Universidade Federal do Paraná

Valber Lemes Zacarkim

Avaliação do Detector de Pontos de Interesse IGFTT em Visual SLAM

Curitiba
2017

Valber Lemes Zacarkim

Avaliação do Detector de Pontos de Interesse IGFTT em Visual SLAM

Dissertação apresentada como requisito parcial à obtenção do grau de Mestre em Informática no Programa de Pós-Graduação em Informática, setor de Ciências Exatas, da Universidade Federal do Paraná.

Área de concentração: *Ciência da Computação*.

Orientador: Prof. Dr. Eduardo Todt.

Curitiba
2017

Z13a Zacarkim, Valber Lemes
Avaliação do Detector de Pontos de Interesse IGFTT em Visual SLAM / Valber Lemes Zacarkim. – Curitiba, 2017.
136 f. : il. color. ; 30 cm.

Dissertação - Universidade Federal do Paraná, Setor de Ciências Exatas, Programa de Pós-Graduação em Informática, 2017.

Orientador: Eduardo Todt.

1. Ciênca da computação. 2. Odometria visual. 3. Visual SLAM. I. Universidade Federal do Paraná. II. Todt, Eduardo. III. Título.

CDD: 006.6

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DO PARANÁ
PRÓ-REITORIA DE PESQUISA E PÓS-GRADUAÇÃO
Setor CIÊNCIAS EXATAS
Programa de Pós-Graduação INFORMÁTICA

TERMO DE APROVAÇÃO

Os membros da Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação em INFORMÁTICA da Universidade Federal do Paraná foram convocados para realizar a arguição da Dissertação de Mestrado de **VALBER LEMES ZACARKIM** intitulada: **Uma avaliação comparativa do detector de características IGFTT em Visual SLAM**, após terem inquirido o aluno e realizado a avaliação do trabalho, são de parecer pela sua aprovação no rito de defesa.
A outorga do título de mestre está sujeita à homologação pelo colegiado, ao atendimento de todas as indicações e correções solicitadas pela banca e ao pleno atendimento das demandas regimentais do Programa de Pós-Graduação.

Curitiba, 30 de Agosto de 2017.

EDUARDO TODT
Presidente da Banca Examinadora (UFPR)

FERNANDO SANTOS OSORIO
Avaliador Externo (USP/SC)

DAVID MENOTTI GOMES
Avaliador Externo (UFMG)

BRUNO MÜLLER JUNIOR
Avaliador Externo (UFPR)

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DO PARANÁ
Setor de Ciências Exatas
Coordenação de Pós-Graduação em Informática

À minha mulher Geisislaine e aos meus dois filhos Gabriel e Miguel.

"É exatamente disso que a vida é feita: de momentos! Momentos os quais temos que passar, sendo bons ou não, para o nosso próprio aprendizado, por algum motivo. Nunca esquecendo do mais importante: nada na vida é por acaso..., não vale tanto o que temos, nem tanto importa o que somos, vale o que realizamos com aquilo que possuímos e, acima de tudo, importa o que fazemos de nós!"
Chico Xavier

Agradecimentos

À Deus, pela oportunidade de adquirir conhecimento nesta vida.

À minha mãe Maria Auxiliadora e ao meu pai Valter Zacarkim, pela educação direcionada.

Aos meus irmãos Valder, Valkerson e Valkiria, pela parceria nos momentos difíceis.

Ao meu orientador Eduardo Todt, pela confiança e dedicação.

Aos meus amigos de Matogrosso Eraldo, Jeferson, Márcio, Gesiel, Patrícia, Elenice, Dona Cida, Géssica, Vinícius, Mirele, Michele,Gleison, seu José Geronídes e dona Maria, pelo apoio incondicional e companheirismo.

À pensão da dona Noedi e seu Rui Barbosa, àqueles que cuidaram de mim como filho em uma terra distante.

Aos inúmeros amigos de longa longa data, àqueles que levaram um pouco de mim e deixaram um pouco de sí.

Aos pesquisadores citados neste trabalho, por fornecerem seus conhecimentos.

Ao Grupo de Pesquisa em Visão, Imagem e Robótica da UFPR, por acreditarem na minha capacidade inicial e a complementarem ao nível de Mestre em Informática.

Aos amigos adquiridos nesta terra distante de Curitiba, amigos estes que jamais esquecerei. Um eterno obrigado!

Resumo

Detectores de Ponto de Interesse (POI) são partes fundamentais de um Sistema Visual de Mapeamento e Localização Simultâneos (VSLAM). Pontos de interesse na imagem são usados pela odometria visual para estimar o deslocamento de um robô por meio do movimento de câmera. Além disso, o robô pode reconhecer lugares já visitados pela correspondência de pontos de interesse entre imagens. Avaliar a qualidade destes detectores no sentido de determinar qual o melhor para ser utilizado em determinado ambiente, é de extrema importância ao VSLAM, pois a qualidade do mapa gerado pelo sistema depende de robustos pontos de interesse encontrados nas imagens. Um novo detector de pontos de interesse conhecido como IGFTT apresenta resultados promissores em relação aos métodos clássicos como SURF, SIFT, ORB e outros. O presente trabalho avalia o novo algoritmo de detecção de pontos de interesse IGFTT em função do desempenho de um sistema Visual SLAM, comparando-o com os detectores mais populares na comunidade acadêmica como o SIFT, SURF, FAST, AGAST, GFTT, entre outros. É utilizada uma base pública como *benchmark* em conjunto com um sistema de VSLAM de código aberto. A qualidade do detector é determinada em função do mapa gerado pelo VSLAM, do tempo de processamento, da taxa de repetibilidade dos pontos de interesse, da probabilidade de sobrevivência destes em *frames* futuros e da quantidade de nós sucessivos no grafo de cena. Os testes demonstram as qualidades e as deficiências do algoritmo IGFTT. Os resultados de cada detector para cada teste são ranqueados de forma global e demonstram a acurácia geral dos algoritmos. No *ranking* obtido, SURF apresenta-se em primeiro lugar, em segundo lugar o IGFTT2, melhoria do algoritmo testado, e em terceiro o IGFTT. A acurácia do detector é definida pelo Erro da Trajetória Absoluta e apresenta o algoritmo GFTT com 0,278 metros de erro obtendo o melhor resultado e IGFTT, o pior resultado com 0,944 metros de erro, e o algoritmo IGFTT2 obteve um erro de 0,712 metros. Em tempo de processamento por *frame* nas fases de detecção, descrição e correspondência dos pontos de interesse, IGFTT obteve o melhor desempenho consumindo em média 52,1 milissegundos, demonstrando ser extremamente rápido. IGFTT alcançou a segunda melhor colocação em relação a taxa de repetibilidade e probabilidade dos POIs, atrás do SURF. E quanto à taxa de nós do grafo de cena em sequência, IGFTT obteve o melhor resultado com 47,07%. SURF conseguiu uma taxa de 39,30% porém destes, somente 1,33% não tiveram poses atribuídas pelo sistema RGBDSLAM. De forma geral, o detector de pontos de interesse IGFTT é veloz em termos de tempo de processamento e estável, dada dispersão dos seus resultados conforme variedade de sequências sob testes.

Palavras-chave: Detectores de Pontos de Interesse, Odometria Visual, Visual SLAM, Avaliação de Detectores de Pontos de Interesse.

Abstract

Keypoints Detectors are fundamental parts of a Visual Simultaneous Localization and Mapping System or VSLAM. Keypoints in the image are used by visual odometry to estimate the displacement of a robot by means of camera movement. Furthermore, the robot can recognize places already visited by matching keypoints between images. To evaluate the quality of these detectors in order to determine the best to be used in a particular environment, it is of extreme importance to SLAM, since the quality of the map generated by the system depends on robust keypoints found in the images. The present work evaluates the new algorithm of keypoints detection IGFTT as a function of the performance of a Visual SLAM system, comparing it with the most popular detectors in the academic community such as SIFT, SURF, FAST, AGAST, GFTT, among others. A public database is used as a benchmark in conjunction with an open source VSLAM system. The quality of the detector is determined by the VSLAM generated map, runtime processing, keypoints repeatability rate, they survival probability rate in future frames and the number of successive nodes in the scene graph. The tests demonstrate the qualities and shortcomings of the IGFTT algorithm. The results of each detector for each test are globally ranked and demonstrate the overall accuracy of the algorithms. In the obtained ranking, SURF presents first, secondly the IGFTT2, improvement of the algorithm tested, and thirdly the IGFTT. The accuracy of the detector is defined by the Absolute Trajectory Error and presents the GFTT algorithm with 0.278 meters of error obtaining the best result and IGFTT, the worst result with 0.944 meters of error, and IGFTT2 algorithm obtained an error of 0.712 meters. In runtime processing per frame in the phases of detection, description and correspondence of keypoints, IGFTT obtained the best performance consuming in average 52.1 milliseconds, proving to be extremely fast. IGFTT achieved the second best placement in relation to the repeatability and probability rate of POIs, behind SURF. As for the node rate of the scene graph in sequence, IGFTT obtained the best result with 47.07%. SURF achieved a rate of 39.30% but of these, only 1.33% had no poses attributed by the system RGBDSLAM. In general, the IGFTT keypoint detector is fast in terms of time processing and stable, given its dispersion of results according to the variety of sequences under test.

Keywords: Keypoints Detectors, Visual Odometry, Visual SLAM, Keypoints Detectors Evaluation.

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

Localizar-se no momento de explorar e mapear um ambiente desconhecido é um dos principais problemas de um robô móvel autônomo. Esta dificuldade é subjacente na maioria das pesquisas em robótica pelo fato de envolver praticamente todos os campos da área de robótica móvel como: sensação, ação, planejamento, arquiteturas, *hardware*, eficiência computacional, além de resolver problemas inerentes ao propósito do aparato mecatrônico. Robôs reativos podem mover-se pelo ambiente sem colisão, mas a navegação envolve deliberação momentânea, onde os robôs precisam ser aptos a planejar e tomar a decisão para qual direção se mover [Murphy, 2000]. Este problema é largamente pesquisado há décadas e conhecido como o problema da localização e mapeamento simultâneo ou *Simultaneous Localization and Mapping* (SLAM).

A qualidade do SLAM está diretamente ligada aos sensores do aparato robótico, os quais captam informações do ambiente, enquanto o robô as processa e infere a nova posição dada a sua pose[1] anterior no mapa criado. Ruído nos sensores ou na forma de processamento do sinal captado interferem diretamente na estimativa da pose e do próprio mapa, tornando-se um problema importante a ser tratado.

1.1 Motivação

Na robótica, um interesse por sensores tipo câmeras vem crescendo atualmente por terem baixo custo e fornecerem inúmeras informações do ambiente navegado. Desta forma, a técnica utilizada em SLAM que explora o sensor de visão do robô é conhecida como Visual SLAM ou *Visual Simutaneous Localization and Mapping* (VSLAM). O problema do VSLAM é selecionar Pontos de Interesse (POI) nas imagens adquiridas pela câmera de forma que, após o processamento, seja possível gerar informações tridimensionais fidedignas do ambiente, representando um mapa robusto.

Mapeamentos não robustos afetam negativamente a robótica, pois degradam a certeza de localização do robô ao ponto deste perder-se no próprio mapa. Assim, diversos estudos vêm sendo realizados na área do VSLAM para produzir algoritmos que detectem boas características de imagens, incrementando a qualidade do mapa gerado pelo robô.

Dada uma trajetória conhecida, avaliar a qualidade dos detectores de POIs em função do mapa produzido pode identificar qualidades ou deficiências dos detectores em relação ao tempo de processamento, a repetibilidade do POI e a estabilidade relacionada à predição de POIs ao longo de várias imagens. O trabalho de Gil et al. [2010] se baseia nos métodos de detecção (onde se tem controle total das transformações nas imagens), avaliando a repetibilidade

[1] Posição e Orientação Relativa de um corpo.

de variados detectores, assim como a invariância e a distintividade de seus respectivos descritores em diferentes condições de escala, pontos de vista e iluminação. Utilizando uma abordagem que mensura o detector em função do mapa criado pelo VSLAM (onde não existe o controle das transformações nas imagens), Endres et al. [2012] não só implementam um método de SLAM, como avaliam vários detectores de características quanto ao erro da trajetória absoluta, o tempo de processamento em relação as fases de detecção / descrição / correspondência dos POIs, estimação do movimento da câmera e otimização do grafo gerado pela odometria visual.

Embora as métricas que avaliam a detecção de POIs em uma imagem, baseadas somente nas características de cada método (imagens controladas) ou no seu desempenho em VSLAM (imagens não controladas), sejam estudadas há diversos anos, não é um campo de pesquisa saturado. Em um curto espaço de tempo, diferentes métricas são elaboradas, ou ainda, novos e robustos métodos de detecção de POIs são desenvolvidos, necessitando assim de métricas específicas e novas avaliações.

Dentre os inúmeros detectores de POIs recentemente propostos, o *Improved Good Features to Track* (IGFTT) vem se destacando. Desenvolvido por Oliveira et al. [2015], o IGFTT apresenta excelentes resultados. O algoritmo foi testado em imagens controladas (disponíveis em *benchmarks*) quanto a precisão e sensibilidade sob mudanças fotométricas e geométricas, onde aproxima ou ultrapassa o desempenho de outros detectores quanto a repetibilidade, distintividade, robustez e tempo de processamento.

Desta forma, conhecendo a qualidade do detector IGFTT para transformações fotométricas, geométricas, e velocidade de processamento, e ainda, sabendo que a qualidade do Visual SLAM depende de robustos POIs, uma avaliação do IGFTT para VSLAM seria uma excelente contribuição sob o aspecto da qualidade do mapa produzido, em relação a outros detectores de pontos de interesse.

1.2 Objetivo

A partir das observações apresentadas acima, a contribuição desta pesquisa é dada pela avaliação de um novo detector de pontos de interesse no problema de VSLAM. Ainda, falhas no detector podem ser encontradas ao analisá-lo *frame* a *frame*, abrindo um leque de oportunidades para trabalhos futuros. Demonstrando resultados robustos, o IGFTT poderá ser a melhor escolha dentre os detectores de pontos de interesse para sistemas VSLAM em relação as métricas analisadas.

Desta forma, este trabalho têm como objetivo avaliar o algoritmo de detecção de pontos de interesse IGFTT conforme o desempenho de um sistema Visual SLAM, comparando-o com os detectores implementados na biblioteca *Open Source Computer Vision Library* (OPENCV). Especificamente, pretende-se avaliar os detectores em relação:

- ao tempo de processamento gasto na identificação, descrição e correspondência de POIs;
- ao erro que o mapa produzido pelo VSLAM têm em relação ao mapa base;
- ao erro de trajetória estimada pela odometria do VSLAM em relação a trajetória base;
- a taxa de repetibilidade que POIs têm dado um conjunto de *frames* em sequência;
- a taxa de probabilidade que POIs têm de serem rastreados em *frames* futuros dado que já foram identificados em *frames* passados, conforme um conjunto de *frames* em sequência;

- a porcentagem de nós do grafo de cena produzido pelo VSLAM que estão em sequência, dado que estes fazem parte do menor caminho do grafo partindo do *frame* 1 até o último *frame*, e;
- a porcentagem de nós sem pose que fazem parte do menor caminho do grafo de cena.

A acurácia do detector de pontos de interesse é então atribuída dado:

- o ranking geral dos resultados de todas as avaliações, demonstrando que o detector é robusto de forma geral para sistemas de VSLAM e;
- o menor erro que o mapa produzido pelo VSLAM têm em relação ao mapa base, demonstrando que, em relação à precisão na criação de mapas, o detector é mais robusto.

1.3 Estrutura do Trabalho

A dissertação está organizada em oito capítulos. No Capítulo 2 são apresentados os fundamentos relacionados ao SLAM que são necessários para a compreensão do problema tratado. Também são detalhadas algumas abordagens utilizadas para resolver o problema de localização e mapeamento simultâneos com foco em visão computacional. Em seguida, o Capítulo 3 descreve os fundamentos com relação à detecção de pontos de interesse, e os métodos de extração (detecção e descrição de POIs) utilizados neste trabalho. O Capítulo 4 faz referência dos trabalhos relacionados ao tema da dissertação. A justificativa detalhada do trabalho e a análise da pesquisa de Oliveira et al. [2015] é apresentada no Capítulo 5. No Capítulo 6 é descrita a metodologia utilizada para avaliar o detector IGFTT em função do VSLAM. Os resultados e a discussão dos testes são abordados no Capítulo 7. O Capítulo 8 conclui a pesquisa apresentado as considerações finais e, por fim, as referências bibliográficas são elencadas.

Capítulo 2

Localização e Mapeamento de Robôs

Identificar pontos de interesse em imagens e correspondê-las é importante para estimar a transformação que uma imagem sofre em relação a outra. A diferença entre transformações fornecem a trajetória do sensor que capturou as imagens. O trajeto que a câmera faz não fornece informações de lugares que já foram visitados. O mapeamento e a localização simultâneas SLAM resolvem este problema e geram mapas do ambiente por onde o sensor navegou.

Este capítulo apresenta os fundamentos necessários à compreensão do problema de localização e mapeamento de robôs com foco no sensor de imagens (câmera). A Seção 2.1 apresenta as características gerais de um robô autônomo, seu problema quanto ao *Simultaneous Localization and Mapping* (SLAM) e abordagens para resolução deste. A Seção 2.2 descreve o SLAM baseado em Visão, onde sensores exteroceptivos passivos do tipo câmera são utilizados para aferir o deslocamento do robô. A Seção detalha ainda todo o processo que envolve o deslocamento, também chamado de Odometria Visual ou *Visual Odometry* (VO). São abordados conceitos desde a aquisição de imagens até a geometria epipolar, que relaciona imagens tomadas a partir de pontos de vista distintos.

2.1 SLAM

Uma das habilidades de um robô é ser autônomo. Ser capaz de deliberar sobre trajetórias, objetivos, movimentos e deslocamentos sem ser necessariamente reativo, torna pesquisas em robótica desafiadoras pelo fato de envolver praticamente todos os campos da Inteligência Artificial de Robôs [Murphy, 2000].

Segundo Siegwart et al. [2011], movimentar-se por um ambiente livremente depende de questões que separam a área da navegação na robótica por funcionalidades, sendo estas:

1. *Onde estou indo*?: Muitas vezes determinado por um humano ou por um plano de missão.
2. *Qual é o melhor caminho*?: Este é um problema de planejamento de rotas e têm sido muito estudado na robótica.
3. *Onde eu estive*?: Fazer (ou atualizar) um mapeamento do ambiente já conhecido é de extrema importância para navegação por auxiliar no planejamento de rotas e localização relativa do robô.
4. *Onde eu estou*?: Para seguir uma rota ou construir um mapa o robô deve saber onde ele está, sendo um problema conhecido por localização.

Todas as quatro perguntas, para serem respondidas, dependem da aquisição de conhecimento sobre o ambiente sendo esta uma das mais importantes tarefas de sistemas autônomos. Esta tarefa demanda o uso de sensores que fazem medições do ambiente e do robô para extrair informações importantes e ajudar nas decisões.

Uma variedade de sensores são utilizados na robótica móvel. Alguns medem a temperatura interna dos componentes eletrônicos do robô, de seus atuadores ou mesmo a força que este aplica sobre determinado objeto. Outros capturam informações do ambiente como distâncias de paredes, pedestres, obstáculos e localizam o robô relativamente, ou mesmo globalmente com sensores *Global Positioning System* (GPS).

Os sensores são classificados em duas funções importantes: proprioceptivo e exteroceptivos. Sensores proprioceptivos medem valores internos do robô como por exemplo, velocidade das rodas, ângulo dos braços mecânicos e potência da bateria. Já os exteroceptivos adquirem informações do ambiente em que o robô se encontra, como intensidade da luz, altura do som e distâncias de objetos. Dentre estes tipos de sensores há os passivos e os ativos, onde os passivos fazem a medição sem o uso de emissão de energia para o ambiente, como câmeras e microfones, diferente dos ativos que emitem uma energia ao ambiente e medem o retorno desta, como sonares e *lasers* [Siegwart et al., 2011].

De fato, dentre as quatro perguntas, "Onde eu estou?", é a que têm recebido maior atenção durante décadas, por necessitar das respostas das outras três perguntas para localizar o robô. Para que o aparato mecatrônico chegue ao objetivo (pergunta 1) é necessário que haja uma rota (pergunta 2) traçada para que seus atuadores leve-o na direção desejada. No entanto, para que haja uma rota, um mapa (pergunta 3) do ambiente deve ser previamente conhecido pelo robô ou este deve criá-lo durante a navegação. Por fim, para se criar um mapa, o robô deve saber sua localização (pergunta 4) relativa ao ambiente que se encontra, e global a um mapa.

Mesmo o robô obtendo um mapa pronto com um objetivo definido, caso o ambiente não seja controlado, o mesmo deve reconhecer as mudanças ocorridas por meio dos sensores para atualizar o seu mapa e a sua posição. A alternativa é gerar e atualizar o mapa durante o processo de aprendizado sobre o ambiente. Na comunidade de robótica, este problema é conhecido como *Simultaneous Localization and Mapping* (SLAM) ou mapeamento e localização simultâneas, e apresenta uma alta relevância por conceber a ideia de um aparato autônomo.

O SLAM recupera a localização do robô e o mapa do ambiente por meio das leituras dos sensores proprioceptivos e exteroceptivos onde, a diferença entre as medições estimadas pela sua odometria, representam o deslocamento do robô. Geralmente estes dados adquiridos pelos sensores possuem ruído, tornando o trabalho do SLAM difícil. A Figura 2.1 ilustra o problema.

O robô identifica um ponto de referência por intermédio das leituras dos sensores e armazena a informação com uma incerteza devido ao ruído nas medições (Figura 2.1 (a)). Quando o robô desloca-se, a incerteza de sua localização aumenta, pois sua odometria também têm certo ruído (Figura 2.1 (b)). Em seguida, mais dois pontos de referência são identificados pelos sensores e armazenados com uma incerteza maior, relativa a incerteza da localização do robô no momento (Figura 2.1 (c)). Com mais um movimento do robô, a incerteza de sua localização aumenta ainda mais por não ter referências conhecidas (Figura 2.1 (d)). Quando o robô localiza um ponto de referência conhecido todas as incertezas são recalculadas e reduzidas com relação a incerteza de sua nova localização (Figura 2.1 (e)), fato conhecido como detecção de fechamento de laço (*Loop Closure Detection*) [Siegwart et al., 2011].

A forma de resolução do problema de SLAM descrita anteriormente é conhecida como Mapeamento Probabilístico, e segundo Cadena et al. [2016] foi uma das primeiras técnicas utilizadas na robótica em conjunto com a ferramenta *Extended Kalman Filter* (EKF) por Smith and Cheeseman [1986], que descrevem matematicamente a abordagem.

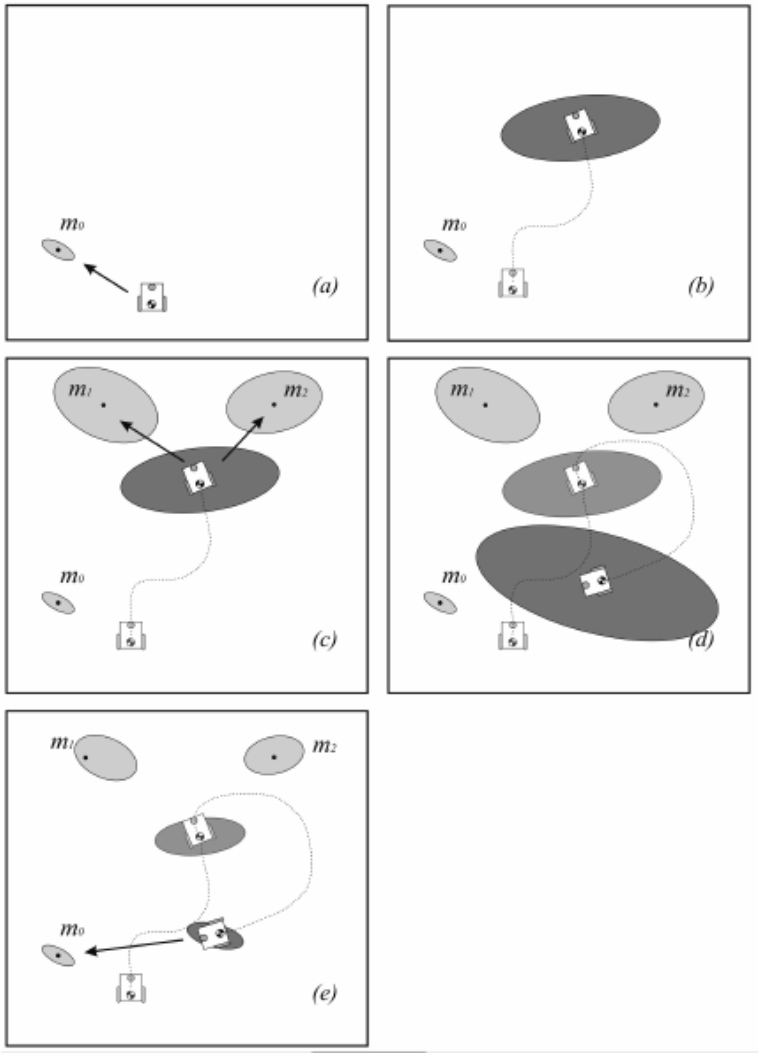

Figura 2.1: SLAM Probabilístico. FONTE: Extraída de [Siegwart et al., 2011, p. 350]. Em (a), o robô identifica um ponto de referência dado seu sensoriamento e atribuí incerteza (elipse) a este devido aos ruídos dos sensores; em (b), o robô move-se para algum lugar e, conforme aumenta a distância da referência juntamente com o ruído da odometria, a incerteza de sua localização atual também aumenta; em (c), o robô identifica mais duas referências e atribuí incertezas maiores a estas dada incerteza de sua localização ser grande; em (d), o robô move-se novamente para algum lugar aumentando ainda mais a incerteza de sua localização atual; em (e) um ponto de referência é reencontrado e a incerteza da posição do robô e das referências encontradas no caminho são diminuídas, dada que as incertezas estão correlacionadas.

Outra técnica amplamente utilizada no SLAM é a baseada em Grafos, inicialmente concebida em Lu and Milios [1997]. O problema pode ser interpretado como um grafo esparso formado por restrições entre seus nós. Os nós do grafo são as localizações do robô e dos pontos de referência adquiridos no tempo. As restrições são posições relativas entre cada nó de poses, obtidas pela odometria, e entre cada nó de poses e pontos de interesse. As restrições são representadas por molas conforme demonstra a Figura 2.2. Assim, uma rede elástica é formada e sua solução é dada pelo cálculo da energia mínima da rede por meio de técnicas de otimização [Siegwart et al., 2011].

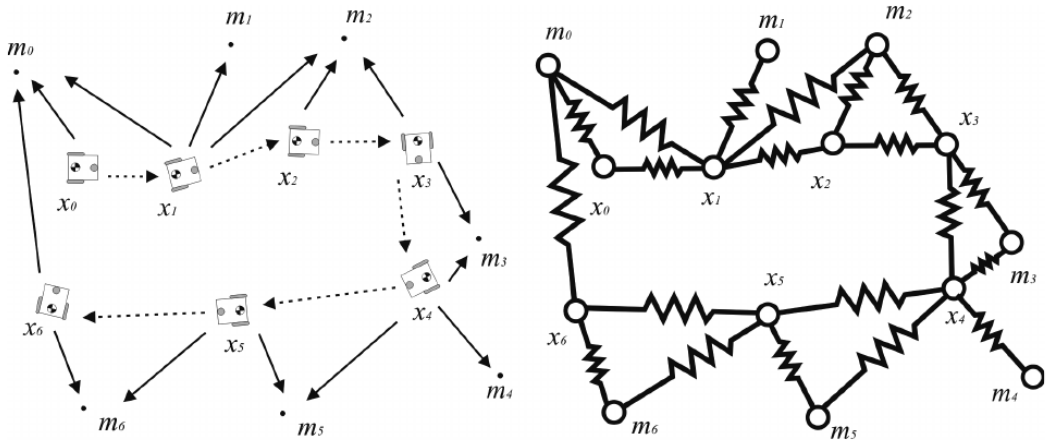

Figura 2.2: SLAM com grafos. FONTE: Extraída de [Siegwart et al., 2011, p. 362]. Todo o problema pode ser interpretado como um grafo esparso que forma uma rede elástica onde as setas cheias são restrições entre o robô e os pontos de referência, as setas pontilhadas são restrições entre localizações do robô e uma restrição é a pose relativa entre nós.

Têm obtido maior relevância trabalhos que utilizam sensores exteroceptivos passivos do tipo câmera nos robôs. Estes sensores têm o custo reduzido e fornecem ao aparato robótico imagens do ambiente que são ricas em informações. Por meio das sucessivas imagens e das informações em comum entre estas, é possível estimar o deslocamento e velocidade do robô. Na área de Visão Computacional, esta técnica é conhecida como *Visual Odometry* (VO) e é largamente utilizada em SLAM, onde a fusão das duas técnicas é conhecida como Visual SLAM (VSLAM).

2.2 VSLAM

Existem diversos estudos sobre Visual SLAM que diferenciam-se nas formas de: como cada analisa os sucessivos *frames* para estimar o deslocamento e velocidade (VO) e; como é estimada a localização do robô e a correção do mapa produzido pelo mesmo (SLAM).

Odometria Visual é um dos principais componentes do VSLAM. É fortemente embasada na técnica de *Structure from Motion* (SfM) ou estrutura de movimento, utilizando geometria projetiva para calcular a posição da câmera no mesmo instante em que calcula a estrutura tridimensional da cena. Conforme a Figura 2.3, de forma genérica a odometria é calculada pela diferença da posição da câmera 2 em relação a câmera 1, conforme visualizam os mesmos objetos (estrelas amarelas) tridimensionais na cena. A cor amarela na estrela indica que sua profundidade é conhecida ou foi estimada.

Odometria visual, assim como em odometria por rodas, sofrem escorregamento durante movimentos sucessivos por acumular erros devido a ruídos no processo de aquisição. Estes ruídos variam de acordo com a resolução da câmera, movimentos de objetos na cena, mudanças na iluminação, imprecisão na modelagem projetiva e tempo de processamento para relacionar e estimar a transformação entre imagens [Siegwart et al., 2011].

Para extrair profundidade de pontos de interesse, além da técnica de SfM que estima a pose por geometria projetiva, sensores *laser* tipo *Light Detection And Ranging* (LIDAR) ou sensores infravermelho tipo *Kinect - Microsoft®* também captam. Ambos sensores espalham pontos luminosos (não necessariamente visíveis a olho nu) pelo ambiente, de forma que o LIDAR

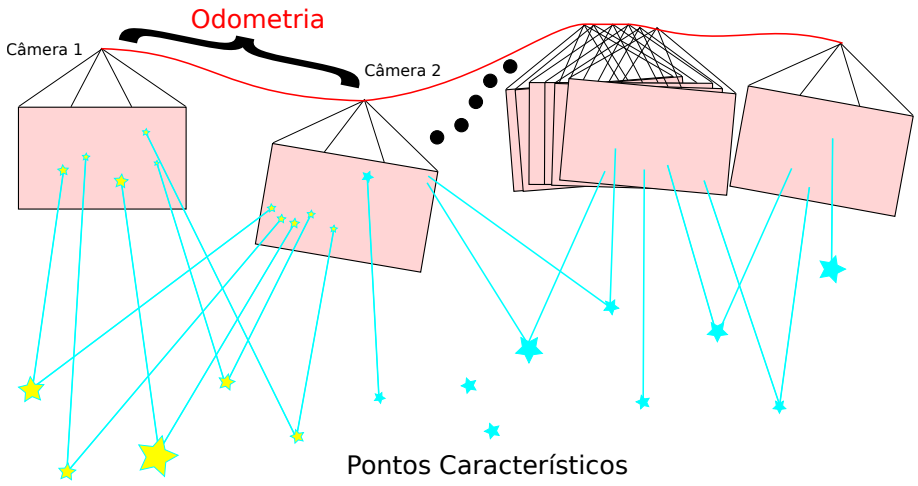

Figura 2.3: Processo da odometria visual. FONTE: Autor. A câmera 1 identifica pontos de referência (estrelas azuis) no ambiente. A câmera 2 também captura as mesmas referências da câmera 1. É então estimada a posição tridimensional destes pontos de referência (estrelas amarelas) e, conhecendo a posição dos pontos 3D, estima-se também o deslocamento relativo entre as câmeras, conhecido como odometria visual.

[Schwarz, 2010] calcula a distância baseada no tempo de retorno dos raios emitidos, diferente do *Kinect®* que faz uso de tecnologias de visão computacional como a luz estruturada [Bell et al., 1999], a profundidade por foco [Pertuz et al., 2013] e a profundidade por estéreo [Hartley and Zisserman, 2004]. Ainda o *Kinect®* utiliza lentes astigmáticas que, ao receber as imagens dos círculos projetados no ambiente, transforma estes em elipses de acordo com a profundidade, incrementando a acurácia da distância dos objetos [Shotton et al., 2011]. Logo, a imagem adquirida contem um canal extra com a informação da profundidade de cada *pixel*.

Algoritmos como ORBSLAM[1] [Klüssendorff et al., 2013] utilizam geometria projetiva para estimar a profundidade do ambiente e a odometria, diferente do Sistema de SLAM baseado no sensor RGBD (RGBDSLAM)[2] [Mur-Artal et al., 2015] que aproveita-se da tecnologia do *Kinect®* para extrair ao mesmo tempo a imagem e a profundidade do ambiente facilitando na estimação da odometria. Para resolver o problema do SLAM ambos utilizam grafos. Todos os algoritmos que utilizam visão para calcular deslocamentos ou gerar mapas dependem de pontos de interesse reconhecidos em mais de uma imagem.

Conhecidos como POIs, os pontos de interesse podem ser caracterizados por cantos, bordas, objetos, ou formas mais elaboradas reconhecidas nas imagens. Variados estudos vêm abordando formas de identificar e descrever POIs, uns focando a repetibilidade destes pontos em vários *frames*, outros em robustez da localização destes nas imagens, e alguns que enfatizam a velocidade de processamento na detecção para utilização em *hardwares* menos poderosos. O *Improved Good Features to Track* (IGFTT) é um destes métodos de detecção de POIs.

Como este trabalho propõe a avaliação de um algoritmo de detecção de pontos de interesse (IGFTT) em um robô que utiliza Visual SLAM para locomover-se em ambientes desconhecidos, é descrito a seguir conceitos sobre odometria visual, especificamente sobre a forma de captura da imagem, o modelo de inferência da profundidade de um POI na cena ou posicionamento da câmera e, como ocorre o calculo do deslocamento desta câmera dado dois ou mais pontos de vista. Métodos de detecção de POIs serão abordados no Capítulo 3.

[1]Sistema de SLAM baseado no extrator ORB
[2]Sistema de SLAM baseado no sensor RGBD

2.2.1 Aquisição da Imagem

A visão natural é um sofisticado sistema que detecta e age por meio de estímulos visuais. Tem evoluído por milhões de anos, principalmente para a defesa ou a sobrevivência do ser que a contém. A visão computacional estuda e tenta obter resultados similares aos da visão natural. O objetivo de ambos os sistemas é interpretar os dados espaciais, estes que podem ter mais de uma dimensão.

Nosso sistema de visão fornece uma enorme quantidade de informações sobre o ambiente e permite a interação rica e inteligente em ambientes dinâmicos. Por isso, um grande esforço têm sido dedicado para elaboração de máquinas com sensores que imitam as capacidades da visão humana. O primeiro passo neste processo é a criação de dispositivos sensores que capturam a luz e convertem para uma imagem digital. A segunda etapa é o processamento da imagem digital, a fim de obter informações salientes como profundidade, detecção de movimento, variações de cor, detecção de objetos, reconhecimento de cena, entre outros.

Sensores de visão são largamente utilizados em aplicações de robótica pelo seu baixo custo, além dos benefícios que a quantidade de recursos visuais proporcionam ao robô. Um destes recursos agrega ao aparato robótico a capacidade de saber o quanto desloca-se no ambiente dado dois ou mais *frames* capturados pelo sensor.

Há diversos estudos sobre sensores de visão (câmeras) e suas peculiaridades como em Bigas et al. [2006], Fossum [1993], Theuwissen [2008] que fogem do escopo deste trabalho. Em Siegwart et al. [2011], Nixon [2008] é descrito de forma detalhada o processo de captura e representação da imagem pelo sensor. Para este trabalho, é assumido então que a aquisição de uma imagem é o processo que converte uma cena tridimensional em bidimensional sem a preocupação de como a mesma foi discretizada pelo *hardware*. O modelo geométrico de câmera utilizada para formação de imagens é descrito a seguir.

Modelo Geométrico do Sensor

A formação de uma imagem em um plano depende da orientação com que os raios de luz incidem neste plano (ortogonal, cônica, axonométrica, oblíqua) e, esta orientação representa o ponto de vista do observador em relação ao objeto tridimensional observado [CONCI, 2008]. Existem duas abstrações que descrevem geometricamente como a projeção do objeto tridimensional observado é representado bidimensionalmente, sendo estas as projeções ortográficas e perspectivas apresentadas na Figura 2.4. No entanto, uma imagem capturada por uma câmera física é descrita geometricamente em um plano somente por projeção perspectiva [Hughes et al., 2014]. Este modelo de câmera conhecido como *Pinhole* é descrito a seguir.

Uma câmera *pinhole* é uma caixa fechada com um pequeno furo (*pinhole*) na frente, onde a luz que parte do objeto no mundo atravessa o furo e forma uma imagem invertida deste objeto na parte de trás da caixa. A literatura utiliza a projeção da imagem em um plano virtual, colocado à frente do centro de projeção, em vez de atrás do mesmo, para que facilite a representação do modelo matemático da câmera.

O furo onde os raios convergem é chamado de Centro Ótico, o qual assume o marco do sistema de coordenadas (u, v, w). O Plano Virtual é formado ao longo do eixo w (Eixo Óptico) e o ponto de encontro entre este eixo e o plano é chamado de Ponto Principal. A distância entre o centro ótico e o ponto principal é conhecido como Distância Focal. A Figura 2.5 esboça o modelo de câmera *pinhole* e sua representação geométrica.

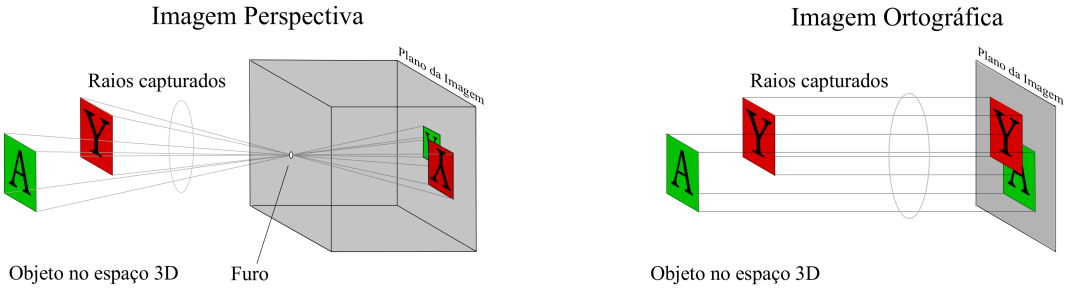

Figura 2.4: Tipos de projeções. FONTE: Modificado de wik [2017a]. A Imagem Perspectiva é formada de acordo com a posição e abertura de um furo, que limita a passagem dos raios de luz que não cruzam em um único ponto, onde estes são projetados por um objeto. A Imagem Ortográfica é formada por raios de luz paralelos que incidem o plano.

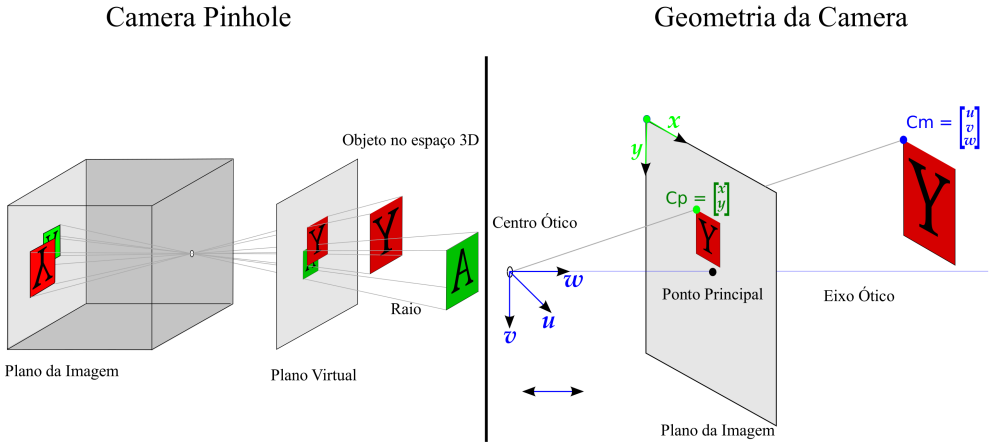

Figura 2.5: Câmera *pinhole*. FONTE: Modificado de wik [2017a]. A Câmera *Pinhole* utilizada pela literatura projeta a imagem em um Plano Virtual, colocado à frente do centro de projeção (furo), em vez de atrás do mesmo, para que facilite a representação do modelo matemático da câmera; A Geometria da Câmera nomeia o furo como Centro Ótico onde assume um sistema de coordenadas 3D (u, v, w). O Plano Virtual se transforma no Plano da Imagem e assume coordenadas 2D (x, y). O ponto de referência 2D (Cp) e 3D (Cm) são então relacionados pelo raio que parte do centro ótico e intercepta ambos.

Ao assumir que o centro ótico é referência do sistema de coordenadas, por uma semelhança de triângulos pode-se relacionar um ponto tridimensional $Cm = [u, v, w]'$ com sua projeção no plano bidimensional $Cp = [x, y]'$, sendo

$$\begin{aligned} x &= \frac{u}{w} f \\ y &= \frac{v}{w} f \end{aligned} \tag{2.1}$$

onde Cm é a coordenada no mundo, Cp é a coordenada no plano e f a distância focal.

Quando uma câmera física captura a projeção do objeto 3D, a posição final do raio projetante é dada em *pixel*. A matriz que agrupa estes *pixels* não necessariamente é quadrada

(número de *pixels* no eixo x é diferente do eixo y), necessitando então uma adequação de dimensão. Logo, um fator de escala é acrescentado na Equação 2.1 sendo

$$\begin{aligned} x &= \frac{\gamma_x u}{w} f \\ y &= \frac{\gamma_y v}{w} f \end{aligned} \tag{2.2}$$

onde γ_x e γ_y são respectivamente o fator de escala no eixo x e y. Estes parâmetros são uma relação da distância focal da câmera com seu campo de visão, cujo resultado são os fatores de escala nos eixos x e y do plano da imagem.

O ponto principal do plano referência ao eixo w do ponto focal, têm a coordenada $Cp = [0, 0]'$ onde não coincide com a coordenada inicial do plano que parte do canto superior esquerdo, sendo necessário parâmetros para transformá-las, logo

$$\begin{aligned} x &= \frac{\gamma_x u}{w} f + \delta_x \\ y &= \frac{\gamma_y v}{w} f + \delta_y \end{aligned} \tag{2.3}$$

onde δ_x e δ_y são respectivamente fatores que transladam a coordenada do ponto principal no plano da imagem para o canto superior esquerdo da mesma.

Outro fator a ser considerado é a inclinação do plano em relação ao eixo x, embora na maioria das câmeras este parâmetro seja 0. Este fator é representado por θ e é agregado a Equação 2.3 sendo

$$\begin{aligned} x &= \frac{\gamma_x u f + \theta v}{w} + \delta_x \\ y &= \frac{\gamma_y v}{w} f + \delta_y \end{aligned} \tag{2.4}$$

e, utilizando coordenadas homogêneas obtemos a seguinte equação

$$\gamma_j \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f\gamma_x & \theta & \delta_x & 0 \\ 0 & f\gamma_y & \delta_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} u \\ v \\ w \\ 1 \end{bmatrix} \tag{2.5}$$

onde γ_j é um fator de escala homogêneo arbitrário.

Na Figura 2.5, a posição do objeto 3D é referenciada pelo sistema de coordenadas do centro ótico da câmera. Pode haver um sistema de coordenadas de referência que represente um mundo onde vários objetos e várias câmeras façam parte. Desta forma, antes que ocorra a transformação da coordenada do objeto 3D para o plano usando o modelo de projeção da Equação 2.5, é necessário que passe por uma transformação de coordenada referente ao mundo para coordenada referente à câmera, sendo

$$\begin{bmatrix} u' \\ v' \\ w' \end{bmatrix} = \begin{bmatrix} \omega_{11} & \omega_{12} & \omega_{13} \\ \omega_{21} & \omega_{22} & \omega_{23} \\ \omega_{31} & \omega_{32} & \omega_{33} \end{bmatrix} \begin{bmatrix} u \\ v \\ w \end{bmatrix} + \begin{bmatrix} \tau_x \\ \tau_y \\ \tau_z \end{bmatrix} \tag{2.6}$$

ou

$$Cm' = \Omega Cm + \tau \tag{2.7}$$

onde Cm' é o ponto transformado, Ω é uma matriz de rotação 3×3 e τ é um vetor de translação 3×1.

A equação da projeção de um objeto 3D referente ao mundo para o plano de imagem de uma câmera com referência neste mesmo mundo, utilizando coordenadas homogêneas dá-se por

$$\gamma_j \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f\gamma_x & \theta & \delta_x \\ 0 & f\gamma_y & \delta_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \omega_{11} & \omega_{12} & \omega_{13} & \tau_x \\ \omega_{21} & \omega_{22} & \omega_{23} & \tau_y \\ \omega_{31} & \omega_{32} & \omega_{33} & \tau_z \end{bmatrix} \begin{bmatrix} Cm_u \\ Cm_v \\ Cm_w \\ 1 \end{bmatrix} \tag{2.8}$$

A Equação 2.8 apresenta o modelo completo de câmera. Neste modelo, dois conjuntos de parâmetros que configuram uma câmera são apresentados, sendo os parâmetros intrínsecos ($f\gamma_x$, $f\gamma_y$, θ, δ_x, δ_y) e extrínsecos (Ω, τ). Os parâmetros intrínsecos (ou matriz de calibração da câmera) configuram as propriedades internas da câmera como campo de visão, distância focal, entre outros, diferente dos parâmetros extrínsecos que descrevem a rotação e translação da câmera em relação ao mundo. Em geral, os parâmetros intrínsecos não mudam entre uma cena e outra, ao contrário dos parâmetros extrínsecos que são alterados com o movimento da câmera [Nixon, 2008]. Pode-se reescrever a Equação 2.8 da seguinte forma

$$\gamma_j Cp = \Lambda[\Omega|\tau]Cm \tag{2.9}$$

onde Cp é a coordenada no plano da câmera, Λ é a matriz com os parâmetros intrínsecos, Ω é a matriz de rotação, τ é o vetor de translação e Cm é a coordenada do objeto no mundo.

Defeitos no Sensor

Após descrever o modelo completo da câmera *pinhole*, é válido salientar que as câmeras do mundo real não utilizam somente um furo para receber os raios de luz e orientá-los no receptor do equipamento, elas possuem uma ou várias lentes que convergem vários raios de luz para uma abertura que não necessariamente é um furo (*pinhole*). Estas lentes somadas ao tamanho da abertura e aos inúmeros defeitos que ocorrem no processo de fabricação podem causar deformações na imagem do plano. As deformações radiais e tangenciais são as mais conhecidas, onde não são previstas no modelo matemático da câmera. No entanto, há formas de tratá-las.

Distorção radial é uma deformação linear da imagem que depende da distância do raio a partir do centro da imagem. Isto ocorre quando o campo de visão do sistema de lentes é grande, e pode ser facilmente detectado em uma imagem já que as linhas retas no mundo não aparecem na imagem projetada como retas e sim como curvas [Prince, 2012].

Segundo Prince [2012], Szeliski [2010], Bradski and Kaehler [2008] a distorção radial geralmente é modelada como uma função polinomial da distância r a partir do centro da imagem, onde a posição (x', y') do *pixel* na imagem é expressa como função da posição (x, y) original do mesmo, dado por

$$\begin{aligned} x^{'} &= x(1 + \beta_1 r^2 + \beta_2 r^4 + \beta_3 r^6) \\ y^{'} &= y(1 + \beta_1 r^2 + \beta_2 r^4 + \beta_3 r^6) \end{aligned} \tag{2.10}$$

onde os parâmetros $\beta 1$, $\beta 2$ e $\beta 3$ modelam o grau de distorção.

