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

RENATA DAS CHAGAS NEULAND

Análise de Intervalos e Restrições Temporais Aplicadas ao Problema de Reconhecimento de Regiões

Tese apresentada como requisito parcial para a obtenção do grau de Doutor em Ciência da Computação

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

Porto Alegre
2021

CIP — CATALOGAÇÃO NA PUBLICAÇÃO

Neuland, Renata das Chagas

Análise de Intervalos e Restrições Temporais Aplicadas ao Problema de Reconhecimento de Regiões / Renata das Chagas Neuland. – Porto Alegre: PPGC da UFRGS, 2021.

93 f.: il.

Tese (doutorado) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR–RS, 2021. Orientador: Edson Prestes e Silva Jr.; Co-orientador: Mariana Luderitz Kolberg.

1. Robótica. 2. Reconhecimento de Regiões. 3. Localização. 4. Análise de Intervalos. I. Silva Jr., Edson Prestes e. II. Kolberg, Mariana Luderitz. III. Título.

“Decisão nunca foi meu forte.”

— RODRIGUES F., 2019

Definitivamente não é o meu também.

AGRADECIMENTOS

À família. Agradeço aos meus pais, que pacientemente me guiaram até aqui, garantindo que eu tivesse todo o carinho e suporte necessários para vencer os desafios em meu caminho. Agradeço pela motivação e consolo nas horas mais difíceis, pelas alegrias compartilhadas mesmo que a distância e pela força em tantos momentos que a vontade de desistir apareceu.

À família POA. Agradeço a todos os membros da família que criamos em POA, amigos que me apoiaram e ajudaram, mesmo que nem sempre concordando com as minhas escolhas. Cristina Otto e Rodrigo Ruas agradeço por tantos anos que dividimos não apenas o apartamento, mas também sonhos, alegrias e conquistas. Agradeço a Fernanda Rodrigues, que me ajudou a superar muitos dias cinzas com bolo, chá de maçã e caminhadas para liberar nosso otimismo latente e bom humor peculiar. Agradeço aos amigos Thiago Rodrigues, Joice Bastos e Lucas Muller que também compõem essa família e me ajudam a levar a vida de forma mais leve e divertida.

Aos meus amigos. Agradeço aos meus amigos e colegas de laboratório pelas conversas e companheirismo, por toda a ajuda e colaboração no desenvolvimento do trabalho. Mas principalmente pela parceria e companhia, sempre vou lembrar daqueles almoços comemorativos que duravam horas e horas de muita comilança.

Aos orientadores. Professores Edson Prestes e Mariana Kolberg que me auxiliaram nessa caminhada. Agradeço pela motivação e dedicação constantes, pela paciência e incentivo nessa caminhada. Muito obrigada por terem me ajudado não só em questões acadêmicas, mas também no crescimento pessoal.

Obrigada a todos!

RESUMO

A área da robótica vem apresentando grandes avanços nas últimas décadas e está se tornando comum no cotidiano de muitas pessoas. Robôs autônomos, desde pequenos aspiradores de pó até carros que não exigem um motorista humano, são exemplos de como a robótica está facilitando tarefas corriqueiras. Para que robôs sejam realmente autônomos eles precisam perceber o ambiente a sua volta assim como sua posição no mundo. Ao encontro dessa necessidade, um dos desafios da robótica móvel é o Problema de Reconhecimento de Regiões. Esse problema pode ser definido como: dada uma observação do ambiente, decidir se essa observação vem de um lugar previamente visitado ou não. Essa informação contribui para a redução da incerteza sobre a localização do robô e sobre o mapa do ambiente no qual está inserido. O problema de Reconhecimento Visual de Regiões é uma variação do problema original que surgiu após a popularização do uso de câmeras como principal fonte de informação. Essa popularização se deve principalmente pelo baixo custo do sensor e pela riqueza das observações obtidas. Em contrapartida, o uso de câmeras trouxe novos desafios relacionados ao tratamento de imagens. Neste trabalho propomos uma nova abordagem para tratar o problema de reconhecimento visual de regiões em ambientes sujeitos à mudanças de iluminação, alterações decorrentes da passagem das estações de ano ou mesmo consequentes das adversidades climáticas. Nossa abordagem é fundamentada em teoria de Análise de Intervalos, sendo o mundo conhecido modelado como um conjunto de intervalos, e através de restrições do problema reduzimos o espaço de busca de soluções usando operações intervalares. A abordagem proposta foi testada usando três *datasets* públicos, comumente utilizados em outros trabalhos da área, e apresentou resultados promissores. As comparações, apresentadas através de *precision-recall*, mostram que o método proposto tem resultados competitivos em relação ao estado da arte, além de mostrar potencial para uso em operações em tempo real.

Palavras-chave: Robótica. Reconhecimento de Regiões. Localização. Análise de Intervalos.

Interval Approach Based On Temporal Restrictions Applied To The Place Recognition Problem

ABSTRACT

The robotics field has been making great strides in recent decades, and it is becoming common in many people's daily lives. Autonomous robots from small vacuum cleaners to cars that do not require a human driver are examples of how the robotics is making everyday tasks easier. For robots to be truly autonomous, they need to perceive the environment around them as well as their position in the world. In this regard, one of the challenges of mobile robotics is the Place Recognition Problem. The problem is defined by identifying whether a new observation of the environment comes from a place previously visited or not. This information contributes to reduce the uncertainty about the robot localization and improve the quality of the map that is being built. Given the growing popularity of the cameras as a primary source of information, a new problem called Visual Place Recognition arose; and with it, all the challenges of dealing with images. This popularization is mainly due to the low cost of the sensor and the richness of the observations obtained. We propose a novel approach to deal with the visual place recognition problem, considering that the environment is subjected to illumination and sesoning changes, or even climatic adversities. It is based on Interval Analysis theory to model the known world as a set of intervals and the problem constraints to reduce the solution search space through interval operations. Our proposal was tested with public datasets and has presented promising results. The comparisons between the proposed method and the methods in the literature show that our method produces competitive results. In addition, it shows potential for use in real-time operations.

Keywords: Robotics, Place Recognition, SLAM, Interval Analysis.

LISTA DE ABREVIATURAS E SIGLAS

BLOB	Binary Large OBjects
BoW	Bag-of-Words
BRIEF	Binary Robust Independent Elementary Features
FAB-MAP	Fast Appearance-Based Mapping
GPW	Garden Points Walking
HOG	Histogram of Oriented Gradients
LDB	Local Difference Binary
SLAM	Simultaneous Localization and Mapping
UofA	University of Alberta

LISTA DE FIGURAS

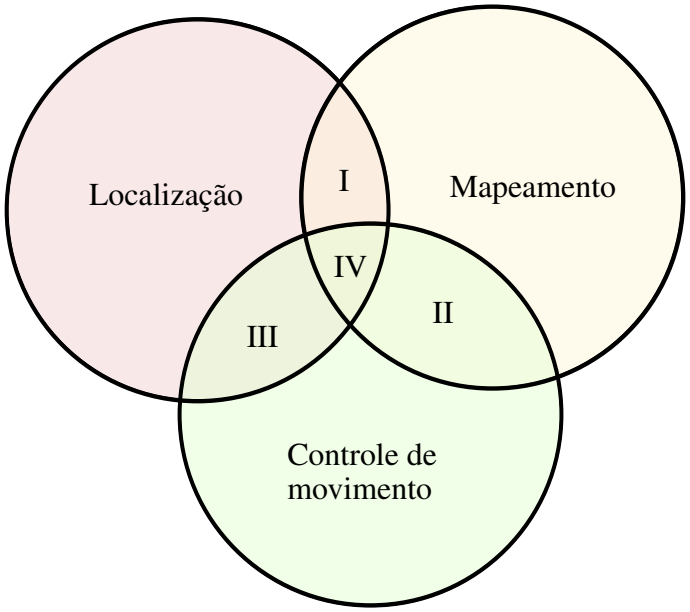
LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

A criação de máquinas autônomas capazes de operar sem supervisão humana é um dos grandes desafios da robótica. O uso de robôs já ultrapassou o limite de ambientes controlados e está cada vez mais presente no mundo real, auxiliando ou mesmo substituindo humanos em tarefas repetitivas, difíceis ou perigosas (BISHOP, 2000) (JAULIN, 2015). Ser capaz de extrair informações sobre o ambiente no qual está inserido, e usá-las na tomada de decisão, é fundamental para que um robô consiga executar diversas tarefas de forma bem sucedida (THRUN et al., 2005) (SIEGWART; NOURBAKHSH, 2004).

Figura 1.1: Intersecção entre problemas clássicos da robótica móvel



(I) SLAM, (II) exploração clássica, (III) localização ativa, (IV) exploração integrada (MAKARENKO et al., 2002).

Em particular, os problemas da robótica móvel podem ser divididos amplamente em três grandes áreas, são elas, localização, mapeamento e controle de movimento (MAKARENKO et al., 2002). Problemas de localização são aqueles que buscam determinar a localização do robô em uma mapa do ambiente através das informações provenientes de sensores (DELLAERT et al., 1999). Problemas de mapeamento são aqueles relacionados com a criação de modelos espaciais de ambientes físicos usando robôs (THRUN et al., 2002). E problemas de controle de movimento são focados na execução de estratégias de movimentação do robô de forma segura, confortável, e otimizada (GONZALEZ et al., 2016).

Como apresentado na Figura 1.1, essas três áreas não estão isoladas, e suas intersecções dão origem a novos desafios. Desafios esses que vêm sendo atacados por diferen-

tes abordagens, algumas delas apresentadas pelo *Phi-Robotics Research Group*[1], grupo no qual a autora desse trabalho está inserida. A relação entre localização e mapeamento (I) dá origem ao problema de Localização e Mapeamento Simultâneos (SLAM), onde o robô precisa mapear o ambiente ao mesmo tempo em que se localiza usando o mapa que está sendo criado (MAFFEI et al., 2013). A intersecção entre mapeamento e controle de movimento (II) representa o problema de exploração, onde o robô precisa se deslocar no ambiente a fim de explorar novas regiões ou pontos de interesse que contribuam para a geração do mapa (PRESTES; ENGEL, 2011). Quando as áreas de localização e controle de movimentos estão sobrepostas (III) temos o problema de localização ativa, onde o robô se desloca no ambiente com o propósito de tentar reduzir incertezas sobre sua localização (PRESTES; RITT; FUHR, 2008). Por fim, com a intersecção das três áreas, temos o problema de exploração integrada. Nesse caso, além de se localizar e gerar o mapa do ambiente de forma simultânea, o robô deve explorar o ambiente a fim de reduzir os erros nos processos de localização e mapeamento (MAFFEI et al., 2014; JORGE et al., 2015).

Nessa pesquisa tratamos o problema de Reconhecimento Visual de Regiões, que apesar de estar bem definido na literatura ainda é um desafio em aberto. Além da importância dele por si só, ele está vinculado a outros grandes problemas da robótica, como Localização e mapeamento simultâneos (SLAM) (LOWRY et al., 2016).

1.1 Problema

O problema de reconhecimento visual de regiões, também conhecido como Detecção de Fechamento de *Loop*, consiste em reconhecer regiões previamente visitadas usando sensores visuais (BAMPIS; AMANATIADIS; GASTERATOS, 2017). Ele pode ser abordado de duas formas. Uma, quando a busca de regiões correspondentes é feita entre dois conjuntos disjuntos de imagens, definido então como uma função

$$x \in \mathbb{Q} \rightarrow f(x) \in \mathbb{D}, \tag{1.1}$$

onde $\mathbb{Q}$ é o conjunto de imagens de busca para o qual procuramos correspondência em um conjunto fonte $\mathbb{D}$. Outra, quando a busca é executada no mesmo conjunto de imagens, nesse caso, o mesmo conjunto contém os subconjuntos busca e fonte, que são definidos

[1] http://www.inf.ufrgs.br/phi-group/officialSite/

por uma variável de tempo t,

$$\mathbb{Q}_t = \{x_t\} \tag{1.2}$$

$$\mathbb{D}_t = \{x_i \mid i < t - \Delta t\}. \tag{1.3}$$

Onde $\Delta t > 0$ define a diferença mínima de tempo entre as imagens dos conjuntos fonte e busca para que não ocorra correspondência entre imagens recém visitadas (CIESLEWSKI et al., 2016).

Sistemas para reconhecimento visual de regiões precisam ter uma representação do ambiente conhecido através de um mapa, de forma que esse possa ser comparado com novas observações do ambiente. Além disso, o sistema precisa ser capaz de definir o grau de certeza sobre a origem de uma nova observação, identificando se ela provém de um lugar conhecido ou não. E, caso seja conhecido, apontá-lo no mapa (LOWRY et al., 2016).

Existem três principais desafios envolvendo o reconhecimento de regiões (LOWRY et al., 2016). O primeiro está relacionado com a mudança na aparência dos lugares. Em operações de longo prazo o ambiente pode sofrer alterações drásticas na aparência, as quais podem ocorrer devido à mudança de iluminação, clima e horário do dia. Diferenças entre duas observações de um mesmo lugar podem ser tão significativas que se torna difícil identificar as semelhanças. Um exemplo de alteração, apresentado na Figura 1.2, é a mudança de iluminação do dia para a noite, neste caso, pontos de luz antes apagados podem se tornar pontos brilhantes, ou a textura de uma árvore pode se tornar um borrão escuro uniforme (MILFORD; WYETH, 2012; SÜNDERHAUF; NEUBERT; PROTZEL, 2013).

Figura 1.2: Observações de um mesmo local extraídas em diferentes ocasiões

Fonte: Milford e Wyeth (2012).

Um segundo desafio é identificar lugares como sendo diferentes mesmo que eles tenham uma aparência similar. A Figura 1.3 ilustra um exemplo, onde é preciso verificar um conjunto de detalhes para perceber que as imagens não provêm de um mesmo lugar.

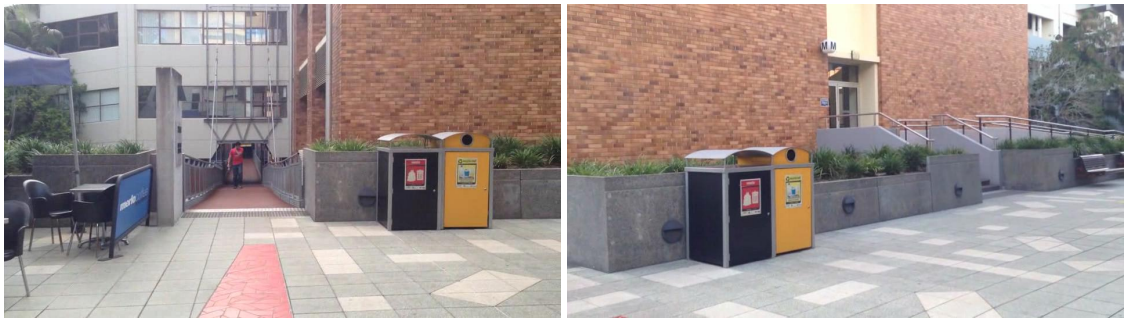
Por fim, também existe o desafio de reconhecer um mesmo lugar quando obser-

Figura 1.3: Observações semelhantes extraídas em diferentes locais

Fonte: Milford e Wyeth (2012).

vado de pontos de vista diferentes dos conhecidos, como mostra a Figura 1.4. Com a mudança no ponto de vista, além de objetos em comum serem observados a partir de diferentes ângulos, nova informação não compartilhada entre as observações também pode aparecer.

Figura 1.4: Mesmo local visto de diferentes pontos de vista



Fonte: Glover (2014).

1.2 Objetivo

Como apresentado na seção anterior, tratar o problema de reconhecimento visual de regiões envolve desafios como perceber a mudança na aparência do ambiente, reconhecer que regiões são diferentes mesmo que tenham aparência similar e reconhecer mesmas regiões a partir de diferentes pontos de vista. Lidar com esse problema se torna ainda mais desafiador quando a entrada de dados são amostras do mundo que não possuem quaisquer coordenadas de referência ao local de coleta.

Até o presente momento, o uso de Análise de Intervalos mostrou resultados positivos no tratamento de problemas da robótica (JAULIN et al., 2001; MERLET, 2010; NEULAND et al., 2014a; NEULAND et al., 2014b; NEULAND et al., 2020). Contudo, o uso de intervalos é naturalmente associado ao espaço de trabalho em que o robô está inserido, comumente o espaço euclidiano. Os intervalos costumam representar partes desse espaço, sendo ancorados em um dado sistema de coordenadas. O que torna o uso de aná-

lise de intervalos para a criação de uma modelagem para o problema de reconhecimento visual de regiões ainda mais desafiador, pois esse tipo de informação frequentemente não está disponível.

O objetivo deste trabalho é tentar responder a seguinte pergunta:

> É possível modelar o problema de reconhecimento visual de regiões de forma a tratá-lo utilizando técnicas de análise de intervalos, onde a informação disponível não está ancorada em um sistema de coordenadas do mundo de trabalho do robô?

A partir do questionamento anterior, propomos a criação de uma metodologia para reconhecimento de regiões baseado em análise de intervalos, onde a única informação de entrada é uma sequência de amostras do mundo explorado, sem coordenadas de localização. Para tanto, propomos:

- uma nova modelagem para o problema onde o mundo conhecido, nesse caso, um conjunto de amostras, seja representado de forma adequada para o uso de técnicas intervalares.
- otimizar a busca por correspondências usando sempre que possível operações sobre intervalos ao invés de observações, simplificando a análise de informações mais complexas. Com isso, reduzindo o tempo de processamento na busca por soluções.

A proposta foi avaliada a partir da execução de experimentos em *datasets* com significativas mudanças perceptuais a fim de demonstrar os benefícios que a análise de intervalos pode proporcionar mesmo em situações de uso não óbvias.

1.3 Organização

Esse trabalho está organizado como segue. O Capítulo 2 exibe uma introdução sobre a teoria de Análise de Intervalos e algumas operações consideradas mais relevantes para o bom entendimento do método que será apresentado a seguir, além disso, apresenta uma visão breve do uso de análise de intervalos na robótica. Definições sobre sistemas para reconhecimento de regiões, desafios e estratégias de solução que marcam o estado da arte, são apresentados no Capítulo 3.

O Capítulo 4 apresenta a modelagem do mundo observado usando teoria de análise de intervalos para a representação de observações que podem não ser métricas, nem ancoradas em um sistema de coordenadas do ambiente. A eficácia dessa modelagem é testada através do uso da mesma para o tratamento do problema de reconhecimento visual de regiões, como descrito no Capítulo 5.

O Capítulo 6 detalha os experimentos e resultados obtidos com a aplicação da modelagem proposta para a tarefa de reconhecimento visual de regiões usando *datasets* com significativas mudanças perceptuais. Por fim, as conclusões e propostas para trabalhos futuros estão no Capítulo 7.

2 CONCEITOS BÁSICOS SOBRE ANÁLISE DE INTERVALOS

Neste capítulo são abordados alguns conceitos relacionados à análise de intervalos, os quais são fundamentais para bom entendimento da modelagem proposta no Capítulo 4 e sua aplicação apresentada no Capítulo 5. As definições apresentadas são relacionadas à aritmética intervalar clássica desenvolvida por Moore (1966).

2.1 Definições Básicas

A ideia de conter números em intervalos é a base da análise de intervalos (JAULIN et al., 2001). Considere, por exemplo, a equação

$$x^2 - 2 = 0. \tag{2.1}$$

Ela tem como solução positiva $\sqrt{2} = 1.414213...$, a qual não tem representação com um número finito de dígitos (MOORE; KEARFOTT; CLOUD, 2009), mas que pode ser representada por um intervalo fechado que contenha esse valor.

Um intervalo fechado, denotado por $[\underline{x}; \overline{x}]$, é um conjunto de números reais onde

$$[\underline{x}; \overline{x}] = \{x \in \mathbb{R} \mid \underline{x} \leq x \leq \overline{x}\} \tag{2.2}$$

e $\underline{x}$ é chamado de ínfimo enquanto $\overline{x}$ é chamado de supremo de $[x]$. O conjunto de todos intervalos possíveis de $\mathbb{R}$ é definido por $\mathbb{IR}$, assim, $[x] \in \mathbb{IR}$.

Dois intervalos são considerados iguais quando representam o mesmo conjunto. Isso acontece quando:

$$[\underline{x}; \overline{x}] = [\underline{y}; \overline{y}] \text{ se } \underline{x} = \underline{y} \text{ e } \overline{x} = \overline{y}. \tag{2.3}$$

Quando um intervalo tem seu ínfimo e supremo iguais ele é chamado *degenerado* (ou pontual). Intervalos degenerados contêm apenas um número, isto é, é equivalente a um escalar. Por exemplo:

$$0 = [0; 0]. \tag{2.4}$$

Usando computação intervalar é possível tratar vários problemas matemáticos, obtendo soluções que garantidamente contêm a resposta exata para o problema de estudo.

Incertezas podem ser representadas como intervalos, e quando esses valores são corretamente modelados a análise de intervalos traz garantias sobre a solução.

2.2 Operações

É possível operar sobre intervalos usando operações semelhantes àquelas realizadas no sistema numérico real. Para isso, é essencial entender que operações com intervalos são operações com conjuntos. Considere os intervalos $[x]$ e $[y]$, os conjuntos resultantes das **operações aritméticas básicas** $(+, -, \cdot, /)$ entre esses intervalos são definidos por (JAULIN et al., 2001):

$$\begin{aligned}
[x] + [y] &= \{x + y \mid x \in [x], y \in [y]\} \\
[x] - [y] &= \{x - y \mid x \in [x], y \in [y]\} \\
[x] \ \cdot \ [y] &= \{xy \mid x \in [x], y \in [y]\} \\
[x] \ / \ [y] &= \{x \ / \ y \mid x \in [x], y \in [y]\}.
\end{aligned} \tag{2.5}$$

Sendo que para a operação de divisão, $0 \notin [y]$. Para computar essas operações não é necessário tratar todo o conjunto. Usando o ínfimo e supremo é possível calcular o intervalo resultante dessas operações (JAULIN et al., 2001), como apresentado a seguir:

$$\begin{aligned}
[x] + [y] &= [\underline{x} + \underline{y}; \overline{x} + \overline{y}] \\
[x] - [y] &= [\underline{x} - \overline{y}; \overline{x} - \underline{y}] \\
[x] \ \cdot \ [y] &= [\min(\underline{x}\underline{y}, \underline{x}\overline{y}, \overline{x}\underline{y}, \overline{x}\overline{y}); \max(\underline{x}\underline{y}, \underline{x}\overline{y}, \overline{x}\underline{y}, \overline{x}\overline{y})] \\
[x] \ / \ [y] &= [\min(\underline{x}/\underline{y}, \underline{x}/\overline{y}, \overline{x}/\underline{y}, \overline{x}/\overline{y}); \max(\underline{x}/\underline{y}, \underline{x}/\overline{y}, \overline{x}/\underline{y}, \overline{x}/\overline{y})].
\end{aligned} \tag{2.6}$$

Para exemplificar as operações acima considere os intervalos $[x] = [1; 3]$ e $[y] = [2; 5]$:

$$\begin{aligned}
[x] + [y] &= [3; 8] \\
[x] - [y] &= [-4; 1] \\
[x] \ \cdot \ [y] &= [2; 15] \\
[x] \ / \ [y] &= [0.2; 1.5].
\end{aligned} \tag{2.7}$$

Operações comumente associadas a conjuntos são as de **intersecção** e **união**. Sobre a operação de intersecção, considere os intervalos $[x]$ e $[y]$. A solução da intersecção é definida por

$$\begin{aligned}[x] \cap [y] &= \{z \mid z \in [x] \text{ e } z \in [y]\} \\ &= [\max(\underline{x}, \underline{y}); \min(\overline{x}, \overline{y})]. \end{aligned} \tag{2.8}$$

Caso $\overline{y} < \underline{x}$ ou $\overline{x} < \underline{y}$ a intersecção entre os intervalos é vazia e denotada por

$$[x] \cap [y] = \emptyset. \tag{2.9}$$

Sobre a operação de união, onde $[x] \cap [y] \neq \emptyset$, o resultado é um intervalo fechado definido por

$$\begin{aligned}[x] \cup [y] &= \{z \mid z \in [x] \text{ ou } z \in [y]\} \\ &= [\min(\underline{x}, \underline{y}); \max(\overline{x}, \overline{y})]. \end{aligned} \tag{2.10}$$

Contudo, caso $[x] \cap [y] = \emptyset$, a união não resulta em um intervalo. Nesse contexto, a operação de ***hull*** pode ser usada, pois sempre resulta em um intervalo. Ela é definida por

$$[x] \sqcup [y] = [\min(\underline{x}, \underline{y}); \max(\overline{x}, \overline{y})]. \tag{2.11}$$

Sendo que

$$\big([x] \cup [y]\big) \subseteq \big([x] \sqcup [y]\big), \tag{2.12}$$

para quaisquer intervalos $[x]$ e $[y]$.

Considere os intervalos $[x] = [1; 7]$, $[y] = [3; 9]$ e $[z] = [10; 15]$, o resultado das

operações de intersecção, união e *hull* entre eles é:

$$[x] \cap [y] = [3; 7] \tag{2.13}$$

$$[x] \cap [z] = \emptyset \tag{2.14}$$

$$[y] \cap [z] = \emptyset \tag{2.15}$$

$$[x] \cup [y] = [1; 9] \tag{2.16}$$

$$[x] \cup [z] = [1; 7] \cup [10; 15] \tag{2.17}$$

$$[y] \cup [z] = [3; 9] \cup [10; 15] \tag{2.18}$$

$$[x] \sqcup [y] = [1; 9] \tag{2.19}$$

$$[x] \sqcup [z] = [1; 15] \tag{2.20}$$

$$[y] \sqcup [z] = [3; 15]. \tag{2.21}$$

Considerando todas as operações apresentadas, a intersecção ganha destaque no contexto de análise de intervalos. Suponha dois intervalos $[x]$ e $[y]$ contendo o resultado de um dado problema. Sabendo que os dois contêm a solução, a sua intersecção também deve conter. Por exemplo, em um cenário em que dois pesquisadores mediram uma propriedade de um novo material, um dos pesquisadores obteve o valor 103 com um erro menor ou igual 2. O segundo pesquisador mediu 104, também com um erro menor ou igual 2. Podemos representar essas medidas com os intervalos $[x] = [101; 105]$ e $[y] = [102; 106]$, respectivamente. Sabendo que a medida correta está contida nesses intervalos, ela também deve estar em $[x] \cap [y] = [102; 105]$. Com a intersecção entre esses intervalos obtemos um intervalo de solução mais restrito. Caso a intersecção seja vazia, fica claro que ao menos uma das medidas está incorreta (MOORE; KEARFOTT; CLOUD, 2009).

Suponha um conjunto de medidas de tamanho n, todas representando a mesma solução. Mesmo que não haja intersecção entre todos os elementos do conjunto, pode existir intersecção entre alguns elementos dele, como mostra a Figura 2.1. No exemplo da figura, com $n = 6$, não existe uma intersecção entre todos os intervalos, assim o resultado da intersecção entre eles é

$$\bigcap [x]_i = \emptyset \quad \text{for } 1 \leq i \leq n. \tag{2.22}$$

Essa situação é uma ocorrência comum no mundo real. Por exemplo, entre um conjunto de sensores, é esperado que alguns falhem ou sofram interferências externas.

