MC920 – Introdução ao Processamento Digital de Imagem
Trabalho 4

Rafael Eiki Matheus Imamura - RA 176127[1]

[1]Instituto de Computação – Universidade Estadual de Campinas (Unicamp)
Caixa Postal 6176 – CEP 13083-970 – Campinas – SP – Brasil

ra176127@students.ic.unicamp.br

1. Introdução

Em diversas aplicações, é de grande interesse poder combinar imagens a fim de se obter uma visão mais completa da figura fotografada. Desde entretenimento e desejos pessoais até aplicações como visualizações por satélite, as fotos panorâmicas são bem versáteis em sua utilidade. No entanto, fazer esse processo manualmente pode ser bastante custoso, além de impreciso.

Neste trabalho foi criado um algoritmo que a partir de duas imagens JPEG (*Joint Photographic Experts Group*), gera uma foto panorâmica, combinando as duas. Além de gerar a imagem combinada, também são mostrados os pontos de semelhança da imagem. O algoritmo possibilita o uso de algoritmos de descritores de imagens diferentes e suas implicações são discutidas neste relatório.

2. Execução

A entrega deste trabalho inclui o presente PDF e um diretório de nome "projeto 4". Nela se encontram o código, as dependências e as imagens de entrada usadas para os testes.

Para executar o código, deve-se ter instalado o OpenCV e o Numpy. Um *freeze* das dependências está presente no arquivo "requirements.txt". Uma alternativa rápida para fazer a instalação é o comando:

pip install -r requirements.txt

Com este comando instalado, é possível chamar o arquivo de script de duas formas.

1. *python main.py example threshold*

   Nesta opção, são gerados os resultados para todas as imagens de exemplo disponíveis no diretório "pictures", usando todos os algoritmos, com o limiar dado pelo valor de "threshold".

2. *python main.py arq1 arq2 algoritmo limiar diretório_de_saída arq_saída*

   Nesta opção, é gerada a imagem usando um algoritmo de descritor específico. O significado de cada parâmetro é detalhado a seguir:

   1. *arq1* e *arq2*: arquivos de imagem de entrada, no formato JPEG;
   2. *algoritmo*: algoritmo do descritor a ser usado. Pode ser *orb* ou *brief*.
   3. *limiar*: valor do limiar para considerar uma correspondência boa;

4. *diretório_de_saída:* diretório de saída das imagens (deve incluir uma "/" no final);
5. *arq_saída:* nome base do arquivo de saída (sem extensão).

Alguns exemplos de uso são mostrados a seguir:

1. python main.py example 0.7
2. python ./main.py ./pictures/foto1A.jpg ./pictures/foto1B.jpg orb 0.75 ./results/ foto1
3. python ./main.py ./pictures/foto2A.jpg ./pictures/foto2B.jpg orb 0.8 ./results/ foto2

2.1. Entrada

A entrada, em todos os casos, é dada por imagens JPEG. Para este relatório, foram usadas 12 imagens (6 pares), das quais 10 foram fornecidas no enunciado do trabalho (*http://www.ic.unicamp.br/~helio/imagens_registro/*) e 2 foram registros fotográficos realizados no Instituto de Computação da Unicamp.

Todas as entradas usadas estão disponíveis dentro do zip, no diretório "pictures".

2.2. Saída

A saída do programa são imagens no formato JPEG. Para cada par de imagens é gerado um par de imagens com os nomes "BASE_lines.jpeg" e "BASE_panoramic.jpeg". Elas representam, respectivamente, as semelhanças entre as duas imagens, ligando os pontos por retas, e a imagem panorâmica resultante. Em casos de que não sejam encontradas correspondências o suficiente (mais detalhes a seguir), o programa não gera as imagens e um erro é dado.

3. Solução

A solução geral criada possui o seguinte formato:

1. **Ambas as imagens de entrada são lidas e convertidas em escala de cinza**

O primeiro passo é converter as imagens em níveis de cinza. Ambas as imagens (original, colorida) e em tons de cinza são guardadas.

2. **São encontrados os pontos de interesse e descritores para cada imagem, usando um dos algoritmos (ORB ou BRIEF)**

Os pontos de interesse e descritores de cada uma das imagens (versão escala de cinza) são computados. Como funciona o processo depende do algoritmo escolhido, o que é detalhado mais adiante.