A Equação 2.10 descreve uma gama de possíveis distorções que ocorrem em uma variedade de lentes comuns. É aplicada ao modelo *pinhole* após a projeção perspectiva e antes dos efeitos dos parâmetros intrínsecos.

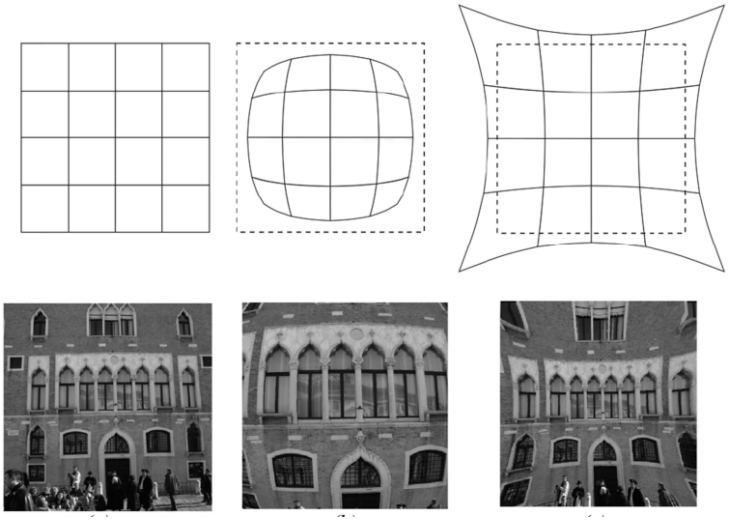

Figura 2.6: Distorções na imagem. FONTE: Extraído de [Siegwart et al., 2011, p. 157]. Da esquerda para direita. Imagem sem distorção; imagem com distorção radial tipo barril; imagem com distorção tipo almofada.

A Distorção Tangencial, conforme Bradski and Kaehler [2008], Horn [2000], dá-se pelo defeito de fabricação da câmera onde a lente e o sensor (*Charge Coupled Device* (CCD) ou *Complementary Metal Oxide on Silicon* (CMOS)) não foram posicionados paralelamente. Assim como a distorção radial, a tangencial também cresce conforme se distancia do centro da imagem. A função utilizada para correção é dada por

$$\begin{aligned} x^{'} &= x + (2\rho_1 y + \rho_2(r^2 + 2x^2)) \\ y^{'} &= y + (\rho_1(r^2 + 2y^2) + 2\rho_2 x) \end{aligned} \tag{2.11}$$

onde $\rho 1$ e $\rho 2$ também são parâmetros que modelam o grau de distorção.

Calibração do Sensor

Nos tópicos anteriores definiu-se o modelo geométrico da câmera *pinhole* e suas equações, conforme seus parâmetros físicos de configuração como distância focal, dimensões do sensor receptor e distorções ocorridas pela lente e defeitos de fabricação. Estes parâmetros físicos de configuração da câmera não são fornecidos pelo fabricante, logo, uma inferência dos parâmetros intrínsecos, extrínsecos e dos coeficientes de distorção faz-se necessários. Este processo é conhecido como Calibração de Câmera.

Os parâmetros intrínsecos e extrínsecos determinam como pontos de interesse na cena são mapeados para os seus respectivos POIs correspondentes na imagem. Logo, conhecendo a coordenada do POI na imagem e a coordenada 3D do seu correspondente na cena, pode-se calcular os parâmetros da câmera resolvendo a Equação 2.9 de projeção perspectiva.

Uma das primeiras técnicas utilizadas para calibração de câmera foi proposta por Tsai [1987], revisada posteriormente por Horn [2000] e modificada por Zhang [2000] cujo objetivo foi acelerar a técnica de calibração de Tsai utilizando um tabuleiro planar. O método usa várias imagens retiradas de diferentes ângulos e posições deste tabuleiro (vide Figura 2.7) onde são conhecidas as dimensões e correspondências de cada canto no plano de imagem. Resolvendo uma minimização linear por mínimos quadrados seguida por um refinamento não linear (Gauss Newton entre outros), é calculado uma aproximação dos valores de todos os parâmetros. Este método é

simples, a acurácia dos parâmetros dependem do número de imagens utilizadas na calibração e é utilizado largamente por ferramentas como MATLAB [MATLAB, 2010] e OPENCV [Bradski, 2000] onde a técnica de calibração destas é similar à solução de Zhang [Siegwart et al., 2011, Bradski and Kaehler, 2008].

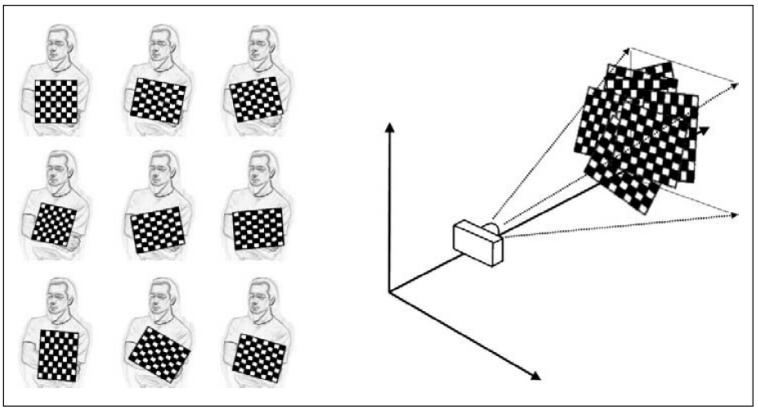

Figura 2.7: Calibração de câmera. FONTE: Extraído de [Bradski and Kaehler, 2008, p. 382]. Método de calibração de câmera proposta por Tsai [1987], Horn [2000], Zhang [2000] que captura várias imagens de variadas posições e ângulos diferentes de um tabuleiro quadriculado planar, onde são conhecidas a dimensão e quantidade dos quadriculados inseridos neste.

O algoritmo para calibração de Zhang [2000] é definido por três etapas. Na primeira, várias imagens do tabuleiro são capturadas por diversos ângulos e posições como apresentado na Figura 2.7. Na segunda etapa, os cantos dos quadrados que compõem o tabuleiro são encontrados por processamento de imagem. A correspondência então é feita entre os POIs 2D nas imagens com os POIs 3D do plano do tabuleiro, sendo este casamento conhecido como homografia (Figura 2.8 a). Ainda, um sistema de coordenadas do mundo é agregado ao POI relativo ao canto de um quadrado no tabuleiro, fazendo com que todos os pontos deste estejam sobre o plano (Figura 2.8 b). Na terceira e última etapa, utilizando as correspondências da segunda etapa, os parâmetros intrínsecos e extrínsecos são estimados.

A base fundamental do algoritmo é a homografia. Bradski and Kaehler [2008] definem homografia como um mapeamento projetivo de um plano para outro, e pode ser expressa como uma multiplicação de matrizes (caso as coordenadas sejam homogêneas) entre um ponto 3D Q (ponto a ser mapeado) para um ponto q na imagem, definido como

$$\begin{aligned} \tilde{Q} &= [XYZ1]^T \\ \tilde{q} &= [xy1]^T \end{aligned} \tag{2.12}$$

onde $\tilde{Q}$ e $\tilde{q}$ representam respectivamente o ponto 3D Q e o ponto 2D q em coordenadas homogêneas e, ainda podem ser expressas como

$$\tilde{q} = sH\tilde{Q} \tag{2.13}$$

onde s é um fator de escala arbitrário e H a matriz que mapeia a homografia. Pode-se ainda calcular a transformação inversa, do ponto da imagem para o ponto do tabuleiro planar com

$$\tilde{Q} = sH^{-1}\tilde{q} \tag{2.14}$$

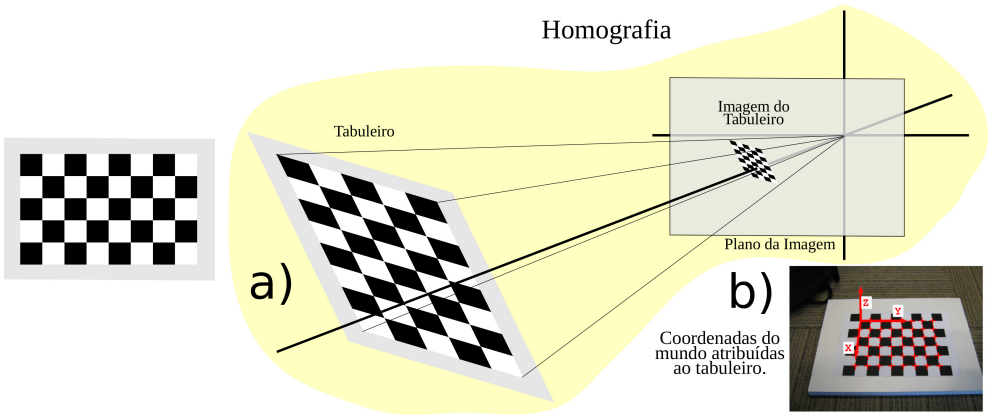

Figura 2.8: Método de calibração de Zhang. FONTE: Modificado de Santos and Rocha [2012]. Etapas que define o método de calibração de câmera refinado por Zhang [2000]. Na primeira etapa imagens do tabuleiro são capturadas conforme Figura 2.7. Na segunda etapa (a), os cantos dos quadrados que compõem o tabuleiro são encontrados por processamento de imagem. A correspondência entre os POIs então é feita e a homografia é estimada. Em (b), um sistema de coordenadas do mundo é atribuído ao canto de um quadrado na quina do tabuleiro, fazendo com que todos os pontos deste estejam sobre o plano. Na terceira e última etapa são estimados os parâmetros intrínsecos e extrínseco utilizando as correspondências entre POIs 2D (da imagem) e 3D (do tabuleiro).

Tanto a homografia quanto os parâmetros intrínsecos e extrínsecos são estimados, existindo na literatura inúmeras formas de resolvê-los, como os métodos de Tsai [1987], Zhang [2000] amplamente utilizados, e métodos diferentes ou mais atuais como em Mendonca and Cipolla [1999], Xu et al. [2000], Sirisantisamrid et al. [2004], Chen et al. [2008], Kukelova et al. [2013], Maddock and Maddock [2015].

Livros como Prince [2012], Siegwart et al. [2011], Hartley and Zisserman [2004], Bradski and Kaehler [2008] apresentam de forma mais detalhada os cálculos de calibração. A biblioteca OPENCV utiliza o método de Zhang para resolver os parâmetros da câmera, diferenciando somente quanto à resolução dos parâmetros de distorção onde utiliza o método de Brown Duane [1971].

Recuperação da Profundidade

No processo de criação da imagem, um POI tridimensional é convertido em um POI bidimensional, acarretando perda de uma dimensão, a profundidade. Para o ser humano, assim como para um robô, perceber a profundidade lhe fornece vantagens, como desviar-se de obstáculos e mensurar seu deslocamento por um ambiente.

O homem percebe a profundidade no ambiente de variadas maneiras. A Estereopsia ou Visão Binocular Estereoscópica é a mais conhecida, pois descreve um aparato biológico que visualiza uma cena por dois pontos de vista diferentes, inerentes à disposição física dos receptores, os olhos, de modo que o cérebro recebe as imagens e, pela diferença entre ambas cria a sensação de profundidade. A visão computacional utiliza do mesmo processo, capturando diferentes imagens do ponto 3D por diferentes pontos de vista da câmera. Este processo é estudado por duas vertentes conhecidas como Estrutura por Visão Estéreo e Estrutura por Movimento (SfM) [Siegwart et al., 2011].

Estrutura por Visão Estéreo é o processo que obtém informação de profundidade por meio de um par de imagens captadas por duas câmeras relacionadas (posição e orientação relativa entre ambas) que visualizam a mesma cena de diferentes posições. Diferente desta, a SfM obtém a informação de profundidade por intermédio de duas ou mais imagens capturadas de diferentes ângulos da mesma cena por uma ou várias câmeras não relacionadas. Estes processos têm dois problemas: a correspondência entre as imagens e a reconstrução 3D. O primeiro problema trata de detectar POIs correspondentes para que as imagens sejam relacionadas geometricamente. Este assunto é tratado separadamente e abordado no próximo capítulo. O segundo problema é conhecido como Geometria Epipolar e trata do relacionamento geométrico entre imagens desde que se conheça a correspondência entre seus POIs [Siegwart et al., 2011].

A geometria epipolar considera que um POI 3D sempre estará localizado sobre a linha projetante que parte do centro focal, atravessa o POI projetado no plano e continua infinitamente. A busca da posição deste POI nesta linha é conhecida como Restrição Epipolar e necessita de outra imagem que observa o mesmo POI 3D de outra posição. Esta outra imagem projeta uma linha que parte de seu foco, intersecta o POI projetado no plano e tende ao infinito, porém agora, esta intersecta a linha projetante da primeira imagem. Estas linhas projetadas em seus respectivos planos são conhecidas como linhas epipolares e dependem dos parâmetros intrínsecos das câmeras e a posição de translação e rotação relativa entre ambas, determinada pelos parâmetros extrínsecos [Prince, 2012]. A Figura 2.9 representa esta descrição.

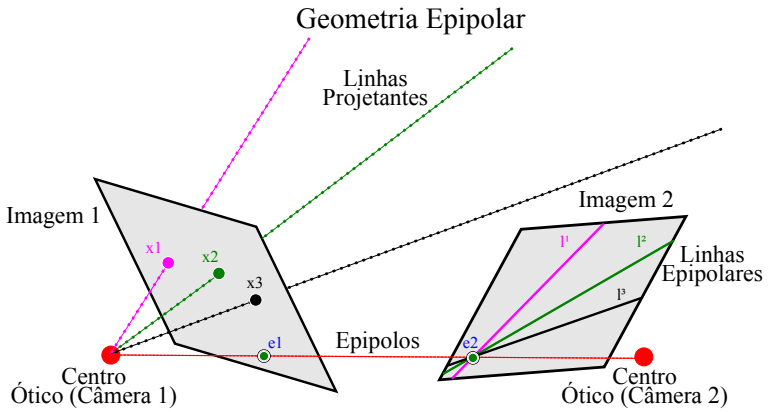

Figura 2.9: Geometria Epipolar. FONTE: Modificado de [Prince, 2012, p. 425]. Raios que partem do centro ótico da câmera 1 interceptando POIs $x1$, $x2$ e $x3$ (respectivamente rosa, verde e preto) no plano da imagem, induzem linhas epipolares na imagem 2, as quais convergem para o epipolo $e2$. Este epipolo é a projeção (linha vermelha) do centro ótico da câmera 1 na imagem 2 ou vice-versa, criando relações entre câmeras.

Ainda segundo Prince [2012], restrições epipolares têm duas implicações práticas sendo:

- Dados os parâmetros extrínsecos e intrínsecos, para buscar um POI na imagem 1, basta procurar este sobre a linha epipolar formada na imagem 2, restringindo assim a busca em uma dimensão;
- A correspondência dos POIs é uma função dos parâmetros intrínsecos e extrínsecos, logo conhecendo somente os parâmetros intrínsecos, pode-se usar a correspondência dos POIs nas imagens para determinar os parâmetros extrínsecos e assim estabelecer uma relação geométrica entre as câmeras.

A projeção de POIs na imagem 1 induzirá linhas epipolares na imagem 2, conforme demonstrada na Figura 2.9. Estas linhas convergem para uma região, não necessariamente no plano da imagem, chamada de epipolo (ponto e2 na imagem 2). Logo, o epipolo ocorrido em um plano é sempre a projeção do centro ótico do outro plano e vice-versa (linha vermelha cruzando os epipolos). Quando as câmeras têm a mesma orientação e estão transladadas perpendicularmente entre seus eixos óticos, as linhas epipolares aparecem paralelamente no plano e a convergência destas (epipolo) dá-se no infinito (Figura 2.10 centro esquerda). Se as câmeras, ainda com a mesma orientação, porém estão transladadas paralelamente entre seus eixos óticos, apresentam linhas epipolares convergindo para o centro do plano (epipolo) (Figura 2.10 centro direita).

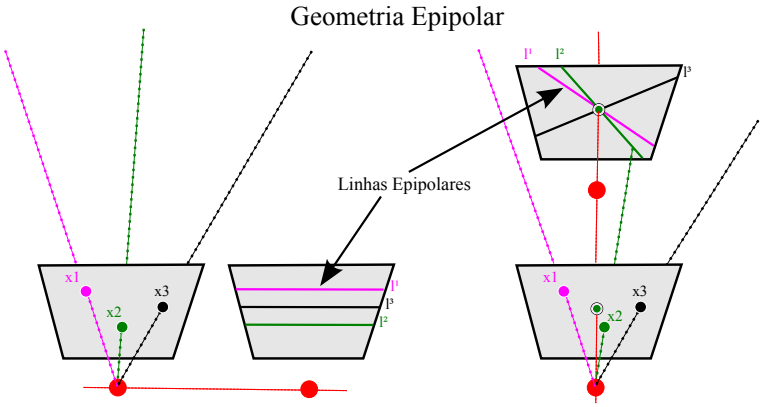

Figura 2.10: Convergência de linhas epipolares. FONTE: Modificado de [Prince, 2012, p. 426]. No centro esquerda da figura é demonstrado como linhas epipolares paralelas são induzidas quando as câmeras tem mesma orientação mas são transladadas perpendicularmente. No centro direta é demonstrado a indução de linhas epipolares que convergem para o centro da imagem quando câmeras com mesma orientação são transladadas paralelamente.

Entendidas as propriedades geométricas das linhas epipolares, pode-se relacionar as câmeras através da Matriz Essencial e Fundamental de forma algébrica. A Matriz Essencial faz a relação puramente geométrica não levando em consideração os parâmetros intrínsecos de cada câmera, já a Matriz Fundamental incorpora informações de calibração das câmeras. Com a geometria epipolar pode-se descobrir onde um POI de uma imagem está em outra, também por meio da correspondência destes POIs pode-se orientar ou descobrir a orientação de uma câmera. Com estas informações, se pode estimar agora a profundidade (posição tridimensional) de um POI dada duas observações diferentes, e então fazer a reconstrução 3D da cena [Hartley and Zisserman, 2004]. As Matrizes Essencial e Fundamental, assim como a estimação da profundidade do POI, são explicadas de forma detalhada no Apêndice B.

A Figura 2.11 exemplifica a utilização da geometria epipolar onde, a localização de um POI no espaço 3D que está projetado em ambas as imagens é a interseção (estrelas vermelhas) de dois raios (linhas em vermelho) infinitos que partem dos centros óticos de cada câmera direcionada ao seu respectivo POI 2D na imagem (círculos em vermelhos), referenciados no mundo do POI 3D que projetou-se nas imagens. No entanto, ruídos na imagem (círculo na imagem em verde) podem fazer com que estes raios nunca se cruzem (linhas em verde), precisando então de um procedimento que minimiza a distância do cruzamento (circulo em verde) conhecido como *Direct Linear Transform* (DLT) ou Transformação Linear Direta [Hartley and Zisserman, 2004].

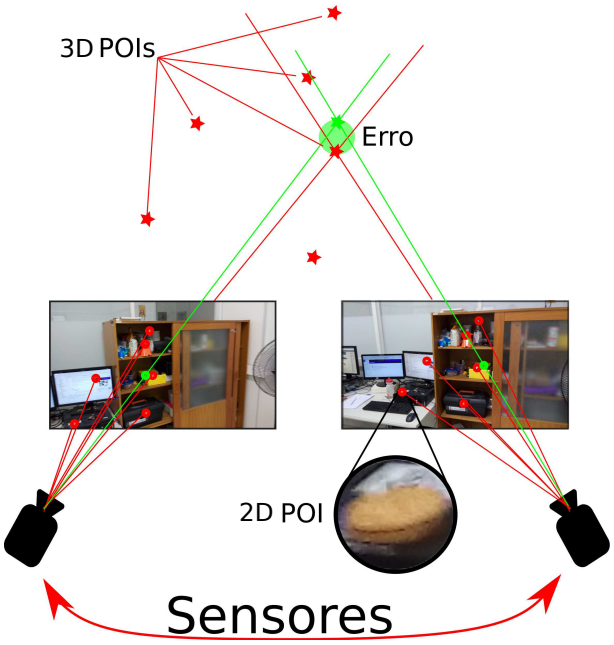

Figura 2.11: Localização do POI no espaço 3D. FONTE: Autor. Um POI no espaço 3D está na interseção (estrelas vermelhas) de dois raios (linhas em vermelho) infinitos que partem dos centros óticos de cada câmera direcionada ao seu respectivo POI 2D nas imagens (círculos em vermelhos). No entanto, ruídos na imagem (círculo na imagem em verde) podem fazer com que estes raios nunca se cruzem (linhas em verde), precisando então de uma Transformação Linear Direta [Hartley and Zisserman, 2004] que minimize a distância do cruzamento (circulo em verde).

2.2.2 Estimação do Movimento

Estimação do movimento é a determinação da trajetória de algo dada relação entre sua posição atual e a anterior. A Odometria Visual (VO) determina o movimento da câmera por meio de imagens relacionadas, dada imagem atual e a anterior. Por meio da concatenação de todos estes movimentos únicos, toda a trajetória da câmera é recuperada [Fraundorfer and Scaramuzza, 2011].

Para reconstrução de um POI 3D por triangulação é necessário que imagens sucessivas sejam observadas. Quando utiliza-se uma única câmera para calcular a odometria, três imagens são necessárias, sendo a primeira imagem para observar POIs, a segunda para reobservar estes POIs e fazer a triangulação para encontrar a posição 3D destes, e a terceira imagem para calcular a transformação (rotação e translação) em relação aos POIs 3D já triangulados.

A escala é o maior problema da odometria visual por visão monocular, pois não pode ser obtida por meio de duas imagens consecutivas, diferente da visão estéreo, onde obtém-se a matriz de transformação dada duas imagens capturadas no mesmo instante. Usualmente um fator de escala constante é definido inicialmente ou toda transformação subsequente segue a escala calculada nas duas primeiras imagens. A escala global não pode ser obtida sem uma heurística do ambiente ou transformação inicial. Alguns trabalhos utilizam como heurística sensores do tipo *Inertial Measurement Unit* (IMU), rodas com *encoders* ou GPS [Yousif et al., 2015].

O procedimento que mensura a odometria visual monocular é semelhante à estéreo, diferenciando-se somente no tempo onde ocorre triangulação entre duas imagens. Na visão por estéreo são duas imagens capturados paralelamente, no mesmo instante de tempo e relacionadas

entre si, já a visão monocular são duas imagens capturadas serialmente (tempos diferentes) e não relacionadas. Uma estrutura para este procedimento é proposta por Fraundorfer and Scaramuzza [2011], Yousif et al. [2015] utilizando transformações de 3D para 2D, e estimando o movimento com visão monocular, apresentado a seguir:

1. Extrair e descrever POIs da primeira imagem F_i no tempo i.
2. Extrair e descrever POIs da segunda imagem F_{i+1}
3. Corresponder os POIs entre ambas as imagens. Estimar a matriz de transformação com uma escala predefinida entre as duas imagens. Fazer triangulação dos POIs correspondentes (Equação B.17).
4. Capturar uma nova imagem F_{i+2}.
5. Detectar, descrever POIs da imagem F_{i+2} e corresponder com POIs da imagem F_{i+1}.
6. Estimar a matriz de transformação 3D para 2D (*Perspective of n Points* (PnP) - Perspectiva por n Pontos de Interesse) da imagem F_{i+2} (Equação B.18).
7. Fazer triangulação dos novos POIs correspondentes entre F_{i+2} e F_{i+1} (Equação B.17).
8. Atualizar a imagem F_{i+1} com a imagem F_{i+2}.
9. Iterar a partir de 4.

A imagem 2.12 apresenta de forma simplificada o processo de odometria. Durante os passos 3, 6 e 7, erros na correspondência de POIs podem gerar resultados não confiáveis. É necessário que se escolha melhores correspondências para o cálculo. Para resolver isto, *Random Sample Consensus* (RANSAC) [Fischler and Bolles, 1981] identifica um conjunto de correspondências que melhor determinem os parâmetros do modelo de transformação entre imagens. Por meio de regressão linear, inicia um conjunto de pontos correspondentes sorteados e calcula os parâmetros do modelo, após checa se todos os pontos correspondentes são consistentes com os parâmetros já calculados. Caso o valor da checagem de consistência seja menor que um *threshold*, o conjunto que foi utilizado para calcular os parâmetros de transformação entre imagens será o representativo, caso contrário RANSAC continuará sorteando novos conjuntos até encontrar um que seja representativo, ou até que o número de iterações seja excedido. [Prince, 2012].

Dado que a Odometria Visual concatena transformações pose após pose da câmera, pequenos erros nas transformações são propagados durante cada passo, podendo gerar um grande erro ou escorregamento (*drift*) na trajetória final. Um refinamento iterativo pode ser usado para minimizar a soma do quadrado dos erros de reprojeção dos pontos 3D sob seus respectivos pontos 2D, reconstruídos pela triangulação entre as últimas imagens. Este método é conhecido como *Bundle Adjustment* (BA) ou Ajuste de Pacotes e resolve problemas grandes de otimização não linear utilizando o algoritmo de Levenberg-Marquardt. De forma prática otimiza os parâmetros da pose da câmera e dos pontos 3D dado um conjunto de *frames*. Pode ser usado tanto para ajustar toda a odometria calculada (Global BA) como parcelas desta (Local BA) [Prince, 2012, Hartley and Zisserman, 2004, Szeliski, 2010].

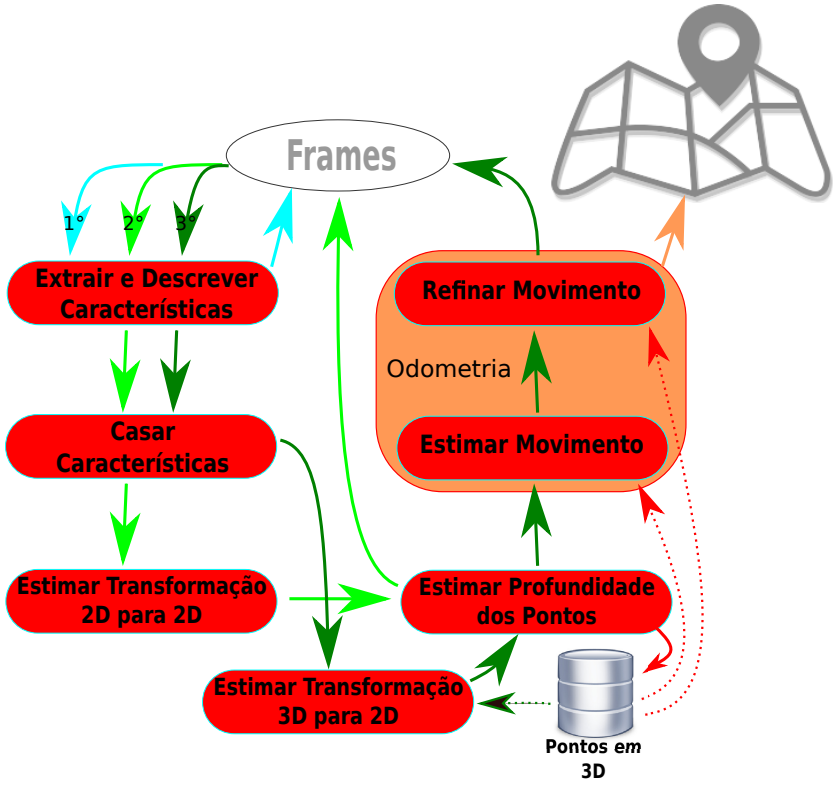

Figura 2.12: Fluxo da odometria visual. FONTE: Autor. A imagem demonstra o fluxo da odometria feita entre três *frames*. No 1º e 2º *frame* (seta azul e verde claro) são detectados, descritos e correspondidos os POIs. Estima-se então a transformação 2D para 2D entre os *frames* 1º e 2º como também a posição 3D dos pontos correspondentes e as armazena (seta vermelha cheia). Com a captura do 3º *frame* (seta verde escuro), detecta-se, descreve-se e corresponde-se seus POIs com POIs do 2º *frame* porém, como se conhece a localização 3D (seta verde escuro pontilhada) dos POIs do 2º *frame*, a transformação entre imagens passa a ser de 3D para 2D. A estimação 3D de POIs que sobraram é então calculada e armazenada (seta vermelha cheia) para próxima iteração. Como se conhece agora a pose do 2º e 3º *frame*, a diferença entre as poses é calculada para estimar o tamanho do deslocamento do 3º *frame* relativo ao 2º e construir o mapa. Erros de escorregamento podem ocorrer e degenerar o mapa, então um refinamento da localização e deslocamento relativo entre poses de três ou mais *frames* é feito para corrigir a trajetória da câmera.

2.3 Resumo

Este capítulo apresentou os fundamentos sobre o problema da Localização e Mapeamento Simultâneos baseado em Visão. Mostrou as características do SLAM, VSLAM, tipos de resolução, e sensores utilizados para estimar o deslocamento pelo ambiente. Além disso, o capítulo descreveu o processo de aquisição de imagem por um sensor passivo tipo câmera. Nesta parte também definiu-se o modelo geométrico de câmera, assim como seus parâmetros intrínsecos, extrínsecos e as suas formas de calibração. A forma de recuperar a posição tridimensional (profundidade) de um POI dada sua correspondência em duas imagens por intermédio da geometria epipolar foi demonstrada. Foi apresentada a relação geométrica de duas câmeras representadas pela Matriz Essencial e Fundamental, bem como o processo de aferição do deslocamento pela transformação geométrica de uma câmera em relação a outra, ou diferentes *frames*. Portanto, este capítulo

apresenta a fundamentação necessária para a compreensão do processo de resolução do problema de mapeamento e localização simultâneos por um robô através de visão computacional. Os problemas que correm no processo de Estimação do Movimento, estão relacionados à qualidade dos POIs detectados nas imagens. Logo, no próximo capítulo serão apresentados diversos métodos de detecção de POIs para utilização no processo de Odometria Visual.

Capítulo 3

Detecção de Características

Este capítulo apresenta os mecanismos de detecção de características para identificar pontos de interesse em imagens. A Seção 3.1 descreve a definição de pontos de interesse em uma imagem e suas propriedades. Logo, a Seção 3.2 apresenta os métodos que detectam estes pontos. Por fim, a Seção 3.3 demonstra os métodos que descrevem os pontos de interesse já detectados.

3.1 Características da Imagem

Entender o processo de aquisição de uma imagem e o modelo matemático de câmera é de extrema importância para que um robô crie mapas do ambiente, localize-se e identifique objetos na cena. Porém, para que possamos relacionar duas ou mais imagens do mesmo ambiente de pontos de vista diferentes, é necessário encontrar pontos de interesse (POIs) comuns entre as imagens (Método por Pontos Chave), ou correspondê-las de forma direta (Método Direto), para que seja recuperada a terceira dimensão do ambiente, antes perdida no processo de aquisição da imagem. Estas duas formas de relacionar imagens são largamente estudadas na literatura dentro do tema Processamento de Imagens. Dado que o foco da dissertação está relacionado ao método por pontos chave com aprofundamento na detecção de POIs, o método direto não será abordado, porém pode ser compreendido em Goh and Vidal [2006], Sheikh et al. [2007], Park and Kweon [2001], Irani and Anandan [2000], Alismail and Browning [2014], Stein and Shashua [1997], Heel [1990]

Siegwart et al. [2011] definem pontos de interesse como sendo uma estrutura reconhecida de um elemento dentro de um ambiente. Esta pode ser extraída a partir de medidas e descritores matemáticos. Bons POIs são facilmente percebidos no ambiente. Distinguem-se como de baixo nível (formas geométricas) sendo linhas, pontos, cantos, regiões, círculos ou polígonos, e de alto nível (objetos) como portas, mesas, janelas, carros, pessoas, entre outros. Um POI local é um padrão na imagem que diferencia-se de seus vizinhos. Usualmente é associado a mudanças de propriedades na imagem geralmente como intensidade, cor e textura. Estas mudanças são utilizadas como medidas por meio de valor ou conjunto de valores que as represente, tipicamente chamado de descritores.

Siegwart et al. [2011], Tuytelaars and Mikolajczyk [2008] descrevem bons pontos de interesse locais como tendo as seguintes propriedades:

- *Repetibilidade*: Dada duas imagens da mesma cena ou objeto, tiradas de pontos de vista diferentes, uma alta porcentagem de características detectadas na cena devem ser detectadas em ambas as imagens.

- *Distinção*: Os padrões de intensidade das características detectadas devem apresentar inúmeras variações para que sejam distintas.
- *Localidade*: A característica deve ser local e permitir um modelo simples de aproximação quando há deformação geométrica e fotométrica entre duas imagens, adquiridas de vistas e condições diferentes.
- *Quantidade*: O número de características detectadas devem ser suficientemente grande para que haja representação até de objetos pequenos. No entanto, este número depende da aplicação.
- *Acurácia*: A característica detectada deve ter acurácia na sua localização dentro da imagem com respeito a escala e formato.
- *Eficiência*: Preferivelmente, a detecção de características na imagem deve ocorrer no menor tempo possível.
- *Invariabilidade*: Quando grandes deformações são esperadas entre duas imagens, um modelo matemático destas deformações é criado. Assim, o método para detecção deve trabalhar independentemente do modelo matemático, de forma que o POI se torne invariável as deformações.
- *Robustez*: Em caso de pequenas deformações, o método de detecção deve ser menos sensível para que não rejeite a correspondência. A acurácia diminui mas não drasticamente. Tipicamente, as deformações encontradas e tratadas pela robustez são os ruídos na imagem, efeitos da discretização, efeitos da compressão, borramento, entre outros. Também os desvios geométricos e fotométricos do modelo matemático usado para obter a invariância são frequentemente superados pela inclusão de mais robustez.

O método de pontos chave utiliza duas técnicas para encontrar e descrever um ponto de interesse em uma imagem, sendo respectivamente, os Detectores e Descritores de características, comentados a seguir.

3.2 Detectores

Conforme a definição de Siegwart et al. [2011], podemos detectar pontos de interesse peculiares em imagens como bordas, cantos, regiões (*blobs*) e outros, sendo estes três primeiros mais estudados na literatura.

Uma borda é a divisa entre duas regiões pertencentes a uma imagem (Figura 3.1 B). Esta borda ocorre pelo alto gradiente entre as regiões. O canto diferencia-se da borda por ter direções diferentes de gradiente formando assim uma curvatura acentuada (Figura 3.1 A). O *blob* detecta uma região descriminante da imagem dada por altos gradientes variando em todas as direções (Figura 3.1 D). E em uma região homogênea não há variação do gradiente (Figura 3.1 C).

Existem diversas técnicas para detectar estes pontos de interesse. Alguns autores têm estudado a curvatura de contorno para encontrar cantos. Outros analisam as intensidades da imagem baseado em derivativas ou regiões com alta variância. Outras linhas de pesquisa são inspiradas no sistema de visão humana e tentam reproduzir o processo cerebral para tal. Alguns exploram informações das cores da imagem. Algumas abordagens são baseadas em modelos e outras em segmentação. Problemas como invariância de transformação geométrica, escala e afim são abordados nos estudos [Tuytelaars and Mikolajczyk, 2008].

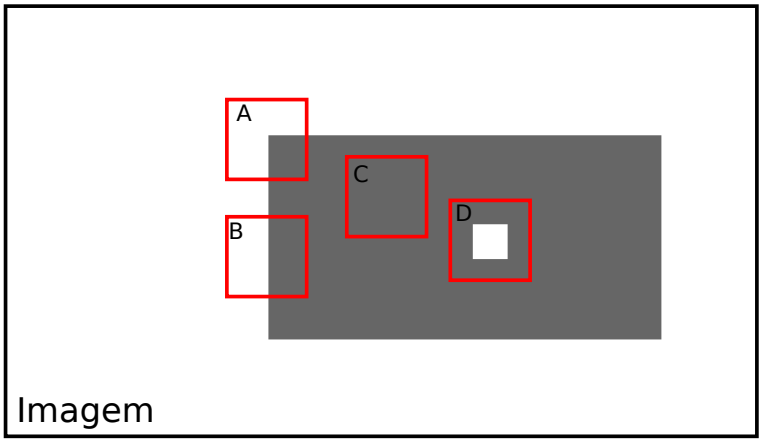

Figura 3.1: Detecção de pontos de interesse. FONTE: Autor. O quadrado vermelho (A) destaca um canto onde as variações de intensidade ocorrem em duas direções; em (B) as intensidades variam em uma única direção; em (C) ocorre pouca variação de intensidade por ruídos ou não ocorre variações; e em (D) há variações de intensidade em todas as direções.

Este trabalho apresenta a seguir algoritmos de detecção de pontos de interesse que serão utilizados para comparações futuras. Detectores que são referência teórica para estes algoritmos são abordados no Apêndice A.

3.2.1 SIFT

Scale Invariant Feature Transform (SIFT) foi concebido por Lowe [1999, 2004] e é um método que detecta pontos de interesse extremamente distintos, têm alta repetibilidade e são invariantes a rotação, escala, pontos de vista e iluminação.

Diferente do método de Harris (vide Apêndice A.2), *Scale Invariant Feature Transform* (SIFT) associa a escala e a orientação em cada ponto de interesse por meio de dois estágios, detecção e descrição, sendo a descrição abordada no tópico de Descritores.

O estágio de detecção do ponto de interesse é dividido em três etapas sendo:

1. Detecção de extremos no espaço de escala;
2. Localização do ponto de interesse;
3. Orientação do ponto de interesse.

Na primeira etapa o método utiliza a *Diference of Gaussian* (DoG) ou diferença Gaussiana para identificar possíveis pontos de interesse. Lowe [1999, 2004] define o espaço de escala como uma função $L(x, y, \sigma)$ produzida pela convolução (operador $\otimes$) de um filtro Gaussiano $G(x, y, \sigma)$ em uma imagem $I(x, y)$, ou seja:

$$\begin{aligned} L(x, y, \sigma) &= G(x, y, \sigma) \otimes I(x, y) \\ G(x, y, \sigma) &= \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \end{aligned} \tag{3.1}$$

Para que o cálculo seja mais simples e eficiente computacionalmente, é utilizado um fator de escala γ para variar a distribuição do filtro Gaussiano entre as imagens, de modo que

a diferença entre estas é uma aproximação ao *Laplacian of Gaussian* (LoG) ou Laplaciano do Gaussiano feita para encontrar os extremos $D(x, y, \sigma)$, conforme a equação:

$$\begin{aligned} D(x, y, \sigma) &= (G(x, y, \gamma\sigma) - G(x, y, \sigma)) \otimes I(x, y) \\ &= L(x, y, \gamma\sigma) - L(x, y, \sigma) \end{aligned} \tag{3.2}$$

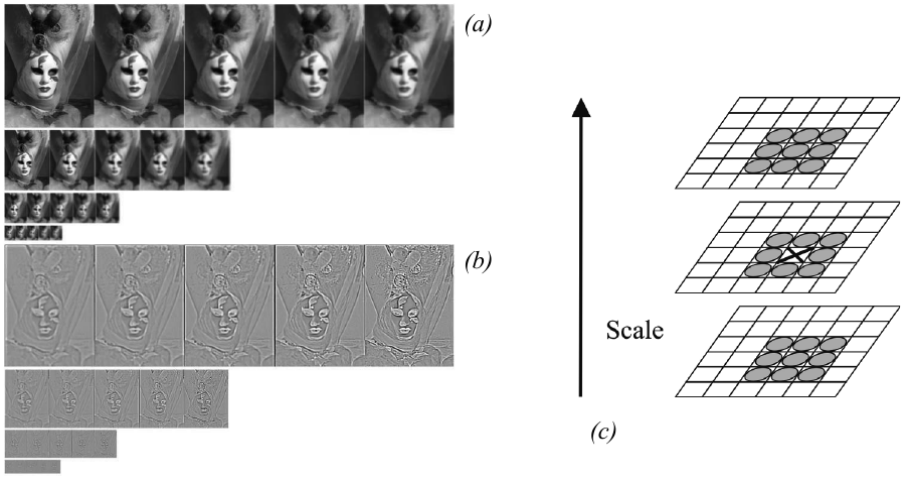

Figura 3.2: Detecção de extremos no espaço de escala - SIFT. FONTE: Retirado de [Siegwart et al., 2011, p. 229]. Em (a), as imagens na vertical representam a pirâmide feita por oitavas, e na horizontal os borramentos Gaussianos de diferentes tamanhos de filtro; em (b) é apresentado a diferença entre imagens Gaussianas de (a) em todas as oitavas; e em (c) é comparado o pixel central com seus oito vizinhos, incluindo as escalas, para identificar extremos máximos e mínimos, sendo estes candidatos a POIs em suas escalas.

A Figura 3.2 demonstra os passos para detecção de pontos de interesse. Primeiro é construída a pirâmide da imagem original por oitavas. Após, a imagem original de cada oitava é borrada por um filtro Gaussiano multiplicado por um fator γ para gerar as escalas (Figura 3.2 (a)). Em seguida é calculada a diferença entre as escalas, para cada oitava da pirâmide, gerando uma imagem aproximada ao filtro LoG (Figura 3.2 (b)). Por fim, cada *pixel* da imagem gerada pela diferença (DoG) (Figura 3.2 (c) marcada com x) é comparado com seus oito vizinhos e com os 9 *pixels* da escala superior e inferior, para saber se este é um extremo máximo ou mínimo, tornando-se então um possível candidato a ponto de interesse.

A estes extremos são aplicados uma aproximação quadrática local para identificar a posição do pico ou vale. A aproximação quadrática é feita tomando uma expansão de Taylor em torno do ponto atual. Isso fornece uma aproximação de posição que têm resolução de *subpixel* e uma estimativa da escala que é mais precisa do que a resolução da amostragem em escala [Prince, 2012].

Na etapa 2 são feitos alguns tratamentos para eliminar pontos de interesse cujo gradiente dos *pixels* vizinhos são baixos ou que encontram-se ao longo de bordas.

A Figura 3.3 (a) evidencia todos os ponto encontrados na primeira etapa onde estes são máximos ou mínimos. Uma eliminação de pontos de interesse que estão em regiões com gradiente baixo é demonstrado em Figura 3.3 (b). E na Figura 3.3 (c) são os pontos restantes, após a remoção dos pontos nas bordas usando os valores singulares do tensor de estrutura conforme Equação A.7. Estes procedimentos retornam um grupo de pontos de interesse que são localizados com acurácia de *subpixel* e representando devidamente uma escala em particular.

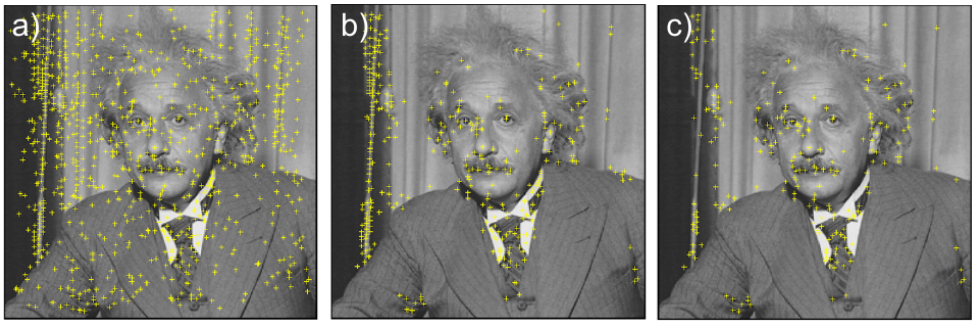

Figura 3.3: Pontos de interesse localizados - SIFT. FONTE: Retirado de [Prince, 2012, p. 340]. Em a), todas os *pixels* máximos e mínimos são representados por pontos amarelos; em b), são eliminados somente POIs que estão em regiões de baixo gradiente; e em c) os POIs em bordas são eliminados conforme valores de resposta da Matriz Hessiana.

A Etapa três finaliza o processo de detecção de pontos de interesse atribuindo-lhe uma orientação. Esta orientação é calculada analisando a intensidade do gradiente (magnitude e orientação) de cada *pixel* vizinho. Um histograma das orientações é construído fazendo com que cada valor do gradiente de cada *pixel* contribua para formação do valor final. Um pico no histograma torna-se a orientação dominante no ponto e, se mais de um pico surgir, um ponto de interesse adicional é criado para cada pico contendo localização e escala do ponto original mensurado. O valor atribuído à orientação do ponto é ponderado pela escala onde foi encontrado (Figura 3.4 (a)). Assim, todas as propriedades dos pontos de interesse serão mensuradas com relação à sua orientação, os tornando então invariantes à escala e rotação (Figura 3.4 (b)) [Lowe, 2004, Nixon, 2008, Prince, 2012, Szeliski, 2010].

Segue as equações que definem a magnitude e orientação:

$$\begin{aligned} M(x,y) &= \sqrt{(L(x+1,y)-L(x-1,y))^2+(L(x,y+1)-L(x,y-1))^2} \\ \theta(x,y) &= tan^{-1}\left(\frac{L(x,y+1)-L(x,y-1)}{L(x+1,y)-L(x-1,y)}\right) \end{aligned} \tag{3.3}$$

onde M é a magnitude do *pixel* e θ é a orientação.

A terceira etapa contribui para a redução da sensibilidade em relação a mudanças do ponto de vista da câmera e alterações não lineares no brilho da imagem (variações lineares são removidas pelas operações de gradiente), analisando regiões na localidade do ponto de interesse.

Em Lowe [2004] esta técnica de detecção é abordada com muito mais detalhes além de descrever fatores que incrementam seu desempenho.

3.2.2 SURF

Bay et al. [2006, 2008] apresentam o método de detecção e descrição de pontos de interesse chamado *Speeded Up Robust Features* (SURF). O método é baseado nos mesmos princípios e passos utilizados pelo SIFT, porém com esquemas diferentes que agilizam o processamento sem perder a robustez.

A detecção de pontos de interesse do SURF usa a Matriz Hessiana (Equação A.6) assim como Harris, para extrair os autovalores e caracterizar o ponto pela determinante. A imagem integral proposta em Viola and Jones [2001] conjuntamente com as janelas de filtro proposto

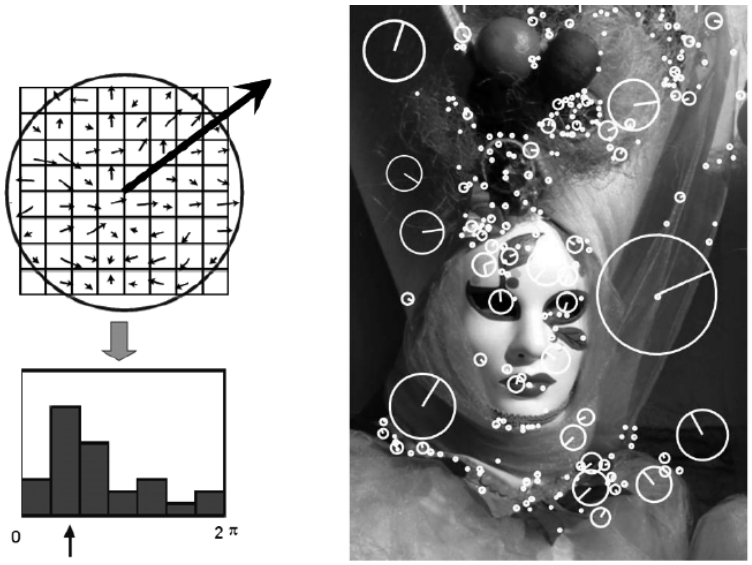

Figura 3.4: Orientação do ponto de interesse - SIFT. FONTE: Retirado de [Siegwart et al., 2011, p. 231]. Em (a), a intensidade do gradiente (magnitude e orientação) de todos os *pixels* vizinhos ao *pixel* central são calculados e após, representados em forma de histograma conforme orientação. Um pico detectado no histograma torna-se a orientação dominante no ponto. Ainda, o valor atribuído a orientação do POI é ponderado pela escala onde este foi detectado; em (b) é apresentado um exemplo de POIs detectados em uma imagem, conforme sua escala e orientação.

em Simard et al. [1999], reduzem drasticamente o tempo de processamento por necessitarem de menos cálculos para determinar o gradiente de uma região. Estes esquemas são o coração do SURF e serão descritos a seguir.

