Casamento de imagens com filtragem adaptativa
de *outliers*

Jonathan da Silva Ramos

SERVIÇO DE PÓS-GRADUAÇÃO DO ICMC-USP

Data de Depósito:

Assinatura: _____

Jonathan da Silva Ramos

Casamento de imagens com filtragem adaptativa de *outliers*

Dissertação apresentada ao Instituto de Ciências Matemáticas e de Computação - ICMC-USP, como parte dos requisitos para obtenção do título de Mestre em Ciências – Ciências de Computação e Matemática Computacional. *VERSÃO REVISADA*

Área de Concentração: Ciências de Computação e Matemática Computacional

Orientadora: Profa. Dra. Agma Juci Machado Traina

USP – São Carlos
Janeiro de 2017

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados fornecidos pelo(a) autor(a)

d175a da Silva Ramos, Jonathan
Algoritmos de casamento de imagens com filtragem adaptativa de outliers / Jonathan da Silva Ramos; orientadora Agma Juci Machado Traina. -- São Carlos, 2017.
102 p.

Dissertação (Mestrado - Programa de Pós-Graduação em Ciências de Computação e Matemática Computacional) -- Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, 2016.

1. Casamento de imagens. 2. Correspondência entre pontos. 3. Remoção de outliers. 4. Filtragem de outliers. I. Juci Machado Traina, Agma, orient. II. Título.

Jonathan da Silva Ramos

Image matching algorithms with adaptive filtering of outliers

Master dissertation submitted to the Instituto de Ciências Matemáticas e de Computação - ICMC-USP, in partial fulfillment of the requirements for the degree of the Master Program in Computer Science and Computational Mathematics. *FINAL VERSION*

Concentration Area: Computer Science and Computational Mathematics

Advisor: Profa. Dra. Agma Juci Machado Traina

USP – São Carlos
Janeiro de 2017

Dedico este trabalho aos meus pais e família,
por todo apoio incondicional e constante motivação.

Dedico também às Profas. Dra. Agma J. M. Traina e Dra. Carolina Y. V. Watanabe,
pela confiança, paciência, incentivo, amizade e excelente orientação.

A Johnny S. Ramos, in memoriam.

AGRADECIMENTOS

A Deus, acima de tudo.

A minha orientadora, Profa. Dra. Agma J. M. Traina pela confiança, paciência, incentivo, amizade e excelente orientação. Agradeço também a Profa. Dra. Carolina Y. V. Watanabe por toda paciência, amizade e orientação desde minha primeira iniciação científica.

Aos meus pais, Rosalvo J. Ramos, e Maria A. Silva, por todo o incentivo e apoio mesmo a quase três mil quilômetros de distância.

Todo o corpo docente do Instituto de Ciências Matemáticas e de Computação (ICMC¹), que diretamente ou indiretamente contribuíram para meu crescimento intelectual.

Todos os colegas do Grupo de Base de Dados e Imagens (GBDI²), que diretamente ou indiretamente contribuíram no desenvolvimento deste trabalho.

Ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq³, processo: 350251/2014-2), pelo apoio financeiro, em especial ao projeto RESCUER⁴ ao qual esta dissertação está inserida.

¹ <<http://www.icmc.usp.br/>>

² <<http://www.gbdi.icmc.usp.br/>>

³ <<http://cnpq.br/>>

⁴ RESCUER: Reliable and Smart Crowdsourcing Solution for Emergency and Crisis Management. <<http://www.rescuer-project.org/>>

“ To know, is to know that you know nothing.
That is the meaning of true knowledge.”
(Socrates)

RESUMO

RAMOS, J. R.. Algoritmos de casamento de imagens com filtragem adaptativa de outliers. 2016. 102 f. Dissertação (Mestrado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação (ICMC/USP), São Carlos – SP.

O registro de imagens tem um papel importante em várias aplicações, tais como reconstrução de objetos 3D, reconhecimento de padrões, imagens microscópicas, entre outras. Este registro é composto por três passos principais: (1) seleção de pontos de interesse; (2) extração de características dos pontos de interesse; (3) correspondência entre os pontos de interesse de uma imagem para a outra. Para os passos 1 e 2, algoritmos como SIFT e SURF têm apresentado resultados satisfatórios. Entretanto, para o passo 3 ocorre a presença de outliers, ou seja, pontos de interesse que foram incorretamente correspondidos. Uma única correspondência incorreta leva a um resultado final indesejável. Os algoritmos para remoção de outliers (consenso) possuem um alto custo computacional, que cresce à medida que a quantidade de outliers aumenta. Com o objetivo de reduzir o tempo de processamento necessário por esses algoritmos, o algoritmo FOMP (do inglês, Filtering out Outliers from Matched Points), foi proposto e desenvolvido neste trabalho para realizar a filtragem de outliers no conjunto de pontos inicialmente correspondidos. O método FOMP considera cada conjunto de pontos como um grafo completo, no qual os pesos são as distâncias entre os pontos. Por meio da soma de diferenças entre os pesos das arestas, o vértice que apresentar maior valor é removido. Para validar o método FOMP, foram realizados experimentos utilizando quatro bases de imagens. Cada base apresenta características intrínsecas: (a) diferenças de rotação zoom da câmera; (b) padrões repetitivos, os quais geram duplicidade nos vetores de características; (c) objetos deformados, tais como plásticos, papéis ou tecido; (d) transformações afins (diferentes pontos de vista). Os experimentos realizados mostraram que o filtro FOMP remove mais de 65% dos outliers, enquanto mantém cerca de 98% dos inliers. A abordagem proposta mantém a precisão dos métodos de consenso, enquanto reduz o tempo de processamento pela metade para os métodos baseados em grafos.

Palavras-chave: Casamento de imagens, Correspondência entre pontos, Remoção de outliers, Filtragem de outliers.

ABSTRACT

RAMOS, J. R.. Image matching algorithms with adaptive filtering of outliers. 2016. 102 p. Monograph (Master student – Program in Computer Science and Computational Mathematics) – Instituto de Ciências Matemáticas e de Computação (ICMC/USP), São Carlos – SP.

Image matching plays a major role in many applications, such as pattern recognition and microscopic imaging. It encompasses three steps: 1) interest point selection; 2) feature extraction from each point; 3) feature point matching. For steps 1 and 2, traditional interest point detectors/extractors have worked well. However, for step 3 even a few points incorrectly matched (outliers), might lead to an undesirable result. State-of-the-art consensus algorithms present a high time cost as the number of outlier increases. Aiming at overcoming this problem, we present FOMP, a preprocessing approach, that reduces the number of outliers in the initial set of matched points. FOMP filters out the vertices that present a higher difference among their edges in a complete graph representation of the points. To validate the proposed method, experiments were performed with four image database: (a) variations of rotation or camera zoom; (b) repetitive patterns, which leads to duplicity of features vectors; (c) deformable objects, such as plastics, clothes or papers; (d) affine transformations (different viewpoint). The experimental results showed that FOMP removes more than 65% of the outliers, while keeping over 98% of the inliers. Moreover, the precision of traditional methods is kept, while reducing the processing time of graph based approaches by half.

Key-words: Image matching, Feature point matching; Outlier removal, Outlier filtering.

LISTA DE ILUSTRAÇÕES

Figura 1	– Passos principais da fusão de imagens com sobreposição.	33
Figura 2	– Pontos de controle: pontos de esquina em verde e junções-T em rosa; vetor de características hipotético extraído de um ponto de esquina.	34
Figura 3	– Extração de vetor de características de um CP (SIFT).	35
Figura 4	– Cada CP detectado na primeira imagem é correspondido com um CP na segunda imagem. A correspondência é realizado pela métrica L2, onde o vizinho o mais	36
Figura 5	– Outliers (em vermelho) removidos usando o método RANSAC.	37
Figura 6	– Resultado final da fusão de duas imagens.	37
Figura 7	– Considerando duas imagens I e I' em um mesmo plano p , a homografia planar H é um mapeamento dos pontos da imagem I para os pontos da imagem I'	38
Figura 8	– Exemplo de iterações do algoritmo RANSAC. Em azul, modelo matemático encontrado para o subconjunto dos dados. Em verde, pontos que estão dentro da margem de erro do modelo matemático, isto é, inliers. O melhor resultado encontrado é a iteração que apresenta 59 inliers.	42
Figura 9	– Exemplo de k -vizinhos mais próximos mediano para os vértices a e d e h , onde $k = 3$ e a linha pontilhada representa o raio da mediana (hipotética).	45
Figura 10	– Exemplos de iterações do método GTM. Nas linhas, G_p corresponde ao grafo de mediana da imagem 1 e G'_p da imagem 2. Nas colunas, as iterações 0, 5, 10 e 16, respectivamente, para as duas imagens.	47
Figura 11	– Vértice v_i , correspondido com v'_i , e seus respectivos k -vizinhos mais próximos, onde $k = 6$	50
Figura 12	– Diagrama do princípio do vizinho-bilateral.	51
Figura 13	– Ilustração do algoritmo SOCBV e seus respectivos grafos k -nn medianos.	53
Figura 14	– Metodologia desenvolvida.	58
Figura 15	– Conjunto de pontos correspondidos. O ponto p_i está correspondido com o ponto p'_i . O ponto p_5 é um outlier	58
Figura 16	– Cada conjunto de ponto é representado como um grafo completo.	58
Figura 17	– Os pesos das arestas são dados pela distância (Equação 4.1) entre os vértices.	59
Figura 18	– O peso de cada grafo é normalizado por seu valor médio (Equação 4.2).	59
Figura 19	– Exemplo de aplicação da Equação 4.3 para um único vértice.	60
Figura 20	– Experimentos realizados.	64

Figura 21 – Comparação de medidas com e sem o pré-processamento FOMP, considerando as porcentagens de outliers manualmente adicionados para a base de imagens com movimentos de câmera. A legenda no gráfico de precisão é também válida para os demais gráficos.	66
Figura 22 – Exemplo de resultados para pares de imagens com 85% de outliers para a base de imagens: Movimentos de câmera.	67
Figura 23 – Comparação de medidas com e sem o pré-processamento FOMP, considerando as porcentagens de outliers manualmente adicionados para a base de imagens com correspondências ambíguas. A legenda no gráfico de precisão é também válida para os demais gráficos.	68
Figura 24 – Exemplo de resultados para pares de imagens com 85% de outliers para a base de imagens: Correspondências ambíguas.	69
Figura 25 – Comparação de medidas com e sem o pré-processamento FOMP, considerando as porcentagens de outliers manualmente adicionados para a base de imagens com objetos deformados. A legenda do gráfico de precisão é também válida para os demais gráficos.	70
Figura 26 – Exemplo de resultados para pares de imagens com 85% de outliers para a base de imagens: Objetos deformados.	71
Figura 27 – Comparação de medidas com e sem o pré-processamento FOMP, considerando as porcentagens de outliers manualmente adicionados para a base de imagens com transformações afim. A legenda do gráfico de precisão é também válida para os demais gráficos.	72
Figura 28 – Exemplo de resultados para pares de imagens com 85% de outliers para a base de imagens: Transformações afim.	73
Figura 29 – Incrementos na quantidade de correspondências corretas (inliers). Ambos os gráficos mostram valores médios para todos os experimentos, ou seja, 35 pares de imagens, totalizando 7:000 execuções para cada porcentagem de outlier.	76
Figura 30 – Valores de revocação e especificidade para variações no valor do parâmetro α do algoritmo FOMP.	77
Figura 31 – Imagens com aplicação do filtro Gaussiano em diferentes escalas e o cálculo da Diferença de Gaussianas - DoG entre pares de imagens adjacentes.	91
Figura 32 – Na esquerda, exemplo de aplicação de 5 filtros gaussianos em 4 escalas (oitavas). Na direita, resultado da DoG para as imagens borradas.	92
Figura 33 – Detecção de máximo/mínimo local, o pixel marcado com $\times$ é comparado com seus 26-vizinhos numa vizinhança de $3 \times 3 \times 3$ que fazem parte das imagens DoG adjacentes.	92

Figura 34 – Detecção de orientação dominante no método SIFT. A orientação e magnitude do gradiente são usados para compor a orientação do CP, um histograma de orientações é criado, o pico mais alto é detectado e qualquer outro pico que estiver 20% abaixo do maior pico é usado para compor a orientação. Se houver mais de uma orientação (pico) com a mesma magnitude, múltiplos CPs são criados na mesma localização, porém com orientações diferentes.	93
Figura 35 – O descritor é criado para cada CP calculando-se a magnitude do gradiente e a orientação ao seu redor. Esses pontos ao redor são ponderados por uma janela Gaussiana (como mostra o círculo à esquerda). Então é feita uma divisão em janelas de 4×4 ao redor do ponto de interesse (centro). Desta forma, é feita uma sumarização acumulando-se o conteúdo de cada janela, como é mostrado à direita. Neste exemplo é usado um descritor 2×2 calculado a partir de uma janela 8×8 . Porém, o método SIFT é implementado, na prática, em uma janela 16×16 que produz um descritor 4×4	94
Figura 36 – Passos na construção da matriz SAT. Índices fora da fronteira são considerados como zero.	94
Figura 37 – Aplicação da Equação A.7 para o cálculo da imagem integral em um dado retângulo ABCD.	95
Figura 38 – Filtros aproximados para o cálculo das derivadas parciais de segunda ordem, sendo é possível usar a SAT para o cálculo da convolução do filtro com a imagem.	95
Figura 39 – Tamanhos dos filtros e espaço-escala do algoritmo SURF.	96
Figura 40 – Cálculo da direção dominante de um ponto de interesse no algoritmo SURF.	97
Figura 41 – Exemplos de CPs detectados pelo SURF e extração de características.	98

LISTA DE ALGORITMOS

Algoritmo 1 – RANSAC. Adaptado de Fischler e Bolles (1981)	43
Algoritmo 2 – GTM, adaptado de Aguilar et al. (2009).	46
Algoritmo 3 – WGTM, adaptado de Izadi e Saeedi (2012).	48
Algoritmo 4 – SOCBV, adaptado de Meng, Li e Pei (2015).	54
Algoritmo 5 – Filtering out Outliers from Matched Points – FOMP.	61

LISTA DE TABELAS

Tabela 1 – Ordem de complexidade para os algoritmos baseados em grafos.	43
Tabela 2 – Quantidade de outliers adicionados para cada porcentagem de outliers de acordo com a Equação 5.5.	65
Tabela 3 – Comparação do tempo médio de execução para os quatro experimentos: movimentos de câmera, correspondências ambíguas, objetos deformados e transformações afim. O tempo é expressado em segundos.	74
Tabela 4 – Tempo médio de execução, quantidade de iterações, especificidade para todos os experimentos usando apenas o filtro FOMP. A média para todas as porcentagens de outliers é representada por m e o desvio padrão é denotado por $\pm$	75

LISTA DE ABREVIATURAS E SIGLAS

MBC _P	Matriz binária de classificação referente ao grafo G _P .
ASIFT . . .	Affine-SIFT
BRISK . . .	Binary Robust Invariant Scalable Keypoints
CP	Control Points
DoG	Difference of Gaussian
FAST	Features from Accelerated Segment Point
GTM	Graph Transformation Matching
HOG	Histogram of Oriented Gradients
LCS	Longest Common Subsequence
LoG	Laplacian of Gaussian
ORB	Oriented FAST and Rotated BRIEF
RANSAC .	RANdom SAmple Consensus
SAT	Summed Area Table
SIFT	Scale-Invariant Feature Transform
SOC	Spatial Order Constraint
SOCBV ..	Spatial Order Constraint Bilateral-Neighbour Vote
SURF	Speeded Up Robust Feature
USURF ..	Up-right SURF
WGTM . . .	Weighted Graph Transformation Matching

LISTA DE SÍMBOLOS

P — Conjunto de coordenadas de pontos na primeira imagem, em que $P = \{ p_i; \dots; p_n \}; 1 \leq i \leq n$, sendo n o total de pontos.

p_i — Ponto i do conjunto de pontos P em coordenadas cartesianas de duas dimensões, ou seja, $p_i = \{ x; y \}$.

P' — Conjunto de coordenadas de pontos na segunda imagem, em que $P' = \{ p'_i; \dots; p'_n \}; 1 \leq i \leq n$, sendo n o total de pontos.

p'_i — Ponto i do conjunto de pontos P' em coordenadas cartesianas de duas dimensões, ou seja, $p'_i = \{ x; y \}$, em que p'_i está correspondido com p_i .

PP — Resultado da concatenação dos conjunto P e P' em coordenadas homogêneas.

t — Número de pontos a serem aleatoriamente selecionados em cada iteração do algoritmo RANSAC.

e — Distância máxima permitida de um ponto à linha gerada para o modelo no algoritmo RANSAC.

$\max t$ — Número máximo de iterações permitidas no algoritmo RANSAC.

G_P — Grafo representativo dos pontos na primeira imagem.

V_P — Conjunto de vértices do grafo G_P .

v_i — Vértice i do conjunto de vértices V_P do grafo G_P .

E_P — Conjunto de arestas do grafo G_P .

A_P — Matriz de adjacências de tamanho $n \times n$, sendo n a quantidade de vértices no grafo G_P .

$A_P(i; j)$ — Distância entre os vértices v_i e v_j na matriz de adjacências A_P referente ao grafo G_P .

$G_{P'}$ — Grafo representativo dos pontos na segunda imagem.

$V_{P'}$ — Conjunto de vértices do grafo $G_{P'}$.

v'_i — Vértice i do conjunto de vértices $V_{P'}$ do grafo $G_{P'}$.

$E_{P'}$ — Conjunto de arestas do grafo $G_{P'}$.

$A_{P'}$ — Matriz de adjacências de tamanho $n \times n$, onde n é a quantidade de vértices no grafo $G_{P'}$.

$A_{P'}(i; j)$ — Distância entre os vértices v_i e v_j na matriz de adjacências $A_{P'}$ referente ao grafo $G_{P'}$.

k — Valor arbitrário que indica uma determinada quantidade de vizinhos.

h — Valor mediano de um dado conjunto de pesos de arestas.

j^{out} — Vértice a ser removido durante as iterações do algoritmo GTM.

R — Matriz residual contendo a diferença ponto a ponto entre as matrizes de adjacências A_P e $A_{P'}$.

h_P — Valor mediano dos pesos do conjunto de arestas do grafo representativo do conjunto de pontos P .

$h_{P'}$ — Valor mediano dos pesos do conjunto de arestas do grafo representativo do conjunto de pontos P' .

k_{min} — Vértice que apresenta o ângulo ótimo de rotação entre cada par de correspondência no algoritmo WGTM.

m_{old} — Média de todos os pesos médio de w para cada iteração anterior a atual do algoritmo WGTM.

W — Matriz de adjacências que contém distâncias angulares entre os vértices dos grafos G_P e $G_{P'}$.

w — Valores médio dos pesos de cada vértice da matriz de adjacências W (pesos angulares).

w_{max} — Valor máximo de todos os pesos da matriz de adjacências W (pesos angulares).

m_{new} — Média dos pesos médio de w para a iteração atual do algoritmo WGTM.

ϵ — Parâmetro utilizando como critério de parada no algoritmo WGTM.

O_{v_i} — Descritor SOC do vértice v_i do grafo G_P .

$O_{v'_i}$ — Descritor SOC do vértice v'_i do grafo $G_{P'}$.

m_i — Representa a relação de correspondência do ponto p_i com p'_i , isto é, $m_i = (p_i; p'_i)$.

$N_P^{in}(m_i)$ — Conjunto de vizinhos de entrada do par de correspondência m_i .

$N_P^{out}(m_i)$ — Conjunto de vizinhos de saída do par de correspondência m_i .

$N_P(m_i)$ — Conjunto de vizinhos bilaterais (entrada e saída) do par de correspondência m_i .

$Z_P(m_i)$ — Conjunto de inliers associados a m_i .

S_P — Voto de vizinho bilateral referente ao grafo G_P .

d_i — Grau do par de correspondência m_i .

b — Parâmetro usado como critério de parada do algoritmo SOCBV.

S — Média entre o voto de vizinho bilateral S_P e $S_{P'}$.