3. **As similaridades entre cada descritor são computadas e são selecionados as melhores correspondências**

Neste passo, são procuradas as similaridades entre as imagens, com um matcher do OpenCV que depende do algoritmo que é usado. Neste caso em específico, os matchers do ORB e BRIEF são iguais, pois usam o *cv2.NORM_HAMMING* sem *crossCheck*. Mais detalhes destes métodos podem ser vistos na documentação do OpenCV.

As similaridades são então filtradas por pares, sendo mantidas apenas aquelas que satisfazem a fórmula a seguir. Dadas duas correspondências A e B, a correspondência A é armazenada se:

*Distancia(A) < Distancia(B) * limiar*

Onde *Distancia(X)* é a informação da distância da correspondência e o *limiar* é uma das entradas do programa. Quanto menor o limiar, mais restritivo é a seleção de similaridades. As 15 melhores semelhanças são então desenhadas em um arquivo.

4. **É executada a técnica RANSAC (*RANdom SAmple Consensus*), estimando a matriz de homografia**

Usando o método do OpenCV *cv2.findHomography* com o parâmetro *cv2.RANSAC*, é estimada a matriz de homografia, usada para realizar a projeção posteriormente.

5. **Usando as informações anteriores das imagens em escala de cinza, as versões coloridas das imagens são alinhadas e sobrepostas, formando a imagem panorâmica**

Usando o método *cv2.warpPerspective* do OpenCV, uma nova imagem é criada, com a soma da altura das duas imagens originais, e largura de ambas as imagens somadas. A imagem é posicionada levando em conta a posição da figura a esquerda. A imagem resultante é preta nas regiões que não há conteúdo da foto panorâmica. Essas regiões pretas são removidas no final do processo.

6. **As imagens coloridas também geram uma nova imagem, onde há segmentos de retas ligando os pontos de semelhança**

É gerada uma imagem que ambas as imagens de entrada estão lado a lado. Suas semelhanças são destacadas através de retas que as ligam.

BRIEF (Binary Robust Independent Elementary Features) é uma técnica de descrição de *features*. Usando uma seleção de pares da imagem, ele se apoia na velocidade de cálculo da distância de Hamming para strings binárias que representam os descritores. Um ponto fraco do BRIEF é possuir problemas com rotações em imagens. Este não é um método que encontra os pontos de interesse da imagem, e por isso é aplicado no projeto usando o CenSurE (*STAR detector* do OpenCV).

O ORB (*Oriented FAST and Rotated BRIEF*) é uma alternativa de boa performance e com custo computacional páreo ao de alguns grandes concorrentes, como o SIFT e o SURF. Ele usa o FAST para encontrar os pontos de interesse e descritores BRIEF. O algoritmo ainda possui modificações que encobrem problemas conhecidos de sua base, já que o FAST é sensível a rotações e o BRIEF tem dificuldades com elas.

4. Resultados

Os resultados para cada par de imagens de teste são apresentados a seguir. Todas são fotografias JPEG coloridas. Vamos começar com o par de imagens da Figura 1.

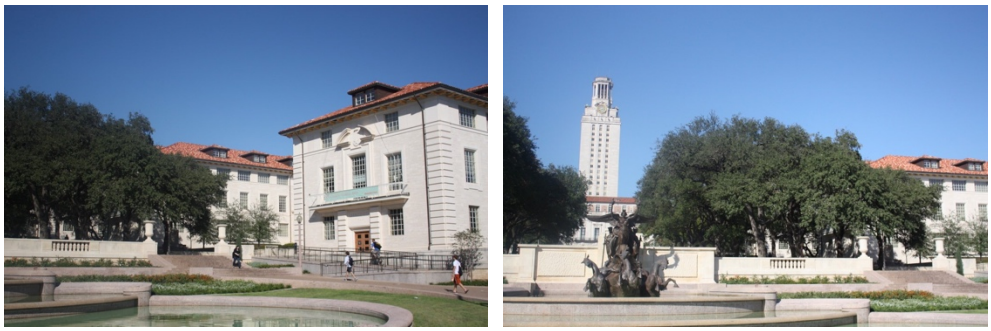

Figura 1. Par de imagens “foto1A” e “foto1B”.

Ao usar esse par de imagens no programa, o resultado são as imagens a seguir, que mostram as semelhanças entre o par de imagens. A Figura 2 mostra as correspondências encontradas com o ORB e a Figura 3, com o BRIEF. Para ambos os casos, o valor de limiar usado é 0,75. Este limiar foi adotado por, dentre os testes, mostrar os melhores resultados nas imagens. Os testes foram gerados com diferenças de 0.05, no intervalo de 0.5 (altamente restritivo) a 0.9 (altamente permissivo).