O conceito de imagem integral é armazenar o somatório cumulativo de todos os *pixels* de uma área retangular (Figura 3.5 pontos A, B, C, D), onde o pixel da extrema direita inferior (ponto A) acaba por ter o resultado do somatório de todos os *pixels* pertencentes ao retângulo. Logo, a imagem integral $I_{\sum(x')}$ (Equação 3.4) na coordenada $x = (x, y)$ representa a soma de todos os *pixels* da imagem original na região retangular de início em O e fim em x' conforme a Figura 3.5.

$$I_{\sum(x')} = \sum_{i=0}^{i \leq x} \sum_{j=0}^{j \leq y} I(i, j) \tag{3.4}$$

Depois que a imagem integral estiver calculada, são necessárias somente três adições e quatro acessos a memória para extrair a soma das intensidades dos *pixels* da área retangular. O tempo de cálculo independe do tamanho do retângulo se tornando importante para o SURF, devido ao tamanho das janelas de filtro utilizados para aproximar a Matriz Hessiana, descrito a seguir [Bay et al., 2008].

Para calcular a Matriz Hessiana é necessário derivadas parciais de segunda ordem nas direções x (horizontal), y (vertical) e xy (diagonal) por meio de janelas de filtro Gaussiano (Figura 3.6 centro esquerda). SURF aproxima estas derivadas por filtros de Haar (Figura 3.6 centro direita), onde a região cinza é igual a zero.

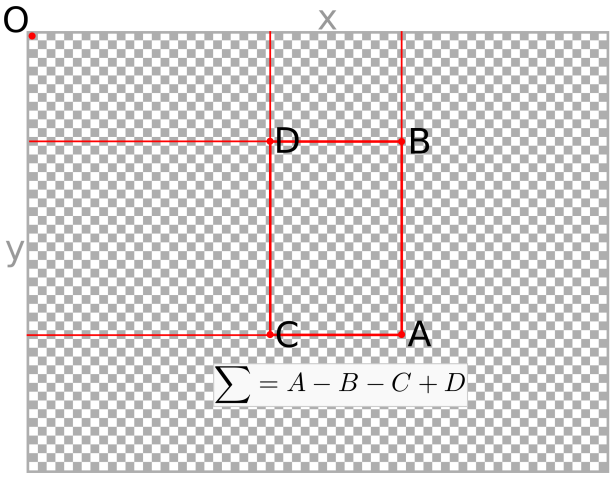

Figura 3.5: Imagem Integral - SURF. FONTE: Autor. A letra (O) representa o início das coordenadas (x) e (y) de uma imagem onde os quadrados brancos e cinzas são posições de *pixels*; em (D) é armazenada a soma acumulativa da intensidade de todos os *pixels* pertencentes a região (O à D), assim como qualquer coordenada da imagem (A, B ou C); então para se calcular o somatório das intensidades de uma região quadrada qualquer, basta acessar os valores de quatro coordenadas (A, B, C e D) da imagem integral e executar três operações (A - B - C + D).

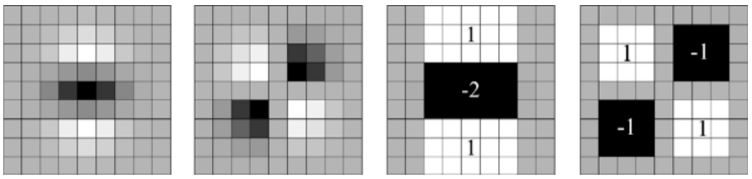

Figura 3.6: Filtros Gaussianos (centro direita) e Filtros Haar (centro esquerda). FONTE: Retirado de [Bay et al., 2008, p. 348]. Os *kernels* centro esquerda representam filtros com distribuições Gaussianas de pesos; os filtros centro direita são conhecidos como Haar e são aproximações dos filtros Gaussianos, estes em conjunto com a imagem integral agilizam o processo de derivação.

Ao convolver na imagem uma janela de filtro 9×9, o resultado gerado representa uma distribuição Gaussiana aproximada com desvio padrão $\sigma = 1,2$ que é a menor escala utilizada para encontrar o ponto característico. Como cada escala aproximada têm um valor de σ diferente, um peso w constante é utilizado para simplificar a equação e manter a energia entre a janela Gaussiana e a janela Gaussiana aproximada. Logo, denotando as derivadas parciais na horizontal como D_{xx}, na vertical como D_{yy} e diagonal como D_{xy}, pode-se calcular a determinante aproximada $Det_(Haprox)$ para qualquer tamanho de janela de filtro conforme Equação 3.5.

$$Det(Haprox) = D_{xx}D_{xy} - (wD_{xy})^2 \tag{3.5}$$

O peso w é sensível a escala mas pode ser mantido com o valor de 0.9 [Bay et al., 2008] sem ter impacto no resultado, conforme Equação 3.6.

$$w = \frac{|L_{xy}(1,2)|_F |D_{yy}(9)|_F}{|L_{yy}(1,2)|_F |D_{xy}(9)|_F} = 0,912... \simeq 0,9 \tag{3.6}$$

onde $||_F$ é a norma de Frobenius, L a derivada de segunda ordem Gaussiana e D a derivada aproximada pela janela de filtro do SURF.

Para que SURF seja invariante a escala, é necessária a criação de oitavas (como SIFT) para encontrar o local máximo. O processo inicia convolvendo uma janela de filtro 9×9 para a menor escala, depois a janela é aumentada para 15×15, 21×21 e 27×27, criando assim quatro escalas para a primeira oitava. Na segunda oitava os filtros aumentam (15, 27, 39, 51). Na terceira para (27, 51, 75, 99) e quarta (51, 99, 147, 195). Diferentemente do método SIFT que diminui pela metade a imagem original em cada oitava, o SURF mantêm as dimensões da imagem original e aumenta para o dobro o tamanho dos filtros. O processo de busca do máximo local é igual ao do SIFT. Bay et al. [2008] detalham em sua pesquisa o processo de aumento do tamanho dos filtros.

3.2.3 GFTT

Good Features to Track (GFTT) ou "Boas Características para Rastreamento", é um método baseado em Fluxo Ótico proposto em Shi and Tomasi [1994] que infere a posição de um *pixel* ou região de um *frame*, no próximo *frame*, identificando a velocidade do fluxo ótico dos *pixels* [Nixon, 2008, Bradski and Kaehler, 2008].

Na área de fluxo ótico três grupos destacam-se, sendo os baseados em técnicas diferenciais, em técnicas de correlação, e em técnicas da frequência ou energia do gradiente. A técnica diferencial leva em consideração que a diferença de intensidade do *pixel* por região, entre imagens diferentes, é constante para pequenos deslocamentos. Algoritmos conhecidos nesta área e baseados em técnicas diferenciais são os pesquisados por Lucas and Kanade [1981], Horn and Schunck [1981] e Shi and Tomasi [1994]. Melhores explicações sobre os diferentes grupos que caracterizam o fluxo ótico podem ser encontrados em Barron et al. [1994].

No estudo de Shi and Tomasi é proposto um monitoramento da qualidade dos pontos de interesse durante o rastreamento, que analisa o quanto a aparência muda entre imagens, utilizando medidas de dissimilaridade. Este estudo demonstra que o método de analisar somente a translação da característica não funciona bem com a dissimilaridade quando os movimentos entre *frames* são pequenos, e que uma transformação afim na imagem é adequada para este caso.

Utiliza o método de minimização de Newton-Raphson para determinar transformação afim na imagem, que assemelha-se ao método utilizado por Lucas and Kanade para somente translação do POI. Definem uma forma de encontrar as melhores características para serem rastreadas. Apresenta um modelo melhorado de rastreamento usando a mistura de dois modelos (translação e afim) demonstrando que somente translação obtém resultados melhores que transformação afim, quando o tempo entre captura de duas imagens é pequeno e, transformações afim são necessárias para comparar duas imagens distantes em termos de tempo de aquisição, utilizando dissimilaridade.

Para não estender a sessão, somente a forma como Shi and Tomasi encontram as melhores características na imagem serão abordadas, pois esta forma de definição de POIs é a relação entre GFTT e o algoritmo IGFTT avaliado. Apenas uma introdução inicial sobre restrição do fluxo ótico será necessária para formação do conhecimento em torno do detector sendo esta descrita no Apêndice A.3.

Shi and Tomasi assumem que boas características podem ser definidas independentemente do método utilizado para o rastreamento, mas que devem ser bem rastreadas entre imagens. Concordando com Harris, Shi and Tomasi perceberam que dois valores pequenos para os autovalores representam uma intensidade constante na janela analisada. Um valor alto e outro baixo representam uma borda e, que dois valores altos representam um canto ou ponto, que são

bons para o rastreamento. Perceberam também que autovalores baixos, se forem suficientemente altos para superar um ruído, a localização deve ser aceita como boa característica. Dada estas premissas, definiram uma equação para escolha do ponto, sendo

$$min(\mu_1, \mu_2) > \mu \tag{3.7}$$

onde μ é valor de corte.

Para definir μ Shi and Tomasi analisam autovalores de janelas na imagem que têm uma distribuição do brilho uniforme, retornando um valor baixo para μ. Após, selecionam vários pontos como cantos e regiões texturizadas para obter um valor alto de μ. Por fim, o valor de μ é selecionado dado a metade do intervalo dos valores alto e baixo.

3.2.4 IGFTT

Improved Good Features to Track (IGFTT) é um método de detecção de pontos de interesse baseado na proposta de Shi and Tomasi [1994] (*Good Features to Track* (GFTT)) e idealizado por Oliveira et al. [2015].

Como o método GFTT completo teve sua concepção com tendências ao fluxo ótico e rastreamento, sua análise dos melhores pontos característicos vai além da somente extração. Após a detecção do ponto de interesse, uma transformação afim é encontrada para posterior comparação de dissimilaridade , o que garante uma correspondência mais robusta em termos de probabilidade. A transformação afim garante invariância a distorções (devido aos pontos de vista diferentes da câmera) e uma pequena escala da imagem. Porém, se analisado somente do ponto de vista de como o mesmo seleciona os melhores pontos em uma única imagem, este não garante invariância a escala.

Para resolver esta necessidade e garantir uma melhor qualidade na detecção de POIs, o método de Oliveira et al. propõe a varredura, não somente em uma única imagem, mas na mesma imagem em diferentes escalas. A técnica utilizada para gerar a pirâmide da imagem é a mesma usada no método SIFT, porém, sem a necessidade de borramento com filtro Gaussiano, pois os autovetores da Matriz Hessiana garantem a invariância ao borramento conforme a escala diminui. Para garantir a orientação do ponto característico, IGFTT utiliza um simples e efetivo cálculo baseado nos autovetores. Estes autovetores preservam a direção dada a transformação linear aplicada na matriz de gradiente [Oliveira et al., 2015].

Por meio da Equação 3.8, os autovetores selecionados são aqueles que correspondem aos menores autovalores da Matriz Hessiana, sendo

$$(x, y) = min(\mu_1, \mu_2) \tag{3.8}$$

As coordenadas (x, y) dos autovetores são usadas para extrair a orientação do ponto por intermédio da Equação 3.9.

$$\theta(x, y) = tan^{-1}\left(\frac{y}{x}\right) \tag{3.9}$$

onde tan^{-1} é a função que calcula a tangente inversa de duas variáveis e retorna o quadrante apropriado do ângulo (em radianos) conforme os sinais dos argumentos.

Para a descrição do ponto detectado o IGFTT usa o *Fast Retina Keypoint* (FREAK) [Alahi et al., 2012], descritor inspirado nos padrões da retina dos olhos, comentado na Sessão 3.3.

IGFTT é um método relativamente novo, testado com variados outros métodos da literatura e obteve resultados animadores tratando-se da performance computacional e da

repetibilidade dos pontos. Características estas que o tornam um robusto candidato a detector de pontos de interesse para VSLAM.

3.2.5 FAST

Proposto em Rosten and Drummond [2006], *Features from Accelerated Segment Test* (FAST) é um detector de canto baseado no detector *Smallest Univalue Segment Assimilating Nucleus* (SUSAN) [Smith and Brady, 1997] que utiliza o centro de uma área circular para determinar a intensidade de todos os seus *pixels* vizinhos. Diferente deste, FAST não analisa toda a área circular, somente *pixels* que estão a um raio r de distância, gerando uma máscara de segmento circular proposto por Bresenham's que acelera o teste para identificação do POI, e é conhecida como *Accelerated Segment Test* (AST).

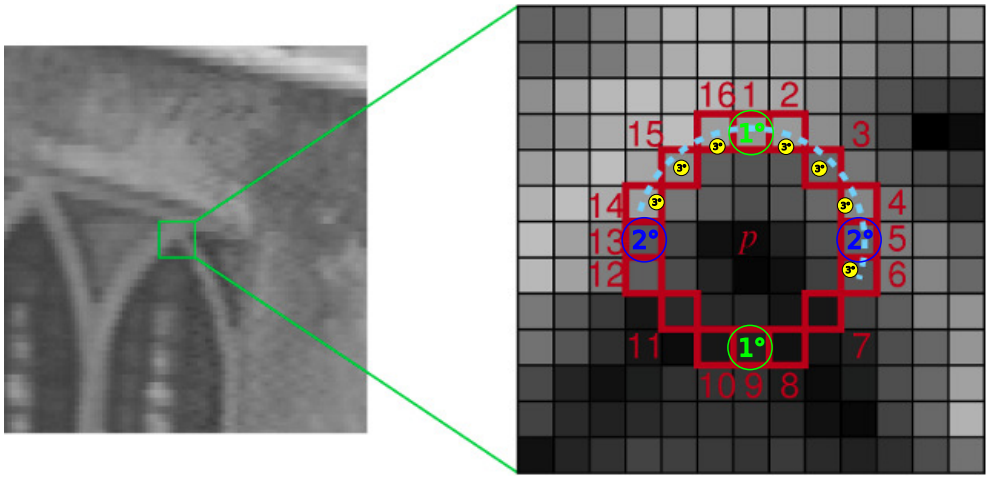

Figura 3.7: Descrição do ponto - AST. FONTE: Modificado de [Rosten and Drummond, 2006, p. 4]. O *pixel* central P é comparado com todos os 16 *pixels* do segmento pertencente a um raio r. Se 9 *pixels* contíguos (pontilhado azul claro) não estiverem entre o intervalo $P + t$ e $P - t$, onde P é a intensidade do *pixel* central e t um *threshold*, então P é candidato a um POI. Para acelerar a busca primeiro são testados os *pixels* 1 e 9 (círculos verdes), se suas intensidades não estiverem entre o intervalo $P + t$ e $P - t$, os *pixels* 5 e 13 (círculos azuis) são verificados também. Se três dos quatro passarem no teste, os demais são verificados para identificar um segmento de 9 *pixels* que passe pelo critério.

O método AST total, apresentado na Figura 3.7, compara o *pixel* central com todos os 16 *pixels* do segmento. Para acelerar a busca por POI o critério de identificação deste é relaxado da forma a seguir. Primeiro são testados os *pixels* 1 e 9, se suas intensidades estiverem entre o intervalo $P + t$ e $P - t$, onde P é a intensidade do *pixel* central e t um *threshold*, então P não é candidato a um POI. Se P passar neste primeiro teste, então os *pixels* 5 e 13 são verificados também. Se três dos quatro passarem no teste, os demais são verificados para identificar um segmento de 9 *pixels* que passe pelo critério. De forma geral, um POI é determinado dado um conjunto de 9 *pixels* conectados pertencentes ao segmento de 16 *pixels*, com intensidades maiores ou menores que um *threshold*, determinado pela intensidade do *pixel* central. O *threshold* determina a quantidade e qualidade (magnitude do gradiente) dos POIs encontrados [Rosten and Drummond, 2006].

Ainda Rosten and Drummond identificam quatro problemas que ocorrem com este método, sendo:

1. Para um conjunto contíguo menor que 12 *pixels*, muitos POIs são detectados.

2. A escolha da ordem dos *pixels* para a primeira comparação assume implicitamente que a distribuição da aparência do POI é conhecida, influenciando diretamente na rapidez do algoritmo.

3. Os critérios de teste que agilizam na validação de um possível POI são descartados.

4. Muitos POIs são detectados adjacentes uns dos outros.

O método FAST surge então pela resolução destes problemas. Os primeiros três problemas Rosten and Drummond [2006] resolvem com uma técnica de aprendizagem de máquina utilizando o algoritmo árvore de decisão ID3 [Quinlan, 1986]. Esta árvore é ternária, pois cada um dos 16 *pixels* a serem comparados pode assumir três estados

$$S_{p\to x} = \begin{cases} e, & I_{p\to x} \leq I_p - t & \text{(escuro)} \\ s, & I_p - t < I_{p\to x} < I_p + t & \text{(similar)} \\ c, & I_p + t \leq I_{p\to x} & \text{(claro)} \end{cases} \tag{3.10}$$

onde $S_{p\to x}$ é o estado, $I_{p\to x}$ é a intensidade do *pixel* x, e t é um *threshold*.

Então, com posse de todos os vetores que são POIs ou não, a árvore é treinada utilizando o princípio da minimização da entropia, e retorna se o *pixel* central é um POI ou não, consultando o vetor de 16 *pixels* o menor número de vezes possível. Esta árvore ternária é treinada para cada novo ambiente. O quarto problema é resolvido com uma supressão de não máximos.

A comparação realizada para suprimir não máximos ocorre entre a soma da diferença absoluta dos *pixels* contíguos e o *pixel* central, descartando o POI com menor valor.

3.2.6 AGAST

O detector de canto *Adaptive and Generic Accelerated Segment Test* (AGAST) proposto em Mair et al. [2010], diferente do FAST, implementa um árvore binária genérica que não necessita ser retreinada para cada novo ambiente. Uma combinação de duas árvores treinadas e especializadas, uma para áreas homogêneas (valores baixos na comparação dos *pixels*) da imagem e a outra para áreas heterogêneas (valores altos na comparação dos *pixels*), garante a variabilidade dos POIs na cena.

Para cada folha da árvore onde o critério de canto é testado, um salto para árvore especializada é realizado com base na configuração de *pixels* desta folha. Estes saltos entre árvores de decisão especializadas não inclui custos adicionais porque a avaliação do nó da folha foi feito na fase de treinamento. Assim, o AST é adaptado a cada cena dinamicamente, aumentando seu desempenho e não precisando de qualquer aprendizado. A Figura 3.8 demonstra a estrutura de duas árvores especializadas. O AGAST salta entre as árvores assim que os *pixels* vizinhos se alteram. Quanto mais claro o cinza de uma folha, mais homogêneos são os *pixels* da configuração.

O problema de POIs adjacentes, semelhante à metodologia do FAST, é resolvido com supressão de não máximos.

3.3 Descritores

Foi apresentado na seção anterior como os pontos de interesse são detectados porém, para que os mesmos sejam comparados, estes devem possuir alguma forma de descrição, onde a

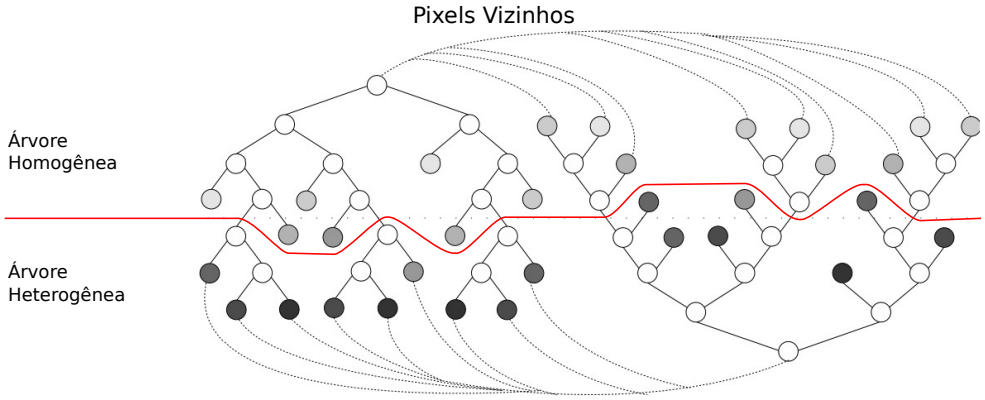

Figura 3.8: Árvore de decisão - AGAST. FONTE: Modificado de [Mair et al., 2010, p. 7]. A árvore homogênea demonstra comparações do *pixel* central com *pixels* vizinhos que tem intensidades próximas; a árvore heterogênea representa *pixels* visinhos com maior contraste.

abordagem usual é utilizar um vetor de características, tornando-o então distinto. Nesta seção serão apresentados os descritores utilizados em conjunto com os métodos de detecção comentados na Sessão 3.2.

3.3.1 SIFT

SIFT encontra pontos de interesse em várias regiões e escalas da imagem e atribui uma orientação a este. Um vetor que represente cada POI é construído para futura correspondência entre POIs de diferentes imagens, chamado de descritor de pontos de interesse.

Para descrever o ponto de interesse, SIFT calcula primeiramente a magnitude do gradiente e orientação dos *pixels* vizinhos ao local do ponto. Para tal, utiliza a escala do ponto para selecionar o grau de borramento Gaussiano da janela, e normaliza os valores em relação à orientação obtida na fase de detecção (Figura 3.9 A e B). Estas medições são acumuladas em um histograma (Figura 3.9 D) que representa sub-regiões de 4×4 *pixels* (Figura 3.9 B), gerando um descritor de 2×2 (Figura 3.9 C), onde a magnitude das flechas correspondem a soma dos gradientes de cada *pixel* da matriz 4×4 numa direção específica. Como cada representação dos gradientes é discretizada em 8 direções, obtemos um descritor de $2 \times 2 \times 8$ gerando um vetor do ponto de interesse com 32 características. Finalizando, o vetor é normalizado para tornar a descrição invariante a iluminação [Lowe, 2004]. Em sua pesquisa Lowe [2004] utiliza um descritor de $4 \times 4 \times 8$ gerando um vetor com 128 características que descrevem o POI.

3.3.2 SURF

O descritor do SURF representa a distribuição de intensidades do gradiente de acordo com os *pixels* vizinhos do ponto de interesse, semelhante ao SIFT, porém utiliza a transformada de Haar de primeira ordem nas direções x e y (em vez de gradiente) explorando a imagem integral. Seu descritor é um vetor que contém 64 características do ponto, reduzindo o custo computacional durante a correspondência de POIs em imagens diferentes, e aumentando a robustez da descrição [Bay et al., 2008, 2006].

Dois passos são necessários para construção do descritor. Primeiro é encontrar uma orientação baseada na informação de uma região circular ao redor do ponto, que seja reproduzível.

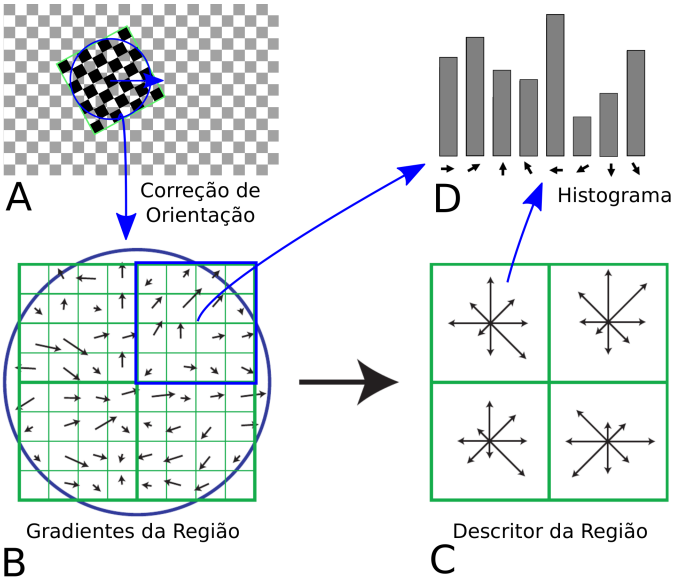

Figura 3.9: Descrição do ponto - SIFT. FONTE: Alterado de [Szeliski, 2010, p. 224]. Em (A), SIFT descreve o POI utilizando a escala e a orientação obtidos na fase de detecção, para normalizar os valores do gradiente de cada *pixel* na vizinhança, de forma a canonizá-los à orientação; Em (B), os valores são acumulados em histogramas (D), conforme oito direções, por sub-região de 4×4 *pixels* gerando o descritor (C); e em (C), o descritor de $2 \times 2 \times 8$ gera um vetor com 32 características representando então o POI.

Segundo, construir uma região quadrada alinhada com a orientação do ponto para extrair as descrições do ponto de interesse.

Para definir a orientação, uma transformada de Haar nas direções x e y (Figura 3.10 A) é aplicada para encontrar intensidades em todos os *pixels* vizinhos ao ponto, a uma distância de raio $6s$, onde s é a escala que o ponto foi encontrado (Figura 3.10 B). As intensidades de cada *pixel* são proporcionais à escala, e de igual modo ocorre com o filtro de Haar onde seu tamanho é $4s$. Para calcular o filtro de Haar em qualquer das direções x e y, somente seis operações são necessárias em qualquer escala devido a imagem integral. Um peso Gaussiano é incrementado na região circular ao ponto de interesse com distribuição $\sigma = 2s$, as respostas obtidas são representadas por pontos no espaço em que as intensidades horizontais aparecem ao longo da abscissa (eixo x), e verticais ao longo das ordenadas (eixo y) (Figura 3.10 C). A orientação dominante é calculada somando todos os pontos que encontram-se em uma janela de tamanho $\frac{\pi}{3}$ (Figura 3.10 D) [Bay et al., 2008, 2006].

Para extrair a descrição do ponto de interesse (segunda fase), uma região quadrada é construída ao redor do POI e orientada conforme o mesmo, onde o tamanho da janela é de $20s$ (Figura 3.11 A). A janela é então dividida regularmente em sub-regiões quadradas de 4×4 (Figura 3.11 B). Para cada sub-região a transformada de Haar é calculada 25 vezes gerando um exemplo quadrado de 5×5 com as respectivas orientações (Figura 3.11 C). Para cada exemplo do quadrado de 5×5, são coletadas as orientações relacionadas as transformadas em x por nome dx e em y de nome dy (Figura 3.11 D). Para incrementar a robustez relativa a deformações geométricas e a erros de localização, são atribuídos a dx e dy pesos Gaussianos com desvio $\sigma = 3.3s$ centrado no ponto de interesse. Uma somatória de dx e dy é feita em cada sub-região para ser armazenada então no vetor de características do POI porém, o sinal da mudança de intensidade também deve ser armazenado, de forma que estes são o somatório dos valores absolutos, $|dx|$ e $|dy|$ (Figura

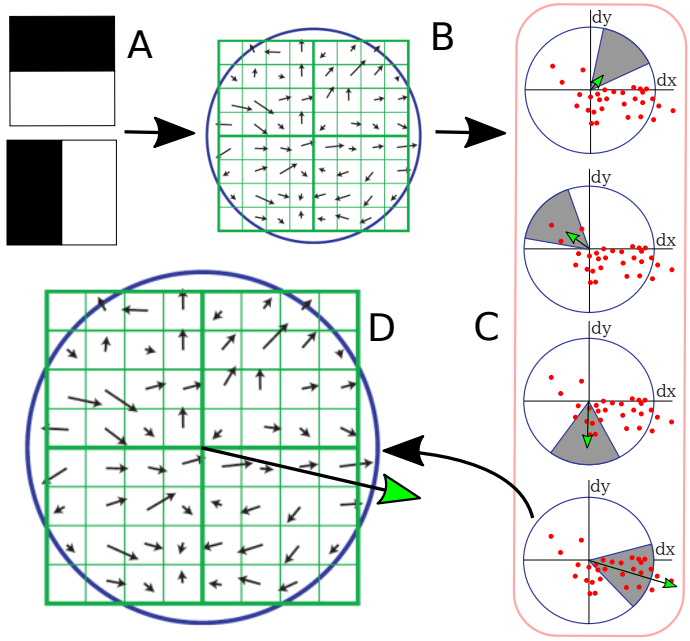

Figura 3.10: Orientação do Ponto - SURF. FONTE: Modificado de [Szeliski, 2010, p. 224]. Em (A) é apresentado filtros de Haar para derivar em direções x e y; em (B), o resultado das derivadas formando gradientes em cada *pixel* é mostrado; em (C), as respostas obtidas de cada *pixel* são representadas por pontos no espaço, em que as intensidades horizontais aparecem ao longo da abscissa (eixo x), e verticais ao longo das ordenadas (eixo y), e a orientação dominante é calculada somando todos os pontos que encontram-se em uma janela de tamanho $\frac{\pi}{3}$; em (D) é demonstrada a orientação final do ponto.

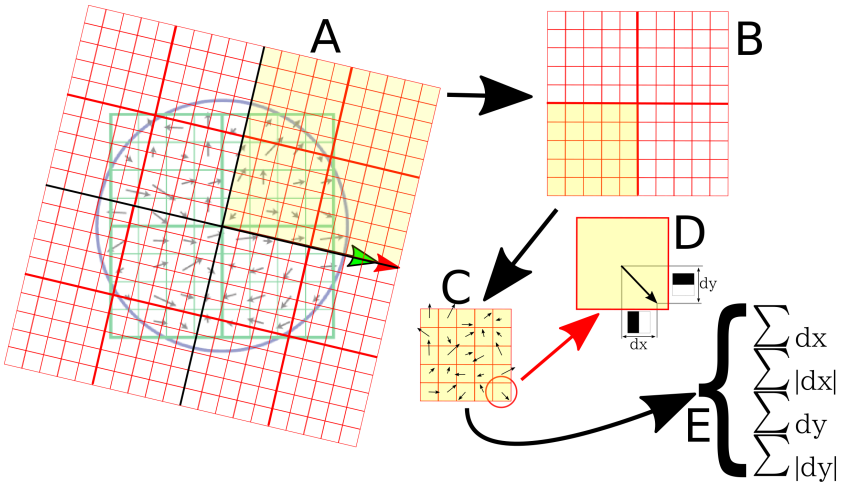

Figura 3.11: Descrição do ponto - SURF. FONTE: Modificado de [Szeliski, 2010, p. 224]. Em (A), uma região quadrada é construída e orientada ao redor do POI; em (B), a janela de (A) é dividida em quatro sub-regiões quadradas; em (C), a transformada de Haar em x e y é calculada (D) identificando intensidades do gradiente de cada *pixel*; e em (E), uma somatória das derivadas em x e em y são feitas em cada sub-região e armazenada no vetor de características do POI.

3.11 E). Desta forma, para cada sub-região existe um vetor que representa suas intensidades

sendo $\sum dx$, $\sum dy$, $\sum |dx|$, $\sum |dy|$, e concatenando todas as 4×4 sub-regiões obtém o vetor do ponto de interesse com 64 características.

3.3.3 FREAK

Fast Retina Keypoint (FREAK) é o método desenvolvido por Alexandre Alahi et al. [2012] e propõem uma descrição binária do ponto de interesse baseado no sistema de visão humana. Segundo Alahi et al., FREAK é mais veloz computacionalmente, têm baixo consumo de memória e é mais robusto quando comparado ao SIFT, SURF ou *Binary Robust Invariant Scalable Keypoints* (BRISK) (apresentado em Leutenegger et al. [2011]).

Conforme Alahi et al., estudos feitos no campo da neurociência sobre o sistema visual humano demonstram que, de certa forma, a retina extrai os detalhes da imagem utilizando uma diferença Gaussiana de vários tamanhos e codificadas com pesos diferentes (Figura 3.12 A). Desta forma FREAK tenta imitar a estratégia descrita, representada na Figura 3.12 B.

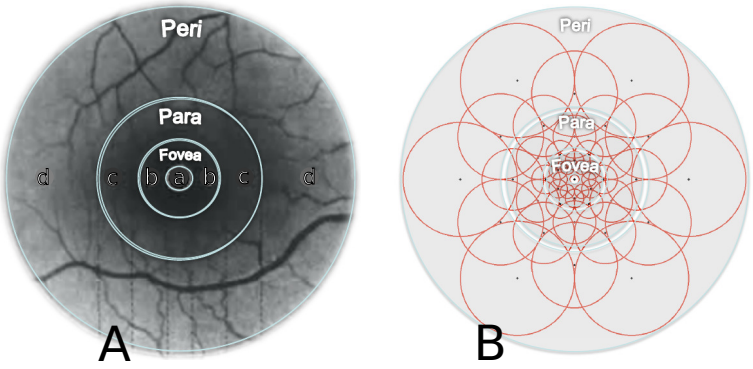

Figura 3.12: Modelo de descritor - FREAK. FONTE: Modificado de [Alahi et al., 2012, p. 3]. Em (A), regiões da retina do olho humano que identificam detalhes de imagens de formas diferentes, onde a região perifoveal tem um borramento maior, e a área foveal tem um borramento menor; e em (B), as regiões de (A) são representadas por círculos concêntricos de tamanhos diferentes, conforme distribuição Gaussiana.

Círculos concêntricos são espalhados na região da retina, cada qual com tamanho proporcional a uma distribuição Gaussiana e localização foveal, parafoveal e perifoveal. Para a construção do descritor binário F, um corte é feito na diferença entre pares de campos receptivos (círculos) com suas respectivas áreas Gaussianas, onde F é um vetor binário formado por uma sequência de *bits* representantes da diferença Gaussiana [Alahi et al., 2012]. A Equação 3.11 descreve F.

$$F = \sum_{0 \leq a < N} 2^a T(P_A) \tag{3.11}$$

onde P_a é o par de campos receptivos, N o tamanho do descritor, e

$$T(P_a) = \begin{cases} 1, & \text{se } (I(P_a^{r_1}) - I(P_a^{r_2})) > 0 \\ 0, & \text{se não} \end{cases} \tag{3.12}$$

onde $I(P_a^{r_1})$ é a intensidade do borramento Gaussiano do primeiro campo receptivo do par P_a.

Centenas de pares podem ser formados com os campos receptivos, porém nem todos ajudam a descrever uma região da imagem com eficiência. FREAK usa um algoritmo similar ao

Oriented FAST and Rotated BRIEF (ORB) [Rublee et al., 2011] que aprende os melhores pares dado um treinamento da imagem. O treinamento demonstra que 512 pares são relevantes para descrever de forma eficiente a região, e apresentam um esquema simétrico capturado conforme a orientação do padrão ao longo do gradiente global (Figura 3.12 A). O primeiro padrão (conjunto de pares) envolve o campo perifoveal (Figura 3.13 etapa 1), ao passo que o último representa a área foveal (Figura 3.13 etapa 4) [Alahi et al., 2012].

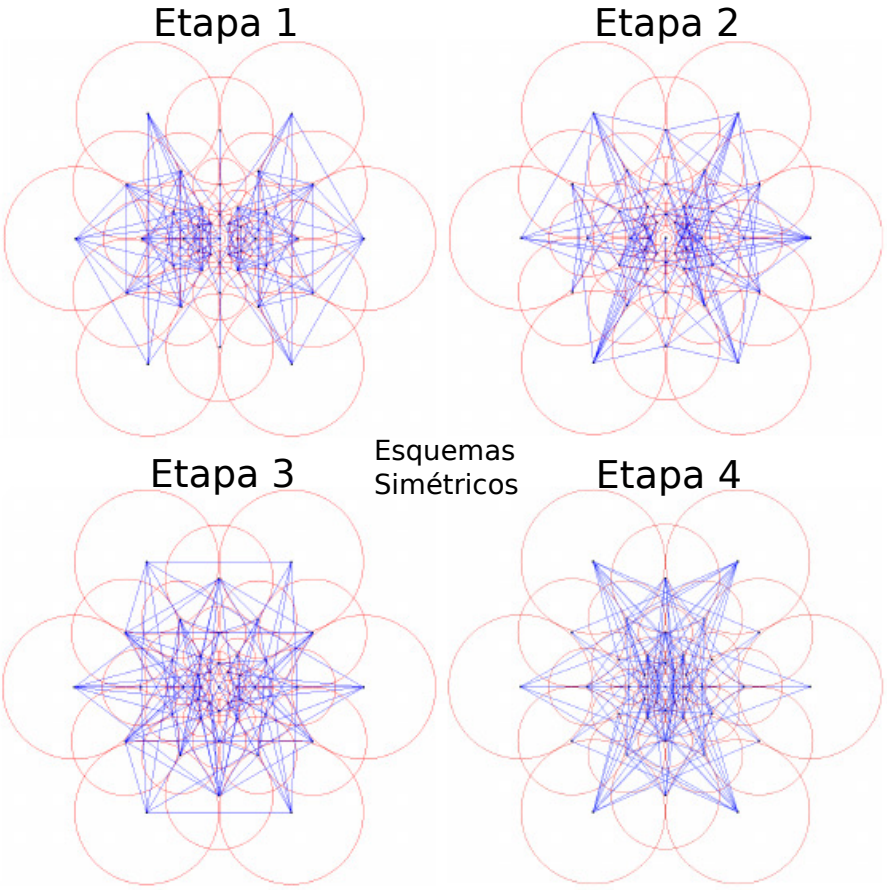

Figura 3.13: Descrição do ponto - FREAK. FONTE: Modificado de [Alahi et al., 2012, p. 5]. A Figura representa simetrias de pares formados por campos receptivos que representam áreas da retina. A simetria da etapa 1 representa mais a área perifoveal enquanto que a etapa 4 representa *links* de pares da área foveal.

Segundo Alahi et al. [2012], o desempenho de seu método aumenta na fase de correspondência dos POIs onde uma cascata de comparação é feita. Primeiro, os 16 *bytes* iniciais dos 64 que representam o vetor binário (Figura 3.13 etapa 1) são comparados, dada a distância de *Hamming* e um valor de corte. Se a distância satisfaz o valor de corte, a comparação continua para o segundo nível (Figura 3.13 etapa 2) sendo os próximos 16 *bytes* e seguindo até os últimos 16 *bytes* (Figura 3.13 etapa 4). Logo na primeira comparação 90% dos candidatos são descartados, decrementando assim o número de comparações nas etapas seguintes.

Para definir a orientação do ponto de interesse por meio do descritor, FREAK soma os gradientes locais estimados conforme os pares selecionados. FREAK usa longos pares baseados

em uma simetria (Figura 3.14) extraída durante o processo de aprendizado, para calcular a orientação global. Dado G o conjunto de pares usados para computar o gradiente local:

$$G = \frac{1}{M}(I(P_o^{r_1}) - I(P_o^{r_2}))\frac{P_o^{r_1} - P_o^{r_2}}{||P_o^{r_1} - P_o^{r_2}||} \tag{3.13}$$

onde M é o numero de pares em G, e $P_o^{r_i}$ é o vetor 2D das coordenadas espaciais do centro do campo receptivo.

FREAK seleciona para orientação 45 pares conforme simetria visualizada na Figura 3.14.

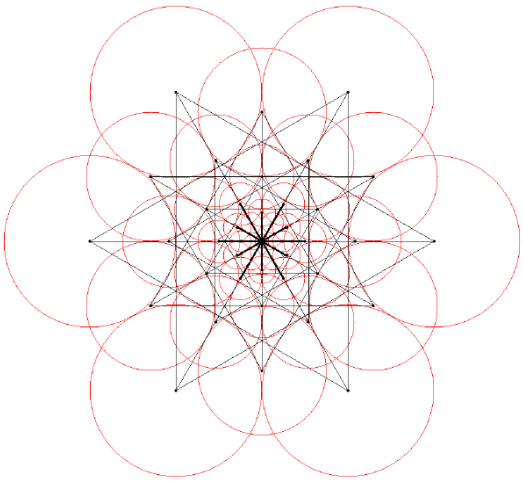

Figura 3.14: Orientação do ponto - FREAK. FONTE: Extraído de [Alahi et al., 2012, p. 5]. Simetria de pares de campos receptivos utilizados para calcular a orientação do ponto de interesse.

3.4 Resumo

Este capítulo apresenta as formas de representar características, assim como as propriedades inerentes às qualidades destas. Em seguida, aborda com detalhes métodos de detecção de pontos de interesse. Por fim, este capítulo demonstra os métodos utilizados para descrever POIs já detectados. O Capítulo 4 apresenta a revisão de trabalhos relacionados à SLAM, VSLAM, avaliação de detectores de características de forma independente e a avaliação do detector em função do VSLAM.

Capítulo 4

Revisão da Literatura

Este capítulo apresenta a revisão de trabalhos relacionadas ao problema de VSLAM e avaliações de detectores de características. A Seção 4.1 apresenta trabalhos que propõem métodos de resolução do SLAM. A Seção 4.2 aborda trabalhos que utilizam visão integrada ao SLAM. A Seção 4.3 detalha trabalhos que avaliam detectores de pontos de interesse. Por fim, a Seção 4.4 descreve os autores que avaliam detectores de características em função do VSLAM, apresentando em seguida na Tabela 4.1, sua comparação em relação aos trabalhos apresentados da Seção 4.3, e a proposta da dissertação.

4.1 SLAM

Sistemas que utilizam filtros probabilísticos para resolver o problema de mapeamento e localização simultâneos (SLAM) foram propostas em Smith and Cheeseman [1986] que usam o Filtro de Kalman Estendido formando o EKFSLAM, e em Montemerlo et al. [2002] que utiliza Filtro de Partículas formando o FASTSLAM. Lu and Milios [1997] propôs que a solução do SLAM fosse feita com grafos. Todos os métodos citados acima trabalham com mapeamento 2D, estendidos facilmente para 3D.

Criação de mapas tridimensionais depende intimamente do tipo de sensores utilizados pelo robô, como *encoders*, sonares, *scaners laser* e câmeras. Atualmente, pesquisas têm dado um interesse maior em Visão Computacional, devido à imagem fornecer um grande número de informações do ambiente e pelo baixo custo de câmeras. O processo de estimar o deslocamento e a velocidade do robô por meio de imagens é conhecido como Odometria Visual (VO) e está relacionada diretamente com SLAM, tornando-se pela fusão de ambos, Visual SLAM ou VSLAM [Yousif et al., 2015].

4.2 Visual SLAM

Duas abordagens principais em VSLAM são os métodos baseados em características esparsas, utilizados em Klüssendorff et al. [2013], Mur-Artal et al. [2015], e os densos Engel et al. [2014], Kerl et al. [2013], Silveira et al. [2008]. Trabalhos como de Engel et al. [2013], Forster et al. [2014], Mur-Artal and Tardós [2015] utilizam as duas abordagens em conjunto.

Tendendo ao foco da dissertação, que avalia um método detector de características, destacam-se então trabalhos de VSLAM que utilizam detectores como o de Lemaire and Lacroix [2007], que faz proveito da visão panorâmica, usa [Harris and Stephens, 1988] para detectar POIs, e resolve o SLAM com base no movimento angular do robô com EKF [Deans, 2005]. Já

Paz et al. [2008a] utilizam um EKF otimizado [Paz et al., 2008b], detectando pontos por meio da técnica de rastreamento de Shi and Tomasi [1994] e usando abordagem estéreo para estimar a profundidade. Similar a Paz et al. [2008a], Davison and Murray [2002] utilizam a visão estéreo de forma ativa (não é fixa no robô). Sim et al. [2005] usam SIFT [Lowe, 1999] como detector de pontos de interesse e resolvem o SLAM com Filtro de Partículas [Doucet et al., 2000]. Ainda Davison [2003] publicou a mesma abordagem de Davison and Murray [2002] utilizando visão monocular em tempo real, e em Davison et al. [2007] atribuíram o nome MONOSLAM ao seu método e apresentaram novos resultados com experimentos em humanoides.

Uma abordagem diferente das apresentadas acima é utilizada pelo algoritmo monocular *Parallel Tracking and Mapping* (PTAM) de Klein and Murray [2007], resolvem o problema de SLAM por meio de *Bundle Adjustment* (BA) [Triggs et al., 2000], separando em *threads* distintas o processo de rastreamento de pontos e o de mapeamento. Para detectar POIs utilizam o método FAST de Nain et al. [2008], e para calcular a matriz essencial e a triangulação utiliza RANSAC de Nister [2004]. Similar à ideia de PTAM, ORBSLAM proposto em Mur-Artal et al. [2015], faz uso de três *threads*, sendo separadas em rastreamento do ponto, mapeamento local e mapeamento global, onde todas utilizam *Bundle Adjustment*. Ainda, utiliza ORB de Rublee et al. [2011] para detectar pontos de interesse e otimização de grafos de Grisetti et al. [2011] para detectar fechamento de laço (*loop closure*). Ligeiramente diferente de PTAM e ORBSLAM, RGBDSLAM apresentado em Endres et al. [2012] utilizam sensores tipo *Kinect®* produzidos pela *Microsoft®* que, além de fornecer a imagem do ambiente, disponibiliza um mapa de profundidade rastreado por infravermelho. Dada correspondência de POIs detectados em duas imagens e suas correspondências em pontos 3D capturados pelo mapa de profundidade, por RANSAC de cinco pontos Endres et al. recuperam a matriz essencial entre os *frames*. Para formar um mapa global é utilizada também uma otimização de grafos de Grisetti et al..

4.3 Avaliação de Detectores

Um fator crítico para gerar boas estimativas da matriz essencial é detectar POIs robustos, aqueles que são invariantes à transformações fotométricas e geométricas na imagem. Vários trabalhos avaliam a detecção destes POIs. Schmid et al. [2000] analisam a repetibilidade dos POIs considerando alterações na rotação, escala, iluminação, ponto de vista, ruído e discriminação do POI (quão distinto cada um é dos demais na mesma imagem). Os métodos de descrição de característica não estão citados por estarem fora do escopo deste trabalho. No artigo de Dwarakanath et al. [2012], o detector é medido em termos de acurácia, detectabilidade e tempo de execução, onde pares de imagens são obtidas por câmeras estéreo. Os métodos analisados são SIFT, SURF e ORB, e resultados dados em relação aos efeitos de distorção, borramento e ruído nas imagens. Miksik and Mikolajczyk [2012] avaliam detectores binários como BRISK, *Binary Robust Independent Elementary Features* (BRIEF), ORB entre outros, em relação a repetibilidade, precisão, sensibilidade e tempo de execução. Um trabalho mais completo em termos de métricas e quantidade de detectores utilizados é o descrito em Barandiaran et al. [2012], que analisam os métodos Harris, GFTT, SIFT, SURF, FAST, *Maximally Stable Extremal Regions* (MSER), STAR[1] e ORB quanto a repetibilidade, distintividade, quantidade, acurácia e eficiência, onde parâmetros de avaliação modificam a rotação, escala e homografia. No trabalho de Oliveira et al. [2015], um melhoramento do método GFTT é proposto e comparado quanto a mudanças na rotação, escala, pontos de vista, borramento, compressão e iluminação, com

[1] Versão modificada do extrator de características CenSurE Agrawal et al. [2008]

os detectores MSER, SIFT, SURF, ORB, GFTT, STAR, BRISK, KAZE[2] [Alcantarilla et al., 2012] e *Accelerated* KAZE (AKAZE) [Alcantarilla et al., 2013], aproxima-se ou ultrapassa o desempenho destes quanto a repetibilidade, distintividade, robustez e tempo de processamento.

4.4 Avaliação de Detectores em função do VSLAM

O trabalho de Endres et al. [2012] não só implementa um método de SLAM como avalia vários detectores de pontos de interesse como SIFT [Lowe, 1999], SURF [Bay et al., 2006] e ORB [Rublee et al., 2011] quanto ao RMSE dada uma trajetória estimada e o tempo de processamento em relação a detecção/descrição, correspondência dos POIs e aferição do movimento, e a otimização do grafo. Em Klippenstein and Zhang [2007], a curva gerada pela precisão/sensibilidade é utilizada como métrica para os métodos de detecção SIFT, Harris e Kanade, Lucas e Tomasi (KLT) [Lucas and Kanade, 1981]. Estes detectores ainda são avaliados com diferentes técnicas de correspondência como *Nearest Neighbor with Distance Ratio* (NNDR), *Normalized Cross Correlation* (NNC) e alinhamento KLT. Um estudo feito em 2009 por Klippenstein and Zhang utiliza SLAM com EKF e avalia os mesmos algoritmos de detecção de características testando a consistência com *Normalized Estimation Error Squared* (NEES) e a grandeza da incerteza nos pontos dado a trajetória. Gil et al. [2010] avaliam a repetibilidade dos detectores (Harris, Harris-Laplace[3], SUSAN, SIFT, SURF, MSER e KDIR[4]) assim como a invariância e a distintividade dos descritores (SIFT, *Gradient Location and Orientation Histogram* (GLOH), SURF, *Gray Level Patch*, Histograma de Orientação e Momentos de Zernike) em diferentes condições de escala, pontos de vista e iluminação. Em seu trabalho Hartmann et al. [2013] analisam a acurácia e o tempo de processamento do detector dado um cenário realístico (*benchmark datasets*). Utilizam RGBDSLAM baseado em grafos e mensuram o *drift* (escorregamento) da odometria visual dado o aumento da velocidade e o erro da posição do robô em diferentes ambientes.