SUMÁRIO

1	INTRODUÇÃO	29
1.1	Motivação e justificativa	29
1.2	Objetivos e contribuições	30
1.3	Organização do trabalho	30
2	FUSÃO DE IMAGENS COM SOBREPOSIÇÃO	33
2.1	Detecção de pontos de interesse	34
2.2	Correlação entre pontos de interesse	35
2.2.1	Extração de vetores de características	35
2.2.2	Correspondência entre vetores de características	35
2.3	Fusão de imagens	36
2.3.1	Consenso entre pontos	36
2.3.2	Matriz homográfica	37
2.4	Considerações finais	38
3	CONSENSO ENTRE PONTOS DE INTERESSE	41
3.1	Randon Sample Consensus (RANSAC)	41
3.2	Métodos baseados em grafos	43
3.2.1	Grafos k-nn mediano	44
3.2.2	Graph Transformation Matching (GTM)	45
3.2.3	Weighted Graph Transformation Matching (WGTM)	46
3.2.4	Spatial Order Constraint Bilateral-neighbor Vote (SOCBV)	49
3.2.4.1	Descritor SOC	49
3.2.4.2	Estimativa de densidade k-nn	50
3.2.4.3	Vizinhos bilaterais	51
3.2.4.4	SOCBV	53
3.3	Considerações finais	55
4	FILTRANDO CORRESPONDÊNCIAS INCORRETAS	57
4.1	Filtering out Outliers from Matched points (FOMP)	57
4.1.1	Complexidade: tempo de processamento	60
4.2	Considerações finais	61
5	EXPERIMENTOS, RESULTADOS E DISCUSSÕES	63

5.1	Movimentos de câmera	65
5.2	Correspondências ambíguas	68
5.3	Objetos deformados	70
5.4	Transformações afins	72
5.5	Tempo de execução	74
5.6	Análise isolada do FOMP	75
5.6.1	Revocação e especificidade	75
5.6.2	Variações na quantidade de correspondências corretas	75
5.6.3	Influência do parâmetro a	76
5.7	Considerações finais	77
6	CONCLUSÃO	79
6.1	Principais contribuições deste trabalho	80
6.2	Proposta para trabalhos futuros	80
REFERÊNCIAS		81
APÊNDICE A DETECÇÃO DE PONTOS DE INTERESSE		89
A.1	Scale-Invariant Feature Transform – SIFT	90
A.1.1	Deteção de pontos de interesse	90
A.1.2	Extração de características	93
A.2	Speeded Up Robust Features – SURF	93
A.2.1	Deteção de pontos de interesse	94
A.2.2	Extração de características	97
A.3	Considerações finais	98
APÊNDICE B CÓDIGOS-FONTE DO ALGORITMO FOMP		101

INTRODUÇÃO

Desde os primórdios, sinais visuais são utilizados como meio de comunicação, como exemplo, sinais de fumaça e desenhos rústicos nas paredes das cavernas (GROBEL; TELLES, 2014). Ao decorrer dos anos, as tecnologias foram evoluindo, permitindo a criação de imagens na forma digital. Desde então, as imagens digitais têm sido utilizadas em diversas áreas do conhecimento humano: na comunicação elas dão dinamismo à linguagem (RAVAL; ZAFAR, 2013); na segurança auxiliam o processo de monitoramento e garantem o uso de senhas confiáveis (XIAO et al., 2014; GONGE; GHATOL, 2014); na medicina são de fundamental importância para a apresentação de diversos tipos de exames médicos (GODINHO et al., 2016; PARIKH; KALVA; ADZIC, 2016).

Apesar de todo o avanço tecnológica, em alguns casos não é possível obter uma imagem fiel e precisa de uma região de interesse de uma determinada cena. Como exemplo, em imagens de satélite e sensoriamento remoto (PATEL; PATEL; HOLIA, 2015; NAZNEEN; SHAFIQ; HAMEED, 2016; SEDAGHAT; EBADI, 2015; LIU et al., 2016), imagens médicas (GALANDE; PATIL, 2013; PHAN et al., 2016), restauração de imagens (BRITO et al., 2016; SATNIK et al., 2016), perícia de documentos para indicar autenticidade (ARDIZZONE; BRUNO; MAZZOLA, 2015), reconhecimento de padrões e recuperação por conteúdo (GURUPRASAD; GURUPRASAD, 2015; TAKÁCS et al., 2015; CHOI; HAN, 2014), entre outras (LIQIAN; YUEHUI, 2010; TONG et al., 2012). Nestes casos, a integração/fusão de imagens pode ser aplicada.

1.1 Motivação e justificativa

Um dos passos necessários para a fusão de imagens é encontrar correspondências entre pares de imagens. Esta correspondência geralmente é realizada por meio de detecção de pontos de interesse nas duas imagens. Estes pontos de interesse podem ser junções-T, bordas ou pontos de esquina. Posteriormente, são extraídos vetores de características para cada ponto de interesse considerando as características de seus pixels vizinhos.

Os pontos de interesse são correspondidos entre as imagens usando seus vetores de características: cada ponto é correspondido com o ponto que estiver mais próximo, considerando uma métrica de distância, como exemplo a distância Euclidiana. Contudo, alguns dos pontos correspondidos entre as imagens podem estar incorretos (outliers). Correspondências incorretas resultam em uma função de transformação incorreta, que acarreta em um resultado distorcido ou errado. Portanto, é necessário um algoritmo robusto para correspondência de pontos de interesse entre imagens.

Alguns algoritmos de consenso têm sido propostos para lidar com este problema, tais como RANdom SAmple Consensus (RANSAC) (FISCHLER; BOLLES, 1981), Graph Transformation Matching (GTM) (AGUILAR et al., 2009), Weighted Graph Transformation Matching (WGTM) (IZADI; SAEEDI, 2012), Spatial Order Constraint Bilateral-Neighbour Vote (SOCBV) (MENG; LI; PEI, 2015). Entretanto, esses algoritmos possuem um custo computacional elevado, que cresce ainda mais à medida que a quantidade de correspondências incorretas aumenta.

1.2 Objetivos e contribuições

O objetivo geral do projeto se concentra em desenvolver um método robusto e rápido para encontrar correspondências entre duas imagens (para posterior fusão). A hipótese inicial é que isto pode ser realizado com a aplicação de um filtro de pré-processamento, o qual reduz a quantidade de outliers no conjunto inicial de pontos correspondidos entre as duas imagens. A redução da quantidade de outliers acarreta em uma quantidade menor de iterações para os algoritmos de consenso, uma vez que a maior parte destes algoritmos de consenso remove um outlier por vez. Essa forma de tratamento adaptativo dos outliers permite efetuar um processamento mais rápido.

1.3 Organização do trabalho

Além deste capítulo introdutório, esta dissertação de mestrado está estruturada em mais cinco capítulos e dois apêndices, organizados da seguinte forma:

- **Capítulo 2:** registro de imagens, onde são discutidos os conceitos e técnicas relacionadas ao registro de imagens.
- **Capítulo 3:** onde são detalhados os principais algoritmos do estado-da-arte para remoção de pontos correspondidos incorretamente.
- **Capítulo 4:** detalha o método de filtragem proposto neste trabalho de mestrado (FOMP).
- **Capítulo 5:** são apresentados os experimentos, resultados e discussões obtidos com o uso do método FOMP.

- [Capítulo 6](#): são explicitadas as conclusões finais desta dissertação, assim como principais contribuições deste trabalho e trabalhos futuros.
- [Apêndice A](#): detecção de pontos de interesse, onde são detalhados os principais métodos para detecção de pontos de interesses em imagens digitais.
- [Apêndice B](#): disponibiliza os códigos-fonte do algoritmo FOMP.

FUSÃO DE IMAGENS COM SOBREPOSIÇÃO

O processo de fundir imagens consiste em alinhar duas ou mais imagens com regiões sobrepostas de forma a compor uma única imagem (BROWN; LOWE, 2007). Na Figura 1 são mostrados os três passos principais deste processo.

Figura 1 – Passos principais da fusão de imagens com sobreposição.

Fonte: Elaborada pelo autor.

No primeiro passo é feita a detecção de pontos de controle/interesse nas imagens, que são pontos que podem ser encontrados em ambas as imagens. Estes pontos devem ser únicos e repetíveis nas imagens que contém aquela mesma cena. Em seguida, os pontos detectados são correlacionados entre as imagens por meio da extração de vetores de características e cor res-

pondência. Este passo é brevemente descrito na [subseção 2.2.1](#), e alguns descritores/extratores, tais como Scale-Invariant Feature Transform (SIFT) e Speeded Up Robust Feature (SURF), são explanados em detalhes no [Apêndice A](#).

Uma vez que os pontos são correlacionados, é possível calcular uma matriz de transformação para realizar a junção entre as imagens. Entretanto, se existirem correspondências incorretas (outliers), o resultado final será drasticamente afetado. Desta forma, são aplicados algoritmos para remoção de outliers, ou algoritmos de consenso. Este passo é brevemente detalhado na [subseção 2.3.2](#) e no [Capítulo 3](#) são explanados em detalhes alguns algoritmos de consenso, tais como RANSAC, GTM, WGTM e SOCBV.

2.1 Detecção de pontos de interesse

Um ponto de controle/interesse deve ser, idealmente, repetido em todas as imagens e também único em cada uma delas. Sendo assim, é necessário um algoritmo detector de pontos de interesse confiável, que encontre pontos equivalentes em todas as imagens analisadas, mesmo quando há diferenças de iluminação, rotação e/ou de resoluções entre as imagens. Desta forma, algumas técnicas de filtragem/processamento de imagens tentam encontrar pontos de destaque visual, tais como esquinas, junções-T e bordas ([LOWE, 1999](#)), como mostra a [Figura 2](#).

Figura 2 – Pontos de controle: pontos de esquina em verde e junções-T em rosa; vetor de características hipotético extraído de um ponto de esquina.

Fonte: Elaborada pelo autor.

Duas abordagens podem ser usadas para corresponder estes CPs (do inglês, Control Points) detectados. A primeira abordagem é a extração de características de cada CP e sua correspondência, por meio de alguma métrica de distância. A segunda abordagem é a correlação cruzada. A correlação cruzada possui um custo computacional maior, que cresce exponencialmente com relação ao tamanho da janela de correlação ([ZITOVÁ; FLUSSER, 2003](#)). Portanto, a extração de vetores de características é amplamente utilizada ([FANG et al., 2014](#); [GALANDE;](#)

PATIL, 2013; TONG et al., 2012). Além do mais, a correspondência por vetores de características apresenta melhores resultados na maior parte dos casos (ZITOVÁ; FLUSSER, 2003).

2.2 Correlação entre pontos de interesse

Uma vez que os CPs são encontrados, estes podem ser representados por vetores de características e correspondidos utilizando alguma métrica de distância entre os vetores.

2.2.1 Extração de vetores de características

As características de cada ponto podem ser extraídas seguindo algum critério de relacionamento entre o CP e seus pixels vizinhos. Como exemplo, supondo que a imagem exibida na Figura 2 tenha apenas três cores: 0, branco; 1, rosa; 2, cinza. Um vetor de características hipotético pode ser extraído da imagem analisando o gradiente dos pixels vizinhos em duas direções, diagonal principal (45°) e secundária (135°). Considerando uma janela de 3×3 ao redor do CP, tem-se então um vetor com 8 características.

Um outro exemplo, mostrado na Figura 3, é o uso da magnitude do gradiente usando os 16 pixels vizinhos do CP, divididos em janelas de 4×4 com 8 orientações de gradientes para cada janela, totalizando assim 128 características. Este extrator é conhecido como SIFT (do inglês, Scale Invariant Feature Transform) (LOWE, 1999).

Figura 3 – Extração de vetor de características de um CP (SIFT).

Fonte: Adaptada de Lowe (1999).

2.2.2 Correspondência entre vetores de características

Os CPs são correspondidos de uma imagem para a outra por meio do cálculo da distância entre seus respectivos vetores de características. A métrica de distância normalmente usada é a distância euclidiana, L2, a qual calcula a menor distância (linha reta) entre dois pontos em um espaço n-dimensional (DEZA; DEZA, 2009). Desta forma, a distância entre um vetor e todos os vetores da outra imagem é calculada, o vetor que estiver mais próximo no espaço de características n-dimensional (n depende da dimensão do vetor de características, no caso do

SIFT é igual a 128) é escolhido como correspondência. Na [Figura 4](#) pode ser visto um exemplo de CPs correspondidos por meio do vizinho mais próximo, calculados pela L2.

Figura 4 – Cada CP detectado na primeira imagem é correspondido com um CP na segunda imagem. A correspondência é realizada pela métrica L2, onde o vizinho o mais .

Fonte: Elaborada pelo autor.

Como pode ser visto na [Figura 4](#), alguns pontos foram incorretamente correspondidos. No contexto deste trabalho, estes pontos são denominados outliers e devem ser removidos antes da estimação da matriz homográfica, a qual é explanada na [subseção 2.3.2](#).

2.3 Fusão de imagens

Duas imagens da mesma cena ou com sobreposição de regiões estão relacionadas por uma homografia, a qual pode ser facilmente calculada usando a matriz homográfica, que utiliza pontos correspondentes entre as imagens. Entretanto, existe a possibilidade de existência de pontos incorretamente correspondidos. Desta forma, faz-se necessário o uso do consenso entre os pontos correspondidos, o qual visa remover correspondências incorretas.

2.3.1 Consenso entre pontos

Como a correspondência entre os CPs é calculada automaticamente, pode haver pontos incorretamente correspondidos, chamados de outliers¹ neste trabalho. O primeiro método proposto para consenso entre pontos de interesse (remoção de outliers) foi o método RANSAC, que tenta minimizar o erro do método dos mínimos quadrados ([FISCHLER; BOLLES, 1981](#)), o qual é detalhado na [seção 3.1](#). Na [Figura 5](#) pode ser visto um exemplo de outliers, removidos utilizando o método RANSAC.

¹ Em estatística, um outlier é uma observação em um conjunto de dados que está distante ou “comporta-se” diferentemente das demais observações do conjunto ([MORANO; TAJANI, 2014](#)).

Figura 5 – Outliers (em vermelho) removidos usando o método RANSAC.

Fonte: Elaborada pelo autor.

Dados então os CPs corretamente correspondidos (em amarelo na [Figura 5](#)), é possível calcular uma matriz homográfica de transformação entre as imagens e realizar a fusão, gerando uma imagem que integra a informação das imagens iniciais, como mostra a [Figura 6](#).

Figura 6 – Resultado final da fusão de duas imagens.

Fonte: Elaborada pelo autor.

2.3.2 Matriz homográfica

Duas imagens de uma mesma cena estão relacionadas por uma homografia, também chamada de transformação projetiva planar, colineação ou projetividade. Uma homografia descreve um mapeamento linear entre espaços planares bidimensionais ([AIRES, 2009](#)).

Figura 7 – Considerando duas imagens I e I' em um mesmo plano p , a homografia planar H é um mapeamento dos pontos da imagem I para os pontos da imagem I' .

Fonte: Aires (2009).

Considere duas imagens I e I' , que serão integradas no mesmo plano p , um ponto Q_p deste plano, e dois pontos q e q' que são, respectivamente, pontos de I e I' que correspondem ao ponto Q_p do plano p . A homografia H é o mapeamento $q \leftrightarrow q'$ induzido unicamente pelo plano p , como pode ser visto na Figura 7.

A transformação projetiva de uma imagem para a outra é expressada da seguinte forma:

$$q' \simeq Hq; \quad (2.1)$$

onde H representa uma homografia planar, composta por uma matriz não singular de tamanho 3×3 . Desta forma, os pontos 2D das duas imagens são transformados em coordenadas homogêneas (3D), pertencentes a um plano no espaço, sendo expresso pela equação:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} \simeq \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}; \quad (2.2)$$

A Transformação Direta Linear (do inglês, Direct Linear Transform - DLT) pode ser usada para estimar a matriz H . A DLT é expressa na forma de produtos vetoriais $q' \times (H q_i) = 0$ (DUBROFSKY, 2009).

2.4 Considerações ais

Dentre todos os passos envolvidos no processo de casamento de imagens com sobreposição, a fusão é o ponto crítico e sensível a erros causados nos outros passos. O mapeamento de uma imagem para a outra por meio da matriz homográfica depende de pontos corretamente

correspondidos. Apesar de ser necessário apenas 4 ou 8 pontos para realizar o mapeamento, estes pontos devem estar corretamente correspondidos, caso contrário o resultado final será indesejável. O uso de métodos de consenso, tal como o RANSAC, exemplificado na [Figura 5](#), apresenta resultados satisfatórios dependendo da quantidade de outliers. Entretanto, esta técnica pode não chegar a um resultado adequado ou pode tomar um tempo de processamento bastante elevado ([AGUILAR et al., 2009](#)). Além do mais, a revocação do método é baixa, ou seja, grande parte das correspondências corretas são dadas como incorretas. Algumas técnicas baseadas em grafos tem sido propostas como tentativa de melhora do RANSAC. Apesar destas técnicas baseadas em grafos apresentarem resultados melhores que o RANSAC, ainda possuem um custo computacional consideravelmente alto ([SANROMÀ; ALQUÉZAR; SERRATOSA, 2012](#)).

CONSENSO ENTRE PONTOS DE INTERESSE

O processo de buscar a correspondência, mesmo que aproximada, de duas representações de dados abstratos, tem sido largamente estudado no campo de visão computacional ([GOLD; RANGARAJAN, 1996](#); [CHUI, 2001](#); [CHUI; RANGARAJAN, 2003](#); [MYRONENKO; SONG; CARREIRA-PERPIÑÁN, 2006](#); [AGUILAR et al., 2009](#); [DU et al., 2010](#); [COLLINS; MESEJO; BARTOLI, 2014](#)).

No caso dos métodos SIFT e SURF, a correspondência entre os pontos de interesse nas duas imagens é feita por meio do cálculo da distância entre seus respectivos vetores de características. Entretanto, na maior parte dos casos, há a presença de pontos que são incorretamente correspondidos, chamados de outliers no contexto deste trabalho.

Uma primeira abordagem para detecção de outliers é o uso de técnicas probabilísticas, como exemplo o algoritmo RANSAC, que tenta iterativamente estimar um modelo matemático que melhor se adeque aos dados, minimizando assim o erro do método dos mínimos quadrados. Entretanto, esta técnica possui um alto custo computacional e nem sempre apresenta resultados adequados ([AGUILAR et al., 2009](#)).

Uma segunda abordagem é a representação dos pontos em cada imagem como grafos. Devido ao poder representativo dos grafos, métodos dessa linha, tais como GTM ([AGUILAR et al., 2009](#)), WGTM ([IZADI; SAEEDI, 2012](#)) e SOCBV ([MENG; LI; PEI, 2015](#)), têm apresentado resultados promissores. Porém, o esforço computacional ainda é alto ([GOLD; RANGARAJAN, 1996](#); [AGUILAR et al., 2009](#); [COLLINS; MESEJO; BARTOLI, 2014](#)). Neste capítulo são detalhados os métodos RANSAC, GTM, WGTM e SOCBV.

3.1 Randon Sample Consensus (RANSAC)

O método RANSAC, usa o método dos mínimos quadrados de forma iterativa. Em cada iteração, um subconjunto dos dados é selecionado e um modelo para este subconjunto é gerado. Baseando-se em uma margem de erro, a quantidade de pontos que se encaixam no modelo é somada. Este processo é repetido até um número fixo de iterações. A iteração que apresentar a maior quantidade de pontos, mantendo-se a margem de erro de um limiar, é considerada como resposta ao problema. Na Figura 8 pode ser visto um exemplo de quatro iterações do método RANSAC em um conjunto de dados com duas dimensões, em que dois pontos foram aleatoriamente selecionados em cada iteração.

Figura 8 – Exemplo de iterações do algoritmo RANSAC. Em azul, modelo matemático encontrado para o subconjunto dos dados. Em verde, pontos que estão dentro da margem de erro do modelo matemático, isto é, inliers. O melhor resultado encontrado é a iteração que apresenta 59 inliers.

Fonte: Adaptada de [Wedge \(2012\)](#).