A diferença das duas é bastante visível, com o BRIEF tendo muito mais pontos que o ORB. O ORB é um método modificado para pegar pontos específicos, o que faz sentido com a imagem apresentada.

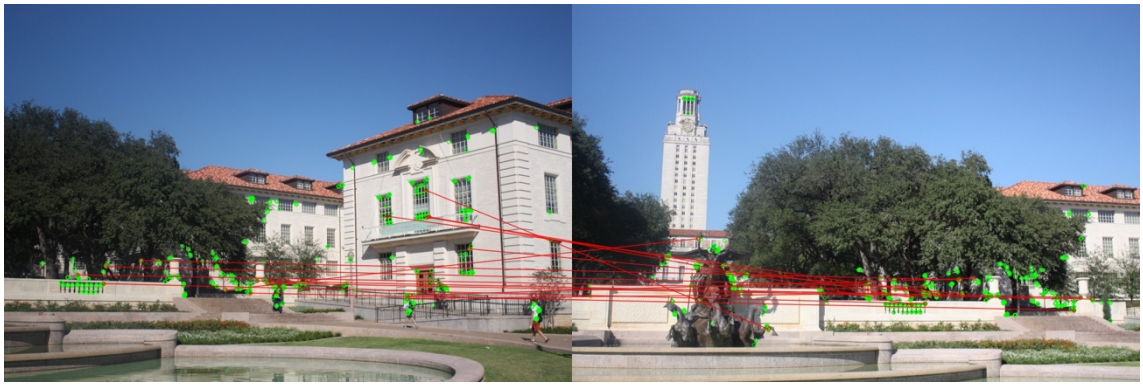

Figura 2. Semelhanças entre a “foto1A” e “foto1B” usando o ORB.

Figura 3. Semelhanças entre a “foto1A” e “foto1B” usando o BRIEF.

Como imagem panorâmica resultante, ambos os métodos resultaram em fotos praticamente idênticas. Veja a comparação com as Figuras 4 e 5.

Figura 4. Resultado da imagem panorâmica para o par "foto1" usando ORB.

Figura 5. Resultado da imagem panorâmica para o par "foto1" usando BRIEF.

São apresentados agora mais alguns dos exemplos que destoam mais entre os métodos. O par de fotos "foto5A" e "foto5B" não encontraram semelhanças o suficiente usando o método BRIEF com nenhum dos limiares usados. Os resultados com o ORB são mostrados nas Figuras 6 e 7.

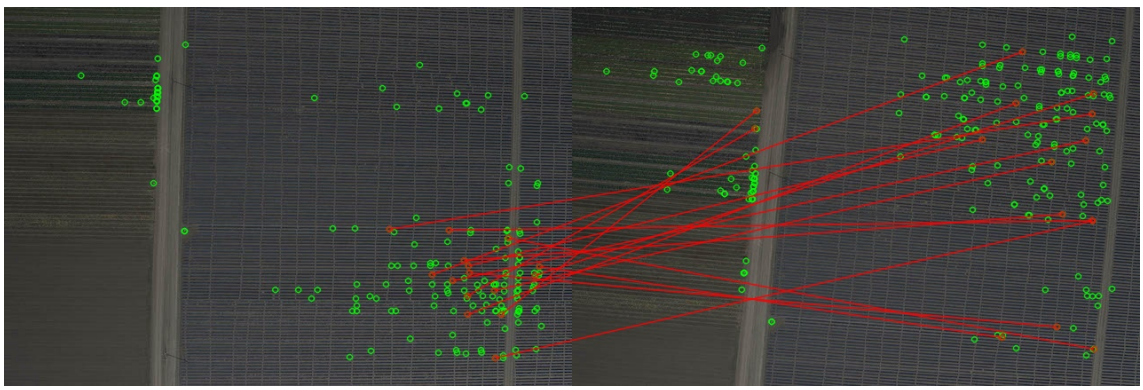

Figura 6. Correspondências entre o par "foto5" usando ORB.

