterior e próxima da região indicada na última iteração, i.e., $F^{*,i-1}$. Se não for, ele verifica se a sequência $F^{*,i}$ é anterior e próxima da região indicada na próxima iteração, definida por $F^{*,i+1}$.

A avaliação de consistência temporal é executada comparando a diferença entre os índices do último local visitado da sequência selecionada na iteração anterior $F^{*,i-1}$, ou seja, $f_\rho^{*,i-1}$ com o primeiro local adicionado na sequência candidata selecionada na iteração atual $F^{*,i}$, i.e., $f_1^{*,i}$, conforme demonstrado na linha 3 do Alg. 3. Então, a diferença

entre os índices

$$\left|\mathcal{I}\left(f_{\rho}^{*,i-1}\right) - \mathcal{I}\left(f_{1}^{*,i}\right)\right| < \alpha, \tag{3.6}$$

define se $F^{*,i-1}$ e $F^{*,i}$ são temporalmente consistentes, em que $\mathcal{I}(f_j^*) = y_{k^*} - \rho + j$. Portanto, as sequências $F^{*,i-1}$ e $F^{*,i}$ são temporalmente consistentes se estiverem próximas ou tiverem uma sobreposição de até α, conforme ilustrado na Fig. 3.3 itens 1.a e 1.b, respectivamente. No caso positivo, a melhor sequência candidata $F^{*,i}$ de $\mathbf{F}$ pode ser processada pelo terceiro nível, assim como mencionado na linha 7 do Alg. 3.

Por outro lado, se uma inconsistência temporal for detectada em $F^{*,i}$, o método de LCD espera que a próxima iteração valide ou não $F^{*,i}$ em relação à próxima melhor sequência candidata selecionada, como mostrado na Fig. 3.3 itens 2.a e 2.b. Em outras palavras, o método deve aguardar a melhor sequência de $\mathbf{F}$ calculada na iteração $i + 1$, i.e., $F^{*,i+1}$, conforme demonstrado na linha 9 do Alg. 3. Então, se a distância for menor que α, ou seja

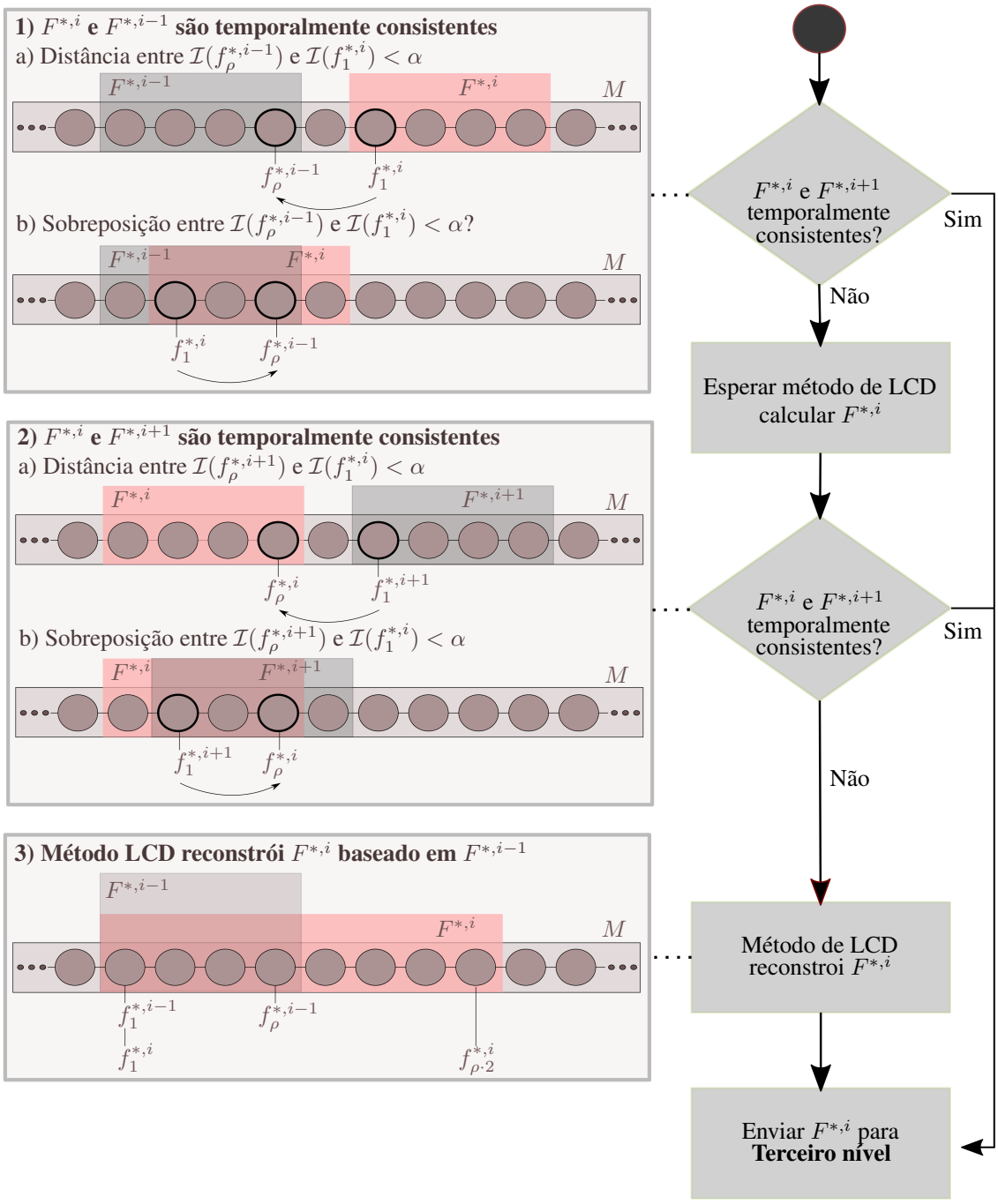
$$\left|\mathcal{I}\left(f_{\rho}^{*,i}\right) - \mathcal{I}\left(f_{1}^{*,i+1}\right)\right| < \alpha, \tag{3.7}$$

isso significa que a sequência $F^{*,i}$ é consistente com $F^{*,i+1}$ pode ser processada pelo terceiro nível conforme mencionado nas linhas 4 e 5 do Alg. 3.

Algorithm 3 Segundo nível do método de LCD

```
1:  procedure LCD_SegundoNivel(F, M)
2:      Encontrar a sequência candidata F^{*,i} em F que é mais similar a Q
3:      if F^{*,i} e F^{*,i-1} são temporalmente consistentes then
4:          if F^{*,i-1} está esperando para ser processada then
5:              LCD_TerceiroNivel(F^{*,i-1}, M)
6:          end if
7:          LCD_TerceiroNivel(F^{*,i}, M)
8:      else
9:          F^{*,i} deve esperar para ser processada
10:         if F^{*,i-1} está esperando para ser processada then
11:             Criar uma sequência sintética _sF baseada em F^{*,i-2}
12:             F^{*,i-1} recebe F^{*,i-2} ∪ _sF
13:             LCD_TerceiroNivel(F^{*,i-1}, M)
14:         end if
15:     end if
16: end procedure
```

Figura 3.3: Depois que nosso método seleciona a sequência candidata mais semelhante a Q, ou seja, $F^{*,i}$, é executada a verificação de consistência temporal de $F^{*,i}$ com as sequências $F^{*,i-1}$ e $F^{*,i+1}$. Se $F^{*,i}$ é consistente com $F^{*,i-1}$ ou $F^{*,i+1}$, então é enviada para o terceiro nível do nosso método. Caso contrário seja inconsistente com ambas as sequências visinhas



, ela é reconstruída com base na sequência $F^{*,i-1}$, por ser considerado o ponto do mapa em que há maior chance de o robô estar próximo.

Caso contrário, se $F^{*,i}$ não for consistente com $F^{*,i-1}$ e $F^{*,i+1}$, uma sequência sintética ${}_sF^i$ é criada conforme linha 11 do Alg. 3. Essa sequência sintética é criada

com base na suposição de que o robô deva estar próximo da melhor sequência de locais calculada na última iteração, i.e., $F^{*,i-1}$. No entanto, uma vez que não é possível inferir o quão longe o robô foi ou mesmo se ele se moveu, aumentamos a incerteza criando uma sequência maior baseada em $F^{*,i-1}$ como mostrado na Fig. 3.3 item 3. Portanto, cria-se uma sequência sintética ${}_sF^i$ composta de um grupo com ρ locais que sucedem o último local visitado na iteração anterior, i.e. $f_\rho^{*,i-1}$. Nesse sentido,

$$ {}_sF^i = \{{}_sf_1, {}_sf_2, \ldots, {}_sf_\rho\} \tag{3.8} $$

onde ${}_sf_j$ representa P_n e $n = \mathcal{I}(f_\rho^{*,i-1}) + j$. Então, a sequência $F^{*,i}$ é reconstruída ao receber $F^{*,i-1} \cup {}_sF^i$ e pode ser processada pelo terceiro nível conforme linhas 11 a 13 do Alg. 3.

3.2.3 Terceiro nível: correspondência de imagens

Nos dois primeiros níveis, nosso sistema de LCD tem como objetivo encontrar a sequência de locais F^* que é temporalmente consistente e que é mais semelhante à sequência de consulta Q. Agora, o objetivo deste terceiro nível é realizar uma correspondência ainda mais refinada. Até então, estávamos realizando comparações envolvendo apenas os representantes dos locais das sequências, agora, no terceiro nível, a ideia é verificar a similaridade entre os descritores de todas as imagens que compõe os locais que pertencem a sequência de consulta Q com os descritores de todas as imagens dos locais da sequência F^* selecionada no segundo nível.

Antes de comparar os descritores de ambas as seqüências F^* e Q, é necessário prepará-las para o processo de correspondência. Ao invés de Q ter ρ diferentes locais, é gerada uma sequência única de seus descritores, chamada $\bar{Q}$. Os descritores de todas as imagens de todos os lugares que pertencem a Q são agrupados sequencialmente pela função $\mathcal{G}(Q)$ e geram o conjunto

$$ \bar{Q} = \{\bar{q}_1, \bar{q}_2, \cdots, \bar{q}_u\} \tag{3.9} $$

onde $\bar{q}_j$ representa o descritor g_n com $n = \mathcal{I}(\mathcal{R}(q_1)) + j - 1$ e $1 \leq j \leq u$ onde $u = \mathcal{I}(\mathcal{L}(q_\rho))$. A função $\mathcal{L}(\cdot)$ retorna o último descritor do local adicionado a sequência de consulta Q. O mesmo processo é aplicado à sequência selecionada no segundo nível

F^*, gerando $\bar{F}$.

Para comparar os dois conjuntos de descritores agrupados, $\bar{Q}$ e $\bar{F}$, temos que garantir que $\left|\bar{F}\right| \geq \left|\bar{Q}\right|$, uma vez que a ideia é verificar se os descritores da sequência de consulta Q, que deu origem a $\bar{Q}$, possuem correspondência com os descritores em $\bar{F}$ gerados a partir da sequência F^* selecionada no segundo nível. Nos casos em que esta relação não é respeitada, é necessário aumentar o tamanho de $\bar{F}$ como mencionado na linha 2 do Alg. 4. Então, os descritores de imagens capturadas antes e depois da sequência de locais F^*, ou seja, pertencentes aos locais que antecedem e procedem F^* no conjunto de locais $\mathbf{P}$, são agrupados na sequência de descritores $\bar{F}$, dados seus respectivos lados, como mostrado na Fig. 3.4.a. Quando o local anterior ou o próximo não existe em $\mathbf{P}$, apenas o lado existente é incluído. Existe uma possibilidade de $\mathbf{P}$ não ter a quantidade de locais cujos descritores permitam o aumento de $\bar{F}$. Neste caso, assumimos que o método não é capaz de determinar se há fechamento de loop para as imagens em $\bar{Q}$. Assim, a sequência de consulta Q é incluída no mapa M e nosso sistema LCD continua na próxima iteração.