Nesses casos, podemos usar uma intersecção relaxada (MARZULLO, 1984), que fornece como solução a intersecção de apenas um subconjunto dos intervalos existentes. O algoritmo *q-relaxed intersection* (BREFORT et al., 2015) foi proposto para executar essa operação. Para um conjunto de intervalos $[x]_1, [x]_2, ..., [x]_n \in \mathbb{IR}$, a intersecção relaxada pode ser definida como

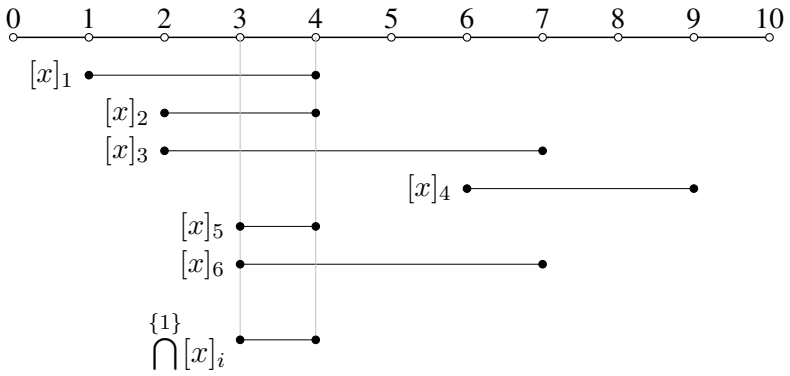
$$\bigcap^{\{q\}}[x]_i \quad \text{for } 1 \leq i \leq n. \tag{2.23}$$

Onde q define quantos intervalos do conjunto podem ser desconsiderados na busca por intersecções. O resultado dessa operação inclui todo x que pertença a pelo menos $(n - q)$ intervalos. Suponha $n = 6$ e $q = 2$, se um dado $x \in \mathbb{R}$ está presente em pelo menos quatro intervalos do conjunto, ele fará parte da solução.

A Figura 2.1 apresenta um exemplo de intersecção relaxada entre os intervalos de um conjunto com $n = 6$ e $q = 1$. Ou seja, o resultado é a intersecção entre pelo menos 5 intervalos do conjunto. Considere os intervalos a seguir:

$$[x]_1 = [1; 4],\ [x]_2 = [2; 4],\ [x]_3 = [2; 7]$$
$$[x]_4 = [6; 9],\ [x]_5 = [3; 4],\ [x]_6 = [3; 7].$$

Figura 2.1: Intersecção relaxada



A intersecção entre os 6 intervalos resulta em um conjunto vazio, entretanto, usando uma intersecção relaxada e variando o valor de q, temos:

$$\bigcap^{\{0\}}[x]_i = \emptyset,\ \bigcap^{\{1\}}[x]_i = [3; 4], \bigcap^{\{2\}}[x]_i = [3; 4],$$

$$\bigcap^{\{3\}}[x]_i = [2; 4] \cup [6; 7],\ \bigcap^{\{4\}}[x]_i = [2; 7],\ \bigcap^{\{5\}}[x]_i = [1; 9],\ \bigcap^{\{6\}}[x]_i = \mathbb{R}$$

2.3 Análise de Intervalos na Robótica

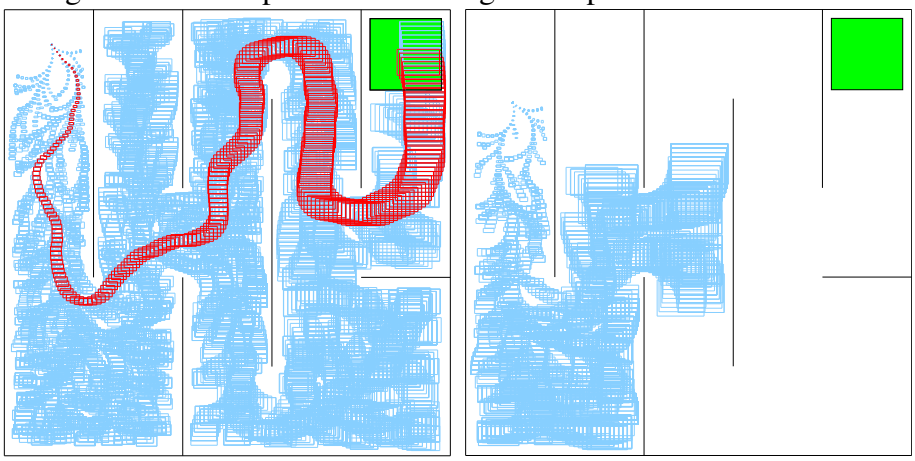
A Análise de Intervalos é uma ferramenta matemática bastante útil para a robótica. Uma característica que ganha destaque é a simplicidade em se lidar com a incerteza dos dados que são inevitáveis na área de robótica (MERLET, 2010), uma vez que as informações usadas são frequentemente derivadas de sensores que além da incerteza inerente, ainda podem falhar por mal funcionamento ou interferências externas. Incertezas essas que podem ser contidas e representadas através de intervalos (JAULIN et al., 2001).

Diferente das técnicas probabilísticas que normalmente precisam de conhecimento *a priori* sobre a distribuição de probabilidade das incertezas, para técnicas intervalares basta apenas saber os limites da incerteza (MUSTAFA et al., 2018). Além disso, técnicas intervalares trazem uma aproximação do conjunto de todas as soluções para um problema. Os resultados são ditos garantidos, i.e., essa aproximação garantidamente contém a solução, supondo a correta modelagem do problema e das incertezas envolvidas (JAULIN et al., 2001).

Diversas abordagens têm sido apresentadas para lidar com problemas da robótica móvel, tais como planejamento de caminhos seguros (PEPY; KIEFFER; WALTER, 2008) (SANDRETTO; CHAPOUTOT; MULLIER, 2017), localização (NEULAND et al., 2014a) (NEULAND et al., 2014b) (ABDALLAH; GNING; BONNIFAIT, 2007) (NEULAND et al., 2020) e SLAM (WANG et al., 2018) (MUSTAFA et al., 2018).

Pepy, Kieffer e Walter (2008) propuseram uma abordagem intervalar para o problema de planejamento de caminhos. O método é garantidamente seguro considerando a correta modelagem da incerteza sobre o veículo, seus sensores e sua configuração inicial. A abordagem é baseada em RRTs (Rapidly-exploring Random Trees) e recebeu o nome de Box-RRT. O método foi testado em ambientes bidimensionais e os resultados mostram que os caminhos encontrados pelo método são inteiramente livres de colisão. Contudo, existem situações em que a incerteza cresce de tal forma que nenhum caminho é considerado seguro. A Figura 2.2 apresenta dois exemplos de execução do método, um concluído com sucesso e outro não, devido ao acúmulo de erros relacionados a odometria.

Figura 2.2: Exemplo de caminhos gerados pelo método Box-RRT



Execução com sucesso. Execução com falha.

Caminhos partindo do canto superior esquerdo em busca do objetivo (quadrado verde) (PEPY; KIEFFER; WALTER, 2008).

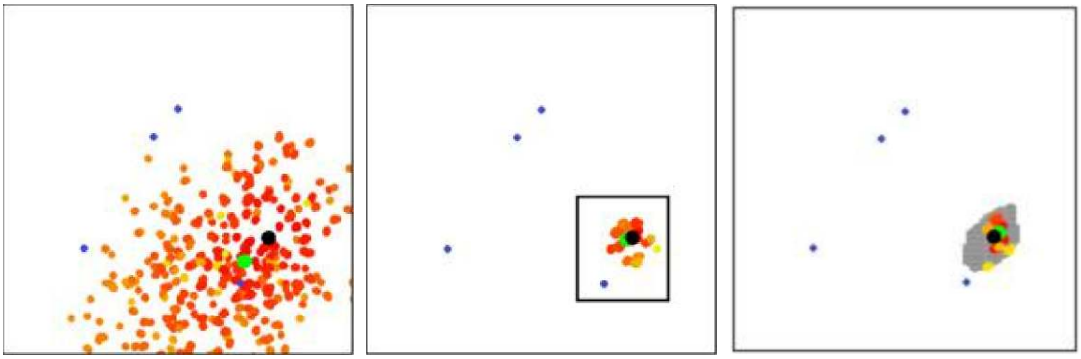
Neuland et al. (2014a), Neuland et al. (2014b) e Abdallah, Gning e Bonnifait (2007) apresentaram abordagens híbridas para o problema de localização. Ambos os trabalhos exploram as melhores características de abordagens probabilísticas e intervalares para lidar com o problema.

Neuland et al. (2014a) propôs um método híbrido usando filtro de partículas e análise de intervalos para o problema de localização. A ideia geral do método é concentrar a distribuição das partículas em áreas que dão maior indicativo de conter a solução. Essas áreas são delimitadas através de técnicas intervalares baseadas em restrições dadas pelas leituras dos sensores. Para a delimitação das regiões foram testadas duas técnicas, uma usando contratores e outra usando SIVIA. O método usando contratores representa a região através de um intervalo multidimensional, ele se mostrou mais rápido porém menos preciso que o uso do SIVIA. Quando usamos o SIVIA, que representa a região através de um conjunto de intervalos multidimensionais, o método se mostrou mais preciso porém mais lento que o método com contratores. As abordagens híbridas propostas apresentaram bons resultados, sendo mais precisas que o uso de uma técnica puramente intervalar, uma vez que temos informação proveniente das partículas dentro do intervalo. Além disso, devido à distribuição mais inteligente das partículas, menos partículas são necessárias, o que torna as abordagens propostas mais rápidas que a abordagem usando puramente o filtro de partículas.

A Figura 2.3 apresenta o mesmo momento de execução do método para as diferentes abordagens, usando apenas o filtro de partículas, a abordagem híbrida com contratores e a abordagem híbrida com SIVIA. Onde pontos azuis são marcadores no ambiente, o

ponto verde representa a média do conjunto de partículas, o ponto preto indica a real posição do robô e os pontos variando do amarelo para o vermelho (da pior para melhor em relação a peso) são partículas. O retângulo de borda preta (imagem central) representa a região delimitada pelo contrator, e a área cinza (imagem direita) representa a região delimitada pelo SIVIA.

Figura 2.3: Exemplo do mesmo momento de execução com três abordagens diferentes



Filtro de partículas | Abordagem usando contratores | Abordagem usando SIVIA

Fonte: Neuland et al. (2014a).

Baseado no trabalho de Neuland et al. (2014a), Weiss et al. (2019) propôs uma otimização na forma de busca pela localização do robô, onde não é necessário manter a delimitação das regiões de interesse usando intervalos durante todo o processo. Essa otimização reduziu o tempo de processamento preservando as vantagens em relação a precisão dos resultados.

Neuland et al. (2020), baseados no trabalho anterior (NEULAND et al., 2014a), propuseram uma variação do método, incluindo o uso de intersecção relaxada, para o problema do robô raptado. Esse problema é uma variação mais complexa do problema de localização, pois a qualquer momento no processo de localização o robô pode ser movido para uma posição totalmente diferente, demandando a reinicialização do processo. O novo método se mostrou robusto a *outliers*, que são inerentes ao uso de sensores, e capaz de encontrar regiões com fortes indícios de conter a solução mesmo que a informação proveniente dos sensores seja conflitante.

Por outro lado, o *Box Particle Filter* (BPF) (ABDALLAH; GNING; BONNIFAIT, 2007; ABDALLAH; GNING; BONNIFAIT, 2008) é uma versão intervalar do filtro de partículas. Ao invés de usar partículas associadas a estados pontuais do mundo, ele as define como sendo intervalos multidimensionais, os quais podem ser vistos de duas formas. Ou o intervalo representa infinitas partículas distribuídas de forma contínua dentro

de seus limites, ou o intervalo representa a incerteza da localização exata de uma partícula no mundo. De qualquer forma, o uso dessa abordagem garantiu bons resultados, superando o filtro de partículas clássico em relação ao tempo de computação.

Uma década depois, o BPF foi estendido e utilizado para tratar o problema de SLAM (WANG et al., 2018). Assim como na versão utilizada para tratar o problema de localização, os resultados dos experimentos apontam que o BPF mesmo usando menos partículas, alcança a mesma precisão que os métodos de SLAM baseados em filtro de partículas, em menos tempo.

Ainda no contexto de SLAM, Rohou et al. (2018) apresentou um método intervalar para detecção de fechamento de *loop* em trajetórias usando apenas informações de sensores proprioceptivos. O método é adequado para situações em que as observações feitas no ambiente não são confiáveis, como, por exemplo, ambientes subaquáticos. Os autores afirmam que apesar do método poder ser utilizado em ambiente de qualquer natureza, por si só não melhora a localização do robô, uma vez que a abordagem não traz nova informação de sensores exteroceptivos e não geram novas restrições para o problema.

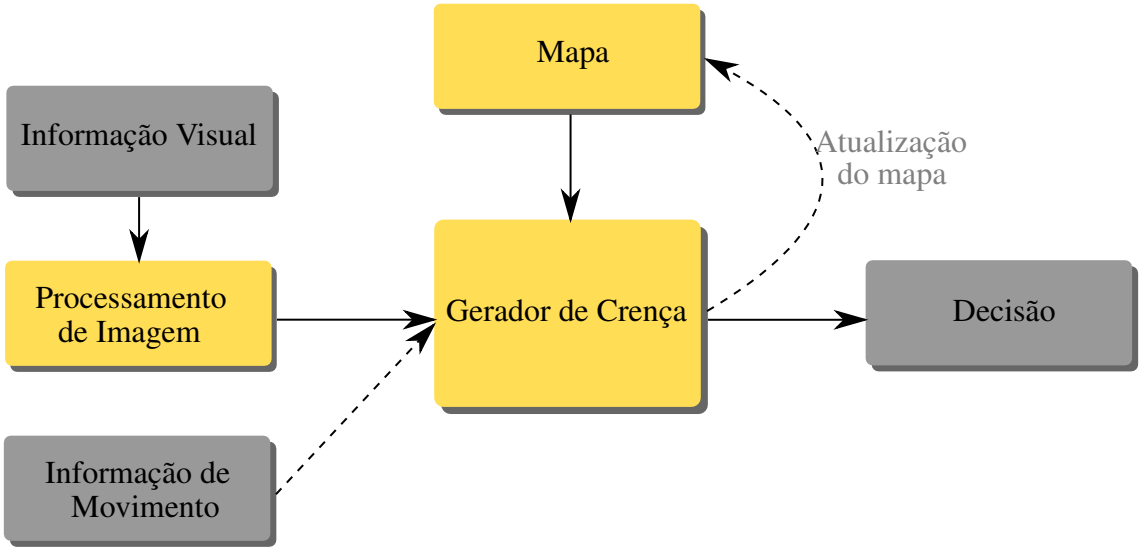
Uma característica comum nos métodos intervalares e presente em todos os citados nessa seção é o uso de informações ancoradas em um sistema de coordenadas do robô, onde, todas as distâncias e representações são relacionadas ao espaço de trabalho. Vale ressaltar que essas informações nem sempre estão disponíveis pois podemos ter apenas amostras de imagens coletadas durante uma travessia do robô no ambiente, por exemplo. A proposta apresentada nos Capítulos 4 e 5 foi elaborada com o objetivo de mostrar que mesmo sem informações dessa natureza ainda podemos tirar proveito de técnicas intervalares.

3 RECONHECIMENTO VISUAL DE REGIÕES

Dada uma observação do ambiente, definir se ela vem de uma região nova ou se foi coletada durante uma revisita a um lugar conhecido caracteriza o problema de reconhecimento de regiões. Nesse capítulo é apresentada uma visão geral das etapas relacionadas a sistemas visuais para reconhecimento de regiões e as principais abordagens consideradas como marcos na literatura.

Um sistema para reconhecimento visual de regiões pode ser dividido em três módulos (LOWRY et al., 2016). Um dos módulos está relacionado ao processamento de imagens, onde as imagens coletadas podem, por exemplo, sofrer alterações para realçar ou suavizar características, ou podem ser redimensionadas ou descritas de forma mais compacta. Outro módulo é responsável pela manutenção do mapa, onde novas informações coletadas são usadas para melhorar o mapa do ambiente, seja trazendo mais informação ou em conjunto com outras técnicas para a redução da incerteza. E um terceiro módulo que é encarregado da geração da crença sobre as observações analisadas. Ele é responsável por definir quais os locais do mapa possuem mais indícios de representar a observação atual, além de definir uma medida de qualidade para essas possíveis combinações. A Figura 3.1 apresenta um esquema com a relação entre esses três módulos do sistema.

Figura 3.1: Representação de um sistema de reconhecimento visual de regiões



Baseada na fonte: Lowry et al. (2016).

3.1 Processamento de Imagens

O módulo de processamento de imagens inclui o tratamento das observações recebidas. Isso engloba, por exemplo, o redimensionamento, a extração de informações e transformação entre os sistemas de cor, o tratamento de ruídos e a descrição da imagem. Dentre estes, a descrição das imagens e suas diferentes técnicas ganham atenção pela diversidade existente. Estas são amplamente classificadas de acordo com o método usado, podendo ser globais, baseadas em características locais, ou baseadas em *Bag-Of-Words* (BoW) (GARCIA-FIDALGO; ORTIZ, 2015).

Técnicas locais para descrição de imagens funcionam em duas etapas. Primeiro as características mais relevantes da imagem são selecionadas, para então serem descritas em uma segunda etapa. Na etapa de seleção de características os métodos varrem a imagem em busca de estruturas específicas como cantos ou BLOBs (*Binary Large OBject*). Na segunda etapa, cada uma das estruturas selecionadas (*keypoints*) é descrita por um vetor numérico. Os primeiros métodos propostos usavam vetores multidimensionais e de ponto flutuante. Posteriormente, o uso de descritores baseados em vetor binário se popularizou, devido essencialmente ao fato de que exigem um menor custo computacional e de armazenamento (GARCIA-FIDALGO; ORTIZ, 2015).

Diferente dos métodos locais que têm uma etapa para extração de partes de interesse da imagem, o método global usa a imagem como um todo. Contudo, descritores globais podem ser gerados a partir de métodos locais através da criação de *keypoints* fixos, usando, por exemplo, um padrão de grade sobre a imagem. A união dos descritores de cada célula da grade forma o descritor global da imagem (LOWRY et al., 2016).

Técnicas locais e globais são amplamente utilizadas nos mais diversos trabalhos, cada uma com suas vantagens e desvantagens. Uma das vantagens de técnicas locais é poder ser estendida para além do uso trivial apontado anteriormente. Elas podem ser usadas para descrever imagens que não foram vistas explicitamente pelo robô, fazendo a reordenação dos descritores para montar diferentes pontos de vista sintéticos. Apesar de descritores globais serem mais sensíveis às mudanças de ponto de vista, oclusão parcial e rotação da câmera, eles funcionam bem para capturar a estrutura geral da imagem (GARCIA-FIDALGO; ORTIZ, 2015). Já os descritores locais têm a performance reduzida em casos de mudanças de iluminação, o que faz com que a performance de técnicas globais supere as locais em problemas que envolvam mudanças de condições ambientais (LOWRY et al., 2016).

Uma imagem também pode ser representada através da sua própria matriz RGB ou de intensidades, sem a computação de um descritor (MILFORD; WYETH, 2012). Nesse caso, o descritor é a própria imagem e com a dimensionalidade da mesma. A comparação entre duas imagens pode ser feita através da soma das diferenças absolutas, somando a diferença absoluta entre os pixeis correspondentes de cada imagem. Quanto maior o valor resultante, menos similares são as imagens, sendo que para imagens iguais o valor resultante é zero.

O HOG (*Histogram of Oriented Gradients*) (DALAL; TRIGGS, 2005) é um descritor global inicialmente proposto para auxiliar na detecção de figuras humanas através de um histograma da direção dos gradientes da imagem de interesse. O descritor divide a imagem em uma grade uniforme e computa um histograma para cada célula da grade, os quais são normalizados e concatenados em um único histograma como descritor final. Apesar de ser uma representação menor que o uso puramente da imagem, o vetor de descrição resultante do HOG é dimensionalmente maior, e seu custo computacional mais alto em comparação com outros como os apresentados a seguir.

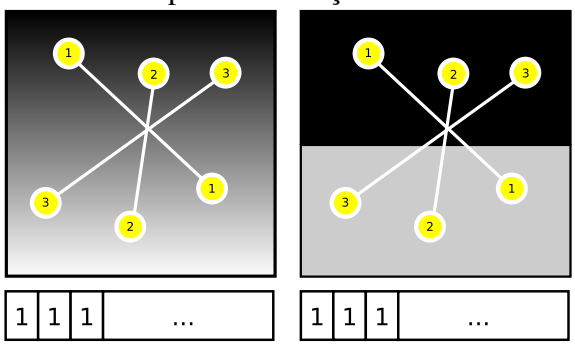
Criado com foco em recuperação de imagens, o PHOG (Pyramid Histogram of Orientated Gradient) tem um processo de descrição em duas fases. Primeiro ele representa características locais usando HOG em seções da imagem. Na segunda fase características espaciais são representadas a partir da combinação dos descritores das seções. O PHOG mantém benefícios de um descritor global com a vantagem de ser menos sensível aos efeitos de rotação. Ele se mostrou adequado para uso em métodos de detecção de fechamento de *loop* (BOSCH; ZISSERMAN; MUNOZ, 2007) (GARCIA-FIDALGO; ORTIZ, 2017).

A necessidade de descritores não apenas discriminativos, mas também mais compactos e rápidos para extrair e comparar, vem aumentando junto com o crescimento do uso de dispositivos móveis. Com isso, cada vez mais descritores binários, que atendem esses requisitos, vêm sendo desenvolvidos. Madeo e Bober (2016) apresenta uma comparação detalhada sobre diferentes opções da literatura.

O BRIEF (CALONDER et al., 2010) é um descritor binário criado para ser computacionalmente eficiente e consumir pouca memória. Ele usa uma sequência de bits para representar uma imagem, sendo que cada bit é obtido através da comparação entre a intensidade de um par de pontos selecionados previamente. Dada a simplicidade do descritor, ele tem limitações na capacidade de distinção entre imagens como apresentado na Figura 3.2. Apesar das imagens serem obviamente diferentes, os pares de pontos selecionados

não são capazes de codificar essa diferença.

Figura 3.2: Exemplo de limitação do descritor BRIEF



Fonte: Yang e Cheng (2014).

O LDB (*Local Difference Binary*) (YANG; CHENG, 2014) é um descritor binário proposto com o objetivo de ser mais robusto e distintivo que o seu concorrente direto, o BRIEF. Ele foi proposto inicialmente como um descritor local, contudo, apresentou resultados promissores quando testado como global (ARROYO et al., 2015). O LDB divide a área de interesse em uma grade regular e computa, para cada célula, a intensidade e os gradientes em x e y. Com essa informação, ele executa uma série de testes binários comparando os resultados entre as células da grade. O método usa múltiplas granularidades de grade para capturar a estrutura da imagem com diferentes níveis de detalhe. Para cada granularidade de grade é gerado um descritor binário, os quais são concatenados para formar o descritor LDB da área de interesse.

Outra técnica encontrada na literatura da área é o BoW (SIVIC; ZISSERMAN, 2003), que no contexto de imagens derivou da técnica de mesmo nome aplicada na área de processamento de texto. Em uma etapa de pré-processamento o BoW extrai e descreve um conjunto de características relevantes de um subconjunto de imagens. Os descritores resultantes do processo são chamados de palavras e são organizados em um vocabulário. Uma cena pode então ser descrita através de um vetor binário indicando a presença ou ausência de dada palavra do vocabulário. Essa técnica é resistente à variação no ponto de vista e à oclusões parciais. Contudo, precisa de uma etapa de treinamento com imagens similares as quais precisa descrever.

Métodos descritores também podem ser baseados em redes neurais. Nos últimos anos, diferentes métodos têm sido apresentados na literatura. Apesar da necessidade de uma etapa de treinamento, ela pode ser feita previamente com um grande volume de imagens e usada em diferentes conjuntos de dados (HOU; ZHANG; ZHOU, 2015) que tenham relação com os dados de treinamento.

3.2 Mapa

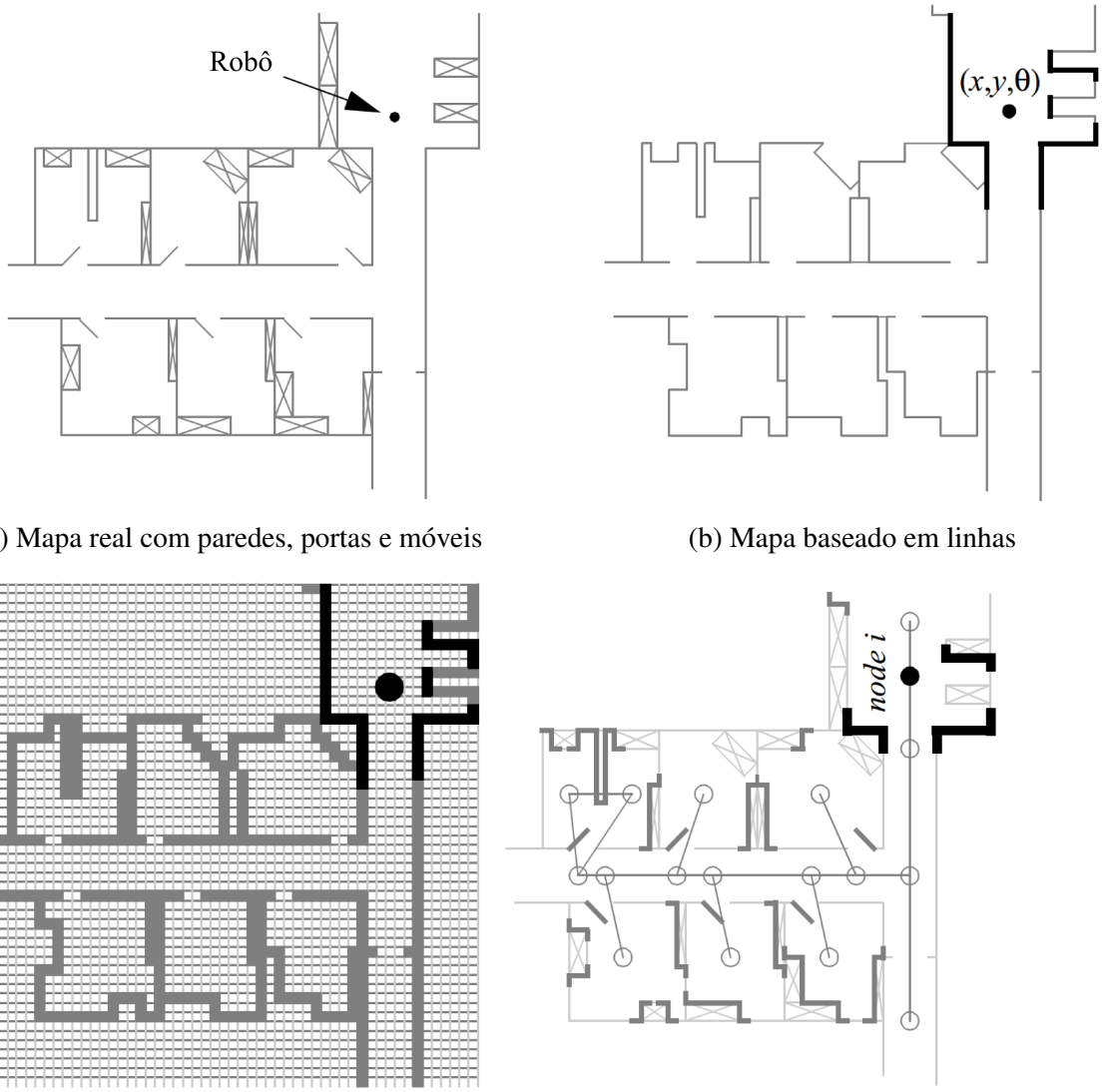
Para executar a tarefa de reconhecimento de regiões, o sistema precisa de um mapa com a representação do ambiente conhecido. Esse mapa é usado para comparação com novas observações a fim de definir se uma observação provém de uma região previamente visitada ou não. Quando as observações são provenientes de múltiplos sensores existem duas opções. Uma é a fusão das informações em um único mapa, outra é o uso e manutenção de múltiplos mapas, um para cada tipo de sensor. Sendo que manter mais de um mapa é preferível caso os sensores sejam adequados para diferentes tipos de obstáculos (THRUN et al., 2005).