Algumas variações do método RANSAC foram propostas ([SATTLER; LEIBE; KOBELT, 2009](#); [CHUM; MATAS; KITTLER, 2003](#); [LEBEDA; MATAS; CHUM, 2012](#); [CHUM; MATAS, 2002](#); [MATAS; CHUM, 2005](#)). Porém, neste trabalho é detalhada apenas a ideia clássica do método. Considerando o conjunto de pontos da primeira imagem como $P = \{p_1; \dots; p_n\}; 1 \leq i \leq n$, sendo n o total de pontos. Cada ponto está em coordenadas cartesianas de duas dimensões, ou seja, $p_i = \{x; y\}$. De forma similar, o conjunto de pontos da segunda imagem como $P' = \{p'_1; \dots; p'_n\}; 1 \leq i \leq n$, sendo n o total de pontos. Cada ponto está em coordenadas cartesianas de duas dimensões, ou seja, $p'_i = \{x; y\}$, em que cada ponto p'_i está correspondido, respectivamente, ao ponto p_i . O método RANSAC é mostrado no [Algoritmo 1](#) e detalhado como segue.

Primeiramente, o conjunto de pontos correspondidos das imagens 1 e 2, respectivamente, P e P' , são colocados em coordenadas homogêneas, isto é, uma nova dimensão atribuída como 1 é adicionada ao ponto ($p_i = (x_i; y_i; 1)^T$ e $p'_i = (x'_i; y'_i; 1)^T$). Em seguida, cada conjunto de pontos é normalizado de forma que a origem esteja no centro e a distância média da origem seja $\sqrt{2}$. Isto faz com que o problema de estimação esteja bem condicionado.

Após a normalização dos pontos das duas imagens, cada ponto em coordenadas homo-

Algoritmo 1: RANSAC. Adaptado de [Fischler e Bolles \(1981\)](#)

Entrada: P, P' : conjuntos de pontos referentes à primeira e segunda imagens, respectivamente.
 t : número de pontos a serem aleatoriamente selecionados em cada iteração.
 e : a distância máxima permitida de um ponto à linha gerada para o modelo.
 $maxIt$: número máximo de iterações permitidas.
Saída: Subconjunto de P e P' contendo os inliers.

```

1 Função RANSAC ( $P, P', t, e, maxIt$ )
2    $P_{norm} \leftarrow \text{normalizarPontosCoordenadasHomogeneas}(P, P')$ 
3    $melhorModelo \leftarrow 0$ 
4   // Critério de parada
5   enquanto contador <  $maxIt$  faça
6      $P_{selecionado} \leftarrow \text{selecionarPontosAleatorios}(t, P_{norm})$ 
7      $modelo \leftarrow \text{calcularParametrosModelo}(P_{selecionado})$ 
8      $qtInliers \leftarrow \text{calcularQuantidadeInliersModelo}(modelo, P_{norm})$ 
9     se  $qtInliers > melhorModelo$  então
10       $melhorModelo \leftarrow qtInliers$ 
11  retorna subconjunto de  $P$  e  $P'$  contido no  $melhorModelo$ 

```

gêneas de P é concatenado a P' de forma que cada par de correspondência possua 6 dimensões, ou seja, $PP = (x; y; 1; x'; y'; 1)$. Desta forma, o algoritmo RANSAC, exemplificado anteriormente em um caso simples de 2 dimensões (Figura 8), pode ser aplicado sobre PP , com 6 dimensões.

Dados então os inliers, obtidos pelo RANSAC, é possível calcular a matriz de transformação que mapeia os pontos de uma imagem para fundir com a outra, que satisfaz a [Equação 3.1](#).

$$P'^T F P = 0; \quad (3.1)$$

em que F é uma matriz 3×3 de transformação dos pontos de P para P' . É extremamente importante que não haja outliers nesse passo, pois um único outlier presente no conjunto resultará em uma transformação incorreta.

O RANSAC é um algoritmo não determinístico, no sentido de que ele produz apenas uma resposta aceitável de acordo com uma determinada probabilidade, sendo que a probabilidade cresce à medida que a quantidade de iterações aumenta. Desta forma, duas instâncias diferentes do algoritmo em um mesmo conjunto de dados pode acarretar em resultados diferentes.

3.2 Métodos baseados em grafos

Nesta seção são apresentados os principais métodos de consenso entre pontos de interesse baseados em grafos. Os métodos estão apresentados em ordem cronológica. Na [subseção 3.2.1](#) são definidos os conceitos utilizados ao longo do capítulo e apresentado o conceito de grafo k-nn mediano, o qual é utilizado pelos métodos GTM ([subseção 3.2.2](#)), WGTm ([subseção 3.2.3](#)) e

SOCBV (subseção 3.2.4). Na Tabela 1 é mostrada a ordem de complexidade e ano de divulgação destes algoritmos.

Tabela 1 – Ordem de complexidade para os algoritmos baseados em grafos.

Autor	Algoritmo	Ordem de Complexidade de tempo	Ano de divulgação
Aguilar et al. (2009)	GTM	$O(n^3 \log(n))$	2009
Izadi e Saeedi (2012)	WGTM	$O(n^2 \log(n))$	2012
Meng, Li e Pei (2015)	SOCBV	$O(n^2 \log(n))$	2015

Fonte: Elaborada pelo autor.

3.2.1 Grafos k-nn mediano

Inicialmente, considera-se P como o conjunto de pontos de interesse detectados na primeira imagem, sendo $P = \{p_i\}; 1 \leq i \leq n$, com n representando o total de pontos de interesse e $p_i = \{x; y\}$ representando as coordenadas de um único ponto. De forma similar, os pontos de interesse detectados na segunda imagem são considerados como $P' = \{p'_1; \dots; p'_n\}$, sendo que p'_i está correspondido com o ponto p_i . A partir destes dois conjuntos de pontos, dois grafos completos são criados: cada ponto é considerado como um vértice do grafo e o peso das arestas é dado pela distância entre eles.

O grafo referente ao primeiro conjunto de pontos, P , é representado por $G_P = (V_P; E_P)$, sendo $V_P = \{v_1; \dots; v_n\}$. Cada vértice v_i refere-se a um ponto em P , isto é, $p_i \equiv v_i$. Inicialmente, o conjunto de arestas E_P é composto por todas as arestas possíveis na rede, ou seja, todos os vértices estão conectados a todos os vértices. Considerando o grafo G_P como uma matriz de adjacências A_P de tamanho $n \times n$, o peso da aresta $v_i - v_j$ é dada pela distância euclidiana, como mostra a Equação 3.2.

$$A_P(i; j) = \|p_i - p_j\|; 1 \leq i; j \leq n \quad (3.2)$$

De forma similar, cada ponto da segunda imagem é considerado como um vértice da rede, representado pelo grafo $G_{P'} = (V_{P'}; E_{P'})$. Cada um dos vértices do conjunto de vértices $V_{P'}$ representa um ponto em P' , ou seja, $p'_i \equiv v'_i$, ressaltando que o vértice v'_i é correspondência do vértice v_i no primeiro grafo. Inicialmente, o conjunto de arestas $E_{P'}$ é composto por todas as conexões possíveis entre os vértices do grafo. Os pesos das arestas são dados pela distâncias calculadas, utilizando-se a distância euclidiana, e representados pela matriz de adjacências $A_{P'}$.

Os grafos k-nn medianos são construídos por meio dos dois grafos completos, G_P e $G_{P'}$. Considerando o grafo G_P como exemplo, cada vértice é conectado, no máximo, aos seus k-vizinhos mais próximos. Entretanto, estes k-vizinhos devem possuir uma distância menor do

que o valor da mediana de todos os pesos do grafo, representado por h na [Equação 3.3](#).

$$h = \text{mediana}_{(i;j) \in V_p \times V_p} \left(\begin{matrix} \boxed{E_0} \\ \boxed{E_1} \\ \boxed{E_2} \end{matrix} p_i - p_j \right). \quad (3.3)$$

As matrizes de adjacências de ambos os grafos, A_P e $A_{P'}$, passam a ter valores binários, de forma que, $A_P(i; j) = 1$ se o vértice v_j for um dos k -vizinhos mais próximos de v_i e a aresta entre v_i e v_j apresenta um valor menor ou igual a h ($\|p_i - p_j\| \leq h$). Caso contrário, $A_P(i; j) = 0$. Todas as arestas apresentam estrutura não-direcionada.

Note que o fato de o vértice j ser um dos k -vizinhos mais próximos de i não implica que o vértice i seja um dos k -vizinhos mais próximos de j e vice-versa. Na [Figura 9](#) é exemplificada esta possibilidade. Apesar do vértice d ser um dos k -vizinhos mais próximos de a , a não é um dos k -vizinhos mais próximos de d . Com isso tem-se que o vértice i poderá ter menos do que k arestas conectadas a ele devido a restrição da mediana, h . No exemplo da [Figura 9](#), o vértice h ilustra essa possibilidade. Em contraste, o vértice i pode ter mais do que k arestas conectadas a ele, uma vez que o vértice i pode ser um dos k -vizinhos próximos de qualquer outro vértice do grafo (vértice d na [Figura 9](#)).

Figura 9 – Exemplo de k -vizinhos mais próximos mediano para os vértices a e d e h , onde $k = 3$ e a linha pontilhada representa o raio da mediana (hipotética).

Fonte: Elaborada pelo autor.

3.2.2 Graph Transformation Matching (GTM)

A ideia principal do método GTM é usar a relação espacial existente entre os pontos correspondidos para a detecção de outliers ([AGUILAR et al., 2009](#)). O algoritmo GTM consiste em eliminar iterativamente pontos com comportamento diferente de seus vizinhos, como mostrado no [Algoritmo 2](#) e resumizado em 3 passos como segue:

1. Inicialmente, são criados dois grafos k -nn medianos, como descrito na [subseção 3.2.1](#). Porém, os vértices que possuem menos do que k -vizinhos mais próximos são desconsiderados.

Algoritmo 2: GTM, adaptado de [Aguilar et al. \(2009\)](#).

Entrada: P, P' : conjuntos de pontos referentes à primeira e segunda imagem, respectivamente.

k : número de vizinhos para criação dos grafos k -nn medianos.

Saída: Q e Q' , que são subconjuntos de P e P' , respectivamente, contendo os inliers.

```

1 Função GTM ( $P, P', k$ )
2    $\text{dist } p \leftarrow \text{calcularMatrizDistâncias}(P)$ 
3    $\text{dist } p' \leftarrow \text{calcularMatrizDistâncias}(P')$ 
4    $h_P \leftarrow \text{calcularMediana}(\text{dist } p)$ 
5    $h_{P'} \leftarrow \text{calcularMediana}(\text{dist } p')$ 
6    $Q \leftarrow P, Q' \leftarrow P'$ 
   // Passo 1
7    $A_p \leftarrow \text{construirGrafoKNNMediano}(\text{dist } p, k, h_P)$ 
8    $A_{p'} \leftarrow \text{construirGrafoKNNMediano}(\text{dist } p', k, h_{P'})$ 
   // Critério de parada
9   enquanto  $A_p \neq A_{p'}$  faça
   // Passo 2
10     $\text{outlier} \leftarrow \underset{j=1::N}{\text{argmax}} \sum_{i=1}^N |A_p(i; j) - A_{p'}(i; j)|$  // Equação 3.4
11    Remover outlier de  $\text{dist } p, \text{dist } p', Q$  e  $Q'$ 
   // Passo 3
12     $A_p \leftarrow \text{construirGrafoKNNMediano}(\text{dist } p, k, h_P)$ 
13     $A_{p'} \leftarrow \text{construirGrafoKNNMediano}(\text{dist } p', k, h_{P'})$ 
14     $Q \leftarrow \text{removerVerticesDesconectados}(A_p)$ 
15     $Q' \leftarrow \text{removerVerticesDesconectados}(A_{p'})$ 
16  retorna  $Q$  e  $Q'$ 

```

2. O vértice que apresentar a maior diferença residual, j^{out} , é removido de ambos os grafos:

$$j^{\text{out}} = \underset{j=1::N}{\text{argmax}} \sum_{i=1}^N R(i; j) \quad (3.4)$$

em que R representa a matriz residual, $R = |A_p - A_{p'}|$, que é a diferença ponto a ponto entre as matrizes de adjacências.

3. Os grafos k -nn medianos são calculados novamente e o algoritmo retorna ao passo 2. Este processo se repete até que os grafos sejam isomorfos¹, ou seja, a diferença residual seja zero para todos os vértices.

O GTM se apoia na ideia de que a transformação que ocorre em ambas as imagens é relativamente pequena. Se todas as correspondências estiverem corretas, os grafos G_p e $G_{p'}$ serão isomorfos. Correspondências incorretas resultam em diferenças estruturais entre os dois grafos, as quais podem ser desconsideradas aplicando um critério estrutural simples (ver [Equação 3.4](#)). Na [Figura 10](#) são apresentados exemplos de iterações do algoritmo.

¹ Quando há um mapeamento um-para-um entre os vértices e suas respectivas arestas nos dois grafos.

Figura 10 – Exemplos de iterações do método GTM. Nas linhas, G_p corresponde ao grafo de mediana da imagem 1 e G'_p da imagem 2. Nas colunas, as iterações 0, 5, 10 e 16, respectivamente, para as duas imagens.

Fonte: Adaptada de [Aguilar et al. \(2009\)](#).

3.2.3 Weighted Graph Transformation Matching (WGTM)

Baseado no método GTM, [Izadi e Saeedi \(2012\)](#) propuseram o método WGTM, que além de utilizar o relacionamento entre os pontos vizinhos, também utiliza as distâncias angulares, as quais são invariantes com relação à escala e rotação, e têm uma menor sensibilidade a ruídos gerados pela distorção da lente da câmera. O método é mostrado no [Algoritmo 3](#) e detalhado em 6 passos como segue:

1. Inicialmente, são criados dois grafos k-nn medianos, como foi descrito na [subseção 3.2.1](#). Entretanto, existe uma alteração na criação destes grafos. As arestas são direcionadas, podendo as matrizes A_p e $A_{p'}$ não serem simétricas. Diferentemente do método GTM, os vértices com menos de k-vizinhos mais próximos não são eliminados, desde que apresentem pelo menos dois vizinhos.
2. Encontrar todos os vértices de G_p com no máximo uma aresta ($\{v_i | \exists v_j; (i;j) \in E_p, A_p(i;j) \leq 1\}$) e removê-los, além de suas correspondências de G_p e G'_p . Recalcular ambos G_p e G'_p . Repetir este passo até que todos os vértices de G_p tenham no mínimo duas arestas.
3. Uma matriz ponderada W é criada para cada ponto p_i usando o grafo G_p . Para a aresta que conecta o vértice v_i a v_m , computar um peso usando a seguinte Equação:

$$W(i;m) = \frac{0 + \cos\left(\frac{(p_m - p_i) \cdot (p'_m - p'_i) \text{Rot}(q(k_{\min};i))}{\|p_m - p_i\| \|p'_m - p'_i\|}\right)}{A} \quad (3.5)$$

em que

$$\text{Rot}(q(k_{\min};i)) = \begin{pmatrix} \cos(q(k_{\min};i)) & \sin(q(k_{\min};i)) \\ -\sin(q(k_{\min};i)) & \cos(q(k_{\min};i)) \end{pmatrix} \quad (3.6)$$

Algoritmo 3: WGTM, adaptado de Izadi e Saeedi (2012).

Entrada: P, P' : conjuntos de pontos referentes à primeira e segunda imagens, respectivamente.

k : número de vizinhos para criação dos grafos k -nn medianos.

Saída: Subconjuntos de P e P' , contendo os inliers.

```

1  Função WGTM ( $P, P', k$ )
2     $m_{old} \leftarrow 2p$ 
3     $stopFlag \leftarrow 1$ 
4    enquanto  $stopFlag = 1$  faça
5      // Passo 1
6       $A_p \leftarrow \text{construirGrafoKNNMediano}(\text{dist } p, k, \text{median } p)$  // Passo 1
7       $A_{p'} \leftarrow \text{construirGrafoKNNMediano}(\text{dist } p', k, \text{median } p')$ 
8      // Passo 2
9      enquanto  $\forall i \in \{1; \dots; N\}; \{v_i | A_p(i; j) \leq 1; \forall j\} \neq \emptyset$  faça
10       Remover  $v_i$  de  $P$  assim como  $v_{i'}$  de  $P'$ .
11       Reduzir  $N$ .
12       Atualizar  $A_p$  e  $A_{p'}$ .
13     // Passos 3 e 4
14      $W \leftarrow \text{criarGrafoPonderado}(P, P', A_p, A_{p'})$  // Equação 3.5 e Equação 3.10
15     // Passo 5
16      $w(i) \leftarrow \text{média}(W(i; j)), \forall (i; j) \in E_p$  // Equação 3.11
17     // Passo 6
18      $w_{max} \leftarrow \max_{i; j} (W(i; j))$  // Equação 3.12
19      $m_{new} \leftarrow \text{média}_{\forall i} (w(i))$  // Equação 3.13
20     // Critério de parada
21     se  $w_{max} < p$  &  $|m_{new} - m_{old}| < e$  então
22        $stopFlag \leftarrow 0$ 
23     senão
24        $m_{old} \leftarrow m_{new}$ 
25        $outlier \leftarrow \underset{i}{\text{argmax}} (\underset{\forall j; A_p(i; j) = 1}{\text{média}} (W(i; j)))$ 
26       Remover vértice correspondente a outlier de  $P$  e  $P'$ 
27   retorna  $P$  e  $P'$ 

```

Aqui, k_{min} representa o ângulo ótimo de rotação entre cada par de correspondência. p_i e p_m são vetores contendo as coordenadas dos vértices v_i e v_m . O ângulo ótimo de rotação é aquele que minimiza a soma de distâncias angulares entre os vetores v_i e $v_{i'}$:

$$k_{min} = \underset{\forall k; (i; k) \in E_p \quad \forall j; (i; j) \in E_{p'}}{\text{argmin}} \quad \hat{A} \quad \text{arccos} @ \frac{(p_j - p_i) \cdot (p'_j - p'_i) \text{Rot}(q(k; i))}{\|p_j - p_i\| \|p'_j - p'_i\|} \quad A \quad (3.7)$$

e

$$q(k; i) = \text{Atan}_{p; p}(p_k - p_i) - \text{Atan}_{p; p}(p'_k - p'_i) \quad (3.8)$$

em que

$$\text{Atan}_{-p;p}(v) = \begin{cases} \text{Atan}(v_y=v_x); & v_x \geq 0 \\ \text{Atan}(v_y=v_x) + p; & v_y \geq 0; v_x < 0 \\ \text{Atan}(v_y=v_x) - p; & v_y < 0; v_x < 0 \end{cases} \quad (3.9)$$

v é um vetor 2D contento as coordenadas x e y , representadas por v_x e v_y . Neste ponto, a matriz ponderada W está preenchida com distâncias angulares entre vetores de vértices correspondentes e todas as suas conexões de vértices. Todos os elementos de W aos quais não foram atribuídos nenhum valor são considerados como nulo.

4. Para cada vértice v_i , é encontrada a porcentagem de arestas conectadas a v_i que também estão conectadas em v_i' . Se a porcentagem for menor que 50%, o peso das arestas diferentes recebem o valor p . Em outras palavras:

$$\forall i; j : W(i; j) = p; \text{ se } \frac{\sum_{\forall k; (i;k) \in E_p} \hat{A}_{p'}(i; j)}{\sum_{\forall k; (i;k) \in E_p} \hat{A}_p(i; j)} < 0.5 \quad (3.10)$$

5. Para cada vértice v_i de G_p , calcular a média dos pesos de todas as arestas conectadas a ele:

$$w(i) = \text{média}_{\forall j; (i;j) \in E_p} (W(i; j)) \quad (3.11)$$

O vértice com maior valor em w é removido de P e P' .

6. Encontrar o elemento com maior valor em W ($w_{\max}$).

$$w_{\max} = \max_{\forall i; j} (W(i; j)) \quad (3.12)$$

Atribuir

$$m_{\text{new}} = \text{média}_{\forall i} (w(i)) \quad (3.13)$$

O algoritmo termina se $w_{\max} < p$ e $|u_{\text{new}} - u_{\text{old}}| < e$; caso contrário, definir $m_{\text{new}} = m_{\text{old}}$ e proceder para a próxima iteração. O valor de e é definido empiricamente pelos autores do método WGTm como 0:001. O valor de m_{old} é definido inicialmente como $2p$.