4.5 Conclusões

Conclusões da revisão apontam que, ou detectores são avaliados especificamente com *benchmarks*, para qualificar sua robustez sob condições controladas (exemplo em Oliveira et al. [2015]), ou são avaliados em função do desempenho de outros sistemas que dependem destes, onde as sequências de imagens podem ter condições controladas nos testes (exemplo em Gil et al. [2010]) ou não (exemplo em Hartmann et al. [2013]). Os detectores mais testados nas pesquisas geralmente são aqueles mais velozes em tempo de processamento, os que detectam muitos POIs e os que identificam POIs mais robustos. SIFT e SURF são clássicos na literatura por serem robustos e vastamente testados, os demais, ou são relativamente novos como o IGFTT, ou são algoritmos velozes que detectam muitos POIs como ORB, GFTT, FAST e AGAST.

Não foi encontrada uma avaliação geral do desempenho do detector para ambientes desconhecidos ou com pouca heurística. Um robô explorador não conhece e nem controla todo o ambiente em que se encontra, precisando ter sistemas genéricos e dinâmicos. O processo de exploração por visão depende da generalidade do detector de POI e (ou) identificar em tempo real

[2]Extrator de Características cujo nome foi dado em tributo ao pesquisador Iijima, considerado o pai da análise no espaço de escala. KAZE é uma palavra Japonesa que significa Vento. Fonte Alcantarilla et al. [2012]

[3]Versão do extrator de características Harris invariante a transformação de escala e afim [Mikolajczyk and Schmid, 2004]

[4]Extrator de Características cujo nome faz referência a um de seus criadores, Timor Kadir Kadir et al. [2004]

a satisfatoriedade do desempenho deste em determinado tipo de ambiente, podendo ser trocado dinamicamente. Desta forma, seriam interessantes testes em sequências de imagens que mudam somente o ponto de vista da câmera conforme deslocamento geral ou visual do robô, onde os parâmetros de variações são os tipos de ambientes que este se encontra, avaliando o detector de forma geral.

O tempo gasto nos processos que envolvem os POIs, são importantes para determinar sua aptidão em sistemas de tempo real. O teste de repetibilidade e estabilidade do detector é interessante para identificar variações no desempenho deste, conforme ocorrem mudanças no ambiente. E o erro geral do robô nas trajetórias de cada sequência demonstram a acurácia do detector, onde seus resultados podem ser visualizados de forma especifica ou genérica. Uma gama de detectores são testados nas pesquisas relacionadas neste capítulo, onde a maioria estão implementadas na biblioteca OPENCV e disponibilizadas livremente para uso, se tornando um ponto de partida.

Em contraste com as abordagens anteriores, apresentamos neste trabalho a avaliação de um novo método de detecção de pontos de interesse IGFTT avaliando a qualidade do mapa produzido por um algoritmo de Visual SLAM em função deste e outros detectores, em lugar de realizar uma avaliação genérica não orientado a um propósito específico. Conforme Endres et al. [2012] o método é avaliado quanto ao erro da pose relativa e da trajetória absoluta, sua acurácia quanto ao RMSE da translação e rotação nas sequências utilizadas e o tempo de processamento em relação a detecção, descrição, correspondência e número de pontos encontrados. Os detectores também são avaliados quanto sua estabilidade e repetibilidade ao longo das sequências de imagens da cena, apresentada por Gil et al. [2010]. As sequências de imagens utilizadas nos experimentos são produtos dos trabalhos de Geiger et al. [2012], Sturm et al. [2012], que as disponibilizam como *benchmark* para avaliação de sistemas de Odometria Visual e SLAM. Os parâmetros de variação considerados serão os variados tipos de ambientes disponibilizados no *benchmark* e as mudanças do ponto de vista da câmera conforme movimento.

A tabela 4.1 apresenta um resumo dos trabalhos de avaliação descritos. Em vermelho são referências aos trabalhos pilares da dissertação, tanto para tipos de avaliações [Hartmann et al., 2013, Gil et al., 2010] quanto para método de detecção avaliado [Oliveira et al., 2015]. Referência à forma de avaliações utilizadas e novo tipo de avaliação, como também detector avaliado e proposta de melhoria deste, estão marcados em cor azul.

4.6 Resumo

Este capítulo apresentou alguns trabalhos referentes a resolução do problema de SLAM e abordagens baseadas em Visão (VSLAM). Em seguida, foram descritos trabalhos que avaliam detectores de pontos de interesse de forma independente. O capítulo também apresentou autores que avaliam métodos de detecção de POIs em função do mapa produzido pelo VSLAM. Por fim, este capítulo apresentou conclusões das referências relacionadas, descreveu a forma de avaliação da dissertação e organizou a tabela 4.1 comparando as formas de avaliações dos detectores. O próximo capítulo apresenta uma visão geral sobre o problema e a contribuição deste trabalho.

[5]Schmid et al. [2000] analisa os métodos Harris (marcado na tabela 4.1), Foerster, Cottier, Heitger e Horaud. Não foram citados por não haver testes destes nos trabalhos relacionados.

			Autores Avaliam Detectores com:										
			Detectores					Detectores e VSLAM					
			Schmid et al. [2000][5]	Dwarakanath et al. [2012]	Miksik and Mikolajczyk [2012]	Barandiaran et al. [2012]	Oliveira et al. [2015]	Endres et al. [2012]	Klippenstein and Zhang [2007]	Klippenstein and Zhang [2009]	Gil et al. [2010]	Hartmann et al. [2013]	Dissertação
Métodos de Detecção e Descrição	SIFT		–	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	SURF		–	✓	✓	✓	✓	✓	–	–	✓	✓	✓
	Harris		✓	–	–	✓	–	–	✓	✓	✓	–	–
	Harris-Laplace		–	–	–	–	–	–	–	–	✓	–	–
	BRISK		–	–	✓	–	✓	–	–	–	–	✓	–
	BRIEF		–	–	✓	–	–	–	–	–	–	✓	–
	FREAK		–	–	–	–	–	–	–	–	–	✓	✓
	KLT		–	–	–	–	–	–	✓	✓	–	–	–
	ORB		–	✓	✓	✓	✓	✓	–	–	–	✓	✓
	GFTT		–	–	–	✓	✓	–	–	–	–	–	✓
	FAST		–	–	–	✓	✓	–	–	–	–	–	✓
	AGAST		–	–	–	–	–	–	–	–	–	–	✓
	MSER		–	–	–	✓	✓	–	–	–	✓	–	✓
	SUSAN		–	–	–	–	–	–	–	–	✓	–	–
	STAR		–	–	–	✓	✓	–	–	–	–	–	✓
	KDIR		–	–	–	–	–	–	–	–	✓	–	–
	KAZE		–	–	–	–	✓	–	–	–	–	–	✓
	AKAZE		–	–	–	–	✓	–	–	–	–	–	✓
	IGFTT		–	–	–	–	✓	–	–	–	–	–	✓
	IGFTT2		–	–	–	–	–	–	–	–	–	–	✓
Análise de Desempenho	Sensibilidade		–	–	✓	–	✓	–	✓	–	–	–	–
	Precisão		–	–	✓	–	✓	–	✓	–	–	–	–
	Tempo de Execução	Detecção	–	✓	✓	✓	✓	✓	–	–	–	✓	✓
		Descrição	–	✓	✓	✓	✓	✓	–	–	–	✓	✓
		Correspondência	–	✓	✓	✓	✓	✓	–	–	–	–	✓
		Est. Movimento	–	–	–	–	–	✓	–	–	–	–	–
	Acurácia		–	✓	–	✓	–	–	–	–	–	✓	✓
	Robustez		–	–	–	–	✓	–	–	–	–	–	–
	Quantidade		–	–	–	✓	–	✓	–	–	–	–	✓
	Detectabilidade		–	✓	–	–	–	–	–	–	–	–	–
	Distinção		✓	–	–	–	–	–	–	–	–	–	–
	Repetibilidade		✓	–	✓	✓	✓	–	–	–	✓	–	✓
	Estabilidade		–	–	–	–	–	–	–	–	✓	–	✓
	Erro Trajetória		–	–	–	–	–	✓	–	–	–	✓	✓
	Erro Odometria		–	–	–	–	–	✓	–	✓	–	✓	✓
	Organização Grafo de Cena		–	–	–	–	–	–	–	–	–	✓	✓
	Desempenho Geral		–	–	–	–	–	–	–	–	–	–	✓
Transformação	Rotação		✓	–	✓	✓	✓	✓	–	–	–	–	–
	Escala		✓	–	✓	✓	✓	✓	–	–	✓	–	–
	Ponto de Vista		✓	–	–	–	✓	–	✓	✓	✓	–	✓
	Iluminação		✓	–	–	–	✓	–	–	–	✓	–	–
	Homografia		–	–	✓	✓	–	–	–	–	–	–	–
	Distorção		–	✓	–	–	–	–	–	–	–	–	–
	Velocidade		–	–	–	–	–	–	–	–	–	✓	–
	Borramento		✓	✓	–	–	✓	–	–	–	–	–	–
	Ruído		–	✓	–	–	–	–	–	–	–	–	–
	Compressão JPEG		–	–	–	–	✓	–	–	–	–	–	–

Tabela 4.1: Comparativo de abordagens conforme revisão da literatura e propósta do autor.

Capítulo 5

Justificativa do Trabalho

Este capítulo apresenta os fundamentos necessários à compreensão do problema em tela para elaboração e escolha dos métodos de avaliação. Na Seção 5.1 é descrita uma visão geral sobre a relação dos detectores de pontos de interesse em métodos de VSLAM e os problemas relacionados. Por fim, a Seção 5.2 apresenta as contribuições da pesquisa assim como os motivos que levaram à escolha do algoritmo IGFTT ser avaliado em função do VSLAM.

5.1 Visão Geral

Detectores de pontos de interesse (POIs) são partes essenciais em aplicações de visão computacional como VSLAM, reconhecimento de objetos, reconstrução 3D, realidade aumentada, rastreamento de imagens e estrutura de movimento. Detectar um POI que seja invariante a mudanças de iluminação, escala, transformação afim, rotação, e ruído do sensor de aquisição de imagens, com um custo computacional baixo, é extremamente difícil. Detectado um POI, resta o problema de descrevê-lo como um vetor de características que é calculado em função das informações locais do POI detectado. Este vetor pode ser usado para resolver o problema da correspondência de pontos.

Em Visual SLAM, o robô observa pontos de interesse já observados anteriormente (primeiro problema), os corresponde (segundo problema), e localiza-se no ambiente em relação a estes, aumentando a confiança sobre sua posição no ambiente. Se os pontos não forem corretamente encontrados ou associados, o mapa produzido pelo robô fica inconsistente e a confiança quanto à localização diminui.

Muitos algoritmos de detecção e descrição de pontos foram propostos para melhorar o desempenho do VSLAM. Métodos como SIFT e SURF são robustos quanto aos problemas de transformação nas imagens, porém custosos computacionalmente. ORB é ligeiramente mais veloz na detecção, mas com resultados inferiores quanto às transformações nas imagens [Oliveira et al., 2015].

Mesmo com variados métodos de detecção, a escolha da melhor solução é um problema em aberto. Ainda, uma avaliação para definir o melhor detector de pontos de interesse é cercada de variáveis devido ao agregado de técnicas envolvidas para resolver o problema de SLAM utilizando visão. Um modelo dos métodos e suas ligações utilizadas no sistema de Visual SLAM é apresentado na Figura 5.1.

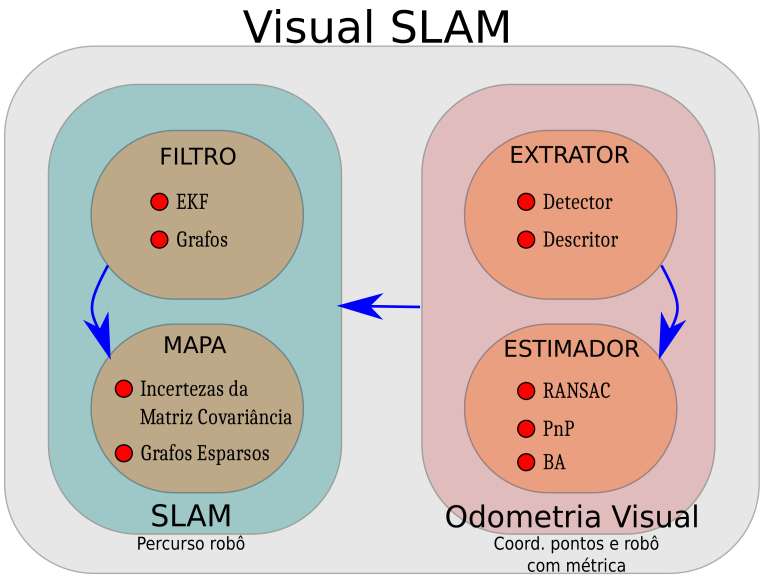

Figura 5.1: Modelo de Visual SLAM. FONTE: Autor. Relação de técnicas envolvidas para solucionar o problema de Visual SLAM.

5.2 Contribuição

Uma análise da qualidade do detector pode ser feita em função do mapa construído pelo VSLAM em relação a um mapa de referência do ambiente [Klippenstein and Zhang, 2009, Endres et al., 2012, Hartmann et al., 2013], ou analisando-o independentemente do mapa, somente na fase de detecção do ponto de interesse, onde a avaliação é dada em relação à robustez do detector (detector e descritor) quando imagens sofrem transformações geométricas e fotométricas [Mozos et al., 2007, Klippenstein and Zhang, 2007, Ballesta et al., 2007, Gil et al., 2010].

Recentemente, um novo algoritmo de detecção de POIs conhecido como IGFTT foi proposto por Oliveira et al. [2015] e apresentou resultados interessantes quanto às mudanças fotométricas, geométricas e tempo de processamento. Oliveira et al. realizou testes quanto a *precision*[1] e *recall*[2] (vide Figura 5.2 superior) de diferentes detectores, aplicando transformações nas imagens[3]. Os autores utilizam o critério de repetibilidade proposto inicialmente em Mikolajczyk et al. [2005] e apresentado na Figura 5.2 inferior, que mede o erro de sobreposição de dois pontos correspondentes, sendo inválidos àqueles com erro menor que 40%. Seus resultados em relação a *precision* e *recall* são apresentados na Figura 5.3.

A relevância dos pontos detectados pelo algoritmo IGFTT, dado pelo escore de precisão (*precision*) (Figura 5.3 gráfico superior) é predominantemente alta em relação aos demais, com uma leve deterioração em imagens com *zoom* (*venice* e *boat*), pelo fato de sua pirâmide de escalas ser simples (sem borramento). Quando analisado o escore dos pontos relevantes encontrados, dado pela sensibilidade (*recall*) (Figura 5.3 gráfico inferior), repete-se o padrão, mantendo-se alta em relação aos outros algoritmos. Os resultados apresentados na pesquisa foram dados pela média de mil execuções do experimento. Nenhum algoritmo analisado conseguiu superar a média da precisão e sensibilidade do IGFTT. Com respeito ao tempo de processamento, BRISK, ORB, GFTT e STAR superaram por pouca diferença [Oliveira et al., 2015].

[1]Parte das instâncias recuperadas que são relevantes - nem tudo que é recuperado têm relevância.
[2]Parte das instâncias pertinentes que são recuperadas - evita instancias não relevantes.
[3]As imagens foram obtidas do site <https://www.cs.unc.edu/~jheinly/binary_descriptors.html> de Jared Heinly.

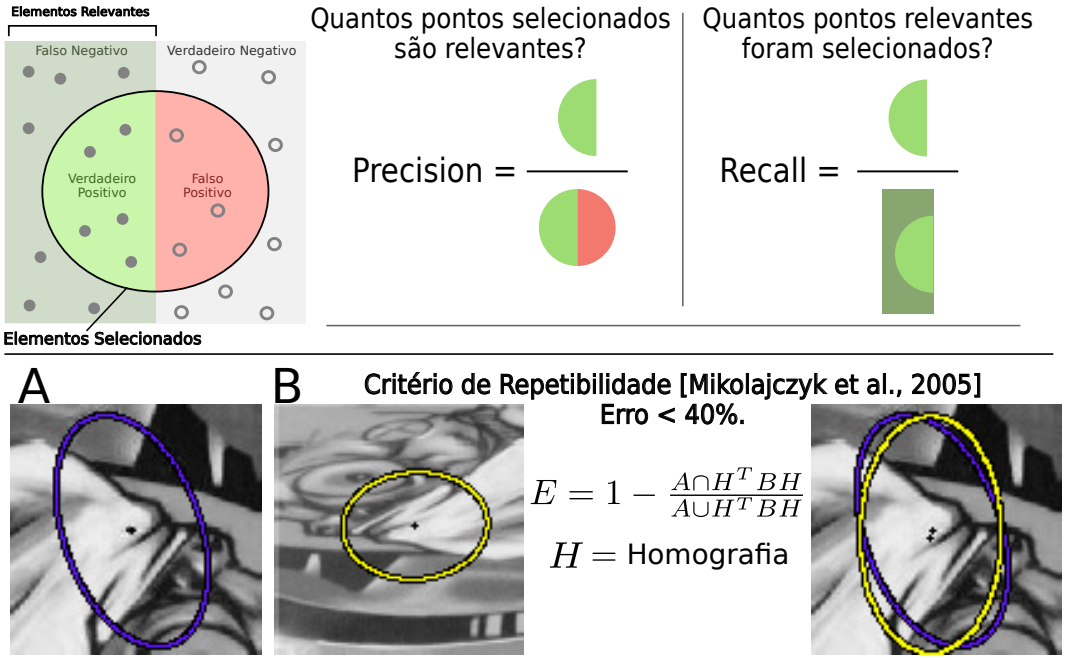

Figura 5.2: Cálculo de *precision*, *recall* e critério de repetibilidade. FONTE: Modificado de Mikolajczyk et al. [2005], wik [2017b]. Na parte superior da figura, são apresentados os cálculos de *precision* e *recall* onde, *precision* é a razão entre POIs correspondentes verdadeiros positivos, por POIs correspondentes selecionados, e *recall* é a razão entre POIs correspondentes verdadeiros positivos, por POIs correspondentes relevantes; e na parte inferior é descrito o cálculo do critério de repetibilidade onde, não são considerados POIs correspondentes aqueles com erro menor que 40%, dada pela razão entre a área de intercessão e a área da união entre dois POIs relacionados de forma homográfica.

Desta forma, conhecendo o bom desempenho do detector IGFTT, quando as imagens são submetidas a transformações fotométricas e geométricas, e ainda, sabendo que a qualidade do Visual SLAM depende de robustos detectores de pontos de interesse, uma avaliação do IGFTT para VSLAM é uma contribuição interessante sob o aspecto da qualidade do mapa produzido, em relação a outros detectores de POIs. Neste sentido, este trabalho avalia a influência do algoritmo de detecção de pontos de interesse IGFTT em função do desempenho do sistema Visual SLAM, onde os resultados serão comparados com os detectores mais utilizados na literatura.

5.3 Resumo

Este capítulo apresenta uma visão geral sobre a importância que um detector de pontos de interesse robusto têm no desempenho final de um algoritmo de VSLAM. Por fim, descreve o potencial do detector IGFTT, propondo sua avaliação em função do sistema de SLAM baseado em Visão, considerando o objetivo de melhorar a qualidade do mapa produzido. No próximo capítulo é apresentada a metodologia elaborada para a avaliação.

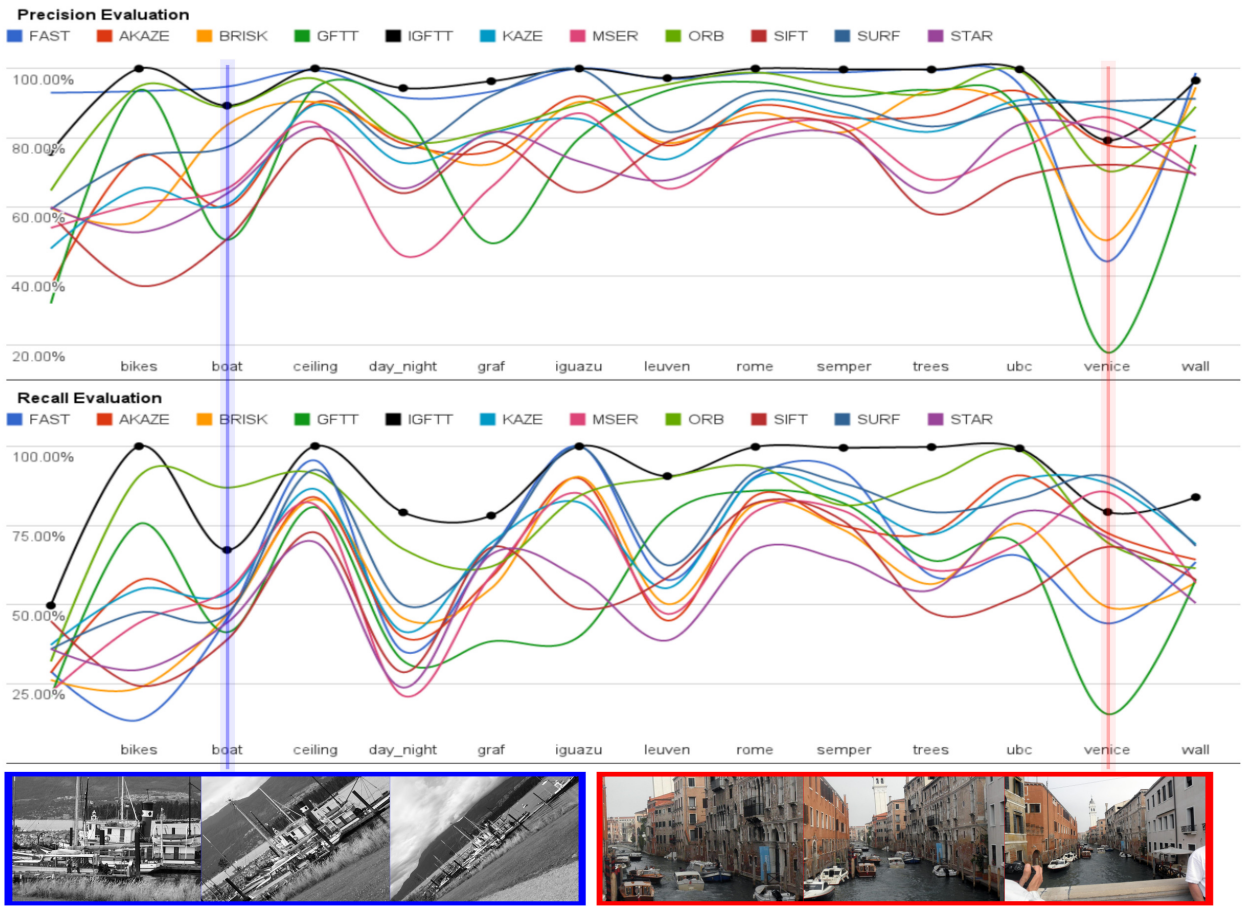

Figura 5.3: Resultados IGFTT - *precision* e *recall*. FONTE: Modificado de [Oliveira et al., 2015, p. 6]. Gráficos de avaliação de *precision* e *recall*. IGFTT apresenta desempenho igual ou superior aos demais detectores testados, exceto em imagens com *zoom* (*venice* em vermelho e *boat* em azul) pelo fato de sua pirâmide de escalas ser simples, sem borramento.

Capítulo 6

Metodologia

Este Capítulo apresenta a metodologia necessária para a avaliação do detector de pontos de interesse IGFTT em relação ao mapa produzido pelo sistema de VSLAM. A Seção 6.1 descreve sobre as sequências de *benchmark* escolhidas para os testes, também apresenta o *software* de VSLAM que embarcará o algoritmo de detecção avaliado e as configurações do *hardware* que executará o RGBDSLAM, além de comentar sobre as configurações fixas do *software* para todas as etapas de teste. A Seção 6.2 define as fases, a forma e os critérios de avaliação dos detectores bem como o procedimento de escolha dos mesmos para serem descritos mais detalhadamente na dissertação. Por fim, a Seção 6.3 aborda o caráter de cada teste como sua função, forma de mensurar e seu objetivo.

6.1 Ferramentas

Uma análise da qualidade do detector pode ser feita em função do mapa construído pelo VSLAM com relação a um mapa de referência, produzido por meio de dados obtidos do ambiente pela fusão de sensores (GPS, Escaneamento *Laser*, IMU, entre outros). Criar um ambiente controlado, com um aparato robótico composto por equipamentos de medição de alta precisão, além de passar por uma rigorosa bateria de testes para validação do mapa que servirá de base genuína do ambiente, é um tanto caro [Klippenstein and Zhang, 2009, Endres et al., 2012, Hartmann et al., 2013].

Benchmark

Diversas universidades e centros de pesquisas como *Karlsruhe Institute of Technology* e *Toyota Technological Institute at Chicago* [Geiger et al., 2012, 2013], *Oxford Mobile Robotics Group* [Smith et al., 2009] entre outros, disponibilizam *datasets* para validação e testes de novos algoritmos de odometria visual, detectores e descritores de pontos de interesse, fluxo ótico, estrutura de movimento, VSLAM e outros.

O grupo de pesquisa em Visão Computacional da Universidade Técnica de Munique - Alemanha Sturm et al. [2012] disponibiliza *datasets* para avaliar sistemas de VSLAM que utilizam sensores *Red Green Blue Depth* (RGBD) como *Kinect®* da *Microsoft®*, porém, com navegação do veículo autônomo em ambientes internos. Também descreve os equipamentos utilizados e fornece todos os parâmetros para calibração dos mesmos. O mapa base é construído através da captura do movimento do robô por marcações dispostas sobre ele, as quais são rastreadas por 8 câmeras fixas de alta velocidade (100 Hz). Trajetórias longas com ou sem fechamento de laço

são parte dos *datasets* para análise dos sistemas de odometria visual e SLAM. O site[1] do grupo de pesquisa disponibiliza ferramentas de avaliação online para a comunidade acadêmica [Sturm et al., 2012].

A escolha dos *datasets* de Sturm et al. se deu pelo fato deste disponibilizar inúmeras sequências de vídeos com suas respectivas configurações de câmera, onde cada uma é encapsulada com extensão .Bag do *Robot Operating System* (ROS), que facilita na manipulação dos experimentos. Oferece ainda códigos para validação e demonstrativos gráficos *online* e para *download*.

As sequências escolhidas e o motivo são descritos abaixo:

- Sequências extraídas com o sensor na mão de uma pessoa. O foco é testar os detectores sob ambientes familiares com filmagens manuais onde os *frames* variam dado o movimento (pequenos e aleatórios) da pessoa.

  rgbd dataset freiburg1 desk: Contém varreduras em 4 mesas em um ambiente de escritório típico. Sequência básica para testes por ser pequena em tempo de duração, poucos movimentos e muita textura.

  rgbd dataset freiburg3 long office household: Cenas domésticas e de escritório. Contem muita textura e estrutura, onde o fim do escritório se sobrepõe ao início criando um fechamento de laço. O video é relativamente grande e com movimentos pequenos.

- Sequências feitas por um robô *Pioneer-P3DX* com o sensor *Kinect®* sobre o mesmo. Proporciona o teste dos detectores sob movimentação uniforme do sensor em ambientes fechados, porém grandes.

  rgbd dataset freiburg2 pioneer 360: Ocorre um giro com mais de 360 graus sem muito deslocamento translacional.

  rgbd dataset freiburg2 pioneer slam: Há deslocamento do robô entre mesas, caixas e paredes com fechamento de laço.

  rgbd dataset freiburg2 pioneer slam3: Desloca-se ao redor de paredes.

- Sequências com sensor na mão de uma pessoa sob alterações da distância do sensor em relação aos objetos, e presença ou ausência de estrutura e textura nos objetos. A ideia é testar os detectores em relação aos fatores textura, estrutura, distância e movimentos.

  rgbd dataset freiburg3 nostructure texture far Movimentos do sensor a dois metros de uma superfície planar com textura.

  rgbd dataset freiburg3 structure notexture far Movimentos *zig-zag* do sensor a um metro de uma estrutura sem textura.

  rgbd dataset freiburg3 structure notexture near Movimentos *zig-zag* do sensor a meio metro de uma estrutura sem textura.

  rgbd dataset freiburg3 structure texture far Movimentos *zig-zag* a um metro de uma estrutura com textura.

  rgbd dataset freiburg3 structure texture near Movimentos *zig-zag* a meio metro de uma estrutura com textura.

[1] <http://vision.in.tum.de/data/datasets/rgbd-dataset>.

Software

Há inúmeros sistemas de VSLAM que processam sequências obtidas por sensor *kinect®*, porém o sistema RGBDSLAM de Endres et al. [2012] oferece recursos para configuração de parâmetros de forma simples para troca de algoritmos de detecção de pontos de interesse durante os testes. Além disso, sua escolha justifica-se por ser integrado ao *Framework* ROS, facilitando a leitura de arquivos .Bag e manuseio por interface gráfica.

O sistema é desenvolvido na linguagem C++ e agrega as bibliotecas do OPENCV 3.1, onde é utilizada neste trabalho para implementação, melhoria e validação dos algoritmos de detecção de características testados. As ferramentas de validação dos testes disponibilizados por Sturm et al. [2012] são escritas em *Python*. Da mesma forma, testes e avaliações sobressalentes propostas neste trabalho estão em conformidade com as linguagens citadas anteriormente.

O funcionamento do sistema RGBDSLAM escolhido é apresentado na Figura 6.1. A odometria visual é feita por meio de pares de imagens coloridas capturados de um sensor tipo *Kinect®*, onde são detectados POIs e correspondidos. Como cada imagem têm seu respectivo mapa de profundidade, a localização tridimensional do ponto é encontrada e relacionada entre o par de imagens. Com as correspondências dos pontos e suas profundidades, a estimativa da transformação entre *frames* é feita por RANSAC [Fischler and Bolles, 1981]. No RGBDSLAM, toda a parte descrita anteriormente é realizada *frame* a *frame* e reconhecida como a parte frontal do sistema. O mapa gerado somente pela odometria (parte frontal) não é consistente, então uma otimização das poses organizadas em grafo é produzida pelo pacote G2O [Grisetti et al., 2011], devolvendo um mapa 3D global consistente. A otimização não é efetuada todo o tempo como a odometria, mas sim realizada dado um conjunto de nós do grafo ou na finalização do sistema, logo é conhecida como a parte traseira do algoritmo. Após o término de toda a otimização o *software* salva a nuvem de pontos gerada, cria e salva *voxels* da nuvem e apresenta um mapa de ocupação 3D da câmera [Endres et al., 2012].

Hardware

O hardware com o qual os experimentos foram realizados é um Intel Core I5 CPU M 480 2.67GHz com 4 Núcleos e 6 GiB de RAM.

Configurações

Os parâmetros de configuração básicos para todos os testes com respeito ao sistema RGBDSLAM e detectores são descritos a seguir:

- Máximo de Características detectadas: 10000, porém alguns detectores não suportam este parâmetro. Testes empíricos demonstram que nenhum detector consegue detectar mais que 5000 POIs, então um valor alto é escolhido para reter todos os POIs que um detector pode encontrar.

- Máximo de Características retidas: 600, pois de forma empírica, valores maiores não alteram significativamente os resultados, isso se deve ao fato de o RANSAC necessitar de no mínimo 5 correspondências para estimar a odometria.

- Mínimo de Correspondências: 20, para o RANSAC ter informações na procedência da escolha dos *Inliers*. De forma empírica, abaixo de 20 correspondências, a pose entre imagens começa a ser mal estimada.

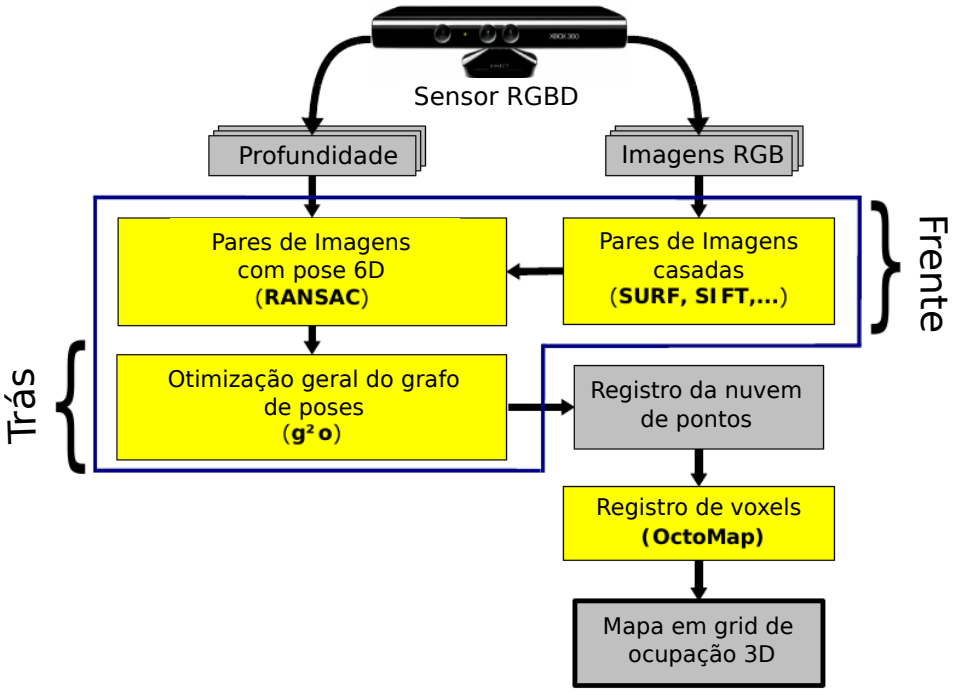

Figura 6.1: Sistema RGBDSLAM. FONTE: Modificado de [Endres et al., 2012, p. 4]. O Sensor RGBD capta as imagens e profundidades dos *pixels* destas. Por frente, o sistema corresponde imagens e calcula a pose da câmera com RANSAC. Ao mesmo tempo, rodando por trás, o sistema vai otimizando o grafo gerado para melhorar o mapa criado. Uma nuvem de pontos vai sendo construída de acordo com a profundidade dos *pixels* e suas relações com a pose da câmera. Ainda o mapa pode ser representado por *voxels* e *grid* de ocupação.

- Algoritmo que faz Correspondência: BRUTEFORCE_OPENCV, pois preferiu-se não utilizar a opção *Fast Library for Approximate Nearest Neighbors* (FLANN), por ser lenta em relação a alguns tipos de extratores, segundo os autores do sistema RGBDSLAM Endres et al. [2012].
- Iterações do RANSAC: 100, empiricamente obteve bons resultados. Valores maiores tornam o sistema desnecessariamente mais lento, e menores prejudicam a odometria fornecendo POIs correspondentes ruins.
- Descritores de Características: FREAK, para todos os detectores que não tenham um descritor padrão. A escolha se dá pelo fato de o detector avaliado IGFTT utilizar o descritor FREAK como padrão.

Para demais parâmetros dos detectores assumem-se valores padrões do OPENCV, assim como para todos os parâmetros de configuração do *Framework* RGBDSLAM. No Apêndice C estão disponíveis todas as configurações do sistema para fase de testes.

6.2 Comparação Prática

De forma a estabelecer uma comparação prática entre detectores, o número de execuções de teste serão cinco, por observar que não há desvios discrepantes da média entre os resultados

de cada rodada para um único teste, mesmo estes sob aleatoriedade do RANSAC. Logo não há necessidade de amostras maiores.

Os detectores utilizados para comparação são os implementados pelo OPENCV 3.1 apresentados na Tabela 6.1. Os detectores IGFTT e IGFTT2 utilizam algoritmos da biblioteca OPENCV porém não são disponíveis pela mesma. O algoritmo IGFTT2 é uma melhoria do detector IGFTT proposta pelo autor neste trabalho.

Algoritmo	**Detector**	**Descritor**	**OPENCV 3.1**	**Testados**
AGAST	✓	–	✓	✓
AKAZE	✓	✓	✓	✓
BRIEF	–	✓	✓	✓
BRISK	–	✓	✓	–
DAISY	–	✓	✓	–
FAST	✓	–	✓	✓
FREAK	–	✓	✓	✓
GFTT	✓	–	✓	✓
IGFTT	✓	–	–	✓
IGFTT2	✓	–	–	✓
KAZE	✓	✓	✓	✓
LATCH	–	✓	✓	–
LUCID	–	✓	✓	–
MSER	✓	–	✓	✓
ORB	✓	✓	✓	✓
SIFT	✓	✓	✓	✓
STAR	✓	–	✓	✓
SURF	✓	✓	✓	✓

Tabela 6.1: Relação de detectores e descritores da biblioteca OPENCV 3.1 testados. Alguns detectores como por exemplo SIFT, SURF, KAZE entre outros, implementam seus próprios métodos de descrição do POI. Os descritores de KAZE e AKAZE, diferentes dos demais, só descrevem pontos detectados pelos mesmos. IGFTT e IGFTT2 são implementações separadas da biblioteca OPENCV que serão avaliadas. Descritores BRIEF e FREAK são parte dos testes por comporem os detectores ORB e melhor opção do IGFTT respectivamente. Detectores que implementam seus próprios descritores serão testados em conjunto, os demais utilizarão o descritor FREAK. Oliveira et al. [2015] testa seu método de detecção descrevendo seus POIs com FREAK, justificando o motivo da padronização.

Os algoritmos de detecção que estão em foco nesta dissertação são aqueles que, dentre todos os testes em todas as sequências, seu *rank* final é menor que a média geral dos *ranks*. O SIFT faz parte das classes de detectores analisados por ser citado largamente na literatura Dwarakanath et al. [2012], Miksik and Mikolajczyk [2012], Barandiaran et al. [2012], Oliveira et al. [2015], Endres et al. [2012], Klippenstein and Zhang [2007, 2009], Gil et al. [2010], Hartmann et al. [2013]. Logo, os algoritmos com melhor *rank* serão submetidos a todos os testes subsequentes, separados em 4 fases, descritas a seguir.

- FASE 1: Erro de translação e rotação do mapa gerado. Onde os dados são apresentados pela média geral do *Root Mean Square Error* (RMSE) em metros e graus.
- FASE 2: Tempo de processamento da detecção, descrição e correspondência. Como o tempo de processamento não varia em relação a uma técnica utilizada, será apresentado em milissegundos através da média.
- FASE 3: Taxa de Repetibilidade de POIs e Taxa de Probabilidade de sobrevivência dos POIs. São apresentados em frações dada a média geral.

- FASE 4: Poses comparadas sequenciais e Poses comparadas sequenciais não estimadas. São apresentadas em frações dada a média geral.

Exceto a Fase 1 e 2, que analisam todos os nós do grafo de cena criados no sistema RGBDSLAM, a Fase 3 analisa os nós que fazem parte do menor caminho entre o nó inicial e o nó final do grafo. A Fase 4 analisa a fração dos nós que fazem parte do menor caminho do grafo de cena, com ou sem pose atribuída. Exemplo do grafo disponível na Figura 6.6.

Para fins de comparação entre algoritmos de detecção de pontos de interesse, uma avaliação estatística dos resultados do Erro da Trajetória Absoluta (ATE) é feita com o intuito de informar se existem diferenças significantes entre as classes de detectores.

O resultado aceitará ou rejeitará a hipótese nula (H0), onde esta quer saber se *"seria idêntica a avaliação do ATE pelos vários métodos de detecção de características, em relação às diversas sequências?"*, ou seja, *"pode-se afirmar que não existem diferenças estatisticamente significantes entre as classes de detectores?"*. Caso a hipótese nula seja rejeitada (indicando que existem diferenças entre os detectores avaliados), uma segunda avaliação será feita para descobrir quais detectores diferem.

Submetendo os dados disponíveis para análise de restrições, a fim de escolher o melhor teste estatístico, verificou-se que estes não são distribuídos normalmente, de acordo com o teste de Jarque and Bera [1980], e nem têm homogeneidade entre si, de acordo com o teste de Brown and Forsythe [1974], como pode ser observado na Tabela 6.2. Também confirmou-se que são dados dependentes, onde o valor apresentado por cada classe de detector depende de cada sequência de *benchmark*, e pareados, onde todas as classes de detector avaliados obtiveram resultados para todas as sequências analisadas.

Desta forma, foi escolhido o teste estatístico que melhor se adapta às características da informação sendo este o método de Friedman Demšar [2006], García et al. [2010]. A ideia do método considera que, se as diversas amostras provêm de uma mesma população, isto é, se elas são estatisticamente iguais, a média dos *ranks* em cada classe de detectores será aproximadamente igual, garantindo a aceitação da hipótese nula. A hipótese alternativa (H1) seria de que as amostras não pertenceriam à mesma população, seriam estatisticamente diferentes, demonstrando que há diferenças entre as médias dos *ranks* das diversas classes de detectores, assim rejeitando a hipótese nula.

Para a segunda avaliação, caso a hipótese nula seja rejeitada pelo teste de Friedman com confiança de 95%, é necessário um pós-teste para identificar quais classes são diferentes entre si. Ainda compreendendo as restrições da informação disponível para os teste estatísticos, optou-se pelo método de Nemenyi [Demšar, 2006, García et al., 2010] e Wilcoxon [Nehmzow, 2006, Wilcoxon, 1992].

O método de Nemenyi compara todas as classes entre si, e as avalia como diferentes quando a diferença das médias de seus *rankings* são maiores que uma diferença crítica. Já o teste de Wilcoxon faz o *ranking* das diferenças entre duas classes para cada *dataset* e compara as diferenças entre *ranks* positivos e negativos. A ideia de usar dois pós-testes é devido ao método de Nemenyi ser menos conservador em seus resultados, pois faz um balanceamento da distância crítica de acordo com o número de classes testadas. Diferente de Wilcoxon que compara par a par as classes, sendo mais discriminativo.

Os cálculos referentes aos testes estatísticos não serão apresentados, pois não são o foco do trabalho, mas são explicados de forma detalhada em Nehmzow [2006], Demšar [2006], García et al. [2010].

Teste de Normalidade de Jarque-Bera (com $\alpha = 0,05$) distribuído de acordo com Qui-Quadrado com 2 graus de liberdade ($\chi^2_F = 5,9915$)							
Classe de Detectores	IGFTT2	IGFTT	GFTT	SIFT	SURF	FAST	AGAST
Estatística	**10,182**	4,172	3,157	1,568	3,061	**9,185**	4,545
P-Valor	**0,006**	0,124	0,206	0,456	0,216	**0,010**	0,103
Hipótese Nula	**Rejeita**	Aceita	Aceita	Aceita	Aceita	**Rejeita**	Aceita
Teste de Homogeneidade de Levene (com $\alpha = 0,05$) distribuído de acordo com F com 6 e 63 graus de liberdade ($F = 2,250$)							
Estatística	**5,963**						
P-Valor	**0,00005**						
Hipótese Nula	**Rejeita**						

Tabela 6.2: Teste de normalidade e homogeneidade dos dados. O teste de normalidade valida se os resultados de cada detector seguem uma distribuição normal. O teste é feito com intervalo de significância de 5% (0,05). A tabela demonstra que os detectores IGFTT2 e FAST tem a hipótese nula rejeitada dado que o p-valor é menor que 0,05 não seguindo uma distribuição normal. O teste de homogeneidade valida se diferentes detectores têm seus resultados parecidos. O teste também é feito com intervalo de significância de 5% e demonstra que a hipótese nula é rejeitada, ou seja, não há homogeneidade entre os resultados dos detectores testados.

6.3 Comparação Teórica

Nesta seção são apresentadas características de cada teste, como sua função, forma de mensurar e seu objetivo. Estas são divididas em fases, conforme Seção 6.2, para facilitar a organização.

6.3.1 FASE 1: Quanto ao Erro da Pose Relativa (RPE)

Disponível entre as ferramentas de avaliação de Sturm et al. [2012], o *Relative Pose Error* (RPE) mede a acurácia da localização do robô na trajetória dado um intervalo de tempo fixo Δ. Este erro corresponde ao escorregamento (*drift*) da trajetória sofrido pelo robô, conforme acumulo de ruídos durante estimação da odometria. Desta forma, o erro é utilizado para avaliar o movimento da câmera ou sistema de odometria visual. Sturm et al. [2012] define RPE como

$$E_i = \left(Q_i^{-1} Q_{i+\Delta}\right)^{-1} \left(P_i^{-1} P_{i+\Delta}\right) \tag{6.1}$$

onde Ei é o erro da pose relativa no intervalo de tempo $i + \Delta$, Q é uma sequência de poses da trajetória base, P é uma sequência de poses da trajetória estimada e ambos pertencem a $SE(3)$[2].

Com os erros ao longo da sequência, calcula-se o *Root Mean Square Error* (RMSE) ou raiz quadrada do erro quadrático médio do componente de translação (ou rotação) sendo

$$RMSE(E_{1:n}, \Delta) = \left(\frac{1}{m} \sum_{i=1}^{m} ||trans(E_i)||^2\right)^{\frac{1}{2}} \tag{6.2}$$

onde $trans(E_i)$ é o componente de translação da diferença entre poses relativas E_i, n é a sequência de poses da câmera e $m = n - \Delta$.

Se $\Delta = 1$, a análise do escorregamento é realizada por *frame*, e a medida dada em metros (ou graus caso seja utilizado o componente de rotação). A Figura 6.2 mostra a ideia.

[2]Grupo especial Euclidiano que agrega transformações de rotação e translação.

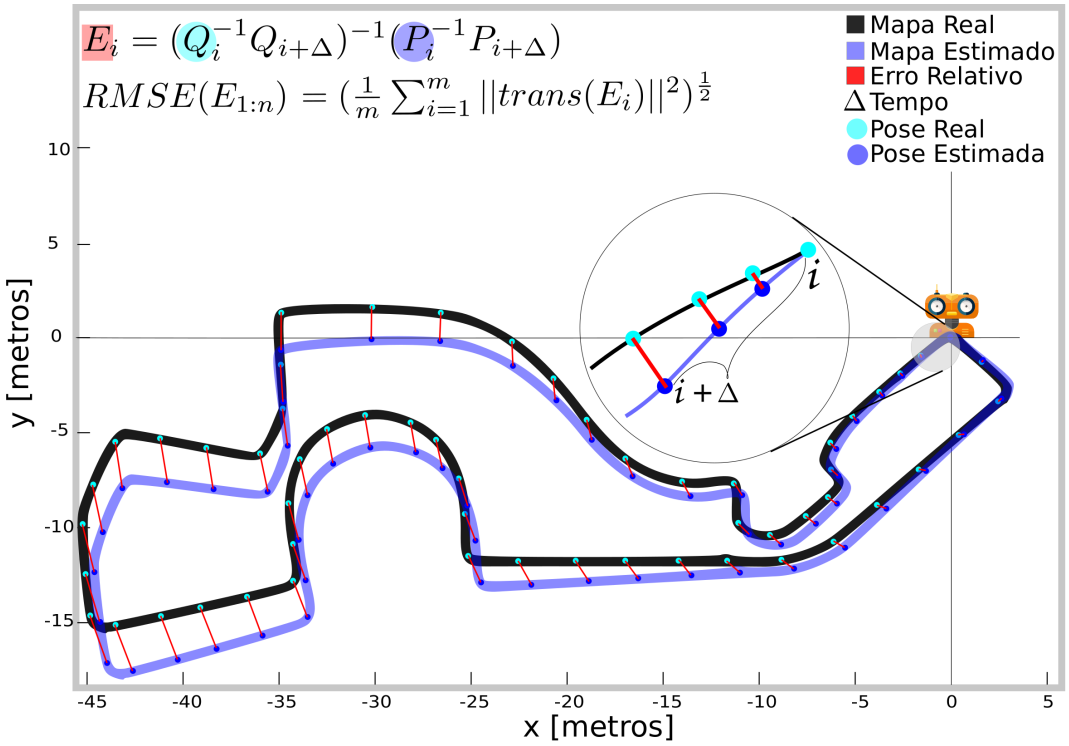

Figura 6.2: Erro da Pose Relativa. FONTE: Autor. O teste avalia a acurácia do deslocamento da câmera, conforme o erro de escorregamento (*drift*) sofrido pelo robô em sua trajetória, comparada à trajetória base. O escorregamento é calculado pela diferença dos componentes de translação ou rotação das poses relativas, conforme a métrica RMSE, dado um intervalo de tempo. O erro de toda a sequência é a média dos RMSE calculados.