Existem alguns fatores importante que impactam a geração e manutenção de mapas (THRUN et al., 2005) e precisam ser levados em consideração no tratamento do problema de reconhecimento de regiões. Dentre os mais importantes estão:

- Tamanho: ambientes muito grandes são desafiadores em dois sentidos. Primeiro, percorrer todo esse ambiente vai exigir tempo. Segundo, quanto maior o ambiente maior é a necessidade de armazenamento de informações.
- Ruídos: quanto mais ruídos e incertezas estão envolvidos no processo, mais difícil é a geração de um mapa preciso, assim como reconhecer uma região já visitada.
- Ambiguidade: quando existem muitos lugares que possuem características similares a dificuldade de se estabelecer uma relação de correspondência entre os lugares já visitados no decorrer do tempo aumenta.

O tipo de mapa usado para representação do ambiente depende das informações disponíveis, isso impacta no custo de manutenção e armazenamento do mapa. A Figura 3.3 apresenta quatro diferentes tipos de mapas representando um mesmo ambiente (SIEGWART; NOURBAKHSH, 2004).

Figura 3.3: Exemplificação de tipos de mapas



(a) Mapa real com paredes, portas e móveis

(b) Mapa baseado em linhas

(c) Mapa baseado em grade de ocupação

(d) Mapa topológico

A Figura 3.3a mostra o mapa real do ambiente com a representação das portas, paredes e mobiliário. A Figura 3.3b apresenta o mapa do ambiente baseado em linhas que delimitam os obstáculos do ambiente. A Figura 3.3c apresenta um mapa baseado em grade, onde cada célula da grade representa uma porção do ambiente. Caso haja um obstáculo na região representada pela célula, ela será marcada como ocupada. A Figura 3.3d mostra um mapa topológico, esse tipo de mapa representa o ambiente através de um grafo, onde os nodos são as regiões do ambiente e os arcos são as relações entre eles. Um modelo mais abstrato do ambiente é o armazenamento de informações puramente sobre a aparência de cada região, como uma lista. Esse modelo não mantém informações sobre a posição das regiões no mundo nem a relação geográfica entre elas (LOWRY et al., 2016). Mais detalhes sobre o assunto podem ser encontrados em (SIEGWART; NOURBAKHSH,

2004)

3.3 Geração de Crença

O módulo de geração da crença é o cerne de um método de reconhecimento de regiões. Nessa etapa o método usa o mapa em conjunto com as novas observações para definir se o robô está ou não em uma região conhecida. É de entendimento geral que se observações dão origem a descritores similares existe um forte indicativo de que elas tenham sido coletadas em uma mesma região. Contudo, o grau de certeza pode variar de acordo com o ambiente. Em ambientes simétricos, por exemplo, observações de diferentes regiões, distantes geograficamente, podem apresentar descrições muito similares. Já em ambientes sujeitos à mudanças – tais como alterações de clima, de iluminação, ou a movimentação de pessoas ou veículos – observações de um mesmo ponto podem ser descritas de forma totalmente diferentes (LOWRY et al., 2016).

Nesse contexto, além de definir se uma observação vem de um lugar conhecido ou não, métodos de reconhecimento de regiões devem atribuir uma crença para o resultado apresentado, seja através de esquemas de votação, probabilidade, ou outras abordagens como será apresentado na seção seguinte. Essa crença é utilizada para avaliação do sistema, que é comumente feita através das métricas de *precision* e *recall*.

3.4 Reconhecimento Visual de Regiões na Literatura

Diferentes abordagens vêm sendo propostas para lidar com o problema de reconhecimento visual de regiões ao longo dos anos. Esta seção apresenta uma amostra da literatura da área assim como os principais marcos.

O FAB-MAP (CUMMINS; NEWMAN, 2008) é um dos marcos da literatura na área, ele é um método com abordagem probabilística para o problema de reconhecimento visual de regiões baseado puramente em informações de aparência. O método é inspirado em sistemas de recuperação de imagens usando BoW (SIVIC; ZISSERMAN, 2003), que foi um de seus diferenciais. Ele cria um vocabulário a partir da clusterização de características extraídas de um conjunto de imagens de treinamento, onde cada *cluster* dá origem a uma palavra. Esse vocabulário é usado na descrição de imagens através de um vetor indicando a ocorrência de cada palavra. Algumas palavras são consideradas mais signi-

ficativas que outras, de acordo com a frequências delas no conjunto de imagens. Assim, para que duas imagens sejam associadas ao mesmo lugar com uma alta probabilidade, não basta que elas compartilhem palavras, elas devem compartilhar palavras consideradas significativas. O FAB-MAP 2.0 (CUMMINS; NEWMAN, 2011) apresenta uma versão melhorada do FAB-MAP. Na versão 2.0 o modelo probabilístico utilizado é alterado a fim de permitir o uso de índice invertido, o que torna o método capaz de lidar com mapas maiores.

Maffra, Chen e Chli (2018) propuseram um *framework* para reconhecimento visual de regiões robusto a variação de ponto de vista. Ele usa um BoW adaptado para criação de um vocabulário baseado em descritor binário. O método funciona em paralelo com um sistema de SLAM e odometria visual que compartilham informações de características coletadas do ambiente.

O SeqSLAM (MILFORD; WYETH, 2012) é um método de reconhecimento visual de regiões para ambientes sujeitos à mudanças na aparência, causadas, por exemplo, pela transição entre dia e noite, mudança das estações do ano, ou mesmo pelas variações climáticas. Ele é considerado um marco na literatura e trouxe como inovação a busca por sequências de regiões ao invés de confiar puramente nas informações extraídas de uma imagem corrente. Ele busca não apenas pela melhor combinação local entre imagens, mas também, visa encontrar entre as melhores imagens àquela com a sequência mais promissora. Para definir se a imagem capturada no momento corrente corresponde a um local já visitado, o método se baseia na sequência de imagens recentemente capturadas para usar como busca no mapa. Essa sequência de busca é comparada a todas as sequências locais de navegação, em um processo computacionalmente custoso (SIAM; ZHANG, 2017).

Uma implementação aberta do SeqSLAM, o openSeqSLAM (SÜNDERHAUF; NEUBERT; PROTZEL, 2013), disponibilizada em matlab, testa o método em um *dataset* desafiador com severas alterações no ambiente geradas a partir das mudanças de estações do ano. Esse trabalho também destaca que um bom resultado do SeqSLAM depende principalmente de um *dataset* com ponto de vista estável, ou seja, praticamente sem variações.

Liu e Zhang (2013) apresentaram um método focado em melhorar a performance do SeqSLAM através do uso de Filtragem Bayesiana (Filtro de Partículas). Cada partícula representa uma possível combinação, incluindo a localização do início da sequência e o seu comprimento. O método proposto pode gerar resultados equivalentes a uma busca

exaustiva em pelo menos uma ordem de magnitude de tempo menor.

Wang et al. (2015) propuseram uma melhoria para o SeqSLAM para operações em tempo real através de três estratégias. A primeira estratégia é a integração de informações de odometria ao processo visando tornar o algoritmo de busca mais eficiente. A segunda é a otimização do método de busca para encontrar combinações para a observação atual, não para aquelas capturadas alguns segundos atrás como o SeqSLAM original. A terceira é a busca por combinações em múltipla-escala, onde o método começa buscando sequências mais longas, para depois fazer uma busca mais precisa em sequências menores.

O Fast-SeqSLAM (SIAM; ZHANG, 2017) foi proposto alguns anos mais tarde com o objetivo de ser uma versão mais eficiente do SeqSLAM. Esse método traz uma redução na complexidade sem degradação da precisão nos resultados em comparação com o SeqSLAM. A ideia chave do método é evitar a busca exaustiva por melhores sequências de imagens no mapa, usando uma estrutura de busca baseada em árvore. Outra diferença é que o Fast-SeqSLAM usa HOG (*histogram of oriented gradients*) para descrever as imagens, o que diminui a dimensão do descritor em relação ao SeqSLAM, que usa a própria imagem como descritor e a compara com com outras imagens através da soma das diferenças absolutas. A principal motivação na otimização proposta no Fast-SeqSLAM é o uso do método em operações grandes e de longo prazo.

Talbot, Garg e Milford (2018) propuseram o OpenSeqSLAM2.0, uma *toolbox* para reconhecimento visual de regiões baseada diretamente no SeqSLAM. Ela usa a mesma estrutura básica do SeqSLAM, também usada no OpenSeqSLAM, mas permite a variação de algumas etapas através de configuração, i.e., existem parâmetros que podem alterar o método a ser executado em uma dada etapa.

Tsintotas, Bampis e Gasteratos (2018) apresentaram um método chamado DO-SeqSLAM. Uma versão modificada do SeqSLAM, onde a principal alteração é o uso de sequências de comprimento dinâmico, ao invés do tamanho fixo usado pelo SeqSLAM original. Essa modificação reduz o custo computacional total permitindo seu uso em sistema *on-line*. Contudo, assim como o SeqSLAM, ele não computa combinações para imagens recentemente coletadas, mas para aquelas coletadas a alguns segundos no passado.

Garg e Milford (2020) propuseram um novo método para reconhecimento visual de regiões que combina uma representação ultra-compacta das regiões, com um armazenamento que escala de forma quase sub-linear e a necessidade de poucos requisitos computacionais. O método converte dados usados como referência em endereços de *hash*.

Os problemas relacionados a colisão desses endereços são tratados usando a natureza sequencial dos dados provenientes de veículos robóticos.

Existem também os métodos que tentam unir características do SeqSLAM e do FAB-MAP a fim de aproveitar as principais vantagens de cada um. Tsintotas et al. (2018) propuseram um método baseado em sequência que combina o SeqSLAM com BoW. O método cria um vocabulário visual em um procedimento *offline* que é usado para descrever as observações coletadas usando palavras visuais. A partir de então, o método segue como a versão original do SeqSLAM. O uso de BoW na descrição das observações traz um ganho de performance em casos com mudanças no ponto de vista, mas perde em casos de variação significativa nas condições do ambiente.

Huishen et al. (2018) apresentaram um método inspirado no FAB-MAP para a detecção de fechamento de *loops* baseado em BoW mas usando descritor binário. O método usa informação de profundidade e um grafo de co-visibilidade para verificação de *loops*, além disso, ele considera a consistência temporal de sequências de imagens. O objetivo é melhorar a performance apresentada pelo *precision-recall* em comparação com outros métodos que usam BoW, i.e., garantir a máxima precisão com a maior recuperação possível. Em sua atual configuração, não é uma abordagem adequada para aplicações em tempo real.

Citamos trabalhos como FAB-MAP e SeqSLAM que representam marcos na literatura, alguns trabalhos inspirados neles, e até mesmo trabalhos que mesclam suas características. Uma outra classe de trabalhos que vem crescendo nos últimos anos são aqueles baseados em redes neurais e *deep learning*. A seguir temos alguns exemplos.

Naseer, Burgard e Stachniss (2018) apresentaram um método de localização visual para longo prazo, o qual é baseado em imagens de uma câmera monocular e no uso de informação de sequência. Eles formularam o problema de combinação de sequências de imagens como um problema de fluxo de custo mínimo. O grafo de associação derivado dessa modelagem possibilita a computação de múltiplas hipóteses para a rota do veículo. O método usa um descritor semi-denso associado a descritores globais gerados por uma rede neural para a comparação entre as imagens.

Dongdong et al. (2018) apresentaram um método chamado SeqCNNSLAM para detecção de fechamento de loop baseado em uma rede neural convolucional (CNN). O método trata variações no ponto de vista e condições do ambiente através da junção da tradicional combinação baseada em sequência e uma CNN pré-treinada. No mesmo trabalho eles propõem variações do método. O A-SeqCNNSLAM reduz a variação na busca

de combinações para se tornar mais eficiente. O O-SeqCNNSLAM foi criado para fazer ajustes dos parâmetros usados pelo A-SeqCNNSLAM em ambiente ainda não visitados e a variação chamada P-SeqCNNSLAM explora os benefícios do uso de características locais.

Uy e Lee (2018) apresentaram o método PointNetVLAD para tratar o problema de reconhecimento de regiões baseado em nuvens de pontos 3D. Os autores apontam que a variação nas condições do ambiente, que é um problema conhecido quando lidamos com imagens, impactam pouco a nuvem de pontos. Contudo, a extração de descritores locais e a criação de uma descrição global para as nuvens de pontos é um desafio. O método é uma combinação entre as redes neurais PointNet (QI et al., 2017) e NetVLAD (ARANDJELOVIC et al., 2016). No trabalho foram propostas funções para obter descritores globais mais discriminativos e generalizáveis.

Merrill e Huang (2018) propuseram um método para detecção de fechamento de *loop* baseado em uma arquitetura de rede neural não supervisionada. Para treinar a rede neural, transformações são aplicadas às imagens de entrada a fim de capturar variações extremas de pontos de vista causadas pela movimentação do robô. Para auxiliar no aprendizado da geometria das cenas o método usa o descritor HOG que comprime a informação da imagem preservando as características mais salientes e a homografia.

Vysotska e Stachniss (2019) propuseram um método para reconhecimento visual de regiões focado em lidar com mudanças sazonais, de clima e de iluminação em um mapa composto por sequências de imagens. As características das cenas são extraídas por uma rede neural convolucional. Os autores modelaram o problema em uma estrutura em grafo de forma que a busca por combinações de imagens é feita através de um algoritmo que busca o caminho mais curto no grafo.

Chancán e Milford (2020) apresentaram um novo método chamado DeepSeqSLAM baseado em uma arquitetura que une redes neurais recorrentes (RNN) e CNN para aprender as características visuais e de posição em imagens provenientes de uma câmera monocular. O modelo CNN precisa ser treinado previamente e a RNN é treinada usando imagens de uma das travessias no ambiente.

Zaffar et al. (2020) propuseram uma abordagem para reconhecimento visual de regiões baseada HOG chamada CoHOG. O método foi inspirado em CNN e tenta reproduzir algumas características das CNNs, tais como, ser capaz de varrer toda uma imagem em busca de uma característica específica, extrair regiões de interesse distintas e informativas, e aprender representações de regiões que sejam imunes à variações sazonais ou de

iluminação.

A Tabela 3.1 apresenta um resumo comparando os trabalhos citados nesta seção. Em casos onde os recursos computacionais são escassos ou não existem dados suficientes para treinamento, técnicas baseadas em *deep learning* podem não ser a melhor opção (TALBOT; GARG; MILFORD, 2018). Apesar do surgimento de diferentes técnicas baseadas em *deep learning*, como as mencionadas anteriormente, outras técnicas para reconhecimento visual de regiões ainda são relevantes para a área de pesquisa. Nas próximas seções vamos apresentar uma nova proposta baseada em teoria de análise de intervalos que também explora as vantagens da natureza sequencial das informações coletadas por veículos robóticos.

Tabela 3.1: Resumo da comparação entre trabalhos relacionados

Método	**Foco**	**Diferencial**	**Descritor**	**Compara com**	***Datasets***
FAB-MAP	Marco na literatura, trata o problema de reconhecimento visual de regiões usando puramente imagens	BoW	SURF	Variações	New College / City Center
FAB-MAP2.0	Melhoramento do modelo probabilístico do FAB-MAP para uso do método em mapas maiores	Novo modelo probabislístico	SURF	FAB-MAP	próprio
Maffra, Chen e Chli (2018)	Método de reconhecimento visual de regiões para veículos aéreos	Usa odometria visual e trata alterações significativas de ponto de vista	BoW + BRISK	ORB-SLAM	próprio
SeqSLAM	Foca em ambientes com mudança de aparência	Busca usando sequências de imagens	Imagem	FAB-MAP	Nurburgring / Alderley
OpenSeqSLAM	Implementação aberta do SeqSLAM	Apresenta o dataset Nordland	Imagem		Nordland
Liu e Zhang (2013)	Melhorar a eficiência do SeqSLAM	Usa filtro de partículas para variar tamanho e começo das sequências	GIST	SeqSLAM	Imagens do Google Street View
Wang et al. (2015)	Melhorar o SeqSLAM para operações em tempo real	Integram uma janela deslizante baseada na odometria visual	Imagem	SeqSLAM	KITTI
Fast-SeqSLAM	Operações de longo prazo	Otimização do SeqSLAM, uso de estrutura de árvore, complexidade de O(nlogn)	HOG	SeqSLAM	Nordland / UofA / GPW
OpenSeqSLAM2.0	*Toolbox* parametrizável baseada no SeqSLAM	Executar o SeqSLAM com diferentes parâmetros	Imagem	Variações	Nordland / Eynsham
DOSeqSLAM	Aumentar a acurácia do SeqSLAM	Segmentação dinâmica das sequências	Imagem	SeqSLAM	Lip6O / KITTI
Garg e Milford (2020)	Representação compacta dos locais, recuperação rápida	Técnicas de hashing para armazenamento da informação	NetVLAD	SeqSLAM (versão modificada)	FAS100K / Deep1B
Tsintotas et al. (2018)	Melhorar o SeqSLAM em relação a descrição das imagens e variação de ponto de vista	Uso de BoW com o SeqSLAM	BoW + ORB	SeqSLAM	Bivosa 2008-09-01 / Biccoca 2009-02-25b / Lip6 Indoor / Lip6 Outdoor / Malaga 2009 Parking 6L
Huishen et al. (2018)	Detecção de loops baseado em aparência usando BoW, focado em melhorar o performance analisada com precison-recall em relação aos métodos do estado da arte	Criação de blocos de palavras baseadas no BoW e uso de grafo de co-visibilidade	BoW + ORB	FAB-MAP / DBoW2	Malaga2013 / próprio
Naseer, Burgard e Stachniss (2018)	Capacidade de execução por longos períodos de tempo	Tratam o problema de reconhecimento visual de regiões como um problema de fluxo de custo mínimo	HOG e ImageNet	FABMAP2 / SeqSLAM	VPRiCE / Nordland / NewCollege / GPW / próprio (FAS)
SeqCNNSLAM	Aumentar eficiência e robustez em relação ao SeqSLAM	Combinação do SeqSLAM com CNN	CNN	CNNLCD / FAB-MAP / SeqSLAM	Nordland / GPW / City centre
Merrill e Huang (2018)	Eficiência e robustez para detecção de fechamento de loop usando uma arquitetura baseada em uma rede neural não-supervisionada	Arquitetura baseada em redes neurais não-supervisionada	HOG	DBoW2 / AlexNet	Alderley / Gardens Point / Nordland
Vysotska e Stachniss (2019)	Execução em mapa composto por múltiplas sequências com variação perceptual	Recuperação de imagens baseada em uma estratégia de hashing	CNN	FABMAP / SeqSLAM / DBoW2	Imagens do Google Street View
DeepSeqSLAM	Uso de redes neurais artificiais para aprender representações visuais e de posição a partir de sequências de imagens monoculares	Uma arquitetura treinável com CNN+RNN	NetVLAD	SeqSLAM	Nordland / Oxford RobotCar
CoHOG	Tratar o problema com baixo consumo de memória e sem necessidade de treinamento	Inspirada por características das CNNs mas sem necessidade de treinamento	HOG	Variação do método com diferentes técnicas de descrição (AlexNet / NetVLAD / HybridNet / ...)	GPW / ESSEX3IN1 / SPEDTest / Synthia

4 UMA NOVA ABORDAGEM INTERVALAR PARA MODELAR O MUNDO OBSERVADO

Esse capítulo apresenta nossa proposta para modelar o mundo observado usando a teoria de Análise de Intervalos, onde regiões do ambiente são representadas por intervalos. Mostraremos como isso pode ser feito mesmo quando as observações feitas no ambiente não contém informação métrica. Em nossa abordagem, consideramos um robô inserido em um espaço de trabalho $\mathbb{W}$, o qual é modelado como um espaço Euclidiano n-dimensional (LATOMBE, 1991). O robô se desloca ao longo de um caminho τ definido como uma função contínua

$$\tau : [0; 1] \rightarrow \quad \mathbb{W}. \tag{4.1}$$

Onde $[0; 1]$ define as posições que o robô assumiu ao longo do caminho, sendo 0 a posição inicial e 1 a posição final.

Em algumas posições do caminho percorrido, o robô faz uma coleta de informações, o que chamamos de observação. Um conjunto de observações $\mathbb{U}$ pode ser ordenado de acordo com o aparecimento dos elementos do conjunto ao longo do caminho. Assim, a ordem de uma observação é dada pela função

$$o : \mathbb{U} \rightarrow \mathbb{N}. \tag{4.2}$$

Sendo cada observação associada a uma ordem, o conjunto das ordens $\mathbb{O}$ é definido como:

$$\mathbb{O} = \big\{o(u) \mid u \in \mathbb{U}\big\}. \tag{4.3}$$

É possível definir a distância no espaço das ordens, o-distância, entre duas observações como

$$\rho : \mathbb{U} \times \mathbb{U} \rightarrow \mathbb{N}. \tag{4.4}$$

Por exemplo, dadas as observações u, $v \in \mathbb{U}$, a o-distância entre elas é

$$\rho(u, v) = |o(u) - o(v)|. \tag{4.5}$$

Vale destacar que a o-distância está relacionada à ordem em que o robô coletou as observações, ou seja, a informação coletada na observação em si não está sendo considerada.

Outra medida essencial entre observações é a similaridade, ela representa o quão parecidas duas observações são, ou seja, quanta informação compartilham. Nesta medida é considerada a informação coletada na observação, e a função de similaridade é definida por

$$\alpha : \mathbb{U} \times \mathbb{U} \rightarrow [0; 1]. \tag{4.6}$$

Onde α representa uma função genérica de similaridade, já que o cálculo depende do tipo de informação coletada que está sendo comparada. Por exemplo, observações baseadas em medidas de distância como laser e sonar podem ser comparadas através da soma das diferenças absolutas.

As definições de o-distância e similaridade quando analisadas juntas podem ser indicativos de que:

- as observações $u, v \in \mathbb{U}$ foram extraídas de pontos contíguos do caminho, se

$$\alpha(u, v) \geq \theta'_{\alpha} \quad \wedge \quad \rho(u, v) \leq \theta'_{\rho}, \tag{4.7}$$

  i.e., u e v são bastante similares e estão próximas. Onde, θ'_{α} e θ'_{ρ} são limiares predefinidos que indicam respectivamente a similaridade mínima e a o-distância máxima entre observações para que sejam consideradas como coletadas em pontos contíguos.

- a observação $u \in \mathbb{U}$ foi coletada durante uma revisita, quando existe pelo menos um $v \in \mathbb{U}$ onde

$$o(u) > o(v) \ \wedge \ \alpha(u, v) \geq \theta'_{\alpha} \ \wedge \quad \rho(u, v) \geq \theta'_{\rho}. \tag{4.8}$$

  Ou seja, u e v são muito similares mas não foram coletadas uma logo após a outra.

- o robô não está se movendo, ou está em uma região simétrica quando coleta um conjunto de observações $\mathbb{U}' \subseteq \mathbb{U}$, sendo que para todos $u, v \in \mathbb{U}'$ a seguinte condição é verdadeira

$$\rho(u, v) < |\mathbb{U}'| \quad \wedge \quad \alpha(u, v) \geq \theta'_{\alpha}. \tag{4.9}$$

  Isto significa que houve a coleta de uma sequência de observações muito similares entre elas.

Observações coletadas em pontos contíguos do caminho normalmente compar-

tilham uma grande quantidade de informação. Sendo assim, essas observações podem ser vinculadas a uma representação única para descrever uma região, criando uma versão simplificada do mundo. Pontos do caminho podem ser agrupados em intervalos de acordo com a *o*-distância e similaridade das observações neles coletadas, onde cada intervalo representa um subcaminho de τ.

Dadas duas observações $u, v \in \mathbb{U}$, onde $u \neq v$, existe um caminho entre as posições de onde essas observações foram coletadas. Mesmo que u e v sejam coletadas uma logo após a outra, existem infinitas possíveis observações entre elas que são desconhecidas, e portanto não pertencem ao conjunto $\mathbb{U}$. Contudo, essa porção do mundo que não tem representação através de observações pode ser representada através do uso de intervalos. Considere, por exemplo, o intervalo $[x] \in \mathbb{IO}$. Sendo $\mathbb{IO}$ o conjunto de todos os intervalos de $\mathbb{O}$.

Ele pode ser um intervalo pontual e representar uma única observação u

$$[x] = [o(u); o(u)]. \tag{4.10}$$

Ou pode representar um conjunto de observações, e a porção de mundo entre elas, como

$$[x] = [o(u); o(v)]. \tag{4.11}$$

A criação de um intervalo deve respeitar algumas condições. Cada intervalo $[x]$ tem um elemento âncora $A_{[x]}$ usado para definir quais elementos fazem ou não parte desse intervalo. Suponha $[x] \in \mathbb{IO}$, para que $o(u) \in [x]$:

1. $u \in \mathbb{U}$;
2. $\alpha(u, A_{[x]}) \geq \theta''_\alpha$, i.e. a similaridade entre $A_{[x]}$ e u deve ser maior que um dado limiar θ''_α;
3. $\rho(u, A_{[x]}) \leq \theta''_\rho$, i.e. a *o*-distância entre $A_{[x]}$ e u deve ser menor que um limiar θ''_ρ;
4. os elementos do conjunto $\mathbb{U}$ com ordem entre $o(A_{[x]})$ e $o(u)$ pertencem ao intervalo $[x]$.

Onde os limiares θ''_α e θ''_ρ representam respectivamente a similaridade mínima entre uma âncora e todos os elementos do intervalo relacionado e a *o*-distância máxima entre uma âncora e todos os elementos do seu intervalo.

Após definidos os limites de um intervalo $[x]$ é necessário definir uma representação que descreva a região como um todo, uma representação global que denominamos

como $G_{[x]}$. Seria possível usar $A_{[x]}$ como representação global, mas nós definimos uma estrutura diferente que visa representar todo o contexto dentro do intervalo. Essa estrutura é apresentada no próximo capítulo.

A partir dos conceitos definidos neste capítulo nós conseguimos representar o mundo conhecido através de um conjunto de intervalos. Essa representação nos permite manter 2 níveis de abstração do ambiente, onde a busca por regiões pode ser feita puramente usando as representações intervalares sem a necessidade de percorrer todas as observações. Enquanto que uma busca mais detalhada, usando as observações coletadas, pode ser feita dentro de cada intervalo. Além disso, a representação intervalar não está ancorada em um sistema de coordenadas do mundo e pode ser usada para representar observações não métricas.