3.2.4 Spatial Order Constraint Bilateral-neighbor Vote (SOCBV)

O método SOCBV (MENG; LI; PEI, 2015) também utiliza o conceito de grafos k -nn medianos, que foi descrito na subseção 3.2.1. Entretanto, as arestas possuem uma estrutura direcionada. Desta forma, a matriz de adjacências (A_p) provavelmente não será simétrica. Para facilitar a compreensão do método, os símbolos utilizados por Meng, Li e Pei (2015) foram adaptados.

O método SOCBV utiliza descritores de Spatial Order Constraint (SOC), detalhada na [subseção 3.2.4.1](#), os quais são usados para o cálculo da densidade k-nn de cada vértice, explanado na [subseção 3.2.4.2](#). A densidade k-nn é usada para identificar outliers. Entretanto, esta densidade k-nn é afetada à medida em que a quantidade de outliers aumenta. Para reduzir esse efeito, [Meng, Li e Pei \(2015\)](#) propuseram um conjunto de vizinhos bilaterais, como detalhado na [subseção 3.2.4.3](#). Após a definição destes conceitos básicos do algoritmo SOCBV, na [subseção 3.2.4.4](#) são mostrados os passos do algoritmo.

3.2.4.1 Descritor SOC

O descritor SOC de um dado vértice do grafo G_P , como exemplo v_i , é denotado por O_{v_i} e é obtido com a ordenação decrescente dos pesos de seus k-vizinhos mais próximos. De modo similar, o vértice correspondente a v_i , v'_i , possui um descritor SOC, denotado por $O'_{v'_i}$, que também é obtido com a ordenação decrescente dos pesos de seus k-vizinhos mais próximos. Para exemplificar, na [Figura 11](#) é mostrado um vértice, v_i , correspondido com v'_i , e seus respectivos k-vizinhos mais próximos. Para o vértice v_i ([Figura 11a](#)), o descritor SOC é $O_{v_i} = \{v_2; v_3; v_4; v_5; v_6; v_1\}$, e seu descritor correspondente, v'_i ([Figura 11b](#)), é $O'_{v'_i} = \{v'_2; v'_3; v'_4; v'_6; v'_1; v'_7\}$. O par de descritores SOC é então denotado por $(O_{v_i}; O'_{v'_i})$.

Figura 11 – Vértice v_i , correspondido com v'_i , e seus respectivos k-vizinhos mais próximos, onde $k = 6$.

Fonte: Adaptada de [Meng, Li e Pei \(2015\)](#).

3.2.4.2 Estimativa de densidade k-nn

Inicialmente, uma matriz binária de classificação, representada por MBC, é criada para cada vértice do grafo G_P , utilizando seus descritores SOC. A matriz referente ao grafo G_P , MBC_P , é preenchida usando a [Equação 3.14](#), sendo que 0 representa um outlier e 1 representa um inlier. Inicialmente, todos os pontos da matriz são definidos como inliers.

Matriz binária de classificação referente ao grafo G_P . (MBC_P)

$$MBC_P(i; j) = \begin{cases} 1; & \text{Se } \text{pos}(O_{v_i}; j) = \text{pos}(O_{v'_i}; j) \\ 0; & \text{caso contrário} \end{cases} \quad (3.14)$$

a função $\text{pos}(O_{v_i}; j)$ informa a posição de j em O_{v_i} , igualmente para $\text{pos}(O_{v'_i}; j)$. Note que quando $MBC_P(i; j) = 1$, v_j é considerado como inlier com relação a v_i . Entretanto, observe que v_j pode assumir diferentes classificações, dependendo do seu conjunto de vizinhos.

Após o cálculo da matriz binária de classificação, a eliminação de outliers pode ser vista como uma estimativa de probabilidade a posteriori:

$$p(\text{classe}|m_i) = \frac{p(m_i; \text{classe})}{p(m_i)} = \frac{p(m_i; \text{classe})}{\sum_{a=0}^C p(m_i; \text{classe})} \approx \frac{k_a}{k} \quad (3.15)$$

sendo que $\text{classe} = \{\text{in}; \text{out}\}$, out representa a classe de outlier, in denota a classe de inlier e m_i representa relação de correspondência entre p_i e p'_i , isto é, $m_i = (p_i; p'_i)$.

Além disso, k_a é a quantidade de classes já classificadas dentro do conjunto de vizinhos de m_i . Desta forma, a probabilidade a posteriori de m_i pode ser reescrita como na [Equação 3.16](#).

$$\hat{p}(\text{in}|m_i) = \frac{1}{n_i} \sum_{m_j \in N_P(v_i)} \hat{A} \cdot MBC_P(i; j) \quad (3.16)$$

sendo n_i a quantidade de vizinhos de saída de m_i no grafo k-nn mediano.

Na [Equação 3.16](#) tem-se então, para m_i , a razão entre a quantidade de inliers na matriz binária de classificação e o total de vizinhos de saída no grafo k-nn mediano. Isto implica que a probabilidade a posteriori de um inlier depende fortemente do seu conjunto de vizinhos (linha sólida em vermelho na [Figura 12](#)), podendo ser afetada facilmente por estes. Portanto, quando existem erros na matriz binária de classificação causados por outliers, a probabilidade a posteriori será imprecisa. Com o objetivo de amenizar estes erros, um conjunto de vizinhos bilaterais é definido como descrito na [subseção 3.2.4.3](#).

3.2.4.3 Vizinhos bilaterais

Inicialmente são definidos, para o grafo G_P , os conjuntos de vizinhos de entrada ($N_P^{\text{in}}(m_i) = \{m_j | A_P(i; j) = 1\}$) e de saída ($N_P^{\text{out}}(m_i) = \{m_j | A_P(j; i) = 1\}$). Desta forma, o conjunto de vizinhos bilaterais (entrada e saída) do par de correspondências m_i é representado por:

$$N_P(m_i) = \{N_P^{\text{in}}(m_i); N_P^{\text{out}}(m_i)\} \quad (3.17)$$

Vale notar que o conjunto de vizinhos bilaterais, $N_P(m_i)$, pode conter elementos repetidos.

Figura 12 – Diagrama do princípio do vizinho-bilateral.

Fonte: Adaptada de Meng, Li e Pei (2015).

Dado este conjunto de vizinhos bilaterais e a matriz binária de classificação, m_j pode ser considerado como um inlier, por associação a m_i , ou seja, $\forall \text{MBC}_P(i; j) = 1 \vee \text{MBC}_P(j; i) = 1$; $m_i, m_j \in \text{in}$. Este conjunto de inliers associados a m_i é representado como $Z_P(m_i)$:

$$Z_P(m_i) = \{m_j | m_j \in \{N_P(m_i); m_i\} \wedge \text{MBC}_P(i; j) = 1\} \quad (3.18)$$

nota-se que $v_i \in Z_P(m_i)$.

Nota-se que existem duas restrições no conjunto de inliers associativos: (1) restrição de vizinhos-bilaterais: garante que existe estrutura similar de vizinhos entre uma correspondência e seu conjunto de inliers associativos. (2) restrição de classe de inlier: garante que uma correspondência e seu conjunto de inliers associativos são todas da classe de inlier dentro do SOC.

Com estas duas restrições, a probabilidade posterior de um inlier entre uma correspondência e seu conjunto de inliers associativos são aproximados: $\forall m_j \in Z_P(m_i); \hat{p}(c_1 | m_i) \approx \hat{p}(c_1; m_j)$. Desta forma, o voto de vizinho-bilateral SOC é definido como segue:

$$\begin{aligned} S_P(i) &= \frac{1}{d_i} \sum_{m_j \in Z_P(m_i)} \hat{A} \hat{p}(\text{in}; m_j) \\ &= \frac{1}{d_i} \left(p(\text{in} | m_i) + \sum_{m_j \in N_P(m_i) \cap m_j \in \text{in}} \hat{A} p(\text{in} | m_j) \right) \end{aligned} \quad (3.19)$$

onde d_i é o grau do par de correspondência m_i e $S_P(i)$ é chamado de voto de vizinho-bilateral SOC de m_i .

A partir da Equação 3.19 observa-se que o voto de vizinho-bilateral SOC calcula a média de todas as probabilidades posteriores de inliers do conjunto de vizinhos associados (como é mostrado na Figura 12 todos os círculos tracejados em preto). Desta forma, uma nova estimativa da probabilidade posterior de inlier é utilizada. Em circunstâncias extremas, quando o conjunto

associativo de inliers apenas inclui ele mesmo, ou seja, $Z_P(m_i) = m_i$, o voto de vizinho-bilateral SOC será degradado para a probabilidade posterior de inlier baseado no conjunto de vizinhos de saída.

Na [Figura 13](#) é mostrado um exemplo simples para melhor ilustrar o voto de vizinho-bilateral SOC, sendo $(P; P') = \{(x_1; y_1); (x_2; y_2); (x_3; y_3); (x_4; y_4); (x_5; y_5)\}$ é o conjunto de pontos inicialmente correspondidos com $(x_3; y_3)$ como outlier. Os quatro vizinhos direcionados mais próximos de $(x_3; y_3)$ são mostrados na [Figura 13](#). O descriptor SOC do par $(x_3; y_3)$ são $O_{x_3} = \{1; 2; 5; 4\}$ e $O'_{y_3} = \{1; 2; 5; 4\}$, respectivamente. Note que os descriptors são iguais, então a probabilidade a posteriori de m_3 será 1 ($p(in|m_3) = 1$). A probabilidade para os demais pontos é 0;5. Portanto, baseando-se apenas na [Equação 3.16](#), o ponto $(x_3; y_3)$ seria considerado como um inlier, enquanto os demais pontos serão considerados como outliers. Em contraste, usando a [Equação 3.19](#), $S_P = \{0; 389; 0; 389; 0; 333; 0; 389; 0; 389\}$. Nota-se que o ponto $(x_3; y_3)$ apresenta a menor probabilidade, podendo assim ser removido.

Figura 13 – Ilustração do algoritmo SOCBV e seus respectivos grafos k-nn medianos.

Fonte: Adaptada de [Meng, Li e Pei \(2015\)](#).

3.2.4.4 SOCBV

Utilizando os conceitos descritos nas seções anteriores, o algoritmo SOCBV é resumizado no [Algoritmo 4](#) e possui os seguintes passos:

Passo 1: O primeiro e mais importante passo é o cálculo preciso da matriz binária de classificação (MBC). O descriptor SOC é obtido na forma de string, podendo assim ser transformado no problema da maior subsequência comum, do inglês, Longest Common Subsequence (LCS). Desta forma, a matriz MBC_P é calculada pela [Equação 3.20](#).

$$MBC_P(i; j) = \begin{cases} 8 & \\ < 1; & \text{Se } j \in L_{m_i} \\ 0; & \text{caso contrário} \end{cases} \quad I_{m_i} = ILCS(O_{v_i}; O'_{v'_i}) \quad (3.20)$$

sendo I_{m_i} a LCS entre O_{v_i} e $O'_{v'_i}$, que é calculada baseada em uma técnica incremental de comparação de strings ([LANDAU et al., 1995](#)). Vale notar que se os elementos do descriptor SOC

Algoritmo 4: SOCBV, adaptado de [Meng, Li e Pei \(2015\)](#).

Entrada: P, P' : conjuntos de pontos referentes à primeira e segunda imagens, respectivamente.

k : número de vizinhos para criação dos grafos k -nn medianos.

b : critério de parada do algoritmo.

Saída: inliers: que é subconjunto de P e P' contendo os inliers.

```

1 Função SOCBV ((P, P', k, b)
2   iterFlag ← true
  // Passo 1
3   MBCP; MBCP' ← criarGrafoKNN(P, P', k)           // Equação 3. 20
4   W ← criarPesos(P, P')                             // Equação 3. 22
5   SP ← obterSOCBV(MBCP, W, k)                     // Equação 3. 23
6   SP' ← obterSOCBV(MBCP', W, k)                  // Equação 3. 23
  // Passo 2
7   S ← (SP + SP') = 2                               // Equação 3. 24
8   jout ← encontrarOutlier(S)
9   enquanto iterFlag faça
10    remover jout de (P, P')
    // Atualizando grafos após a remoção de uma correspondência
11    MBCP, MBCP' ← atualizarGrafoKNN(MBCP, MBCP', P, P', k)
12    SP ← atualizarSOCBV(MBCP, W, k)
13    SP' ← atualizarSOCBV(MBCP', W, k)
14    S ← (SP + SP') = 2
    // Passo 3: Critério de parada
15    se 1 – min(S) ≤ b então
16      iterFlag ← false
17    senão
18      jout ← encontrarOutlier(S)                     // Equação 3. 25
19  retorna inliers

```

for menor que quatro, a LCS será sempre igual a 4 (usando a técnica incremental de comparação). Com o objetivo de eliminar este impasse, a LCS é definida como vazia quando a quantidade de elementos no descritor SOC for menor que quatro:

$$\forall \hat{A} \quad \forall j; (i;j) \in E_P \quad A_P(i;j) < 4; I_m = \emptyset \quad (3.21)$$

Passo 2: Uma vez que a MBC é calculada, o descritor de voto de vizinho-bilateral SOC pode ser obtido a partir da [Equação 3.19](#). Embora esta equação apresente uma probabilidade aproximada, ainda existem diferenças entre a probabilidade posterior de um inlier. Considerando essas diferenças, uma função kernel é utilizada como peso para a probabilidade posterior entre

uma correspondência e seu conjunto de inliers associados:

$$w(i; j) = \exp\left(-\frac{\|x_i - x_j\|_2}{h_P} - \frac{\|y_i - y_j\|_2}{h_{P'}}\right) \quad (3.22)$$

onde h_P e $h_{P'}$ são as distâncias medianas dos grafos G_P e $G_{P'}$, respectivamente. Assim, o voto de vizinho-bilateral SOC pode ser recalculado como na [Equação 3.23](#).

$$S_P(i) = \frac{\sum_{m_j \in \{Z_X(m)\}} \hat{A} w(i; j) \cdot \hat{p}(c_1|m_j)}{\sum_{m_j \in \{Z_X(m)\}} \hat{A} w(i; j)} \quad (3.23)$$

Devido à natureza assimétrica da matriz de adjacências, os dois votos de vizinho-bilateral SOC de uma correspondência são usualmente diferentes. Para eliminar este efeito de assimetria, o SOC bilateral final será a média entre os dois:

$$S(i) = \frac{S_P(i) + S_{P'}(i)}{2} \quad (3.24)$$

Passo 3: A cada iteração, um outlier é encontrado e removido de acordo com a [Equação 3.25](#).

$$j^{\text{out}} = \underset{m_j}{\text{argmin}} S(j) \quad (3.25)$$

Após a remoção de um outlier, os grafos k-nn são gerados novamente, baseados nos inliers restantes. O algoritmo termina quando $1 - \min(S) < b$, sendo b um limiar definido pelo usuário.

3.3 Considerações finais

O processo de corresponder, aproximadamente, duas representações de dados abstratos é uma tarefa complexa e tem sido muito estudado no campo de visão computacional. Métodos estatísticos como RANSAC apresentam bons resultados, porém falham com a presença de ruídos nos dados. Métodos baseados em grafos apresentam bons resultados, porém possuem um alto custo computacional, sendo inviável para aplicações que exigem tempo real de processamento. Com o intuito de reduzir o custo computacional, uma nova abordagem, que filtra uma parcela de outliers do conjunto de pontos inicialmente correspondidos é proposta e detalhada no capítulo seguinte.

FILTRANDO CORRESPONDÊNCIAS INCORRETAS

Os algoritmos de consenso detalhados no capítulo anterior normalmente eliminam uma correspondência incorreta (um outlier) por vez. Isto implica que, quanto maior a quantidade de outliers, maior será a quantidade de iterações necessária para o algoritmo chegar a um resultado, acarretando em um maior custo de processamento. Com o intuito de filtrar uma parte dos outliers antes do processamento pelos algoritmos de consenso, neste capítulo é proposto o método FOMP (do inglês, Filtering out Outliers From Matched Points), cujo objetivo é reduzir a quantidade de outliers no conjunto de pontos inicialmente correspondidos entre duas imagens.

4.1 Filtering out Outliers from Matched points (FOMP)

O algoritmo FOMP filtra pontos que possuem uma diferença consideravelmente grande entre os pesos das arestas dos dois conjuntos de pontos, modelados como dois grafos completos. Um vértice que está conectado a um outlier normalmente apresenta uma diferença de pesos maior entre os dois grafos. Contudo, este vértice pode também estar conectado a um inlier. Para sumarizar esta diferença, é realizada a soma das diferenças de todas as arestas conectadas a um vértice. Desta forma, o vértice que apresentar a maior soma de diferenças é iterativamente removido. Na [Figura 14](#) pode ser visto um fluxograma da metodologia desenvolvida.

Sejam dois conjuntos de pontos, em que os pontos na primeira imagem são definidos como $P = \{p_1; p_2; \dots; p_n\}$, sendo $p_i = (x; y)$ e n o total de pontos. De forma similar para a segunda imagem, $P' = \{p'_1; p'_2; \dots; p'_n\}$, e p'_i está correspondido com o ponto p_i . Na [Figura 15](#) é mostrado um exemplo de dois conjuntos de pontos correspondidos.

A tarefa é encontrar um subconjunto de P e P' que apresenta um consenso espacial (consistência geométrica), ou seja, ambos os conjuntos devem apresentar uma forma similar, mesmo se existir diferenças de rotação, escala e transformação afim entre eles. No exemplo da

Figura 14 – Metodologia desenvolvida.

Fonte: Elaborada pelo autor.

Figura 15 – Conjunto de pontos correspondidos. O ponto p_i está correspondido com o ponto p'_i . O ponto p_5 é um outlier

Fonte: Elaborada pelo autor.

Figura 15 pode-se notar que p_5 é um outlier. O método FOMP filtra outliers em três passos:

Passo 1: Uma representação dos pontos na forma de grafo completo com pesos é criada para cada um dos dois conjuntos de pontos, $G_p(V_p; E_p)$, em que $V_p = \{v_1; \dots; v_n\}$ representa coordenadas de pontos em P . De forma similar para os pontos da segunda imagem, $G_{p'}(V_{p'}; E_{p'})$, em que $V_{p'} = \{v'_1; \dots; v'_n\}$ representa coordenadas de pontos em P' , como mostra a Figura 16.

Figura 16 – Cada conjunto de ponto é representado como um grafo completo.

Fonte: Elaborada pelo autor.

Cada grafo é então representado por uma matriz de adjacências de tamanho $n \times n$, W para o primeiro grafo e W' para o segundo. O conjunto de arestas em cada grafo, E_p e $E_{p'}$, é dado pela conexão de todos os vértices para todos os vértices. A conexão do vértice i para o vértice j , ou vice-versa, é dada pela distância euclidiana (Equação 4.1) entre suas coordenadas de ponto, como mostra a Figura 17.

$$W(i; j) := ||p_i - p_j||; \forall i, j | 1 \leq i, j \leq n \quad (4.1)$$

Figura 17 – Os pesos das arestas são dados pela distância (Equação 4.1) entre os vértices.

Fonte: Elaborada pelo autor.

Passo 2: Para minimizar a influência da escala da imagem, todos os pesos são normalizados (Equação 4.2) por sua própria média, como mostra a Figura 18.

$$W_m := \frac{W}{\text{média}(W(i; j))}; \forall i, j \in E_p \quad (4.2)$$

Figura 18 – O peso de cada grafo é normalizado por seu valor médio (Equação 4.2).

Fonte: Elaborada pelo autor.

Passo 3: As diferenças entre as arestas correspondentes nos dois grafos são somadas e normalizadas pela quantidade de vértices:

$$D(i) := \frac{1}{n} \sum_{j=1}^n |W_m(i; j) - W'_m(i; j)|; 1 \leq i \leq n: \quad (4.3)$$

A Figura 19 mostra a aplicação da Equação 4.3 para um único vértice. Após o cálculo para todos os vértices, o valor máximo ($d_{\max}$) é calculado:

$$d_{\max} := \max_{i=1::n} D(i) \quad (4.4)$$

Figura 19 – Exemplo de aplicação da Equação 4.3 para um único vértice.

Fonte: Elaborada pelo autor.

