

# Trabalho 4

## MO443 - Introdução ao Processamento de Imagem Digital

Thales Eduardo Nazatto  
074388  
tenazatto@gmail.com

### I. INTRODUÇÃO

Registro de imagens, conforme mencionado em [1], é o processo de correspondência ou alinhamento entre duas ou mais imagens capturadas da mesma cena, porém obtidas por diferentes sensores, em diferentes instantes de tempo ou sob diferentes pontos de observação. A operação de registro é fundamental em processamento e análise de imagens, auxiliando as etapas de identificação e reconhecimento de objetos nas mesmas. Ela pode ser utilizada em áreas como:

- **Sensoriamento Remoto:** É possível detectar mudanças em uma sequência de imagens, fundir imagens com características diferentes para permitir que informações sejam adequadamente integradas, formar mosaicos a partir de imagens de satélites, a integração de imagens de satélite ou fotografias aéreas com mapas e a localização de regiões de interesse, como edificações e rodovias, em imagens digitais
- **Medicina:** Combinação de dados obtidos a partir de diferentes técnicas de aquisição, como tomografia computadorizada, tomografia por emissão de pósitrons, ultrassonografia ou ressonância magnética, para obter informações mais completas sobre o paciente.
- **Visão Computacional:** Recuperação de informação tridimensional por meio de visão estereoscópica, rastreamento de objetos em movimento e reconstrução de objetos.

Este trabalho tem como objetivo efetuar técnicas de registro de imagens para combinar diferentes imagens, detectando pontos-chaves das imagens e criando uma imagem panorâmica formada pela ligação entre as imagens após sua correspondência. A seção II definirá a metodologia do programa elaborado para a executar a combinação de imagens, a seção III discutirá a respeito dos experimentos realizados e, por fim, as conclusões a respeito dos experimentos e sugestões serão realizadas no seção IV.

### II. METODOLOGIA

#### A. Execução e utilização do programa

Para a criação da nova imagem, foi criado um programa na linguagem *Python*. Nele é realizado uma detecção de pontos-chave em duas imagens para uní-las e criar uma imagem panorâmica em seguida.

Para executar o programa, é necessário o *Python 3* e são necessárias as seguintes bibliotecas instaladas:

- *NumPy*: Utilizada para realizar os cálculos matriciais necessários para manipular as imagens.

- *OpenCV*: Utilizado para manipular as imagens de entrada, inicializar os descritores, detectar os pontos-chave, obter a matriz de homografia da imagem e gerar as imagens de saída.

Nas versões mais recentes do *OpenCV*, os descritores *SIFT* e *SURF* estão desabilitados por padrão devido ao algoritmo de ambos ser patenteado. Para contornar o problema, é possível habilitar a flag *OPENCV\_ENABLE\_NONFREE* utilizando o programa *CMake* ou utilizar a versão 3.4.2 ou inferior da biblioteca. A segunda opção foi a escolhida por ser a mais simples de ser executada.

Para utilizar o programa, *flags* foram configuradas para realizar experimentos de forma separada:

- **-images**: Utilizada para listar as imagens que serão combinadas para criar a nova imagem panorâmica. Parâmetro obrigatório.
- **-detector**: Utilizado para determinar o detector utilizado para obter os pontos-chave de cada imagem. Parâmetro obrigatório.
- **-threshold**: Utilizada para determinar o limiar permitido para fazer as comparações entre as distâncias.
- **-show-keypoints**: Utilizada para exibir os pontos-chaves das duas imagens.
- **-show-matches**: Utilizada para exibir as correspondências dos pontos-chave entre as imagens.

#### B. Entrada e Saída de dados

Como entrada de dados, foi utilizado um banco de imagens presente em [3], que possui 10 imagens no formato JPEG. A saída de dados serão as imagens panorâmicas resultantes também no formato JPEG, disponibilizados em uma pasta **result\_images** que o programa também criará caso não exista. Caso a flag **-show-keypoints** ou a flag **-show-matches** sejam passadas, também serão geradas imagens no formato JPEG de acordo com a finalidade de cada flag.