A Tabela 4.1 resume os limiares citados no capítulo. Ela apresenta os símbolos e significados de cada um a fim de facilitar eventuais consultas.

Tabela 4.1: Limiares usados para a modelagem proposta

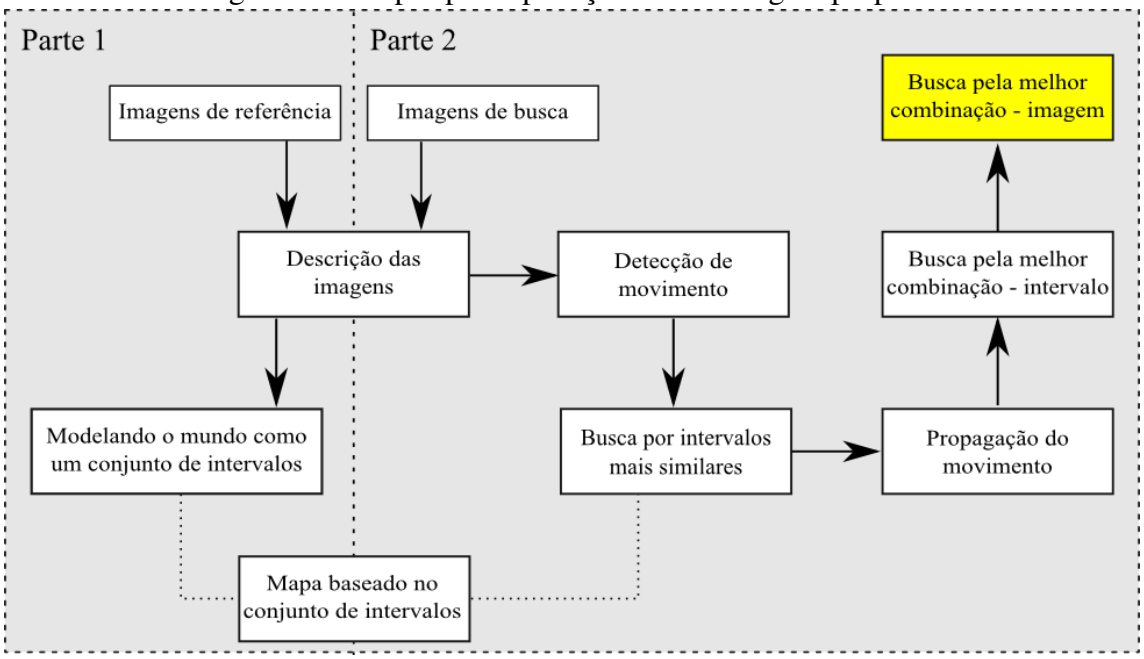
Símbolo	Significado
θ'_{α}	indica a similaridade mínima entre duas observações coletadas em pontos próximos
θ''_{α}	indica a similaridade mínima entre uma âncora e todos os elementos do intervalo relacionado
θ'_{ρ}	indica a *o*-distância máxima entre duas observações para que elas sejam consideradas provenientes de pontos contíguos
θ''_{ρ}	indica a *o*-distância máxima entre uma âncora e todos os elementos do seu intervalo

5 PROBLEMA DE RECONHECIMENTO DE REGIÕES EM UM MUNDO DE INTERVALOS - APLICAÇÃO DA MODELAGEM PROPOSTA

A partir dos conceitos abordados no capítulo anterior vamos descrever uma aplicação prática para a modelagem proposta. Nesse capítulo nós apresentamos um novo método para tratar o problema de reconhecimento visual de regiões usando uma abordagem intervalar onde as observações coletadas são estritamente imagens coletadas por uma câmera monocular.

Em relação ao cenário adotado, o método recebe dois conjuntos de observações - um conjunto referência e um conjunto busca - gerados a partir de duas trajetórias feitas no mesmo ambiente sob diferentes condições. O método tem o objetivo de identificar imagens de lugares correspondentes entre os conjuntos. O uso da teoria de análise de intervalos nos permite manter uma representação compacta das observações coletadas pelo robô e reduzir o número de computações sobre essas observações. Essa abstração da informação coletada reduz o espaço de busca e acelera a procura por regiões similares.

Figura 5.1: Etapas para aplicação da modelagem proposta



A Figura 5.1 apresenta o fluxo de processamento para o reconhecimento visual de regiões usando a modelagem proposta. O processo se divide em duas partes, a primeira (esquerda) representa desde a coleta do conjunto de imagens usadas como referência até a geração do mapa. Note que o processo de descrição das imagens é compartilhado por ambas as partes. Assim como o mapa representado através de um conjunto de intervalos

que é criado na primeira parte, como resultado da modelagem do mundo, e é consultado na segunda parte, durante a busca por intervalos mais similares (essas relações estão representadas por linhas pontilhadas). A segunda parte (direita) apresenta desde a coleta das imagens do conjunto de busca, até a seleção de uma imagem do mapa com mais indícios de representar o mesmo local de onde a busca foi coletada. Cada um dos processos representados na figura são descritos em detalhes a seguir.

5.1 Coleta e processamento das observações

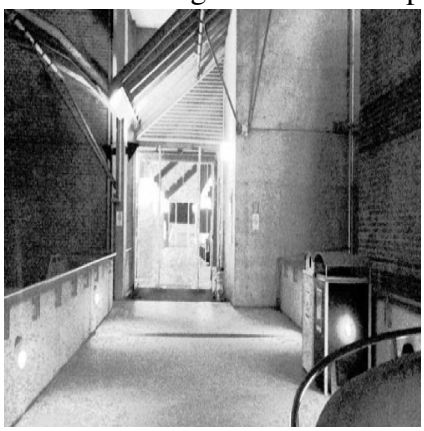
Todas as observações coletadas pelo robô são imagens provenientes de uma câmera monocular. O primeiro passo após a coleta é definir como essa informação será armazenada. O uso de descritores é comum em abordagens que lidam com imagens, uma vez que são capazes de gerar uma representação mais compacta da observação. Assim, ao invés de processar e armazenar matrizes (imagens), o método lida com vetores que são consideravelmente menores.

Como discutido anteriormente no Capítulo 3.1, soluções para o problema de reconhecimento visual de regiões que sofrem alterações perceptuais mostraram que o uso de descritores globais é mais adequado do que o uso de descritores locais. Descritores binários, no geral, produzem representações mais compactas, requerendo menos memória para o armazenamento. Além disso, possuem um processo de comparação rápido baseado em distância de Hamming (HAMMING, 1950). Com isso em mente, propomos o uso de uma representação binária global baseada no descritor LDB[1].

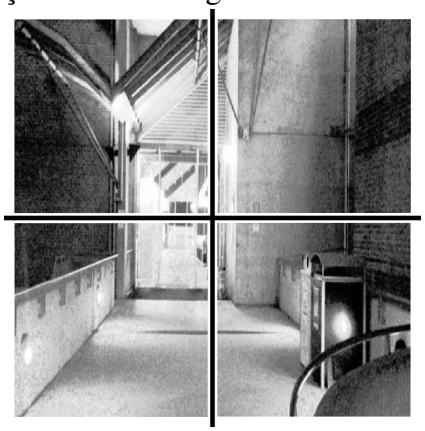
As etapas desse processo são ilustradas na Figura 5.2. Onde, dada uma imagem coletada pelo robô (Figura 5.2a), ela é dividida em quatro partes (Figura 5.2b), as quais são descritas individualmente usando o LDB (Figura 5.2c), cada uma dando origem a um vetor binário. O descritor global da imagem é obtido pela concatenação desses quatro vetores (Figura 5.2d). O conceito de divisão da imagem em partes fixas pode ser visto em outros descritores (BOSCH; ZISSERMAN; MUNOZ, 2007), e traz benefícios como a inclusão de informação espacial e a possibilidade de recuperação em situações onde ocorrem oclusões parciais das observações.

[1]Código disponível em: «https://github.com/roberto-arroyo/OpenABLE/tree/master/lib/ldb»

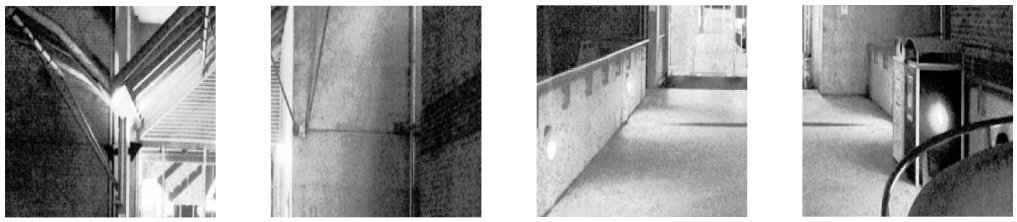
Figura 5.2: Passos para a descrição de uma imagem



(a) Imagem coletada

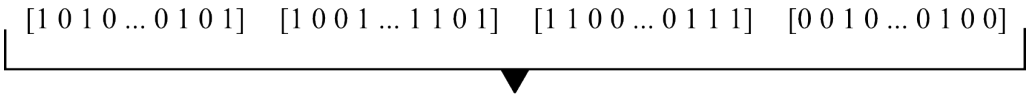


(b) Divisão da imagem em quadrantes



[1 0 1 0 ... 0 1 0 1] [1 0 0 1 ... 1 1 0 1] [1 1 0 0 ... 0 1 1 1] [0 0 1 0 ... 0 1 0 0]

(c) Descrição de cada quadrante individualmente



[1 0 1 0 ... 0 1 0 1] [1 0 0 1 ... 1 1 0 1] [1 1 0 0 ... 0 1 1 1] [0 0 1 0 ... 0 1 0 0]

[1 0 1 0 ... 0 1 0 0]

(d) Geração do descritor global

5.2 Modelando o mundo conhecido como um conjunto de intervalos

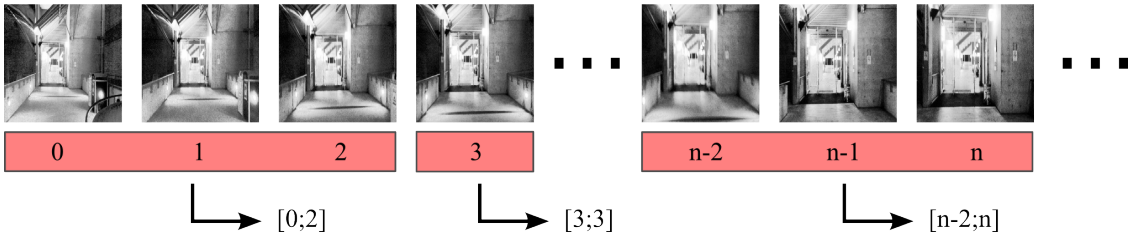
Como visto na seção anterior, o resultado da observação coletada pelo robô após o processamento é um descritor (vetor binário), que é chamado a partir de agora de observação. Nessa seção vamos apresentar como essas observações são armazenadas e usadas para a criação do mapa. Como citado anteriormente, as imagens se dividem em dois conjuntos: o conjunto referência e o conjunto de busca. Nessa etapa de criação do mapa as imagens utilizadas são as de referência.

Nesse trabalho o modelo de mapa utilizado foi a lista de descrições. Contudo, a fim de acelerar a busca por regiões similares, o método proposto agrupa as observações em conjuntos de acordo com funções de similaridade e proximidade temporal. Esses conjuntos são representados por intervalos.

A Figura 5.3 ilustra o processo de criação dos intervalos e a geração de um mapa

em dois níveis. No primeiro nível são armazenadas as observações (descritores/imagens). Logo abaixo, em vermelho, temos a representação das ordens. Essas ordens dão origem ao segundo nível, onde propomos uma representação mais compacta agrupando em intervalos as imagens de uma região de alta similaridade.

Figura 5.3: Geração de intervalos a partir de observações do ambiente



Cada observação é rotulada de acordo com a ordem de coleta, onde $\mathbb{O}$ é o conjunto contendo todos os rótulos (vermelho). Considere as imagens rotuladas de 0 a 2. Supondo que elas respeitem as quatro condições descritas no Capítulo 4, o método cria o intervalo $[0; 2]$ para representar essa região do ambiente. É importante destacar que os intervalos podem ser criados enquanto o robô se desloca coletando a informação. Além disso, devido às características do conjunto $\mathbb{O}$ os intervalos criados contêm valores inteiros não negativos.

Nosso método representa o mundo conhecido pelo robô através de um conjunto de intervalos adaptando as **quatro condições** descritas previamente. Considerando a **primeira condição**, todas as observações contidas no intervalo vêm do conjunto referência. Em relação à **segunda condição**, uma observação só faz parte de um intervalo $[x]$ caso ela tenha uma similaridade com $A_{[x]}$ maior que um dado limiar θ''_α. A função de similaridade é baseada na distância de Hamming. Nós consideramos para fim de implementação que a primeira observação de um intervalo é a âncora dele, e θ''_α é um valor dinâmico computado durante a execução e atualizado a cada nova observação coletada. Ele representa a média da similaridade entre todas as observações capturadas consecutivamente, exceto aquelas que não tem diferença significativa, ou seja, tem similaridade superior a 95%, existindo um forte indício de que o robô estava parado no momento da coleta.

A **terceira condição** é usada para limitar o tamanho do intervalo, o qual pode variar de 1 até o tamanho máximo do conjunto referência. No caso do limite máximo, se todas as imagens do conjunto apresentarem uma similaridade muito alta entre elas o mapa pode se resumir a apenas um intervalo. Isso pode ocorrer caso todas as observações sejam coletadas, por exemplo, em um longo corredor sem características relevantes distinguíveis ao longo do caminho. E finalmente, a **quarta condição** garante que uma nova

observação vai fazer parte do último intervalo criado ou será a primeira observação de um novo intervalo.

Considere um intervalo $[x]$ criado usando $A_{[x]}$ como âncora. Até agora analisamos a similaridade de observações, mas para o método é importante também analisar a similaridade entre intervalos. Para isso, $[x]$ precisa ter uma representação da informação que ele contém, essa representação global do intervalo $[x]$ chamamos de $G_{[x]}$. É possível considerar $A_{[x]} = G_{[x]}$, contudo, nós propomos uma nova estrutura que considera todas as observações do intervalo.

A Figura 5.4 mostra um exemplo de como criamos a representação global de um intervalo. Considere o intervalo $[x] = [7; 10]$ criado a partir de 4 observações. A representação global $G_{[x]}$ é dada por um par de vetores contendo respectivamente o mais frequente e sua frequência relativa $G_{[x]} = (G^M_{[x]}, G^F_{[x]})$. Onde $G^M_{[x]}$ (maioria) armazena os valores mais frequentes em cada índice do vetor de descrição (0 ou 1) e $G^F_{[x]}$ (frequência relativa) armazena o percentual que esses valores aparecem. No exemplo, a primeira posição de todos os descritores é 0, então a primeira posição dos vetores $G^M_{[x]}$ e $G^F_{[x]}$ são respectivamente 0 (maioria dos elementos no primeiro índice) e 1 (100% dos elementos são 0). Em caso de empate, o método associa a maioria como 0 e a frequência relativa de 50%, isso ocorre considerando a segunda posição dos descritores. Por uma questão de simplificação, a partir daqui vamos chamar frequência relativa apenas de frequência.

Figura 5.4: Representação das observações associadas ao intervalo $[x]$ através de $G_{[x]}$

Intervalo $[x]$	**Descritores em** $[x]$	**Representação global** $G_{[x]}$
$[x] = [7; 10]$	[0 1 1 1 ... 1 1 1 0]	$G^M_{[x]} = [0 \quad 0 \quad 1 \quad 0 \quad \cdots \quad 1 \quad 0 \quad 0 \quad 0]$
	[0 0 1 0 ... 1 0 0 0]	$G^F_{[x]} = [1 \quad 0.5 \quad 0.75 \quad 0.75 \quad \cdots \quad 1 \quad 0.75 \quad 0.5 \quad 1]$
	[0 0 1 0 ... 1 0 0 0]	
	[0 1 0 0 ... 1 0 1 0]	

Em suma, todas as observações do conjunto de referência são agrupadas usando a representação intervalar e cada intervalo ganha uma representação global. Dessa forma, é possível acelerar a busca por combinações entre os conjuntos através da busca por regiões (intervalos) similares, para só então buscar observações similares dentro dessas regiões.

5.3 Buscando correspondências no mapa

Nesta seção vamos detalhar a segunda parte da imagem 5.1, onde novas observações são analisadas a fim de definir se elas foram coletadas em um lugar previamente visitado ou não. O método proposto busca combinações baseado não somente na similaridade entre observações, mas também na sequência de combinações no decorrer de uma janela de tempo. O processo passa por cinco etapas. Após a coleta e descrição das imagens, a primeira etapa do método define se o robô está ou não se movendo. Na segunda etapa, o método busca no conjunto de intervalos por aquele com maior similaridade com a observação atual. Na terceira etapa é feita a propagação da movimentação do robô no conjunto de melhores intervalos selecionados em cada iteração. Na quarta etapa o intervalo com maior indicativo de conter a região onde a observação atual foi coletada é definido. Baseado nesse intervalo, na quinta etapa, o método varre as observações representadas por ele a fim de encontrar aquela com maior similaridade com a observação atual. Mais detalhes de cada etapa serão descritos a seguir.

O método proposto tem dois parâmetros a serem considerados: k e w. Sendo que o parâmetro k define a quantidade de candidatos selecionados a cada iteração, na segunda etapa do método. E w representa o tamanho da janela de iterações passadas que serão consideradas para a busca da melhor combinação da observação atual.

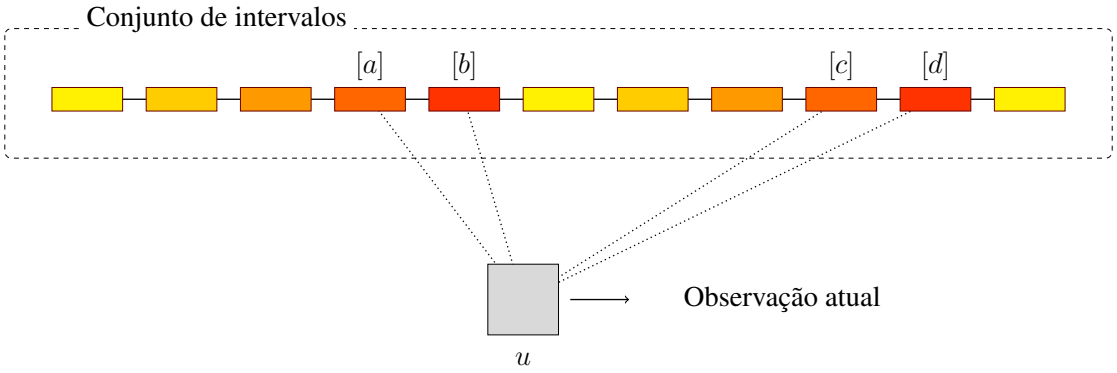
5.3.1 Etapa 1, detecção do movimento do robô

A informação utilizada sobre a movimentação do robô é binária, i.e., o método assume que o robô está se movendo (1) ou está parado (0). O método determina o movimento do robô de acordo com um valor de similaridade média entre observações consecutivas do conjunto de busca. Se a similaridade média das últimas observações coletadas for inferior a 95% da similaridade média do conjunto o método considera que o robô está se movendo. Em suma, caso não seja detectada uma alteração significativa nas últimas observações, consideramos que o robô está parado. A informação de movimento é armazenada em um conjunto $\mathbb{M}$, onde o tamanho máximo desse conjunto é definido por $w - 1$. Caso $|\mathbb{M}| = w - 1$ e mais informação precise ser armazenada, os elementos mais antigos de $\mathbb{M}$ serão descartadas.

5.3.2 Etapa 2, busca por intervalos mais similares

Nesta etapa o método usa a observação atual para pesquisar no conjunto de intervalos pelos k vizinhos mais próximos. Esses k intervalos representam regiões do ambiente com fortes indicativos de conter a região na qual a observação atual foi coletada. A Figura 5.5 ilustra esse processo. Supondo $k = 4$ e dada a observação atual u, o método seleciona 4 intervalos do conjunto com maior similaridade com u. Nesse caso, $[a]$, $[b]$, $[c]$ e $[d]$. Considere as cores dos intervalos do amarelo ao vermelho como resultado da similaridade, onde quanto mais vermelho mais similar o intervalo é da observação atual.

Figura 5.5: Seleção dos k intervalos mais similares à observação atual



O método mantém um conjunto $\mathbb{V} = \{v_1, v_2, ..., v_k\}$ contendo os vizinhos mais próximos selecionados na iteração. A cada iteração um conjunto é gerado e armazenado em $\mathbb{Y} = \{\mathbb{V}_1, \mathbb{V}_2, ..., \mathbb{V}_w\}$, o qual mantém os conjuntos selecionados durante as últimas w iterações. Caso $|\mathbb{Y}| = w$ e um novo conjunto precise ser armazenado, o elemento mais antigo em $\mathbb{Y}$ é descartado.

Como citado anteriormente, o método precisa ser capaz de comparar intervalos e observações em relação à similaridade. Com isso em mente, nós definimos a função de similaridade como

$$\beta : \mathbb{U} \times \mathbb{U} \rightarrow [0; 1]. \tag{5.1}$$

Note que para computar β é necessário transformar as observações de entrada da função em intervalos, uma vez que a computação é baseada na representação global do intervalo. Para fins de implementação nós definimos a função β como

$$\beta = \eta \sum_{i=1}^{|G^M_{[x]}|} r(i), \tag{5.2}$$

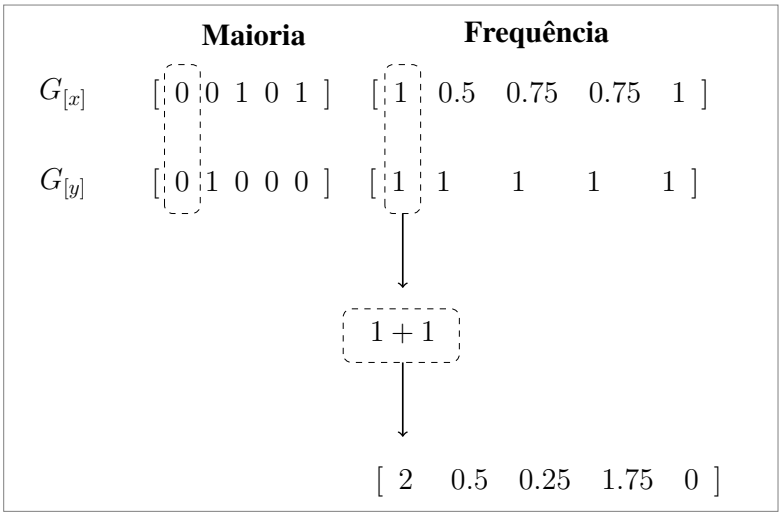
onde η é uma constante de normalização para manter o valor final entre 0 e 1, e r é dado por

$$r(i) = \begin{cases} G^F_{[x]}(i) + G^F_{[y]}(i) & \text{se } G^M_{[x]}(i) = G^M_{[y]}(i), \\ 2 - G^F_{[x]}(i) - G^F_{[y]}(i) & \text{caso contrário.} \end{cases} \tag{5.3}$$

Dessa forma, β é capaz de representar até mesmo similaridades parciais de cada posição dos vetores. Tradicionalmente, calculando a similaridade baseada na distância de hamming, cada posição do vetor de descrição tem uma contribuição binária para o cálculo de similaridade. Com a representação global proposta, cada posição pode ter uma contribuição equivalente ao seu percentual de similaridade.

As Figuras 5.6 e 5.7 mostram como calcular o valor de β através de um exemplo. Considerando dois intervalos $[x]$ e $[y]$ com suas representações globais $G_{[x]}$ e $G_{[y]}$. Nosso método varre o vetor maioria de $G_{[x]}$ e $G_{[y]}$ verificando se os elementos são iguais posição a posição. Caso os valores sejam iguais, o método soma os valores do vetor frequência como apresentada na Figura 5.6.

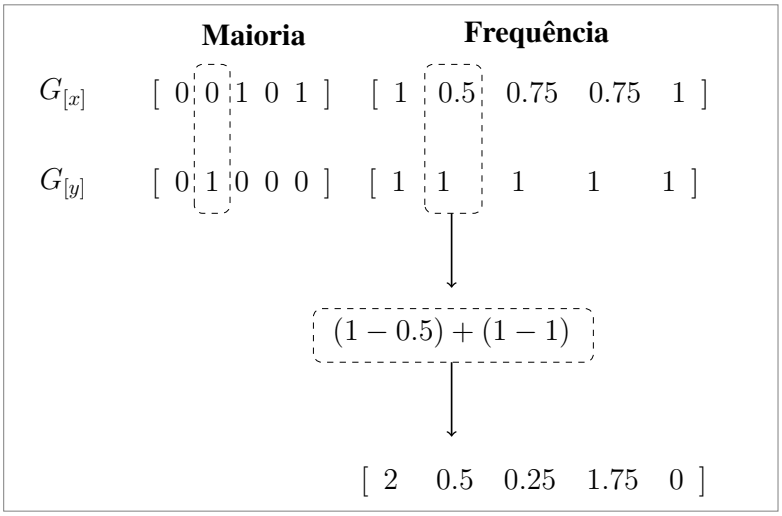
Figura 5.6: Computando β. Caso 1: valores iguais



Por outro lado, caso os valores de uma mesma posição sejam diferentes, o método soma o resultado de 1 menos o valor de frequência (destacado na Figura 5.7). Dessa forma, mesmo que os elementos de uma mesma posição do vetor sejam diferentes eles ainda podem ter uma parcela de similaridade, e dessa forma contribuem para o valor final de β.

Por fim, os resultados parciais são somados e normalizados. Sendo assim, a simi-

Figura 5.7: Computando β. Caso 2: valores diferentes



laridade entre $[x]$ e $[y]$ apresentados no exemplo é

$$\frac{2 + 0.5 + 0.25 + 1.75 + 0}{10} = 0.45. \tag{5.4}$$

5.3.3 Etapa 3, propagando o movimento do robô

O próximo passo é propagar o movimento do robô para todos os intervalos selecionados no decorrer das últimas w iterações. Cada intervalo em $\mathbb{Y}$ é somado a um δ que é computado a partir dos valores em $\mathbb{M}$. Lembrando que os intervalos contêm as ordens das observações e que o somatório dos valores de $\mathbb{M}$ resulta em um escalar. O intervalo resultante é incluído no conjunto $\mathbb{C}$, que contém os candidatos a melhor combinação da iteração corrente. Para todos os $\mathbb{V}_i \in \mathbb{Y}$,

$$\mathbb{C} = \bigcup_i \Big\{ [x] + \delta_i \mid [x] \in \mathbb{V}_i \Big\} \tag{5.5}$$

onde

$$\delta_i = \sum_{j=i}^{|\mathbb{M}|} m_j. \tag{5.6}$$

A criação do conjunto $\mathbb{C}$ é um passo importante para incorporar a movimentação do robô na busca por soluções. A Figura 5.8 apresenta um exemplo de como é feita a propagação dos intervalos em $\mathbb{Y} = \{\mathbb{V}_1, \mathbb{V}_2, \mathbb{V}_3, \mathbb{V}_4\}$, considerando $w = 4$, $k = 2$ e um conjunto $\mathbb{M} = \{0, 1, 1\}$. A cada iteração (representada pelos retângulos cinzas), k intervalos mais similares à observação atual são selecionados e propagados de acordo

com a movimentação do robô. As cores servem meramente para facilitar a visualização e diferenciar os intervalos de cada iteração.