Aplicando a Equação 4.3 para todos os vértices do exemplo da Figura 19, tem-se $D = \{0,28; 0,21; 0,21; 0,28; 0,58\}$. Nota-se que p_5 apresenta o maior valor ($d_{\max}$) e portanto é removido, se $d_{\max}$ for maior que o limiar a . O valor de a é definido para 0,5 para todos os experimentos realizados neste trabalho. O valor de a foi encontrado empiricamente, sendo o valor encontrado razoável para as quatro classes de imagens utilizadas nos experimentos. Após a remoção de uma correspondência de W e W' , o algoritmo retorna ao passo 2. Este processo é repetido até que o valor de $d_{\max}$ não seja maior que a . No Algoritmo 5 é mostrado o pseudo-código do algoritmo FOMP.

4.1.1 Complexidade: tempo de processamento

Analisando o Algoritmo 5, podemos notar que, a criação do grafo completo (linhas 2 e 3) tem uma complexidade de ordem quadrática ($O(n^2)$), uma vez que o cálculo da distância euclidiana é feito entre todos os pares de vértices. A normalização da matriz de adjacências pela sua própria média (linhas 6 e 7) depende da quantidade de arestas na rede, normalmente $(n * (n -$

Algoritmo 5: Filtering out Outliers from Matched Points – FOMP.

Entrada: P, P' : conjuntos de pontos referentes à primeira e segunda imagens, respectivamente.
 a : critério de parada, $[0; 1]$.
 Saída: inliers: subconjunto de P e P' contendo os inliers.

```

1  Função FOMP ( $P, P', a$ )
    // Passo 1
2   $W \leftarrow ||p_i - p_j||; \forall i, j$  // Equação 4.1
3   $W' \leftarrow ||p'_i - p'_j||; \forall i, j$  // Equação 4.1
4   $flag \leftarrow true$ 
5  enquanto  $flag$  faça
    // Passo 2
6   $W_m \leftarrow \frac{W}{média(W(i; j))}; \forall i, j > i$  // Equação 4.2
7   $W'_m \leftarrow \frac{W'}{média(W'(i; j))}; \forall i, j > i$  // Equação 4.2
    // Passo 3
8   $D(i) \leftarrow \frac{1}{n} \sum_{j=1}^n |W_m(i; j) - W'_m(i; j)|; \forall i$  // Equação 4.3
9   $d_{max} \leftarrow \max_{i=1::n} D(i)$  // Equação 4.4
    // Critério de parada
10 se  $d_{max} > a$  então
11    $flag \leftarrow false$ 
12 senão
13   remover vértice correspondente a  $d_{max}$  de  $W, W', P$  e  $P'$ 
14 retorna  $P$  e  $P'$ 

```

1))=2 considerando a matriz de adjacências simétrica, o que leva a $O(n^2)$. A soma das diferenças entre os vértices (linha 8) tem uma complexidade de $O(n^2)$. Na linha 9, para encontrar o valor máximo de D é necessário percorrer todo o vetor, o que leva a uma complexidade de $O(n)$. O laço de repetição while depende do parâmetro a . Contudo, considerando o pior caso (o algoritmo itera n vezes removendo um outlier por vez), a complexidade será $O(1^2 + 2^2 + \dots + n^2) = O(n^3)$.

4.2 Considerações mais

Apesar do algoritmo FOMP apresentar uma ordem de complexidade consideravelmente alta ($O(n^3)$), os algoritmos de consenso, tais como GTM, WGTM e SOCBV sempre calculam grafos completos com o objetivo de gerar grafos k-nn. Desta forma, os grafos completos calculados pelo algoritmo FOMP podem ser reutilizados dentro desses algoritmos, o que reduz o tempo de processamento global, uma vez que os grafos completos passados a esses algoritmos usando FOMP possuem uma quantidade menor de outliers. Isto faz com que os algoritmos de consenso necessitem de uma quantidade menor de iterações. Além do mais, o algoritmo FOMP necessita apenas de alguns passos simples. O passo mais custoso do algoritmo é o cálculo dos grafos completos.

EXPERIMENTOS, RESULTADOS E DISCUSSÕES

Para validar o desempenho do algoritmo FOMP, foram realizados testes comparativos com e sem pré-processamento para os métodos RANSAC ([FISCHLER; BOLLES, 1981](#)), GTM ([AGUILAR et al., 2009](#)), WGTm ([IZADI; SAEEDI, 2012](#)) e SOCBV ([MENG; LI; PEI, 2015](#)). A implementação do método RANSAC usada está disponível em ([KOVESI, 2009](#)). As implementações dos métodos GTM e WGTm foram generosamente cedidas por [Izadi e Saeedi \(2012\)](#). O método SOCBV também foi generosamente cedido por [Meng, Li e Pei \(2015\)](#). A análise comparativa foi realizada usando as medidas de acurácia, precisão, revocação e especificidade.

$$\text{Acurácia} = \frac{VP + VN}{VP + FP + VN + FN} \quad (5.1)$$

$$\text{Precisão} = \frac{VP}{VP + FP} \quad (5.2)$$

$$\text{Revocação} = \frac{VP}{VP + FN} \quad (5.3)$$

$$\text{Especificidade} = \frac{VN}{VN + FP}; \quad (5.4)$$

em que:

- Verdadeiro positivo (VP): inlier corretamente detectado como inlier;
- Falso positivo (FP): outlier incorretamente identificado como inlier;
- Verdadeiro negativo (VN): outlier corretamente detectado como outlier;
- Falso negativo (FN): inlier incorretamente identificado como outlier.

A acurácia, [Equação 5.1](#), mede a porcentagem de correspondências corretamente detectadas, incluindo verdadeiros positivos (VP) e verdadeiros negativos (VN). Na [Equação 5.2](#), a precisão avalia a porcentagem de inliers (VP) corretamente detectados, incluindo os pontos incorretamente detectados como inliers (FP). A revocação, [Equação 5.2](#), verifica a porcentagem de inliers (VP) corretamente detectados dentre todos os inliers presentes. Na [Equação 5.4](#), a especificidade calcula a porcentagem de outliers (TN) corretamente detectados dentre todos os outliers presentes.

Os experimentos foram divididos em sete partes, resumidas na [Figura 20](#) e detalhadas como segue: Inicialmente, foram realizados experimentos com quatro base de imagens, as quais foram usados em outros experimentos por autores dos algoritmos GTM, WTGM e SOCBV. Estas bases apresentam algumas variações, tais como movimento/zoom de câmera, correspondências ambíguas, objetos deformados ([AGUILAR et al., 2009](#)) e variações afim ([MIKOLAJCZYK et al., 2005](#)). Para estes quatro experimentos foram analisadas as medidas de acurácia, precisão, revocação e especificidade, com e sem o pré-processamento (FOMP).

Figura 20 – Experimentos realizados.

Fonte: Elaborada pelo autor.

Na quinta parte, foi realizada a comparação do tempo de execução para as quatro bases de imagens supracitadas com e sem FOMP, com o intuito de validar o ganho na velocidade de processamento com o uso do filtro proposto. Finalmente, foi realizada uma análise quantitativa da quantidade de outliers filtrados (especificidade) e inliers mantidos (revocação) após aplicação do filtro FOMP, para todas as porcentagens de outliers nos experimentos envolvendo as quatro bases de imagens utilizadas neste trabalho.

Para todos os experimentos, inicialmente foram extraídos pontos de interesse utilizando o algoritmo SURF. A extração de características de cada ponto de interesse também é feita

usando o SURF. Os pontos de interesse são correspondidos entre pares de imagens usando seus respectivos vetores de características. Foram manualmente selecionados 60 inliers para cada par de imagem. Então foram introduzidas porcentagens de outliers de forma controlada, de modo que um outlier aleatoriamente gerado não fosse equivalente a um inlier. O número de outliers introduzidos, baseado em sua porcentagem, foi calculado de acordo com a [Equação 5.5](#) (IZADI; SAEEDI, 2012):

$$\text{outliers} = \frac{\text{porcentagem de outliers} \times \text{inliers}}{100 - \text{porcentagem de outliers}} \quad (5.5)$$

sendo inliers o número total de correspondências corretas e outliers a quantidade total de outliers introduzidos aleatoriamente. A porcentagem inicial de outliers adicionada foi 5%, com incrementos de 10%, até 95%. Na [Tabela 2](#) é mostrada a quantidade de outliers adicionados para cada porcentagem de outliers. Note que há uma diferença significativa na quantidade de outliers adicionados entre 85% e 95% de outliers.

Tabela 2 – Quantidade de outliers adicionados para cada porcentagem de outliers de acordo com a [Equação 5.5](#).

Outliers										
	5%	15%	25%	35%	45%	55%	65%	75%	85%	95%
Adicionados	3	11	20	32	49	73	111	180	340	1140

Este processo foi feito 20 vezes para cada porcentagem de outlier, totalizando $n \times 20 \times 10$ execuções para cada experimento, sendo n o total de pares de imagens em cada base de imagens.

Para todos os experimentos, foram usados os parâmetros definidos pelos autores de cada método. O valor k do método GTM foi definido para 5, enquanto que para o WGTM e SOCBV este valor foi definido para 20. A condição de parada do algoritmo WGTM (ϵ) foi definida para 0;001. No algoritmo SOCBV, o parâmetro b foi definido para 0;2. O método RANSAC o limiar de erro para definição de outliers foi definido como 0;1.

Os experimentos foram realizados em uma máquina desktop com processador Intel(R) Core(TM) i7 de 2.67 GHz, com 6 GB de memória RAM na plataforma Matlab. Considerando que os métodos GTM, WGTM e SOCBV também utilizam matrizes representando grafos completos, as matrizes de adjacências geradas no algoritmo FOMP foram reutilizadas nesses algoritmos com a finalidade de reduzir o tempo de processamento e evitando o processamento redundante de informação.

5.1 Movimentos de câmera

Para efetuar experimentos considerando movimentação de câmera, foi usada uma base contendo 10 pares de imagens, que apresentam diferenças de translação, rotação e escala. Estas

imagens possuem uma resolução de 640×480 pixels. Os resultados comparativos para os métodos com e sem o pré-processamento FOMP são mostrados na Figura 21, em que foram empregadas as medidas de acurácia (Equação 5.1), precisão (Equação 5.2), revocação (Equação 5.3) e especificidade (Equação 5.4) para a análise.

Figura 21 – Comparação de medidas com e sem o pré-processamento FOMP, considerando as porcentagens de outliers manualmente adicionados para a base de imagens com movimentos de câmera. A legenda no gráfico de precisão é também válida para os demais gráficos.

Fonte: Dados da pesquisa.

Como mostrado na Figura 21, o resultado para a acurácia foi equivalente ou melhor com o uso do filtro FOMP. Para o método GTM, o desempenho com o FOMP foi consideravelmente melhor. Para o método WGTM os resultados foram levemente melhores até 45% de outliers. Analisando a precisão, todos os métodos apresentaram resultados semelhantes de 5% até 85% de outliers. Em 95% de outliers, os métodos SOCBV e WGTM apresentaram melhores resultados sem o FOMP. Por outro lado, o método RANSAC apresentou melhores resultados com o FOMP, enquanto o GTM apresentou resultado equivalente. Para a revocação, todos os métodos apresentaram resultados similares ou melhores com o uso do filtro FOMP, com exceção do método GTM, o qual apresentou uma melhora considerável com o uso do filtro. A especificidade, apresenta diferenças pequenas para todas as porcentagens de outlier. Porém, vale notar que o método SOCBV obteve melhores resultados de 5% até 65% de outliers com o FOMP.

Na Figura 22 podem ser vistos os resultados comparativos com/sem FOMP para um único par de imagens desta base de imagens. Neste exemplo foi considerado 85% de outliers no conjunto de pontos inicialmente correspondidos.

Figura 22 – Exemplo de resultados para pares de imagens com 85% de outliers para as base de imagens: Movimentos de câmara.

Fonte: Dados da pesquisa.

É possível notar na [Figura 22](#) que, para os métodos GTM e RANSAC, o filtro FOMP leva os métodos a detectar uma maior quantidade de correspondências corretas (VP). Para o método WGTM, foi detectada uma maior quantidade de outliers sem o uso do filtro FOMP. Entretanto, a quantidade de falsos positivos (FP) foi menor. Portanto, neste exemplo, o uso do filtro FOMP apresenta uma melhor precisão. Para o método SOCBV, a quantidade de verdadeiros positivos (VP) foi ligeiramente maior com o FOMP, enquanto que a quantidade de falsos positivos foi maior também, o que pode ser compensado pelo ganho no tempo de processamento (ganho de 71%).

Analisando o tempo de processamento dos métodos GTM e WGTM, foi possível notar que há um grande ganho, em média, 73%. Por outro lado, o método RANSAC não apresentou nenhum ganho de tempo de processamento, em comparação com o FOMP.

5.2 Correspondências ambíguas

Para efetuar experimentos considerando ambiguidade, foi usada uma base contendo 10 pares de imagens, que apresentam padrões repetitivos, que acarretam em ambiguidades na correspondência entre os pontos devido à duplicidade de alguns vetores de características. Estas imagens possuem uma resolução de 640×480 pixels. Os resultados comparativos para os métodos com e sem o pré-processamento FOMP são mostrados na Figura 23, em que são empregadas as medidas de acurácia, precisão, revocação e especificidade para a análise.

Figura 23 – Comparação de medidas com e sem o pré-processamento FOMP, considerando as porcentagens de outliers manualmente adicionados para a base de imagens com correspondências ambíguas. A legenda no gráfico de precisão é também válida para os demais gráficos.

Fonte: Dados da pesquisa.

Como exibido na Figura 23, para esta base de imagens, novamente a acurácia apresentou resultado similar ou melhor com o uso do filtro FOMP, com uma acentuada melhora para o método GTM. Para a precisão, nenhum método apresentou diferenças de desempenho de 5% até 65% de outliers. O método RANSAC apresentou uma precisão menor sem o filtro FOMP de 65% até 95% de outliers, enquanto os demais métodos não apresentaram diferenças relevantes nessa mesma faixa de porcentagem de outliers. Analisando a revocação, nota-se que o método GTM

Figura 24 – Exemplo de resultados para pares de imagens com 85% de outliers para as base de imagens: Correspondências ambíguas.

Fonte: Dados da pesquisa.

apresentou resultados consideravelmente melhores com FOMP para todas as porcentagens de correspondências incorretas. De 45% a 95% houve uma melhora considerável para o método RANSAC com o uso do filtro. Para os demais métodos, os resultados foram similares de 5% até 75% de outliers. A especificidade apresenta diferenças pequenas para todas as porcentagens de outliers. Porém, para o método F-RANSAC houve uma pequena melhora nos resultados de 5%

até 55% de outliers.

Na [Figura 24](#) podem ser vistos os resultados comparativos com/sem FOMP para um único par de imagens desta base de imagens. A quantidade de VP e FP para os métodos GTM e RANSAC é consideravelmente melhor (todos os inliers (VP) e nenhum FP foram detectados com o FOMP), o que leva a uma melhor precisão. Os métodos WGTM e SOCBV apresentaram uma quantidade de VP e FP similar. Entretanto, houve um considerável ganho no tempo de processamento com o uso do FOMP, em média, 80%. Para o método GTM também houve um ganho no tempo de processamento, cerca de 82%. Apesar de que não houve nenhum ganho no tempo de processamento para o RANSAC, a diferença é de apenas 0;01 segundos no tempo de processamento.

5.3 Objetos deformados

Para efetuar experimentos considerando objetos não-rígidos ou deformados, foi usada uma base contendo 10 pares de imagens, que apresentam objetos deformados, tais como papéis ou embalagens de plástico. Estas imagens possuem uma resolução de 640×480 pixels. Os resultados comparativos para os métodos com e sem o pré-processamento FOMP são mostrados na [Figura 25](#), em que são empregadas as medidas de acurácia, precisão, revocação e especificidade para a análise.

Como mostrado na [Figura 25](#), a acurácia apresentou resultados melhores com o filtro FOMP para o método GTM, enquanto os demais métodos não apresentaram diferenças relevantes. Para a precisão, o método RANSAC apresenta uma perda gradativa sem o filtro FOMP a partir de 25% até 95% de outliers. Analisando a revocação, o método GTM apresentou resultados melhores com o FOMP para todas as porcentagens de correspondências incorretas. O RANSAC apresentou revocação levemente melhor com FOMP. Para os demais métodos, não houve melhor revocação de 5% até 75% de outliers. Em 85% de outliers, a revocação foi levemente melhor para o método F-SOCBV. Em 95% de correspondências incorretas, todos os métodos apresentaram melhores resultados com FOMP. Verificando a especificidade, apesar das diferenças serem pequenas, pode-se notar que, em geral, todos os métodos apresentaram especificidade similares ou melhores com FOMP.

Na [Figura 26](#) podem ser vistos os resultados comparativos com/sem FOMP para um único par de imagens desta base de imagens. Nota-se que a quantidade de VP é maior quando usado o filtro FOMP. Para os métodos WGTM e SOCBV a quantidade de FP é menor com FOMP, enquanto que esta quantidade é maior para os métodos GTM e RANSAC. O ganho no tempo de processamento é consideravelmente melhor para os métodos GTM, WGTM e SOCBV, em média, 72% de ganho.

Figura 25 – Comparação de medidas com e sem o pré-processamento FOMP, considerando as porcentagens de outliers manualmente adicionados para a base de imagens com objetos deformados. A legenda do gráfico precisão é também válida para os demais gráficos.

Fonte: Dados da pesquisa.

5.4 Transformações afins

A base usada neste experimento é composta por 5 pares de imagens, que apresentam transformações afim devido a diferentes pontos de vista (MIKOLAJCZYK et al., 2005). Os resultados comparativos para os métodos com e sem o pré-processamento FOMP são mostrados na Figura 27, em que são empregadas as medidas de acurácia, precisão, revocação e especificidade para a análise.

Analisando a Figura 27, a acurácia apresentou resultados semelhantes para o método SOCBV. O método GTM apresentou uma grande melhora com o uso do FOMP, já o método WGTM apresentou uma diferença pequena sem o FOMP. Para a precisão, todos os métodos apresentaram resultados semelhantes de 5% a 75% de correspondências incorretas. De 85% a 95%, o método RANSAC apresentou uma queda sem o filtro, enquanto as diferenças para os demais métodos não foram relevantes. Analisando a revocação, o método GTM apresentou desempenho melhores com FOMP para todas as porcentagens de correspondências incorretas. O método WGTM apresentou pequenas melhoras sem o FOMP. Os métodos RANSAC e SOCBV apresentaram resultados similares ou melhores com o filtro FOMP para todas as porcentagens de correspondências incorretas. A especificidade não apresentou diferenças relevantes para os métodos GTM, SOCBV. O método WGTM apresentou uma pequena melhora com FOMP de 15% a 45% de correspondências incorretas. Enquanto o método RANSAC apresentou um

Figura 26 – Exemplo de resultados para pares de imagens com 85% de outliers para as base de imagens: Objetos deformados.

Fonte: Dados da pesquisa.

melhora com o filtro para todas as porcentagens de outliers

Na [Figura 28](#) podem ser vistos os resultados comparativos com/sem FOMP para um único par de imagens desta base de imagens. Note que, para os métodos GTM, SOCBV e RANSAC, o uso do filtro FOMP apresentou uma quantidade maior de VP e uma quantidade menor de FP. Para o método WGTM, a quantidade de VP é a mesma, enquanto a quantidade de FP é um pouco menor. O ganho no tempo de execução é consideravelmente melhor com FOMP para os métodos GTM, WGTM e SOCBV, apresentando em média, 77% de ganho. Em contraste, o método RANSAC não apresentou ganho positivo de tempo de execução. Entretanto, a quantidade de VP e FP é muito melhor com FOMP, chegando a 100% de precisão e revocação

Figura 27 – Comparação de medidas com e sem o pré-processamento FOMP, considerando as porcentagens de outliers manualmente adicionados para a base de imagens com transformações afim. A legenda do gráfico de precisão é também válida para os demais gráficos.

Fonte: Dados da pesquisa.

neste exemplo.