Algorithm 4 Terceiro Nível do método de LCD

1: **procedure** LCD_TERCEIRONIVEL(F^*, M)
2: Complementar a diferença de tamanho entre $\bar{Q}$ e $\bar{F}$
3: Selecionar a sequência $\bar{F}^{*,i} \in \bar{F}$ mais similar a $\bar{Q}$
4: Criar a sequência sintética de descritores ${}_s\bar{F}^i$ baseada em $\bar{F}^{*,i-1}$
5: Escolher entre $\bar{F}^{*,i}$ e a sequência sintética ${}_s\bar{F}^i$
6: **return** confiança das correspondências
7: **end procedure**

Dado que $\left|\bar{F}\right| \geq \left|\bar{Q}\right|$, o terceiro nível do nosso método de LCD é capaz de calcular a distância entre os descritores usando a abordagem de *Template Matching*, na qual a sequência de descritores $\bar{Q}$ é deslocada sobre $\bar{F}$ como mostrado na Fig. 3.4.b. Então, na iteração i, conforme apresentado na linha 3 no Alg. 4, nós procuramos a subsequência $\bar{F}^* \subseteq \bar{F}$ que tem a menor distância para $\bar{Q}$, dada por

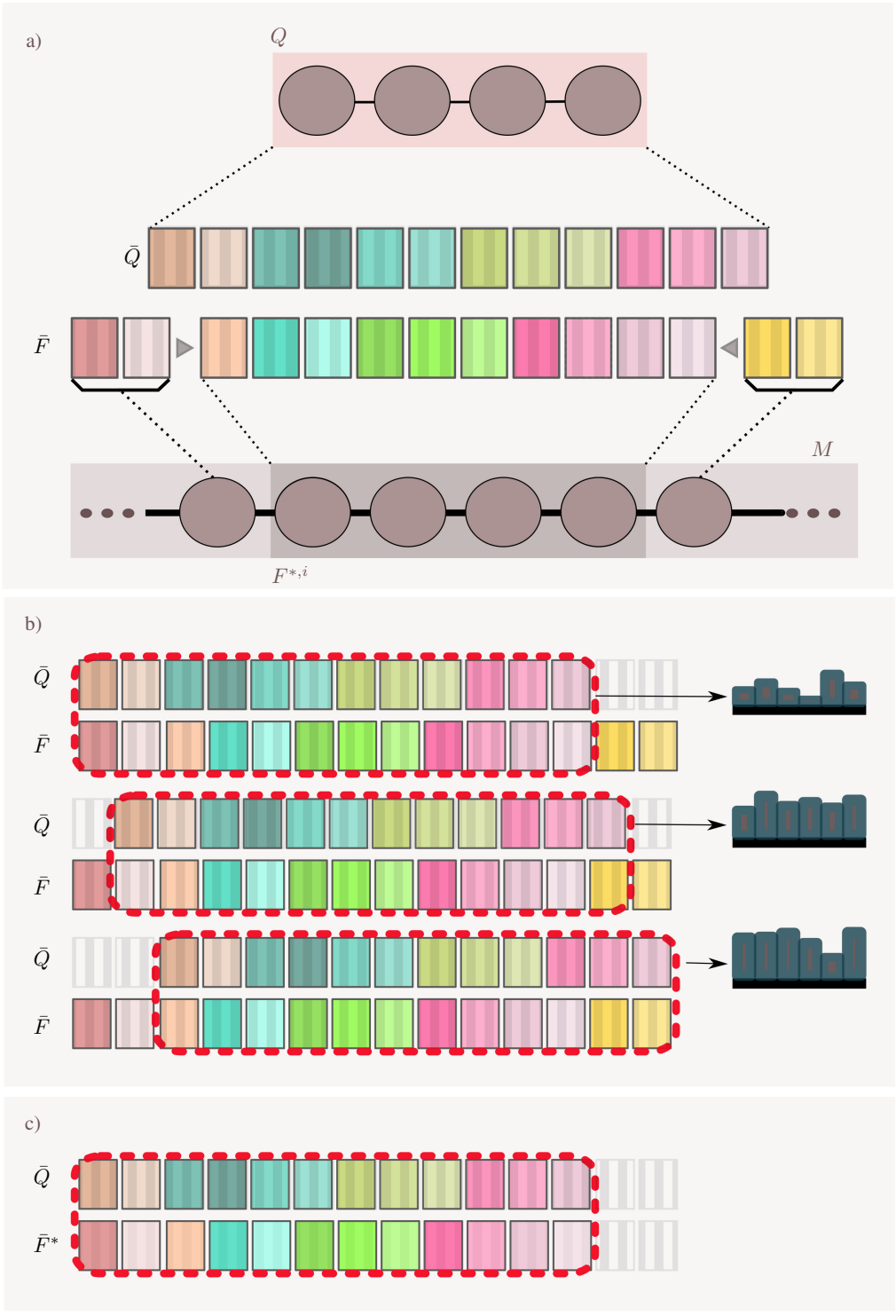
$$j^* = \underset{0 \leq j \leq b}{\arg\min} \Big(\sum_{k=1}^{|\bar{Q}|} \mathcal{D}(\bar{q}_k, \bar{f}_{(j+k)}) \Big), \tag{3.10}$$

onde $b = \left|\bar{F}\right| - \left|\bar{Q}\right|$ e j^* representa o índice do conjunto de descritores mais semelhantes a $\bar{Q}$, definidos como

$$\bar{F}^* = \left\{ \bar{f}_1^*, \bar{f}_2^*, \cdots, \bar{f}_u^* \right\}, \tag{3.11}$$

onde $\bar{f}_j^*$ representa $\bar{f}_{j^*+j-1}$ e $u = \left|\bar{F}\right|$, como pode ser visto na Fig. 3.4.c.

Figura 3.4: No terceiro nível, primeiramente é verificado se $\left|\bar{F}\right| \geq \left|\bar{Q}\right|$, caso contrário, como demonstrado em a), alguns descritores vizinhos são agrupados em $F^{*,i}$ a fim de complementar a sequência de descritores $\bar{F}$. Posteriormente, conforme exposto em b), é aplicado o *Template Matching* em $\bar{Q}$ e $\bar{F}$. A partir desse passo, apresentado em c), o método de LCD seleciona a subsequência $\bar{F}^*$, que corresponde à porção de $\bar{F}$ mais semelhante a $\bar{Q}$, de acordo com a Eq. 3.10.



O grupo de descritores $\bar{F}^*$ representa a região escolhida pelo nosso método de LCD, em que há uma chance maior de ocorrência de fechamento de *loop* com os descritores em $\bar{Q}$. Essa escolha é fortemente baseada na similaridade entre os descritores das imagens capturadas e aquelas já armazenadas no mapa. No entanto, essa informação pode ser ainda mais precisa quando consideramos neste processo que o robô provavelmente deve estar em uma região posterior e próxima da região calculada no terceiro nível da última iteração. Assim, além de comparar $\bar{Q}$ a $\bar{F}^*$, também comparamos $\bar{Q}$ a uma sequência sintética de descritores ${}_s\bar{F}$ que é criada de acordo com a Fig. 3.5.a. Esta sequência sintética tem o papel de representar a região que é adjacente e posterior à calculada no terceiro nível da última iteração e portanto é composta dos descritores seguintes à sequência de descritores selecionada na iteração $\bar{F}^{*,i-1}$ dentro do conjunto G que contém os descritores das imagens capturadas até então. Nesse sentido,

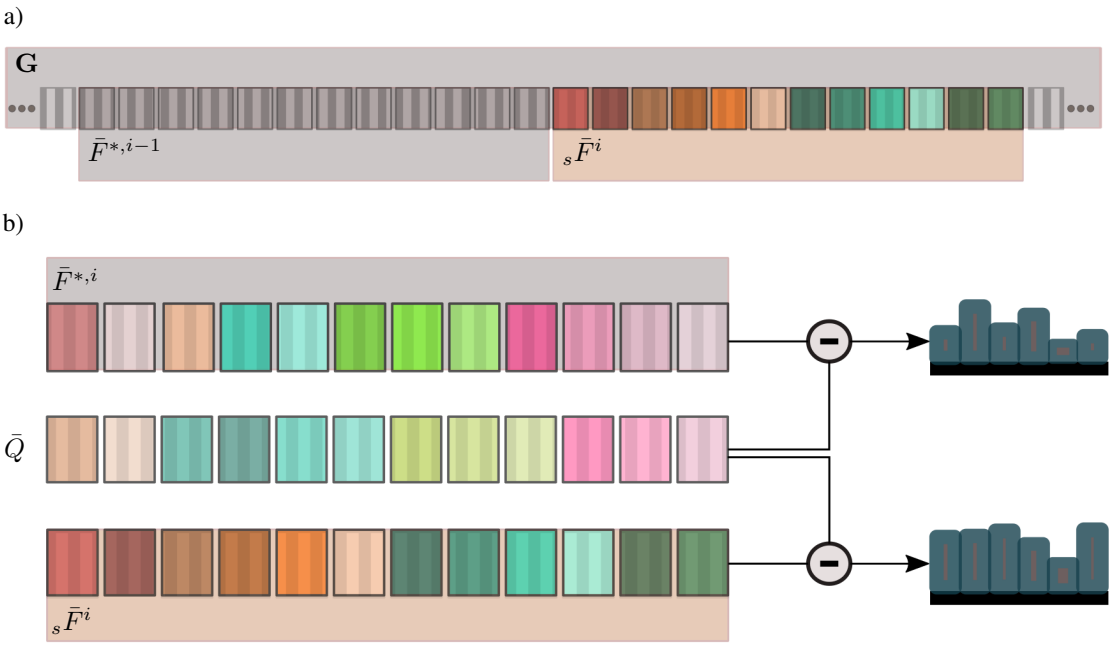
$$ {}_s\bar{F} = \left\{ {}_s\bar{f}_1, {}_s\bar{f}_2, \cdots, {}_s\bar{f}_u \right\} \tag{3.12} $$

onde ${}_s\bar{f}_j$ representa o descritor g_n com $n = \mathcal{I}(\mathcal{L}(\bar{F}^{*,i-1})) + j$ e $1 \leq j \leq \left|\bar{F}^{*,i-1}\right|$. Por fim o método verifica se a sequência sintética ${}_s\bar{F}$ é mais similar a $\bar{Q}$ que a sequência de descritores $\bar{F}^{*,i}$ através da equação

$$ \sum_{d=1}^{|\bar{Q}|} \mathcal{D}\left({}_s\bar{f}_d, \bar{q}_d\right) \leq \sum_{d=1}^{|\bar{Q}|} \mathcal{D}\left(\bar{f}_d^*, \bar{q}_d\right), \tag{3.13} $$

que é ilustrada na Fig. 3.5.b. Se for o caso, então $\bar{F}^{*,i}$ recebe a sequência de descritores sintética ${}_s\bar{F}^i$ para o próximo cálculo. Vale mencionar que ${}_s\bar{F}$ não é criada quando a sequência de locais $F^{*,i}$ é apenas temporalmente consistente com $F^{*,i+1}$ no segundo nível do método. A razão é que não faz sentido usar informações dos descritores agrupados em $\bar{F}^{*,i-1}$ para gerar uma sequência sintética no terceiro nível quando no segundo nível $F^{*,i}$ foi considerado não temporalmente consistente com a sequência anterior $F^{*,i-1}$.

Figura 3.5: A sequência sintética ${}_s\bar{F}^i$ é criada a partir dos descritores subsequentes a $\bar{F}^{*,i-1}$, onde $\left|{}_s\bar{F}^i\right| = \left|\bar{F}^{*,i-1}\right|$, como mostrado em a). Após a criação de ${}_s\bar{F}^i$, o método verifica se a subsequencia $\bar{F}^{*,i}$ é mais parecida com $\bar{Q}$ do que a sequência sintética ${}_s\bar{F}^i$, como mostrado em b).



Dado que $\bar{F}^{*,i}$ é a subsequência mais semelhante a $\bar{Q}$, assumimos que existe a possibilidade de ter um fechamento de *loop*. Dessa forma, a confiança do fechamento de *loop* υ de todos os pares de descritores é a mesma e é calculada por

$$\upsilon = \eta \cdot \sum_{d=1}^{|\bar{Q}|} \mathcal{D}\left({}_s\bar{f}_d, \bar{q}_d\right), \tag{3.14}$$

onde $\eta = \left|\bar{Q}\right|^{-1}$.

4 RESULTADOS

Nesta seção, avaliamos nosso sistema LCD considerando os resultados obtidos da execução em dois *datasets*, ambos descridos na Seção 4.1. Na Seção 4.2 é realizada a avaliação da fase de pré-processamento de imagens proposta. Na Seção 4.3 é descrita a implementação do método SeqSLAM, denominada OpenSeqSLAM, que é utilizada para realizar a comparação com nosso método. Na Seção 4.4 é apresentada a análise dos testes aplicados aos datasets supracitados a fim de verificar os resultados obtidos pelo nosso método. Finalmente, na Seção 4.5 é realizada uma análise do espaço de busca utilizado nos métodos de LCD comparados. Em relação aos parâmetros, nosso método assumiu em todos os testes os valores descritos na Tab. 4.1.