#### C. Abordagens tratadas nos algoritmos

O algoritmo para a geração da imagem panorâmica, conforme detalhado em [2], consiste em 8 passos:

- 1) Converter as imagens coloridas de entrada em imagens de níveis de cinza.
- 2) Encontrar pontos de interesse e descritores invariantes locais para o par de imagens.
- 3) Computar distâncias (similaridades) entre cada descritor das duas imagens.

- 4) Selecionar as melhores correspondências para cada descriptor de imagem.
- 5) Estimar a matriz de homografia.
- 6) Aplicar uma projeção de perspectiva para alinhar as imagens.
- 7) Unir as imagens alinhadas e criar a imagem panorâmica.
- 8) Desenhar retas entre pontos correspondentes no par de imagens.

Para o passo 1 é utilizada a função `cv2.imread` para ler a imagem colorida, e logo após é utilizada a função `cv2.cvtColor` configurada com a constante `cv2.COLOR_BGR2GRAY` para transformá-la em tons de cinza.

Para o passo 2, foram implementados neste programa 4 detectores diferentes:

- **SIFT** (*Scale Invariant Feature Transform*).
- **SURF** (*Speed Up Robust Feature*).
- **BRIEF** (*Binary Robust Independent Elementary Features*).
- **ORB** (*Oriented FAST, Rotated BRIEF*).

Os detectores *SIFT* e *SURF* são gerados utilizando as funções `cv2.xfeatures2d.SIFT_create` e `cv2.xfeatures2d.SURF_create` respectivamente e os pontos-chave e os descritores das imagens são gerados utilizando a função `detectAndCompute` presentes em ambos. O detector *ORB* é gerado utilizando a função `cv2.ORB_create`, os pontos-chave das imagens são gerados utilizando a função `detect` e logo em seguida a função `compute` gerará os descritores. O detector *BRIEF* é gerado em duas etapas: Para a detecção dos pontos-chave, é utilizada a função `cv2.xfeatures2d.StarDetector_create` seguida da função `detect`. Para os descritores, é utilizada a função `cv2.xfeatures2d.BriefDescriptorExtractor_create` seguida da função `compute`.

Para o passo 3 é utilizado um classificador de Força Bruta, criado com a função `cv2.BFMatcher`. Como é necessário computar as distâncias para encontrar similaridades entre as imagens, foi escolhido um classificador KNN usando a função `knnMatch` e passando os descritores encontrados no passo 2. Para o passo 4, o classificador obtém as similaridades em pares, e esses pares são comparados: Caso o primeiro valor seja menor que o segundo valor multiplicado pelo limiar passado pela flag `-threshold`, a correspondência é dada como válida. Como a flag `-threshold` não é obrigatória, foi dado um valor padrão de 0,75, pois conforme mostrado em [4] um valor acima de 0,7 já possui um valor pequeno de pontos-chave que são falsos positivos e um valor pequeno de descartes de falsos negativos. O autor mostra também que um limiar acima de 0,8 apresenta uma taxa de eliminação de falsas correspondências de 90% e uma taxa de descarte de correspondências verdadeiras de menos de 5%.

Para o passo 5 é utilizada a função `cv2.findHomography`, desde que hajam, no mínimo, 4 correspondências encontradas no passo 4. Nela é utilizada a técnica RANSAC

(RANdom SAMple Consensus), configurada com a constante `cv2.RANSAC` e passada na função.

Para o passo 6 é utilizada a função `cv2.warpPerspective`, onde uma nova imagem é gerada para depois ser combinada no passo 7, e após essa combinação é realizado um pós-processamento para cortar as partes pretas da imagem antes de gerar a imagem resultante. Para o passo 8 é utilizada a função `cv2.drawMatches` e a imagem resultante é gerada caso a flag `-show-matches` seja passada.

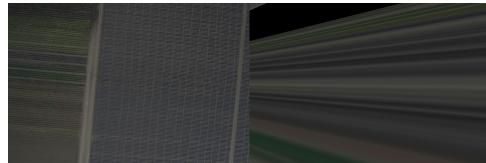
### III. EXPERIMENTOS E DISCUSSÃO

Como primeiro experimento, foi realizada a execução do programa com todas as combinações de fotos presentes em [3] e limiar padrão de 0,75 para determinar as diferenças entre os detectores de pontos-chave. Neste experimento, as principais diferenças podem ser classificadas em duas características: precisão e compatibilidade.