5.5 Tempo de execução

O tempo de execução para os quatro experimentos, movimentos de câmera, correspondências ambíguas, objetos deformados e transformações afim, é sumarizado na [Tabela 3](#). Note que há um grande ganho no tempo de processamento quando usado o filtro FOMP com os métodos GTM, WGTM e SOCBV para todas as porcentagens de outliers. O ganho médio para os métodos GTM e SOCBV é de 50%, enquanto que para o método WGTM este ganho é de quase 100%. Para o método RANSAC não há ganho positivo, uma vez que este não utiliza as matrizes representativas dos grafos completos calculados dentro do método FOMP. Entretanto, como foi mostrado nos experimentos, a quantidade de correspondências corretas encontradas com o uso do filtro FOMP no método RANSAC é maior.

5.6 Análise isolada do FOMP

Com o fim de analisar somente o comportamento do filtro FOMP, foram verificadas as porcentagens de correspondências corretas mantidas (revocação) e a porcentagem de correspondências incorretas filtradas (especificidade).

Figura 28 – Exemplo de resultados para pares de imagens com 85% de outliers para as base de imagens: Transformações afim.

Fonte: Dados da pesquisa.

5.6.1 Revocação e especificidade

A Tabela 4 sumariza o tempo de execução, quantidade de iterações, especificidade e revocação para os quatro experimentos, movimentos de câmera, correspondências ambíguas, objetos deformados e transformações afim. Note que a revocação é igual ou superior a 90% para todas as porcentagens de correspondências incorretas. Isto significa que o filtro FOMP mantém a maior parte das correspondências corretas, mesmo com diferentes porcentagens de correspondências incorretas. Além do mais, para porcentagens pequenas de correspondências incorretas, de 5% a 15%, o filtro consegue manter praticamente todas as correspondências corretas, enquanto

Tabela 3 – Comparação do tempo médio de execução para os quatro experimentos: movimentos de câmera, correspondências ambíguas, objetos deformados e transformações afim. O tempo é expressado em segundos.

		GTM	F-GTM	WGTM	F-WGTM	SOCBV	F-SOCBV	RANSAC	F-RANSAC
Outliers (%)	5	0,03	0,025	1,44	0,080	0,04	0,054	0,248	0,296
	15	0,05	0,029	2,12	0,089	0,09	0,080	0,266	0,408
	25	0,07	0,042	3,23	0,097	0,15	0,100	0,299	0,513
	35	0,10	0,046	4,97	0,107	0,24	0,336	0,023	0,630
	45	0,17	0,065	7,68	0,119	0,40	0,171	0,394	0,684
	55	0,28	0,095	12,5	0,158	0,68	0,481	0,481	0,755
	65	0,53	0,163	21,2	0,210	1,21	0,428	0,606	0,814
	75	1,24	0,386	38,4	0,358	2,41	0,668	0,668	0,968
	85	4,82	1,423	85,7	0,965	7,53	2,342	0,685	1,608
	95	131	15,78	616	16,36	141	31,97	0,749	20,84
Ganho		58%		98%		46%		-82%	

Fonte: Dados da pesquisa.

remove, em média, mais de 70% das correspondências incorretas. Analisando a especificidade, o filtro FOMP removeu mais da metade das correspondências incorretas, em média 65%. Para o tempo de processamento, de 5% a 85%, o algoritmo executa consideravelmente rápido, levando menos de 1 segundo.

Tabela 4 – Tempo médio de execução, quantidade de iterações, especificidade para todos os experimentos usando apenas o filtro FOMP. A média para todas as porcentagens de outliers é representada por m e o desvio padrão é denotado por $\pm$.

		Tempo de Execução (s)	Quantidade de Iterações	Especificidade	Revocação
Outliers (%)	5	$0,016 \pm 0,01$	$3 \pm 0,96$	$0,69 \pm 0,34$	$1,00 \pm 0$
	15	$0,019 \pm 0,01$	$7 \pm 2,98$	$0,70 \pm 0,22$	$1,00 \pm 0$
	25	$0,025 \pm 0,01$	$12 \pm 5,37$	$0,70 \pm 0,18$	$0,99 \pm 0,003$
	35	$0,037 \pm 0,02$	$18 \pm 8,36$	$0,67 \pm 0,15$	$0,99 \pm 0,003$
	45	$0,057 \pm 0,03$	$26 \pm 12,5$	$0,67 \pm 0,14$	$0,99 \pm 0,004$
	55	$0,100 \pm 0,06$	$38 \pm 18,0$	$0,66 \pm 0,11$	$0,99 \pm 0,006$
	65	$0,171 \pm 0,10$	$56 \pm 26,5$	$0,64 \pm 0,11$	$0,99 \pm 0,026$
	75	$0,289 \pm 0,17$	$89 \pm 42,1$	$0,62 \pm 0,09$	$0,98 \pm 0,064$
	85	$0,882 \pm 0,54$	$162 \pm 75,3$	$0,61 \pm 0,07$	$0,97 \pm 0,081$
	95	$16,83 \pm 10,5$	510 ± 231	$0,57 \pm 0,03$	$0,90 \pm 0,182$
m		$1,84 \pm 6,01$	$92,3 \pm 166$	$0,65 \pm 0,17$	$0,98 \pm 0,07$

Fonte: Dados da pesquisa.

5.6.2 Variações na quantidade de correspondências corretas

Para analisar o efeito da alteração da quantidade de inliers no conjunto inicial de pontos correspondidos, foram realizados experimentos variando este valor, iniciando em 10, incrementado por 10 até 60 correspondências corretas. Os resultados para este experimento podem ser vistos na Figura 29. A diferença de revocação e especificidade acima de 20 correspondências corretas é pequena para todas as porcentagens de correspondências incorretas. Quando a quantidade de correspondências corretas é 10, a revocação cai consideravelmente a partir de 45% de correspondências incorretas, assumindo os valores mais baixos. Portanto, a quantidade mínima de correspondências corretas recomendada é 20.

Figura 29 – Incrementos na quantidade de correspondências corretas (inliers). Ambos os gráficos mostram valores médios para todos os experimentos, ou seja, 35 pares de imagens, totalizando 7:000 execuções para cada porcentagem de outlier.

Fonte: Dados da pesquisa.

5.6.3 Influência do parâmetro α

O parâmetro α pode ser definido de acordo com o objeto da aplicação. Com o intuito de analisar o efeito da variação do valor α nas medidas de revocação e especificidade, foram realizados experimentos em cada uma das bases de imagens usadas neste trabalho. O valor de α é modificado, iniciando em 0;1, incrementado por 0;1 até 1. Os resultados para este experimento são mostrados na Figura 30. É possível notar que existe uma relação de perda-e-ganho entre a revocação e especificidade: quanto maior o valor de α , menor o valor de especificidade e maior o valor de revocação. Quanto menor o valor de α , maior o valor de especificidade e menor o valor de revocação.

Se existir um conhecimento prévio sobre as imagens, o valor α pode ser definido de modo que a revocação seja menos afetada. Como mostrado na Figura 30, para imagens com movimentos de câmera (Figura 30a), ambiguidades (Figura 30b) e objetos deformados (Figura 30c), valores baixos de α funcionam adequadamente. Entretanto, para imagens com transformações afim (Figura 30d), um valor de α maior que 0;6 seria uma melhor escolha, uma vez que este apresenta um balanço entre revocação e especificidade.

Se o objetivo for filtrar correspondências incorretas, mas ao mesmo tempo manter o máximo possível de correspondências corretas, a melhor escolha seria um valor de a igual ou maior que 0,6. De forma inversa, se o objetivo for remover o máximo possível de correspondências incorretas, mas ao mesmo tempo manter pelo menos 50% das correspondências corretas, a melhor escolha seria um valor de a igual a 0,1.

Figura 30 – Valores de revocação e especificidade para variações no valor do parâmetro a do algoritmo FOMP.

Fonte: Dados da pesquisa.

5.7 Considerações finais

Neste capítulo foram apresentados os experimentos, resultados e discussões referentes ao filtro FOMP, que é uma técnica de pré-processamento aplicada antes dos algoritmos de consenso entre imagens, tais como GTM, WGTM, SOCBV e RANSAC. O filtro FOMP tem o intuito de remover uma parte das correspondências incorretas presentes no conjunto de pontos inicialmente correspondidos entre duas imagens, e, como consequência, obter um ganho no tempo de processamento. O filtro FOMP considera os pontos correspondidos de cada imagem como dois grafos completos e elimina iterativamente o ponto que apresenta a maior diferença de pesos. O critério de parada a pode ser definido de acordo com os objetivos da aplicação, se há um conhecimento prévio do tipo de imagem. Baixos valores de a (0; 2) removem uma porcentagem maior de correspondências incorretas, enquanto também removem uma porcentagem pequena de correspondências corretas. Valores de a grandes, ao redor de 1, eliminam uma quantidade menor de correspondência incorretas, contudo, mantém a maior parte de correspondências corretas. Os resultados experimentais mostraram que, ao aplicar o filtro FOMP, a precisão dos métodos de

consenso são similares e a revocação, em geral, é melhorada. Além do mais, o tempo de execução é consideravelmente menor, cerca de 50% mais rápido, do que sem usar o filtro FOMP. A análise individual do filtro mostrou que ele manteve, em média, mais de 98% das correspondências corretas e removeu mais de 65% das correspondências incorretas.

CONCLUSÃO

A remoção de outliers de um conjunto de pontos correspondidos entre duas imagens representa um dos passos mais importantes no processo de fundir imagens com sobreposição de regiões. Um simples ponto incorretamente correspondido acarreta em um resultado final incorreto. Os algoritmos para remoção de outliers apresentam um resultado aceitável na maior parte dos casos. Entretanto, o custo computacional é inviável para aplicações que necessitam de uma resposta rápida.

Os algoritmos de consenso baseados em grafos, tais como GTM, WGTM e SOCBV, removem apenas uma correspondência incorreta por vez. Desta forma, a quantidade de iterações cresce a medida que a quantidade de outliers aumenta. Com a finalidade de diminuir a quantidade de correspondências incorretas e validar a hipótese inicial, nesta dissertação foi proposto o método FOMP, o qual filtra uma grande parcela de outliers do conjunto de pontos inicialmente correspondidos. Isto faz com que os algoritmos de consenso necessitem de uma quantidade menor de iterações, reduzindo, assim, o tempo final de processamento. Existem dois fatores que contribuem fortemente para esse ganho em tempo de processamento:

1. Os grafos completos calculados pelo método FOMP podem ser reaproveitados nos algoritmos de consenso, uma vez que estes já os calculam internamente. Portanto, o algoritmo FOMP realiza externamente um passo comum a todos estes algoritmos baseados em grafos, tornando o processo mais eficiente como um todo. Além do mais, o cálculo de grafos completos é um dos passos mais custosos a todos esses algoritmos, que ao ser realizado sobre um conjunto de pontos reduzido, agiliza ainda mais o processo.
2. Com a filtragem de uma parcela das correspondências incorretas, os algoritmos necessitam de uma quantidade menor de iterações para chegar a um resultado final.

Para o método RANSAC, apesar de não haver ganho de tempo de processamento, vale ressaltar que, com a diminuição de correspondências incorretas no conjunto inicial de

correspondências, o método é capaz de detectar uma quantidade maior de verdadeiros positivos (correspondências corretas) e uma quantidade menor de falsos positivos (outlier identificado como inlier), o que é uma melhora relevante, considerando uma aplicação em que a precisão é um fator importante.

6.1 Principais contribuições deste trabalho

A principal contribuição deste trabalho de mestrado foi o filtro de pré-processamento proposto, o qual reduziu consideravelmente a quantidade de correspondências incorretas para os algoritmos de consenso, tais como, RANSAC, GTM, WGTM e SOCBV. Isto permitiu que estes algoritmos realizassem uma quantidade menor de iterações, implicando em um menor tempo de processamento, o que é um grande aprimoramento para o estado-da-arte nesta área. O filtro FOMP, do inglês, Filtering out Outliers from Matched Points foi publicado no SIBGRAPI 2016, na área de processamento de imagens, como pode ser visto em ([RAMOS; WATANABE; TRAINA, 2016](#)).

6.2 Proposta para trabalhos futuros

Baseando-se no filtro de pré-processamento, acredita-se que um filtro de pós-processamento também possa ser proposto. Após a aplicação do filtro FOMP e algum algoritmo de consenso, como exemplo, SOCBV, a quantidade de correspondências incorretas será pequena. Desta forma, um filtro pós-processamento poderia ser aplicado a fim de se obter maiores valores de precisão.

REFERÊNCIAS

AGUILAR, W.; FRAUEL, Y.; ESCOLANO, F.; MARTINEZ-PEREZ, M. E.; ESPINOSA-ROMERO, A.; LOZANO, M. A. A robust graph transformation matching for non-rigid registration. *Image and Vision Computing*, v. 27, n. 7, p. 897 – 910, 2009. ISSN 0262-8856. 7th IAPR-TC15 Workshop on Graph-based Representations (GbR 2007). Disponível em: <http://www.sciencedirect.com/science/article/pii/S0262885608001078>. Citado 10 vezes nas páginas 19, 30, 38, 41, 43, 45, 46, 47, 63 e 64.

AIRES, K. R. T. Segmentação de Planos Baseada em Homografia Afim, Fluxo Óptico e Reconstrução Métrica. Tese (Doutorado) — Universidade Federal do Rio Grande do Norte, Natal/RN, out. 2009. Disponível em: http://www.dominiopublico.gov.br/pesquisa/DetalheObraForm.do?select_action=&co_obra=162886. Citado 2 vezes nas páginas 37 e 38.

ALMEIDA, L. L. Construção automática de imagens de super-resolução a partir de mosaicos formados por sequências de imagens. Tese (Doutorado) — Universidade de São Paulo, São Carlos, maio 2013. Disponível em: <http://www.teses.usp.br/teses/disponiveis/18/18152/tde-03012014-100834/>. Citado na página 90.

ARDIZZONE, E.; BRUNO, A.; MAZZOLA, G. Copy-move forgery detection by matching triangles of keypoints. *IEEE Transactions on Information Forensics and Security*, v. 10, n. 10, p. 2084–2094, Oct 2015. ISSN 1556-6013. Citado na página 29.

BAY, H.; ESS, A.; TUYTELAARS, T.; GOOL, L. V. Speeded-up robust features (SURF). *Comput. Vis. Image Underst.*, Elsevier Science Inc., New York, NY, USA, v. 110, n. 3, p. 346–359, jun. 2008. ISSN 1077-3142. Disponível em: <http://dx.doi.org/10.1016/j.cviu.2007.09.014>. Citado 2 vezes nas páginas 93 e 97.

BAY, H.; TUYTELAARS, T.; GOOL, L. V. SURF: Speeded Up Robust Features. In: *ECCV*. [S.l.: s.n.], 2006. p. 404–417. Citado 6 vezes nas páginas 89, 93, 95, 96, 97 e 98.

BRITO, D. N.; NUNES, C. F. G.; PADUA, F. L. C.; LACERDA, A. Evaluation of interest point matching methods for projective reconstruction of 3d scenes. *IEEE Latin America Transactions*, v. 14, n. 3, p. 1393–1400, March 2016. ISSN 1548-0992. Citado na página 29.

BROWN, M.; LOWE, D. G. Automatic panoramic image stitching using invariant features. *Int. J. Comput. Vision*, Kluwer Academic Publishers, Hingham, MA, USA, v. 74, n. 1, p. 59–73, ago. 2007. ISSN 0920-5691. Disponível em: <http://dx.doi.org/10.1007/s11263-006-0002-3>. Citado na página 33.

CHIHAILOU, T.; JLASSI, H.; KACHOURI, R.; HAMROUNI, K.; AKIL, M. Personal verification system based on retina and surf descriptors. In: *2016 13th International Multi-Conference on Systems, Signals Devices (SSD)*. [S.l.: s.n.], 2016. p. 280–286. Citado na página 89.

CHOI, S.; HAN, S. Fast image retrieval with grid-based keypoint detector and binary descriptor. In: *2014 International Conference on Information and Communication Technology Convergence (ICTC)*. [S.l.: s.n.], 2014. p. 679–680. ISSN 2162-1233. Citado na página 29.

CHUI, H. Non-Rigid Point Matching: Algorithms, Extensions and Applications. Tese (Doutorado) — Yale University, May 2001. Citado na página 41.

CHUI, H.; RANGARAJAN, A. A new point matching algorithm for non-rigid registration. *Comput. Vis. Image Underst.*, Elsevier Science Inc., New York, NY, USA, v. 89, n. 2-3, p. 114–141, fev. 2003. ISSN 1077-3142. Disponível em: <[http://dx.doi.org/10.1016/S1077-3142\(03\)00009-2](http://dx.doi.org/10.1016/S1077-3142(03)00009-2)>. Citado na página 41.

CHUM, O.; MATAS, J. Randomized RANSAC with t,d test. In: *Image and Vision Computing*. [S.l.: s.n.], 2002. p. 448–457. Citado na página 42.

CHUM, O.; MATAS, J.; KITTLER, J. Locally optimized RANSAC. In: MICHAELIS, B.; KRELL, G. (Ed.). *DAGM-Symposium*. Springer, 2003. (Lecture Notes in Computer Science, v. 2781), p. 236–243. ISBN 3-540-40861-4. Disponível em: <<http://dblp.uni-trier.de/db/conf/dagm/dagm2003.html#ChumMK03>>. Citado na página 42.

COLLINS, T.; MESEJO, P.; BARTOLI, A. An analysis of errors in graph-based keypoint matching and proposed solutions. In: SPRINGER. *European Conference on Computer Vision (ECCV)*. Zurich, Switzerland: Springer, 2014. Citado na página 41.

CROW, F. C. Summed-area tables for texture mapping. In: *Proceedings of the 11th annual conference on Computer graphics and interactive techniques*. New York, NY, USA: ACM, 1984. (SIGGRAPH '84), p. 207–212. ISBN 0-89791-138-5. Disponível em: <<http://doi.acm.org/10.1145/800031.808600>>. Citado 2 vezes nas páginas 93 e 95.

DALAL, N.; TRIGGS, B. Histograms of oriented gradients for human detection. In: SCHMID, C.; SOATTO, S.; TOMASI, C. (Ed.). *International Conference on Computer Vision & Pattern Recognition*. INRIA Rhône-Alpes, ZIRST-655, av. de l'Europe, Montbonnot-38334: [s.n.], 2005. v. 2, p. 886–893. Disponível em: <<http://lear.inrialpes.fr/pubs/2005/DT05>>. Citado na página 89.

DEZA, M. M.; DEZA, E. *Encyclopedia of Distances*. [S.l.]: Springer Berlin Heidelberg, 2009. Citado na página 35.

DU, S.; ZHENG, N.; YING, S.; LIU, J. Affine iterative closest point algorithm for point set registration. *Pattern Recogn. Lett.*, Elsevier Science Inc., New York, NY, USA, v. 31, n. 9, p. 791–799, jul. 2010. ISSN 0167-8655. Disponível em: <<http://dx.doi.org/10.1016/j.patrec.2010.01.020>>. Citado na página 41.

DUBROFSKY, E. Homography Estimation. Dissertação (Mestrado) — University of British Columbia, March 2009. Citado na página 38.

FANG, L.; LI, S.; KANG, X.; BENEDIKTSSON, J. A. Spectral-spatial hyperspectral image classification via multiscale adaptive sparse representation. *IEEE T. Geoscience and Remote Sensing*, p. 7738–7749, 2014. Citado na página 34.

FISCHLER, M. A.; BOLLES, R. C. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, ACM, New York, NY, USA, v. 24, n. 6, p. 381–395, jun. 1981. ISSN 0001-0782. Disponível em: <<http://doi.acm.org/10.1145/358669.358692>>. Citado 5 vezes nas páginas 19, 30, 36, 43 e 63.

GALANDE, A.; PATIL, R. The art of medical image fusion: A survey. In: *Advances in Computing, Communications and Informatics (ICACCI)*, 2013 International Conference on. [S.l.: s.n.], 2013. p. 400–405. Citado 2 vezes nas páginas 29 e 34.

GODINHO, T. M.; VIANA-FERREIRA, C.; SILVA, L. A. B.; COSTA, C. A routing mechanism for cloud outsourcing of medical imaging repositories. *IEEE Journal of Biomedical and Health Informatics*, v. 20, n. 1, p. 367–375, Jan 2016. ISSN 2168-2194. Citado na página 29.