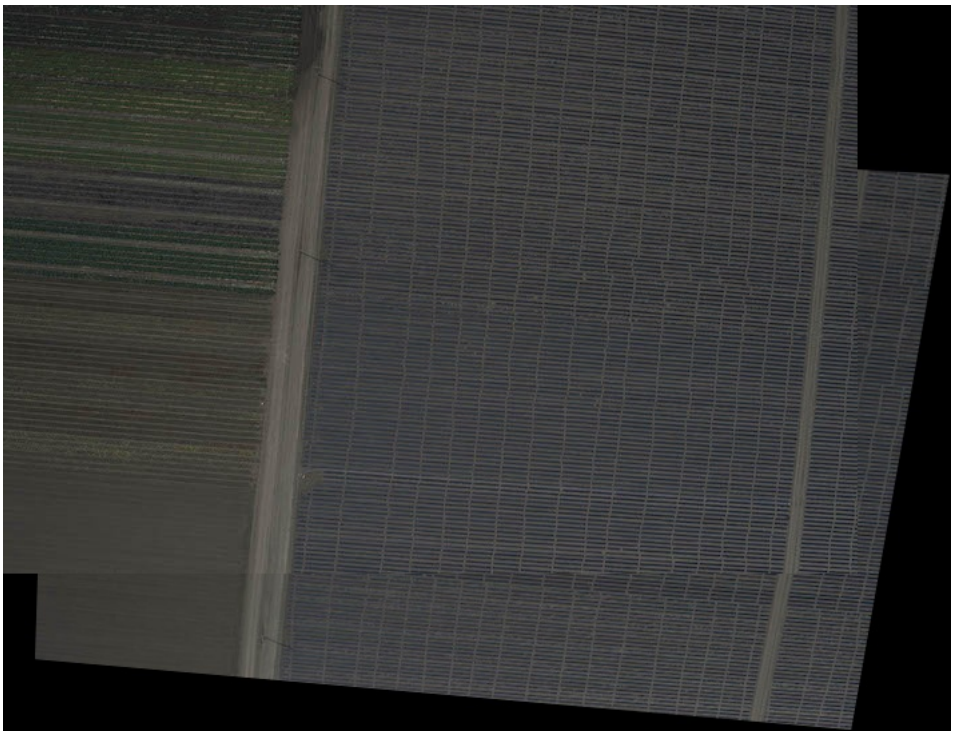

Figura 7. Imagem panorâmica criada a partir do par “foto5” usando ORB.

Os resultados são bastante positivos, mostrando a imagem panorâmica alinhada e coerente com as imagens originais.

Para testar uma imagem um pouco mais pesada (1.5MB, enquanto as demais eram cerca de 100KB) e tirada por uma câmera digital de celular comum, foi criado o par de imagens “foto6”, que iremos analisar a seguir (Figura 8). O método ORB não conseguiu obter resultados para o par com o limiar 0,75, obtendo somente quando o limiar foi aumentado para 0.8 (Figura 9). Em nenhum dos casos o resultado com o ORB foi positivo, enquanto que o BRIEF teve alinhamento bom visualmente (Figura 10).

Figura 8. Par de imagens “foto6A” e “foto6B”.

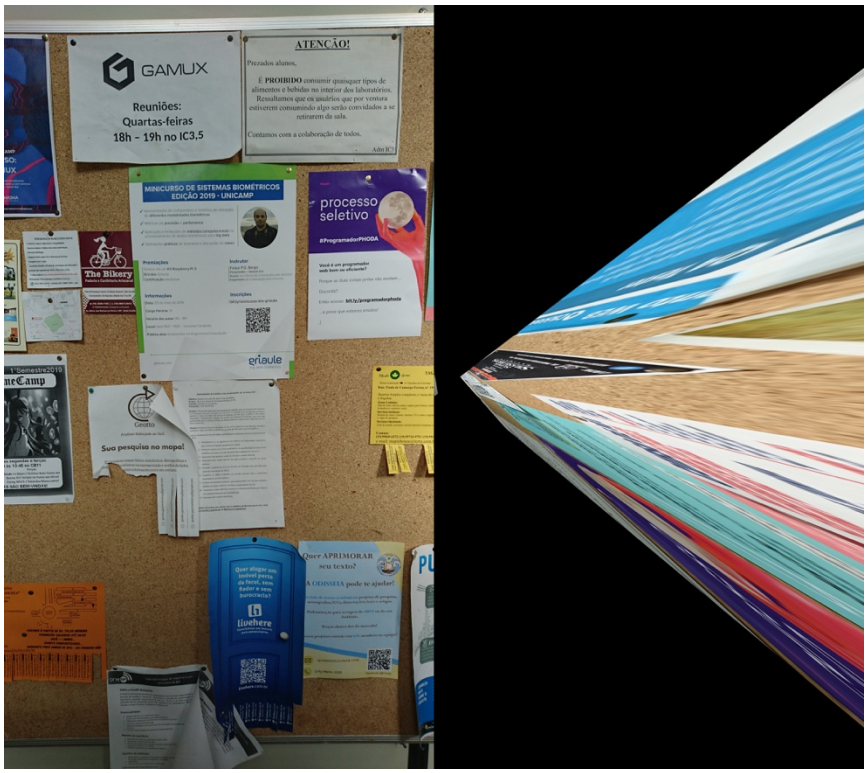

Figura 9. Imagem panorâmica do par "foto6" usando o ORB e limiar 0.8.