A iteração $t-3$ não considera os candidatos anteriores pois o exemplo foi definido como $w = 4$. Na mesma iteração k intervalos foram selecionados e armazenados em $\mathbb{V}_1$ (azul). Na iteração $t-2$, o robô não se moveu, e dois novos candidatos foram selecionados (verde) e incluídos no conjunto $\mathbb{V}_2$. Entre as iterações $t-2$ e $t-1$ foi detectada a movimentação do robô, assim os candidatos selecionados previamente ($\mathbb{V}_1$ e $\mathbb{V}_2$) também são deslocados e novos intervalos são selecionados e armazenados em $\mathbb{V}_3$ (laranja). Na iteração atual t, todos os candidatos ($\mathbb{V}_1$, $\mathbb{V}_2$, e $\mathbb{V}_3$) são deslocados e novos candidatos são selecionados e incluídos no conjunto $\mathbb{V}_4$ (rosa). Assim, temos o conjunto $\mathbb{C}$ para o exemplo na iteração atual (t) representado na Figura 5.9.

Figura 5.8: Propagação dos intervalos para a criação do conjunto de candidatos $\mathbb{C}$

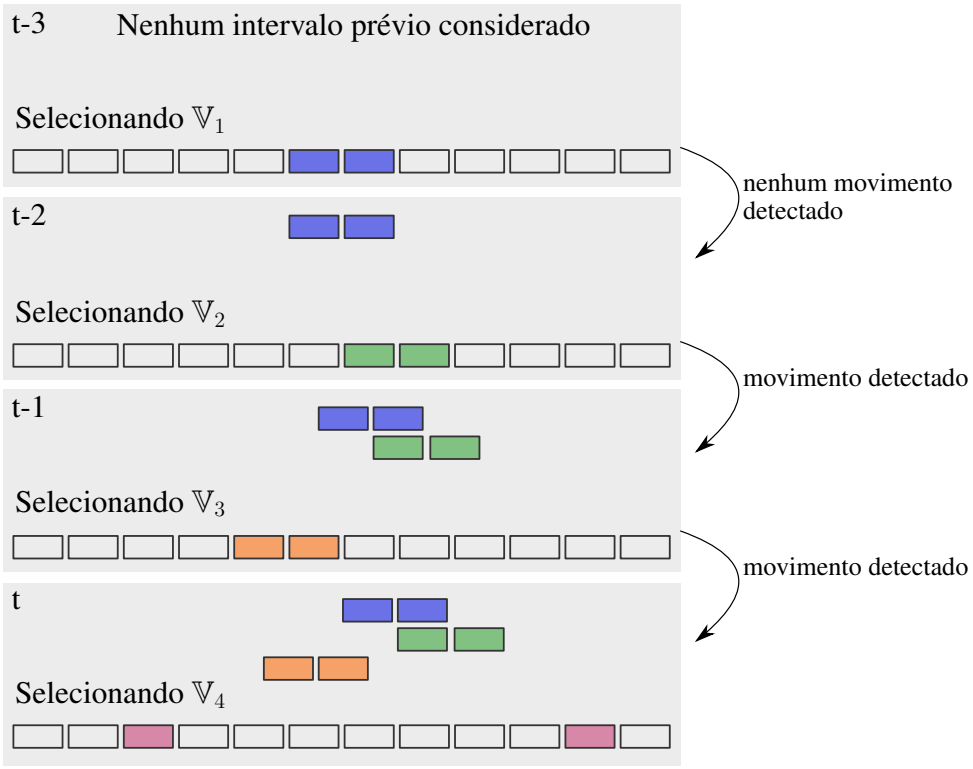
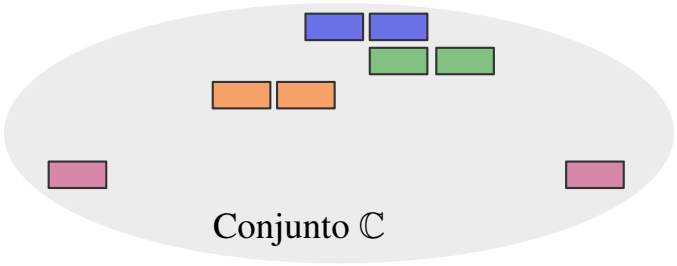


Figura 5.9: Conjunto $\mathbb{C}$ ao final da iteração t

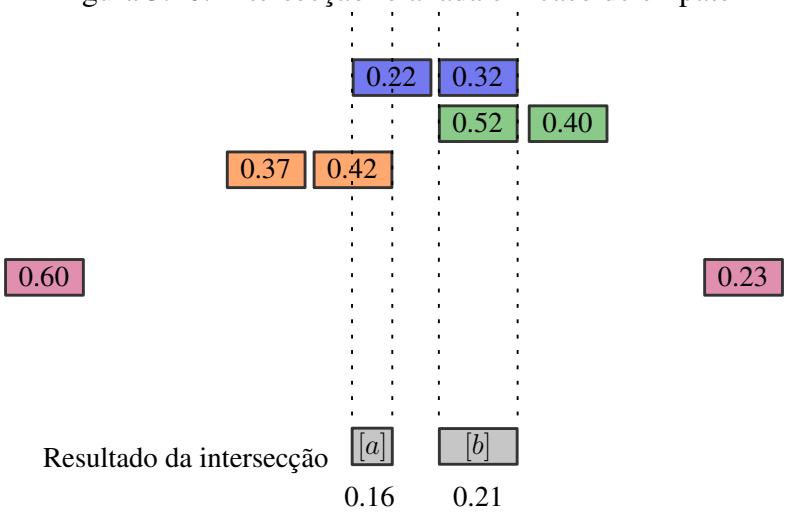


5.3.4 Etapa 4, busca pelo intervalo mais similar

Nesta etapa o método busca pelo intervalo mais similar à observação de busca atual. Assumindo que o conjunto $\mathbb{C}$ contenha a solução, ele é usado como entrada para o algoritmo de intersecção relaxada (*q-relaxed intersection*) apresentado no Capítulo 2.2. Onde o parâmetro q é o menor valor que retorna uma solução não vazia. O resultado desse método de intersecção relaxada define qual o melhor intervalo que combina com a observação de busca.

Na seleção de um intervalo durante a etapa 2 (Seção 5.3.1) a similaridade entre o intervalo e a observação atual é computada. Esse valor é útil nesta etapa em casos de empate. Se o algoritmo de intersecção relaxada encontrar mais de um intervalo disjunto como solução, o desempate é feito usando o valor de similaridade e apenas um intervalo é dado como resultado.

Figura 5.10: Intersecção relaxada em caso de empate



A Figura 5.10 mostra um exemplo onde a intersecção relaxada gera dois intervalos como resultado. Note que na figura, cada intervalo tem um valor associado que representa a similaridade calculada no momento da seleção de cada um deles. Em cinza está representado o resultado da intersecção relaxada, que nesse caso, são dois intervalos disjuntos chamados de $[a]$ e $[b]$. Para o desempate, cada um desses intervalos resultantes é associado ao valor (normalizado) da soma dos valores de similaridade dos intervalos que geraram a intersecção. Por exemplo, o intervalo $[a]$ resulta da intersecção de dois intervalos que quando selecionados apresentaram valores de similaridade de 0.22 e 0.42 em relação a observação usada como busca em suas respectivas iterações. Assim, o valor associado ao intervalo $[a]$ é dado por $(0.22 + 0.42)/4 = 0.16$. O intervalo com maior valor será

considerado como resultado final. No exemplo, o intervalo $[b]$ associado ao valor 0.21 seria o selecionado.

Resumindo, considerando que todo intervalo $[x] \in \mathbb{C}$ tem um valor $v_{[x]}$ associado a ele, um intervalo $[i]$ resultante da intersecção relaxada vai ter seu valor $v_{[i]}$ definido por:

$$v_{[i]} = \eta \sum_{\substack{[x]\in\mathbb{C} \\ [x]\cap[i]\neq\emptyset}} v_{[x]}, \tag{5.7}$$

onde η é um coeficiente de normalização.

5.3.5 Etapa 5, busca pela observação mais similar

Baseado no intervalo obtido na etapa anterior, o método refina o resultado para combinar a observação de busca com uma observação do intervalo. Existem diferentes formas de selecionar uma combinação. Nós optamos pelo uso da média ponderada, considerando como peso para o cálculo da média o valor dado pela similaridade entre cada uma das observações representadas pelo intervalo e a observação de busca. Usando o exemplo da Figura 5.10, suponha que na iteração atual o método recebeu uma observação u do conjunto de busca, i.e., $[b] = [\underline{b}; \overline{b}]$ foi selecionado a partir de u. A observação escolhida para a combinação se dá por

$$\left\lceil \frac{\sum_{i=\underline{b}}^{\overline{b}} i\beta(u,i)}{\sum_{i=\underline{b}}^{\overline{b}} \beta(u,i)} \right\rceil . \tag{5.8}$$

O uso da média ponderada foi inspirada no método chamado Filtro de Partículas (DELLAERT et al., 1999) (THRUN et al., 2005), onde é comum o uso da média ponderada para definir uma pose que melhor represente o conjunto de partículas como um todo.

A observação selecionada tem uma crença associada a ela, a qual é dada pela média entre a similaridade do intervalo selecionado (como descrito na seção anterior) e a similaridade da observação selecionada dentro desse intervalo. Dessa forma a seleção de ambos, intervalo e observação, tem um peso no valor de crença na decisão final. Suponha que a observação $b' \in [b]$ foi selecionada como melhor combinação. O valor associado a

essa combinação, também chamado de crença, é dado por

$$\frac{\beta(u, b') + v_{[b]}}{2}. \tag{5.9}$$

6 EXPERIMENTOS

Nesta seção são apresentados os experimentos e resultados obtidos com o método derivado da modelagem proposta anteriormente. Para a execução dos testes, o método recebe dois conjuntos de imagens referentes a duas travessias feitas em diferentes momentos. O objetivo geral é combinar imagens de um conjunto com o outro, definindo uma crença de acordo com a qualidade dessa combinação.

6.1 Dados de entrada

Para a avaliação do método proposto foram escolhidos alguns *datasets* desafiadores em relação à mudança de aparência e com variações de ponto de vista. Todos são usados em trabalhos da área, o que colabora para a comparação dos resultados obtidos com a nossa proposta e outros trabalhos. Para os nossos testes é necessário que todos os *datasets* possuam conjuntos de imagens ordenadas de acordo com a sequência de coleta. Além disso, para que possamos analisar a qualidade do resultado, os *datasets* devem disponibilizar informações sobre *groundtruth*. Mais detalhes podem ser vistos a seguir.

6.1.1 GPW - *Garden Points Walking*

O *dataset* GPW (GLOVER, 2014) é composto por três conjuntos de 200 imagens cada, as quais foram coletadas a partir de uma mesma rota. Dois dos conjuntos foram gerados durante o dia, e o outro durante a noite. As imagens são do campus da Universidade de Tecnologia de Queensland, em Brisbane, na Austrália, e foram capturadas com a câmera de um iPhone5. A Figura 6.1 apresenta uma amostra do *dataset* com quatro lugares vistos a partir de três travessias. Sendo que todas as imagens de uma mesma coluna são do mesmo lugar. Na coleta das imagens da primeira linha o celular estava sendo carregado na mão esquerda, enquanto na segunda, ele estava na mão direita. Isso causa alterações no ponto de vista entre imagens de um mesmo lugar. Na terceira linha, são apresentadas imagens coletadas durante a noite, sendo que o celular estava na mão direita. Para abreviar o nome dos conjuntos de dados deste *dataset* vamos nos referir a eles como GPW-DL (*day-left*), GPW-DR (*day-right*) e GPW-NR (*night-right*), respectivamente.

Figura 6.1: GPW - amostra de imagens

6.1.2 UofA - University of Alberta

O *dataset* UofA (SIAM; ZHANG, 2017) foi extraído na Universidade de Alberta, em Edmonton, Canadá. Ele é composto por dois conjuntos com 645 imagens cada, um deles coletado durante o dia e o outro ao entardecer. Esse *dataset* dispõe de imagens dos mesmos lugares com condições diferentes de iluminação. A captura dos dados foi feita por uma câmera RGB embarcada em um robô Husky. Amostras desse *dataset* são apresentadas na Figura 6.2, onde quatro lugares diferentes podem ser vistos a partir de duas travessias. As imagens de uma mesma coluna representam o mesmo lugar, sendo que, na primeira linha estão as imagens coletadas durante o dia e na segunda aquelas coletadas ao anoitecer. Para abreviar o nome dos conjuntos de dados deste *dataset*, vamos nos referir a eles como UofA-D (*day*) e UofA-E (*evening*), respectivamente.

Figura 6.2: UofA - amostra de imagens

6.1.3 Nord - Nordland

O *dataset* Nordland (SÜNDERHAUF; NEUBERT; PROTZEL, 2013) contém quatro vídeos de uma viagem de trem de 728km pelo norte da Noruega. Cada vídeo foi feito em uma estação do ano diferente, capturando mudanças significativas na aparência do ambiente. Para os experimentos executados foram usados conjuntos de imagens extraídas dos vídeos referentes as estações inverno, verão e primavera. As imagens foram extraídas com a frequência de 1 imagem a cada 10 segundos, sendo que as imagens de quando o trem estava parado foram removidas. Sendo assim, cada conjunto contém 3052 imagens. A Figura 6.3 apresenta uma amostra do *dataset* onde as imagens de uma mesma coluna representam o mesmo lugar. Na primeira linha estão imagens coletadas no inverno, na segunda estão imagens coletadas na primavera e na terceira linha estão as imagens coletadas no verão. Para abreviar o nome dos conjuntos de dados deste *dataset*, vamos nos referir a eles como Nord-Win (*winter*), Nord-Spr (*spring*) e Nord-Sum (*summer*), para as travessias coletadas no inverno, primavera e verão, respectivamente.

Figura 6.3: Nord - amostra de imagens

6.2 Uso de *precision-recall* para a avaliação e comparação de métodos para reconhecimento de regiões

Na literatura relacionada ao problema de reconhecimento visual de regiões é bastante comum a utilização de gráficos de *precision-recall* para a avaliação dos resultados. O *recall* e a *precision* são usados como forma de medir a efetividade de sistemas de recuperação de dados. Em outras palavras, ele mensura o quanto um sistema é capaz

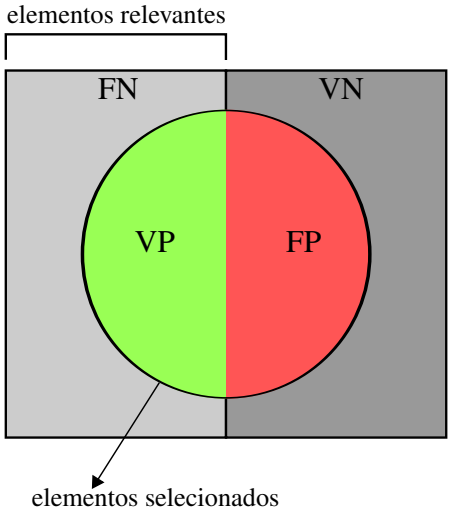
de obter respostas certas/relevantes enquanto mantém fora do conjunto de soluções respostas erradas/irrelevantes (RIJSBERGEN, 1979). Esta seção traz um breve resumo dos componentes dessa métrica.

Existem quatro conceitos essenciais para o cálculo do *precision-recall*, como ilustra a Figura 6.4), são eles:

- FN (Falso Negativo): um elemento verdadeiro, que foi classificado como falso;
- VN (Verdadeiro Negativo): um elemento falso, que foi classificado como falso;
- VP (Verdadeiro positivo): um elemento verdadeiro, que foi classificado como verdadeiro;
- FP (Falso positivo): um elemento falso, que foi classificado como verdadeiro.

Sendo que os elementos relevantes (FN ∪ VP) são aqueles que deveriam ter sido classificados como verdadeiros e os elementos selecionados (VP ∪ FP) são aqueles que o sistema classificou como verdadeiros.

Figura 6.4: Componentes das equações de *Precision-Recall*.



O cálculo da *precision* é dado na Equação 6.1, o valor resultante desse cálculo é a relação entre as respostas corretas que foram retornadas e todas as respostas retornadas pelo sistema (CUMMINS; NEWMAN, 2011).

$$precision = \frac{|\text{VP}|}{|\text{VP} \cup \text{FP}|} \tag{6.1}$$

A Equação 6.2 mostra o cálculo do *recall*, o qual apresenta a relação entre as respostas corretas retornadas pelo sistema e os elementos relevantes (CUMMINS; NEW-

MAN, 2011).

$$recall = \frac{|\text{VP}|}{|\text{VP} \cup \text{FN}|} \tag{6.2}$$

Para a criação das curvas de *precision-recall*, nós variamos o limiar relacionado à significância de cada resposta dada pelo método. Essa significância é uma crença relacionada a qualidade da resposta. Além disso, nós consideramos correspondências verdadeiramente positivas (TP) aquelas que ficam até uma certa distância máxima do que seria a resposta correta. Nós adotamos 3 imagens como distância máxima aceita, valor baseado no código disponível do FastSeqSLAM[1]. Dentro da nossa proposta essa distância se refere a distância em ordem (o-distância) dada pela função ρ. Por exemplo, se o *groundtruth* define que as imagens 2 e 5 são uma combinação correta e o método encontrou a combinação entre as imagens 2 e 7, consideramos que o método acertou. Caso a combinação seja 2 e 9, por exemplo, consideramos que método gerou uma combinação inválida, ou seja um falso positivo. Note que o numeração dada para as imagens representa a ordem de coleta. Caso exista a informação de distância espacial entre os pontos de coleta, outras métricas podem ser aplicadas.

O *precision-recall* é um indicador útil para sistemas de reconhecimento de regiões, principalmente para aqueles que fazem parte de um sistema de SLAM. Isso, porque o *recall* a 100% de *precision* indica o percentual de correspondências corretas feitas pelo sistema sem falsos positivos, que poderiam causar falhas no processo de SLAM (CUMMINS; NEWMAN, 2011).

6.3 Parametrização

Nesta seção vamos descrever os parâmetros usados para a execução dos métodos testados. Os parâmetros listados são referentes à abordagem proposta e ao OpenSeqSLAM2.0 (TALBOT; GARG; MILFORD, 2018), o qual é considerado parte relevante do estado da arte atual no que se refere a problemas de reconhecimento visual de regiões.

[1] https://github.com/siam1251/Fast-SeqSLAM

6.3.1 Definição de parâmetros para o OpenSeqSLAM2.0

A escolha dos parâmetros para execução de algoritmos costuma impactar significativamente nos resultados. Devido a isso e tentando alcançar os melhores resultados possíveis com OpenSeqSLAM2.0 usando os *datasets* mencionados anteriormente, nós testamos diferentes valores para os parâmetros seguindo as indicações apresentadas pelos autores do método.

R_{window}, d_s, $searchMethod$, e $matchingMethod$ são os principais parâmetros utilizados pelo OpenSeqSLAM2.0. O valor de R_{window} foi fixado a 2% da quantidade de imagens na travessia, como recomendado pelos autores. Isso significa 13 para UofA, 4 para o GPW e 61 para o Nordland. Os autores do OpenSeqSLAM2.0 também concluíram que as opções disponíveis para o $matchingMethod$ tiveram performances similares, sendo assim, fixamos o método chamado *score threshold*.

Os parâmetros $searchMethod$ e d_s mostraram maior impacto nos resultados, por isso, testamos algumas variações dentro dos limites sugeridos. As opções para o $searchMethod$ são *cone* e *trajectory-based*.

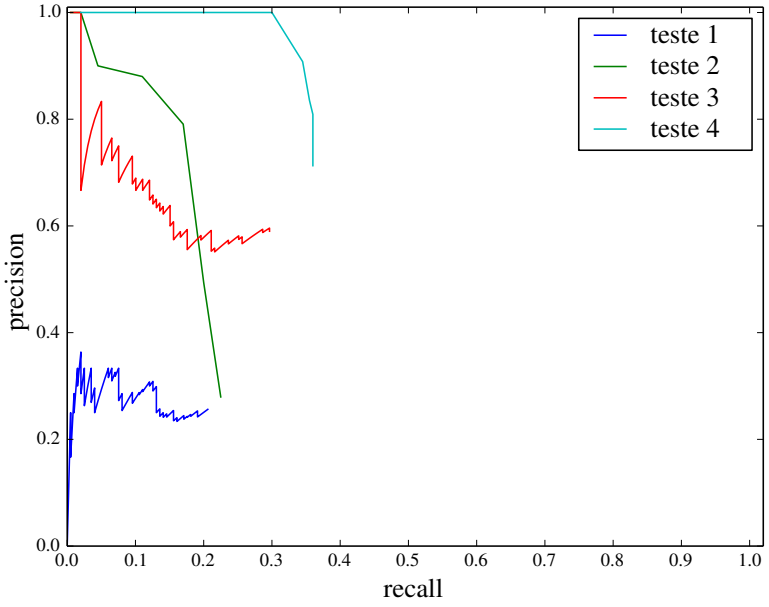
O parâmetro d_s afeta diretamente a performance do OpenSeqSLAM2.0 podendo aumentar o tempo de processamento. Os autores do OpenSeqSLAM2.0 sugeriram o uso de valores entre 2 e 40, sendo que os resultados deles indicam que quanto maior o valor de d_s melhores são os resultados das combinações. Em contrapartida, o uso de um d_s alto também aumenta o tempo de processamento necessário. Talbot, Garg e Milford (2018) apresentaram seus experimentos usando o *dataset* Nordland, e sugeriram um $d_s = 20$ e $trajectory\text{-}based$ como $searchMethod$, sendo assim, mantivemos esses valores nos experimentos apresentados a seguir.

A Figura 6.5 apresenta as curvas de *precision-recall* obtidas pelo OpenSeqSLAM2.0 usando os *datasets* GPW e UofA. A variação de parâmetros para cada curva é apresentada na Tabela 6.1. Os resultados desses testes apontaram que, assim como esperado, o uso de valores altos para d_s geram melhores combinações. Alguns testes usaram $d_s = 100$, muito mais alto que o recomendado. Entretanto, apesar do aumento relacionado ao tempo de processamento, o uso de valores mais baixos não geraram resultados competitivos. Note que em relação ao $searchMethod$, o GPW obteve melhores resultados usando a opção $cone$ enquanto $trajectory\text{-}base$ foi a melhor opção para o UofA.

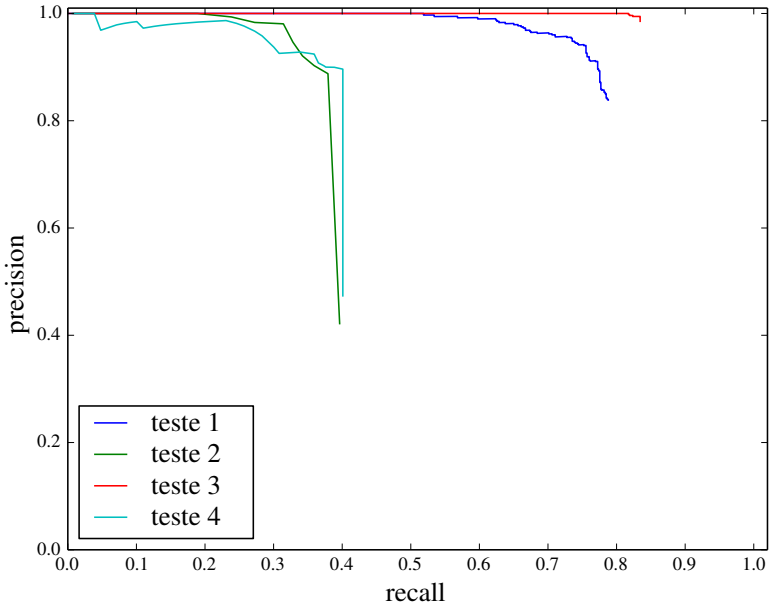
Tabela 6.1: OpenSeqSLAM2.0 parâmetros para a geração das curvas apresentadas na Figura 6.5.

	$searchMethod$	d_s
teste 1	$trajectory\text{-}base$	40
teste 2	$cone$	40
teste 3	$trajectory\text{-}base$	100
teste 4	$cone$	100

Figura 6.5: Testes sobre a variação de parâmetros para o OpenSeqSLAM2.0



(a) GPW-DR vs. GPW-NR



(b) UofA-D vs. UofA-E

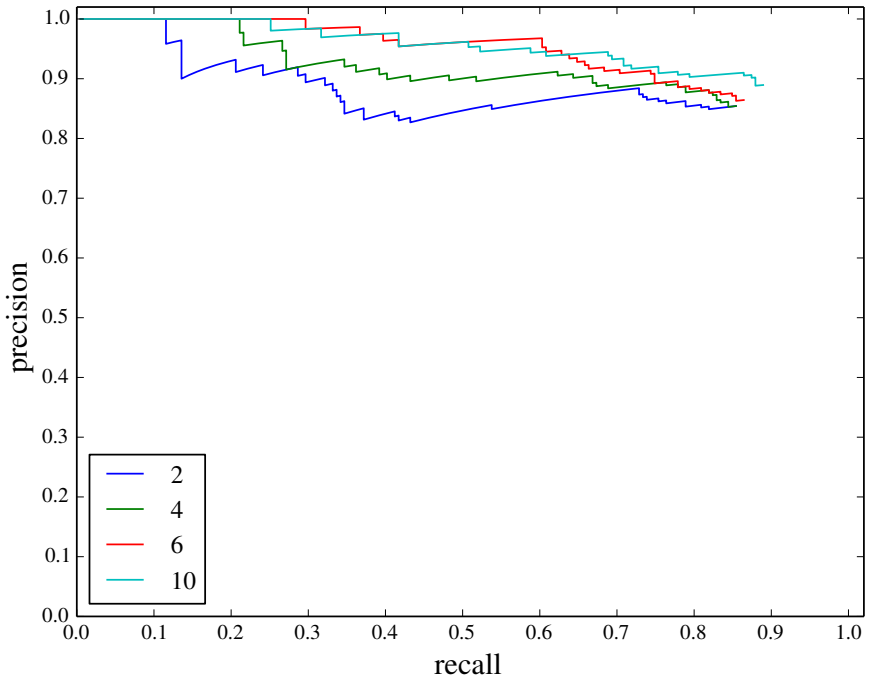
A Tabela 6.1 apresenta a variação dos parâmetros usados para a geração desses gráficos.

6.3.2 Definição de parâmetros para o método proposto

Nesta seção vamos definir os parâmetros usados para o método proposto. São eles, o número de candidatos por iteração (k) e o tamanho da janela que considera iterações passadas para a computação da combinação atual (w). A seguir serão apresentados experimentos realizados usando as variações desses parâmetros e o impacto nos resultados.