GOLD, S.; RANGARAJAN, A. A graduated assignment algorithm for graph matching. *IEEE Trans. Pattern Anal. Mach. Intell.*, IEEE Computer Society, Washington, DC, USA, v. 18, n. 4, p. 377–388, abr. 1996. ISSN 0162-8828. Disponível em: <<http://dx.doi.org/10.1109/34.491619>>. Citado na página 41.

GONGE, S. S.; GHATOL, A. A. Combination of encryption and digital watermarking techniques used for security and copyright protection of still image. In: *Recent Advances and Innovations in Engineering (ICRAIE)*, 2014. [S.l.: s.n.], 2014. p. 1–6. Citado na página 29.

GONZALEZ, R. C.; WOODS, R. E. *Digital Image Processing* (3rd Edition). Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 2006. ISBN 013168728X. Citado na página 90.

GROBEL, M. C. B.; TELLES, V. L. C. N. Da comunicação visual pré-histórica ao desenvolvimento da linguagem escrita, e a evolução da autenticidade documentoscópica. *Revista Acadêmica Oswaldo Cruz*, v. 1, p. 1–12, 2014. ISSN 2357-81873. Citado na página 29.

GUPTA, M.; KUMAR, S.; GARG, S.; KEJRIWAL, N.; BEHERA, L. A novel surf-based algorithm for tracking a 'human' in a dynamic environment. In: *Control Automation Robotics Vision (ICARCV)*, 2014 13th International Conference on. [S.l.: s.n.], 2014. p. 1004–1009. Citado na página 90.

GURUPRASAD, P.; GURUPRASAD. Handwritten devanagari word recognition using robust invariant feature transforms. In: *2015 International Conference on Applied and Theoretical Computing and Communication Technology (iCATccT)*. [S.l.: s.n.], 2015. p. 327–330. Citado na página 29.

HARTLEY, R. I.; ZISSERMAN, A. *Multiple View Geometry in Computer Vision*. [S.l.]: Cambridge University Press, ISBN: 0521623049, 2000. Citado na página 89.

HAYASHIDA, M.; KOYANO, H.; AKUTSU, T. Measuring the similarity of protein structures using image local feature descriptors SIFT and SURF. In: *Systems Biology (ISB)*, 2014 8th International Conference on. [S.l.: s.n.], 2014. p. 164–168. Citado na página 89.

IZADI, M.; SAEEDI, P. Robust weighted graph transformation matching for rigid and nonrigid image registration. *Image Processing, IEEE Transactions on*, v. 21, n. 10, p. 4369–4382, Oct 2012. ISSN 1057-7149. Citado 8 vezes nas páginas 19, 30, 41, 43, 46, 48, 63 e 65.

Jl, D.; YANG, R.; ZHANG, L.; WANG, B.; CHEN, X. The research of medical microscopic image mosaic based on the algorithm of surf. In: *Wavelet Active Media Technology and Information Processing (ICCWAMTIP)*, 2013 10th International Computer Conference on. [S.l.: s.n.], 2013. p. 16–20. Citado na página 89.

KOVESI, P. D. *MATLAB and Octave Functions for Computer Vision and Image Processing*. 2009. <<http://www.peterkovesi.com/matlabfns/>>. Online, acessado em 29 de Julho de 2016. Citado na página 63.

LANDAU, G. M.; MYERS, E. W.; SCHMIDT, J. P.; SCHMIDT, P. Incremental string comparison. *SIAM Journal on Computing*, v. 27, p. 557–582, 1995. Citado na página 53.

LEBEDA, K.; MATAS, J.; CHUM, O. Fixing the locally optimized RANSAC. In: Proceedings of the British Machine Vision Conference. [S.l.]: BMVA Press, 2012. p. 95.1–95.11. ISBN 1-901725-46-4. Citado na página 42.

LEUTENEGGER, S.; CHLI, M.; SIEGWART, R. Brisk: Binary robust invariant scalable keypoints. In: Computer Vision (ICCV), 2011 IEEE International Conference on. [S.l.: s.n.], 2011. p. 2548–2555. ISSN 1550-5499. Citado na página 89.

LINDEBERG, T. Feature detection with automatic scale selection. Int. J. Comput. Vision, Kluwer Academic Publishers, Hingham, MA, USA, v. 30, n. 2, p. 79–116, nov. 1998. ISSN 0920-5691. Disponível em: <<http://dx.doi.org/10.1023/A:1008045108935>>. Citado na página 89.

LIQIAN, D.; YUEHUI, J. Moon landform images fusion and mosaic based on SIFT method. In: Computer and Information Application (ICCIA), 2010 International Conference on. [S.l.: s.n.], 2010. p. 29–32. Citado na página 29.

LIU, F.; BI, F.; CHEN, L.; WEI, H.; TIE, W. An optimization algorithm based on bi-sift for sar image registration. In: IET International Radar Conference 2015. [S.l.: s.n.], 2015. p. 1–5. Citado na página 90.

LIU, Y.; LAN, C.; YAO, F.; LI, L.; LI, C. Oblique remote sensing image matching based on improved akaze algorithm. In: 2016 Sixth International Conference on Information Science and Technology (ICIST). [S.l.: s.n.], 2016. p. 448–454. Citado na página 29.

LOWE, D. G. Object recognition from local scale-invariant features. In: Proceedings of the International Conference on Computer Vision-Volume 2 - Volume 2. Washington, DC, USA: IEEE Computer Society, 1999. (ICCV '99), p. 1150–. ISBN 0-7695-0164-8. Disponível em: <<http://dl.acm.org/citation.cfm?id=850924.851523>>. Citado 8 vezes nas páginas 34, 35, 89, 90, 91, 92, 93 e 94.

MAREE, R.; GEURTS, P.; PIATER, J.; WEHENKEL, L. Random subwindows for robust image classification. In: Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on. [S.l.: s.n.], 2005. v. 1, p. 34–40 vol. 1. ISSN 1063-6919. Citado na página 89.

MARSZALEK, M.; SCHMID, C. Spatial weighting for bag-of-features. In: CVPR (2). IEEE Computer Society, 2006. p. 2118–2125. ISBN 0-7695-2597-0. Disponível em: <<http://dblp.uni-trier.de/db/conf/cvpr/cvpr2006-2.html#MarszalekS06>>. Citado na página 89.

MATAS, J.; CHUM, O. Randomized RANSAC with sequential probability ratio test. In: ICCV. IEEE Computer Society, 2005. p. 1727–1732. ISBN 0-7695-2334-X. Disponível em: <<http://dblp.uni-trier.de/db/conf/iccv/iccv2005-2.html#MatasC05>>. Citado na página 42.

MATAS, J.; CHUM, O.; URBAN, M.; PAJDLA, T. Robust wide baseline stereo from maximally stable extremal regions. In: Proceedings of the British Machine Vision Conference. [S.l.]: BMVA Press, 2002. p. 36.1–36.10. ISBN 1-901725-19-7. Doi:10.5244/C.16.36. Citado na página 89.

MENG, F.; LI, X.; PEI, J. A feature point matching based on spatial order constraints bilateral-neighbor vote. IEEE Transactions on Image Processing, v. 24, n. 11, p. 4160–4171, Nov 2015. ISSN 1057-7149. Citado 10 vezes nas páginas 19, 30, 41, 43, 49, 50, 51, 53, 54 e 63.

MIKOLAJCZYK, K.; SCHMID, C. Scale and affine invariant interest point detectors. *International Journal of Computer Vision*, v. 60, n. 1, p. 63–86, 2004. Disponível em: <<http://lear.inrialpes.fr/pubs/2004/MS04>>. Citado na página 89.

MIKOLAJCZYK, K.; TUYTELAARS, T.; SCHMID, C.; ZISSERMAN, A.; MATAS, J.; SCHAFFALITZKY, F.; KADIR, T.; GOOL, L. V. A comparison of affine region detectors. *Int. J. Comput. Vision*, Kluwer Academic Publishers, Hingham, MA, USA, v. 65, n. 1-2, p. 43–72, nov. 2005. ISSN 0920-5691. Disponível em: <<http://dx.doi.org/10.1007/s11263-005-3848-x>>. Citado 3 vezes nas páginas 64, 72 e 89.

MORANO, P.; TAJANI, F. Least median of squares regression and minimum volume ellipsoid estimator for outliers detection in housing appraisal. *Int. J. Bus. Intell. Data Min.*, Inderscience Publishers, Inderscience Publishers, Geneva, SWITZERLAND, v. 9, n. 2, p. 91–111, out. 2014. ISSN 1743-8195. Disponível em: <<http://dx.doi.org/10.1504/IJBIDM.2014.065074>>. Citado na página 36.

MOREL, J.-M.; YU, G. Asift: A new framework for fully affine invariant image comparison. *SIAM Journal on Imaging Sciences*, v. 2, n. 2, p. 438–469, 2009. Disponível em: <<http://dx.doi.org/10.1137/080732730>>. Citado na página 89.

MURUGAN, M.; MATHEWS, M. 2d and 3d active shape model with SURF algorithm for object retrieval: Content based image retrieval. In: *Advanced Computing and Communication Systems (ICACCS)*, 2013 International Conference on. [S.l.: s.n.], 2013. p. 1–7. Citado na página 89.

MYRONENKO, A.; SONG, X.; CARREIRA-PERPIÑÁN, . Non-rigid point set registration: Coherent point drift (cpd). In: *IN ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS 19*. [S.l.]: MIT Press, 2006. Citado na página 41.

NAZNEEN, N.; SHAFIQ, M.; HAMEED, A. Template matching of aerial images using gpu. In: *2016 13th International Bhurban Conference on Applied Sciences and Technology (IB-CAST)*. [S.l.: s.n.], 2016. p. 206–212. Citado na página 29.

PANDEY, R. C.; SINGH, S. K.; SHUKLA, K.; AGRAWAL, R. Fast and robust passive copy-move forgery detection using SURF and SIFT image features. In: *Industrial and Information Systems (ICIIS)*, 2014 9th International Conference on. [S.l.: s.n.], 2014. p. 1–6. Citado na página 89.

PARIKH, S.; KALVA, H.; ADZIC, V. Evaluation of hevc compression for high bit depth medical images. In: *2016 IEEE International Conference on Consumer Electronics (ICCE)*. [S.l.: s.n.], 2016. p. 311–314. Citado na página 29.

PATEL, M. S.; PATEL, N. M.; HOLIA, M. S. Feature based multi-view image registration using surf. In: *Advanced Computing and Communication (ISACC)*, 2015 International Symposium on. [S.l.: s.n.], 2015. p. 213–218. Citado na página 29.

PHAN, J. H.; HOFFMAN, R.; KOTHARI, S.; WU, P. Y.; WANG, M. D. Integration of multi-modal biomedical data to predict cancer grade and patient survival. In: *2016 IEEE-EMBS International Conference on Biomedical and Health Informatics (BHI)*. [S.l.: s.n.], 2016. p. 577–580. Citado na página 29.

RAMOS, J. S.; WATANABE, C. Y. V.; TRAINA, A. J. M. FOMP: A novel preprocessing technique to speed-up the outlier removal from matched points. In: CAPPABIANCO, F. A. M.; FARIA, F. A.; ALMEIDA, J.; KÖRTING, T. S. (Ed.). *Electronic Proceedings of the 29th Conference on Graphics, Patterns and Images (SIBGRAPI '16)*. São José dos Campos, SP, Brazil: [s.n.], 2016. Disponível em: <<http://gibis.unifesp.br/sibgrapi16>>. Citado na página 80.

RAVAL, K.; ZAFAR, S. Digital watermarking with copyright authentication for image communication. In: *Intelligent Systems and Signal Processing (ISSP), 2013 International Conference on*. [S.l.: s.n.], 2013. p. 111–116. Citado na página 29.

ROSTEN, E.; DRUMMOND, T. Machine learning for high-speed corner detection. In: *European Conference on Computer Vision*. [S.l.: s.n.], 2006. p. 430–443. Citado na página 89.

RUBLEE, E.; RABAUD, V.; KONOLIGE, K.; BRADSKI, G. ORB: An efficient alternative to SIFT or SURF. In: *Proceedings of the 2011 International Conference on Computer Vision*. Washington, DC, USA: IEEE Computer Society, 2011. (ICCV '11), p. 2564–2571. ISBN 978-1-4577-1101-5. Disponível em: <<http://dx.doi.org/10.1109/ICCV.2011.6126544>>. Citado na página 89.

SANROMÀ, G.; ALQUÉZAR, R.; SERRATOSA, F. A new graph matching method for point-set correspondence using the EM algorithm and softassign. *Computer Vision and Image Understanding*, v. 116, n. 2, p. 292 – 304, 2012. ISSN 1077-3142. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S1077314211002360>>. Citado na página 39.

SATNIK, A.; HUDEC, R.; KAMENCAY, P.; HLUBIK, J.; BENCO, M. A comparison of key-point descriptors for the stereo matching algorithm. In: *2016 26th International Conference Radioelektronika (RADIOELEKTRONIKA)*. [S.l.: s.n.], 2016. p. 292–295. Citado na página 29.

SATTTLER, T.; LEIBE, B.; KOBELT, L. Scramsac: Improving RANSAC's efficiency with a spatial consistency filter. In: *ICCV*. IEEE, 2009. p. 2090–2097. Disponível em: <<http://dblp.uni-trier.de/db/conf/iccv/iccv2009.html#SattlerLK09>>. Citado na página 42.

SEDAGHAT, A.; EBADI, H. Remote sensing image matching based on adaptive binning sift descriptor. *IEEE Transactions on Geoscience and Remote Sensing*, v. 53, n. 10, p. 5283–5293, Oct 2015. ISSN 0196-2892. Citado na página 29.

TAKÁCS, M.; BENCZE, T.; SAZABÓ-RESCH, M. Z.; VÁMOSSY, Z. Object recognition to support indoor robot navigation. In: *Computational Intelligence and Informatics (CINTI), 2015 16th IEEE International Symposium on*. [S.l.: s.n.], 2015. p. 239–242. Citado na página 29.

TONG, L.; ZHAN, G.; PENG, Q.; LI, Y.; LI, Y. Warped document image mosaicing method based on inflection point detection and registration. In: *Multimedia Information Networking and Security (MINES), 2012 Fourth International Conference on*. [S.l.: s.n.], 2012. p. 306–310. Citado 2 vezes nas páginas 29 e 34.

VIOLA, P.; JONES, M. Robust real-time object detection. In: *International Journal of Computer Vision*. [S.l.: s.n.], 2001. Citado na página 94.

WEDGE, D. The RANSAC Song. 2012. <<http://danielwedge.com/ransac/>>. Online, acessado em 1 de Agosto de 2016. Citado na página 42.

XIAO, H.; YUAN, K.; HE, W. An improvement on the keypoint detection method of sift for hardware computation. In: Intelligent Control and Automation (WCICA), 2011 9th World Congress on. [S.l.: s.n.], 2011. p. 63–68. Citado na página [90](#).

XIAO, X.; WAKUNAMI, K.; CHEN, X.; SHEN, X.; JAVIDI, B.; KIM, J.; NAM, J. Three-dimensional holographic display using dense ray sampling and integral imaging capture. Journal of Display Technology, v. 10, n. 8, p. 688–694, Aug 2014. ISSN 1551-319X. Citado na página [29](#).

ZHENG, W.; ZHOU, F.; WANG, Z. Robust and accurate monocular visual navigation combining imu for a quadrotor. Automatica Sinica, IEEE/CAA Journal of, v. 2, n. 1, p. 33–44, January 2015. ISSN 2329-9266. Citado na página [89](#).

ZHU, L.; WANG, Y.; ZHAO, B.; ZHANG, X. A fast image stitching algorithm based on improved surf. In: Computational Intelligence and Security (CIS), 2014 Tenth International Conference on. [S.l.: s.n.], 2014. p. 171–175. Citado na página [90](#).

ZITOVÁ, B.; FLUSSER, J. Image registration methods: a survey. Image and Vision Computing, v. 21, p. 977–1000, 2003. Citado 2 vezes nas páginas [34](#) e [89](#).

DETECÇÃO DE PONTOS DE INTERESSE

Encontrar pontos correspondentes confiáveis entre duas imagens da mesma cena, tiradas de pontos de vista diferentes com dispositivos diferentes e com diferenças de iluminação, é uma tarefa difícil e crucial para várias aplicações, tais como reconstrução de cenas em 3D, reconhecimento de objetos, registro de imagens, entre outras ([MATAS et al., 2002](#); [HARTLEY; ZISSERMAN, 2000](#)).

Para encontrar pontos correspondentes entre duas imagens é necessário que se tenha pontos de interesse em cada uma das imagens. Assim, estes pontos podem ser correspondidos com o uso de alguma métrica de similaridade entre eles.

Os pontos de interesse especificam pontos singulares e de destaque nas imagens, como exemplo, cantos, regiões (blobs) e esquinas. Após a detecção destes pontos, estes podem ser representados por descritores locais (vetores de características), que definem a representação das características ao seu redor, que podem ser valores de intensidades ou histograma de gradientes. Existem vários detectores de pontos de interesse na literatura ([LOWE, 1999](#); [MIKOLAJCZYK; SCHMID, 2004](#); [LINDEBERG, 1998](#); [MIKOLAJCZYK et al., 2005](#)). Porém, os mais difundidos são: Laplacian of Gaussian (LoG) ([LINDEBERG, 1998](#)), Harris Laplace ([MIKOLAJCZYK; SCHMID, 2004](#)), Hessian Laplace ([MIKOLAJCZYK et al., 2005](#)), Harris Affine ([MIKOLAJCZYK; SCHMID, 2004](#)), Hessian Affine ([MIKOLAJCZYK et al., 2005](#)) e a Difference of Gaussian (DoG) ([LOWE, 1999](#)). Alguns trabalhos utilizam a combinação destes detectores ([MARSZALEK; SCHMID, 2006](#)). Entretanto, alguns trabalhos selecionam pontos aleatórios na imagem ([MAREE et al., 2005](#)). Vários outros métodos/variações para detecção de pontos de interesse tem sido propostos, como Binary Robust Invariant Scalable Keypoints (BRISK) ([LEUTENEGGER; CHLI; SIEGWART, 2011](#)), Features from Accelerated Segment Point (FAST) ([ROSTEN; DRUMMOND, 2006](#)), Oriented FAST and Rotated BRIEF (ORB) ([RUBLEE et al., 2011](#)), Histogram of Oriented Gradients (HOG) ([DALAL; TRIGGS, 2005](#)), Scale-Invariant Feature Transform (SIFT) ([LOWE, 1999](#)), Affine-SIFT (ASIFT) ([MO-](#)

REL; YU, 2009), Speeded Up Robust Features (SURF) (BAY; TUYTELAARS; GOOL, 2006).

É complexo desenvolver um detector que se adapte a todos os tipos de imagens devido a várias degradações e diversidade das mesmas (ZITOVÁ; FLUSSER, 2003). Contudo, os métodos SIFT e SURF tem apresentado bons resultados em diversas aplicações (JI et al., 2013; PANDEY et al., 2014; HAYASHIDA; KOYANO; AKUTSU, 2014; MURUGAN; MATHEWS, 2013; ZHENG; ZHOU; WANG, 2015). Algumas adaptações destes métodos têm sido propostas para adequação em aplicações específicas, como exemplo, Chihaoui et al. (2016) propõe o uso do método SURF para reconhecimento de humano por meio de imagens de retina. Gupta et al. (2014) propõem o uso de características de cor juntamente com o método SURF para detecção de pessoas em ambientes dinâmicos. Zhu et al. (2014) apresenta uma versão melhorada do método SURF para casamento de imagens, que utiliza aprendizado de máquina para selecionar os pontos de interesse mais confiáveis. Xiao, Yuan e He (2011) adapta o método SIFT para computação paralela. Liu et al. (2015) modifica o método SIFT para imagens de radar de abertura sintética (do inglês, Synthetic Aperture Radar – SAR).

Neste apêndice serão detalhadas as versões clássicas dos métodos SIFT e SURF.