Desta forma, a trajetória do robô é avaliada, com relação à odômetros visuais, em diversas sequências de poses e com diferentes detectores de pontos de interesse, utilizando como métrica o erro quadrático médio dos erros relativos de pose encontrados.

6.3.2 FASE 1: Quanto ao Erro da Trajetória Absoluta (ATE)

Segundo Sturm et al. [2012], a consistência global do mapa construído pelo SLAM pode ser avaliada pelas diferenças entre o mapa base e o mapa estimado pelo robô. Após o alinhamento do mapa estimado $P_{1:n}$ por uma transformação de corpos rígidos S utilizando mínimos quadrados no mapa base $Q_{1:n}$, o *Absolute Trajectory Error* (ATE) no tempo fixo i é apresentada pela equação

$$F_i = Q_i^{-1} S P_i \tag{6.3}$$

onde F_i é o erro da pose relativa direta no tempo i entre o mapa base Q_i e o mapa estimado P_i.

Similar ao RPE, ATE é baseada na métrica RMSE dada em metros, considerando somente o componente de translação $trans$ de toda a sequência, sendo

$$RMSE(F_{1:n}) = \left(\frac{1}{n} \sum_{i=1}^{n} ||trans(F_i)||^2 \right)^{\frac{1}{2}} \tag{6.4}$$

A Figura 6.3 apresenta o método.

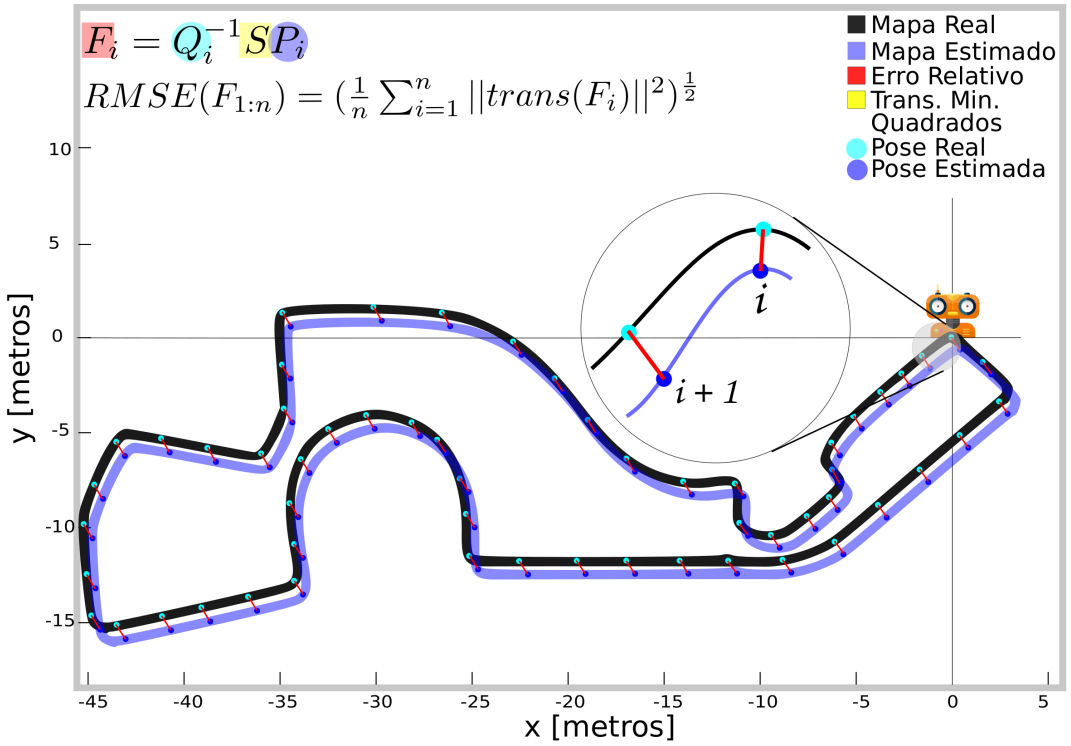

Figura 6.3: Erro da Trajetória Absoluta. FONTE: Autor. O teste avalia a consistência global do mapa, conforme erros existentes entre poses do mapa estimado e o mapa base. O alinhamento entre os mapas é calculado por mínimos quadrados. O erro é dado pela métrica RMSE.

Desta forma, a trajetória do robô é avaliada, com relação ao mapeamento global de um ambiente, em diversas sequências e com diferentes detectores de pontos de interesse, utilizando como métrica o erro quadrático médio dos erros relativos entre poses (translação) que constituem tanto o mapa base como o mapa estimado.

6.3.3 FASE 2: Quanto ao tempo de processamento dos métodos conforme detecção, descrição, correspondência e número de POIs encontrados

O tempo de processamento do algoritmo na fase de detecção, descrição e correspondência do POI é avaliado pela média em relação a todas as sequências de *benchmark*.

Os tempos contabilizados são aqueles de cada *frame*, onde ao menos um POI foi detectado, ou descrito, ou correspondido pelo sistema RGBDSLAM. Então é calculada a média de todos os tempos (por tipo de análise) conforme o número de *frames* utilizados. O resultado é apresentado em milissegundos por *frame*.

Pretende-se com estes testes encontrar o detector de pontos de interesse com a maior média em relação ao número de POIs detectados por *frame*, e os menores tempos de processamento.

6.3.4 FASE 3: Quanto à taxa de repetibilidade dos POIs

Gil et al. [2010] avaliam em sua pesquisa a repetibilidade dos pontos de interesse sob mudanças geométricas e fotométricas, analisando sequências com no máximo 21 frames e em condições controladas, conforme critério de repetibilidade por rastreamento de POIs, baseado na Matriz Fundamental estimada por RANSAC, onde o ponto correspondente entre duas imagens é

aquele que se encontra mais próximo da linha epipolar. Logo, Gil et al. não utiliza descritores de pontos de interesse para correspondê-los e ainda faz a verificação das correspondências entre POIs manualmente.

Como a avaliação do detector ocorre em função do sistema VSLAM e sob variados tipos de sequência com milhares de *frames*, se torna impossível validar a correspondência de POIs manualmente. Desta forma, utiliza-se também o RANSAC para escolher as melhores correspondências, pois estas definem as transformações entre imagens para estimação da pose da câmera.

Gil et al. definem que a taxa de repetibilidade tr_i no *frame i* da sequência é dada por

$$tr_i = \frac{np_i}{np_r} \tag{6.5}$$

onde np_i é o número de pontos de interesse encontrados no *frame i* da sequência e np_r é o número de pontos de interesse detectados no *frame* de referência da sequência.

Um detector perfeito é aquele que detecta os mesmos pontos do *frame* de referência nos *frames* posteriores de forma que $tr_i = 1$, porém observa-se um decaimento no valor de tr_i indicando que alguns pontos do *frame* de referência são perdidos nos *frames* subsequentes [Gil et al., 2010].

Desta forma, o objetivo neste trabalho é avaliar a taxa de repetibilidade do detector de POIs de forma similar a Gil et al., mas utilizando uma janela de tempo Δ para visualizar a variação da taxa em toda a trajetória, logo o *frame* $i = r + \Delta$. A taxa de repetibilidade geral Gtr do detector em relação à sequência é dada pela média das taxas de cada conjunto de *frames*. A taxa do conjunto Gtr_n é dada pela média das taxas dos *frames* deste conjunto. A Equação 6.6 mostra o cálculo. É esperado então valores altos de média, pois estes determinam detectores robustos. A Figura 6.4 mostra a ideia.

$$\begin{aligned} Gtr_n(tr_{1:n}) &= \frac{1}{n}\sum_{i=1}^{n} tr_i \\ Gtr &= \frac{1}{n}\sum_{i=1}^{n} Gtr_i \end{aligned} \tag{6.6}$$

6.3.5 FASE 3: Quanto à probabilidade dos POIs aparecerem em *frames* futuros

Durante uma trajetória, o robô detecta e armazena bons marcos visuais do ambiente. Bons marcos visuais dizem respeito aos pontos de interesse que são detectados em vários *frames* consecutivos e incluem somente os pontos que podem ser detectados nestes *frames*, excluindo então POIs pouco rastreáveis. Desta forma, o número de marcos visuais, incluídos no cálculo do sistema de VSLAM que determina a pose da câmera e construção do mapa global diminuem, reduzindo também a complexidade do problema do SLAM [Gil et al., 2010].

Gil et al. então analisa em quantos *frames* um ponto de interesse pode ser rastreado antes de ser integrado ao mapa, usando probabilidade condicional, sendo

$$P(f_j|f_i) = \frac{np_{1:j}}{np_{1:i}}, j >= i \tag{6.7}$$

onde $np_{1:k}$ é o número de POIs que podem ser rastreados a partir do primeiro *frame* até o *frame* k da sequência.

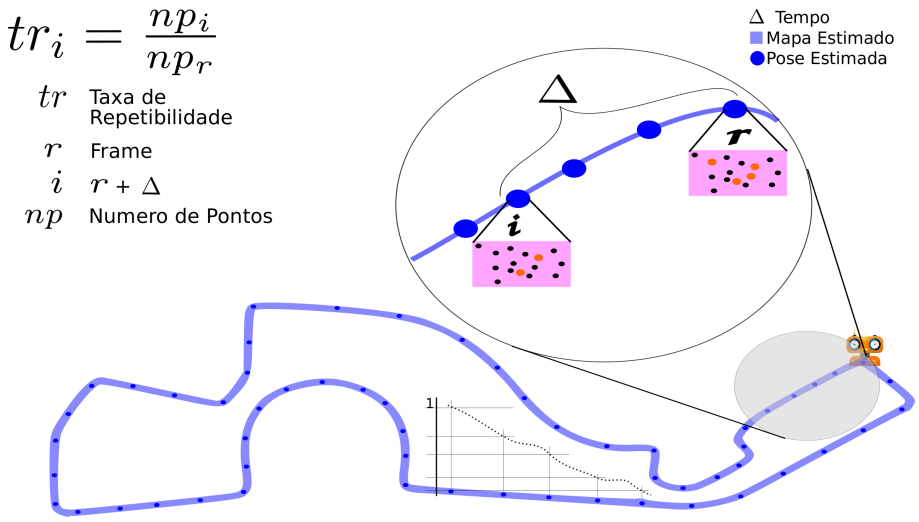

Figura 6.4: Taxa de Repetibilidade. FONTE: Autor. O teste avalia o quanto POIs se repetem dado sucessivos *frames*. A taxa de repetibilidade no intervalo Δ é a razão do número de POIs do *frame* np_i pelo número de POIs do *frame* base np_r. O resultado total é dado pela média das taxas parciais tr_i e determina a capacidade que um detector oferece à odometria de rastrear o mesmo POI várias vezes.

O valor de $P(f_j|f_i)$ representa a probabilidade de um ponto de interesse ser rastreado até o quadro f_j dado que já foi rastreado até o quadro f_i. $P(f_j|f_i) = 1$ se f_i e f_j tiverem os mesmos pontos rastreados. Deste modo, pode-se estimar a sobrevivência de um ponto de interesse nos *frames* futuros.

Similar a Gil et al., neste trabalho a estabilidade do detector de pontos de interesse é avaliado, observando a probabilidade de rastreamento dos pontos em *frames* futuros conforme seus rastreamentos em *frames* passados. Uma janela de tempo Δ é utilizada para visualizar a variação da média das probabilidades em toda a trajetória, logo os *frames* i e j pertencem aos *frames* do intervalo Δ desde que o *frame* j seja igual ou posterior ao *frame* i, logo $j >= i$. A taxa de probabilidade geral do detector é dada conforme a Equação 6.6 da taxa de repetibilidade, onde é esperado valores altos de média, determinando então detectores estáveis. A Figura 6.5 exibe o método de análise.

6.3.6 FASE 4: Quanto às taxas de nós do grafo da cena contínuos com e sem pose

O sistema RGBDSLAM constrói um grafo orientado para representar o posicionamento da câmera dadas as correspondências entre dois *frames*, conforme mostrado na Figura 6.6. Como a avaliação de detecção de pontos de interesse proposta na Fase 3 depende da continuidade de interligação dos *frames* (ex: *frame* 1 com *frame* 2, *frame* 2 com *frame* 3, assim sucessivamente), é de interesse observar se há *frames* processados que não fazem parte da continuidade de interligações (nós em vermelho na Figura 6.6), ou seja, analisar se há, e quanto é a fração de nós que não fazem parte do menor caminho do grafo dados nós inicial e final. Outro caso a ser analisado é o de nós que fazem parte da sequência porém suas posições não foram atribuídas pelo sistema RGBDSLAM (nós em azul claro na Figura 6.6).

A métrica por fração de nós é aplicada para avaliar a relação dos algoritmos de detecção com a quantidade de *frames* utilizados, e *frames* utilizados com ausência de poses.

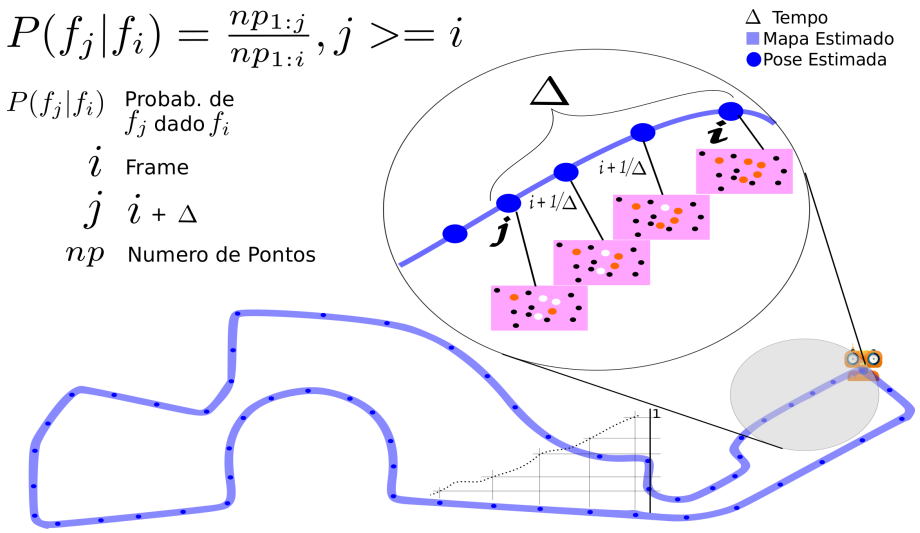

Figura 6.5: Taxa de Probabilidade. FONTE: Autor. O teste avalia a probabilidade que um detector de POIs tem de rastrear pontos de interesse em *frames* futuros dado que estes já foram detectados em *frames* anteriores. O resultado é dado pela razão entre o número de POIs np, que são detectados no *frame* j, conforme já foram rastreados a partir do *frame* i, determinando então a capacidade de um detector em se manter estável quanto à rastreabilidade de POIs futuros e demonstrando confiança para redução de pontos visuais calculados no SLAM, diminuindo sua complexidade.

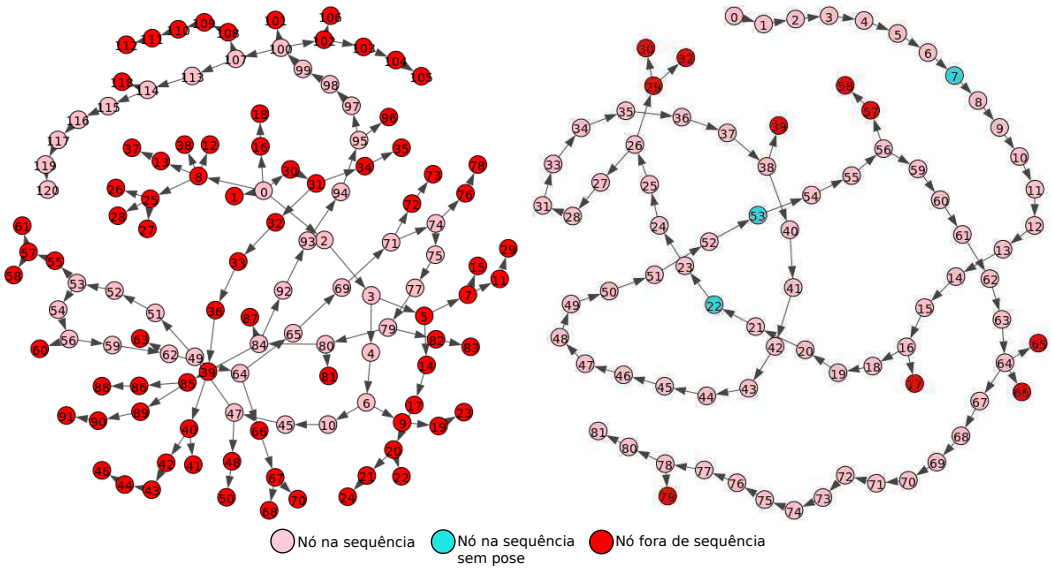

Figura 6.6: Grafo de *Frames*. FONTE: Autor. O sistema RGBDSLAM cria um grafo dirigido com as poses de cada câmera conforme o *frame* capturado. A seta indica relação de correspondência de POIs e transformação de pose entre *frames*. Nós de cor salmão indicam o menor caminho do grafo entre os *frames* inicial e final. Em vermelho estão os caminhos que não fazem parte da sequência e prejudicam a odometria. E em azul claro são aqueles que fazem parte da sequência mas suas poses são ruins para odometria e não foram atribuídas pelo sistema.

Pretende-se com estes testes avaliar a influência do método de detecção no desempenho do sistema RGBDSLAM em relacionar *frames* sucessivos. Espera-se encontrar o detector de

POIs com a maior proporção em relação a *frames* sequenciais, e a menor proporção se tratando de *frames* sequenciais sem poses.

6.4 Resumo

Este capítulo apresenta as ferramentas utilizadas para procedimento dos testes como os *benchmarks*, o sistema de VSLAM escolhido e suas respectivas configurações quanto aos parâmetros e ao *hardware*. Apresenta a forma de escolha dos algoritmos utilizados na avaliação dos detectores, assim como os métodos estatísticos escolhidos para diferenciá-los. Ainda apresenta o escopo dos testes e a forma com que cada um irá proceder em suas avaliações. Por fim, demonstra o arcabouço teórico das avaliações e expectativas de resultados. No próximo capítulo são descritos os resultados e a avaliação dos testes proferidos.

Capítulo 7

Resultados e Discussão

Este capítulo apresenta os resultados e discussões relativas aos testes realizados no trabalho. Na Seção 7.1 são avaliados os algoritmos de detecção em relação à odometria e ao mapa gerado pelo sistema RGBDSLAM. Na Seção 7.2 são analisados o tempo de processamento e o número de POIs encontrados pelos detectores de características. Resultados referentes às taxas de repetibilidade e probabilidade são descritos na Seção 7.3. Já na Seção 7.4 a avaliação é apresentada conforme a sequência dos nós do grafo de cena. Por fim, na Seção 7.5 os resultados dos detectores quanto ao erro da trajetória absoluta são analisados estatisticamente.

O *ranking* dos algoritmos sob testes em todas as fases é apresentado na Figura 7.1, onde as três linhas horizontais representam os resultados dos primeiros colocados. Na Figura 7.2 os detectores que ficaram abaixo da média no *ranking* foram SURF, IGFTT2 (aperfeiçoamento do IGFTT proposto neste trabalho), AGAST, GFTT, IGFTT e FAST e estão indicados em azul e os demais em laranja. Também, o algoritmo SIFT é destacado por compor os testes mesmo estando em 11º no *ranking*, pois este é base para quase todos os artigos que testam novos detectores e descritores de características. Logo, os resultados apresentados a seguir são destes algoritmos citados, e a discussão ocorre conforme a abordagem dos testes.

7.1 FASE 1 - Avaliação dos Detectores em Função da Odometria e do Mapa

Nesta fase são avaliados os erros de translação e rotação da odometria e do mapa gerado através da otimização do grafo de cena feita pelo sistema RGBDSLAM. A Figura 7.3 apresenta dois exemplos em sequências distintas e de processos entre *frames* para identificar o fluxo ótico dos POIs e assim determinar o movimento da câmera. A Figura 7.4 demonstra como exemplo a correspondência entre o mapa estimado pelo robô e o mapa de referência da sequência de *benchmark Desk*, em 3D.

7.1.1 ATE - Erro da Trajetória Absoluta

A Figura 7.5 apresenta para cada sequência a média do escore fornecido pelo *Absolute Trajectory Error* (ATE) por detector, dada por 5 execuções. Este erro é a métrica que determina a acurácia do detector com relação à correspondência da trajetória absoluta e a trajetória real, alinhadas por mínimos quadrados e realizada pelo sistema RGBDSLAM.

Percebe-se que a maioria dos algoritmos, em sequências que contém estruturas 3D e texturas, adquiridas próximas ou não do sensor, têm um pequeno erro translacional de pose.

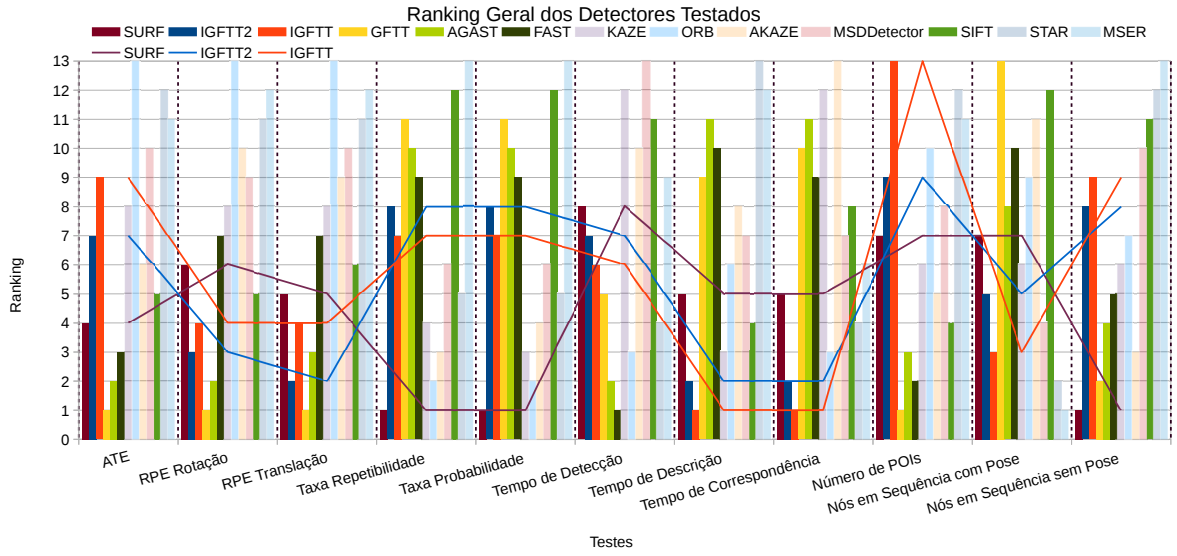

Figura 7.1: *Ranking* dos detectores por teste. FONTE: Autor. O gráfico apresenta o *ranking* dos detectores testados conforme todos os testes propostos neste trabalho. Quanto menor a barra, melhor a colocação do detector. Barras com cores mais fortes são de detectores melhor ranqueados conforme Figura 7.2. A ordem das barras (esquerda para direita) segue a ordem dos *rankings* e são separadas por teste. As linhas no gráfico são utilizadas para demonstrar os resultados dos três melhores colocados no *ranking* geral, sendo respectivamente SURF, IGFTT2 e IGFTT. Estas linhas não representam a continuidade dos resultados, somente facilitam a visualização global das informações.

Sequências que utilizam *Pioneer-P3DX* tiveram os piores resultados devido ao fato de nestas sequências o robô fazer movimentos bruscos de rotação, gerando um fluxo ótico muito grande e fazendo com que POIs que estavam em um *frame* desaparecessem no próximo. Outro efeito causado pela rápida rotação do robô é o borramento das imagens, o que diminui a magnitude do gradiente de um local onde era considerado POI, fazendo com que este não seja encontrado pelo algoritmo de detecção.

Quando estruturas tridimensionais na cena não refletem os pontos luminosos projetados pelo sensor *Kinect®*, ou estas estruturas estão demasiadamente longe da câmera, a varredura da profundidade feita pelo sensor falha ou a distância dos objetos se tornam iguais, conforme podem serem vistas na primeira linha da Figura 7.3 onde o fundo da imagem aparece todo preto (falha da varredura) ou branca (igualdade das distâncias dos objetos), mesmo que a imagem contenha textura. Isto provoca, principalmente no caso das sequências produzidas pelo robô *Pioneer*, o não espalhamento dos POIs por toda a imagem, pelo fato de o sistema RGBDSLAM depender do vínculo da profundidade ao POI detectado, fazendo com que a odometria perca qualidade e influencie diretamente no resultado do teste de ATE.

A Figura 7.6 apresenta a média geral do ATE por detector. Com um RMSE de 0,944 metros o detector IGFTT demonstrou ser a pior escolha para os conjuntos de sequências testados no sistema RGBDSLAM. O melhor resultado foi o do detector do qual ele é derivado, o GFTT, com erro de 0,278 metros. É perceptível que o número de POIs detectados por *frame* influencia diretamente neste resultado.

Para um sistema de odometria visual aferir posições satisfatórias da câmera é necessário que a estimação da Matriz Fundamental, feita pelo RANSAC, utilize POIs robustos. Significa que

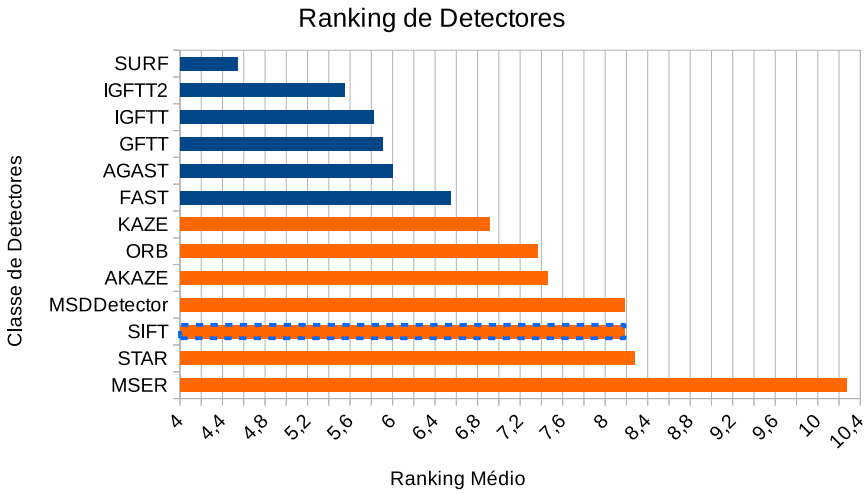

Figura 7.2: *Ranking* geral dos detectores. FONTE: Autor. O gráfico apresenta o *ranking* geral dos detectores testados conforme a média do *ranking* de todos os testes. Quanto menor a média, melhor a colocação do detector. Em barras azuis são apresentados os melhores colocados onde foram escolhidos por se encontrarem abaixo da média do *ranking* geral. A barra em laranja com traços azuis representa o detector SIFT, pois este, mesmo estando em 11º lugar, foi escolhido por sua alta citação em trabalhos relacionados ao contexto desta pesquisa.

a localização do POI em dois ou mais *frames* deve se encontrar na mesma posição tridimensional dos objetos ou texturas representados pela imagem. Assim, a pose da câmera, dada a relação de *frames*, é melhor estimada [Sroba et al., 2015].

Algoritmos de detecção como SIFT e SURF são robustos porque estimam a localização central dos POIs em nível de *subpixel* dada a distribuição do gradiente dos *pixels* vizinhos. Diferente destes detectores, POIs localizados por GFTT e IGFTT não têm precisão de *subpixel*, tornando seus resultados ruins com relação à odometria.

O detector de características IGFTT se baseia no algoritmo GFTT que encontra cantos bons para rastreamento (descrito na Subseção 3.2.4). Porém, o algoritmo IGFTT em suas configurações originais, faz a varredura da imagem com uma janela de 31×31 *pixels*, diferentemente do GFTT, que utiliza uma janela de 3×3 *pixels*. Uma janela maior faz com que IGFTT encontre cantos relativamente maiores, ignorando regiões menores que poderiam vir a ser um canto.

IGFTT possui um elevado custo computacional, pois sem uma heurística varre toda a imagem, além de o cálculos da Matriz Hessiana ser executado em uma janela de 31×31 *pixels*. O que reduz seu esforço total de processamento é o baixo número de POIs que retorna, pois a configuração original assume distância euclidiana mínima de 10 *pixels* entre POIs. Por outro lado, o baixo número de POIs prejudica o RANSAC durante a escolha de POIs correspondentes que melhor representem o modelo de transformação entre imagens, o que decrementa a qualidade da aferição da odometria.

Pelos problemas expostos acima, propôs-se uma melhoria ao algoritmo de detecção IGFTT, sendo:

- Localização com precisão de *subpixel*;
- Aumento da repetibilidade dos POIs;

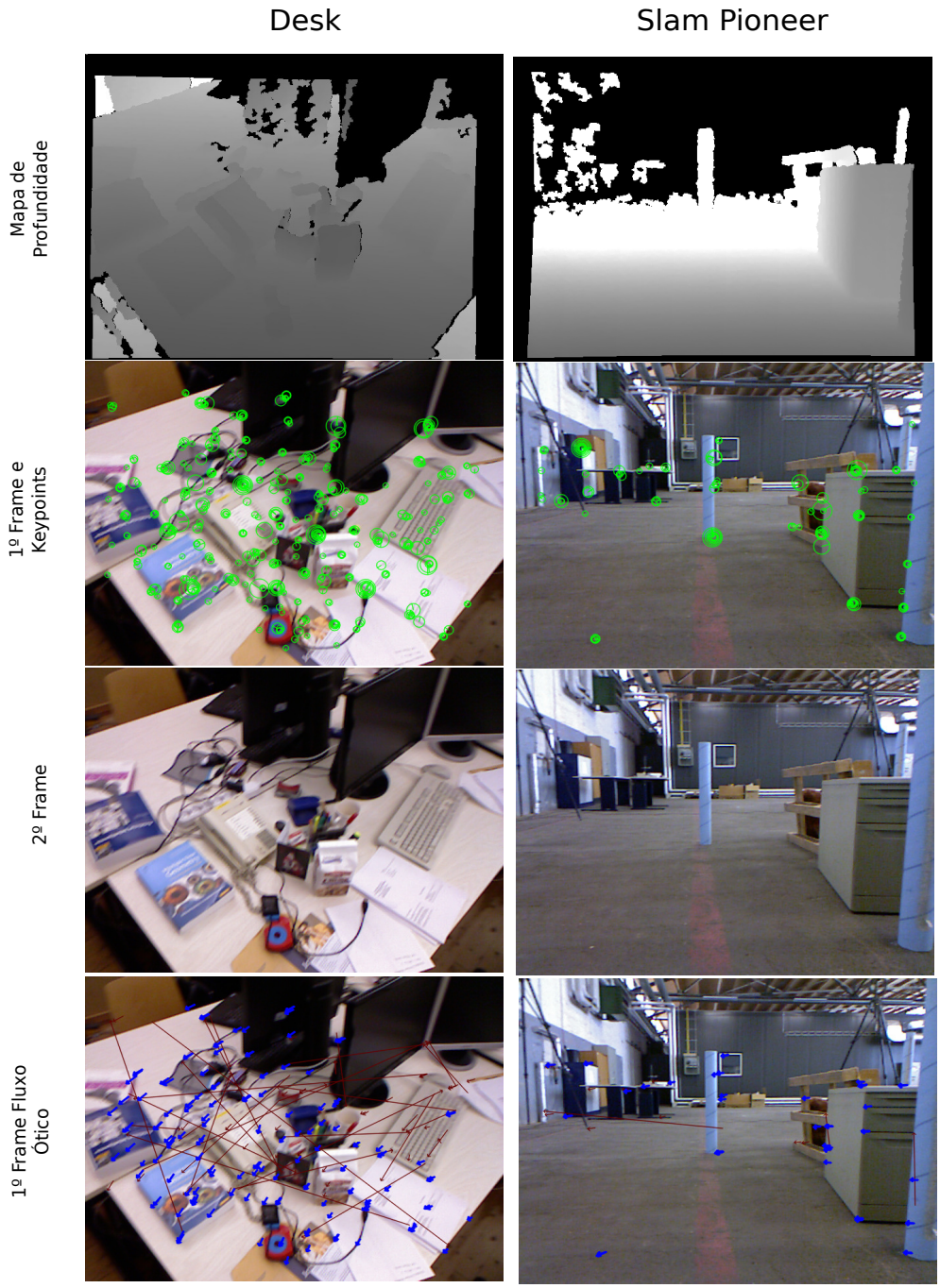

Figura 7.3: Processo de aquisição da imagem, detecção de POIs e correspondência de *frames* das sequências *Desk* e *Pioneer Slam*. FONTE: Modificado do Sistema RGBDSLAM. A figura mostra na primeira linha a aquisição de profundidade dos pixels da imagem capturada pelo sensor *Kinect®* em duas sequências de *benchmark* onde, quanto mais claro o *pixel*, mais distante da câmera se encontra. Na segunda linha é apresentada a localização dos POIs no primeiro *frame* capturado onde, são marcados por círculos verdes de tamanho variando conforme a escala em que foi detectado e orientação. O segundo *frame* é então capturado e apresentado na terceira linha. Conforme ocorre o deslocamento dos POIs entre *frames*, o fluxo ótico é calculado e mostrado com setas azuis, indicando a direção do deslocamento dos melhores POIs, escolhidos pelo RANSAC. As setas marrons são correspondências ruins desprezadas pelo RANSAC.

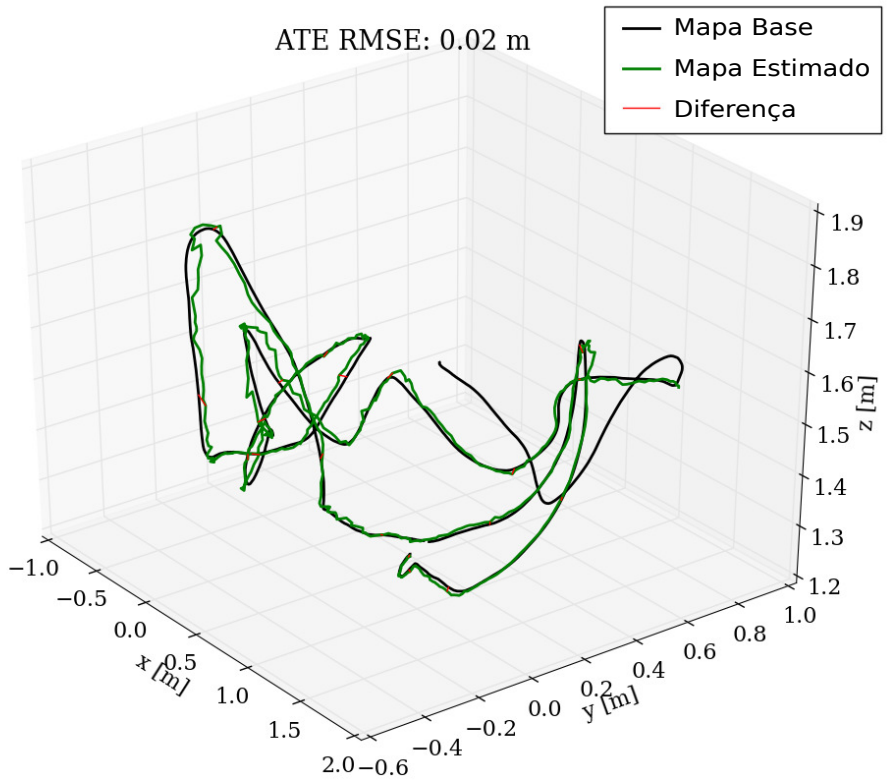

Figura 7.4: Erro da trajetória absoluta entre o mapa de referência da sequência *Desk* e o mapa estimado pelo robô. FONTE: Retirada do Sistema RGBDSLAM. A linha em preto representa a trajetória de referência fornecida pela sequência de *benchmark*. Em verde é representada a trajetória estimada pelo robô. Os erros entre as duas trajetórias é demonstrada pelas linhas vermelhas.

- Aumento do número de POIs detectados na imagem.

Este detector melhorado foi chamado de IGFTT2 e utiliza o algoritmo de *subpixel* proposto em Zhu et al. [2009] e implementado na biblioteca OPENCV. O aumento da repetibilidade se deu ao escalar a imagem original 1,2 vezes seu tamanho, o que equivale a reduzir a janela de varredura, de forma a criar 8 escalas, onde a escala original encontra-se na segunda posição do vetor de escalas. Na Figura 7.6 é possível perceber a melhora do detector.

É possível manipular de várias formas os parâmetros do algoritmo IGFTT2 para que este tenha melhores desempenhos em determinados tipos de testes ou em sequências específicas de *benchmarks*. Logo, as alterações ocorridas foram parametrizadas em função do *ranking* geral dos detectores, de forma a torná-lo robusto genericamente, onde o IGFTT2 alcançou a segunda posição. Para um melhor desempenho no teste ATE (desprezando o tempo gasto na detecção), diminuiu-se a janela de varredura para 15×15 com distância euclidiana de 5 *pixels* entre POIs, sem necessidade de aumento de escala, logo aumentando o número de POIs detectados. Com estes parâmetros modificados, o erro ATE caiu de 0,944 para 0,388 metros, superando SIFT (0,618), SURF (0,404) e FAST (0,399), pois o aumento de POIs na imagem proporcionou ao RANSAC melhores escolhas de pontos correspondentes para triangulação. Esta parametrização específica do algoritmo IGFTT2 para o teste ATE é apresentada como exemplo somente, e não foi considerada nos teste de forma geral.

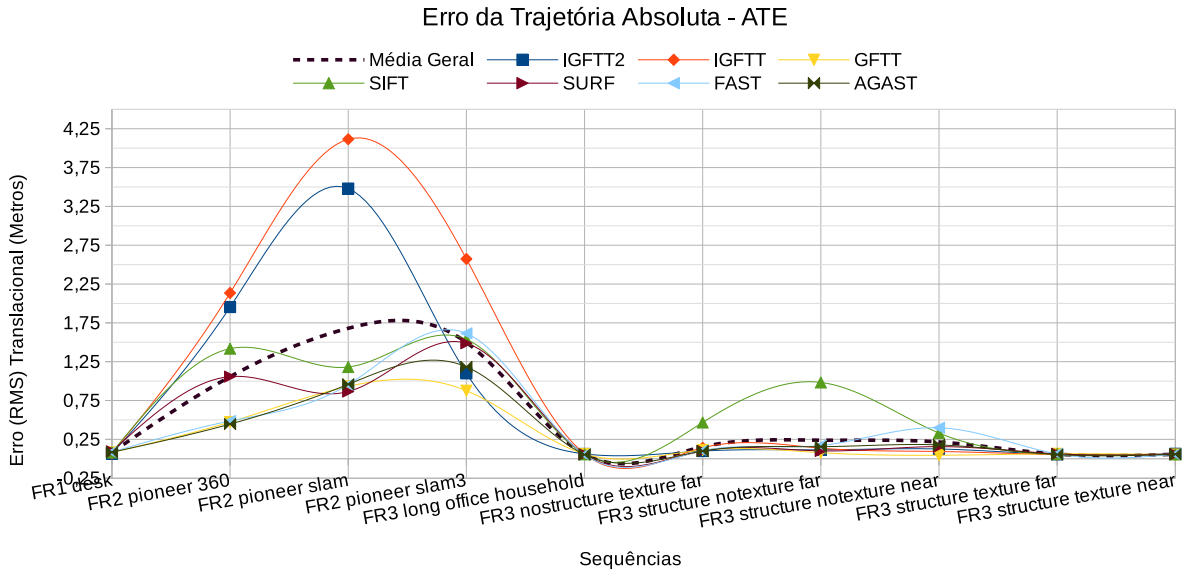

Figura 7.5: Erro (RMSE) da trajetória absoluta por sequência. FONTE: Autor. O grafico apresenta o erro da trajetória absoluta dos detectores testados em relação às sequências utilizadas como *benchmark*. O gráfico é apresentado em forma de linhas coloridas para facilitar a visualização de forma global, não representando a continuidade dos resultados no eixo horizontal.

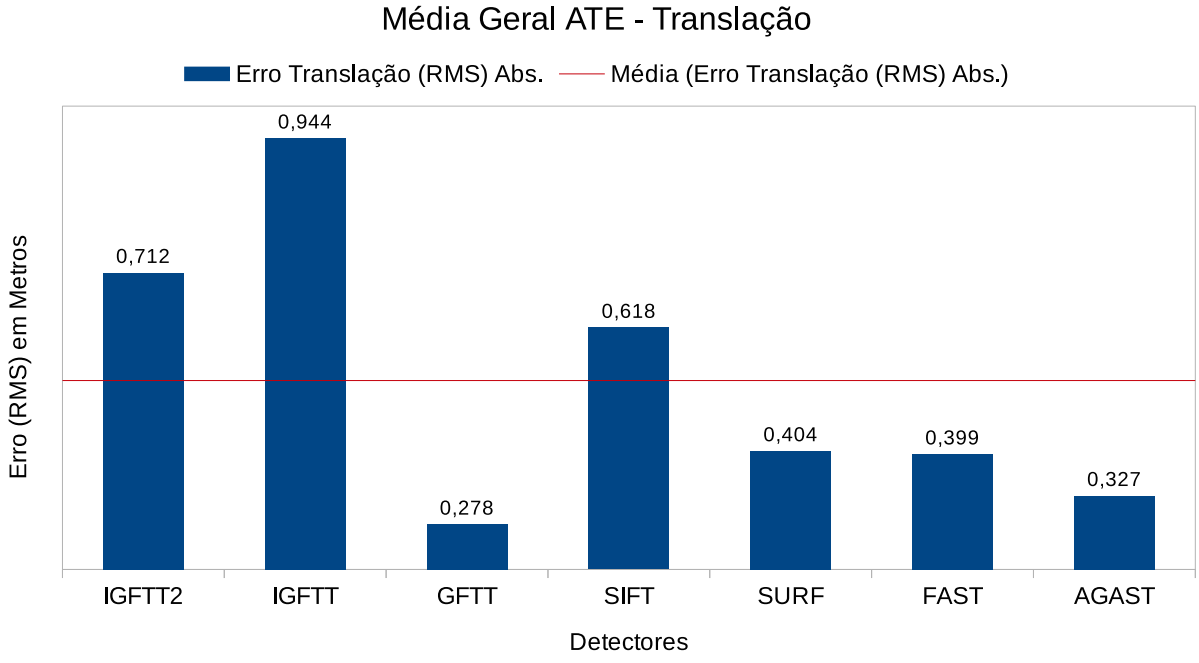

Figura 7.6: Erro (RMSE) médio da trajetória absoluta por detector. FONTE: Autor. Para fim de comparação entre resultados apresentados no gráfico, a linha vermelha representa a média dos valores dos detectores avaliados. O melhor desempenho foi conquistado pelo algoritmo GFTT e o pior IGFTT.

A Figura 7.7 apresenta os erros ATE em um gráfico de *boxplot*. No gráfico, o × representa a média dos dados. O retângulo representa 50% dos dados, estando entre o primeiro e terceiro quartil. A linha dentro do retângulo mostra a mediana dos dados indicando tendência. As linhas que partem dos extremos do retângulo exibem os limites interquartis, que representam mais de 99% dos dados. O circulo cheio mostra *outliers*, valores de erro discrepantes dentro de uma mesma amostra, que não se encontram dentro de 99% da distribuição dos dados, ultrapassando os limites interquartis superior ou inferior. É possível perceber então no gráfico, a estabilidade dos detectores de cantos GFTT e AGAST, quanto à menor dispersão dos erros obtidos em todas as sequências. IGFTT2 e FAST apresenta *outliers*, ou seja, para alguma sequência de *benchmark*, IGFTT2 e FAST tiveram desempenho muito discrepante das demais sequências.

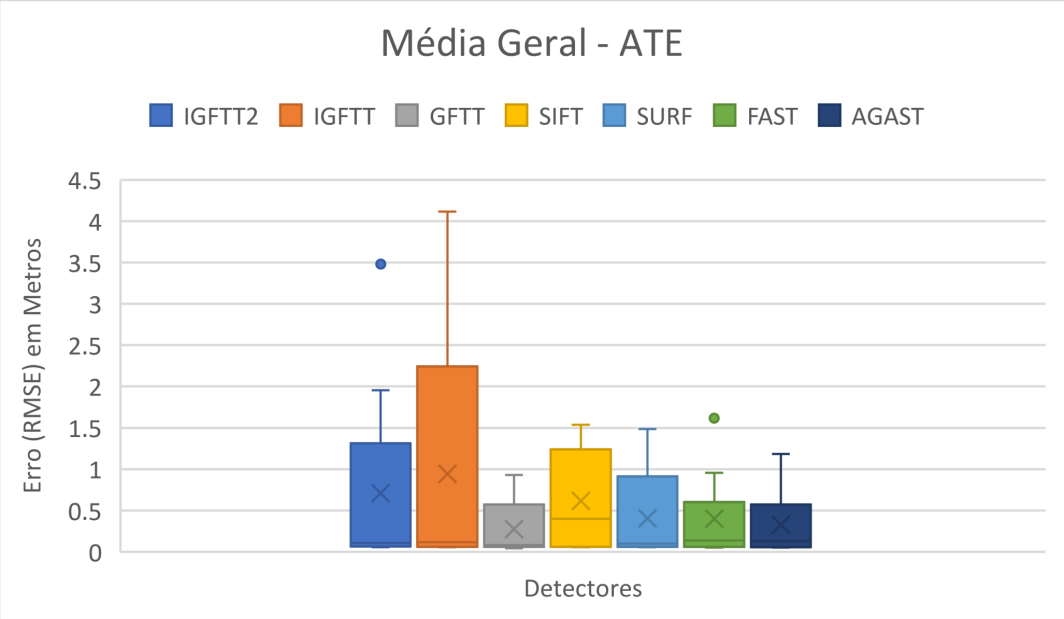

Figura 7.7: *Boxplot* do erro (RMSE) médio da trajetória absoluta por detector. FONTE: Autor. GFTT e AGAST têm resultados menos dispersos em relação aos demais detectores, demonstrando estabilidade sob utilização em variados tipos de sequências. IGFTT2 e FAST apresentam *outliers* demonstrando desempenho muito discrepante com relação à alguma sequência específica.

7.1.2 RPE - Erro da Pose Relativa

As Figuras 7.8 e 7.9 apresentam respectivamente a média geral dos erros (RMSE) das poses relativas translacionais e rotacionais com relação a todas as sequências testadas conforme os algoritmos de detecção de POIs. Este erro está relacionado ao escorregamento (*drift*) que a trajetória estimada da câmera sofre em relação a trajetória real.