Tabela 4.1: Parâmetros

Parâmetro	**Descrição**	**Valor**
ρ	$\|Q\|$ e $\|F\|$	5
ψ	Número de vizinhos mais próximos	10
ζ	A diferença máxima para o *ground truth* para ser considerada como uma correspondência válida	3
α	A distância máxima permitida entre o índice de $F^{*,i-1}$ e $F^{*,i}$ para considerá-los temporalmente consistentes	5

4.1 *Datasets*

Dois *datasets* públicos foram usados nos experimentos e suas descrições podem ser vistas na Tab. 4.2. Cada *dataset* é composto por uma travessia de consulta e uma travessia de destino, que são representadas por duas sequências de imagens. Essas sequências foram capturadas em momentos distintos, no entanto representam o mesmo caminho em um ambiente que alterna passagens por lugares internos e ao ar livre. Por esse motivo, é possível perceber diferenças visuais na iluminação, no ponto de vista, a presença do movimento de pessoas e a discrepância na qualidade das imagens. Para ambos os *datasets*, uma imagem da **travessia de consulta** tem sempre uma imagem equivalente na **travessia**

de destino, isto é, as travessias são compostas por pares de imagens do mesmo ponto, que foram tiradas em momentos diferentes. No caso específico do *dataset* GPW, ele é composto por três travessias, sendo que todas possuem imagens correspondentes entre si.

Tabela 4.2: Descrição dos *Datasets*

Nome	**# Imagens**	**Configuração**
Gardens Point Walking (GPW)[1]	3 x 200	Três travessias do mesmo caminho: duas capturadas durante o dia: *Day - Left* e *Day - Right* (imagens capturadas manualmente com a mão esquerda e direita) e a terceita a noite: *Night - Right* (com a mão direita), através do Gardens Point Campus, Queensland University of Technology, Brisbane, Australia.
UofAlberta[2]	2 x 646	Duas travessias do mesmo caminho: a primeira capturada durante o dia: *Day*, a segunda ao entardecer: *Evening*, através da University of Alberta, Edmonton, Canada. Ambas coletadas por um robô Husky.

4.2 Avaliação do pré-processamento de imagens

O objetivo deste experimento é demonstrar a melhoria na recuperação de imagens correspondentes ao aplicar a abordagem de pré-processamento apresentada na Seç. 3.1.1. Para tanto, primeiramente os datasets são submetidos ao pré-processamento proposto, i.e.,

[1] https://wiki.qut.edu.au/display/cyphy/Day+and+Night+with+Lateral+Pose+Change+Datasets
[2] https://github.com/siam1251/Fast-SeqSLAM

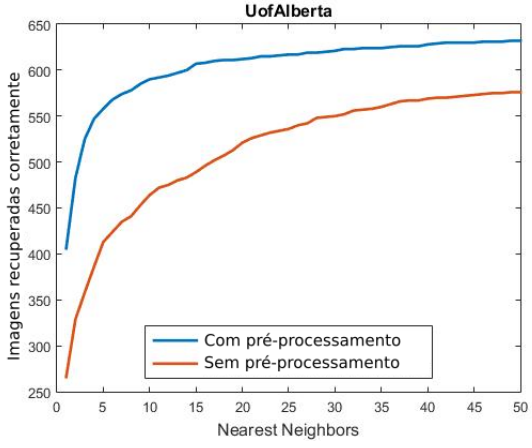
as imagens de cada uma das travessias foram submetidas a um filtro gaussiano, com uma máscara de tamanho igual a 5 e ao processo de equalização do histograma, fornecido pelo OpenCV[3]. Em seguida, foi extraído o descritor RST-SHelo de cada uma dessas imagens pré-processadas.

As travessias de um dataset representam a mesma rota e possuem a mesma quantidade de imagens correspondentes. Portanto, para cada descritor g_u pertencente a travessia de consulta, há um descritor equivalente e de mesmo índice na travessia de destino. Consideramos como correspondências válidas para o descritor g_u da travessia de consulta, os descritores $\{g_{u-\zeta}, g_{u-\zeta+1}, \cdots, g_{u+\zeta}\}$ pertencentes à travessia de destino.

Para verificar se o método de pré-processamento melhora a qualidade das correspondências, para cada imagem na travessia da consulta, são buscados os ψ descritores da travessia de destino mais similares. Espera-se que entre os vizinhos mais próximos selecionados haja ao menos um em que o índice do descritor de imagem de destino esteja no máximo ζ de diferença do índice do descritor de consulta.

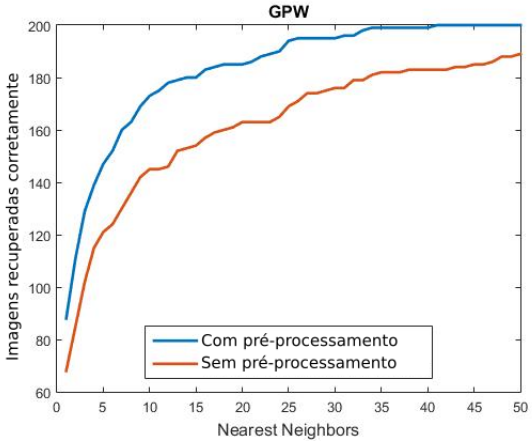
Conforme pode ser visto nas Figs. 4.1 e 4.2, o eixo horizontal mostra a variação do número de vizinhos mais próximos considerados na busca. Já no eixo vertical há a soma das correspondências corretas de acordo com o número de vizinhos mais próximos do eixo horizontal. Para realização dos testes, no caso do dataset UofAlberta foi utilizada a travessia *Day* como destino e a *Evening* como travessia de consulta. Já para o dataset GPW a travessia de destino é a *Day - Left* e a travessia de consulta é a *Night - Right*. É possível notar que a abordagem de pré-processamento aumentou a quantidade de imagens recuperadas corretamente em ambos os casos.

Figura 4.1: Capacidade de recuperação correta de imagens com e sem o uso do pré-processamento proposto. Os resultados variam de acordo com a quantidade de vizinhos mais próximos considerados para o *dataset* UofAlberta.



[3]https://docs.opencv.org/3.0.0/d4/d1b/tutorial_histogram_equalization.html

Figura 4.2: Capacidade de recuperação correta de imagens com e sem o uso do pré-processamento proposto. Os resultados variam de acordo com a quantidade de vizinhos mais próximos considerados para o *dataset* GPW.



Nossa abordagem de pré-processamento de imagens proposta permite a geração de descritores mais distintos, dessa forma é possível recuperar imagens com maior precisão. O melhoria da qualidade na recuperação de imagens impacta positivamente em nosso método de LCD proposto. Isto porque a detecção de fechamento de *loop* baseado em sequência de três níveis depende da busca de imagens para selecionar locais corretos no primeiro e no segundo níveis. Da mesma forma, o uso do pré-processamento auxilia na correspondência direta entre os descritores no terceiro nível do nosso método.

4.3 Utilização do framework Open SeqSLAM

A fim de validarmos o método proposto o comparamos com o atual estado da arte SeqSLAM (MILFORD; WYETH, 2012). Originalmente, este método não disponibiliza a implementação oficial, no entanto, os autores Sünderhauf, Neubert e Protzel (2013) criaram o OpenSeqSLAM que é uma implementação que possibilita o teste do método SeqSLAM.

Para executar o método OpenSeqSLAM é necessário utilizar uma **travessia de consulta** e uma **travessia de destino**, ambas com o mesmo número de imagens. Durante a execução, cada imagem da travessia de consulta é submetida ao método do OpenSeqSlam que retorna qual imagem da travessia de destino tem a maior chance de representar um fechamento de *loop*. Dessa forma, o resultado final do método para cada imagem pesquisada corresponde a três colunas:

Índice imagem do GT	Índice imagem selecionada na travessia destino	% de certeza do *matching*

A partir dessas informações é possível calcular o valor de *precision-recall* através da variação do valor de % de certeza do *matching*. É considerado como *mathings* válidos, i.e., *true positives* todas as imagens selecionadas cujo índice estiver até ζ do GT. O mesmo critério de avaliação é utilizado em nosso método.

Quanto aos parâmetros para execução do Open SeqSLAM, foram selecionados os valores que maximizam os resultados do SeqSLAM conforme demonstrado na Tab. 4.3. Os valores foram selecionados de acordo com os escolhidos no próprio artigo do SeqSLAM (MILFORD; WYETH, 2012) e no Fast-SeqSLAM (SIAM; ZHANG, 2017). No caso do parâmetro do tamanho da sequência, selecionamos o valor de 40 por ter resultado em uma melhora dos resultados apresentados. De acordo com Sünderhauf, Neubert e Protzel (2013) o parâmetro do tamanho da sequência τ é o que tem maior influência na qualidade de execução do SeqSLAM. Quanto maior o valor indicado, maior a chance de detectar um *loop* corretamente. Em contrapartida, sequências muito grandes causam dificuldades em detectar loops em *datasets* que possuem sequências similares menores que o tamanho de τ. Essa situação ocorre, e.g., quando há muitas curvas.

Tabela 4.3: Parâmetros do Open SeqSLAM

Parâmetro	**Descrição**	**Valor**
τ	Tamanho da sequência	40
υ_x, υ_y	Redimensionamento das imagens	32,32
ν_{min}	Velocidade mínima da trajetória	1
ν_{max}	Velocidade máxima da trajetória	1.5
ζ	A diferença máxima para o *ground truth* para ser considerada como uma correspondência válida	3

Os testes a seguir executam o OpenSeqSLAM utilizando os mesmos parâmetros da Tab. 4.3 para todos os datasets. O propósito desses testes é de demonstrar os resultados do OpenSeqSLAM em comparação com o nosso método quando executados nos *datasets* documentados na Tab. 4.2.

4.4 Precisão do método de LCD proposto

Para verificar a precisão do nosso método proposto de LCD, analisamos os resultados em dois aspectos: a proximidade do *ground truth* (GT) e o índice do descritor da imagem selecionada como candidata a fechamento de loop; a segunda análise é em relação ao *precision-recall* resultante da execução do método. Nossos experimentos são baseados em trabalhos semelhantes já propostos (SIAM; ZHANG, 2017; SÜNDERHAUF; NEUBERT; PROTZEL, 2013), nos quais a travessia de destino é usada para criar um mapa e, em seguida, a travessia de consulta é processada para procurar fechamentos de loop na travessia de destino.

Para todos os testes descritos a seguir submetemos a travessia de destino e a travessia de consulta do *dataset* ao processo descrito em *Pré-processamento de imagens*, Seç. 3.1.1. Em seguida, agrupamos as imagens da travessia de destino ao processo descrito em *Criação do local*, Seç. 3.1.2. Os locais resultantes e suas conexões constituem o mapa M. Depois que M é criado, processamos de forma incremental a travessia de consulta para gerar uma sequência de consulta Q, de acordo com o Alg. 1. Sempre que uma nova sequência Q é criada, o processo de LCD proposto é executado sobre o mapa M.

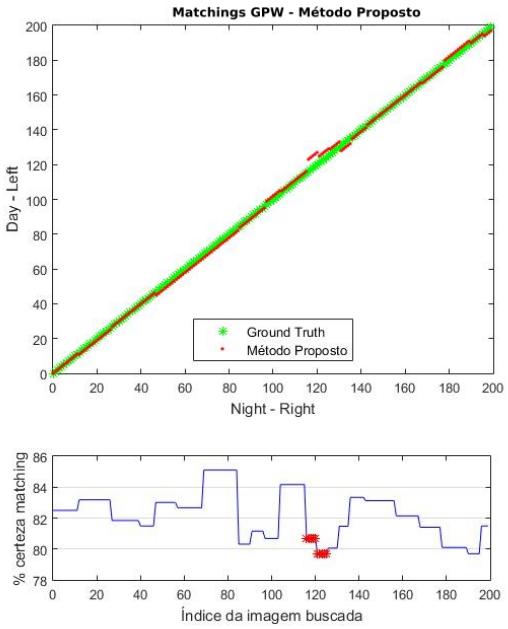
4.4.1 Primeiro experimento: comparação da execução dos métodos de LCD sobre os *datasets* originais

Primeiramente, verificamos a proximidade dos *matchings* selecionados por nosso método e pelo método OpenSeqSLAM em relação ao *ground truth* (GT) para os datasets descritos na Tab. 4.2. No caso de ambos os datasets, para toda a imagem da travessia de consulta há uma imagem correspondente e de mesmo índice na travessia de destino, portanto, o gráfico de *matchings* ideal deveria corresponder a uma diagonal, como demonstrado nos GT da Fig. 4.3 à Fig. 4.8. Ainda nessas imagens, é demonstrado o % de certeza do *matching* para cada uma das correspondências encontradas em que a variabilidade apresentada está de acordo com estratégia adotada por cada método. No caso de ambos os métodos comparados, os valores apresentados tem relação com o cálculo de similaridade entre descritores buscados e os candidatos a fechamento de *loop*.