A Figura 6.6 mostra o efeito do uso de diferentes valores para k (2, 4, 6 e 10) nas curvas de *precision-recall*. Esse parâmetro está fortemente relacionado a qualidade do descritor utilizado. Por exemplo, considere uma observação e uma lista de candidatos a qual queremos selecionar o elemento mais similar à observação atual. Quando essa lista é ordenada de acordo com o valor de similaridade, é possível que o elemento em primeiro lugar não seja realmente aquele considerado correto como a melhor combinação para observação. Por isso a escolha dos k elementos mais similares aumenta a chance de seleção do elemento correto. A Figura 6.6 mostra que valores mais altos de k não necessariamente apresentam melhores resultados. Isso ocorre devido a inclusão de mais incerteza na computação quando mais candidatos são considerados.

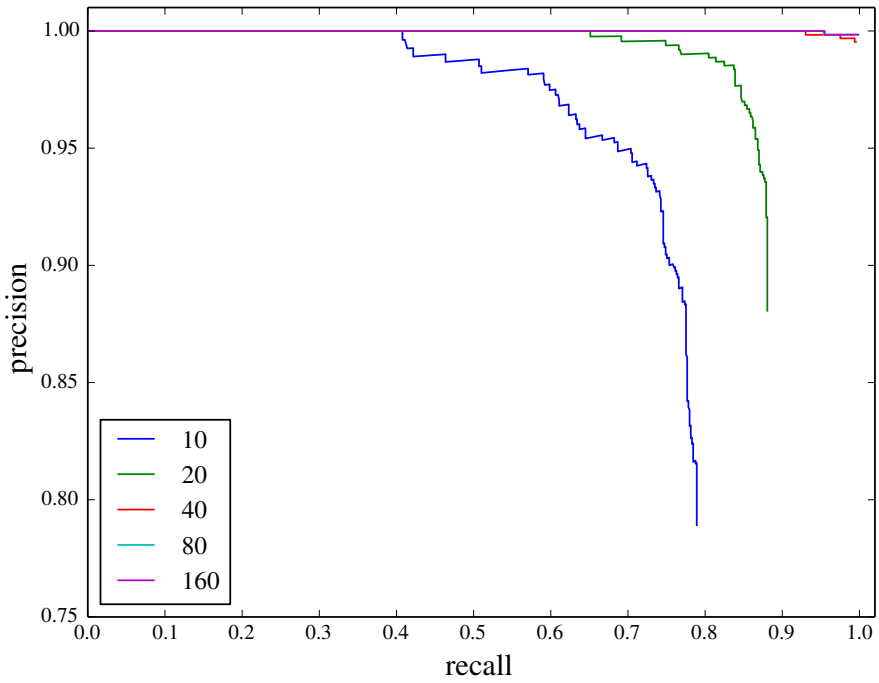
Figura 6.6: *Precision-recall* - variação do parâmetro k



Informação gerada usando o *dataset* GPW.

A Figura 6.7 mostra o efeito de diferentes valores do parâmetro w no resultado do

Figura 6.7: *Precision-recall* - variação do parâmetro w



Informação gerada usando o *dataset* UofA.

método. Onde w define quantas iterações passadas serão consideradas para a computação da iteração atual. A estratégia base do método é selecionar intervalos baseado em quantas vezes eles foram escolhidos como possíveis combinações no decorrer das iterações. Uma sequência de combinações próximas no decorrer das iterações é um forte indicativo de uma combinação correta. Assim como o parâmetro k, o uso de valores altos para w melhora os resultados até um dado limite, que quando alcançado apenas inclui mais incertezas ao processo.

Com os testes preliminares para análise dos parâmetros observamos que w em torno de 100 produziu resultados competitivos sem queda significativa no tempo de execução. Considerando o parâmetro k, percebemos que $k = 1$ não é suficiente e a melhor escolha para esse parâmetro pode variar de acordo com o *dataset*. No caso de regiões similares distribuídas ao longo do caminho percorrido pelo robô, é importante que a maioria delas seja representada por algum candidato, possivelmente exigindo valores mais altos para k. Isso aumenta as chances de escolher a combinação correta mais rapidamente quando uma região significativamente distinguível for analisada.

Além do comportamento apresentado nas Figuras 6.6 e 6.7, a escolha dos parâmetros também afeta o tempo de processamento necessário para a geração do resultado. Em resumo, a cada iteração temos kw intervalos sendo processados.

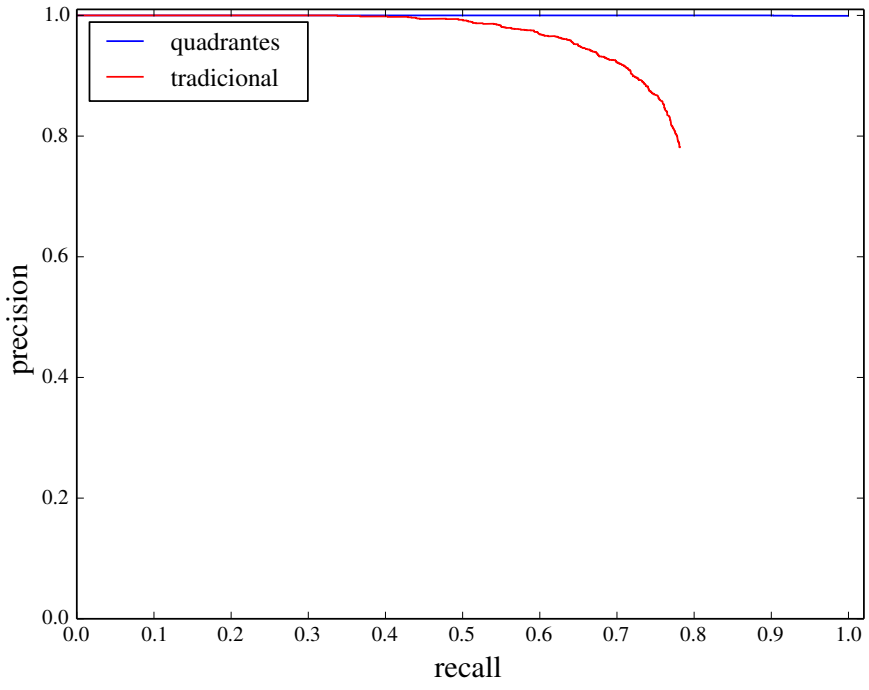
6.4 Avaliação da abordagem proposta

Nesta seção é apresentada uma avaliação dos resultados obtidos pela abordagem proposta para o problema de reconhecimento visual de regiões. Comparamos o método proposto com o OpenSeqSLAM2.0 e outros métodos da literatura. Também é apresentada uma comparação entre o descritor LDB e a variação proposta usando a descrição por quadrantes.

6.4.1 Avaliação do descritor LDB modificado

Iniciamos as comparações apresentando a diferença entre o uso do descritor LDB tradicional e a modificação proposta usando a descrição por quadrantes, apesentada no Capítulo 5.2. As Figuras 6.8 e 6.9 mostram a diferença na curva de *precison-recall* usando as duas abordagens em duas configurações do *dataset* Nordland, Nord-Spr vs. Nord-Win e Nord-Sum vs. Nord-Win. Testes foram executados nos outros *datasets* e apresentaram resultados muito similares.

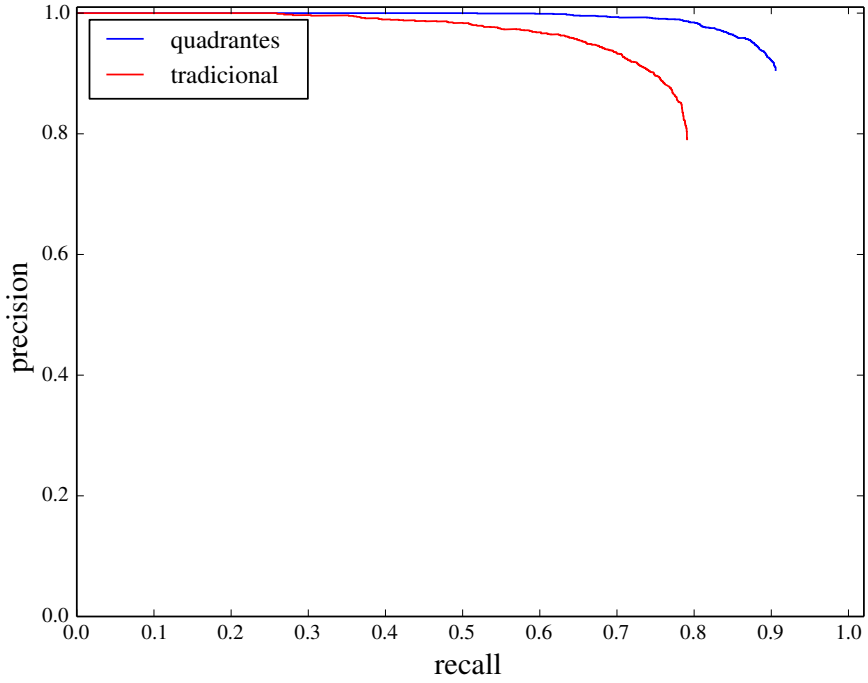
Figura 6.8: *Precision-recall* - comparação do descritor LDB usando Nord-Spr vs. Nord-Win



As curvas mostram que o método obtém melhores resultados usando a variação

proposta para o descritor. Isso era esperado, uma vez que a abordagem usando a descrição por quadrantes inclui mais informação para o descritor final. A descrição separada de cada quadrante da imagem reduz a granularidade usada pelo LDB e de acordo com Yang e Cheng (2014) o uso de grades com baixa granularidade, i.e., a divisão da imagem em partes menores, permite a captura mais detalhada de padrões, aumentando a capacidade de distinção. Além disso o uso da abordagem proposta é útil em casos de oclusão parcial, por exemplo. Entretanto, quanto mais divisões são feitas na imagem, maior é o custo de armazenamento.

Figura 6.9: *Precision-recall* - comparação do descritor LDB usando Nord-Sum vs. Nord-Win



A Figura 6.10 apresenta os mapas de calor gerados usando a variação proposta para o descritor nos *datasets* selecionados. Os mapas mostram em escala de cinza as distâncias de hamming entre as imagens, onde quanto menor a distância (mais semelhantes são as imagens) mais escura é a cor pra representá-la. O *groundtruth* de todos os mapas é uma linha reta partindo do canto superior esquerdo até o canto inferior direito.

A Figura 6.10b apresenta o mapa de calor baseado nas travessias GPW-DR e GPW-NR e a Figura 6.10a é baseado nas travessias GPW-DL e GPW-NR. Em ambos os mapas é possível distinguir áreas mais escuras, o canto superior direito é um exemplo. Ainda que exista a indicação de alta similaridade, essa área representa uma falsa correspondência já que o *groundtruth* está na diagonal do mapa.

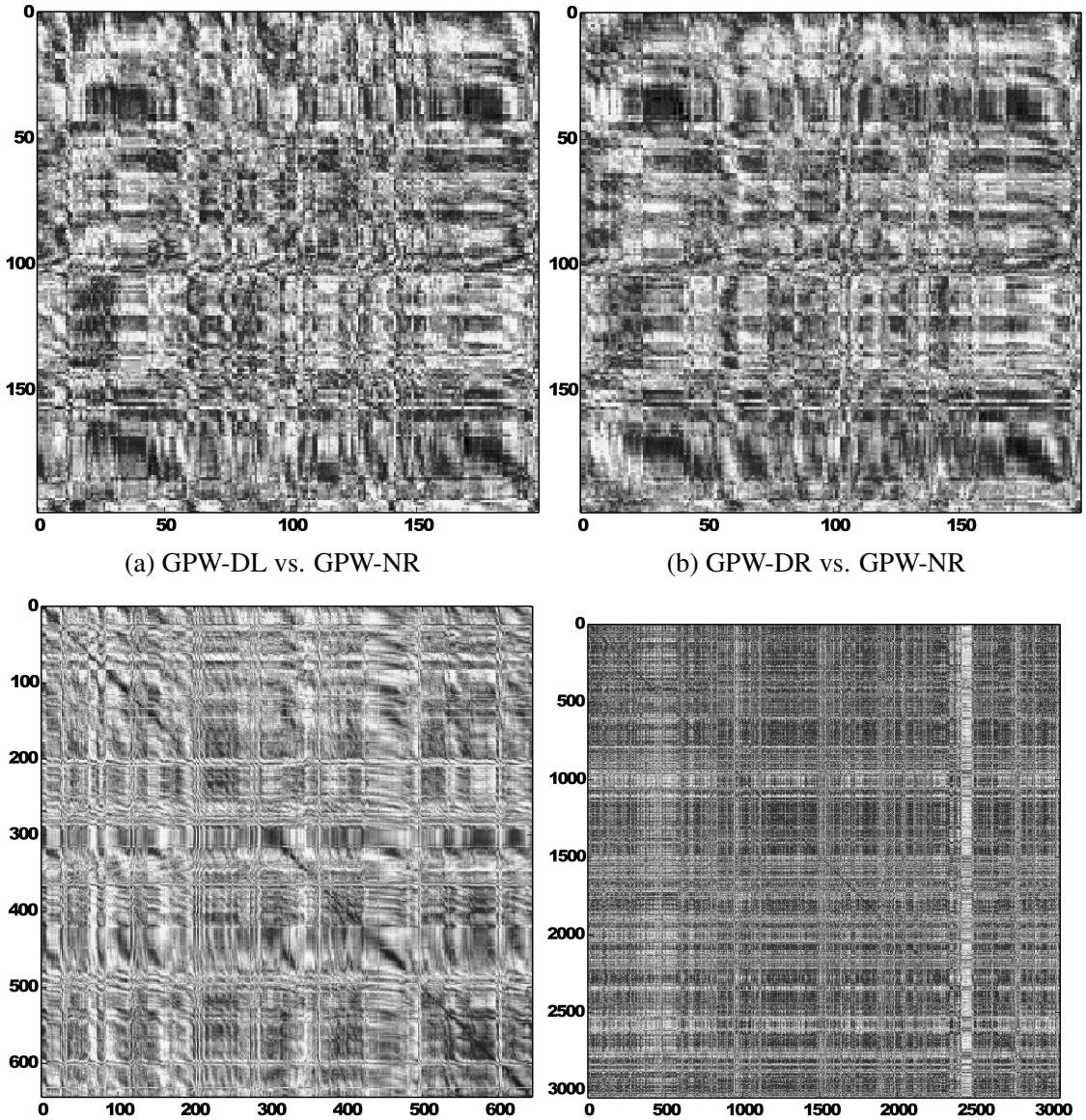
O mapa da Figura 6.10c representa as distâncias entre as imagens das travessias UofA-D e UofA-E. A diagonal representando as correspondências corretas é mais fácil de perceber do que as geradas com o *dataset* GPW. Entretanto ainda é possível perceber regiões indicando alta similaridade longe do que sabemos ser correto.

A Figura 6.10d apresenta o mapa de calor entre as travessias Nord-Spr e Nord-Win. Apesar da significativa diferença entre as travessias, devido à mudança de estação, podemos perceber claramente uma linha mais escura próxima ao que seria o *ground-truth*. Falhas nas diagonais onde é o *groundtruth* também são comuns em todos os mapas. Mesmo que as duas imagens sejam coletadas aproximadamente do mesmo lugar e ângulo, elas podem apresentar baixa similaridade entre elas. Isso pode ser causado pela alteração na iluminação, gerando novas sombras, ou por oclusões parciais causadas pelo dinamismo natural do ambiente, pela movimentação de pessoas ou veículos.

6.4.2 Avaliação do método proposto para o problema de reconhecimento visual de regiões

A partir deste ponto todos os resultados apresentados foram obtidos empregando a forma de descrição proposta usando quadrantes. A seguir vamos apresentar a comparação de resultados obtidos usando o método baseado na modelagem proposta e o OpenSeqSLAM2.0. A Tabela 6.2 resume as informações descritas na Seção 6.3 mostrando os parâmetros selecionados para a execução dos experimentos. Note que para GPW-DL vs. GPW-NR foi necessária uma configuração de parâmetros diferente das usadas nos demais *datasets*. Isso acontece devido principalmente à diferença de iluminação associada à diferença de ponto de vista.

Figura 6.10: Mapa de calor comparando travessias



(a) GPW-DL vs. GPW-NR

(b) GPW-DR vs. GPW-NR

(c) UofA-D vs. UofA-E

(d) Nord-Spr vs. Nord-Win

Tabela 6.2: Parâmetros usados durante a execução dos experimentos

	UofA	**GPW**		**NORD**
		DR vs. NR	DL vs. NR	
Método Proposto				
w	100	100	200	100
k	6	6	20	6
OpenSeqSLAM2.0				
R_{window}	13	4	4	61
d_s	100	100	100	20
$searchMethod$	trajectory	cone	trajectory	trajectory
$matchingMethod$		thresholding		

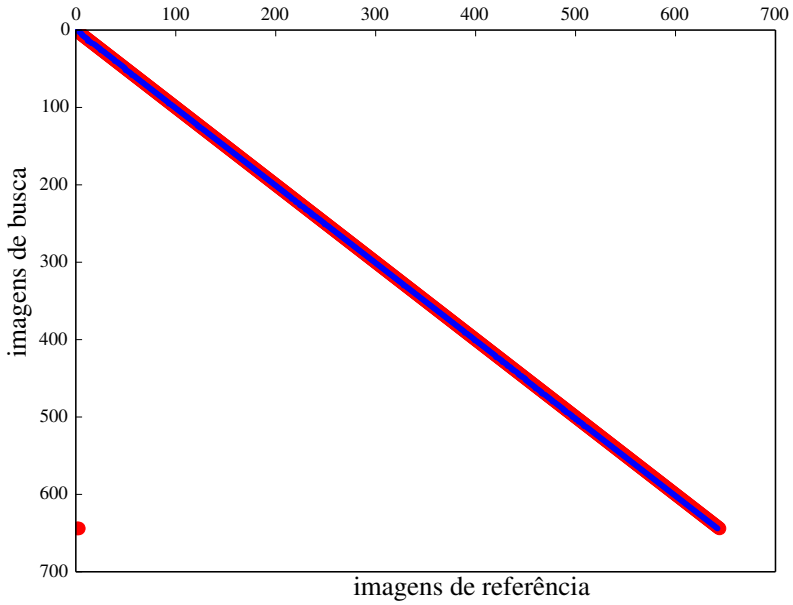
As principais informações apresentadas a seguir são as matrizes de combinação

e as curvas de *precision-recall*. As matrizes de combinação mostram quais foram as combinações escolhidas para cada imagem do *dataset* e qual seria a combinação correta. Assim, podemos observar a quantidade de acertos e o quão distantes estão as combinações selecionados pelo método em relação a resposta correta. As curvas de *precision-recall* são comumente usadas para a comparação de métodos. Elas facilitam a visualização de quantos resultados corretos foram possíveis de recuperar sem a inclusão de erros, i.e. com 100% de precisão.

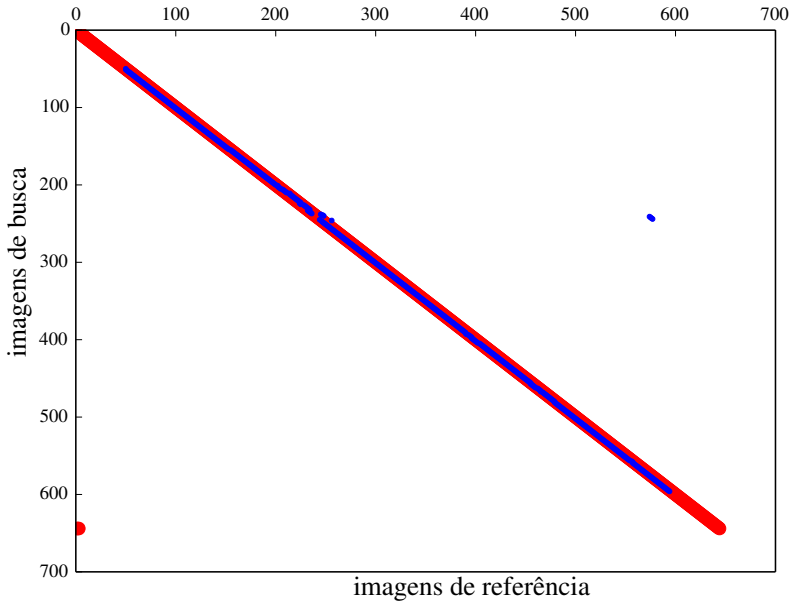
As Figuras 6.11 e 6.12 mostram as combinações selecionadas usando o método proposto e o OpenSeqSLAM2.0. Onde as imagens usadas como referência são apresentada no eixo horizontal e as imagens de busca no eixo vertical. Os pontos vermelhos representam o *groundtruth* e os azuis as combinações selecionadas pelos métodos. O *dataset* UofA possui dois conjuntos de imagens, um conjunto coletado durante o dia e outro ao entardecer. As Figuras 6.11a e 6.11b usam as imagens coletadas durante o dia como referência e as imagens coletadas ao entardecer como busca, o contrário é apresentado nas Figuras 6.12a e 6.12b.

Apesar das diferenças apresentadas no *dataset* UofA, principalmente de ilumina-ção, ambos os métodos apresentaram bons resultados, i.e., uma linha quase perfeita em consonância com o *groundtruh*. Ainda assim, é possível perceber uma falha no início e no fim da linha de combinações quando usamos o OpenSeqSLAM2.0. Isso ocorre devido a forma como o método busca por combinações, ele usa uma janela de busca, onde o elemento a ser combinado fica no centro dessa janela. Assim, usando um $d_s = 100$, ele não gera combinações para as primeiras e últimas 50 imagens de busca.

Figura 6.11: Combinações - UofA-D vs. UofA-E



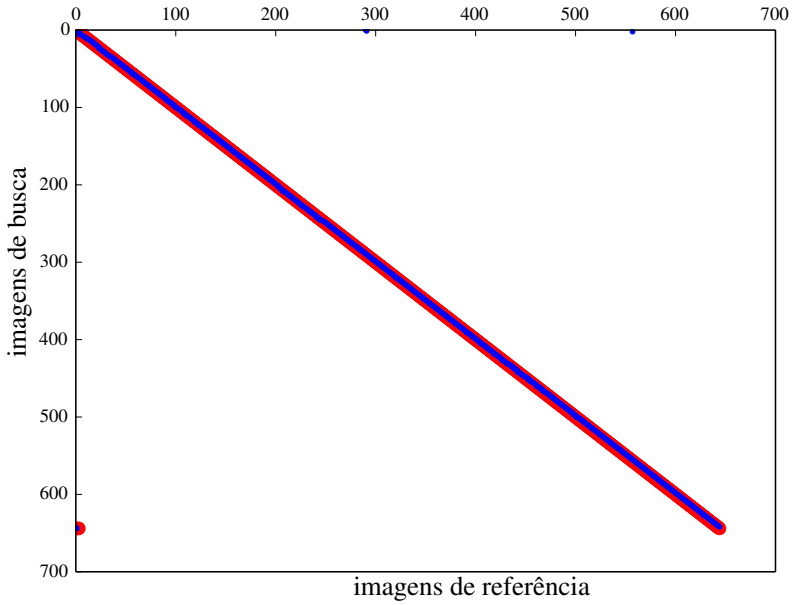
(a) Método proposto



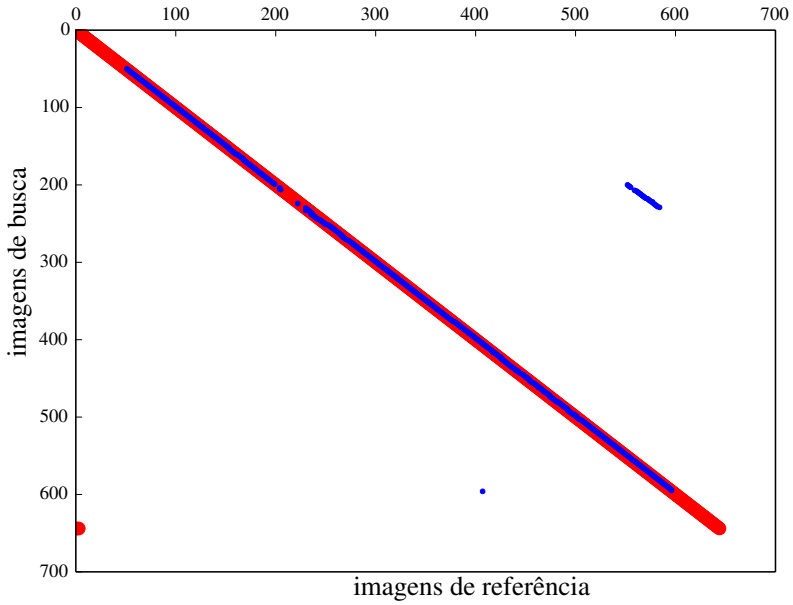
(b) OpenSeqSLAM2.0

Groundtruth em vermelho e combinações selecionadas em azul.

Figura 6.12: Combinações - UofA-E vs. UofA-D



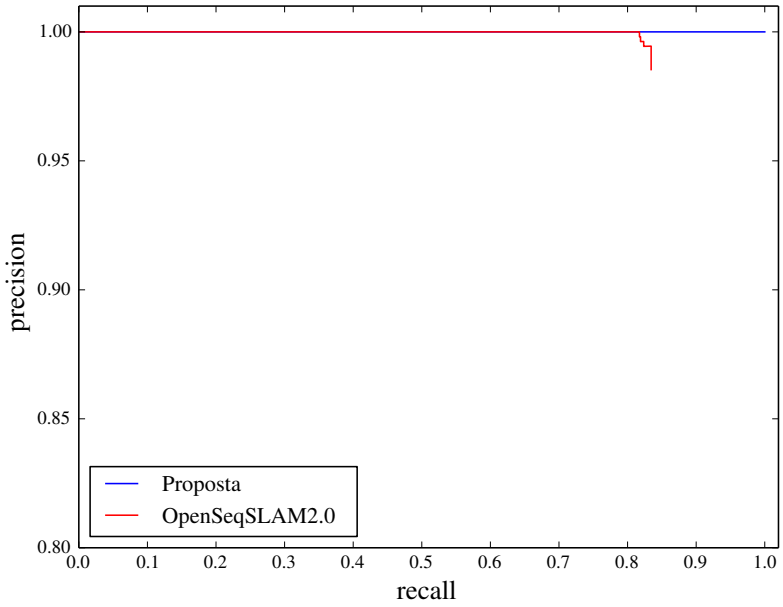
(a) Método proposto



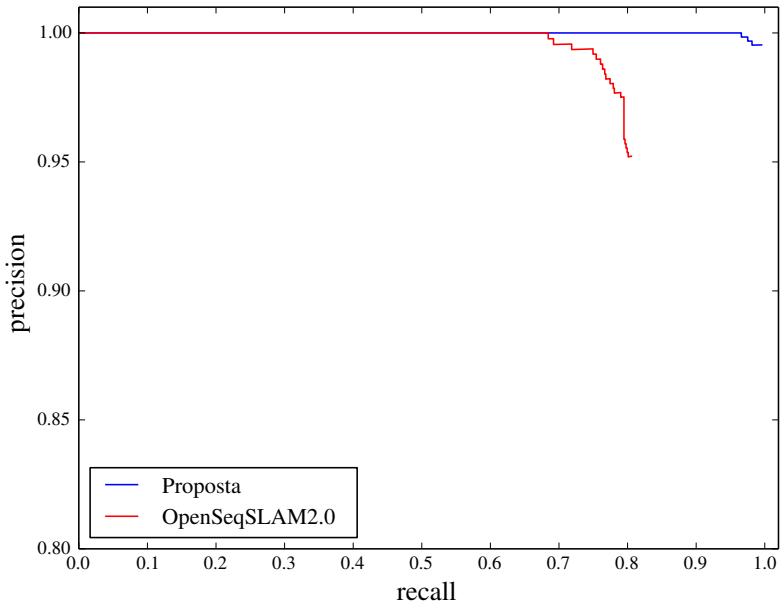
(b) OpenSeqSLAM2.0

Groundtruth em vermelho e combinações selecionadas em azul.