O erro de rotação e translação na estimativa da pose ao longo das sequências ocorre devido às características peculiares de cada algoritmo de detecção e da sequência ao qual o

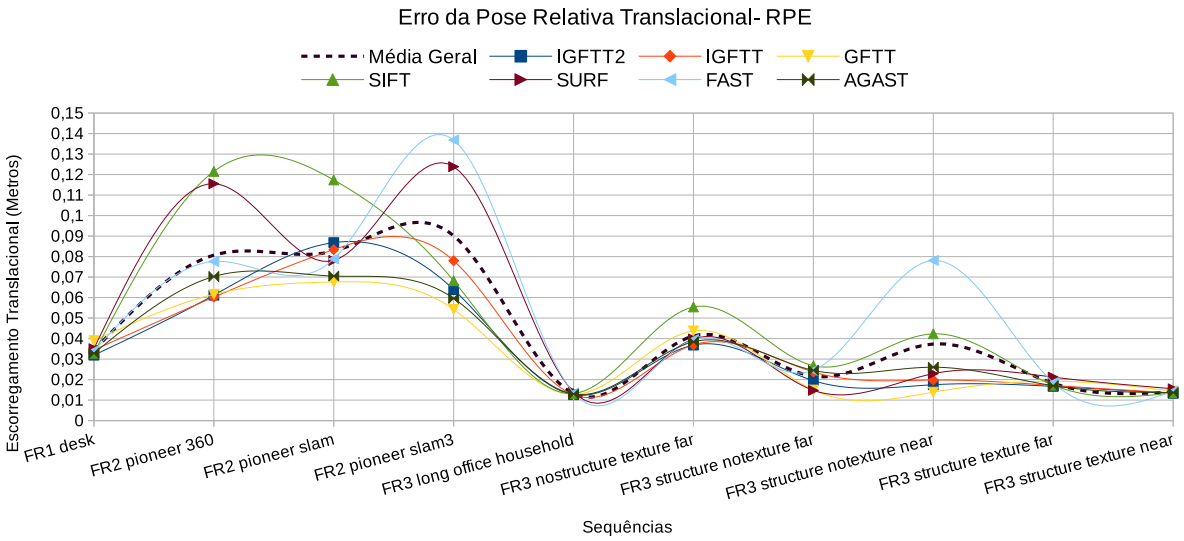

Figura 7.8: Média RMSE da pose relativa translacional por sequência. FONTE: Autor. O gráfico apresenta o erro da pose relativa dado pelo escorregamento translacional do robô, conforme detectores testados em relação às sequências utilizadas como *benchmark*. As sequências *Pioneer 360*, *Pioneer SLAM* e *Pioneer SLAM 3* mostram interferir mais nos erros de translação sob os algoritmos testados, pois ocorrem demasiadas transformações fotométricas e geométricas nas imagens capturadas. Ambientes sem estruturas 3D mas com textura, longe da câmera, também influenciam no aumento dos erros. O gráfico é apresentado em forma de linhas coloridas para facilitar a visualização de forma global, não representando a continuidade dos resultados no eixo horizontal.

detector foi testado. Os algoritmos IGFTT, GFTT, AGAST e FAST são baseados em cantos, os quais são encontrados em praticamente todas as sequências. Uma simples sombra ou um canto de um objeto tridimensional sem textura, mas com variação de luminosidade suficiente, pode ser detectado como sendo um POIs por estes detectores.

O FAST teve um desempenho ruim, comparado aos seus semelhantes aqui testados, pois o *threshold* de comparação entre o *pixel* central e o segmento contíguo da borda é relativamente alto, prejudicando a detecção de pontos em sequências sem textura. Diferente destes são o SIFT e SURF que se baseiam em regiões bastante discriminativas, onde estas dependem de intenso gradiente e, em sequências não texturizadas, têm o desempenho bastante inferior. A quantidade de POIs detectados influencia diretamente na estimativa da pose da câmera. Quanto maior seu número, maior o conjunto de POIs que o RANSAC escolherá para representar uma transformação entre imagens, e mais precisa será a estimativa da pose.

A qualidade do sensor ótico também influencia na geração de erros translacionais e rotacionais. Quando a câmera tem auto foco, auto brilho e seus movimentos são bruscos, mudanças fotométricas repentinas alteram a direção e intensidade do gradiente no local onde foi detectado um POI. Outro fato ocorre quando o sensor não é do tipo CCD *Global Shutter* gerando transformações geométricas difícil de prever porque dependem da velocidade e do tamanho do objeto na cena, além do tempo de exposição do obturador do sensor, e fazendo com que a triangulação feita pelo RANSAC para estimar a pose da câmera seja imprecisa, mesmo que receba os mais robustos POIs correspondentes.

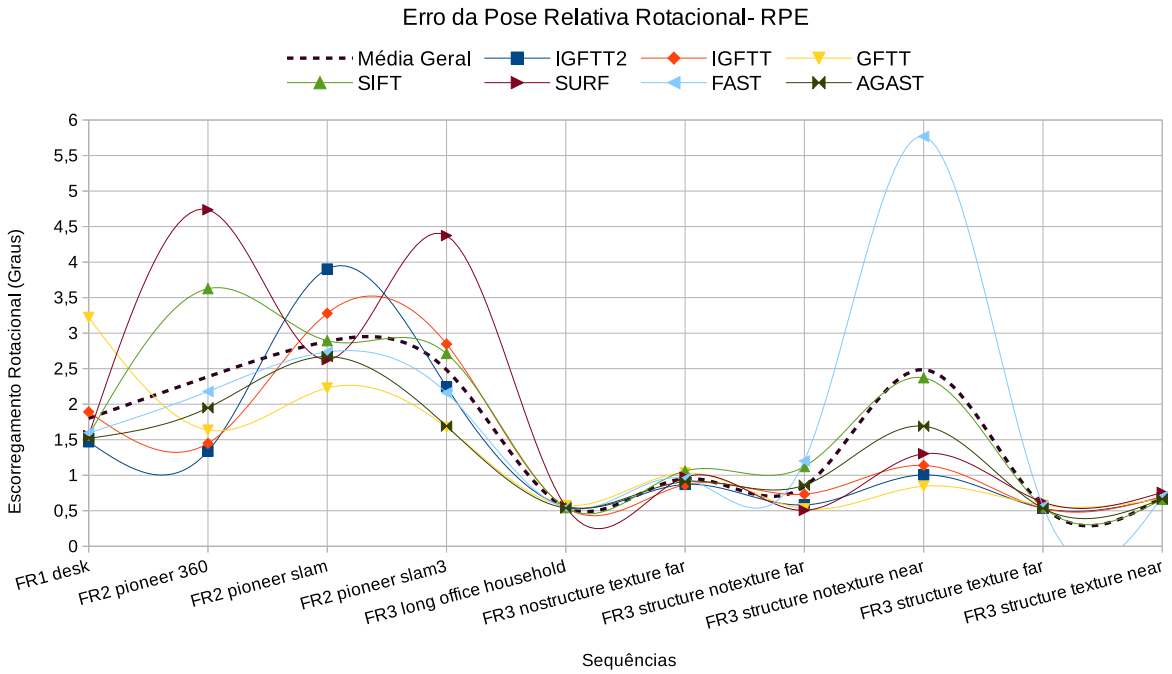

Figura 7.9: Média RMSE da pose relativa rotacional por sequência. FONTE: Autor. O gráfico apresenta o erro da pose relativa dado pelo escorregamento rotacional do robô, conforme detectores testados em relação às sequências utilizadas como *benchmark*. Assim como nos erros de translação, as sequências *Pioneer 360*, *Pioneer SLAM* e *Pioneer SLAM 3* também interferem mais nos erros de rotação. Mudanças de iluminação e movimentos bruscos que borram as imagens, degradam a localização dos POIs, prejudicando na estimação da pose da câmera. O gráfico é apresentado em forma de linhas coloridas para facilitar a visualização de forma global, não representando a continuidade dos resultados no eixo horizontal.

Os problemas citados anteriormente podem ser visualizados nas Figuras 7.8 e 7.9 onde a magnitude dos erros de todos os detectores testados se apresentam bem mais elevados que os demais no gráfico. As sequências que sofrem transformações fotométricas (devido ao ambiente ter entrada de luz externa e o sensor ter auto foco e auto brilho) e geométricas (sensor do tipo CMOS - *Rolling Shutter* e movimentos bruscos) são a *Pioneer-360*, *Pioneer-Slam* e *Pioneer-Slam-3*.

As Figuras 7.10 e 7.11 apresentam respectivamente a média geral dos erros translacionais e rotacionais conforme o detector de pontos de interesse. O algoritmo SURF, que está em primeiro do *ranking* geral teve o pior resultado com 1,87 graus de erro na rotação, e o AGAST conseguiu o melhor resultado com 1,26 graus. Já para translação, SIFT teve o pior desempenho com um erro de 4,98 cm, enquanto o primeiro colocado apresentou erro de 3,25 cm. O IGFTT2 manteve a segunda melhor colocação tanto no erro de rotação quanto translação com respectivamente 1,3 graus e 3,55 cm.

As dispersões dos dados de erros translacionais e rotacionais das sequências, em função do detector de POIs, podem ser visualizadas nas Figuras 7.12 e 7.13. SIFT, SURF e FAST apresentaram dispersões maiores que os demais algoritmos em ambos os testes. Na média geral de translação, FAST apresentou a menor dispersão dos dados e teve sua média mais próxima da mediana, o que mostra simetria nos erros adquiridos pelo detector, sob testes em variadas sequências. Em relação ao erro de rotação, o algoritmo IGFTT2 obteve a menor dispersão

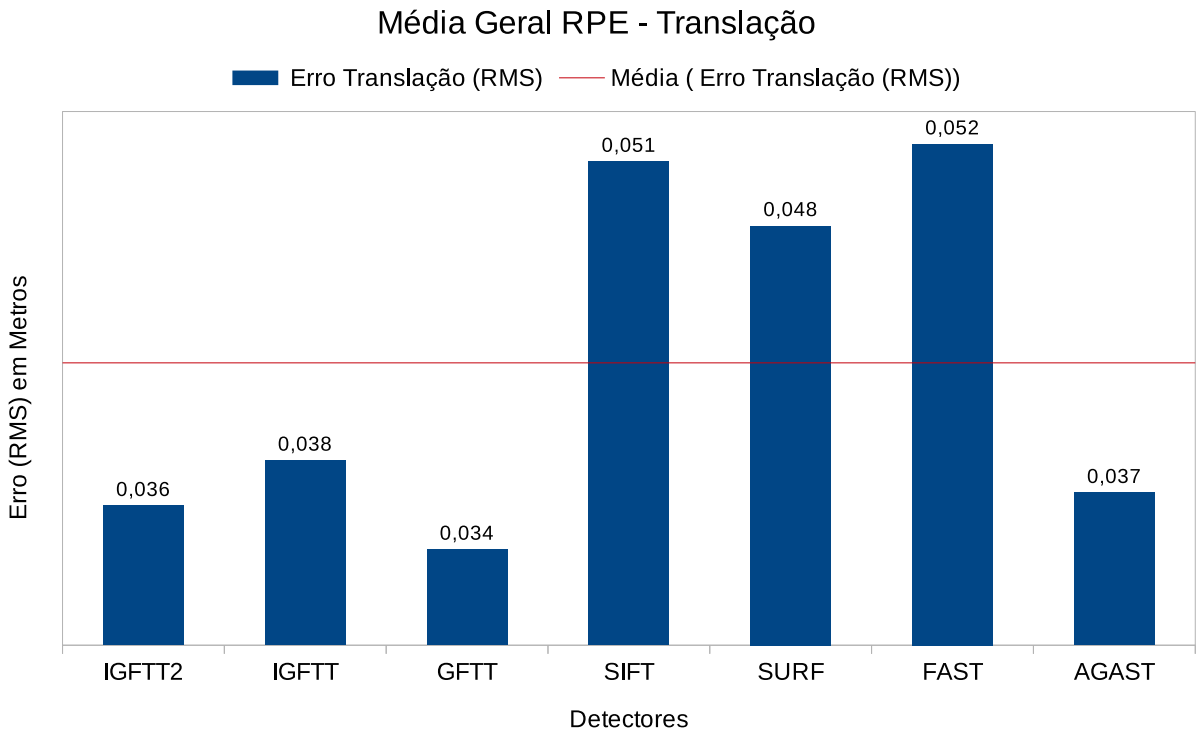

Figura 7.10: Média geral RMSE de poses relativas translacionais. FONTE: Autor. Para fim de comparação entre resultados apresentados no gráfico, a linha vermelha representa a média dos valores dos detectores avaliados. O melhor desempenho foi conquistado pelo algoritmo GFTT, e FAST o pior, mas muito próximo do resultado alcançado pelo SIFT.

mas apresentou *outliers*, demonstrando ser estável para maioria das sequências testadas, porém com erro alto e discrepante em alguma sequência. FAST também apresentou *outliers*. AGAST mostrou ser o mais estável com relação a dispersão dos erros, dentre os algoritmos testados.

7.2 FASE 2 - Avaliação dos Detectores em Função do Tempo de Processamento por *Frame*

Nesta fase os detectores são avaliados quanto ao tempo de processamento em relação à detecção, descrição e correspondência dos POIs. Como estes dados são mensurados com relação à média de POIs por *frame*, esta informação também é utilizada como parâmetro de avaliação.

A Figura 7.14 apresenta a média geral de todas as rodadas de testes em todas sequências conforme o algoritmo de detecção. Os gráficos são empilhados para uma melhor visualização, de baixo para cima, sendo o número de POIs por *Frame* e o tempo de processamento da detecção, descrição e correspondência (*matching*), onde os tempos são dados em milissegundos. Estas medidas remetem a uma possível escolha dentre os detectores de POIs, sobre o qual é melhor para ser embarcado em *hardware*, conforme o poder computacional disponível, ou ser processado em tempo real por um sistema RGBDSLAM.

O modelo de detector ideal seria aquele que, para esta fase de avaliação, detectasse o maior número de POIs em uma imagem, e o tempo de detecção, descrição e correspondência

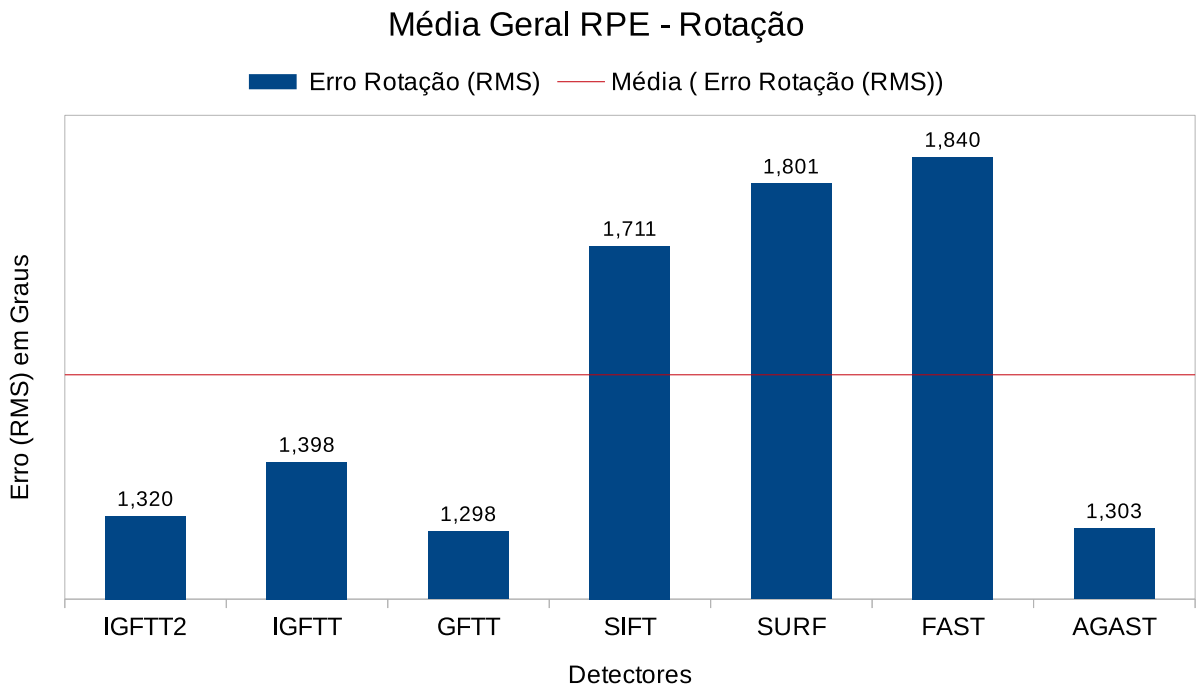

Figura 7.11: Média geral RMSE de poses relativas rotacionais. FONTE: Autor. A linha vermelha representa a média dos valores dos detectores avaliados. O melhor desempenho foi conquistado pelo algoritmo GFTT, e FAST o pior. Os resultados do AGAST e IGFTT2 então muito próximos do primeiro colocado.

fossem os menores possíveis, de forma que todo este tempo de processamento fosse menor que o período de *frames*. Se considerarmos 24 *frames* por segundo, o intervalo entre dois *frames* seria de 41,66 milissegundos, logo, um tempo máximo para ocorrer toda a fase de processamento dos POIs. Claro que toda a parte que envolve o Visual SLAM em si demanda tempo, mas assume-se fora do escopo desta comparação.

Segundo a Figura 7.14 o processo de detecção de POIs (gráfico com barras vermelhas) é onde se consome o maior tempo. O SIFT teve o pior desempenho precisando de 170 milissegundos somente para detectar uma média de 348 POIs por *frame* (gráfico de barras azuis). Resultado esperado, porque o SIFT detecta os pontos através da diferença de Gaussianas, em várias escalas e interpolando-as, para detectar o melhor ponto na melhor escala, demandando muito processamento. O melhor resultado foi de 3 milissegundos na detecção de 433 POIs, obtido pelo FAST. Este detector descarta regiões logo na primeira comparação do *pixel* central com seus vizinhos direito e esquerdo, consumindo tempo somente em regiões promissoras, tornando-se assim extremamente rápido. Quanto maior o número de POIs detectados por *frame*, melhores as chances do RANSAC encontrar bons pontos para representar transformações entre imagens.

Na Figura 7.15 é muito perceptível a amplitude da dispersão que os resultados do SIFT têm em relação aos demais detectores, de acordo com o tempo de detecção dos POIs por *frame*. O IGFTT mesmo tendo um resultado 24 vezes pior que o FAST, manteve a dispersão pequena, demonstrando ser estável no tempo gasto para detectar POIs, conforme as sequências analisadas. FAST é o mais estável.

No gráfico com barras de cor verde que mostra o tempo de descrição do POI, percebe-se uma tendência em seguir o número de POIs detectados. Isso ocorre porque, exceto o SIFT e

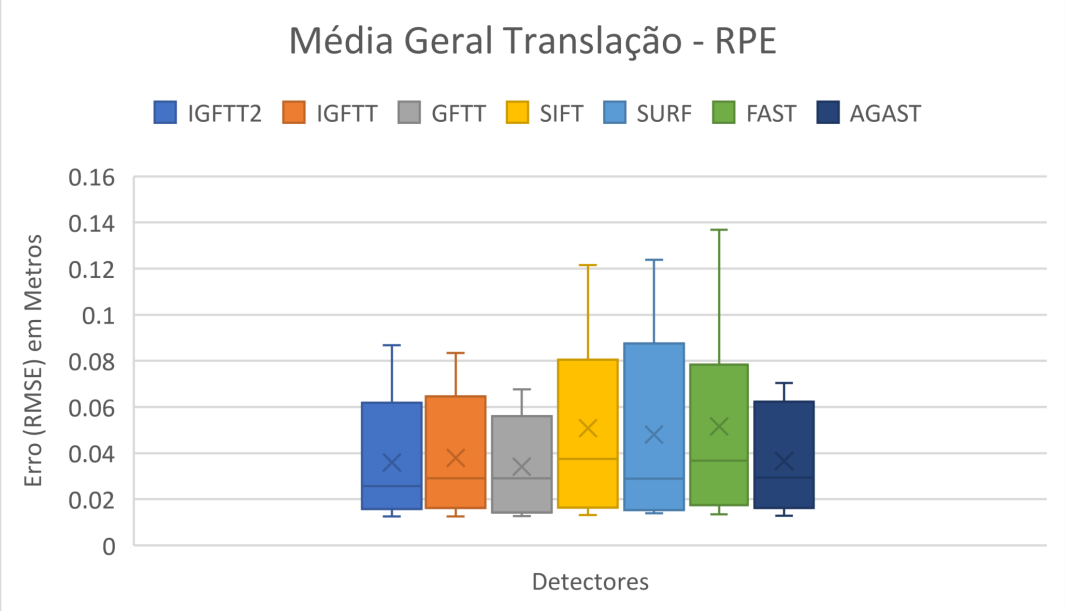

Figura 7.12: *Boxplot* do erro (RMSE) médio da pose relativa translacional por detector. FONTE: Autor. GFTT tem resultados menos dispersos em relação aos demais detectores, demonstrando maior estabilidade sob utilização em variados tipos de sequências. SIFT, SURF e FAST apresentaram dispersões bem maiores que os demais detectores.

o SURF que têm seus próprios métodos de descrição, os demais utilizam o FREAK. IGFTT têm o menor tempo de descrição com 2,21 ms e AGAST com o maior tempo, 11,77 ms. Estes resultados dependem do número de POIs detectados por *frame*. Os detectores SIFT e o SURF apresentaram resultados praticamente iguais mesmo tendo número de POIs diferentes. Em sequências sem textura SIFT detecta poucos pontos, fazendo com que o tempo de descrição seja bem pequeno e influenciando então na média geral.

No gráfico com barras amarelas são apresentados os valores de correspondência *matching* para cada detector. IGFTT conseguiu o menor tempo com 7,15 ms, o pior resultado obteve o AGAST com 46,87 ms. Analisando os valores surge a dúvida: porque os tempos de correspondências dos detectores estão proporcionais aos números de POIs detectados, sendo que vetores binários são mais velozes que vetores de pontos flutuantes, em termos de comparação? Segundo Alahi et al. [2012], FREAK é veloz por utilizar a operação "ou exclusivo"para comparação, além de descartar aproximadamente 90% dos candidatos à correspondência já no teste dos primeiros 16 *bytes*. SIFT e SURF calculam a distância euclidiana dos K vizinhos entre vetores de pontos flutuantes, deixando-o lento. O problema está no algoritmo de correspondência *Brute Force Matcher* onde compara todos os 64 *bytes* de uma única vez, não utilizando a metodologia de Alahi et al. para agilizar o processo. Ainda deve-se considerar o volume dos POI por *frame* no tempo de correspondência.

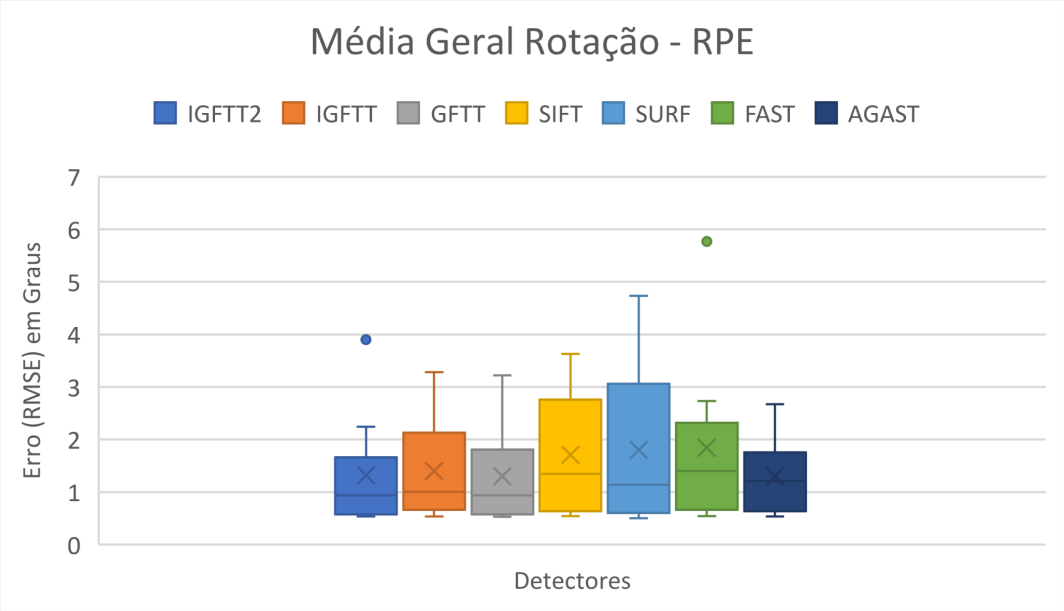

Figura 7.13: *Boxplot* do erro (RMSE) médio da pose relativa rotacional por detector. FONTE: Autor. IGFTT2 obteve a menor dispersão porém, com *outliers*, demonstrando ser estável para maioria das sequências testadas, no entanto, com erro discrepante em alguma sequência. AGAST mostrou ser o mais estável com relação a dispersão dos erros, dentre os algoritmos testados.

O tempo total de processamento gasto por *frame* de acordo com os detectores testados é apresentado na Figura 7.16. Os detectores IGFTT, FAST e AGAST foram os que tiveram melhores desempenhos com uma média de 50 ms, muito próximo ao tempo ideal de 41,66 ms, sendo este o tempo de referência para ocorrer todos os passos de detecção, descrição e correspondência dos POIs. SIFT teve o resultado 2 vezes pior que qualquer outro algoritmo, acima de 200 ms. Vale salientar que em todos os testes de tempo de processamento, os resultados são referentes a média do número de POIs por *frame*, dada a média de cada sequência.

7.3 FASE 3 - Avaliação dos Detectores em Função da Taxa de Repetibilidade e Probabilidade dos Pontos de Interesse

Nesta Fase os detectores são avaliados quanto à taxa de repetibilidade dos POIs dado um conjunto de *frames*, assim como a probabilidade destes aparecerem nos próximos *frames*, uma vez que já foram rastreados no *frame* base.

Estes testes foram feitos utilizando conjuntos de *frames* sequenciais, por toda a trajetória de cada sequência. Cada conjunto pode ser criado por intervalo de tempo ou por número fixo de *frames*. Optou-se em criar o conjunto por número fixo de *frames* porque o tamanho do conjunto criado por tempo varia conforme o *timestamp* de cada *frame*, gerando conjuntos maiores ou

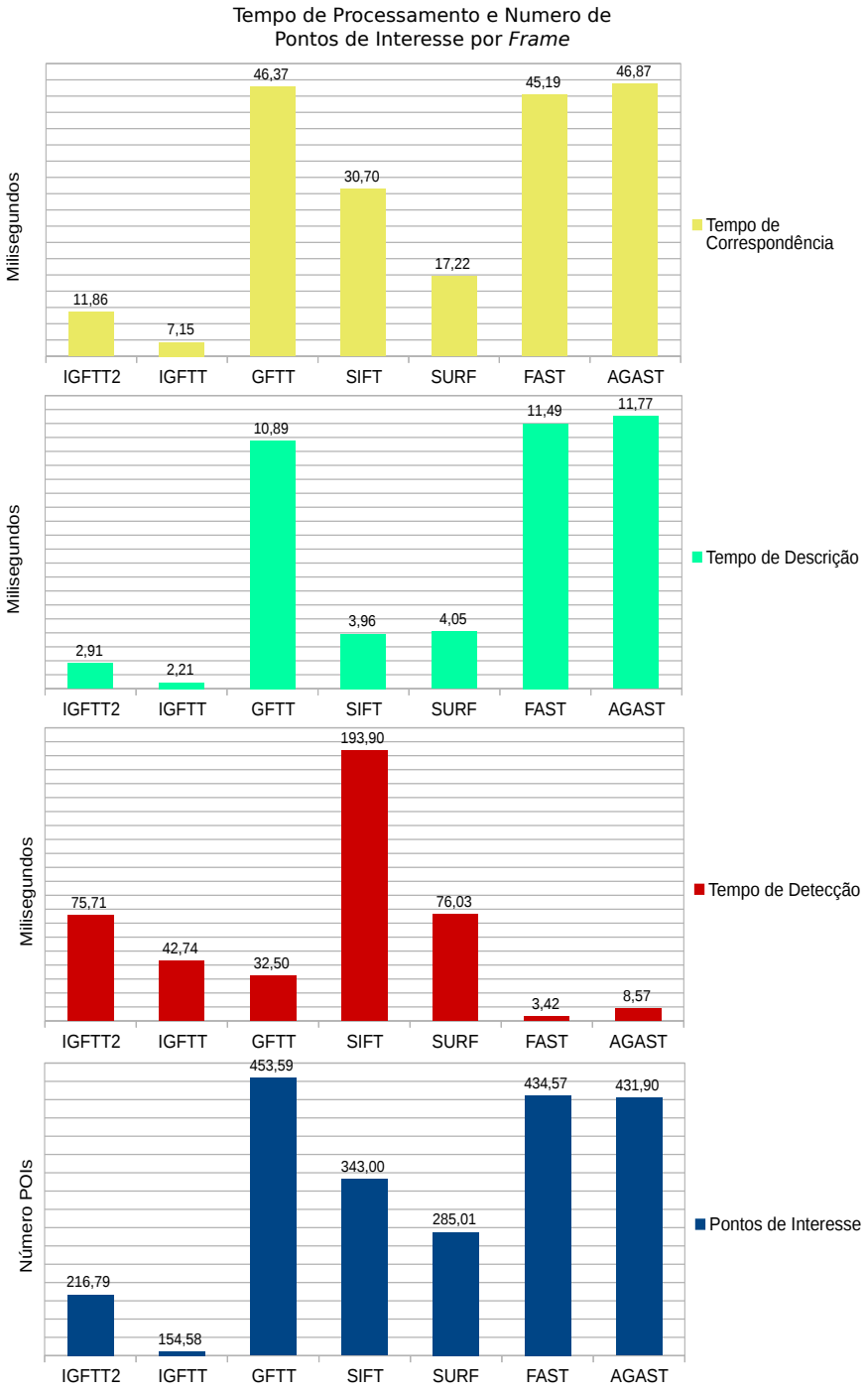

Figura 7.14: Média de tempo de processamento (detecção, descrição e correspondência) e número de POIs por *frame* conforme detector. FONTE: Autor. O algoritmo IGFTT teve melhores resultados em tempo de descrição (barras verdes) e tempo de correspondência (barras amarelas) por detectar somente 154 POIs por *frame* (barras azuis). O GFTT foi quem detectou mais POIs por *frame*. FAST conseguiu agrupar o menor tempo de detecção com o segundo melhor número de POIs por *frame*. IGFTT manteve equilíbrio em todos os tempo e alcançou o melhor resultado quanto ao tempo de processamento por *frame*.

menores, e influenciando na média geral. O conjunto por número fixo de *frames* e a escolha de agrupar cada conjunto com 6 imagens, ocorreu pelo fato deste gerar uma melhor estimativa das

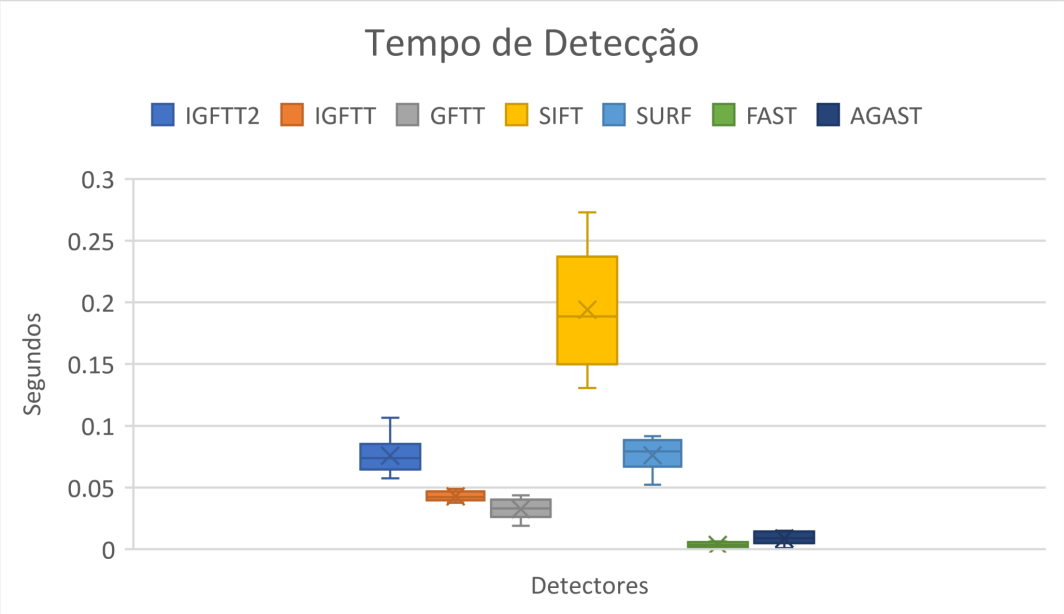

Figura 7.15: *Boxplot* da média do tempo de detecção por *frame* conforme detector. FONTE: Autor. O algoritmo de detecção FAST teve a menor dispersão dos resultados para todas as sequências testadas, se mostrando muito veloz e estável, quanto ao tempo de detecção de POIs. SIFT teve os dados mais dispersos em relação aos detectores testados. O IGFTT, mesmo tendo um resultado pior que o FAST, teve a dispersão bem pequena dos dados, mostrando que é estável também.

taxas de repetibilidade e probabilidade dos POIs, dado que números maiores de *frames* fazem com que POIs não sejam encontrados nas últimas imagens, tendo como consequência uma vasta diminuição da média.

A sequência dos *frames* desse conjunto pode ser configurada conforme ordem de captura dos *frames* pela câmera, organizadas em forma de grafo de cena, ou pela sequência de poses criadas pelo sistema RGBDSLAM, onde este utiliza somente *frames* que estão relacionados conforme a transformação de pose entre eles, e mantém continuidade entre o nó inicial (1º *frame*) e o final (ultimo *frame*). Preferiu-se as sequências de *frames* que estão relacionadas e mantém continuidade, devido ao fato de não ter tantos *frames* fora do menor caminho como o grafo total, mantendo a média geral dos conjuntos alta. O problema da média baixa ocorre quando existe conjuntos de *frames* onde a imagem base não tem POIs identificados, gerando como resultado, 0% de taxa de repetibilidade ou probabilidade em todos os *frames* do conjunto, fazendo com que a média geral da trajetória decresce. Poderia-se suprimir estes conjuntos, porém os valores das médias beneficiariam detectores que identificaram poucos POIs.

A Taxa de Repetibilidade mede o quão robusto é um POI encontrado por um detector, aquele que não muda a localização mesmo que ocorram transformações nas imagens. A Figura 7.17 demonstra a taxa de repetição de POIs conforme seu rastreamento em *frames* subsequentes.

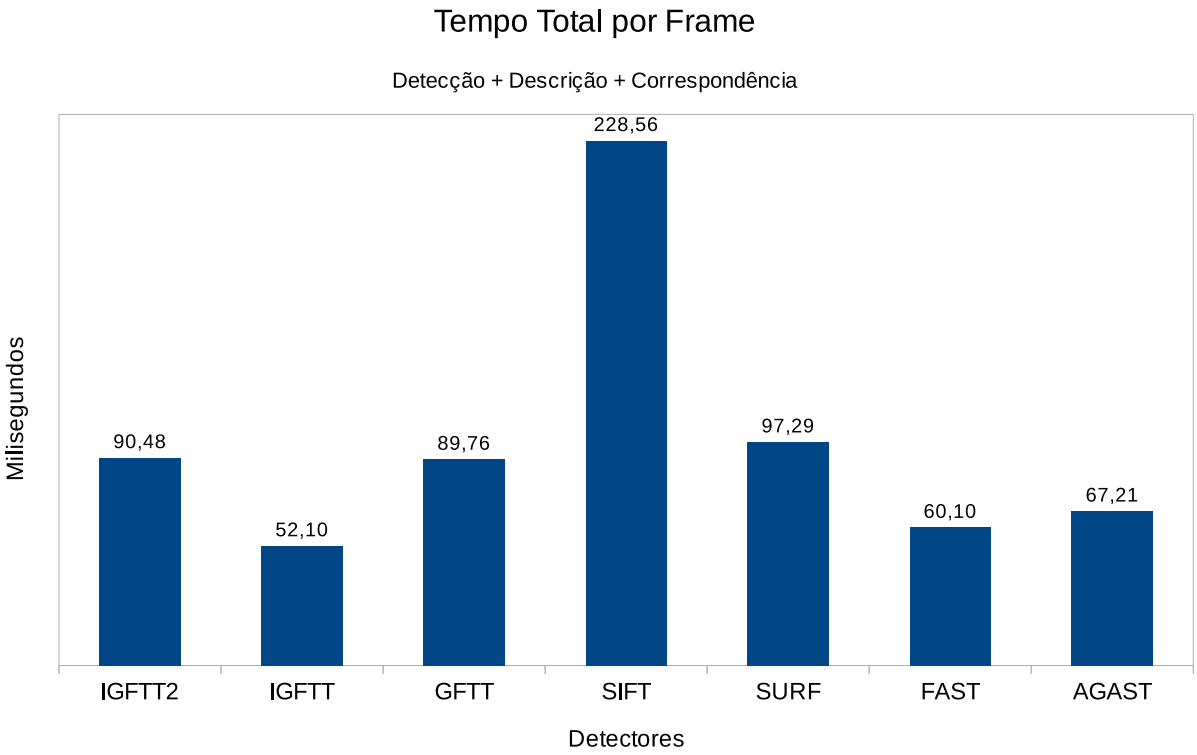

Figura 7.16: Tempo total dos processamentos por *frame* conforme detector. FONTE: Autor. O detector SIFT apresenta o pior desempenho, consumindo duas vezes mais tempo para detectar, descrever e ser correspondido, que qualquer outro detector. IGFTT manteve um equilíbrio entre os tempos chegando ao melhor resultado.

SURF obteve taxas de repetibilidade por frame mais expressivas em relação aos demais detectores. 41% dos pontos do *frame* 1 repetiram no *frame* 3, enquanto que 22% dos POIs detectados no primeiro *frame* repetiram no segundo, para o algoritmo IGFTT. Curvas mais suaves demonstram proporcionalidade no declive de repetibilidade entre os *frames* conforme distanciam-se da base. E curvas mais abruptas, principalmente entre o primeiro e terceiro *frame*, inferem em perca demasiada de POIs. Após o terceiro *frame* a curva se estabiliza indicando que os POIs remanescentes são poucos, mas robustos.

A Figura 7.18 separa melhor os resultados gerais dos testes de repetibilidade e probabilidade, com relação à média de todas as taxas dos conjuntos de 6 *frames*. A média geral do SURF em todas as sequências de *benchmark*, foi de 24,6% para repetibilidade. O IGFTT e IGFTT2 se encontram respectivamente em segundo (11,2%) e terceiro (10,5%) lugar. Vale ressaltar que estas porcentagens não estão normalizadas, de forma que são influenciadas pela média de conjuntos válidos (Figura **??**).

A Figura 7.19 apresenta a probabilidade de sobrevivência de POIs identificados no *frame* base. Se um POI foi rastreado por um número de *frames*, é porque é mais estável, e consequentemente a chance deste aparecer no último *frame* aumenta. Dizemos que um detector é mais estável que outro se este garantir uma maior probabilidade de que POIs já rastreados também serão identificados posteriormente.

Podemos dizer que POIs do SURF são mais prováveis de futuros rastreamentos do que IGFTT e IGFTT2, porque dos POIs que foram rastreados até o *frame* 4, existe uma probabilidade

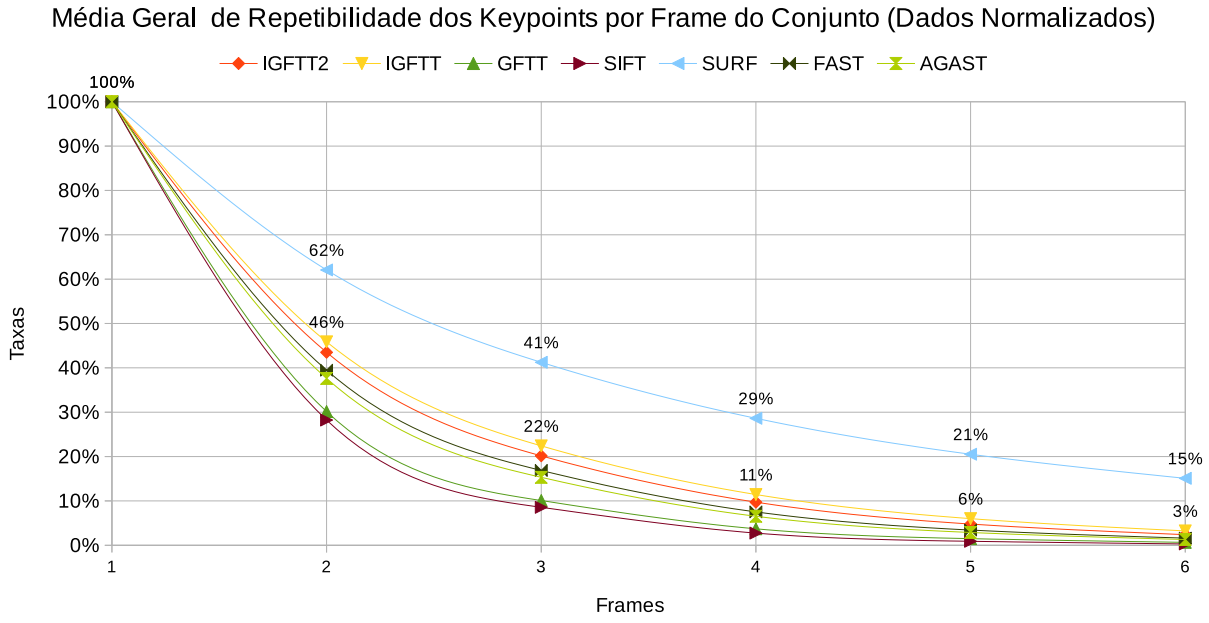

Figura 7.17: Média geral da taxa de repetibilidade dos POIs por *frame* do conjunto. FONTE: Autor. O gráfico apresenta dados normalizados pelo valor máximo. SURF apresenta resultados de repetibilidade de POIs por *frames* ligeiramente melhores que os demais detectores. IGFTT teve o segundo melhor resultado, e SIFT o pior.

de 47% de continuarem sendo rastreados até o último *frame*, enquanto a probabilidade do IGFTT é de 32,2% e IGFTT2 é de somente 30,1%. Visualizando a taxa de probabilidade que os POIs têm de aparecerem no último *frame* dado que foram rastreados no primeiro *frame*, percebemos que o SURF tem 14,4% de chances, 3 vezes mais que o segundo melhor resultado (IGFTT com 4,4%). Ná média geral conforme a Figura 7.18, com 41.1% SURF mostra o melhor desempenho, e com 6.6%, SIFT apresenta o pior. Vale ressaltar que estas porcentagens não estão normalizadas, de forma que são influenciadas pela média de conjuntos válidos (Figura **??**).

Quanto mais suave a curva, mais estável será o detector. Diferente do SURF que teve um resultado bem discrepante em relação à média, os demais tiveram resultados próximos, de forma que a Figura 7.19 não os separa visualmente. Então, para melhor demonstrar os resultados, é calculada a área da curva gerada pelo detector testado em relação à reta ótima para este resultado.

A Figura 7.20 apresenta as áreas de afastamento da reta ideal de cada detector. Vale salientar que um detector estável é reconhecido por ter a menor área entre a curva de probabilidade e a reta ideal. Uma tendência de suavização da curva ocorre quando maior a probabilidade atribuída ao primeiro *frame*. SURF registrou a menor área com 9,93%, e o detector com a maior área foi o AGAST com 19,32%.

Não é interessante um detector ser melhor em relação à repetibilidade de POIs em Visual SLAM se as variações da repetição entre *frames* forem altas demais. Não são precisos muitos POIs para estimar a pose do sensor, mas são necessários bons e constantes POIs. A probabilidade de sobrevivência dos POIs nos fornece uma escolha mais segura sobre um detector, sabendo que este, mesmo detectando poucos POIs, garantirá uma maior probabilidade de identificação contínua e estável destes pontos de interesse em *frames* posteriores. Em Visual SLAM, quanto menor o número de marcos visuais (POIs) incluídos no mapa, menor é o custo computacional do SLAM. Mesmo o SURF detectando poucos POIs (gráfico com barras azuis na Figura 7.14),

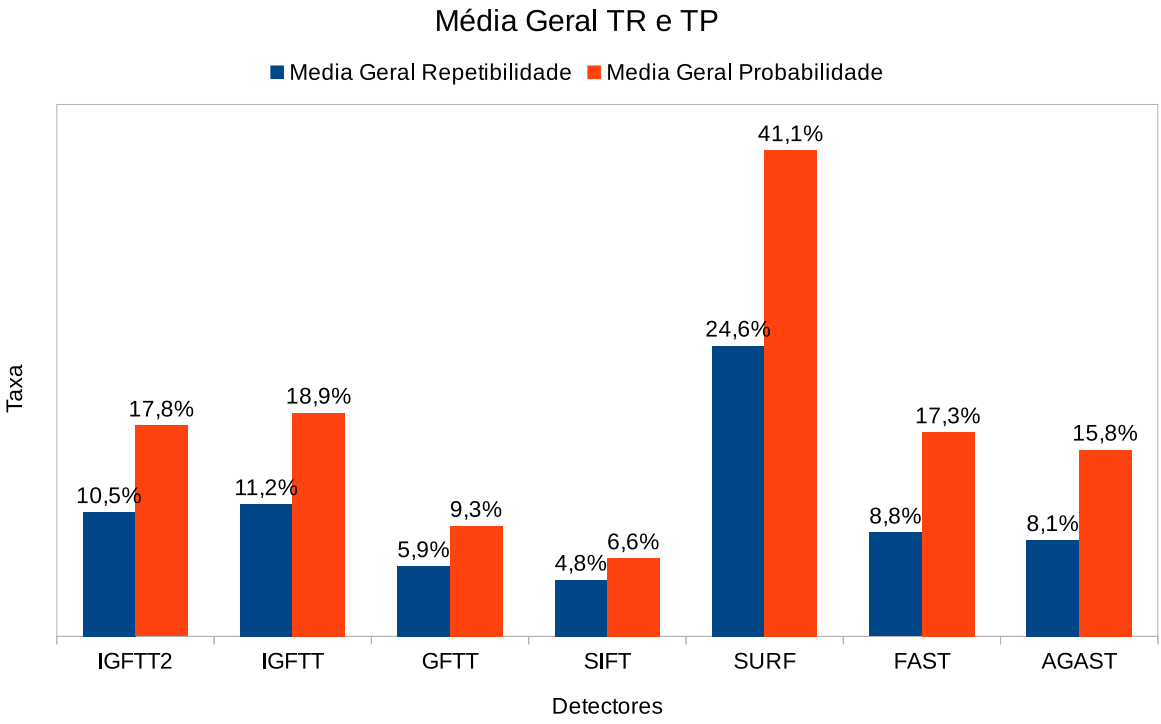

Figura 7.18: Média geral da taxa de repetibilidade e probabilidade dos POIs dado conjunto de 6 *frames*. FONTE: Autor. A média geral, dada pelas médias das taxas dos seis *frames* por conjunto, tanto para o teste de repetibilidade e probabilidade, mostra o SURF com melhor desempenho, e o SIFT com o pior.

sua detecção e descrição são robustas quanto a transformações geométricas e fotométricas nas imagens, garantindo repetibilidade dos POIs (Figura 7.17) e probabilidade alta de rastreamento posterior dos mesmos, facilitando o processamento do sistema de VSLAM

Porém, ao analisar a Figura 7.21 que mostra a dispersão das médias de taxas por sequência, percebe-se que SURF tem valores mais espalhados em relação aos demais detectores, confirmando que ele não é tão estável quanto à repetibilidade dos pontos para todas as sequências, e que por ser especialista em sequências consideradas boas para rastreamento (textura, contraste e pouco movimento), sua média é influenciada por resultados excelentes nestas sequências. De forma inversa, o GFTT teve um desempenho ruim em relação ao SURF, mas teve a menor dispersão demonstrando ser estável de forma geral, em todas as sequências. A Figura 7.22 também mostra que o GFTT é menos disperso, referendando sua estabilidade quanto à probabilidade de prever pontos em frames futuros, em sequências distintas. O FAST teve a maior dispersão das médias.

Destes resultados, percebe-se que à taxa de repetibilidade e probabilidade dos POIs atribuídas ao algoritmo de detecção utilizado é dependente do tipo de transformação que a imagem sofre (translação, rotação, escala, ruído, borramento, etc). Isto sugere uma escolha do melhor algoritmo para determinado tipo de sequência, ou um método genérico que tem resultados bons de forma geral.