Na primeira execução utilizamos a travessia "Night - Right" como consulta e a "Day - Left" como destino, ambas do dataset GPW. Nas Figs. 4.3 e 4.4 é possível verificar os *matchings* encontrados pelo nosso método proposto de LCD e pelo OpenSeqSLAM,

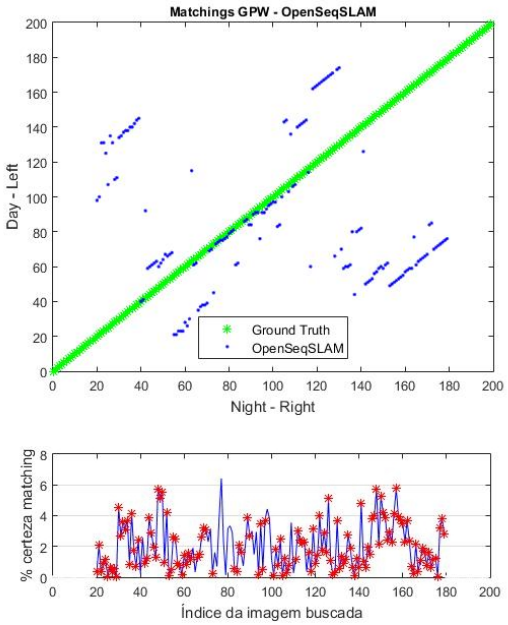
respectivamente. No caso da execução do nosso método é possível verificar que a maioria dos resultados foi próximo ao GT. Ocorrem irregularidades nas seqüências correspondentes às imagens 116 a 120 e 121 a 125, que representam duas sequências de consultas Q executadas consecutivamente. Em ambos os casos, o Segundo Nível do nosso método selecionou uma sequência de locais F^* que contém a sequência correta de imagens correspondentes à Q. No entanto, no terceiro nível, ao realizar o *Template Matching*, o resultado selecionado foi uma sequência de imagens que é deslocada em relação ao GT. Apesar de escolher esses *matchings* incorretamente, atribui a eles um % de certeza do matching baixo, como pode ser visto destacado em vermelho no segundo gráfico da Fig. 4.3. É importante que o método responda o % de certeza do matching baixo para um *matching* incorreto já que é este valor que define se há um fechamento de *loop*. Por ser um valor baixo há pouco impacto na precisão do método, como poderá ser visto mais a frente na análise de *Precision-recall* dessa execução.

Figura 4.3: *Matchings* selecionados em nosso método para a versão original do *dataset* GPW - travessias "Day - Left"e "Night - Right"



Por outro lado, os matchings encontrados pelo método OpenSeqSLAM para as mesmas travessias são muito diferentes do GT. Esses resultados demonstram uma das principais limitações do método SeqSLAM: de acordo com Sünderhauf, Neubert e Protzel (2013), como o método define a similaridade entre sequências de imagens baseado na subtração direta entre elas, é imprescindível que estejam sob o mesmo ponto de vista. Além disso, diferentemente do nosso método, o OpenSeqSLAM atribui o valor de % de certeza do *matching* alto mesmo para correspondências incorretas.

Figura 4.4: *Matchings* selecionados no OpenSeqSLAM para a versão original do *dataset* GPW - travessias "Day - Left"e "Night - Right"



Quando executamos nossa abordagem e o OpenSeqSLAM, mas agora utilizando a travessia "Day - Right" como destino é possível observar uma melhora dos resultados obtidos em ambos os métodos, assim como no % de certeza dos *matchings* escolhidos. Isso se deve ao fato da travessia "Day - Right" possuir imagens com ponto de vista mais similar a "Night - Right". Ao comparar com a primeira execução demonstrada nas Figs. 4.3 e 4.4, a melhora do OpenSeqSLAM é mais expressiva que a de nosso método, como pode ser visto nas Figs. 4.5 e 4.6, já que ele depende fortemente do ponto de vista.

Figura 4.5: *Matchings* selecionados em nosso método para a versão original do *dataset* GPW - travessias "Day - Right"e "Night - Right"

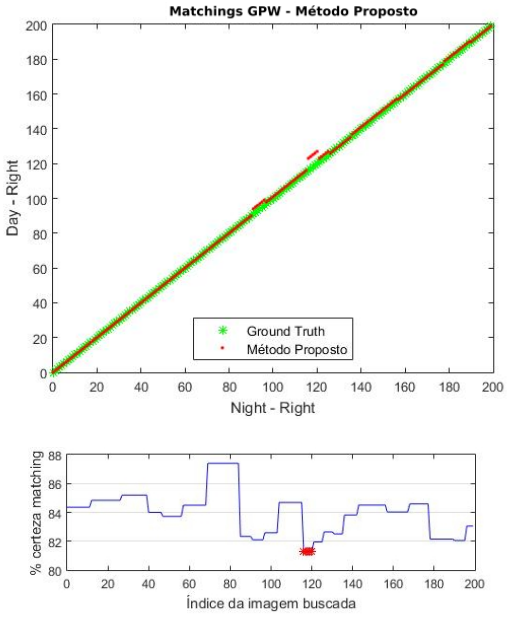
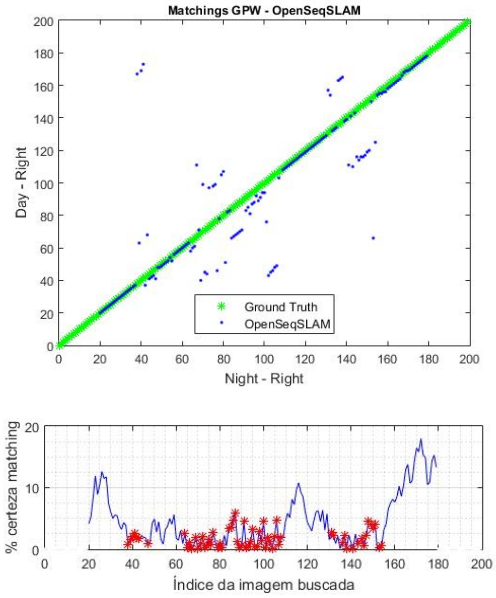
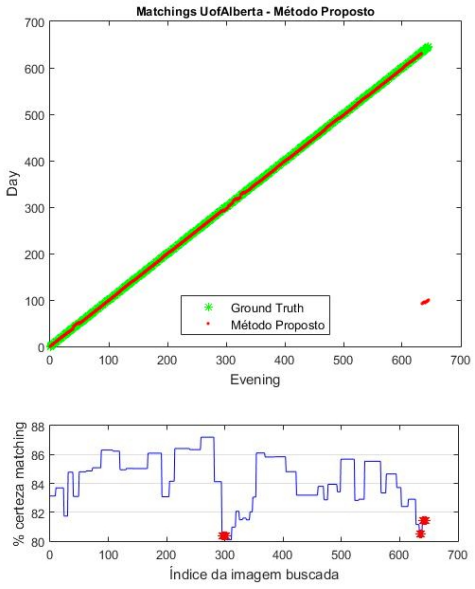


Figura 4.6: *Matchings* selecionados no OpenSeqSLAM para a versão original do *dataset* GPW - travessias "Day - Right"e "Night - Right"



Para o *dataset* UofAlberta, é possível verificar a ocorrência de erros apenas no final da execução de nosso método, conforme mostrado na Fig.4.7. Vale ressaltar que esses erros ocorreram em uma situação específica em relação à consistência temporal de três seqüências selecionadas $F^{*,i-2}$, $F^{*,i-1}$ e $F^{*,i}$. Durante a execução de uma iteração do nosso método de LCD, $F^{*,i-1}$ foi considerada não temporalmente consistente com $F^{*,i-2}$, embora $F^{*,i-2}$ estivesse correta e próxima ao GT. Então, nosso método de LCD aguardou $F^{*,i}$ para verificar a consistência temporal com $F^{*,i-1}$. Quando $F^{*,i}$ foi processada, foi considerada temporalmente consistente com $F^{*,i-1}$, validando ambas, mesmo que não estivessem próximas de $F^{*,i-2}$.

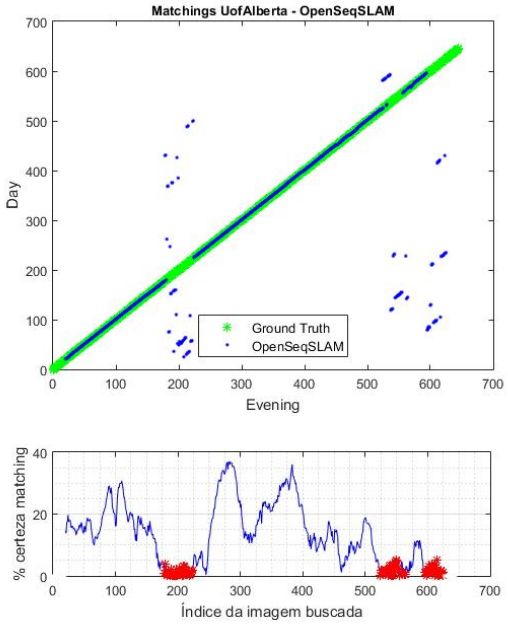
Figura 4.7: *Matchings* selecionados em nosso método para a versão original do *dataset* UofAlberta - travessias "Day"e "Evening"



Apesar desse erro pontual, os resultados obtidos para o *dataset* UofAlberta são precisos e em sua maioria próximos ao GT. Além disso, assim como no *dataset* GPW, para os casos em que as sequências estão mais distantes do GT, nosso método atribuiu valores baixos de % de certeza do *matching*.

Para o mesmo dataset o método OpenSeqSLAM demonstrou dois trechos com erros, como pode ser visto na Fig 4.8. Se compararmos a execução do OpenSeqSLAM sobre o dataset GPW, os resultados obtidos no UofAlberta foram mais próximos ao GT para a maior parte das imagens, embora inferiores aos resultados apresentados pelo nosso método no mesmo dataset. A diferença entre a quantidade de *matchings* corretos também se dá pelo fato de que nosso método analisa a existência de fechamento de *loop* para todas as imagens das travessias, ao contrário do OpenSeqSLAM que não processa as $\frac{\tau}{2}$ imagens do início e fim da travessia de busca.

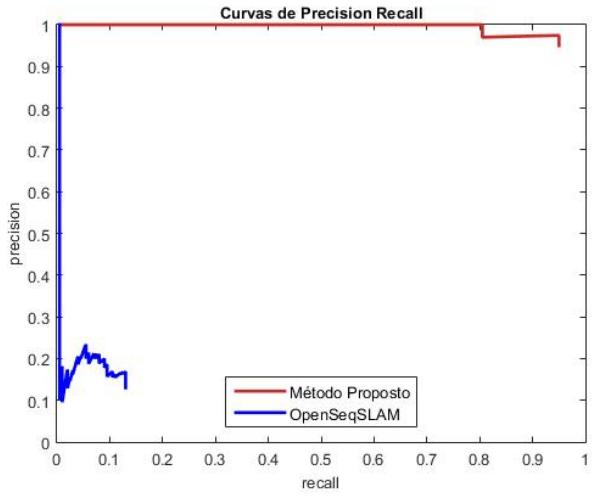
Figura 4.8: *Matchings* selecionados no OpenSeqSLAM para a versão original do *dataset* UofAlberta - travessias "Day"e "Evening"



A segunda análise avalia o *precision-recall* do nosso método em relação ao OpenSeqSLAM. As Figs. 4.9 e 4.10 mostram os resultados para o *dataset* GPW e UofAlberta, respectivamente, em que variamos o *threshold* de % de certeza do *matching* para determinar se há um fechamento de *loop*. Com 100% de precisão, nosso método consegue recuperar 80% do *dataset* GPW para a combinação de trajetórias "Day - Left" e "Night - Right", 97% para as trajetórias "Day - Right" e "Night - Right"e 93% para o *dataset* UofAlberta. Esses resultados são superiores ao OpenSeqSLAM, que atinge 0,5%, 21% e 67% de *recall* com 100% de *precision*, respectivamente.