A.1 Scale-Invariant Feature Transform – SIFT

O método SIFT (LOWE, 1999) pode ser dividido em duas partes: detecção de pontos de controle/interesse Control Points (CP) e descrição de características. A detecção de CPs em uma imagem é feita usando a diferença de gaussianas - DoG, em que os pontos de máximo/mínimo são considerados como pontos de interesse. A descrição de características em cada CP é feita usando histogramas de gradientes orientados nos vizinhos ao redor do CP (ALMEIDA, 2013).

O método SIFT extrai características locais das imagens que são razoavelmente invariantes a mudanças de iluminação, ruídos, rotação, escala e pequenas mudanças de pontos de vista.

A.1.1 Detecção de pontos de interesse

Os pontos de interesse detectados pelo SIFT correspondem aos pontos de máximos/mínimos encontrados pelo filtro de Diferença de Gaussianas em diferentes escalas.

O espaço de escala de uma imagem é definido como uma função $L(x; y; s)$, que é produzido por convolução de filtro Gaussiano $G(x; y; s)$ em uma imagem de entrada $I(x; y)$ (LOWE, 1999):

$$L(x; y; s) = G(x; y; s) * I(x; y); \quad (\text{A.1})$$

em que $*$ é a operação de convolução em x e y , e

$$G(x; y; s) = \frac{1}{2\pi s^2} e^{-\frac{(x^2 + y^2)}{2s^2}} \quad (\text{A.2})$$

corresponde a um filtro Gaussiano¹ com um determinado valor s . .

Após a convolução da imagem com diferentes valores de s , é possível detectar pontos de máximos/mínimos usando a diferença de Gaussianas, $D(x; y; s)$, que pode ser calculada como a diferença entre duas imagens borradas adjacentes por um valor constante multiplicativo k :

$$\begin{aligned} D(x; y; ks) &= (G(x; y; ks) - G(x; y; s)) * I(x; y) \\ &= L(x; y; ks) - L(x; y; s), \end{aligned} \quad (\text{A.3})$$

que é apenas a diferença de Gaussianas nas imagens nas escalas s e ks , como pode ser visto na Figura 31.

Figura 31 – Imagens com aplicação do filtro Gaussiano em diferentes escalas e o cálculo da Diferença de Gaussianas - DoG entre pares de imagens adjacentes.

Fonte: Lowe (1999).

As imagens que foram aplicadas o filtro Gaussiano são agrupadas por oitavas (uma oitava corresponde ao dobro do valor de s), o valor de k é selecionado de forma que possa ser obtido um número fixo de imagens por oitava. Isto também garante que se tenha um número fixo de DoGs por oitava. A recomendação de Lowe (1999) é que a quantidade de borramentos e oitavas seja, respectivamente, 4 e 5, como pode ser visto na Figura 32.

¹ O filtro Gaussiano é conhecido por “borrar” a imagem (GONZALEZ; WOODS, 2006).

Figura 32 – Na esquerda, exemplo de aplicação de 5 filtros gaussianos em 4 escalas (oitavas). Na direita, resultado da DoG para as imagens borradas.

Fonte: Elaborada pelo autor.

Os pontos de interesse são identificados nas regiões de máximos ou mínimos locais das imagens DoG em todas as escalas. Cada um desses pontos encontrados é comparado com seus 8-vizinhos na mesma escala e seus 9-vizinhos correspondentes nas escalas adjacentes (Figura 33). Se este ponto for um máximo ou mínimo local, ele é então selecionado como um ponto de interesse candidato.

Figura 33 – Detecção de máximo/mínimo local, o pixel marcado com $\times$ é comparado com seus 26-vizinhos numa vizinhança de $3 \times 3 \times 3$ que fazem parte das imagens DoG adjacentes.

Fonte: Adaptada de Lowe (1999).

Para cada ponto de interesse candidato encontrado são desconsiderados pontos que possuem um baixo contraste (sensíveis a ruídos) ou que estão ao longo de uma aresta. Para cada ponto também é dada uma orientação, de forma que este ponto se torne invariante a rotação. Para cada imagem $L(x; y)$, é calculada a magnitude do gradiente, $m(x; y)$, e a orientação, $q(x; y)$, usando a diferença de pixels:

$$m(x; y) = \frac{q}{(L(x+1; y) - L(x-1; y))^2 + (L(x; y+1) - L(x; y-1))^2} \quad (\text{A.4})$$

$$q(x; y) = \tan^{-1} \frac{\frac{L(x; y+1) - L(x; y-1)}{L(x+1; y) - L(x-1; y)}}{\frac{L(x; y+1) - L(x; y-1)}{L(x+1; y) - L(x-1; y)}} \quad (\text{A.5})$$

Para cada ponto candidato um histograma de orientações é formado das orientações de gradientes. Este histograma possui 36 intervalos, que correspondem às 360 orientações possíveis.

Os picos no histograma correspondem a orientações dominantes para aquele ponto. O pico mais alto é detectado, e assim qualquer pico local que estiver 20% abaixo do maior pico é usado para compor a orientação daquele ponto, como pode ser visto na Figura 34. Portanto, para pontos que possuem mais de uma orientação com a mesma magnitude, haverá múltiplos CPs na mesma localização.

Figura 34 – Detecção de orientação dominante no método SIFT. A orientação e magnitude do gradiente são usados para compor a orientação do CP, um histograma de orientações é criado, o pico mais alto é detectado e qualquer outro pico que estiver 20% abaixo do maior pico é usado para compor a orientação. Se houver mais de uma orientação (pico) com a mesma magnitude, múltiplos CPs são criados na mesma localização, porém com orientações diferentes.

Fonte: Adaptada de [Lowe \(1999\)](#).

A.1.2 Extração de características

Após a identificação dos CPs por meio de sua escala e orientação, tem-se valores de coordenadas 2D para representação destes pontos. Desta forma, pode-se descrever a região de cada CP extraindo-se características.

No SIFT, a característica extraída de cada ponto de interesse é a orientação do gradiente, como mostra a Figura 35. Os elementos ao redor do ponto de interesse são ponderados por uma função Gaussiana, com σ igual a metade do tamanho da janela do descritor, como pode ser visto no círculo na esquerda da Figura 35. Esta ponderação é feita para evitar mudanças abruptas no descritor e para dar menor ênfase em gradientes que estão distantes do centro do descritor, uma vez que estes são os mais afetados por erros ([LOWE, 1999](#)).

O descritor final do ponto de interesse é mostrado na direita da Figura 35. Pode ser visto que há oito direções para cada janela 4×4 . O SIFT usa uma região 16×16 para compor um descritor 4×4 , sendo que para cada janela sumarizada tem-se oito orientações de gradiente. Portanto, tem-se assim $4 \times 4 \times 8 = 128$ características.

Figura 35 – O descritor é criado para cada CP calculando-se a magnitude do gradiente e a orientação ao seu redor. Esses pontos ao redor são ponderados por uma janela Gaussiana (como mostra o círculo à esquerda). Então é feita uma divisão em janelas de 4×4 ao redor do ponto de interesse (centro). Desta forma, é feita uma sumarização acumulando-se o conteúdo de cada janela, como é mostrado à direita. Neste exemplo é usado um descritor 2×2 calculado a partir de uma janela 8×8 . Porém, o método SIFT é implementado, na prática, em uma janela 16×16 que produz um descritor 4×4 .

Fonte: Adaptada de Lowe (1999).

A.2 Speeded Up Robust Features – SURF

O SURF (BAY; TUYTELAARS; GOOL, 2006; BAY et al., 2008) é um método que se baseia no SIFT, porém possui um custo computacional menor, pois aproxima o LoG com filtros Caixa (Box Filter), que possibilita o uso de imagens integrais, também conhecida como Summed Area Table (SAT) (CROW, 1984). A SAT pode ser calculada de forma rápida, pois necessita apenas da soma de valores da matriz.

A.2.1 Detecção de pontos de interesse

No SIFT o LoG é aproximado usando DoG para encontrar pontos de interesse no espaço-escala. O SURF vai um pouco mais a fundo aproximando o LoG com filtros Caixa (Box Filter). Uma das vantagens desta abordagem é que os filtros Caixa podem ser calculados usando a imagem integral. SAT foi desenvolvida em 1984, porém apenas aplicada em visão computacional para detecção de faces em 2001 por Viola Jones (VIOLA; JONES, 2001).

A construção do SAT de uma imagem em um dado pixel $(x; y)$ é calculado como:

$$s(x; y) = i(x; y) + s(x - 1; y) + s(x; y - 1) - s(x - 1; y - 1); \quad (\text{A.6})$$

em que i representa a imagem original e s representa a matriz SAT, que nada mais é do que a soma dos valores de linha e coluna acima do ponto $(x; y)$ subtraído por este mesmo valor no ponto antecedente $s(x - 1; y - 1)$. Na Figura 36 é exemplificada a Equação A.6.

Após o cálculo da matriz SAT é possível calcular a imagem integral em qualquer retângulo desta com complexidade $O(1)$ (CROW, 1984). Dados quatro pontos que formam o retângulo ABCD, o cálculo da soma daquela região, denotado por $s(x'; y')$, pode ser calculado

Figura 36 – Passos na construção da matriz SAT. Índices fora da fronteira são considerados como zero.

Fonte: Elaborada pelo autor.

usando a [Equação A.7](#), como ilustrado na [Figura 37](#).

$$s(x';y') = s(A) + s(D) - s(B) - s(C): \quad (\text{A.7})$$

Figura 37 – Aplicação da [Equação A.7](#) para o cálculo da imagem integral em um dado retângulo ABCD.

Desta forma, a matriz SAT pode ser usada para o cálculo, aproximado, das derivadas parciais de segunda ordem, aos quais o SURF usa para detectar pontos de interesse. Na [Figura 38](#) é mostrada uma comparação dos filtros de tamanho 9×9 usados pelo SIFT e pelo SURF para o cálculo das derivadas parciais de segunda ordem nas direções L_{xx} , L_{yy} e L_{xy} .

Desta forma, é possível calcular a matriz Hessiana:

$$H(x;s) = \begin{pmatrix} L_{xx}(x;s) & L_{xy}(x;s) \\ L_{xy}(x;s) & L_{yy}(x;s) \end{pmatrix}; \quad (\text{A.8})$$

que contém as derivadas parciais de segunda ordem (aproximadas) nas direções L_{xx} , L_{yy} e L_{xy} , que são calculadas usando os filtros aproximados mostrados na [Figura 38](#).

Portanto, no SURF os CPs são detectados usando o determinante da matriz Hessiana:

$$\det(H_{\text{aprox}}) = D_{xx}D_{yy} - (0.9D_{xy})^2; \quad (\text{A.9})$$

se o determinante for maior que 0 existe um extremo, i.e., um ponto de interesse.

Figura 38 – Filtros aproximados para o cálculo das derivadas parciais de segunda ordem, sendo é possível usar a SAT para o cálculo da convolução do filtro com a imagem.

Fonte: Adaptada de Bay, Tuytelaars e Gool (2006).

Para a detecção do ponto de interesse no espaço-escala, não é realizada a minimização/-maximização da imagem, apenas o tamanho do filtro Caixa é decrementado/incrementado, como pode ser visto na Figura 39a.

Figura 39 – Tamanhos dos filtros e espaço-escala do algoritmo SURF.

(a) Incremento do tamanho do box filter.

(b) Espaço-escala usado pelo SURF. São usadas 3 oitavas com a opção para uma quarta e 4 filtros de diferentes tamanhos por oitava.

Fonte: Adaptada de Bay, Tuytelaars e Gool (2006).

Os autores do SURF (BAY; TUYTELAARS; GOOL, 2006) recomendam o uso de 3 oitavas com a opção para uma quarta oitava. São aplicados 4 tamanhos de filtros por oitava, como mostra a Figura 39(b).

O menor filtro, de tamanho 9×9 (Figura 39(b)), corresponde a um filtro gaussiano com $s = 1:2$. Com o aumento do tamanho do filtro, o valor associado ao filtro gaussiano também aumenta. O tamanho de um filtro para o outro deve conter no mínimo 6 de diferença. Desta forma é possível manter a estrutura do filtro (BAY; TUYTELAARS; GOOL, 2006).

Os pontos candidatos são selecionados de forma similar ao método SIFT. Os valores de máximo/mínimo para cada ponto são calculados pela comparação com seus 8-vizinhos na

escala original, e seus 18 pixels nas escalas acima e abaixo, totalizando 26 vizinhos.

Contudo, no SURF, um valor de limiar é definido, pontos que possuem valores menores que o limiar são descartados e pontos maiores são considerados. Desta forma, quanto maior o valor do limiar, menos pontos de interesse são detectados, porém, esses pontos possuem maior destaque visual. Quanto menor o valor do limiar, maior a quantidade de pontos detectados. O valor do limiar é definido pelo usuário.

A.2.2 Extração de características

No SURF é utilizado um vetor de 64 características, composto de informações de gradiente, similar ao SIFT, e algumas variantes, tal como o primeiro momento de Haar (BAY et al., 2008). O momento de Haar pode ser calculado em conjunto com as imagens integrais ou SAT, assim o tempo para computar o vetor de característica é reduzido.

A orientação de cada ponto de interesse é calculada primeiramente em uma janela circular. Em seguida, uma janela quadrada é construída de acordo com a orientação calculada. Desta forma, o extrator de características do SURF pode ser aplicado sobre esta janela quadrada.

O cálculo da orientação é feito usando as respostas de wavelet de Haar de tamanho 4s nas direções x e y, como mostra a Figura 40a, em um raio 6s nos vizinhos ao redor do ponto de interesse, sendo s equivalente a escala na qual o ponto de interesse foi detectado (BAY; TUYTELAARS; GOOL, 2006).

Figura 40 – Cálculo da direção dominante de um ponto de interesse no algoritmo SURF.

Fonte: Adaptada de Bay, Tuytelaars e Gool (2006).

Uma vez que a resposta wavelet é calculada, esta é ponderada por uma função Gaussiana (2:5s), centrada no ponto de interesse. Em seguida, essas respostas são representadas como vetores no plano cartesiano. A orientação dominante é estimada pelo cálculo da soma de todas as respostas dentro de uma janela de ângulo $\frac{\pi}{3}$, que é movida incrementalmente até cobrir todos os ângulos. Desta forma, a soma das duas respostas gera um novo vetor. O vetor somado dentro

do intervalo da janela a que apresentar maior valor é escolhido como orientação dominante. Este processo está ilustrado na Figura 40b.

Em algumas aplicações não é necessária a invariância à rotação. Sendo assim, o passo de encontrar a orientação do ponto pode ser omitido, deixando o processo ainda mais rápido. O SURF com pontos de interesse sem orientação é conhecido como Up-right SURF (USURF) e mantém a robustez para imagens com rotação de até mais ou menos 15 graus (BAY; TUYTELAARS; GOOL, 2006).

Uma vez que a orientação de cada ponto de interesse é encontrada, as características podem ser extraídas deste ponto. Para realizar a extração, primeiramente é construída uma janela quadrada de tamanho $20s$ (sendo s referente a escala na qual o ponto foi detectado) de acordo com a orientação do ponto, como mostra a Figura 41a.

Figura 41 – Exemplos de CPs detectados pelo SURF e extração de características.

(a) Cada janela é do tamanho de 20 vezes a escala na qual o ponto de interesse foi detectado, sua orientação é a direção dominante (em verde).

(b) O quadrado em verde limita uma das 16 sub-regiões e os círculos em azul representam os pontos de amostras dos quais são calculadas as respostas wavelets nas direções x e y . Como ilustrado, as respostas nas direções x e y são calculadas de acordo com a orientação dominante do ponto.

Fonte: Adaptada de Bay, Tuytelaars e Gool (2006).

O descritor para cada CP é dividido em sub-regiões regulares de tamanho 4×4 , totalizando 16 regiões. Para cada uma das sub-regiões são calculadas as respostas Haar wavelets de tamanho $2s$ para 25 amostras de pontos regularmente distribuídos. Representado as respostas wavelets nas direções x e y , respectivamente, como dx e dy , para cada sub-região tem o resultado da Equação A.10. Na Figura 41b pode ser visto um exemplo deste processo.

$$v_{\text{sub-região}} = \begin{matrix} h & i \\ \hat{A}_1 dx; \hat{A}_1 dy; \hat{A}_1 |dx|; \hat{A}_1 |dy| \end{matrix}; \quad (\text{A.10})$$

Cada sub-região do ponto de interesse contribui com 4 valores no vetor de características, que leva a um vetor com $4 \times 16 = 64$ características. O vetor de características gerado pelo SURF

é invariante a rotação, escala, brilho e contraste.

A.3 Considerações finais

Os pontos de interesse podem ser entendidos como pontos de destaque visual nas imagens, i.e., pontos de esquina, junções-T e regiões (blobs). Os principais métodos de detecção de pontos de interesse usam técnicas de processamento de imagens, tal como filtros gaussianos ou combinações de outros filtros. Cada método possui vantagens sob determinadas circunstâncias, não havendo um método que seja adequado para todos os tipos de cenas. Entretanto, o SIFT e o SURF apresentam melhores resultados na maior parte dos casos. O SURF é mais rápido que o SIFT e apresenta resultados similares ou até melhores em determinados casos. Uma vez que os pontos são detectados e seus respectivos vetores de características são extraídos e correspondidos entre as imagens pode ocorrer de haver alguns pontos incorretamente correspondidos, necessitando assim de um algoritmo auxiliar para detectar e remover estes pontos.

CÓDIGOS-FONTE DO ALGORITMO FOMP

Neste apêndice são mostrados os códigos-fonte do algoritmo FOMP utilizados em todos os experimentos mostrados ao longo da dissertação.

Código em Matlab

```

1 % Filtering Out Ouliers from Matched Points – FOMP Algorithm:
2 %     The algorithm takes two sets of corresponding points
3 %     (initial matches) and filter out some matches
4 %     that presents a higher sum of edges difference.
5 % Input – P1: an array with size N X 2 containing the first
6 %         set of points.
7 %         – P2: N X 2 vactor containing the corresponding points of
8 %         the first set of points (the m-th point of P1 and P2
9 %         are initially matched).
10 %         – alpha: threshold value for the stop condition
11 % Output – inliers: the inlier matches
12 %         – outliers: the outlier matches
13 %
14 % Copyright (c) 2015–2016 by Jonathan Ramos (jonathan@usp.br)
15 % Institute of Mathematics and Computer Science
16 % ICMC – USP, SP, Brazil.
17 function [inliers] = FOMP(P1, P2, alpha)
18     n = length(P1);
19     points(1:tam) = [1:tam];
20     [net1, net2] = createCompleteGraphs(P1, P2);
21     for i=1:n
22         r1 = net1./( mean( net1(:) ) );
23         r2 = net2./( mean( net2(:) ) );
24         d = sum( abs(r1-r2) )./ length(r1);
25         [m pos] = max(d);

```

```

26     if (m > alpha)
27         net1(pos, :) = [];
28         net1(:, pos) = [];
29         net2(:, pos) = [];
30         net2(pos, :) = [];
31         points(pos) = [];
32     else
33         break;
34     end
35 end
36 inliers = calculateRem(points, tam);
37 end

```

```

1  % Create two complete graphs given a set of matched points (P, P')
2  function [graph1, graph2] = createCompleteGraphs(P1, P2)
3      s = size(P1);
4      n = s(1);
5      graph1(n, n) = 0;
6      graph2(n, n) = 0;
7      for i=1:n
8          % Since both graphs are simetrical, we only need to calculate
9          % upper diagonal matrix.
10         for j=i+1:n
11             % The weights of the graph are the Euclidean Distance among the
12             % points in cartesian coordinates
13             graph1(i, j) = norm(P1(i, :) - P1(j, :));
14             graph2(i, j) = norm(P2(i, :) - P2(j, :));
15             % Both graphs are simetrical
16             graph1(j, i) = graph1(i, j);
17             graph2(j, i) = graph2(i, j);
18         end
19     end
20 end

```

```

1  % Returns a boolean vector in wich each index indicates if the
2  % vertice i was filtered (outlier) or not:
3  % 1 = removed
4  % 0 = not removed
5  function removed = calculateRem(inliersPos, n)
6      removed(n) = 0;
7      for i=1:length(inliersPos)
8          removed(inliersPos(i)) = 1;
9      end
10     removed = removed < 1;
11 end

```