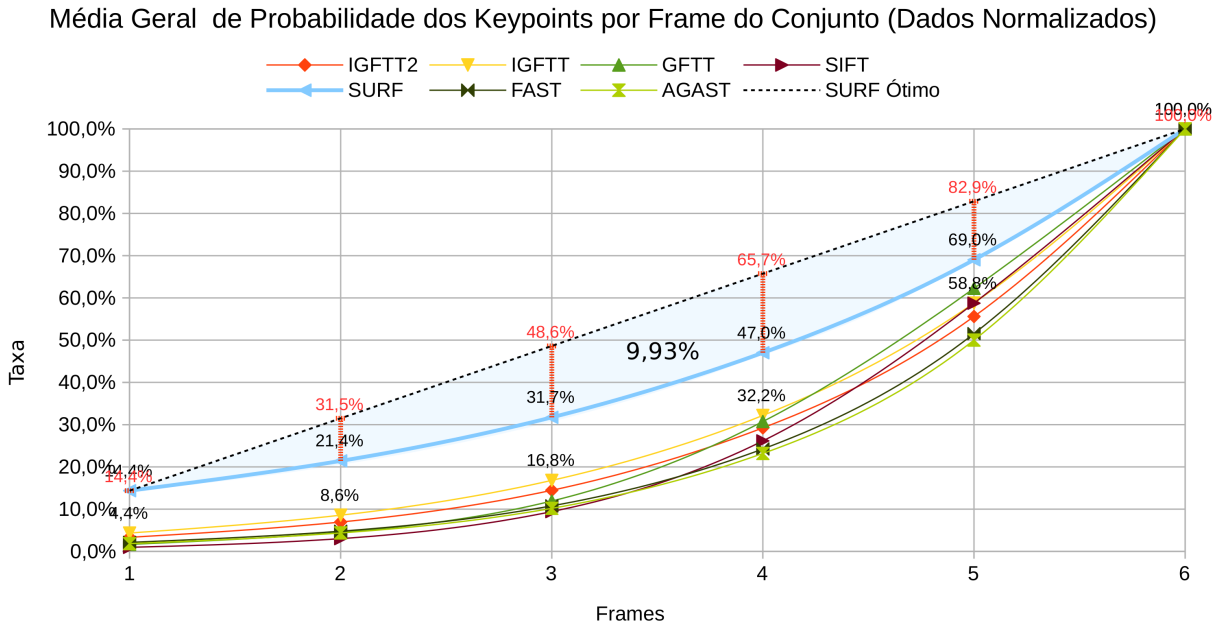

Figura 7.19: Média geral da taxa de probabilidade dos POIs dado conjunto de 6 *frames*. FONTE: Autor. O gráfico apresenta dados normalizados pelo valor máximo. O detector SURF registrou a menor área entre sua curva e a reta ideal, mostrando maior estabilidade em relação aos demais detectores.

7.4 FASE 4 - Avaliação dos Detectores em Função dos Grafos de Cena

Nesta fase, a avaliação dos detectores ocorre em função da sequência dos *frames* registrados pelo grafo de cena, onde são observados o número e a informação da pose dos nós sequenciais.

Há sequências em que muitos *frames* não são aproveitados para estimação da odometria da câmera, desperdiçando tempo de processamento na fase de detecção de POIs e de correspondência, para uma nova tentativa de *matching* entre *frames*.

O grafo da cena é produzido pelo sistema RGBDSLAM. Um *frame* só é adicionado como um nó no grafo de cena se neste foi detectado um mínimo de 20 POIs. Com estes *frames* em sequência, a odometria tenta estimar a pose entre eles, e para tal necessita de no mínimo 5 POIs correspondentes com *frames* anteriores. Como o processamento do *frame* atual, a primeira opção para buscar pontos correspondentes é o *frame* anterior, mas caso não haja, o algoritmo aprofunda a busca em *frames* anteriores conforme um parâmetro de profundidade, fazendo assim com que a odometria não se perca. Desta forma é explicado o porque existem galhos no grafo que não estão dentro da sequência do menor caminho entre o *frame* inicial e o final.

Ainda existem nós dentro da sequência que não têm pose estimada, também desperdiçando tempo e não contribuindo para a odometria. Quando a odometria estima a transformação do *frame* atual dado o anterior, a qualidade da pose é analisada. Um mapa robusto é feito de poses robustas, desta forma, poses que têm taxa de erro acima de um valor de *threshold* não são atribuídas ao nó, mesmo este estando na sequência. Logo, a qualidade da posição dos POIs determina a atribuição das poses nos nós.

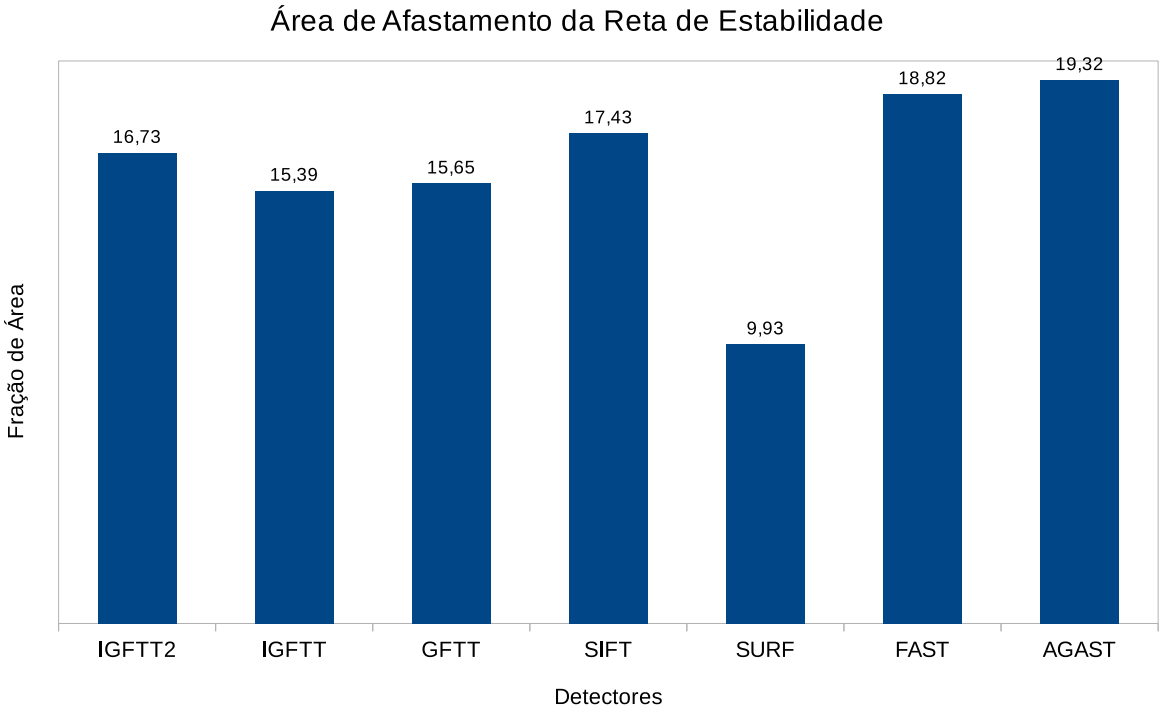

Figura 7.20: Fração da área de afastamento da curva de probabilidade em relação a reta ideal. FONTE: Autor. As áreas de afastamento das curvas, em relação as retas ideais dos detectores, são apresentadas em forma de barras para facilitar a diferenciação entre curvas parecidas. SURF tem seu resultado ligeiramente melhor que os demais. IGFTT e GFTT então quase empatados na segunda colocação. AGAST tem o pior resultado.

Um exemplo pode ser visto na Figura 6.6 onde o os *frames* são representados em forma de grafo dirigido. Então é de interesse mensurar quais detectores promovem um desperdício de tempo menor, sob o ponto de vista do grafo de cena que organiza os nós utilizados em sequência, e também a fração destes nós que não têm pose atribuída.

A Figura 7.23 indica que o IGFTT é o algoritmo que melhor auxilia a odometria, fazendo com que 47,07% dos nós do grafo estejam em sequência. O pior resultado foi de 29,17% conseguido pelo GFTT. Destes nós em sequência, o detector que fez com que um maior número de poses não fossem atribuídas aos *frames*, foi o SIFT em 22,61% dos nós. O melhor em não deixar nós sem pose foi o SURF em 1,33% dos *frames* sequenciais.

A quantidade de nós em sequência ou não, e dos que estão em sequência terem pose ou não, também varia conforme o trajeto de *benchmark*. A Figura 7.24 demonstra que sequências do *benchmark* consideradas boas como a *Desk* e a *Long Office Household* tem a maior porcentagem de nós sequenciais e com a menor taxa de nós sem pose. Pistas que não têm textura, ou são borradas por movimentos bruscos da câmera, auto foco ou auto brilho, têm os piores resultados.

É observado que imagens que sofrem transformações bruscas gerando borramento, têm um número de POIs detectados pequeno, de acordo com cada tipo de detector, influenciando na quantidade e qualidade das correspondências entre *frames*. Assim, a qualidade da pose estimada é prejudicada. Então, *frames* com borramento e o tipo do algoritmo de detecção determinam o número de nós em sequência e o número de nós sem pose nestas sequências.

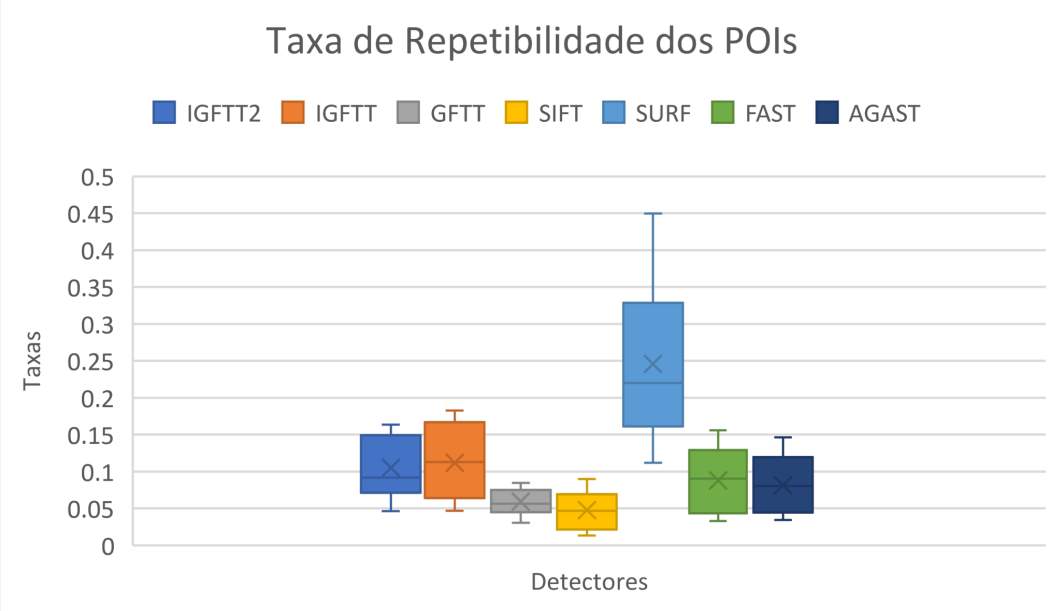

Figura 7.21: *Boxplot* média geral da taxa de repetibilidade dos POIs por detector. FONTE: Autor. O algoritmo SURF, mesmo com melhores taxas de média, apresenta alta dispersão em seus resultados, demonstrando variar conforme o tipo de sequência analisada. IGFTT é o segundo colocado e apresenta uma dispersão bem menor das taxas sob variadas sequências, mostrando ser bem mais estável que o SURF.

7.5 Análise Estatística

Para fins de comparação entre algoritmos de detecção de pontos de interesse, uma avaliação estatística dos resultados do Erro da Trajetória Absoluta (ATE) é feita com o intuito de descobrir se existem diferenças significantes entre as classes de detectores.

As densidades dos valores de erro da trajetória absoluta por detector, podem ser visualizadas na Figura 7.25, onde percebe-se a não normalidade das distribuições dos dados analisados. Desta forma, optou-se por testes estatísticos não paramétricos para avaliar as diferenças entre detectores, onde o método de Friedman foi o escolhido por satisfazer as restrições dos dados analisados, já comentado na Seção 6.2.

O primeiro teste foi realizado para validar ou rejeitar a hipótese nula (H0), onde se quer saber se *"seria idêntica a avaliação do ATE pelos vários métodos de detecção de características, em relação as diversas sequências"*, ou seja, se é possível afirmar que não existem diferenças estatisticamente relevantes entre as classes de detectores. A hipótese alternativa (H1) que afirma que os dados são diferentes é validada caso H0 seja rejeitada. E caso H1 seja validada, pós-testes serão necessários para identificar quais detectores são diferentes de quais.

A Tabela 7.1 apresenta o resultado do teste de Friedman e de Nemenyi (mesmo com H0 validada). O p-valor de 0,1912 é bem maior que o nível de significância de 0,05, aceitando

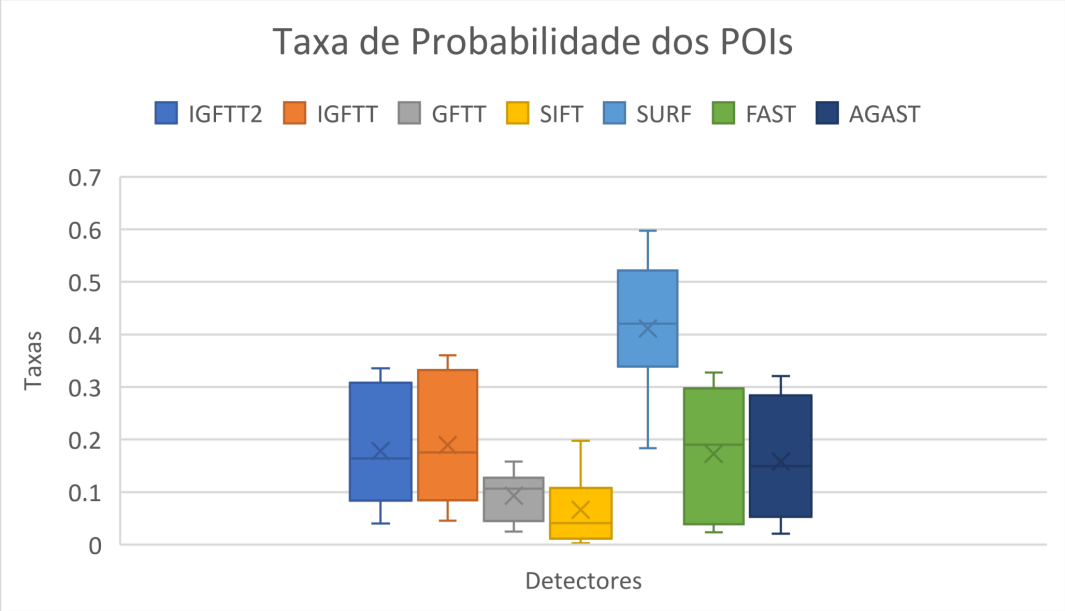

Figura 7.22: *Boxplot* média geral da taxa de probabilidade dos POIs por detector. FONTE: Autor. O algoritmo SURF, além de ter uma taxa melhor, apresenta dispersão inter quantílica menor, com relação aos detectores IGFTT2, IGFTT, FAST e AGAST, mostrando ser mais estável que estes, sob variadas sequências.

a hipótese H0 e afirmando que não existem diferenças significativas entre os detectores de características avaliados.

Não haveria a necessidade do pós-teste de Nemenyi porém, o mesmo foi realizado a fim de comparar as distâncias das médias de *ranking* entre detectores. Conforme a CD (Distância Crítica) de Nemenyi que os detectores devem ter uns dos outros para indicar que são diferentes, nota-se que, os que mais se aproximam da CD de 2,9423 são IGFTT e GFTT, conforme Tabela 7.1 e mostrado na Figura 7.26 entre as linhas pontilhadas, o que confirma o teste de Friedman sobre a igualdade dos detectores.

Um teste mais fino foi realizado a fim de comparar os detectores par a par com um intervalo de significância de 0,05 para aceitar H0 (que não há diferenças significantes entre ambos os detectores). O teste não paramétrico pareado de Wilcoxon foi o escolhido, pois satisfaz as restrições estatísticas das informações avaliadas, conforme comentado na Seção 6.2.

Logo, a Tabela 7.2 apresenta os pares testados com seus respectivos valores. Resultados que rejeitam H0, afirmando que existem diferenças significantes entre ambos os algoritmos testados (valores apresentados em negrito na Tabela). O maior valor de rejeição de H0 foi o apresentado pelos pares SIFT e AGAST, sendo 0,01. IGFTT mostrou-se diferente de GFTT, mesmo um sendo derivado do outro.

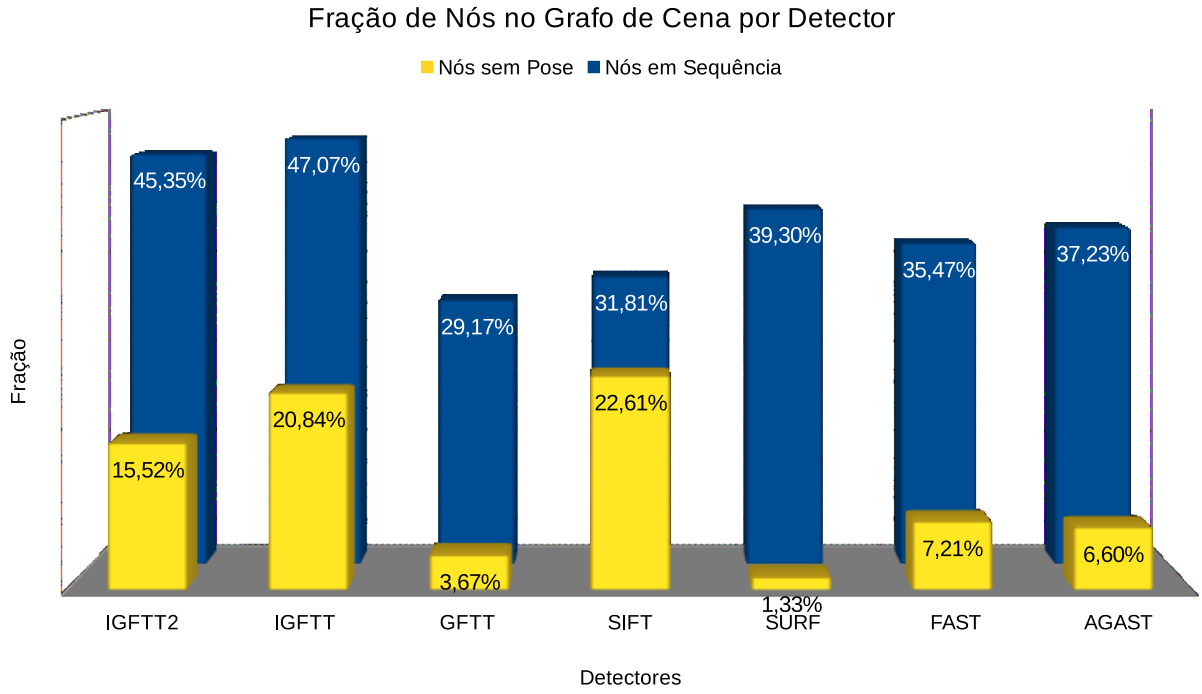

Figura 7.23: Fração de nós no menor caminho do grafo de cena por detector. FONTE: Autor. O melhor detector é aquele que tem a maior fração de nós em sequência, e destes, a menor fração de nós sem pose. SURF tem a melhor relação. A pior relação é alcançada pelo detector SIFT.

7.6 Discussão do Capítulo

Este capítulo descreveu os resultados da metodologia proposta para a avaliação dos detectores. Na Fase 1 apresentou os melhores e piores algoritmos para geração da odometria e do mapa completo. Propôs uma melhora para o algoritmo IGFTT chamado por IGFTT2, onde obteve o segundo lugar do *ranking* geral. Apresentou quais problemas na imagem interferem nos resultados de rotação e translação da câmera. Na Fase 2 demonstrou diferenças de tempo de processamento de cada detector com relação a detecção, descrição e correspondência dos POIs dado a média de POIs por *frame*. Dados POIs detectados, a Fase 3 avaliou-os de acordo com taxas de repetibilidade e a probabilidade destes aparecerem em *frames* futuros, onde o SURF obteve resultados superiores aos demais. A Fase 4 avaliou os algoritmos pelo impacto que estes geram no grafo de cena, criado pelo sistema RGBDSLAM, onde o SURF obteve os melhores resultados. Por fim, demonstrou através de análise estatística, que os algoritmos de detecção de pontos de interesse testados conforme ATE, não são significativamente diferentes quando comparados em grupo. Mas comparados par a par, alguns demostraram diferenças significativas como SIFT e AGAST, e IGFTT que mostrou-se diferente de GFTT, mesmo o primeiro sendo derivado do segundo. No próximo capítulo são apresentadas as conclusões da pesquisa.

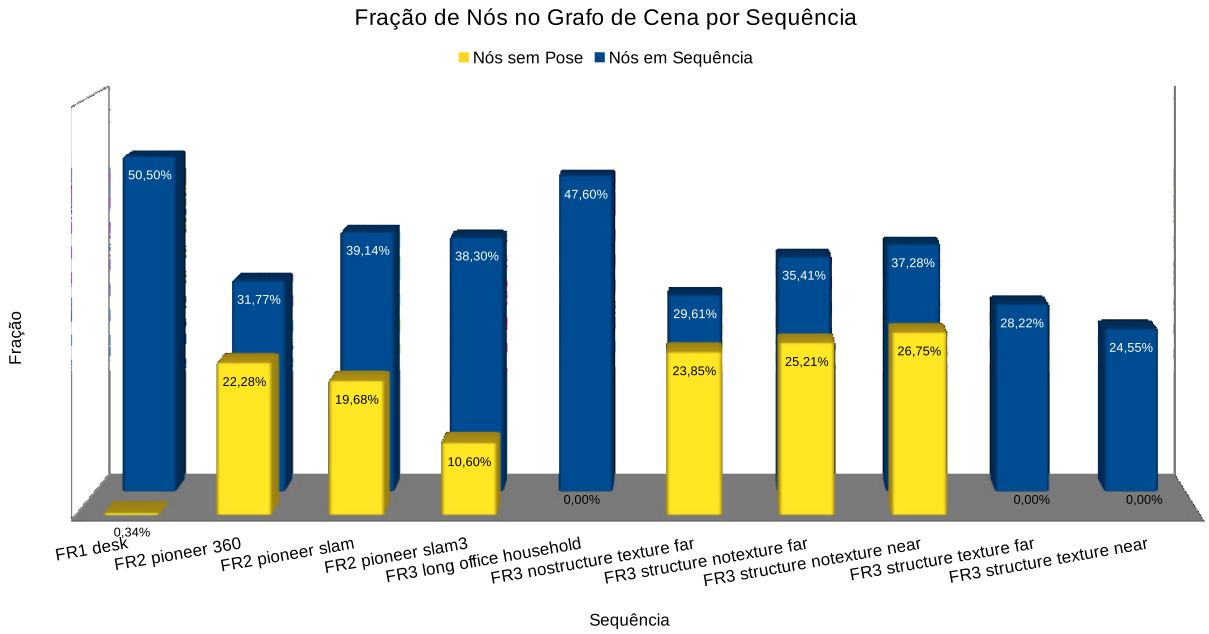

Figura 7.24: Fração de nós no menor caminho do grafo de cena por sequência de *benchmark*. FONTE: Autor. Qualidade das sequências de *benchmark* influenciam também nas características do grafo gerado. Sequências que tem a melhores relações de nós em sequencia, e destes, menores frações de nós sem pose, são respectivamente as sequências *Desk*, *Long Office Household*, *Structure Texture Far* e *Structure Texture Near*. As piores são respectivamente *Nostructure Texture Far*, *Pioneer 360*, *Structure Notexture Far*, *Structure Notexture Near*, *Pioneer Slam* e *Pioneer Slam 3*.

Teste de Friedman (com alfa (significância) = 0,05) distribuído de acordo com Qui-Quadrado, grau de liberdade = 6 ($x^2_F = 8,7$), p-valor = 0,1912. H0 Aceito (Classes Idênticas)

Teste de Nemenyi (com alfa (significância) = 0,05), Distância Crítica (CD) = 2,9423.

	IGFTT2	IGFTT	GFTT	SIFT	SURF	FAST	AGAST
IGFTT2	-	1,10	1,10	0,90	0,10	0,50	0,80
IGFTT	1,10	-	2,20	0,20	1,00	0,60	1,90
GFTT	1,10	2,20	-	2,00	1,20	1,60	0,30
SIFT	0,90	0,20	2,00	-	0,80	0,40	1,70
SURF	0,10	1,00	1,20	0,80	-	0,40	0,90
FAST	0,50	0,60	1,60	0,40	0,40	-	1,30
AGAST	0,80	1,90	0,30	1,70	0,90	1,30	-

Tabela 7.1: Teste estatístico Friedman com pós-teste de Nemenyi. O teste de Friedman valida a hipótese nula de que os detectores são estatisticamente iguais. O teste é feito com intervalo de significância de 0,05. A tabela demonstra que a hipótese nula é aceita, com um $p - valor$ de 0,1912 maior que 0,05, afirmando que os detectores são estatisticamente iguais. O pós-teste de Nemenyi valida quais detectores são diferentes entre si, também com um intervalo de significância de 0,05. Não seria necessário o pós-teste, pois o teste de Friedman ja garante a igualdade estatística. Logo, não é encontrada diferenças no teste de Nemenyi.

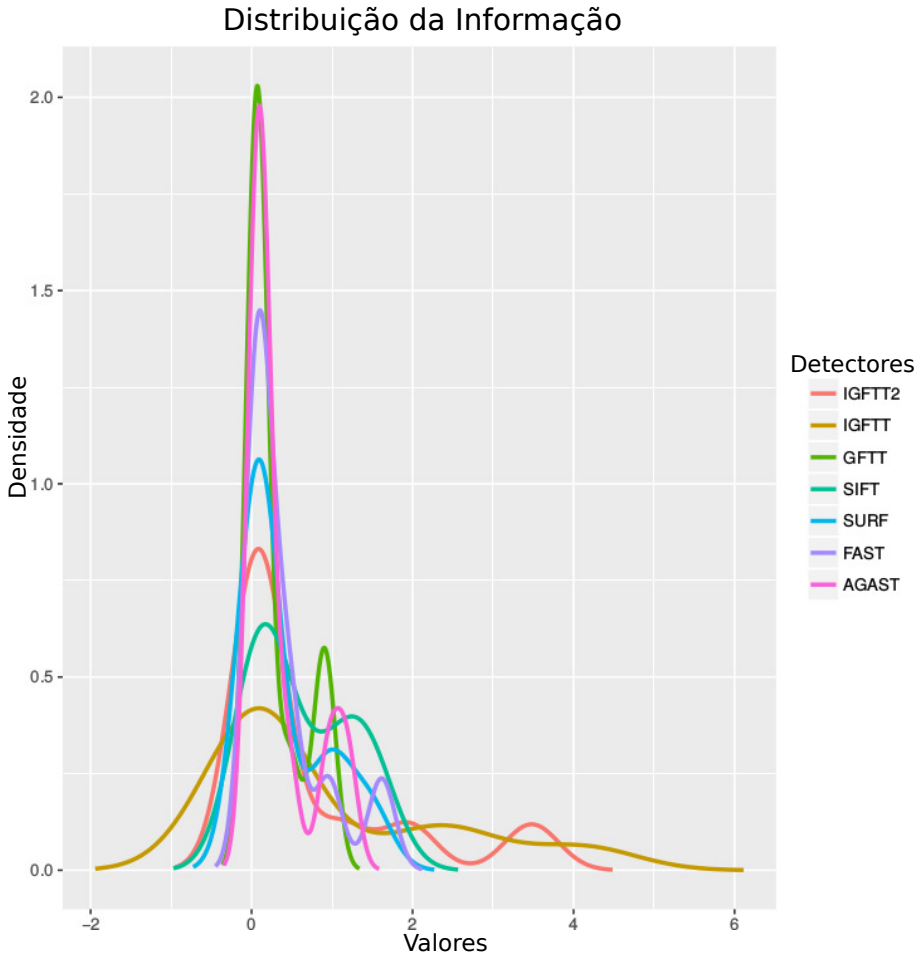

Figura 7.25: Densidade da informação. FONTE: Autor. A imagem mostra as densidades dos valores distribuídos conforme resultados dos testes dos *benchmarks* para cada tipo de detector de POIs. Todas as curvas não apresentam um aspecto de distribuição normal, descartando a opção de análise estatística dos detectores por variância.

Teste de Wilcoxon *Ranking* Assinado (com alfa (significância) = 0,05).							
	IGFTT2	IGFTT	GFTT	SIFT	SURF	FAST	AGAST
IGFTT2	-	0,06	**0,05**	0,32	0,44	0,44	0,44
IGFTT	0,06	-	**0,02**	0,44	0,08	0,17	0,12
GFTT	**0,05**	**0,02**	-	**0,02**	0,08	**0,04**	0,17
SIFT	0,32	0,44	**0,02**	-	**0,04**	0,14	**0,01**
SURF	0,44	0,08	0,08	**0,04**	-	0,29	0,25
FAST	0,44	0,17	**0,04**	0,14	0,29	-	**0,02**
AGAST	0,44	0,12	0,17	**0,01**	0,25	**0,02**	-

Tabela 7.2: Teste estatístico Wilcoxon *ranking* assinado. O teste valida a igualdade estatística de pares de detectores. Resultados que rejeitam H0, afirmando que existem diferenças significantes entre ambos os algoritmos testados, são apresentados em negrito na tabela. O maior valor de rejeição foi atribuída aos pares SIFT e AGAST, sendo 0,01. IGFTT mostrou-se diferente de GFTT (0,02), mesmo um sendo derivado do outro.

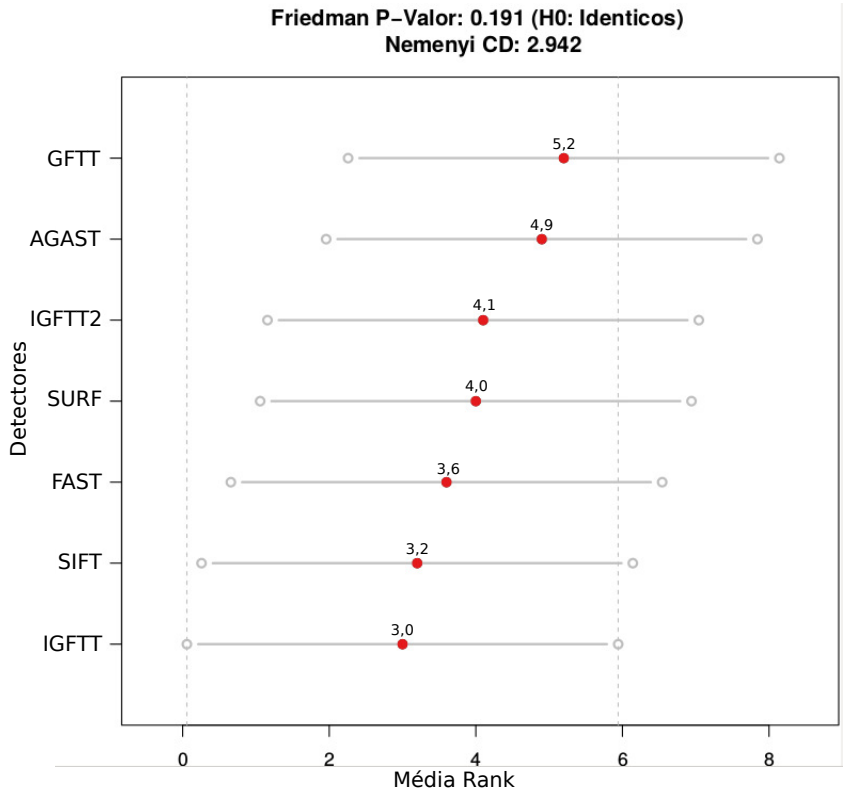

Figura 7.26: Teste de Nemenyi com área de diferença crítica. FONTE: Autor. Qualquer média de *rank* (bolas vermelhas) dos detectores avaliados que estiverem dentro da área de distância crítica (linhas pontilhadas) são consideradas como pertencentes à mesma população.

Capítulo 8

Conclusão

Este capítulo apresenta as contribuições deste trabalho, assim como as dificuldades encontradas e proposta de trabalhos futuros.

8.1 Contribuição

A primeira contribuição deste trabalho foi avaliar o recente algoritmo de detecção de pontos de interesse IGFTT no desempenho do sistema Visual SLAM, em comparação com os detectores já implementados na biblioteca OPENCV (SURF, SIFT, AGAST, FAST, GFTT, ORB, KAZE, AKAZE, MSDDetector, STAR e MSER) que tiveram posições abaixo da média no *ranking* geral dos testes (SURF, SIFT, AGAST, FAST, GFTT). O escore de cada detector no *ranking* geral foi dado pela média aritmética dos *ranks* que este conquistou em cada teste. O IGFTT ficou com a quarta posição no *ranking* geral. Demonstrou ser um detector rápido sendo adequado para ser utilizado em sistemas embarcados, consumindo apenas 52,10 ms para identificar POIs e estes serem descrito e correspondidos entre dois *frames* sequenciais. Entretanto, quanto ao Erro da Trajetória Absoluta (ATE), que define a sua acurácia, obteve o pior resultado com 0,944 mts, entre os detectores analisados. Quanto às taxas de repetibilidade dos POI e a probabilidade de sua detecção em *frames* futuros, IGFTT obteve os segundos melhores resultados sendo respectivamente 11,2% e 18,9%, demonstrando estabilidade em variadas sequências. Em relação ao teste que avalia a porcentagem de nós no grafo de cena, IGFTT alcançou o melhor resultado com 45,35% dos nós em sequência porém, destes nós em sequência, 20,84% têm poses ruins e não foram atribuídas aos nós pelo baixo número de POIs detectados. O teste estatístico de Friedman afirma com nível de significância de 0,05 que não existem diferenças significativas entre os detectores avaliados conforme o Erro da Trajetória Absoluta. Já o teste de Wilcoxon afirma que o algoritmo GFTT é diferente de IGFTT2, IGFTT, SIFT e FAST, que o SIFT é diferente do SURF e AGAST, e que o FAST e AGAST são diferentes também.

A sequência de testes realizados mostrou os pontos fracos no algoritmo IGFTT. Desta forma, a segunda contribuição deste trabalho foi melhorar o desempenho do detector sem alterar suas características peculiares. Agregando acurácia de *subpixel*, reduzindo o tamanho da janela de varredura, escalando a imagem e diminuindo a distância admissível entre os pontos detectados, o IGFTT foi aperfeiçoado, resultando no IGFTT2. Este alcançou o segundo lugar no *ranking* geral, diminuiu o erro de 0,944 metros para 0,712 metros no teste de ATE e ficou com o segundo melhor resultado no teste de escorregamento (*drift*) da odometria na translação, com 0,036 metros, e em terceiro na rotação com 1,320 graus.

Trabalhos que analisam a repetibilidade dos POIs e a probabilidade destes aparecerem em *frames* futuros, dado que foram detectados em *frames* passados, não testam o detector de

pontos de interesse sob ambientes não controlados. Há uma restrição do número de *frames* testados sequencialmente, assim como o tipo de transformações que a imagem sofre. Como terceira contribuição, este trabalho propõe testes quanto às taxas de repetibilidade e probabilidade de detecção de POIs nas sequências de *benchmark*, utilizando para tal, a estrutura do grafo gerado pelo sistema de VSLAM.

Como quarta contribuição, uma visão genérica dos detectores é mostrada por meio de *ranking* dos resultados dos testes específicos realizados na pesquisa. Então, a escolha do melhor detector para ser utilizado em um sistema Visual SLAM, pode ser feita em função das características específicas do ambiente (texturizado, com estruturas) e do robô (propriedades de *hardware* e *software*), ou de forma genérica, aquele detector melhor colocado no *ranking* geral. A Figura 7.1 apresenta em forma de barras o *ranking* dos detectores em todos os testes, e em forma de linhas os resultados dos três primeiros colocados no *ranking* geral. Para o mapeamento de um ambiente desconhecido utilizando um sistema de Visual SLAM, SURF seria a melhor opção para detecção de POI já que obteve a melhor colocação no *ranking* geral, o tornando genérico. Se o robô conhece uma heurística a priori, ou adquire conforme vai conhecendo o ambiente, como por exemplo o *drift* que é um fator crítico para o mapeamento, IGFTT2 seria a melhor opção, já que tem os menores erros de odometria. E em termos de tempo de processamento IGFTT seria a melhor escolha.

8.2 Dificuldades

O RGBDSLAM é um software robusto em relação à qualidade do mapa gerado e fornece praticidade de configurações para testes. Porém, um problema encontrado foi mapear a dependência entre os parâmetros no código, para que se compreendesse o motivo de alguns resultados obtidos. Por exemplo, o entendimento de como o algoritmo contabiliza o tempo de detecção de *frames* que retornavam poucos ou não retornavam POIs.

Houve dificuldades quanto à implementação dos testes de repetibilidade e probabilidade dos pontos de interesse, já que originalmente estes são parametrizados pela quantidade de *frames* na sequência de *benchmark*, e neste trabalho os testes independem do número de *frames* nas sequências. Isto foi possível ser contornado graças à estrutura em forma de grafo dirigido que o sistema RGBDSLAM gera, dados os *frames* da sequência, onde apresenta cada aresta como sendo a transformação de um *frame* em relação ao *frame* anterior.

Definir qual detector é melhor envolve ponderá-lo por suas características ao testá-lo por variados tipos de sequências. Logo, ocorreu o problema de definir uma forma global de avaliação, de forma a classificar os melhores detectores, para posterior avaliação e discussão. A solução foi o *ranking* geral que permitiu a seleção dos melhores detectores.

8.3 Trabalhos Futuros

Dado que o sistema RGBDSLAM recupera a profundidade do ponto através do sensor *Kinect®* e contabiliza somente aqueles POIs que têm profundidade para estimação do movimento, seria interessante testar o detector em função do mapa produzido por um sistema de VSLAM independentemente da nuvem de pontos de profundidade adquiridas por sensores do tipo *Kinect®* ou LIDAR, utilizando somente projeções de pontos de interesse 2D para 3D e estimação de pose por triangulação.

Outra ideia para pesquisa futura é calcular as taxas de repetibilidade e probabilidade em tempo real para inferir, por exemplo em uma curva, qual detector seria melhor, a ponto de

trocá-lo dinamicamente para melhorar a navegação. Ainda, uma estimativa da velocidade do robô poderia ser refinada dadas as taxas de POIs em um conjunto de *frames*. Se um detector é estável, a porcentagem de POIs também será para uma determinada velocidade do robô no ambiente. Logo, uma relação entre a taxa de probabilidade de POIs pela distância entre posições das imagens, poderia ser uma heurística para refinar a estimativa de velocidade do robô.

Referências Bibliográficas

Robin R. Murphy. *Introduction to AI Robotics*. MIT Press, Cambridge, MA, USA, 1st edition, 2000. ISBN 0262133830.

Arturo Gil, Oscar Martinez Mozos, Monica Ballesta, and Oscar Reinoso. A comparative evaluation of interest point detectors and local descriptors for visual SLAM. *Machine Vision and Applications*, 21(6):905–920, 2010. ISSN 1432-1769. doi: 10.1007/s00138-009-0195-x. URL https://doi.org/10.1007/s00138-009-0195-x.

Felix Endres, Juergen Hess, Nikolas Engelhard, Juergen Sturm, Daniel Cremers, and Wolfram Burgard. An evaluation of the RGB-D SLAM system. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, St. Paul, Minnesota, USA, May 2012.

Ícaro De Oliveira, Eduardo Todt, and Keiko Veronica Ono Fonseca. IGFTT: towards an efficient alternative to SIFT and SURF. *WSCG International Conferences in Central Europe on Computer Graphics, Visualization and Computer Vision*, 1, 2015. doi: 10.13140/RG.2.1.1911.3048.

Roland Siegwart, Illah Reza Nourbakhsh, and Davide Scaramuzza. *Introduction to autonomous mobile robots*. The MIT press, 2 edition, 2011.

Cesar Cadena, Luca Carlone, Henry Carrillo, Yasir Latif, Davide Scaramuzza, José Neira, Ian Reid, and John J Leonard. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Transactions on Robotics*, 32(6):1309–1332, 2016.

Randall C. Smith and Peter Cheeseman. On the representation and estimation of spatial uncertainty. *The International Journal of Robotics Research*, 5(4):56–68, 1986. doi: 10.1177/027836498600500404. URL http://dx.doi.org/10.1177/027836498600500404.

F. Lu and E. Milios. Globally consistent range scan alignment for environment mapping. *Autonomous Robots*, 4(4):333–349, Oct 1997. ISSN 1573-7527. doi: 10.1023/A:1008854305733. URL https://doi.org/10.1023/A:1008854305733.

Brent Schwarz. Lidar: Mapping the world in 3d. *Nature Photonics*, 4(7):429–430, 2010. doi: 10.1038/nphoton.2010.148. URL http://dx.doi.org/10.1038/nphoton.2010.148.

Tyler Bell, Beiwen Li, Song Zhang, and John G. Webster. Structured light techniques and applications. 1999. doi: 10.1002/047134608X.W8298. URL http://dx.doi.org/10.1002/047134608X.W8298.

Said Pertuz, Domenec Puig, and Miguel Angel Garcia. Analysis of focus measure operators for shape-from-focus. *Pattern Recognition*, 46(5):1415–1432, 2013. ISSN 0031-3203. doi: http://dx.doi.org/10.1016/j.patcog.2012.11.011. URL http://www.sciencedirect.com/science/article/pii/S0031320312004736.

R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, second edition, 2004. ISBN 0521540518.

Jamie Shotton, Andrew Fitzgibbon, Andrew Blake, Alex Kipman, Mark Finocchio, Bob Moore, and Toby Sharp. Real-time human pose recognition in parts from a single depth image. June 2011. URL https://www.microsoft.com/en-us/research/publication/real-time-human-pose-recognition-in-parts-from-a-single-depth-image/.

J. H. Klüssendorff, J. Hartmann, D. Forouher, and E. Maehle. Graph-based visual SLAM and visual odometry using an RGB-D camera. In *9th International Workshop on Robot Motion and Control*, pages 288–293, July 2013. doi: 10.1109/RoMoCo.2013.6614623.

R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós. ORB-SLAM: A versatile and accurate monocular SLAM system. *IEEE Transactions on Robotics*, 31(5):1147–1163, Oct 2015. ISSN 1552-3098. doi: 10.1109/TRO.2015.2463671.

M. Bigas, E. Cabruja, J. Forest, and J. Salvi. Review of CMOS image sensors. *Microelectronics Journal*, 37(5):433 – 451, 2006. ISSN 0026-2692. doi: http://dx.doi.org/10.1016/j.mejo.2005.07.002. URL http://www.sciencedirect.com/science/article/pii/S0026269205002764.

Eric R Fossum. Active pixel sensors: Are CCDs dinosaurs? In *SPIE's Symposium on Electronic Imaging: Science and Technology*, pages 2–14. International Society for Optics and Photonics, 1993. doi: 10.1117/12.148585. URL http://dx.doi.org/10.1117/12.148585.

Albert JP Theuwissen. CMOS image sensors: State-of-the-art. *Solid-State Electronics*, 52(9): 1401–1406, 2008.

Mark Nixon. *Feature extraction & image processing*. Academic Press, 2 edition, 2008. ISBN 9780080556727.

Aura CONCI. *Computação Gráfica: Teoria e Prática*, volume 2. 2008.

John F Hughes, Andries Van Dam, James D Foley, and Steven K Feiner. *Computer graphics: principles and practice*. Pearson Education, 3 edition, 2014.

Modelo Pinhole. Wikimedia, 2017a. URL https://commons.wikimedia.org/wiki/Category:Pinhole_photography#/media/File:Pinhole-camera.svg. Acessado em 21 de novembro, 2016.

Simon JD Prince. *Computer vision: models, learning, and inference*. Cambridge University Press, 2012.

Richard Szeliski. *Computer vision: algorithms and applications*. Springer Science & Business Media, 2010.

Gary Bradski and Adrian Kaehler. *Learning OpenCV: Computer vision with the OpenCV library*. "O'Reilly Media, Inc.", 2008.

Berthold KP Horn. Tsai's camera calibration method revisited, 2000. URL `http://people.csail.mit.edu/bkph/articles/Tsai_Revisited.pdf`.

R. Tsai. A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses. *IEEE Journal on Robotics and Automation*, 3(4): 323–344, August 1987. ISSN 0882-4967. doi: 10.1109/JRA.1987.1087109.

Z. Zhang. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, Nov 2000. ISSN 0162-8828. doi: 10.1109/34.888718.

MATLAB. *version 7.10.0 (R2010a)*. The MathWorks Inc., Natick, Massachusetts, 2010.

Gary Bradski. The opencv library. *Dr. Dobb's Journal: Software Tools for the Professional Programmer*, 25(11):120–123, 2000.

Maikon Cismoski dos Santos and Anderson Rocha. Revisão de conceitos em projeção, homografia, calibração de câmera, geometria epipolar, mapas de profundidade e varredura de planos. Unicamp, 2012. URL `http://www.ic.unicamp.br/~rocha/teaching/2012s1/mc949/aulas/additional-material-revision-of-concepts-homography-and-related-topics.pdf`.

P. R. S. Mendonca and R. Cipolla. A simple technique for self-calibration. In *Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149)*, volume 1, page 505, 1999. doi: 10.1109/CVPR.1999.786984.

Gang Xu, J. Terai, and Heung-Yeung Shum. A linear algorithm for camera self-calibration, motion and structure recovery for multi-planar scenes from two perspective images. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2000 (Cat. No.PR00662)*, volume 2, pages 474–479, 2000. doi: 10.1109/CVPR.2000.854886.

K. Sirisantisamrid, T. Matsuura, and K. Tirasesth. A simple technique to determine calibration parameters for coplanar camera calibration. In *2004 IEEE Region 10 Conference TENCON 2004.*, volume A, pages 677–680, Nov 2004. doi: 10.1109/TENCON.2004.1414511.

Yisong Chen, Horace Ip, Zhangjin Huang, and Guoping Wang. Full camera calibration from a single view of planar scene. In *Advances in Visual Computing*, pages 815–824. Springer, 2008.

Z. Kukelova, M. Bujnak, and T. Pajdla. Real-time solution to the absolute pose problem with unknown radial distortion and focal length. In *2013 IEEE International Conference on Computer Vision*, pages 2816–2823, Dec 2013.

T.M. Maddock and J.P. Maddock. System and method for automatic camera calibration and alignment determination, jul, 28 2015. URL `https://www.google.ch/patents/US9091662`. US Patent 9,091,662.

C BROWN Duane. Close-range camera calibration. *Photogram. Eng. Remote Sens*, 37:855–866, 1971.

F Fraundorfer and D Scaramuzza. Visual odometry: Part I: The first 30 years and fundamentals. *IEEE Robotics and Automation Magazine*, 18(4):80–92, 2011.

Khalid Yousif, Alireza Bab-Hadiashar, and Reza Hoseinnezhad. An overview to visual odometry and visual SLAM: Applications to mobile robotics. *Intelligent Industrial Systems*, 1(4): 289–311, Dec 2015. ISSN 2199-854X. doi: 10.1007/s40903-015-0032-7. URL https://doi.org/10.1007/s40903-015-0032-7.

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

Alvina Goh and René Vidal. *Algebraic Methods for Direct and Feature Based Registration of Diffusion Tensor Images*, pages 514–525. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006. ISBN 978-3-540-33837-6. doi: 10.1007/11744078$_4$0. URL https://doi.org/10.1007/11744078$_4$0.

Y. Sheikh, A. Hakeem, and M. Shah. On the direct estimation of the fundamental matrix. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–7, June 2007. doi: 10.1109/CVPR.2007.383064.

Seong Kee Park and In So Kweon. Robust and direct estimation of 3D motion and scene depth from stereo image sequences. *Pattern Recognition*, 34(9):1713 – 1728, 2001. ISSN 0031-3203. doi: http://dx.doi.org/10.1016/S0031-3203(00)00104-7. URL http://www.sciencedirect.com/science/article/pii/S0031320300001047.

Michal Irani and P. Anandan. About direct methods. In *Proceedings of the International Workshop on Vision Algorithms: Theory and Practice*, ICCV '99, pages 267–277, London, UK, UK, 2000. Springer-Verlag. ISBN 3-540-67973-1. URL http://dl.acm.org/citation.cfm?id=646271.685624.