Figura 6.13: *precision-recall - dataset* UofA



(a) UofA-D vs. UofA-E



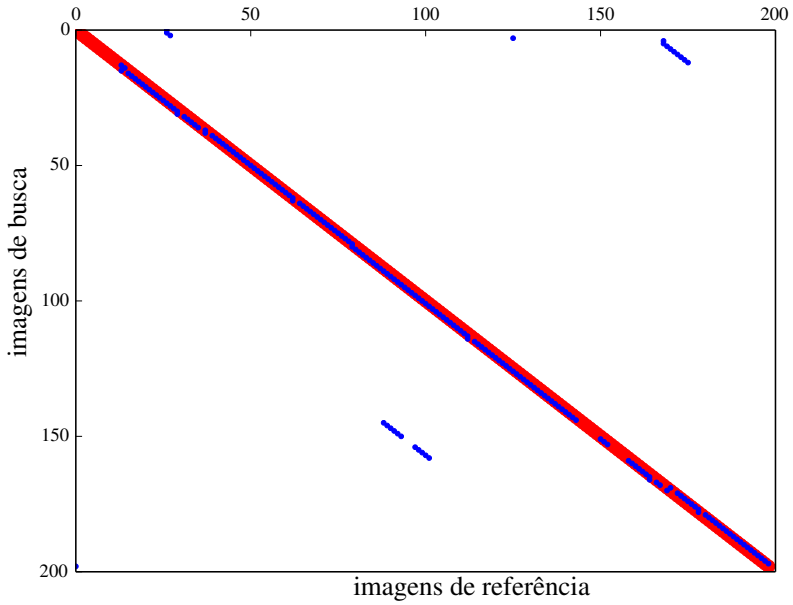
(b) UofA-E vs. UofA-D

A Figura 6.13 apresenta as curvas de *precision-recal* do método proposto em comparação com o OpenSeqSLAM2.0 usando o *dataset* UofA. Na Figura 6.13a estão as curvas usando UofA-D como referência e UofA-E como busca. Nesta combinação o método proposto obteve uma linha perfeita, obtendo 100% de *recall* com 100% de *precision*. O OpenSeqSLAM2.0 também obteve bons resultados, chegando a 82% de *recall* com 100%

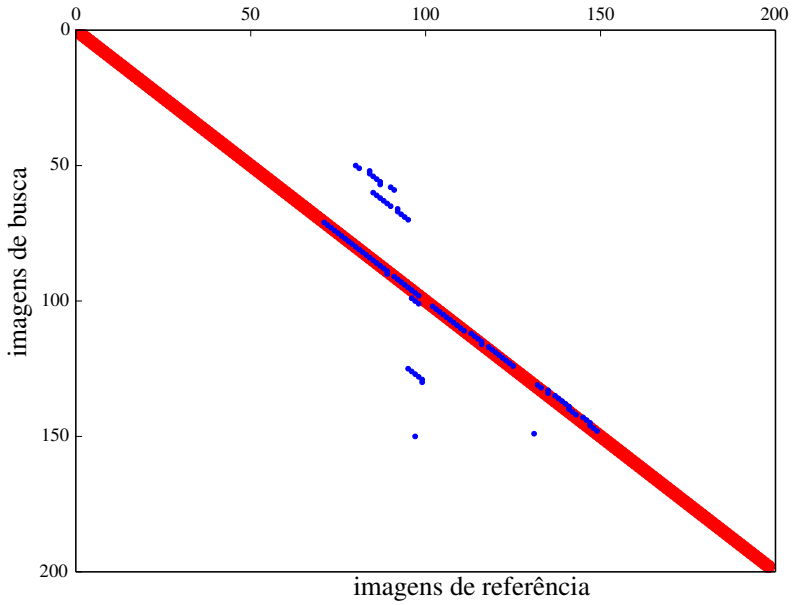
de *precision*. Na Figura 6.13b são apresentadas as curvas usando UofA-E como referência e UofA-D como busca. Nessa configuração o OpenSeqSLAM2.0 obteve 68% de *recall* com 100% de *precision* enquanto o método proposto alcançou chegou aos 96% de *recall*. A configuração mais comum vista na literatura é usando a travessia do dia como referência, e podemos ver nas curvas de *precision-recall* que o uso dessa configuração obtém melhores resultados. Usando a configuração contrária, a queda do *recall* chegou em torno de 18% para a abordagem proposta e 29% para o OpenSeqSLAM2.0. O uso de conjuntos de referência com imagens com menos informação, como é o caso do UofA-E que possui imagens escuras coletadas quase à noite, tende a gerar mais erros de combinação. Isso pode ocorrer porque é mais difícil procurar por uma imagem mais descritiva em meio a um conjunto pouco descritivo que o contrário.

As Figuras 6.14 e 6.15 mostram as combinações selecionadas pelos métodos usando o *dataset* GPW. Nesse experimento temos uma grande variação de iluminação entre os conjuntos de imagens de referência e busca, isso reflete nos resultados obtidos. Podemos ver alguns trechos significativamente distantes do *groundtruth* (linha vermelha) nas Figuras 6.14a e 6.14b.

Figura 6.14: Combinações - GPW-DR vs. GPW-NR



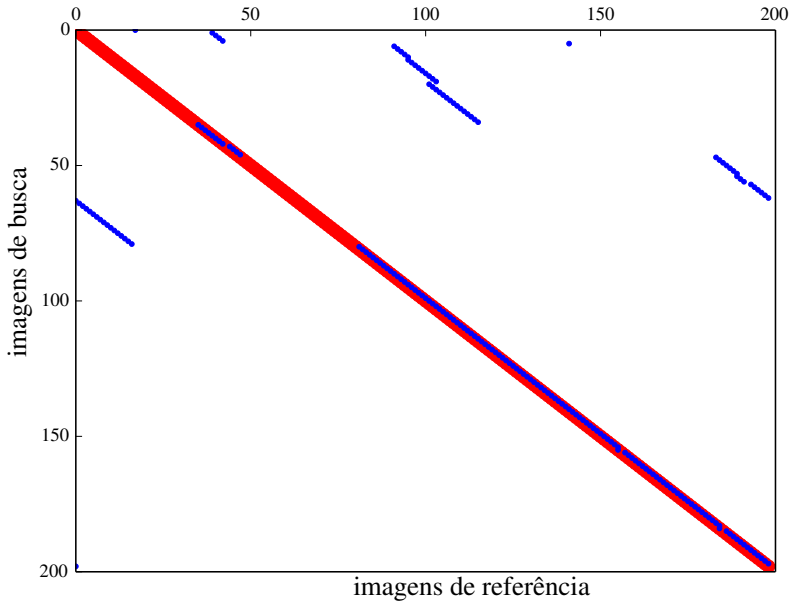
(a) Método proposto



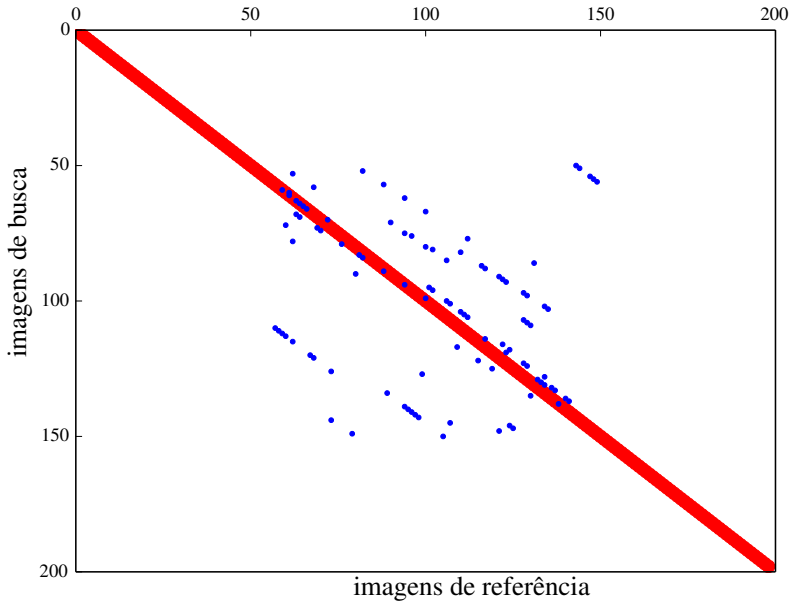
(b) OpenSeqSLAM2.0

Groundtruth em vermelho e combinações selecionadas em azul.

Figura 6.15: Combinações - GPW-DL vs. GPW-NR



(a) Método proposto



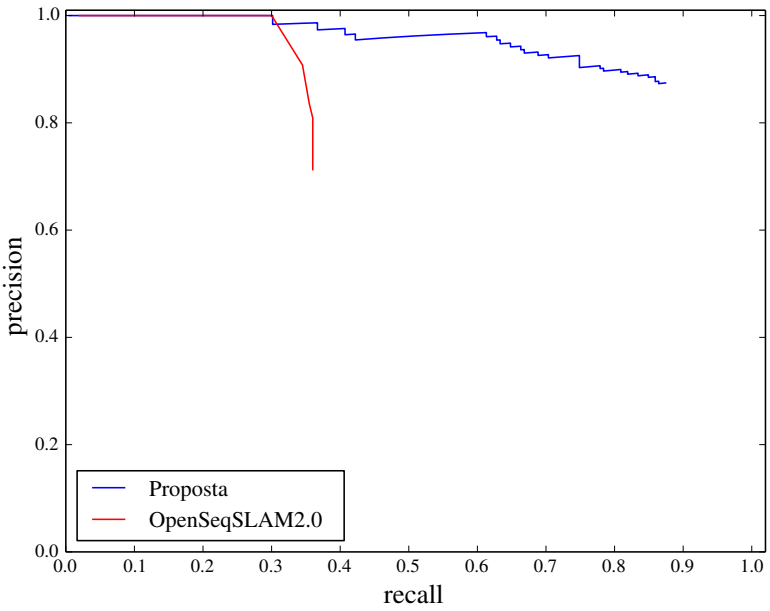
(b) OpenSeqSLAM2.0

Groundtruth em vermelho e combinações selecionadas em azul.

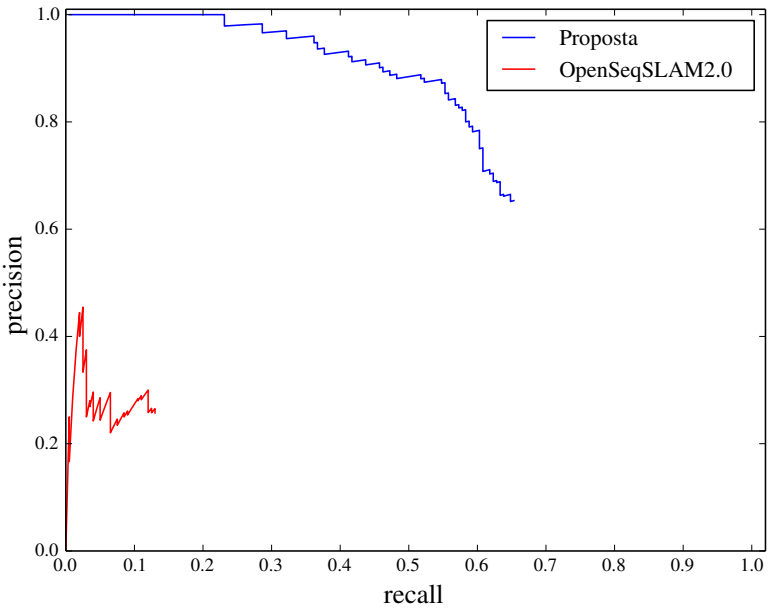
Esses erros são ainda mais evidentes nas Figuras 6.15a e 6.15b, onde o GPW-DL foi usado como referência e o GPW-NR como busca, que além de apresentarem uma grande diferença de iluminação entre eles também possuem uma pequena variação de ponto de vista. Fica claro com os resultados das Figuras 6.14 e 6.15 que o método pro-

posto conseguiu selecionar mais combinações em concordância com o *groundtruth* que o OpenSeqSLAM2.0.

Figura 6.16: *precision-recall - dataset* GPW



(a) GPW-DR vs. GPW-NR



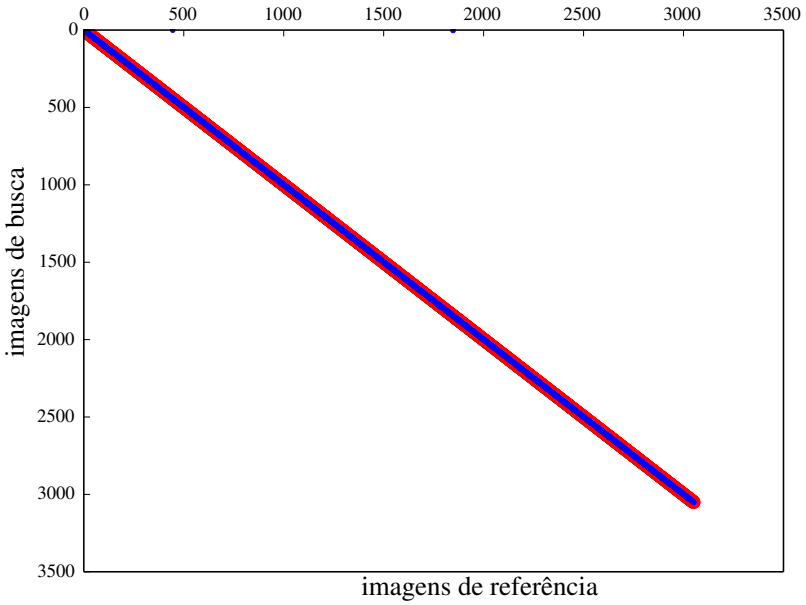
(b) GPW-DL vs. GPW-NR

As curvas de *precision-recall* usando o *dataset* GPW são presentadas na Figura 6.16. Ambos os métodos apresentaram um resultado pior quando comparados com aqueles obtidos usando *dataset* UofA. Considerando o uso de GPW-DR vs. GPW-NR os dois

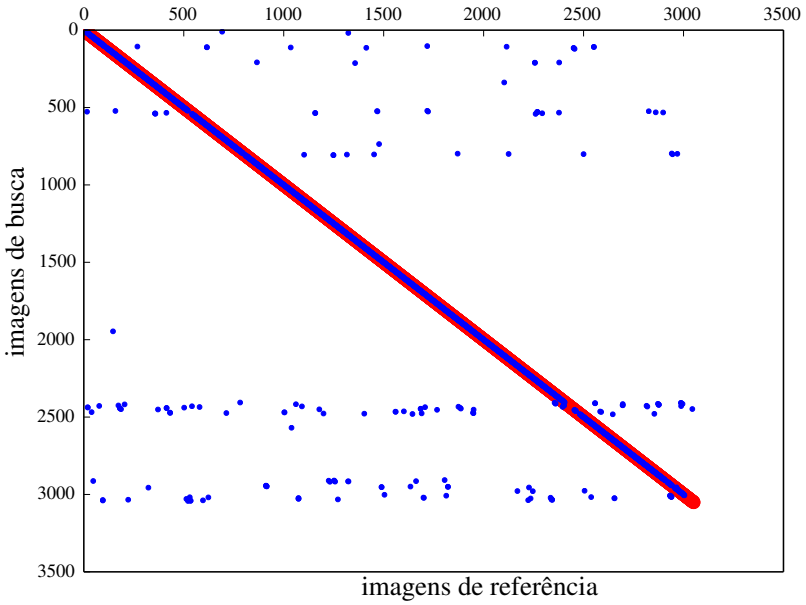
métodos obtiveram aproximadamente 30% de *recall* com 100% de *precision*. Contudo, o método proposto se manteve acima de 85% de *precision* todo o tempo. Usando os conjuntos GPW-DL vs. GPW-NR, nosso método obteve 23% de *recall* com 100% de *precision*, e o OpenSeqSLAM2.0 obteve no maior pico, não mais que 45% de *precision* com *recall* inferior a 1%, nunca alcançando os 100% de *precision*.

As Figuras 6.17 e 6.18 apresentam as combinações encontradas usando o *dataset* Nordland. Esse *dataset* é mais longo que os demais (3052 imagens) e apresenta mudanças no ambiente que acontecem no decorrer das estações. As Figuras 6.17a e 6.17b mostram as combinações selecionadas pelo método proposto e pelo OpenSeqSLAM2.0 usando o conjunto de imagens Nord-Spr como referência e Nord-Win como busca. As Figuras 6.18a e 6.18b apresentam as combinações selecionadas usando o conjunto de imagens Nord-Sum como referência e Nord-Win como busca. Ambos os métodos mostram resultados próximos ao *groudtruth* (vermelho).

Figura 6.17: Combinações - Nord-Spr vs. Nord-Win



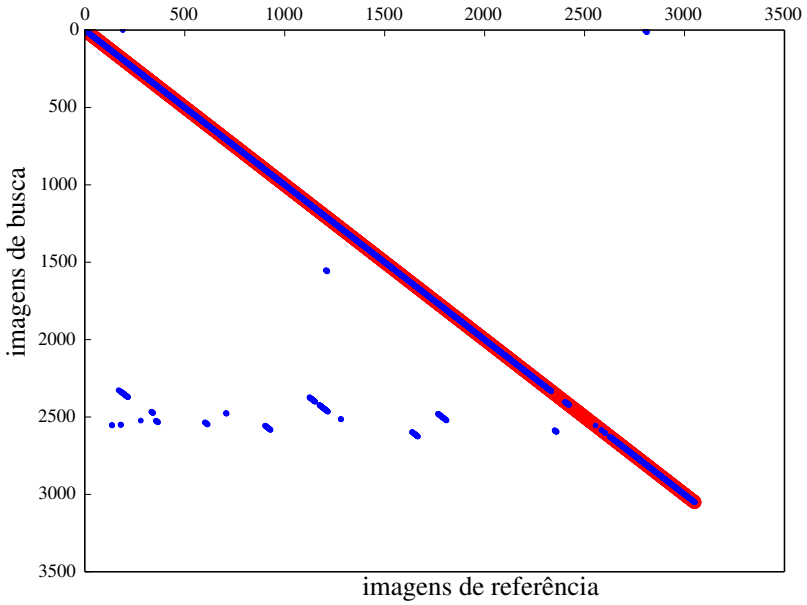
(a) Método proposto



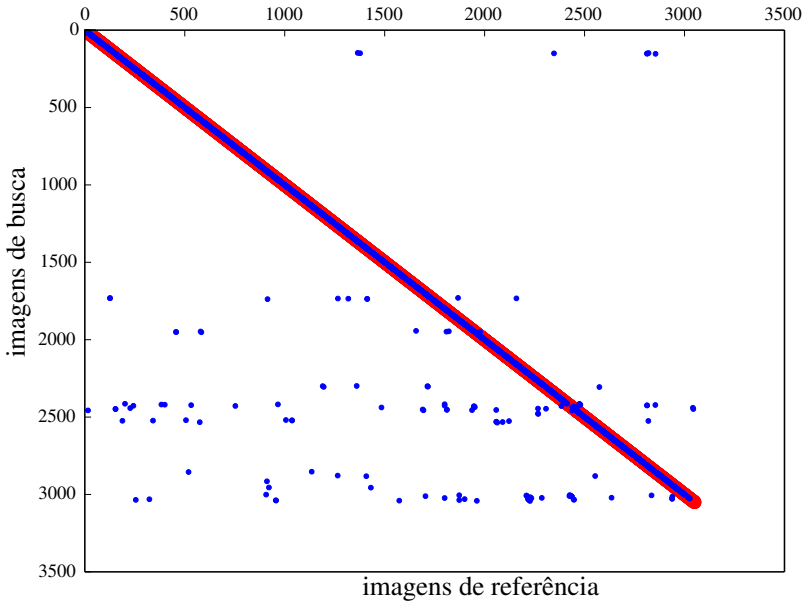
(b) OpenSeqSLAM2.0

Groundtruth em vermelho e combinações selecionadas em azul.

Figura 6.18: Combinações - Nord-Sum vs. Nord-Win



(a) Método proposto



(b) OpenSeqSLAM2.0

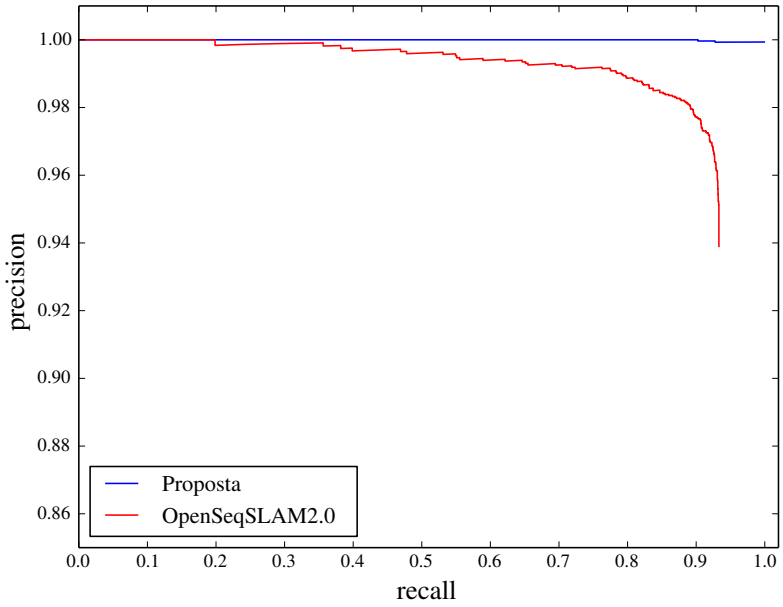
Groundtruth em vermelho e combinações selecionadas em azul.

A Figura 6.19 mostra as curvas de *precision-recall* para o *dataset* Nordland. A Figura 6.19a mostra as curvas de *precision-recall* usando as travessias Nord-Spr vs. Nord-Win. O método proposto obteve 90% de *recall* com 100% de *precision* e mantém mais de 99% de *precision* para os demais valores de *recall*. O OpenSeqSLAM2.0 obteve 20% de

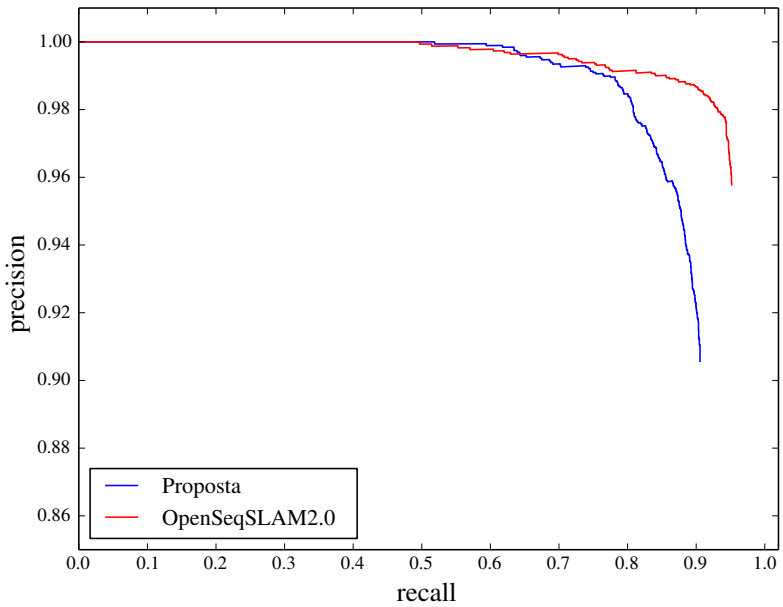
recall com 100% de *precision*, mantendo a *precision* maior que 99% até 79% de *recall*.

As curvas criadas com as travessias Nord-Sum vs. Nord-Win são apresentadas na Figura 6.19b. O método proposto conseguiu alcançar 52% de *recall* com 100% de *precision*, mas mantém a *precision* em 99% até 79% de *recall*. Além disso, a precisão nunca cai abaixo dos 90%. Em comparação, o OpenSeqSLAM2.0 obteve 50% de *recall* com 100% de *precision* e mantém 99% de *precision* até 85% de *recall*.

Figura 6.19: *precision-recall* - *dataset* Nordland



(a) Nord-Spr vs. Nord-Win

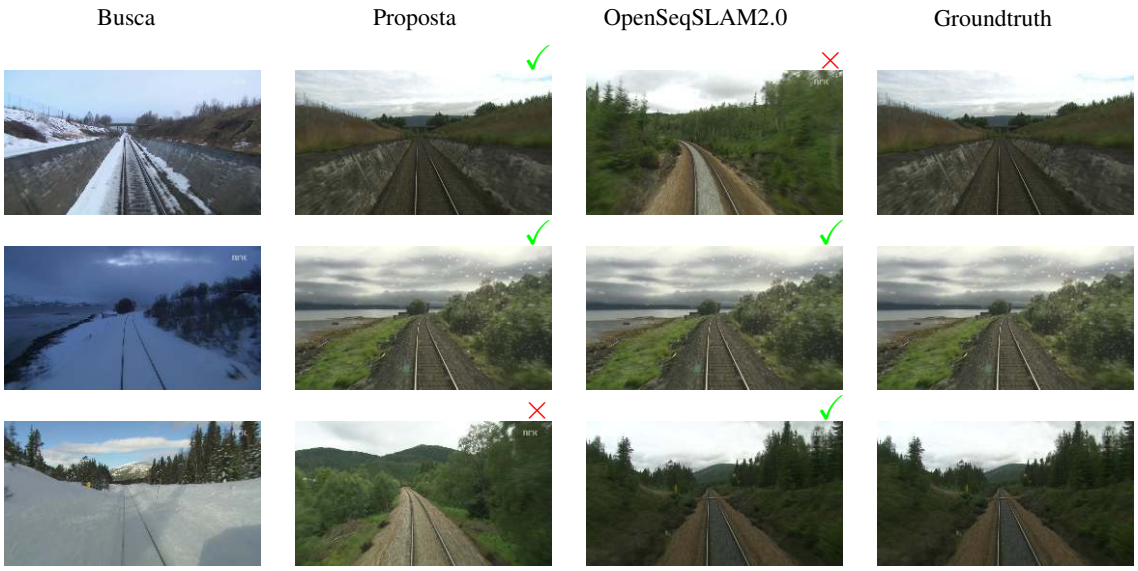


(b) Nord-Sum vs. Nord-Win

As Figuras 6.20 e 6.21 apresentam uma amostra das combinações feitas pela abordagem proposta e pelo OpenSeqSLAM2.0 para uma avaliação qualitativa. Cada linha apresenta uma combinação, sendo que a primeira coluna apresenta a observação atual, a segunda e terceira apresentam respectivamente a combinação selecionada pela abordagem proposta e pelo OpenSeqSLAM2.0, e a quarta mostra qual a combinação correta de acordo com o *groundtruth*.