Figura 4.9: *Precision-recall* para o *dataset* GPW

(a) Day Left - Night Right



(b) Day Right - Night Right

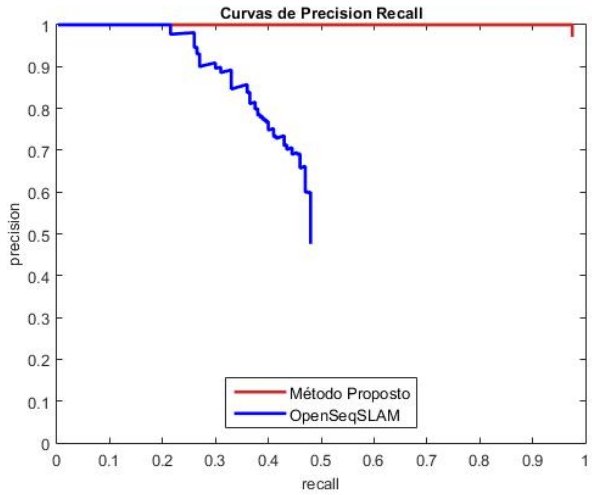
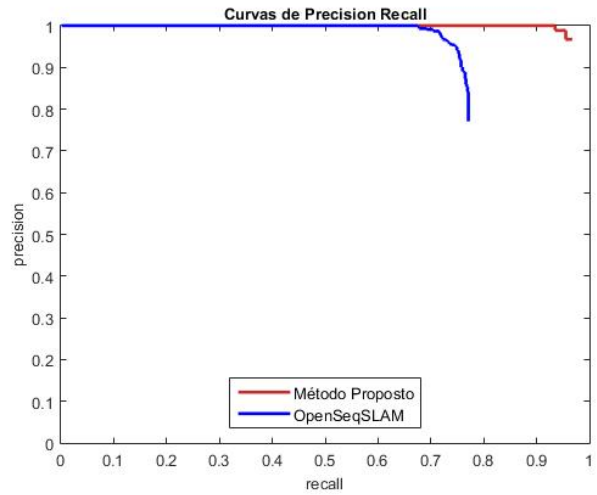


Figura 4.10: *Precision-recall* para o *dataset* UofAlberta



Como demonstrado pelos gráficos da Fig. 4.3 a Fig. 4.8, além da escolha de *matchings* em nosso método ser mais precisa, o cálculo % de certeza do *matching* também o é se comparado ao OpenSeqSLAM. Isto se dá porque nosso método tem a capacidade de atribuir um valor de % de certeza do *matching* baixo ao escolher *matchings* errados, diferentemente do que ocorre com o OpenSeqSLAM. Além disso, os testes evidenciaram que a diferença de ponto de vista influencia a execução do OpenSeqSLAM, impactando expressivamente os valores de *precision recall* para a combinação de trajetórias "Day Left" e "Night Right" ao contrário do nosso método que não sofreu a mesma influência.

Por conta da incapacidade do OpenSeqSLAM em operar sobre trajetórias com ponto de vista muito distintos, nos próximos testes trabalharemos apenas com as trajetórias "Day Right" e "Night Right" do *dataset* GPW.

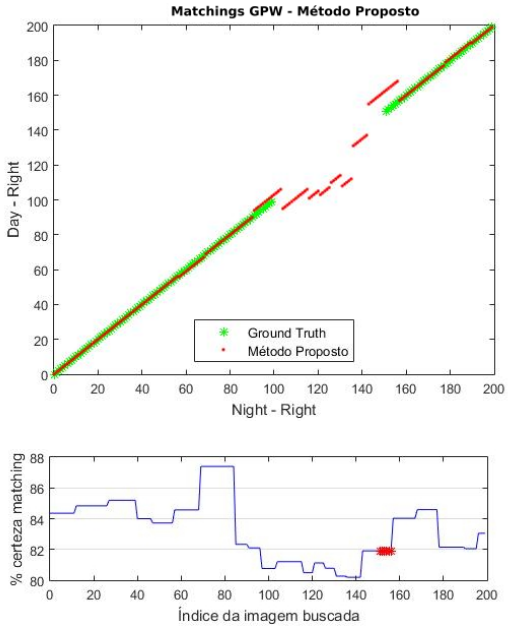
4.4.2 Segundo experimento: comparação da execução dos métodos de LCD sobre os *datasets* com trajetória parcialmente sem correspondência

No primeiro experimento, para cada imagem da trejetória de consulta há uma imagem correspondente na trajetória de destino. Em outras palavras, há sempre um *loop* a ser detectado para cada imagem pesquisada. No entanto, ao executar um método de LCD no mundo real é essencial que ele esteja preparado para lidar com trajetórias diferentes. Isto porque, para que um método de LCD auxilie a estratégia de SLAM a construir um mapa corretamente, é importante que ele indique corretamente quando há um *loop* mas também quando não há.

Para realizar esse teste substituimos um trecho da trajetória de destino "Day - Right" do *dataset* GPW por uma sequência de imagens retirada da trajetória "Evening" do *dataset* UofAlberta. Como consequência, parte do GT não tem correspondência, como pode ser visto nos gráficos das Figs. 4.11 e 4.12.

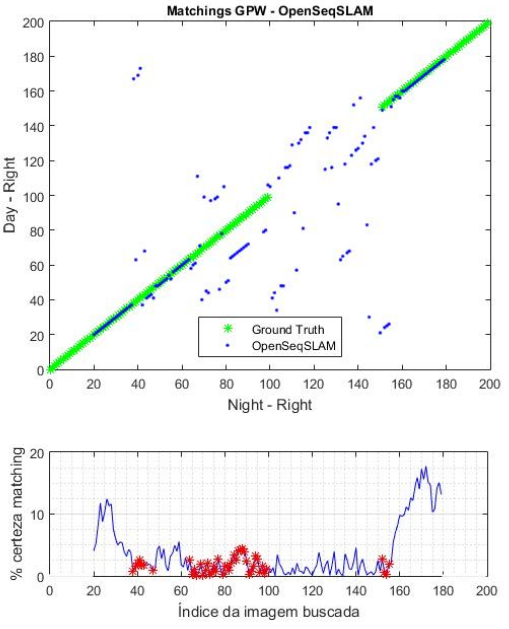
Por padrão, nosso método e o OpenSeqSLAM sempre retornam uma resposta de fechamento de *loop*. Por esse motivo, nas Figs. 4.11 e 4.12 é possível ver que há valores de *matchings* mesmo quando não há correspondência no GT. A diferença entre os métodos se dá no cálculo do % de certeza do *matching*. A Fig. 4.11 mostra que nosso método responde valores baixos para o trecho do GT que não possui correspondência se comparado aos demais valores calculados para o restante da trajetória, evidenciando que não há fechamento de *loop* nessa parte do trajeto.

Figura 4.11: *Matchings* selecionados em nosso método a versão com trajetória parcialmente sem correspondência do *dataset* GPW - travessias "Day - Right"e "Night - Right"



Na execução do método OpenSeqSLAM, demonstrada na Fig. 4.12, o trecho da trajetória que não contém correspondências também recebe valores mais baixos se comparado aos demais calculados. Por outro lado, há maior variação em relação aos valores de % de certeza de *matching* calculados pelo nosso método no mesmo trecho.

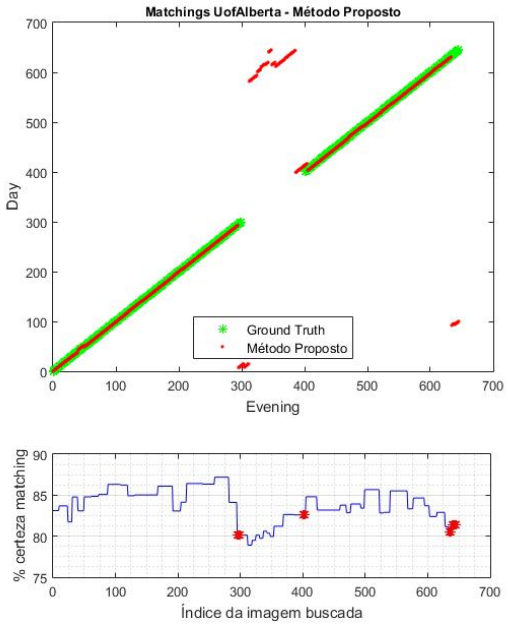
Figura 4.12: *Matchings* selecionados no OpenSeqSLAM para a versão com trajetória parcialmente sem correspondência do *dataset* GPW - travessias "Day - Right"e "Night - Right"



O comportamento do OpenSeqSLAM sob essa configuração de dataset mostrou ser similar a execução sobre o dataset original (primeiro experimento) quando utilizando as trajetórias "Day - Right" e "Night - Right". Percebe-se que na região em que não há correspondências há um espalhamento dos *matchings* selecionados uma vez que o método não possui o comportamento de considerar escolhas de execuções anteriores como influência na escolha do *matching* da iteração atual. Ao contrário, nosso método considera as iterações conseguintes para decidir pela sequência que acredita ser a melhor candidata. Como resultado, no gráfico da Fig. 4.11 é possível observar que próximo ao ponto do GT em que não há correspondência (aproximadamente imagem 100) há um leve distanciamento da sequência escolhida em relação ao GT. Da mesma forma, quanto o GT passa a ter correspondências (aproximadamente imagem 150) é selecionada uma sequência próxima ao esperado e as próximas execuções retornam resultados sobre o GT.

Para realizarmos esse mesmo teste sobre o *dataset* UofAlberta alteramos um trecho da trajetória "Day" que passou a receber uma sequência de imagens da trajetória "Day" do dataset GPW. Por esse motivo, o GT exibido nos gráficos das Figs. 4.13 e 4.14 não possuem correspondências das imagens 300 a 400.

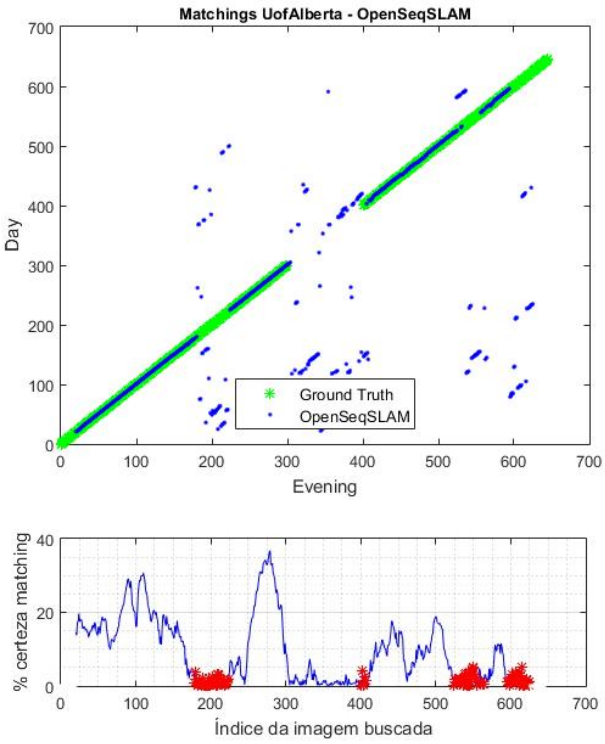
Figura 4.13: *Matchings* do nosso método para a versão parcialmente sem correspondência do UofAlberta



Em ambos os métodos foram calculados valores baixos de % de certeza de *matching* para a maioria do trecho do *dataset* que não tem correspondência, como pode ser visto nas Figs. 4.13 e 4.14. Em nosso método há um aumento do valor de % de certeza de *matching* próximo a imagem 400 porque parte da sequência selecionada está próxima

do GT. Já no método OpenSeqSLAM há um aumento do % de certeza de *matching* aproximadamente na imagem 320 que é indevido, uma vez não há correspondências para esse trecho do *dataset*, como pode ser visto no segundo gráfico da Fig. 4.14.