Hatem Alismail and Brett Browning. Direct disparity space: Robust and real-time visual odometry. Technical report, 2014.

Gideon P. Stein and Amnon Shashua. Direct estimation of motion and extended scene structure from a moving stereo Rig. Cambridge, MA, USA, 1997. Massachusetts Institute of Technology.

Joachim Heel. Direct estimation of structure and motion from multiple frames. Technical report, Cambridge, MA, USA, 1990.

Tinne Tuytelaars and Krystian Mikolajczyk. Local invariant feature detectors: A survey. *Found. Trends. Comput. Graph. Vis.*, 3(3):177–280, jul 2008. ISSN 1572-2740. doi: 10.1561/0600000017. URL http://dx.doi.org/10.1561/0600000017.

David G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the International Conference on Computer Vision*, volume 2 of *ICCV '99*, pages 1150–, Washington, DC, USA, 1999. IEEE Computer Society. ISBN 0-7695-0164-8. URL http://dl.acm.org/citation.cfm?id=850924.851523.

David G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2):91–110, #nov# 2004. ISSN 0920-5691. doi: 10.1023/B:VISI.0000029664.99615.94. URL http://dx.doi.org/10.1023/B:VISI.0000029664.99615.94.

Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. SURF: Speeded up robust features. In Aleš Leonardis, Horst Bischof, and Axel Pinz, editors, *Computer Vision – ECCV 2006: 9th European Conference on Computer Vision, Graz, Austria, May 7-13, 2006. Proceedings, Part I*, pages 404–417. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006. ISBN 978-3-540-33833-8. doi: 10.1007/11744023$_3$2. URL https://doi.org/10.1007/11744023$_3$2.

Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (SURF). *Computer Vision and Image Understanding*, 110(3):346 – 359, 2008. ISSN 1077-3142. doi: http://dx.doi.org/10.1016/j.cviu.2007.09.014. URL http://www.sciencedirect.com/science/article/pii/S1077314207001555.

P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, volume 1, pages I–511–I–518 vol.1, 2001. doi: 10.1109/CVPR.2001.990517.

Patrice Simard, Léon Bottou, Patrick Haffner, and Yann LeCun. Boxlets: A fast convolution algorithm for signal processing and neural networks. In M. J. Kearns, S. A. Solla, and D. A. Cohn, editors, *Advances in Neural Information Processing Systems 11*, pages 571–577. MIT Press, 1999. URL http://papers.nips.cc/paper/1602-boxlets-a-fast-convolution-algorithm-for-signal-processing-and-neural-networks.pdf.

Jianbo Shi and C. Tomasi. Good features to track. In *1994 Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 593–600, Jun 1994. doi: 10.1109/CVPR.1994.323794.

Bruce D. Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI'81, pages 674–679, San Francisco, CA, USA, 1981. Morgan Kaufmann Publishers Inc. URL http://dl.acm.org/citation.cfm?id=1623264.1623280.

Berthold K.P. Horn and Brian G. Schunck. Determining optical flow. volume 17, pages 185 – 203, 1981. doi: http://dx.doi.org/10.1016/0004-3702(81)90024-2. URL http://www.sciencedirect.com/science/article/pii/0004370281900242.

J. L. Barron, D. J. Fleet, and S. S. Beauchemin. Performance of optical flow techniques. *International Journal of Computer Vision*, 12(1):43–77, Feb 1994. ISSN 1573-1405. doi: 10.1007/BF01420984. URL https://doi.org/10.1007/BF01420984.

A. Alahi, R. Ortiz, and P. Vandergheynst. FREAK: Fast retina keypoint. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 510–517, June 2012. doi: 10.1109/CVPR.2012.6247715.

Edward Rosten and Tom Drummond. Machine learning for high-speed corner detection. In *European Conference on Computer Vision*, volume 1, pages 430–443, May 2006. doi: 10.1007/11744023$_3$4. URL http://www.edwardrosten.com/work/rosten$_2$006$_m$achine.pdf.

Stephen M. Smith and J. Michael Brady. SUSAN—a new approach to low level image processing. *International Journal of Computer Vision*, 23(1):45–78, May 1997. ISSN

1573-1405. doi: 10.1023/A:1007963824710. URL http://dx.doi.org/10.1023/A:1007963824710.

J. R. Quinlan. Induction of decision trees. *Mach. Learn.*, 1(1):81–106, mar 1986. ISSN 0885-6125. doi: 10.1023/A:1022643204877. URL http://dx.doi.org/10.1023/A:1022643204877.

Elmar Mair, Gregory D. Hager, Darius Burschka, Michael Suppa, and Gerhard Hirzinger. Adaptive and generic corner detection based on the accelerated segment test. In *European Conference on Computer Vision (ECCV'10)*, September 2010. URL http://www6.in.tum.de/Main/ResearchAgast.

S. Leutenegger, M. Chli, and R. Y. Siegwart. BRISK: Binary robust invariant scalable keypoints. In *2011 International Conference on Computer Vision*, pages 2548–2555, Nov 2011. doi: 10.1109/ICCV.2011.6126542.

E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. ORB: An efficient alternative to SIFT or SURF. In *2011 International Conference on Computer Vision*, pages 2564–2571, Nov 2011. doi: 10.1109/ICCV.2011.6126544.

Michael Montemerlo, Sebastian Thrun, Daphne Koller, and Ben Wegbreit. FastSLAM: A factored solution to the simultaneous localization and mapping problem. In *Eighteenth National Conference on Artificial Intelligence*, pages 593–598, Menlo Park, CA, USA, 2002. American Association for Artificial Intelligence. ISBN 0-262-51129-0. URL http://dl.acm.org/citation.cfm?id=777092.777184.

Jakob Engel, Thomas Schöps, and Daniel Cremers. *LSD-SLAM: Large-Scale Direct Monocular SLAM*, pages 834–849. Springer International Publishing, Cham, 2014. ISBN 978-3-319-10605-2. doi: 10.1007/978-3-319-10605-2$_5$4. URL https://doi.org/10.1007/978-3-319-10605-2$_5$4.

Christian Kerl, Jürgen Sturm, and Daniel Cremers. Dense visual SLAM for RGB-D cameras. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2100–2106. IEEE, 2013.

G. Silveira, E. Malis, and P. Rives. An efficient direct approach to visual SLAM. *IEEE Transactions on Robotics*, 24(5):969–979, Oct 2008. ISSN 1552-3098. doi: 10.1109/TRO.2008.2004829.

J. Engel, J. Sturm, and D. Cremers. Semi-dense visual odometry for a monocular camera. In *IEEE International Conference on Computer Vision (ICCV)*, Sydney, Australia, December 2013.

C. Forster, M. Pizzoli, and D. Scaramuzza. SVO: Fast semi-direct monocular visual odometry. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 15–22, May 2014. doi: 10.1109/ICRA.2014.6906584.

Raúl Mur-Artal and Juan D Tardós. Probabilistic semi-dense mapping from highly accurate feature-based monocular SLAM. *Proceedings of Robotics: Science and Systems, Rome, Italy*, 1, 2015.

Thomas Lemaire and Simon Lacroix. SLAM with panoramic vision. *Journal of Field Robotics*, 24 (1-2):91–111, 2007. ISSN 1556-4967. doi: 10.1002/rob.20175. URL http://dx.doi.org/10.1002/rob.20175.

Chris Harris and Mike Stephens. A combined corner and edge detector. In *Alvey vision conference*, pages 147–151. Citeseer, 1988.

Matthew Deans. *Bearings-Only Localization and Mapping*. PhD thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, September 2005.

L. M. Paz, P. Piniés, J. D. Tardós, and J. Neira. Large-scale 6-DOF SLAM with stereo-in-hand. *IEEE Transactions on Robotics*, 24(5):946–957, Oct 2008a. ISSN 1552-3098. doi: 10.1109/TRO.2008.2004637.

L. M. Paz, J. D. Tardós, and J. Neira. *Divide and Conquer: EKF SLAM in $O(n)$*. PhD thesis, Oct 2008b.

Andrew J Davison and David W Murray. Simultaneous localization and map-building using active vision. *IEEE transactions on pattern analysis and machine intelligence*, 24(7):865–880, 2002.

Robert Sim, Pantelis Elinas, Matt Griffin, James J Little, et al. Vision-based SLAM using the Rao-Blackwellised particle filter. In *IJCAI Workshop on Reasoning with Uncertainty in Robotics*, volume 14, pages 9–16, 2005.

Arnaud Doucet, Simon Godsill, and Christophe Andrieu. On sequential Monte Carlo sampling methods for Bayesian filtering. *Statistics and computing*, 10(3):197–208, 2000. ISSN 1573-1375. doi: 10.1023/A:1008935410038. URL https://doi.org/10.1023/A:1008935410038.

Andrew J Davison. Real-time simultaneous localisation and mapping with a single camera. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 1403–1410, 2003.

Andrew J Davison, Ian D Reid, Nicholas D Molton, and Olivier Stasse. MonoSLAM: Real-time single camera SLAM. *IEEE transactions on pattern analysis and machine intelligence*, 29(6): 1052–1067, 2007.

Georg Klein and David Murray. Parallel tracking and mapping for small AR workspaces. In *Proc. Sixth IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'07)*, Nara, Japan, November 2007.

Bill Triggs, Philip F. McLauchlan, Richard I. Hartley, and Andrew W. Fitzgibbon. *Bundle Adjustment — A Modern Synthesis*, pages 298–372. Springer Berlin Heidelberg, Berlin, Heidelberg, 2000. ISBN 978-3-540-44480-0. doi: 10.1007/3-540-44480-$7_2$1. URL https://doi.org/10.1007/3-540-44480-$7_2$1.

N. Nain, V. Laxmi, B. Bhadviya, D. B M, and M. Ahmed. Fast feature point detector. In *2008 IEEE International Conference on Signal Image Technology and Internet Based Systems*, pages 301–306, Nov 2008. doi: 10.1109/SITIS.2008.97.

D. Nister. An efficient solution to the five-point relative pose problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(6):756–770, June 2004. ISSN 0162-8828. doi: 10.1109/TPAMI.2004.17.

G Grisetti, H Strasdat, K Konolige, and W Burgard. g2o: A general framework for graph optimization. In *IEEE International Conference on Robotics and Automation*, 2011.

Cordelia Schmid, Roger Mohr, and Christian Bauckhage. Evaluation of interest point detectors. *International Journal of Computer Vision*, 37(2):151–172, Jun 2000. ISSN 1573-1405. doi: 10.1023/A:1008199403446. URL https://doi.org/10.1023/A:1008199403446.

Deepak Dwarakanath, Alexander Eichhorn, Pål Halvorsen, and Carsten Griwodz. Evaluating performance of feature extraction methods for practical 3D imaging systems. In *Proceedings of the 27th Conference on Image and Vision Computing New Zealand*, pages 250–255. ACM, 2012.

Ondrej Miksik and Krystian Mikolajczyk. Evaluation of local detectors and descriptors for fast feature matching. In *Pattern Recognition (ICPR), 2012 21st International Conference on*, pages 2681–2684. IEEE, 2012. ISBN 978-1-4673-2216-4.

Iñigo Barandiaran, John Congote, Jon Goenetxea, and Oscar E Ruiz. Evaluation of interest point detectors for image information extraction. In *Advances in Knowledge-Based and Intelligent Information and Engineering Systems*, volume 243, pages 2170–2179, 2012. doi: 10.3233/978-1-61499-105-2-2170.

Motilal Agrawal, Kurt Konolige, and Morten Rufus Blas. CenSurE: Center surround extremas for realtime feature detection and matching. In David Forsyth, Philip Torr, and Andrew Zisserman, editors, *Computer Vision – ECCV 2008: 10th European Conference on Computer Vision, Marseille, France, October 12-18, 2008, Proceedings, Part IV*, pages 102–115, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg. ISBN 978-3-540-88693-8. doi: 10.1007/978-3-540-88693-8_8. URL https://doi.org/10.1007/978-3-540-88693-8_8.

Pablo Fernandez Alcantarilla, Adrien Bartoli, and Andrew J. Davison. KAZE features. In *Proceedings of the 12th European Conference on Computer Vision - Volume Part VI*, ECCV'12, pages 214–227, Berlin, Heidelberg, 2012. Springer-Verlag. ISBN 978-3-642-33782-6. doi: 10.1007/978-3-642-33783-$3_1$6. URL http://dx.doi.org/10.1007/978-3-642-33783-$3_1$6.

Pablo Fernández Alcantarilla, Jesús Nuevo, and Adrien Bartoli. Fast explicit diffusion for accelerated features in nonlinear scale spaces. In *Conference: British Machine Vision Conference (BMVC)*, 2013.

J. Klippenstein and H. Zhang. Quantitative evaluation of feature extractors for visual SLAM. In *Computer and Robot Vision, 2007. CRV '07. Fourth Canadian Conference on*, pages 157–164, May 2007. doi: 10.1109/CRV.2007.52.

J. Klippenstein and H. Zhang. Performance evaluation of visual SLAM using several feature extractors. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1574–1581, Oct 2009. doi: 10.1109/IROS.2009.5354001.

Krystian Mikolajczyk and Cordelia Schmid. Scale & affine invariant interest point detectors. *International Journal of Computer Vision*, 60(1):63–86, Oct 2004. ISSN 1573-1405. doi: 10.1023/B:VISI.0000027790.02288.f2. URL https://doi.org/10.1023/B:VISI.0000027790.02288.f2.

Timor Kadir, Andrew Zisserman, and Michael Brady. An affine invariant salient region detector. In *European conference on computer vision*, pages 228–241. Springer, Springer-Verlag, 2004.

Jean-Michel Hartmann, Jan Helge Klussendorff, and Erik Maehle. A comparison of feature descriptors for visual SLAM. In *Mobile Robots (ECMR), 2013 European Conference on*, pages 56–61. IEEE, 2013.

Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the KITTI vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of RGB-D SLAM systems. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 573–580, Oct 2012. doi: 10.1109/IROS.2012.6385773.

Óscar Martínez Mozos, Arturo Gil, Monica Ballesta, and Oscar Reinoso. *Interest Point Detectors for Visual SLAM*, pages 170–179. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007. ISBN 978-3-540-75271-4. doi: 10.1007/978-3-540-75271-4$_1$8. URL https://doi.org/10.1007/978-3-540-75271-4$_1$8.

Mónica Ballesta, Arturo Gil, Oscar Martınez Mozos, and Oscar Reinoso. Local descriptors for visual SLAM. In *Workshop on Robotics and Mathematics (ROBOMAT07), Portugal*, pages 209–215. Citeseer, 2007.

K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Van Gool. A comparison of affine region detectors. *International Journal of Computer Vision*, 65(1):43–72, Nov 2005. ISSN 1573-1405. doi: 10.1007/s11263-005-3848-x. URL https://doi.org/10.1007/s11263-005-3848-x.

Precision and Recall. Wikimedia, 2017b. URL https://commons.wikimedia.org/wiki/File:Precisionrecall.svg. Acessado em 28 de novembro, 2016.

Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The KITTI dataset. *International Journal of Robotics Research (IJRR)*, 2013.

M. Smith, I. Baldwin, W. Churchill, R. Paul, and P. Newman. The new college vision and laser data set. *The International Journal of Robotics Research*, 28(5):595–599, May 2009. ISSN 0278-3649. doi: http://dx.doi.org/10.1177/0278364909103911. URL http://www.robots.ox.ac.uk/NewCollegeData/.

Carlos M Jarque and Anil K Bera. Efficient tests for normality, homoscedasticity and serial independence of regression residuals. *Economics letters*, 6(3):255–259, 1980.

Morton B. Brown and Alan B. Forsythe. Robust tests for the equality of variances. *Journal of the American Statistical Association*, 69(346):364–367, 1974. ISSN 01621459. URL http://www.jstor.org/stable/2285659.

Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.*, 7:1–30, dec 2006. ISSN 1532-4435. URL http://dl.acm.org/citation.cfm?id=1248547.1248548.

Salvador García, Alberto Fernández, Julián Luengo, and Francisco Herrera. Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power. *Inf. Sci.*, 180 (10):2044–2064, may 2010. ISSN 0020-0255. doi: 10.1016/j.ins.2009.12.010. URL http://dx.doi.org/10.1016/j.ins.2009.12.010.

Ulrich Nehmzow. *Scientific Methods in Mobile Robotics: quantitative analysis of agent behaviour*. Springer Science & Business Media, 2006. ISBN 978-1-84628-019-1.

Frank Wilcoxon. *Individual Comparisons by Ranking Methods*, pages 196–202. Springer New York, New York, NY, 1992. ISBN 978-1-4612-4380-9. doi: 10.1007/978-1-4612-4380-9$_1$6. URL http://dx.doi.org/10.1007/978-1-4612-4380-9$_1$6.

Lukas Sroba, Rudolf Ravas, and Jan Grman. The influence of subpixel corner detection to determine the camera displacement. *Procedia Engineering*, 100:834 – 840, 2015. ISSN 1877-7058. doi: http://dx.doi.org/10.1016/j.proeng.2015.01.438. URL http://www.sciencedirect.com/science/article/pii/S1877705815004658. 25th DAAAM International Symposium on Intelligent Manufacturing and Automation, 2014.

W. Zhu, C. Ma, L. Xia, and X. Li. A fast and accurate algorithm for chessboard corner detection. In *2009 2nd International Congress on Image and Signal Processing*, pages 1–5, Oct 2009. doi: 10.1109/CISP.2009.5304332.

Hans P. Morevec. Towards automatic visual obstacle avoidance. In *Proceedings of the 5th International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI'77, pages 584–584, San Francisco, CA, USA, 1977. Morgan Kaufmann Publishers Inc. URL http://dl.acm.org/citation.cfm?id=1622943.1622947.

Donovan Parks and Jean-Philippe Gravel. Corner detection. *International Journal of Computer Vision*, 2004.

Tony Lindeberg. Detecting salient blob-like image structures and their scales with a scale-space primal sketch: A method for focus-of-attention. *International Journal of Computer Vision*, 11(3):283–318, Dec 1993. ISSN 1573-1405. doi: 10.1007/BF01469346. URL https://doi.org/10.1007/BF01469346.

Tony Lindeberg. Feature detection with automatic scale selection. *International Journal of Computer Vision*, 30(2):79–116, Nov 1998. ISSN 1573-1405. doi: 10.1023/A:1008045108935. URL https://doi.org/10.1023/A:1008045108935.

Long Quan and Zhongdan Lan. Linear n-point camera pose determination. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(8):774–780, Aug 1999. ISSN 0162-8828. doi: 10.1109/34.784291.

Apêndice A

Detectores de Pontos de Interesse Base

A.1 Moravec

O primeiro algoritmo com intuito de detectar pontos de interesse foi concebido por Moravec[1] Morevec [1977]. Propôs mensurar a variação da intensidade dos *pixels* convolvendo uma janela quadrada de 3×3 *pixels* sob cada *pixel* da imagem principal, e em oito direções diferentes (horizontal, vertical e diagonais). O cálculo da variação da intensidade dada a convolução é feito pela *Sum of Squared Diferences* (SSD) ou soma dos quadrados da diferença. Se a janela analisada encontra-se em um fundo homogêneo da imagem, a variação da intensidade em qualquer direção será muito baixa ou nula (Figura A.1 C). Se a janela estiver analisando uma borda de forma perpendicular, haverá uma grande variação na intensidade em uma direção (Figura A.1 B), porém, se esta janela percorrer a borda de forma paralela, a variação será baixa e também em uma direção (Figura A.1 E). Quando a janela analisa uma borda ou um ponto central, a variação da intensidade ocorrerá em todas as direções e será alta (Figura A.1 A e D).

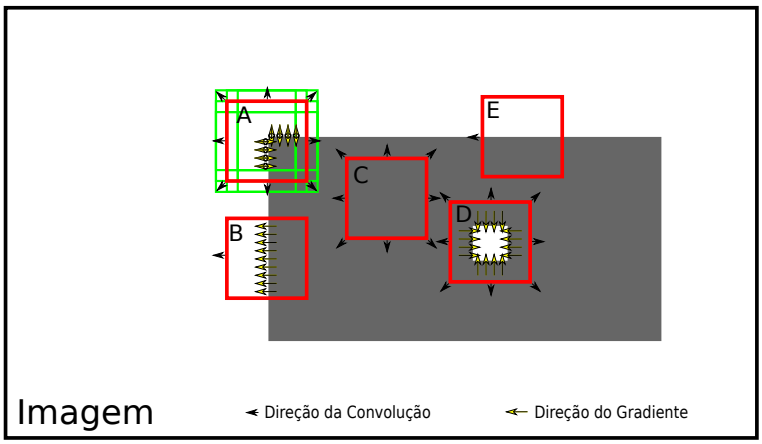

Figura A.1: Detecção de pontos de interesse - Moravec. FONTE: Autor.

Tendo a imagem representada por I, a janela quadrada de convolução centrada em (u, v) e a janela quadrada amostrada da imagem (x, y), a soma do quadrado das diferenças (SSD) entre as janelas é dada por

$$SSD_{u,v}(x, y) = \sum_x \sum_y ((I(x, y)) - I(x + u, y + v))^2 \tag{A.1}$$

[1]Extrator de Características cujo nome faz referência a um de seus criadores, Hans Moravec Morevec [1977]

onde as direções consideradas de (u, v) são $(1, 0)$, $(1, 1)$, $(0, 1)$, $(-1, 1)$, $(-1, 0)$, $(-1, -1)$, $(0, -1)$ e $(1, -1)$.

Para que os pontos de interesse sejam de fato encontrados por este modelo, é necessário fazer um *threshold* para setar os *pixels* não tão intensos como preto (supressão de não máximos), e após, encontrar o máximo local. Assim todos os pontos brancos restantes na imagem resultante serão pontos de interesse.

Problemas de repetibilidade de pontos de interesse ocorrem no método de Moravec. Por analisar a variação de intensidade somente em 8 direções, qualquer canto ou borda que não estejam em 45° são descartadas. Ignora qualquer ponto que não seja um máximo mesmo sendo um bom ponto de interesse. Por ter janela binária, *pixels* vizinhos próximos do *pixel* central da janela, tem mesmo peso de *pixels* distantes, tornando a resposta do algoritmo sensível a ruídos.

A.2 Harris

Para corrigir problemas no método de Moravec, Harris and Stephens [1988] desenvolveram o detector de pontos de interesse Harris. Para remover a limitação de invariância a rotação, é necessário uma função matemática que permita o cálculo da variação de intensidade em todas as direções. Para tal, uma expansão analítica do operador de Moravec é realizada para derivar a função. Esta função melhora significativamente a detecção e repetibilidade dos POIs porém incrementa o custo computacional [Parks and Gravel, 2004].

Harris utiliza um filtro para calcular o gradiente de forma aproximada, similar ao operador de Moravec. Dado A_i como janela quadrada da imagem e B_i a janela de convolução, pode-se expressar a equação que calcula a intensidade nas direções horizontal, vertical, e diagonal conforme segue:

$$
\begin{aligned}
SSD(horizontal) &= \sum_{i=1}^{9}(A_i - B_i)^2 = \sum_{i=1}^{9}(B_i - A_i)^2 \approx \sum_{i=1}^{9}\left(\frac{\partial I_i}{\partial x}\right)^2 \\
\text{onde: } \frac{\partial I_i}{\partial x} &\equiv I_i \otimes (-1, 0, 1) \approx B_i - A_i \\
SSD(vertical) &= \sum_{i=1}^{9}(A_i - B_i)^2 = \sum_{i=1}^{9}(B_i - A_i)^2 \approx \sum_{i=1}^{9}\left(\frac{\partial I_i}{\partial y}\right)^2 \\
\text{onde: } \frac{\partial I_i}{\partial y} &\equiv I_i \otimes (-1, 0, 1)^T \approx B_i - A_i \\
SSD(diagonal) &= \sum_{i=1}^{9}(A_i - B_i)^2 = \sum_{i=1}^{9}(B_i - A_i)^2 \approx \sum_{i=1}^{9}\left(\frac{\partial I_i}{\partial d}\right)^2 \\
\text{onde: } \frac{\partial I_i}{\partial d} &\equiv I_i \otimes \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{bmatrix} \approx B_i - A_i \\
\otimes &= \text{convolução}
\end{aligned}
\tag{A.2}
$$

Logo, a intensidade de variação pode ser reescrita como sendo uma função do gradiente da imagem, sendo

$$
SSD_{u,v}(x, y) = \sum_{\forall i \text{ centrado em } (x,y)} \left(u\frac{\partial I_i}{\partial x} + v\frac{\partial I_i}{\partial y}\right)^2 \tag{A.3}
$$

onde u e v representam a direção da convolução (exemplo horizontal = (u,v) = (1,0)).

Assim, pode-se aproximar a intensidade da variação em qualquer direção somente alterando valores para (u,v). Ainda, pesos são acrescentados em cada pixel da janela de acordo com uma distribuição Gaussiana, incrementando a equação acima como

$$SSD_{u,v}(x,y) = \sum_{\forall i \text{ centrado em } (x,y)} \phi_i \left(u\frac{\partial I_i}{\partial x} + v\frac{\partial I_i}{\partial y} \right)^2 \tag{A.4}$$

onde ϕ_i é o peso da janela Gaussiana na posição i.

Reescrevendo a equação acima, temos

$$\begin{aligned} SSD_{u,v}(x,y) &= \sum_{\forall i \text{ centrado em } (x,y)} \phi_i \left(u^2\frac{\partial I_i^2}{\partial x} + 2uv\frac{\partial I_i}{\partial x}\frac{\partial I_i}{\partial y} + v^2\frac{\partial I_i^2}{\partial y} \right) \\ &= Au^2 + 2Cuv + Bv^2 \\ \text{onde: } A &= \left(\frac{\partial I}{\partial x}\right)^2 \otimes \phi, \\ B &= \left(\frac{\partial I}{\partial y}\right)^2 \otimes \phi, \\ C &= \left(\frac{\partial I}{\partial x}\frac{\partial I}{\partial y}\right)^2 \otimes \phi \end{aligned} \tag{A.5}$$

De forma matricial, temos

$$\begin{aligned} SSD_{u,v}(x,y) &= Au^2 + 2Cuv + Bv^2 \\ &= \begin{bmatrix} u & v \end{bmatrix} M \begin{bmatrix} u \\ v \end{bmatrix} \\ \text{onde: } M &= \begin{bmatrix} A & C \\ C & B \end{bmatrix} \end{aligned} \tag{A.6}$$

Desta forma, a matriz M guarda todos os operadores diferenciais que descrevem a geometria da janela amostrada da imagem no ponto (x, y). Os autovalores μ da matriz M representam as variações de intensidades formando a curvatura da imagem amostrada, além de descrever a rotação desta. Porém, se os componentes de M forem estimados somente medindo as variações de intensidade na horizontal e vertical, os autovalores não serão invariantes a rotação.

Uma região de canto, borda, interna ou isolada é reconhecida por meio dos autovalores. Se um autovalor for baixo e outro alto significa que há uma borda onde o autovalor alto representa uma alta curvatura perpendicular à borda, e o autovalor menor uma pequena curvatura paralela a esta. Se ambos forem pequenos, representa uma região interna onde há uma pequena variação na intensidade ou nenhuma e, se ambos tiverem um valor alto, significa que há uma curvatura alta em ambas as direções e estas representam um canto ou ponto isolado [Harris and Stephens, 1988]. A imagem a seguir demonstra as regiões referentes aos autovalores obtidos da matriz M.

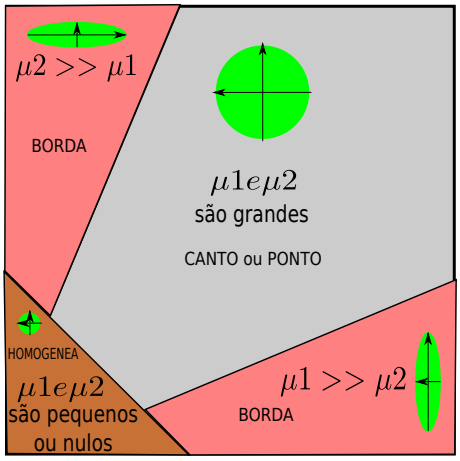

Figura A.2: Representação de pontos de interesse - Harris. FONTE: Inspirado de [Harris and Stephens, 1988, p. 150].

Para afirmar que o pixel analisado é um canto, Harris and Stephens [1988] definem a Equação A.7. Atribuindo a representação dos autovalores da matriz M como μ_1 e μ_2, temos

$$\begin{aligned} C(x,y) &= det(M) - k(tr(M))^2 \\ det(M) &= \mu_1\mu_2 = AB - C^2 \\ tr(M) &= \mu_1 + \mu_2 = A + B \\ k &= constante \end{aligned} \tag{A.7}$$

onde k é um parâmetro que, empiricamente, reportado na literatura, varia entre 0.04 a 0.15 [Prince, 2012]. $det(M)$ e $tr(M)$ são respectivamente a determinante e o traço da matriz (M). $C(x,y)$ é a resposta positiva para um canto, negativa para borda e pequena para regiões homogêneas.

Após é realizada a supressão de não máximos para restar somente o canto.

Mesmo com variações baixas de iluminação e com rotação, o algoritmo de Harris tem uma alta repetibilidade de pontos de interesse. Harris não é invariante a transformações de escala e afim na imagem, de forma que, a medida que imagens variam a escala, a repetibilidade diminui e, mudanças afim distorcem os *pixels* da região analisada nas direções x e y de forma a incrementar ou decrementar sua curvatura. Modificações do algoritmo para correção dos problemas descritos acima podem ser encontrados em Mikolajczyk and Schmid [2004], Lindeberg [1993, 1998].

A.3 GFTT

Dada duas imagens I, uma posição qualquer $I(x,y,t)$ do *frame* 1 no tempo t, pode ser prevista no *frame* 2 no tempo τ transladando a posição inicial de $I(x,y)$ por vetores (u,v). Devido premissa de que o deslocamento entre *frames* deve ser pequeno, assume-se que $\tau = 1$. A Equação A.8 descreve o modelo do fluxo ótico e a Figura A.3 demonstra a ideia de previsão do ponto.

$$I(x,y,t) = I(x+u, y+v, t+\tau) \tag{A.8}$$

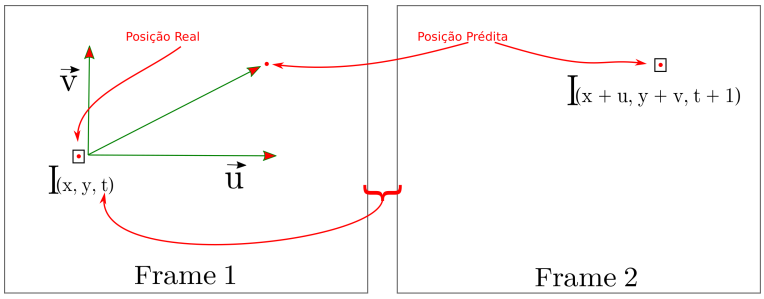

Figura A.3: Fluxo Ótico - GFTT. FONTE: Autor.

Aplicando a expansão de Taylor de primeira ordem na Equação A.8, obtemos

$$I(x, y, t) = I(x, y, t) + \frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v + \frac{\partial I}{\partial t}\tau \tag{A.9}$$

Eliminando os termos I(x, y, t), resta

$$0 = \frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v + \frac{\partial I}{\partial t}\tau \tag{A.10}$$

e dividindo todos os termos por τ, obtemos

$$\frac{\partial I}{\partial x}\frac{u}{\tau} + \frac{\partial I}{\partial y}\frac{v}{\tau} + \frac{I_t}{\tau} = 0 \tag{A.11}$$

onde I_t é a derivada de I em relação a τ.

Logo, os dois componentes $\frac{u}{\tau}$ e $\frac{v}{\tau}$ do vetor velocidade $\vec{v}$ são os termos procurados e, $\frac{\partial I}{\partial x}$ e $\frac{\partial I}{\partial y}$ são denominados respectivamente gradiente do ponto nas direções x e y ou ∇I. Então a equação de restrição do fluxo ótico é dado como:

$$\nabla I \times \vec{v} + I_t = 0 \tag{A.12}$$

O problema de Abertura surge quando se analisa apenas um *pixel* (não se sabe a origem ou destino do gradiente) ou quando a direção do gradiente é perpendicular ao deslocamento da janela na imagem (mesmo havendo gradiente, não há mudanças de intensidade e direção dentro da janela), se tornando impossível estimar o fluxo.

O cálculo do fluxo de um *pixel* é dado em relação a seus *pixels* vizinhos, onde utiliza-se geralmente uma janela Gaussiana. Como a imagem dos *frames* tem muito ruido, a restrição do fluxo ótico é dada como uma minimização SSD do problema de encontrar u e v sendo:

$$SSD_{u,v}(\sum_{\Phi}[I(x + u, y + v, t + \tau) - I(x, y, t)]^2) \tag{A.13}$$

onde Φ é a janela Gaussiana.

Expandindo a Equação A.13 por Taylor, obtêm-se uma equação linear do tipo $Ax = b$ que de forma matricial resulta em

$$\begin{aligned} A &= \sum_{\Phi} \begin{bmatrix} I_u^2 & I_u I_v \\ I_u I_v & I_v^2 \end{bmatrix} \\ x &= \begin{bmatrix} u \\ v \end{bmatrix} \\ b &= \tau \sum_{\Phi} I_t \begin{bmatrix} I_u & I_v \end{bmatrix}^T \end{aligned} \tag{A.14}$$

onde $[I_u I_v] = \frac{\partial I}{\partial x}$ e $\frac{\partial I}{\partial y} = \nabla I$.

Dado que x é a solução do problema, temos a Equação A.14 final que prevê onde estará o ponto de interesse no próximo *frame* sendo $x = A^{-1}b$.

Dado que a matriz A é uma matriz Hessiana, de forma semelhante ao método de Harris (vide Apêndice A.2) de caracterizar um ponto de interesse, Shi and Tomasi baseiam-se também nos autovalores μ_1 e μ_2 da matriz A.

Apêndice B

Matriz Essencial, Matriz Fundamental e Profundidade do POI

B.1 Matriz Essencial

Pode-se relacionar duas câmeras matematicamente. Prince [2012] interpreta e descreve o modelo matemático da forma que será apresentada a seguir. Um sistema de coordenadas está posicionado sobre uma câmera $C1$ e seus parâmetros extrínsecos são $\{Id, O\}$ (Id é matriz identidade e O é vetor nulo). Outra câmera $C2$ está em qualquer posição $\{\Omega, \tau\}$ (Ω é matriz de rotação e τ é vetor de translação). Assume-se que as câmeras estejam normalizadas $\Lambda 1 = \Lambda 2 = Id$ (Λ é uma matriz contendo os parâmetros intrínsecos da câmera). Em coordenadas homogêneas, um POI tridimensional $\tilde{Q}$ é projetado dentro de duas câmeras como

$$\begin{aligned} \gamma_1 \tilde{q}_1 &= [Id, O]\tilde{Q} \\ \gamma_2 \tilde{q}_2 &= [\Omega, \tau]\tilde{Q}. \end{aligned} \tag{B.1}$$

onde $\tilde{q}1$ e $\tilde{q}2$ são as posições observadas nas respectivas câmeras com coordenadas homogêneas e γ é fator de escala.

Expandindo a primeira relação, temos

$$\gamma_1 \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} u \\ v \\ w \\ 1 \end{bmatrix} = \begin{bmatrix} u \\ v \\ w \end{bmatrix} \tag{B.2}$$

Simplificadamente, temos

$$\begin{aligned} \gamma_1 \tilde{q}_1 &= \tilde{Q} \\ \gamma_2 \tilde{q}_2 &= \Omega\tilde{Q} + \tau. \end{aligned} \tag{B.3}$$

Substituindo a primeira relação dentro da segunda,

$$\gamma_2 \tilde{q}_2 = \gamma_1 \Omega \tilde{q}_1 + \tau. \tag{B.4}$$

Prince [2012] informa que esta relação representa uma restrição entre possíveis posições de correspondência entre pontos $x1$ e $x2$ nas duas imagens e são parametrizadas pela rotação e translação $\{\Omega, \tau\}$ da câmera $C2$ relativa a $C1$. Ainda manipulando a Equação B.4, há uma

forma mais elegante de expressar as linhas epipolares e os epipolos. Primeiro é feito um produto cruzado do vetor de translação τ

$$\gamma_2 \tau \times \tilde{q}_2 = \gamma_1 \tau \times \Omega \tilde{q}_1 \tag{B.5}$$

Após é feito um produto interno por $\tilde{q}_2$ dos dois lados da igualdade. O lado esquerdo desaparecerá desde que $\tau \times \tilde{q}_2$ seja perpendicular a $\tilde{q}_2$, ficando

$$\tilde{q}_2^T \tau \times \Omega \tilde{q}_1 = 0 \tag{B.6}$$

onde os fatores de escala γ foram eliminados.

Finalmente, pode-se expressar o produto cruzado de τ por uma matriz 3 x 3 antissimétrica como

$$\tau\times = \begin{bmatrix} 0 & -\tau_z & \tau_y \\ \tau_z & 0 & -\tau_x \\ -\tau_y & \tau_x & 0 \end{bmatrix} \tag{B.7}$$

A Equação B.6 adquire a forma de

$$\tilde{q}_2^T E \tilde{q}_1 = 0 \tag{B.8}$$

onde $E = \tau \times \Omega$ é conhecida como Matriz Essencial, e esta captura a relação geométrica entre duas câmeras.

Assim, as linhas epipolares podem ser recuperadas facilmente por meio da matriz essencial. A condição para que um ponto esteja sobre uma linha epipolar é expressa por $ax + by + c = 0$ ou

$$\begin{bmatrix} a & b & c \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = 0 \tag{B.9}$$

Em coordenadas homogêneas esta equação pode ser escrita como $l\tilde{q} = 0$ em que $l = [a\ b\ c]$ é vetor 1 x 3 que representa uma linha. Considerando a Equação B.8, desde que $\tilde{q}_2^T E$ seja vetor 1 x 3, pode-se representá-la como $l_1 \tilde{q}_1 = 0$. Logo, a linha $l_1 = \tilde{q}_2^T E$ é uma linha epipolar na imagem 1 dado o ponto $x2$ na imagem 2, e de forma similar, podemos encontrar a linha epipolar l_2 na imagem 2 dado o ponto $x1$ na imagem 1. A relação final é expressa então por

$$\begin{aligned} l_1 &= \tilde{q}_2^T E \\ l_2 &= \tilde{q}_1^T E^T \end{aligned} \tag{B.10}$$

Segundo Bradski and Kaehler [2008], é válido afirmar que a matriz essencial é puramente geométrica e representa um mapeamento da posição física de um POI A visto por uma câmera, para a posição do mesmo POI, vista por outra câmera. Existe ainda a necessidade de relacionar a posição do POI observado por cada câmera, com as coordenadas nas imagens (em *pixels*). Para tal, é obrigatório estabelecer a relação da posição física do POI observado com os parâmetros intrínsecos de cada câmera. A esta relação dá-se o nome de Matriz Fundamental.

B.2 Matriz Fundamental

A matriz essencial utiliza em seus cálculos o modelo de câmeras normalizadas onde não há parâmetros intrínsecos relacionados. A matriz fundamental acrescenta os parâmetros intrínsecos aos cálculos da matriz essencial como

$$\begin{aligned} \gamma_1 \tilde{q}_1 &= \Lambda_1 [Id, O] \tilde{Q} \\ \gamma_2 \tilde{q}_2 &= \Lambda_2 [\Omega, \tau] \tilde{Q} \end{aligned} \tag{B.11}$$

onde Λ_1 e Λ_2 são os parâmetros intrínsecos das câmeras.

Manipulando as equações da matriz fundamental, chega-se as restrições

$$\tilde{q}_2^T \Lambda_2^{-T} E \Lambda_1^{-1} \tilde{q}_1 = 0 \tag{B.12}$$

ou

$$\tilde{q}_2^T F \tilde{q}_1 = 0 \tag{B.13}$$

onde a matriz 3×3 $F = \Lambda_2^{-T} E \Lambda_1^{-1} = \Lambda_2^{-T} \tau \times \Omega \Lambda_1^{-1}$. Conhecendo a matriz fundamental F e os parâmetros intrínsecos Λ_1 e Λ_2, é possível recuperar a matriz essencial E pela relação

$$E = \Lambda_2^T F \Lambda_1 \tag{B.14}$$

Logo, para câmeras calibradas, podendo-se estimar a matriz fundamental, então se pode também achar a rotação e a translação entre as câmeras [Hartley and Zisserman, 2004].

B.3 Profundidade do POI

A Equação B.1 descreve como o POI $\tilde{q}_i$ na imagem relaciona-se com sua correspondência 3D $\tilde{Q}$ em coordenadas homogêneas. Multiplicando os dois lados da Equação B.1 pela matriz de parâmetros intrínsecos Λ^{-1} obtemos:

$$\gamma_j \begin{bmatrix} x'_j \\ y'_j \\ 1 \end{bmatrix} = \begin{bmatrix} \omega_{11_j} & \omega_{12_j} & \omega_{13_j} & \tau_{xj} \\ \omega_{21_j} & \omega_{22_j} & \omega_{23_j} & \tau_{yj} \\ \omega_{31_j} & \omega_{32_j} & \omega_{33_j} & \tau_{zj} \end{bmatrix} \begin{bmatrix} u \\ v \\ w \\ 1 \end{bmatrix} \tag{B.15}$$

onde x'_j e y'_j são as coordenadas normalizadas da imagem da $j$ésima câmera.

Prince [2012] utiliza a terceira equação da matriz Ω para estabelecer que $\gamma_j = \omega_{31_j} u + \omega_{32_j} v + \omega_{33_j} w + \tau_z j$, e substituindo esta na Equação B.15, temos

$$\begin{bmatrix} (\omega_{31_j} u + \omega_{32_j} v + \omega_{33_j} w + \tau_z j) x'_j \\ (\omega_{31_j} u + \omega_{32_j} v + \omega_{33_j} w + \tau_z j) y'_j \end{bmatrix} = \begin{bmatrix} \omega_{11_j} & \omega_{12_j} & \omega_{13_j} & \tau_{xj} \\ \omega_{21_j} & \omega_{22_j} & \omega_{23_j} & \tau_{yj} \end{bmatrix} \begin{bmatrix} u \\ v \\ w \\ 1 \end{bmatrix} \tag{B.16}$$

Estas equações são então rearranjadas para gerar duas restrições lineares e três incógnitas em $\tilde{Q} = [u, v, w]^T$

$$\begin{bmatrix} \omega_{31_j} x'_j - \omega_{11_j} & \omega_{32_j} x'_j - \omega_{12_j} & \omega_{33_j} x'_j - \omega_{13_j} \\ \omega_{31_j} y'_j - \omega_{21_j} & \omega_{32_j} y'_j - \omega_{22_j} & \omega_{33_j} y'_j - \omega_{23_j} \end{bmatrix} \begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} \tau_x j - \tau_z j x'_j \\ \tau_y j - \tau_z j y'_j \end{bmatrix} \tag{B.17}$$

produzindo um sistema do tipo $Ax = b$ que pode ser resolvido por mínimos quadrados.

Resolvendo o sistema para cada conjunto de pontos correspondentes x_1 e x_2, obtém-se a posição 3D $\tilde{Q}$ com referência na primeira imagem, dado um par destas, desde que conheça-se os parâmetros intrínsecos e extrínsecos da câmera.

Pode-se ainda utilizar a triangulação linear para calcular a pose de uma câmera a partir de uma estrutura 3D cujo pontos de interesse 2D relacionam-se com pontos tridimensionais conhecidos. Está técnica é conhecida como um problema de PnP ou Perspectiva de n Pontos [Szeliski, 2010, Quan and Lan, 1999] ou Resseção (*Resectioning*) [Hartley and Zisserman, 2004].

Conforme a Equação B.15, dada em coordenadas normalizadas, os dois lados da igualdade são iguais a menos de um fator de escala γ_j. Logo, retirando a escala, podemos aplicar um produto vetorial de $\tilde{q} \times \tilde{P}\tilde{Q} = 0$, onde $\tilde{P} = [\Omega\tau]$, e dando origem ao sistema $Ax = 0$ com duas restrições, sendo

$$\begin{bmatrix} 0 & 0 & 0 & 0 & -u & -v & -w & -1 & ux'_j & vx'_j & wx'_j & x'_j \\ u & v & w & 1 & 0 & 0 & 0 & 0 & -uy'_j & -vy'_j & -wy'_j & -y'_j \end{bmatrix} \begin{bmatrix} \omega_{11_j} & \omega_{12_j} & \omega_{13_j} & \tau_{xj} \\ \omega_{21_j} & \omega_{22_j} & \omega_{23_j} & \tau_{yj} \\ \omega_{31_j} & \omega_{32_j} & \omega_{33_j} & \tau_{zj} \end{bmatrix} = 0 \tag{B.18}$$

Então, dada n correspondências de um ponto com coordenadas normalizadas e a sua posição 3D, pode-se por mínimos quadrados resolver a equação. Como cada correspondência ajuda na resolução de duas equações, são necessárias no mínimo seis correspondências para solucionar os doze parâmetros da matriz de transformação. A solução dá-se decompondo a matriz A em valores singulares por *Singular Value Decomposition* (SVD).

Apêndice C

Configurações do Sistema RGBDSLAM

Parametros

Estes parametros foram retirados diretamente do arquivo de log do *software*.

* /rgbdslam/config/adjuster_max_iterations: 0
* /rgbdslam/config/backend_solver: pcg
* /rgbdslam/config/batch_processing: True
* /rgbdslam/config/cloud_creation_skip_step: 8
* /rgbdslam/config/concurrent_edge_construction: True
* /rgbdslam/config/concurrent_node_construction: True
* /rgbdslam/config/concurrent_optimization: True
* /rgbdslam/config/data_skip_step: 1
* /rgbdslam/config/detector_grid_resolution: 0
* /rgbdslam/config/drop_async_frames: False
* /rgbdslam/config/emm__mark_outliers: False
* /rgbdslam/config/emm__skip_step: 1
* /rgbdslam/config/encoding_bgr: False
* /rgbdslam/config/feature_detector_type: IGFTT2 KAZE AKAZE FAST IGFTT GFTT SIFT SURF ORB MSDDetector AGAST MSER STAR
* /rgbdslam/config/feature_extractor_type: FREAK KAZE AKAZE SIFT SURF ORB
* /rgbdslam/config/fixed_camera: True
* /rgbdslam/config/gicp_max_cloud_size: 25000
* /rgbdslam/config/keep_all_nodes: True
* /rgbdslam/config/matcher_type: BRUTEFORCE_OPENCV
* /rgbdslam/config/max_connections: -1
* /rgbdslam/config/max_dist_for_inliers: 2.0
* /rgbdslam/config/max_keypoints: 600
* /rgbdslam/config/max_rotation_degree: 900
* /rgbdslam/config/max_translation_meter: 100.0
* /rgbdslam/config/min_keypoints: 0
* /rgbdslam/config/min_matches: 20
* /rgbdslam/config/min_rotation_degree: -1.0
* /rgbdslam/config/min_sampled_candidates: 10
* /rgbdslam/config/min_time_reported: 0.02
* /rgbdslam/config/min_translation_meter: -1.0
* /rgbdslam/config/neighbor_candidates: 10
* /rgbdslam/config/nn_distance_ratio: 0.95

* /rgbdslam/config/observability_threshold: 0.0
* /rgbdslam/config/octomap_resolution: 0.01
* /rgbdslam/config/optimize_landmarks: False
* /rgbdslam/config/optimizer_iterations: 0.001
* /rgbdslam/config/optimizer_skip_step: 10
* /rgbdslam/config/pose_relative_to: inaffected
* /rgbdslam/config/predecessor_candidates: 10
* /rgbdslam/config/ransac_iterations: 100
* /rgbdslam/config/save_all_matches_nodes: True
* /rgbdslam/config/squared_meshing_threshold: 0.0081
* /rgbdslam/config/start_paused: False
* /rgbdslam/config/store_pointclouds: True
* /rgbdslam/config/sufficient_matches: 18001
* /rgbdslam/config/use_glwidget: False
* /rgbdslam/config/use_gui: False
* /rgbdslam/config/use_icp: False
* /rgbdslam/config/use_root_sift: False
* /rgbdslam/config/visualization_skip_step: 1
* /rgbdslam/config/visualize_mono_depth_overlay: False
* /rosdistro: kinetic
* /rosversion: 1.12.7