Quanto a compatibilidade, *SURF* e *ORB* demonstraram maior compatibilidade para gerar as imagens panorâmicas, gerando imagens em todas as ocasiões. *SIFT* não conseguiu executar a combinação das imagens *foto5A.jpg* e *foto5B.jpg* pela matriz de homografia não ter sido calculada com os pontos encontrados e *BRIEF* não conseguiu executar a mesma combinação por não encontrar pontos suficientes para calcular a matriz de homografia. *SIFT* consegue gerar uma imagem se o limiar for aumentado para 0,85, mas o resultado não é satisfatório, apresentando deformações ao aplicar a projeção de perspectiva, e *BRIEF* apresenta deformações parecidas na combinação das imagens *foto4A.jpg* e *foto4B.jpg* independentemente do limiar utilizado. Tais deformações são mostradas na Figura 1.



(a) Projeção usando BRIEF com limiar 0,75



(b) Projeção usando SIFT com limiar 0,85

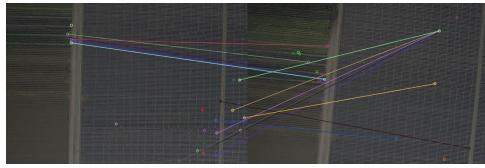
Figura 1. Deformações presentes nas projeções.

Tais resultados negativos aconteceram por fatores diferentes, ilustrados na Figura 2. No caso do detector *SIFT*, isto aconteceu por causa do número pequeno de pontos detectados, que encontrou um número de pontos baixo e isto fez com que o classificador utilizado encontrasse poucas correspondências, algumas apontando até para o mesmo ponto. No caso do

detector *BRIEF*, foi detectado um número satisfatório de pontos mas o classificador utilizado apresentou correspondências incorretas. É possível que a imagem resultante seria diferente se fosse implementado um outro classificador.



(a) Correspondências encontradas usando BRIEF



(b) Correspondências encontradas usando SIFT

Figura 2. Correspondências encontradas nas imagens com deformações.

Quanto a precisão, a combinação entre as imagens *foto1A.jpg* e *foto1B.jpg* exemplifica o que foi encontrado nas outras fotos, ilustradas em um trecho da imagem final presente na Figura 3. *SURF* e *SIFT* foram os mais precisos na hora de obter a imagem resultante final, seguido por *BRIEF* e *ORB* sendo o mais impreciso. Isso acontece pois a quantidade de correspondências geralmente é maior em *SURF* e *SIFT*, que geralmente também consegue detectar mais pontos-chave nas imagens que *BRIEF* e *ORB*. A Figura 4 mostra a diferença entre as correspondências.



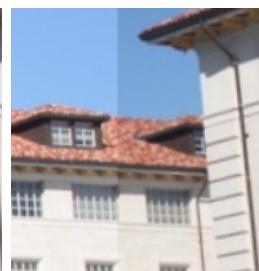
(a) BRIEF



(b) SIFT



(c) ORB



(d) SURF

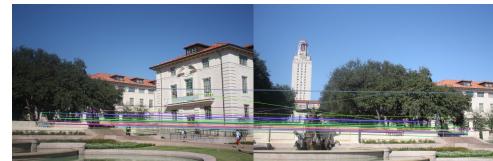
Figura 3. Projeção de perspectiva de cada detector.



(a) Correspondências encontradas usando BRIEF



(b) Correspondências encontradas usando SIFT



(c) Correspondências encontradas usando ORB



(d) Correspondências encontradas usando SURF

Figura 4. Correspondências encontradas na combinação entre as imagens *foto1A.jpg* e *foto1B.jpg*.

Após o primeiro experimento, foi concluído que o detector *SURF* é o que mais se adaptou a diversas situações de imagens considerando rotação, escala e perspectiva, dando resultados muito satisfatórios em todas as imagens disponíveis. Como segundo experimento, usamos a combinação entre as imagens *foto4A.jpg* e *foto4B.jpg* usando o detector *SURF* que obteve melhores resultados, mas alterando o limiar para verificar como sua alteração de valor afeta a projeção de perspectiva.