Figura 4.14: *Matchings* selecionados no OpenSeqSLAM para a versão com trajetória parcialmente sem correspondência do *dataset* UofAlberta - travessias "Day"e "Evening"



Assim como no primeiro cenário, nosso método permanece escolhendo valores baixos de % de certeza de *matching* para quando seleciona *matchings* incorretos (destacado em vermelho no segundo gráfico da Fig. 4.13). Esses erros ocorrem quando o *dataset* deixa de ter correspondência (aproximadamente imagem 300) e quando volta a ter (aproximadamente imagem 400). Esse comportamento é similar ao do OpenSeqSLAM que seleciona o *matching* incorreto próximo a imagem 400 na Fig. 4.14.

A eficiência do método de LCD é dependente do cálculo de % de certeza do *matching* uma vez que se um *loop* é detectado incorretamente a topologia do mapa construído é comprometida. Conforme mencionado, ambos os métodos possuem o comportamento de sempre responder um *matching* para uma imagem da trajetória de busca. No caso desse cenário de teste proposto é esperado que os métodos respondam o % de certeza do *matching* baixos para as respostas relativas a áreas do GT em que não há correspondência e altos para quando há correspondência e ela foi escolhida de forma correta. A variação desses valores influencia o cálculo das curvas de *precision recall* como pode ser visto na

Fig. 4.15 e na Fig. 4.16.

Figura 4.15: *Precision-recall* para o *dataset* GPW com trajetória parcialmente sem correspondência

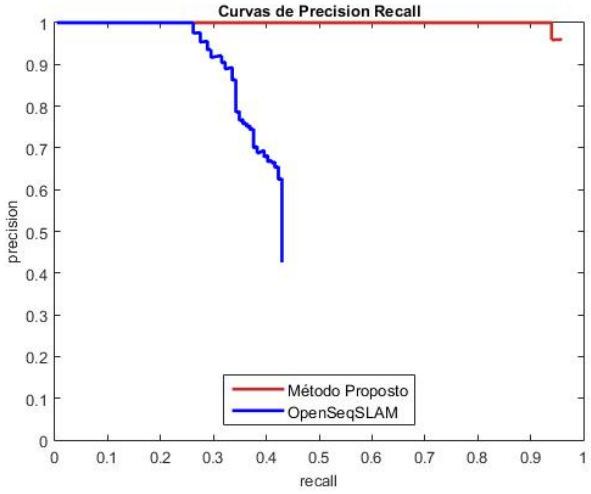
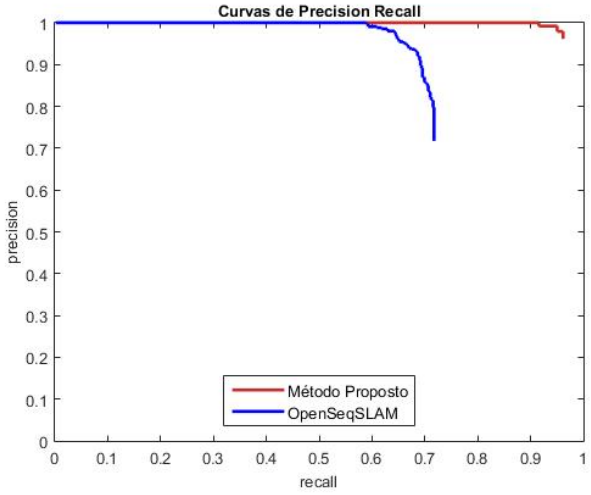


Figura 4.16: *Precision-recall* para o *dataset* UofAlberta com trajetória parcialmente sem correspondência



Com 100% *precision* nosso método atingiu o *recall* de 93% para o *dataset* GPW e 91% para UofAlberta. Em contrapartida, o SeqSlam conseguiu 26% e 58% respectivamente.

4.4.3 Terceiro experimento: comparação da execução dos métodos de LCD sobre os *datasets* com trajetórias em ordens distintas

No primeiro experimento o desafio se dá porque entre as imagens correspondentes das trajetórias há diferença de iluminação, ponto de vista e a presença de dinamicidade

no ambiente. Por outro lado, os resultados são auxiliados pelo fato de que as imagens de ambas as trajetórias são capturadas na mesma sequência ao se executar o mesmo caminho. Já no mundo real, podemos navegar entre ambientes em ordens diferentes, e.g., na primeira vez visitamos as regiões *A*, *B* e depois a *C*. Quando as revisitamos, vamos primeiro a região *B*, depois a *A* e por fim a *C*. Ambos os métodos, tanto o que propomos quanto o OpenSeqSlam, são dependentes da sequência de imagens capturada. Portanto, ao alteramos a ordem de captura das imagens os métodos de LCD são capazes de detectar o *loop* corretamente?

A fim de testarmos essa configuração no *dataset* GPW, deslocamos um trecho de imagens dispostas aproximadamente do centro da trajetória "Day" para o fim dela assim como demonstrado no GT dos gráficos das Figs. 4.17 e 4.18.

Através dos resultados é possível observar que os matchings escolhidos pelo nosso método são mais próximos do GT que os selecionados pelo OpenSeqSLAM. Além disso, quando há as quebras na trejetória nosso método se perde na escolha da sequência por tentar priorizar a que foi escolhida no passo anterior (aproximadamente imagem 100 e 150). No entanto, nas iterações seguintes se recupera escolhendo as sequências corretamente.

Figura 4.17: *Matchings* do nosso método para o *dataset* GPW com trajetórias em ordens distintas.

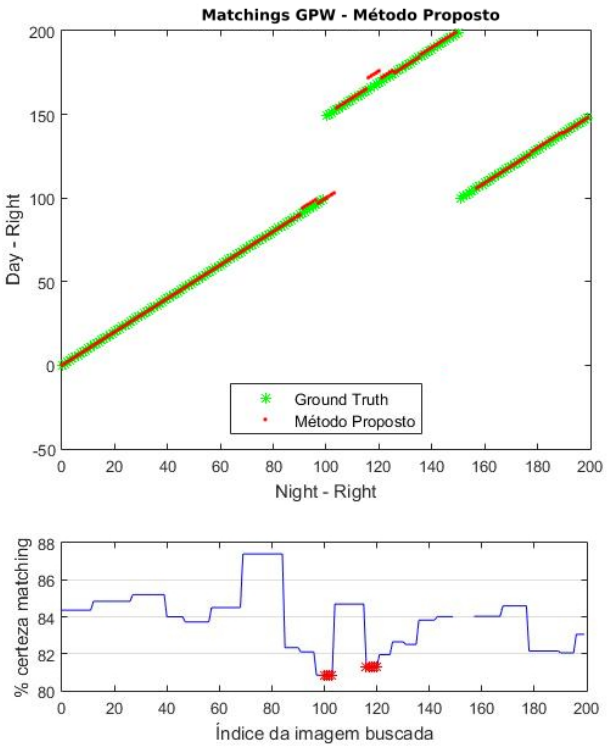
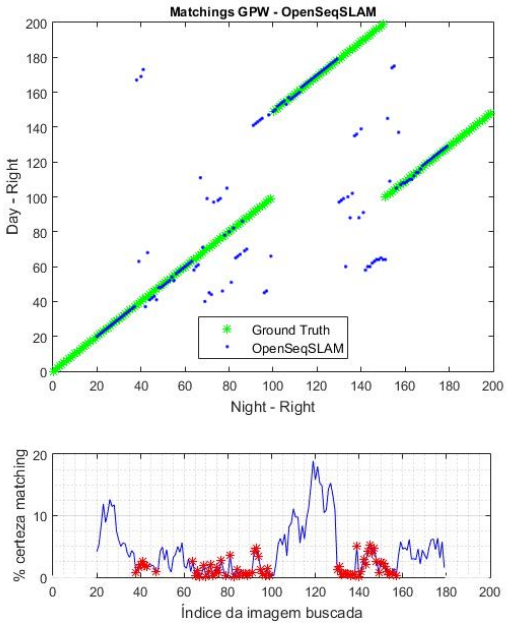


Figura 4.18: *Matchings* do OpenSeqSLAM para o *dataset* GPW com trajetórias em ordens distintas



O comportamento do nosso método se repete no dataset UofAlberta (Fig. 4.19), i.e., demonstrando capacidade de recuperar resultados corretos apesar das quebras do *dataset*. Por outro lado, o OpenSeqSLAM tem dificuldades em manter os resultados, como pode ser visto nos *matchings* da Fig. 4.20. Tal comportamento reforça o fato de que o OpenSeqSLAM depende fortemente das sequências de imagens já que seleciona o *matching* baseado na subtração destas sequências, logo, a existência de quebras na travessia impacta diretamente a qualidade do resultado do método (SÜNDERHAUF; NEUBERT; PROTZEL, 2013).

Figura 4.19: *Matchings* para o *dataset* UofAlberta com trajetórias em ordens distintas

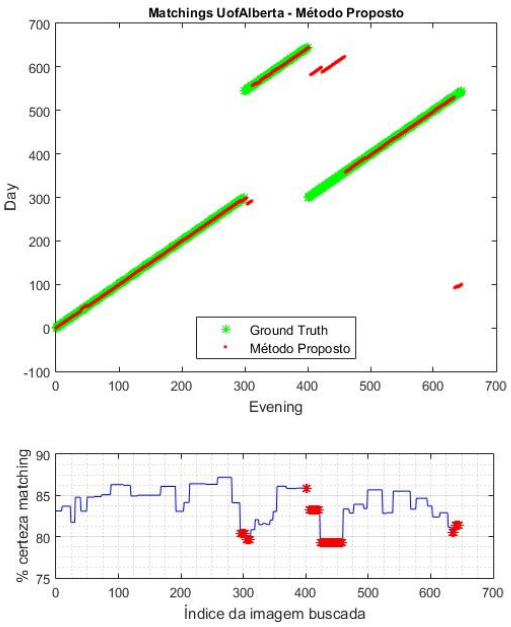
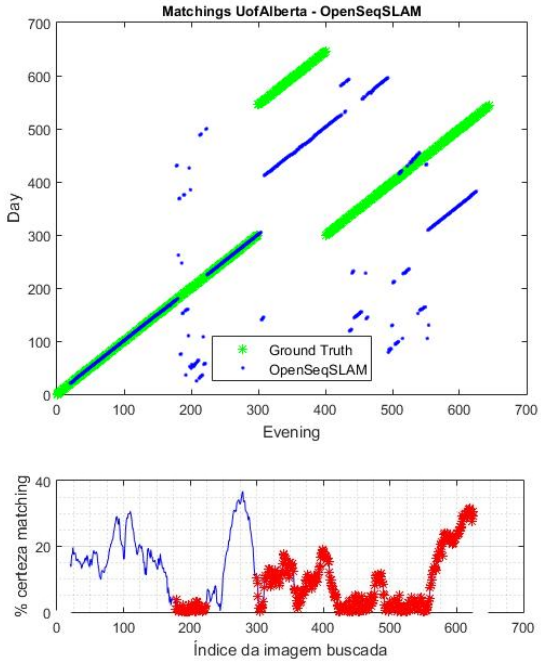


Figura 4.20: *Matchings* para o *dataset* UofAlberta com trajetórias em ordens distintas



Como testamos nosso método utilizando duas trajetórias finitas verifica-se o comportamento demonstrado aproximadamente entre as imagens 150 e 155 da Fig. 4.17 e na imagem 400 da Fig. 4.19, em que nosso método não retorna uma resposta de fechamento de *loop*. Essa situação acontece porque no terceiro nível do nosso método ele seleciona como resposta uma sequência sintética de imagens que é menor que a sequência de imagens de busca. Esse efeito ocorre porque a sequência sintética é gerada próxima do fim da trajetória de destino. Tal comportamento não aconteceria durante a execução do método no mundo real, uma vez que a trajetória de destino, que seria sempre alimentada pelas imagens pertencentes a sequencia de busca recém processada, não teria fim.

Em relação a análise do cálculo de *precision recall*, a diferença entre o nosso método e o OpenSeqSLAM é ainda acentuada, como pode ser visto nos gráficos das Figs. 4.21 e 4.22. Isto ocorre porque, além da inabilidade do OpenSeqSLAM em escolher os *matchings* corretamente, ele também calcula o % de certeza do *matching* de forma indevida. Nosso método, em ambos os *datasets*, mostrou-se mais apto que o OpenSeqSLAM na mesma função. Para 100% de *precision* o OpenSeqSLAM obteve 22% de *recall* para o dataset GPW e 0.2% para o UofAlberta. Já no caso do nosso método, é interessante mencionar apesar de recuperar 90% para o *dataset* GPW com 100% de *precision* ele apresentou apenas 21% de recall para o UofAlberta, enquanto que nos cenários anteriores passava de 90% para o mesmo dataset. Verificamos que essa queda ocorreu por conta do cálculo incorreto de % de certeza do *matching* para uma das sequências selecionadas

pelo nosso método (aproximadamente imagem 400 da Fig. 4.19). Apesar de ser um erro isolado, impactou expressivamente no resultado apresentado.