Figura 6.20: Comparação qualitativa - GPW-DR vs. GPW-NR

Figura 6.21: Comparação qualitativa - Nord-Sum vs. Nord-Win



No canto superior direito das imagens correspondentes às combinações existe um símbolo indicando se a combinação foi considerada correta (✓) ou não (×). Podemos

perceber que mesmo em combinações erradas existe uma certa coerência na disposição dos elementos na cena.

A Tabela 6.3 apresenta uma comparação entre alguns métodos publicados nos últimos anos. Os valores na tabela representam o *recall* aproximado com 100% de *precision*. Nós executamos os experimentos usando o nosso método e o código disponível do OpenSeqSLAM2.0. Os valores associados aos demais métodos apresentados foram extraídos diretamente dos artigos. Contudo, nem todos os trabalhos referenciados tem um conjunto de experimentos comparáveis com o que nós mostramos nesta pesquisa. Esses casos estão marcados com um traço (-).

Tabela 6.3: Comparação entre métodos avaliando *recall* a 100% de *precision*

	UofA		Nord		GPW	
	D vs. E	E vs. D	Sum vs. Win	Spr vs. Win	DL vs. NR	DR vs. NR
Abordagem proposta	100%	96%	52%	90%	23%	30%
OpenSeqSLAM	48%	–	–	–	42%	–
Fast-SeqSLAM	41%	–	–	93%	61%	–
OpenSeqSLAM2.0	82%	68%	50%	20%	0	30%
SeqCNNSLAM	–	–	–	97%	100%	–
Naseer	–	–	0	–	–	30%
Merril	–	–	–	–	10%	–
ConvSequential-SLAM	–	–	10%	–	–	21%

Os valores apresentados na Tabela 6.3 estão relacionados aos seguintes trabalhos: OpenSeqSLAM (SÜNDERHAUF; NEUBERT; PROTZEL, 2013), Fast-SeqSLAM (SIAM; ZHANG, 2017), OpenSeqSLAM2.0 (TALBOT; GARG; MILFORD, 2018), SeqCNNSLAM (BAI et al., 2018), Naseer (NASEER; BURGARD; STACHNISS, 2018), Merril (MERRILL; HUANG, 2018) e ConvSequential-SLAM (TOMITĂ et al., 2020).

6.4.3 Custo Computacional

O custo computacional para métodos de reconhecimento de regiões pode ser um limitador para algumas aplicações. Nesse contexto, o principal ponto de atenção está relacionado ao tamanho do conjunto de intervalos processados a cada iteração. O qual é dado pelos parâmetros k e w discutidos na Seção 6.3.2, onde kw é o número de intervalos processados a cada iteração. Lembrando que os parâmetros utilizados nos testes são apresentados na Tabela 6.2.

A Tabela 6.4 apresenta uma comparação relacionada ao tempo de execução de cada método usando as diferentes configurações descritas na Seção 6.3. Todos os experimentos foram executados em um computador com processador Intel Core i7 e 8GRAM.

Precisamos ressaltar alguns detalhes relacionados a como foi feita a comparação de tempo entre os métodos. O método proposto foi implementado em C++ e os tempos apresentados incluem o tempo de carregamento dos descritores já pré-computados.

Em relação ao OpenSeqSLAM2.0, usamos o código implementado em Matlab que está disponível para *download*[2]. Para marcar o tempo de execução, circundamos a função $OpenSeqSLAMRun()$ com um cronômetro. Criar a matriz de diferença e processar as buscas nessa matriz são processos custosos, e o uso de valores altos para o parâmetro d_s apesar de melhorar os resultados, aumenta o tempo de computação.

Tabela 6.4: Tempo aproximado total para a execução dos métodos (mm:ss)

	GPW-DR vs. GPW-NR	GPW-DL vs. GPW-NR	UofA	Nord
Método proposto	00:03	00:04	00:22	12:18
OpenSeqSLAM2.0	00:52	00:41	15:25	31:14

Considerando os resultados apresentados nesse capítulo e os tempos de execução na Tabela 6.4 podemos ver que o método proposto apresenta resultados competitivos para o problema de reconhecimento de regiões.

[2]https://michaelmilford.com/seqslam/

7 CONCLUSÕES

O objetivo deste trabalho foi investigar o uso de análise intervalar para o tratamento do problema de reconhecimento visual de regiões em um contexto onde a informação disponível não é métrica, nem está ancorada em um sistema de coordenadas do mundo. Para isso, foi proposta uma nova modelagem baseada na teoria de análise intervalar para a representação do mundo observado através de intervalos.

Antes de apresentarmos nossa teoria, mostramos alguns conceitos básicos sobre aritmética intervalar (ver Capítulo 2) que são importantes para entendimento da modelagem proposta. Mostramos também trabalhos que usam análise intervalar pra lidar com problemas da robótica móvel.

No Capítulo 3 são apresentados detalhes sobre o problema de reconhecimento visual de regiões, bem como as descrições dos principais módulos relacionados a sistemas para lidar com esse problema (LOWRY et al., 2016). Os módulos essenciais são processamento de imagem, mapa e gerador de crença. Nesse capítulo também destacamos alguns trabalhos relacionados ao nosso, incluindo os principais marcos na literatura da área, FAB-MAP (CUMMINS; NEWMAN, 2008) e SeqSLAM (MILFORD; WYETH, 2012).

No Capítulo 4 apresentamos uma nova modelagem baseada em análise intervalar que além de proporcionar uma representação compacta do mundo é capaz de conter observações não métricas e não ancoradas em um sistema de coordenadas do ambiente em que o robô está inserido. Para validação dessa modelagem, nós propomos um novo método para o problema de reconhecimento visual de regiões usando apenas informações sobre a aparência do ambiente, extraídas por uma câmera monocular, sem qualquer informação sobre a posição no mundo de onde foi extraída. O método modela o mundo conhecido através de intervalos unidimensionais que representam porções do mundo através de amostras dele. Além disso, ele busca por resultados baseados em um conjunto de restrições temporais e de similaridade. A cada iteração, ele computa um conjunto de soluções candidatas, cada candidata aponta para uma determinada região considerada similar à observação atual do robô. Assim, mesmo que a combinação correta não seja a mais similar, ela ainda pode estar entre as soluções candidatas.

Nós buscamos por uma resposta não apenas no conjunto de soluções candidatas relacionadas a última coleta, mas numa janela de iterações passadas. Cada candidata nessa janela é deslocada a fim de simular a movimentação do robô. Nesse novo conjunto,

usamos a operação de intersecção relaxada, baseada na premissa de que o intervalo demarcando a região correta deve ter sido selecionado como solução candidata mais vezes durante essa janela do que quaisquer outros, formando uma sequência de intervalos que se interseccionam no decorrer das iterações.

A aplicação para modelagem foi o problema de reconhecimento visual de regiões usando imagens provenientes de câmeras monoculares. Entretanto, o método pode ser usado como um *framework* podendo ser adaptado para diferentes tipos de observações além de imagens, tais como sonar, laser ou *beacons*. Além disso, nós optamos pelo uso do descritor LDB, contudo, a abordagem proposta aceita qualquer descritor binário. O uso de um descritor não binário também é possível desde que alguns pontos sejam adaptados. Por exemplo, seria necessário uma nova representação global para os intervalos gerados, assim como uma nova função de similaridade.

Apresentamos também os resultados dos experimentos, para os quais foram utilizados *datasets* públicos, que apresentam conjuntos de imagens com alteração de iluminação e variação no ponto de vista. Como resultado dos experimentos, nossa abordagem apresentou uma excelente taxa de acerto na busca por correspondências entre conjuntos de imagens, mesmo com as significativas mudanças perceptuais.

Usando UofA-D como referência e UofA-E como busca o método proposto obteve um resultado perfeito, obtendo 100% de *recall* com 100% de *precision*. O OpenSeqSLAM2.0 também obteve bons resultados, chegando a 82% de *recall* com 100% de *precision*. Considerando o uso de GPW-DR vs. GPW-NR o os dois métodos obtiveram aproximadamente 30% de *recall* com 100% de *precision*. Contudo, o método proposto se manteve acima de 85% de *precision* todo o tempo. Usando o *dataset* Nord-Summer vs. Nord-Winter o método proposto conseguiu alcançar 52% de *recall* com 100% de *precision*, mas mantém a *precision* a 99% até 79% de *recall*. Além disso, a precisão nunca cai abaixo dos 90%. Em comparação, o OpenSeqSLAM2.0 obteve 50% de *recall* com 100% de *precision* e mantém 99% de *precision* até 85% de *recall*. Mais detalhes sobre os experimentos são descritos no Capítulo 6.

Com o uso de análise de intervalos conseguimos reduzir parte do espaço de busca de soluções através de computações puramente intervalares, i.e., sem o uso de comparações entre imagens, o que contribui para uma busca otimizada. Com as configurações usadas durante os experimentos apresentados, nosso método foi capaz de produzir resultados em um tempo promissor para uso em tempo real em um computador Intel Core i7 com 8GiB RAM.

A partir dessa pesquisa destacamos as seguintes contribuições:

- Investigação e documentação sobre o uso de intervalos para problemas da robótica móvel, em especial o problema de reconhecimento visual de regiões, em situações onde as observações coletadas não contêm informação métrica do ambiente nem estão ancoradas em um sistema de coordenadas do mundo.
- Proposta de uma nova modelagem intervalar para representação do mundo através de um conjunto de intervalos baseada na investigação citada no item anterior.
- Novo método intervalar baseado na aplicação da modelagem proposta para o problema de reconhecimento visual de regiões usando apenas imagens de câmeras monoculares.
- Uso de análise intervalar, em especial a técnica de intersecção relaxada, para seleção da melhor sequência de intervalos que indicam a combinação mais promissora dada uma observação de busca. Essa técnica se mostrou eficiente e com bons resultados para a busca de sequências de combinações.
- Elaboração e publicação de artigos em periódicos de alto impacto, todos relacionados ao uso de análise intervalar para atacar problemas da robótica móvel (NEULAND et al., 2020) (NEULAND et al., 2021).
- Também podemos citar como contribuição, a manutenção do vínculo de colaboração com o professor Dr. Luc Jaulin da ENSTA-Bretagne, o qual é especialista na área de análise intervalar e participou da elaboração dos artigos citados no item anterior.

A contribuição desta tese no avanço da arte abre espaço para uma série de novas pesquisas. Assim, listamos como trabalhos futuro:

- Criar mais testes inclusive variando a velocidade do robô durante a coleta das observações.
- Estudar sobre o uso de informações mais complexas sobre a movimentação do robô. Essa informação poderia ser extraída através de odometria visual. Lembrando que atualmente usamos apenas informação binária, o robô se moveu ou não.
- Investigar uma forma de calibração dos parâmetros k e w em tempo de execução. A informação acumulada sobre o ambiente pode ser útil para essa tarefa.
- Investigar o uso de pre-processamento de imagens. Analisar se existe melhoria nos resultados e se ela justificaria o custo dessa nova etapa de processamento.

- Estudar mais sobre o uso de diferentes descritores. Propôr as alterações necessárias na abordagem para o uso de descritores não binários, tais como nova representação global dos intervalos e função de similaridade.
- A abordagem proposta faz a busca por combinações usando apenas informações de iterações anteriores para a seleção da combinação corrente. Isso traz potencial para a aplicação do método em operações em tempo real. Mais investigações e testes precisam ser feitos para validar essa hipótese.
- Explorar o uso da modelagem proposta para outros problemas, como localização, localização global e o problema do robô raptado.

REFERÊNCIAS

ABDALLAH, F.; GNING, A.; BONNIFAIT, P. Adapting particle filter on interval data for dynamic state estimation. In: IEEE. **2007 IEEE International Conference on Acoustics, Speech and Signal Processing-ICASSP'07**. Honolulu, HI, USA, 2007. v. 2, p. II–1153.

ABDALLAH, F.; GNING, A.; BONNIFAIT, P. Box particle filtering for nonlinear state estimation using interval analysis. **Automatica**, Elsevier BV, v. 44, n. 3, p. 807–815, mar. 2008. Disponível em: <https://doi.org/10.1016/j.automatica.2007.07.024>.

ARANDJELOVIC, R. et al. Netvlad: Cnn architecture for weakly supervised place recognition. In: **The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. Las Vegas, Nevada: IEEE, 2016.

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>.

BAI, D. et al. Sequence searching with cnn features for robust and fast visual place recognition. **Computers & Graphics**, Elsevier, v. 70, p. 270–280, 2018. Disponível em: <https://doi.org/10.1016/j.cag.2017.07.019>.

BAMPIS, L.; AMANATIADIS, A.; GASTERATOS, A. Fast loop-closure detection using visual-word-vectors from image sequences. **The International Journal of Robotics Research**, SAGE Publications Sage UK: London, England, p. 0278364917740639, 2017.

BISHOP, R. Intelligent vehicle applications worldwide. **IEEE Intelligent Systems**, Institute of Electrical and Electronics Engineers (IEEE), v. 15, n. 1, p. 78–81, jan. 2000. Disponível em: <https://doi.org/10.1109/5254.820333>.

BOSCH, A.; ZISSERMAN, A.; MUNOZ, X. Representing shape with a spatial pyramid kernel. In: ACM. **Inter. Conf. on Image and video retrieval**. Amsterdam, The Netherlands: ACM Press, 2007. p. 401–408.

BREFORT, Q. et al. Towards fast and reliable localization of an underwater object: an interval approach. **Journal of Uncertain Systems**, Citeseer, v. 9, 2015.

CALONDER, M. et al. Brief: Binary robust independent elementary features. In: SPRINGER. **European conference on computer vision**. Heraklion, Crete, Greece, 2010. p. 778–792.

CHANCÁN, M.; MILFORD, M. Deepseqslam: A trainable cnn+ rnn for joint global description and sequence-based place recognition. **arXiv preprint arXiv:2011.08518**, 2020.

CIESLEWSKI, T. et al. Point cloud descriptors for place recognition using sparse visual information. In: **2016 IEEE International Conference on Robotics and Automation (ICRA)**. IEEE, 2016. Disponível em: <https://doi.org/10.1109/icra.2016.7487687>.

CUMMINS, M.; NEWMAN, P. Fab-map: Probabilistic localization and mapping in the space of appearance. **The International Journal of Robotics Research**, SAGE Publications Sage UK: London, England, v. 27, n. 6, p. 647–665, 2008.

CUMMINS, M.; NEWMAN, P. Appearance-only slam at large scale with fab-map 2.0. **The International Journal of Robotics Research**, SAGE Publications Sage UK: London, England, v. 30, n. 9, p. 1100–1123, 2011.

DALAL, N.; TRIGGS, B. Histograms of oriented gradients for human detection. In: IEEE. **Computer Vision and Pattern Recognition (CVPR), Conference on**. IEEE, 2005. v. 1, p. 886–893. Disponível em: <https://doi.org/10.1109/cvpr.2005.177>.

DELLAERT, F. et al. Monte carlo localization for mobile robots. In: **ICRA**. Detroit, MI, USA: IEEE, 1999. v. 2, p. 1322–1328.

DONGDONG, B. et al. Cnn feature boosted seqslam for real-time loop closure detection. **Chinese Journal of Electronics**, IET, v. 27, n. 3, p. 488–499, 2018.

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, feb 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>.

GARG, S.; MILFORD, M. Fast, compact and highly scalable visual place recognition through sequence-based matching of overloaded representations. In: IEEE. **2020 IEEE International Conference on Robotics and Automation (ICRA)**. Paris, France, 2020. p. 3341–3348.

GLOVER, A. **Datasets**. 2014. Data retrieved from Robotics@QUT, <https://wiki.qut.edu.au/display/raq/Day+and+Night+with+Lateral+Pose+Change+Datasets>.

GONZALEZ, D. et al. A review of motion planning techniques for automated vehicles. **IEEE Transactions on Intelligent Transportation Systems**, Institute of Electrical and Electronics Engineers (IEEE), v. 17, n. 4, p. 1135–1145, abr. 2016. Disponível em: <https://doi.org/10.1109/tits.2015.2498841>.

HAMMING, R. W. Error detecting and error correcting codes. **The Bell system technical journal**, Nokia Bell Labs, v. 29, n. 2, p. 147–160, 1950.

HOU, Y.; ZHANG, H.; ZHOU, S. Convolutional neural network-based image representation for visual loop closure detection. In: IEEE. **2015 IEEE international conference on information and automation**. Lijiang, China, 2015. p. 2238–2245.

HUISHEN, Z. et al. An improved bag of words method for appearance based visual loop closure detection. In: IEEE. **2018 Chinese Control And Decision Conference (CCDC)**. 2018. p. 5682–5687. Disponível em: <https://doi.org/10.1109/CCDC.2018.8408123>.

JAULIN, L. **Mobile Robotics**. Grã-Bretanha e Estados Unidos: ISTE Press - Elsevier, 2015. ISBN 1785480480.

JAULIN, L. et al. **Applied Interval Analysis**. London: Springer, 2001. Disponível em: <https://doi.org/10.1007/978-1-4471-0249-6>.

JORGE, V. A. et al. Ouroboros: Using potential field in unexplored regions to close loops. In: IEEE. **2015 IEEE International Conference on Robotics and Automation (ICRA)**. Seattle, WA, USA, 2015. p. 2125–2131.

LATOMBE, J.-C. **Robot Motion Planning**. New York: Kluwer Academic, 1991. ISBN 9780792392064.

LIU, Y.; ZHANG, H. Towards improving the efficiency of sequence-based slam. In: IEEE. **2013 IEEE International Conference on Mechatronics and Automation**. Takamatsu, Japan, 2013. p. 1261–1266.

LOWRY, S. et al. Visual place recognition: A survey. **IEEE Transactions on Robotics**, IEEE, v. 32, n. 1, p. 1–19, 2016.

MADEO, S.; BOBER, M. Fast, compact, and discriminative: Evaluation of binary descriptors for mobile applications. **IEEE Transactions on Multimedia**, IEEE, v. 19, n. 2, p. 221–235, 2016.

MAFFEI, R. et al. Segmented dp-slam. In: IEEE. **2013 IEEE/RSJ International Conference on Intelligent Robots and Systems**. Tokyo, Japan, 2013. p. 31–36.

MAFFEI, R. et al. Integrated exploration using time-based potential rails. In: IEEE. **2014 IEEE International Conference on Robotics and Automation (ICRA)**. Hong Kong, China, 2014. p. 3694–3699.

MAFFRA, F.; CHEN, Z.; CHLI, M. Viewpoint-tolerant place recognition combining 2d and 3d information for uav navigation. In: IEEE. **2018 IEEE International Conference on Robotics and Automation (ICRA)**. 2018. p. 2542–2549. Disponível em: <https://doi.org/10.1109/ICRA.2018.8460786>.

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>.

MARZULLO, K. A. **Maintaining the Time in a Distributed System: An Example of a Loosely-coupled Distributed Service (Synchronization, Fault-tolerance, Debugging)**. Tese (Doutorado) — Stanford University, Stanford, CA, USA, 1984. AAI8506272.

MERLET, J. P. Interval analysis and robotics. In: **Springer Tracts in Advanced Robotics**. Springer Berlin Heidelberg, 2010. p. 147–156. Disponível em: <https://doi.org/10.1007/978-3-642-14743-2_13>.

MERRILL, N.; HUANG, G. Lightweight unsupervised deep loop closure. **arXiv preprint arXiv:1805.07703**, 2018.

MILFORD, M. J.; WYETH, G. F. Seqslam: Visual route-based navigation for sunny summer days and stormy winter nights. In: **Robotics and Automation (ICRA), International Conference on**. IEEE, 2012. p. 1643–1649. Disponível em: <https://ieeexplore.ieee.org/abstract/document/6224623/>.

MOORE, R. E. **Interval analysis**. [S.l.]: Prentice-Hall Englewood Cliffs, 1966. v. 2.

MOORE, R. E.; KEARFOTT, R. B.; CLOUD, M. J. **Introduction to interval analysis**. Philadelphia: Siam, 2009. v. 110. ISBN 978-0-898716-69-6.

MUSTAFA, M. et al. Guaranteed slam—an interval approach. **Robotics and Autonomous Systems**, Elsevier, v. 100, p. 160–170, 2018.

NASEER, T.; BURGARD, W.; STACHNISS, C. Robust visual localization across seasons. **IEEE Transactions on Robotics**, IEEE, v. 34, n. 2, p. 289–302, 2018. Disponível em: <https://doi.org/10.1109/TRO.2017.2788045>.

NEULAND, R. et al. Robust hybrid interval-probabilistic approach for the kidnapped robot problem. **International Journal of Uncertainty Fuzziness and Knowledge-Based Systems**, 2020.

NEULAND, R. et al. Hybridization of monte carlo and set-membership methods for the global localization of underwater robots. In: **2014 IEEE/RSJ International Conference on Intelligent Robots and Systems**. Chicago, IL, USA: IEEE, 2014. p. 199–204. ISSN 2153-0858. Disponível em: <http://ieeexplore.ieee.org/document/6942561/>.

NEULAND, R. et al. Improving the precision of auvs localization in a hybrid interval-probabilistic approach using a set-inversion strategy. **Unmanned Systems**, v. 02, n. 04, p. 361–375, 2014. Disponível em: <http://www.worldscientific.com/doi/abs/10.1142/S230138501440010X>.

NEULAND, R. et al. Interval inspired approach based on temporal sequence constraints to place recognition. **Journal of Intelligent & Robotic Systems**, 2021.

PEPY, R.; KIEFFER, M.; WALTER, E. Reliable robust path planner. In: **2008 IEEE/RSJ International Conference on Intelligent Robots and Systems**. IEEE, 2008. Disponível em: <https://doi.org/10.1109/iros.2008.4650833>.

PRESTES, E.; ENGEL, P. M. Exploration driven by local potential distortions. In: **2011 IEEE/RSJ International Conference on Intelligent Robots and Systems**. IEEE, 2011. p. 1122–1127. ISSN 2153-0858. Disponível em: <http://ieeexplore.ieee.org/document/6094674/>.

PRESTES, E.; RITT, M.; FUHR, G. Improving monte carlo localization in sparse environments using structural environment information. In: **2008 IEEE/RSJ International Conference on Intelligent Robots and Systems**. IEEE, 2008. p. 3465–3470. ISSN 2153-0858. Disponível em: <http://ieeexplore.ieee.org/document/4651099/>.

QI, C. R. et al. Pointnet: Deep learning on point sets for 3d classification and segmentation. In: **The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. Honolulu, HI, USA: IEEE, 2017.

RIJSBERGEN, C. J. V. **Information Retrieval**. Butterworth-Heinemann, 1979. ISBN 0408709294. Disponível em: <http://www.dcs.gla.ac.uk/Keith/Preface.html>.

ROHOU, S. et al. Proving the existence of loops in robot trajectories. **The International Journal of Robotics Research**, SAGE Publications Sage UK: London, England, v. 37, n. 12, p. 1500–1516, 2018.

SANDRETTO, J. A. dit; CHAPOUTOT, A.; MULLIER, O. Formal verification of robotic behaviors in presence of bounded uncertainties. In: IEEE. **2017 First IEEE International Conference on Robotic Computing (IRC)**. Taichung, Taiwan, 2017. p. 81–88.

SIAM, S. M.; ZHANG, H. Fast-seqslam: A fast appearance based place recognition algorithm. In: **Robotics and Automation (ICRA), International Conference on**. IEEE, 2017. p. 5702–5708. Dataset download: <https://github.com/siam1251/Fast-SeqSLAM/blob/master/dataset/uofadatasetdrivelink>. Disponível em: <https://ieeexplore.ieee.org/abstract/document/7989671/>.

SIEGWART, R.; NOURBAKHSH, I. R. **Introduction to Autonomous Mobile Robots (Intelligent Robotics and Autonomous Agents series)**. Massachusetts: The MIT Press, 2004. ISBN 026219502X.

SIVIC, J.; ZISSERMAN, A. Video google: A text retrieval approach to object matching in videos. In: **Computer Vision (ICCV), International Conference on**. IEEE, 2003. p. 1470. Disponível em: <https://members.loria.fr/MOBerger/Enseignement/Master2/Exposes/sivic03.pdf>.

SÜNDERHAUF, N.; NEUBERT, P.; PROTZEL, P. Are we there yet? challenging seqslam on a 3000 km journey across all four seasons. In: **Proceedings of Workshop on Long-Term Autonomy, International Conference on Robotics and Automation (ICRA)**. IEEE, 2013. p. 2013. Dataset download: http://nrkbeta.no/2013/01/15/. Disponível em: <https://nikosuenderhauf.github.io/assets/papers/openseqslam.pdf>.

TALBOT, B.; GARG, S.; MILFORD, M. Openseqslam2. 0: An open source toolbox for visual place recognition under changing conditions. In: IEEE. **2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)**. Madrid, Spain, 2018. p. 7758–7765.

THRUN, S. et al. **Probabilistic robotics**. Massachusetts: MIT press Cambridge, 2005. v. 1.

THRUN, S. et al. Robotic mapping: A survey. **Exploring artificial intelligence in the new millennium**, v. 1, n. 1-35, p. 1, 2002.

TOMITĂ, M.-A. et al. Convsequential-slam: A sequence-based, training-less visual place recognition technique for changing environments. **arXiv preprint arXiv:2009.13454**, 2020.

TSINTOTAS, K. A.; BAMPIS, L.; GASTERATOS, A. Doseqslam: dynamic on-line sequence based loop closure detection algorithm for slam. In: IEEE. **2018 IEEE International Conference on Imaging Systems and Techniques (IST)**. Krakow, Poland, 2018. p. 1–6.

TSINTOTAS, K. A. et al. Seqslam with bag of visual words for appearance based loop closure detection. In: SPRINGER. **International Conference on Robotics in Alpe-Adria Danube Region**. Italy, 2018. p. 580–587.

UY, M. A.; LEE, G. H. Pointnetvlad: Deep point cloud based retrieval for large-scale place recognition. In: **The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. Salt Lake City, UT, USA: IEEE, 2018.

VYSOTSKA, O.; STACHNISS, C. Effective visual place recognition using multi-sequence maps. **IEEE Robotics and Automation Letters**, Institute of Electrical and Electronics Engineers (IEEE), v. 4, n. 2, p. 1730–1736, abr. 2019. Disponível em: <https://doi.org/10.1109/lra.2019.2897160>.

WANG, P. et al. Box particle filtering for slam with bounded errors. In: IEEE. **2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV)**. Singapore, 2018. p. 1032–1038.

WANG, Y. et al. Improved seqslam for real-time place recognition and navigation error correction. In: IEEE. **2015 7th International Conference on Intelligent Human-Machine Systems and Cybernetics**. Hangzhou, China, 2015. v. 1, p. 260–264.

WEISS, R. et al. Hybridisation of sequential monte carlo simulation with non-linear bounded-error state estimation applied to global localisation of mobile robots. **Journal of Intelligent & Robotic Systems**, Springer Science and Business Media LLC, v. 99, n. 2, p. 335–357, dez. 2019. Disponível em: <https://doi.org/10.1007/s10846-019-01118-7>.

YANG, X.; CHENG, K.-T. Local difference binary for ultrafast and distinctive feature description. **IEEE transactions on pattern analysis and machine intelligence**, IEEE, v. 36, n. 1, p. 188–194, 2014.

ZAFFAR, M. et al. Cohog: A light-weight, compute-efficient, and training-free visual place recognition technique for changing environments. **IEEE Robotics and Automation Letters**, IEEE, v. 5, n. 2, p. 1835–1842, 2020.