Foram usados os valores 0,1, 0,3, 0,5, 0,7 e 0,9. Com o valor 0,1 a matriz de homografia não foi calculada com os pontos encontrados, mas com os outros limiares a matriz foi encontrada e a imagem foi gerada. Limiares acima de 1 não foram calculados pois poderiam gerar uma precisão menor que os limiares no intervalo  $[0, 1]$ , sendo irrelevantes para o resultado. A Figura 5 ilustra os resultados através de um trecho da combinação.

Com o limiar 0,3, a projeção mostra uma precisão baixa, com certos trechos não mostrando continuidade e um "corte" entre as duas imagens bastante aparente. A partir do limiar 0,5 o corte não é mais aparente e conforme este limiar aumenta é notado um aumento de detalhes e uma continuidade das formas, formando imagens panorâmicas mais precisas. Novamente, isso se deve graças ao número de correspondências presentes, ilustradas na Figura 6. Conforme o limiar diminui,

o número de correspondências também diminuia e a precisão das combinações fica menor. Esse experimento reforça o que foi explicado em [4], uma vez que quanto maior o limiar no intervalo  $[0, 1]$ , menor o número de correspondências falsas e menor o número de descartes de correspondências verdadeiras. Se o limiar estivesse no intervalo  $[1, \infty]$ , a grandeza continuaria na mesma proporcionalidade até chegar em um valor máximo, uma vez que o limiar é aplicado entre as distâncias dessas correspondências. Entretanto, não é interessante deixar todas as correspondências devido a algumas delas que podem ser consideradas falsos positivos.



Figura 5. Projeção de perspectiva de cada limiar usando SURF.

#### IV. CONCLUSÕES E TRABALHOS FUTUROS

Podemos concluir que é possível unir duas imagens usando diversos tipos de detecção de pontos-chave e descritores para encontrar similaridades entre elas e realizar a combinação de modo correto. Entretanto, podemos notar também que há certas imprecisões e incompatibilidades dependendo das imagens a serem combinadas, dos detectores de pontos escolhidos, os limiares de distância configurados e do classificador de descritores escolhido pelo programador.

Por exemplo, apesar do detector *SIFT* ser considerado um detector muito preciso, ele foi considerado incompatível com certas combinações de imagens, enquanto o detector *ORB* foi considerado impreciso apesar de ser compatível com mais situações. A escolha do limiar também pode melhorar a precisão da combinação, com um número ideal entre 0,7 e 0,9.

É possível perceber também que fatores externos podem definir a escolha de um detector. *SURF* e *SIFT* tiveram os melhores resultados como um todo, mas como são detectores com algoritmos patenteados, é possível que um desenvolvedor

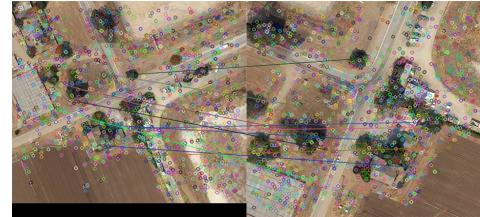


Figura 6. Correspondências encontradas na combinação entre as imagens foto4A.jpg e foto4B.jpg usando SURF.

escolha *BRIEF* ou *ORB* dependendo de suas prioridades de seu código para aliviar o orçamento de seu projeto.

Como trabalhos futuros, sugere-se a implementação com novos classificadores para verificar se a mudança do algoritmo de classificação é benéfica para os detectores ou não, a implementação de combinação de 3 ou mais imagens de entrada para formar a imagem panorâmica resultante e, consequentemente, testes com mais tipos de imagens diferentes.

#### REFERÊNCIAS

- [1] [http://www.ic.unicamp.br/~helio/disciplinas/MO443/aula\\_registro.pdf](http://www.ic.unicamp.br/~helio/disciplinas/MO443/aula_registro.pdf)
- [2] <http://www.ic.unicamp.br/~helio/disciplinas/MO443/trabalho4.pdf>
- [3] [http://www.ic.unicamp.br/~helio/imagens\\_registro/](http://www.ic.unicamp.br/~helio/imagens_registro/)
- [4] <https://www.cs.ubc.ca/~lowe/papers/ijcv04.pdf>