Figura 4.21: *Precision-recall* para o *dataset* GPW com trajetórias em ordens distintas

(a) *Dataset* GPW

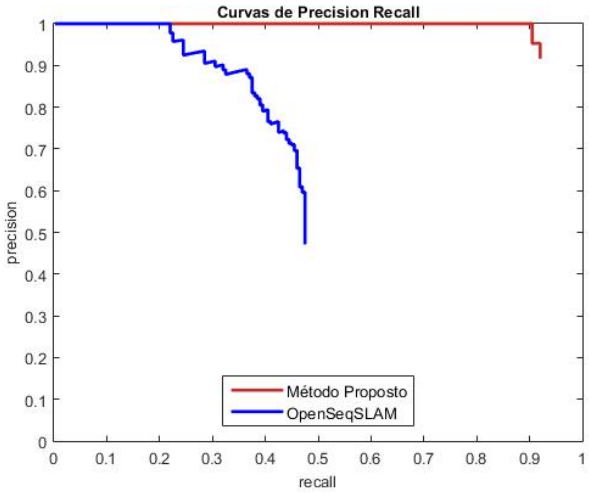
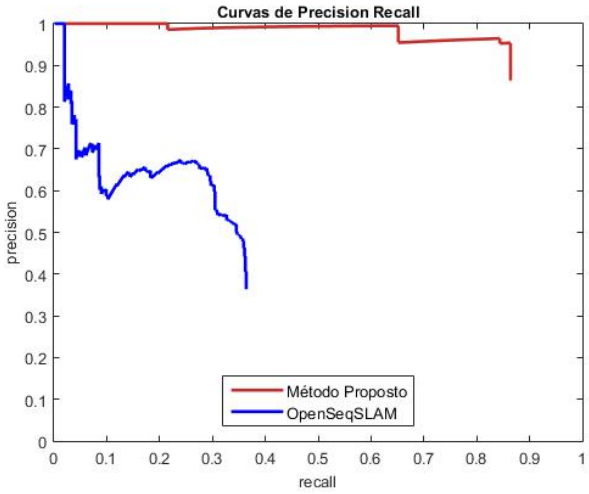


Figura 4.22: *Precision-recall* para o *dataset* UofAlberta com trajetórias em ordens distintas

(a) *Dataset* UofAlberta



4.5 Análise do uso de espaço de busca

Os testes realizados na seção anterior analisam a precisão do método SeqSLAM em relação ao OpenSeqSLAM (SÜNDERHAUF; NEUBERT; PROTZEL, 2013). Assim como a implementação de nosso método, o OpenSeqSLAM foi criado para ser executado

sobre duas trajetórias de tamanho fixo. Essa configuração não corresponde a execução da proposta desses métodos no mundo real.

Ao analizarmos puramente a proposta de ambos os métodos, toda imagem da trajetória de busca, após processada pelo respectivo método de LCD, passa a compor a trajetória de destino. Sob essa perspectiva, realizaremos a comparação do uso de espaço de busca e o número de pesquisas para fechamento de *loop* entre o método que propusemos e o método SeqSLAM criado por Milford e Wyeth (2012), que pode ser vista na Tab. 4.4.

A coluna *# imagens por travessia* representa o número de imagens contidas na travessia de destino e na travessia de consulta do respectivo *dataset* - considerando que ambas as travessias possuem inicialmente o mesmo número de imagens. A coluna *# locais destino* apresenta o tamanho inicial do espaço de busca que é correspondente ao número de locais já visitados no mapa considerados no início da execução do método do LCD. No caso do SeqSLAM, cada imagem do percurso de destino representa um local no mapa, portanto, não há redução no espaço de pesquisa inicial. Por outro lado, nosso método agrupa as imagens da travessia de destino em locais, diminuindo o tamanho inicial do espaço de busca.

Vale ressaltar que o tamanho do espaço de busca aumenta durante a execução do método. A diferença entre as duas abordagens é que o mapa do SeqSLAM é incrementado cada vez que uma imagem é capturada e processada. No caso do nosso método, o mapa recebe locais que podem conter mais de uma imagem. Assim, nosso mapa recebe os representantes de locais recém-criados assim que uma nova consulta é criada. Portanto, o tamanho do nosso espaço de pesquisa sempre será menor que o do SeqSLAM.

Tabela 4.4: Comparação entre espaço de busca inicial e o número de buscas por um fechamento de *loop*

Método	**Dataset**	**# imagens por travessia**	**# locais destino**	**# locais busca**	**# buscas por um fechamento de loop**
SeqSLAM	GPW	200	200	200	200
	UofAlberta	646	646	646	646
Proposto	GPW	200	112	108	22
	UofAlberta	646	236	248	50

A coluna *# locais busca* representa o número de imagens da trajetória de consulta que são usadas como entrada para o método de LCD. No caso do SeqSLAM, todas as imagens da travessia de consulta são usadas para procurar locais já visitados. Em nosso método, devido ao agrupamento de imagens em locais, há a redução no número de entra-

das.

Finalmente, a Tab. 4.4 apresenta o número de buscas pelo fechamento de *loop* executadas em cada *dataset* na coluna *# buscas por um fechamento de loop*. O SeqSLAM executa a busca por um fechamento de loop sempre que uma nova imagem é capturada, portanto, o número de buscas é igual ao número de imagens da travessia de consulta. Em nosso caso, a busca pelo fechamento do *loop* ocorre somente quando os últimos ρ locais visitados são agrupados. Por esse motivo, o número de buscas para fechamento de *loop* é sempre menor em nosso método quando comparado ao SeqSLAM.

5 CONCLUSÕES

Propusemos uma abordagem que faz uso do conceito de sequência apresentado no SeqSLAM, além da criação de um mapa hierárquico para detecção de fechamento de *loop* similar ao desenvolvido por Garcia-Fidalgo e Ortiz (2017). Apesar de nos basearmos em técnicas já utilizadas por métodos de LCD, usamos esses conceitos aplicados de forma distinta em nosso método. Ao agrupar as imagens capturadas sequencialmente em locais, reduzimos a quantidade de informações a serem processadas pelo método. Ao mesmo tempo, destacamos os pontos distintos de uma travessia, uma vez que cada local é representado por um descritor da imagem que seja significativamente diferente da sequência de imagens que a antecede. Tal capacidade descritiva é consequência do pré-processamento de imagens proposto junto as propriedades do descritor STR-SHelo, até então não utilizado no contexto de LCD.

A partir de uma sequência de locais de consulta, nosso método seleciona sequências de locais candidatas verificando apenas a semelhança dos representantes das sequências comparadas. A partir das sequências de locais candidatas selecionadas, juntamente com a estratégia de consistência temporal, é decidido pela sequência de locais do mapa com maior probabilidade de conter fechamentos de *loop*. Com o espaço de busca reduzido, usamos novamente o conceito de sequência para verificar os fechamentos de *loop* entre as imagens de ambas as sequências.

Para avaliar a capacidade em detectar *loops* de nosso método, o executamos em dois *datasets* que apresentam mudanças de ponto de vista, movimentação de pessoas, mudanças significativas de iluminação e resolução de imagens. Para tal, foram propostos três cenários de testes que abordam diferentes propriedades de um método de LCD. No primeiro cenário utilizamos as travessias originais dos datasets onde pudemos verificar a precisão de nosso método se comparado ao OpenSeqSLAM em trajetórias com diferença de iluminação e dinaminicidade no ambiente. Nesse teste foi possível verificar que os resultados do método OpenSeqSLAM são sensíveis à mudança de ponto de vista enquanto que nosso método é mais robusto a tais variações em ambos os datasets, para todas as trajetórias testadas.

O segundo teste simulou a inexistência de *loop* em parte das trajetórias. Novamente nosso método apresentou resultados superiores ao OpenSeqSLAM, pois além de selecionar mais *matchings* corretos, também estimou com maior corretude o % de certeza desses *matchings*. Esses resultados representam um indicativo de que nosso método tem

a capacidade de identificar casos em que não há um fechamento de *loop*.

No último teste foi verificado a precisão dos métodos de LCD no caso de trajetórias divergentes. Ambos os métodos tiveram redução da quantidade de *matchings* corretos assim como nos valores de *precision* e *recall* se comparados ao demais cenários testados. Apesar dessa redução, nosso método se mostrou capaz de conseguir detectar a maioria dos *matchings* corretamente ao contrário do método OpenSeqSLAM para ambos os datasets em todas as trajetórias. Verificamos que os valores de *precision recall* do *dataset* UofAlberta são ínfimos para o método OpenSeqSLAM. Nesse quesito, nosso método se comportou melhor que o OpenSeqSLAM, no entanto também teve uma redução expressiva se comparada aos demais cenários. Esse fato ocorreu em consequência do cálculo do % de certeza do *matching* incorreto para uma das sequências escolhidas. Apesar do nosso método ter realizado o mesmo cálculo corretamente para todas as demais sequências de todos os datasets em todos os cenários propostos, entendemos que esse erro isolado traz a importância em analizarmos outras estratégias para o cálculo do % de certeza do *matching*. Atualmente consideramos a média das similaridades entre as imagens da sequência de busca e da sequência de imagens selecionada. Talvez seja interessante incorporar à esse cálculo as decisões tomadas nos passos anteriores à escolha da sequência de imagens.

Os resultados de todos os cenários são ainda mais expressivos para nosso método quando consideramos que eles foram obtidos em um espaço de busca reduzido e em um número menor de buscas por fechamento de *loop* se comparado ao utilizado pelo OpenSeqSLAM. Portanto, os resultados de nossos experimentos mostram que o nosso método é uma abordagem interessante para a detecção de fechamento de *loop* a longo prazo.

REFERÊNCIAS

ANGELI, A. et al. Incremental vision-based topological SLAM. In: **2008 IEEE/RSJ International Conference on Intelligent Robots and Systems**. IEEE, 2008. Disponível em: <https://doi.org/10.1109/iros.2008.4650675>.

ARROYO, R. et al. Towards life-long visual localization using an efficient matching of binary sequences from images. In: **2015 IEEE International Conference on Robotics and Automation (ICRA)**. IEEE, 2015. Disponível em: <https://doi.org/10.1109/icra.2015.7140088>.

BAILEY, T.; DURRANT-WHYTE, H. Simultaneous localization and mapping (SLAM): part II. **IEEE Robotics & Automation Magazine**, Institute of Electrical and Electronics Engineers (IEEE), v. 13, n. 3, p. 108–117, set. 2006. Disponível em: <https://doi.org/10.1109/mra.2006.1678144>.

BARGAR, W. L. Robots in orthopaedic surgery. **Clinical Orthopaedics and Related Research**, Ovid Technologies (Wolters Kluwer Health), PAP, jul. 2007. Disponível em: <https://doi.org/10.1097/blo.0b013e318146874f>.

BAY, H.; TUYTELAARS, T.; GOOL, L. V. SURF: Speeded up robust features. In: **Computer Vision – ECCV 2006**. Springer Berlin Heidelberg, 2006. p. 404–417. Disponível em: <https://doi.org/10.1007/11744023_32>.

BHATTACHARJEE, S. D. et al. Query adaptive multiview object instance search and localization using sketches. **IEEE Transactions on Multimedia**, Institute of Electrical and Electronics Engineers (IEEE), v. 20, n. 10, p. 2761–2773, out. 2018. Disponível em: <https://doi.org/10.1109/tmm.2018.2814338>.

BORENSTEIN, J. et al. Where am i? sensors and methods for mobile robot positioning. **University of Michigan**, Citeseer, v. 119, n. 120, p. 27, 1996.

BOSCH, A.; ZISSERMAN, A.; MUNOZ, X. Representing shape with a spatial pyramid kernel. In: ACM. **Inter. Conf. on Image and video retrieval**. [S.l.], 2007. p. 401–408.

BRESSON, G. et al. Simultaneous localization and mapping: A survey of current trends in autonomous driving. **IEEE Transactions on Intelligent Vehicles**, Institute of Electrical and Electronics Engineers (IEEE), v. 2, n. 3, p. 194–220, set. 2017. Disponível em: <https://doi.org/10.1109/tiv.2017.2749181>.

BROADBENT, E.; STAFFORD, R.; MACDONALD, B. Acceptance of healthcare robots for the older population: Review and future directions. **International Journal of Social Robotics**, v. 1, n. 4, p. 319, Oct 2009. ISSN 1875-4805. Disponível em: <https://doi.org/10.1007/s12369-009-0030-6>.

CADENA, C. et al. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. **IEEE Transactions on robotics**, IEEE, v. 32, n. 6, p. 1309–1332, 2016.

CHATZICHRISTOFIS, S. A.; ZAGORIS, K.; ARAMPATZIS, A. Bag-of-visual-words vs global image descriptors on two-stage multimodal retrieval. In: **Proceedings of the 34th international ACM SIGIR conference on Research**

and development in Information - SIGIR. ACM Press, 2011. Disponível em: <https://doi.org/10.1145/2009916.2010144>.

CHOW, C.; LIU, C. Approximating discrete probability distributions with dependence trees. **IEEE Transactions on Information Theory**, Institute of Electrical and Electronics Engineers (IEEE), v. 14, n. 3, p. 462–467, maio 1968. Disponível em: <https://doi.org/10.1109/tit.1968.1054142>.

CUMMINS, M.; NEWMAN, P. FAB-MAP: Probabilistic localization and mapping in the space of appearance. **The International Journal of Robotics Research**, SAGE Publications, v. 27, n. 6, p. 647–665, jun. 2008. Disponível em: <https://doi.org/10.1177/0278364908090961>.

DALAL, N.; TRIGGS, B. Histograms of oriented gradients for human detection. In: **2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR05)**. IEEE, 2005. Disponível em: <https://doi.org/10.1109/cvpr.2005.177>.

DELLAERT, F. **Factor graphs and GTSAM: A hands-on introduction**. [S.l.], 2012.

DONG, X. et al. A two-layered pipeline for topological place recognition based on topic model. In: **Communications in Computer and Information Science**. Springer Singapore, 2017. p. 430–441. Disponível em: <https://doi.org/10.1007/978-981-10-7305-2_37>.

DURRANT-WHYTE, H.; BAILEY, T. Simultaneous localization and mapping: part i. **IEEE Robotics & Automation Magazine**, Institute of Electrical and Electronics Engineers (IEEE), v. 13, n. 2, p. 99–110, jun. 2006. Disponível em: <https://doi.org/10.1109/mra.2006.1638022>.

FORD, M. **Rise of the Robots: Technology and the Threat of a Jobless Future**. [S.l.]: Basic Books, 2015.

FOX, D.; BURGARD, W.; THRUN, S. Active markov localization for mobile robots. **Robotics and Autonomous Systems**, Elsevier, v. 25, n. 3-4, p. 195–207, 1998.

GARCIA-FIDALGO, E.; ORTIZ, A. On the use of binary feature descriptors for loop closure detection. In: **Proceedings of the 2014 IEEE Emerging Technology and Factory Automation (ETFA)**. IEEE, 2014. Disponível em: <https://doi.org/10.1109/etfa.2014.7005121>.

GARCIA-FIDALGO, E.; ORTIZ, A. Vision-based topological mapping and localization methods: A survey. **Robotics and Autonomous Systems**, Elsevier BV, v. 64, p. 1–20, fev. 2015. Disponível em: <https://doi.org/10.1016/j.robot.2014.11.009>.

GARCIA-FIDALGO, E.; ORTIZ, A. Hierarchical place recognition for topological mapping. **IEEE Transactions on Robotics**, Institute of Electrical and Electronics Engineers (IEEE), v. 33, n. 5, p. 1061–1074, out. 2017. Disponível em: <https://doi.org/10.1109/tro.2017.2704598>.

GARCIA-FIDALGO, E.; ORTIZ, A. **Methods for Appearance-based Loop Closure Detection**. Springer International Publishing, 2018. Disponível em: <https://doi.org/10.1007/978-3-319-75993-7>.

GONZALEZ, R. C.; WOODS, R. E.; MASTERS, B. R. **Digital Image Processing, Third Edition**. SPIE-Intl Soc Optical Eng, 2009. v. 14. 029901 p. Disponível em: <https://doi.org/10.1117/1.3115362>.

GRISETTI, G. et al. A tutorial on graph-based slam. **IEEE Intelligent Transportation Systems Magazine**, IEEE, v. 2, n. 4, p. 31–43, 2010.

HARRIS, C. G.; STEPHENS, M. et al. A combined corner and edge detector. In: CITESEER. **Alvey vision conference**. [S.l.], 1988. v. 15, n. 50, p. 10–5244.

KEJRIWAL, N.; KUMAR, S.; SHIBATA, T. High performance loop closure detection using bag of word pairs. **Robotics and Autonomous Systems**, Elsevier BV, v. 77, p. 55–65, mar. 2016. Disponível em: <https://doi.org/10.1016/j.robot.2015.12.003>.

KHAN, S.; WOLLHERR, D. Ibuild: Incremental bag of binary words for appearance based loop closure detection. In: IEEE. **Inter. Conf. on Robotics and Automation (ICRA)**. [S.l.], 2015. p. 5441–5447.

KONOLIGE, K.; AGRAWAL, M. FrameSLAM: From bundle adjustment to real-time visual mapping. **IEEE Transactions on Robotics**, Institute of Electrical and Electronics Engineers (IEEE), v. 24, n. 5, p. 1066–1077, out. 2008. Disponível em: <https://doi.org/10.1109/tro.2008.2004832>.

KÜMMERLE, R. et al. g 2 o: A general framework for graph optimization. In: IEEE. **2011 IEEE International Conference on Robotics and Automation**. [S.l.], 2011. p. 3607–3613.

LEE, J. H. et al. Outdoor place recognition in urban environments using straight lines. In: **2014 IEEE International Conference on Robotics and Automation (ICRA)**. IEEE, 2014. Disponível em: <https://doi.org/10.1109/icra.2014.6907675>.

LEE, J. H. et al. Place recognition using straight lines for vision-based SLAM. In: **2013 IEEE International Conference on Robotics and Automation**. IEEE, 2013. Disponível em: <https://doi.org/10.1109/icra.2013.6631111>.

LIMA, P. U. Search and rescue robots: The civil protection teams of the future. In: **2012 Third International Conference on Emerging Security Technologies**. IEEE, 2012. Disponível em: <https://doi.org/10.1109/est.2012.40>.

LIU, Y.; ZHANG, H. Visual loop closure detection with a compact image descriptor. In: **2012 IEEE/RSJ International Conference on Intelligent Robots and Systems**. IEEE, 2012. Disponível em: <https://doi.org/10.1109/iros.2012.6386145>.

LOWE, D. G. Distinctive image features from scale-invariant keypoints. **International journal of computer vision**, Springer, v. 60, n. 2, p. 91–110, 2004.

LOWRY, S. et al. Visual place recognition: A survey. **IEEE Transactions on Robotics**, Institute of Electrical and Electronics Engineers (IEEE), v. 32, n. 1, p. 1–19, fev. 2016. Disponível em: <https://doi.org/10.1109/tro.2015.2496823>.

MAIER, D.; KLEINER, A. Improved GPS sensor model for mobile robots in urban terrain. In: **2010 IEEE International Conference on Robotics and Automation**. IEEE, 2010. Disponível em: <https://doi.org/10.1109/robot.2010.5509895>.

MAKARENKO, A. et al. An experiment in integrated exploration. In: **IEEE/RSJ International Conference on Intelligent Robots and System**. IEEE, 2002. Disponível em: <https://doi.org/10.1109/irds.2002.1041445>.

MILFORD, M. J.; WYETH, G. F. SeqSLAM: Visual route-based navigation for sunny summer days and stormy winter nights. In: **2012 IEEE International Conference on Robotics and Automation**. IEEE, 2012. Disponível em: <https://doi.org/10.1109/icra.2012.6224623>.

NEWMAN, P.; HO, K. Slam-loop closing with visually salient features. In: IEEE. **proceedings of the 2005 IEEE International Conference on Robotics and Automation**. [S.l.], 2005. p. 635–642.

NUSKE, S.; ROBERTS, J.; WYETH, G. Robust outdoor visual localization using a three-dimensional-edge map. **Journal of Field Robotics**, Wiley, v. 26, n. 9, p. 728–756, set. 2009. Disponível em: <https://doi.org/10.1002/rob.20306>.

OLIVA, A.; TORRALBA, A. **International Journal of Computer Vision**, Springer Nature, v. 42, n. 3, p. 145–175, 2001. Disponível em: <https://doi.org/10.1023/a:1011139631724>.

ROMERO, R. A. F. et al. **Robótica móvel**. [S.l.]: LTC, 2014.

ROSTEN, E.; DRUMMOND, T. Machine learning for high-speed corner detection. In: SPRINGER. **European conference on computer vision**. [S.l.], 2006. p. 430–443.

RUBLEE, E. et al. Orb: An efficient alternative to sift or surf. In: CITESEER. **ICCV**. [S.l.], 2011. v. 11, n. 1, p. 2.

SAAVEDRA, J. M. RST-SHELO: sketch-based image retrieval using sketch tokens and square root normalization. **Multimedia Tools and Applications**, Springer Nature, v. 76, n. 1, p. 931–951, nov. 2015. Disponível em: <https://doi.org/10.1007/s11042-015-3076-5>.

SHAPLEY, R. M.; TOLHURST, D. J. Edge detectors in human vision. **The Journal of Physiology**, Wiley, v. 229, n. 1, p. 165–183, fev. 1973. Disponível em: <https://doi.org/10.1113/jphysiol.1973.sp010133>.

SIAM, S. M.; ZHANG, H. Fast-SeqSLAM: A fast appearance based place recognition algorithm. In: **2017 IEEE International Conference on Robotics and Automation (ICRA)**. IEEE, 2017. Disponível em: <https://doi.org/10.1109/icra.2017.7989671>.

SIEGWART, R.; NOURBAKHSH, I. R. **Introduction to Autonomous Mobile Robots**. Scituate, MA, USA: Bradford Company, 2004. ISBN 026219502X.

SIVIC, J.; ZISSERMAN, A. Video google: A text retrieval approach to object matching in videos. In: IEEE. **null**. [S.l.], 2003. p. 1470.

SÜNDERHAUF, N.; NEUBERT, P.; PROTZEL, P. Are we there yet? challenging seqslam on a 3000 km journey across all four seasons. In: CITESEER. **Inter. Conf. on Robotics and Automation (ICRA)**. [S.l.], 2013. p. 2013.

SUNDERHAUF, N.; PROTZEL, P. BRIEF-gist - closing the loop by simple means. In: **2011 IEEE/RSJ International Conference on Intelligent Robots and Systems**. IEEE, 2011. Disponível em: <https://doi.org/10.1109/iros.2011.6094921>.

SÜNDERHAUF, N.; PROTZEL, P. Towards a robust back-end for pose graph slam. In: IEEE. **2012 IEEE International Conference on Robotics and Automation**. [S.l.], 2012. p. 1254–1261.

SZELISKI, R. **Computer Vision**. Springer London, 2011. Disponível em: <https://doi.org/10.1007/978-1-84882-935-0>.

THRUN, S. Exploring artificial intelligence in the new millennium. In: LAKEMEYER, G.; NEBEL, B. (Ed.). San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2003. cap. Robotic Mapping: A Survey, p. 1–35. ISBN 1-55860-811-7. Disponível em: <http://dl.acm.org/citation.cfm?id=779343.779345>.

THRUN, S.; BURGARD, W.; FOX, D. **Probabilistic robotics**. [S.l.]: MIT press, 2005.

TSINTOTAS, K. A.; BAMPIS, L.; GASTERATOS, A. Assigning visual words to places for loop closure detection. In: **2018 IEEE International Conference on Robotics and Automation (ICRA)**. IEEE, 2018. Disponível em: <https://doi.org/10.1109/icra.2018.8461146>.

WU, J.; REHG, J. M. CENTRIST: A visual descriptor for scene categorization. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, Institute of Electrical and Electronics Engineers (IEEE), v. 33, n. 8, p. 1489–1501, ago. 2011. Disponível em: <https://doi.org/10.1109/tpami.2010.224>.

YIANILOS, P. N. Data structures and algorithms for nearest neighbor search in general metric spaces. In: **Symposium on Discrete Algorithms (SODA)**. [S.l.]: ACM-SIAM, 1993. v. 93, n. 194, p. 311–321.