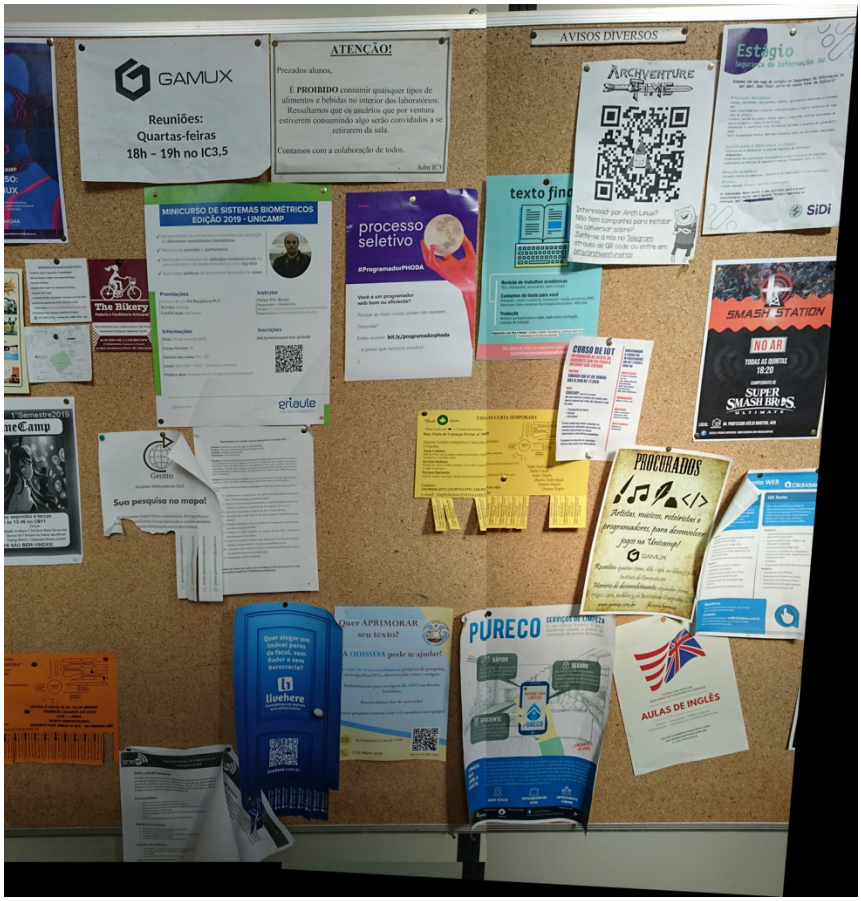

Figura 10. Imagem panorâmica do par "foto6" usando o BRIEF e limiar 0.75.

5. Conclusões

Neste trabalho, foi desenvolvido um programa que gera imagens panorâmicas usando pares de imagens JPEG. Foi possível analisar o funcionamento dos algoritmos usando diferentes descritores e limiares.

No geral, o limiar ótimo para os resultados pareceu ser 0,75. Isso pode ser limitado devido a quantidade de testes realizados. Do ponto de vista das imagens geradas, geralmente o BRIEF e o ORB foram muito próximos quando não eram iguais, exceto nos pares 5 e 6. Os resultados de 5 para o BRIEF estão de acordo, já que ele tem maior dificuldade com rotações. A falta dos pontos de interesse no ORB pode ter feito ele não conseguir encontrar a semelhança corretamente nas fotos 6A e 6B.

O programa ainda se mostrou funcional com diferentes tipos de imagens, desde fotografias profissionais, de satélite, até tiradas com um dispositivo celular comum. Com um tempo de processamento relativamente baixo, este tipo de aplicação pode ser usado para processamentos em tempo real, levando a novas aplicações.

Referências

OpenCV. (2013) "BRIEF (Binary Robust Independent Elementary Features)". Disponível em: < https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_feature2d/py_brief/py_brief.html>. Acesso em: jun. 2019.

OpenCV. (2013) "ORB (Oriented FAST and Rotated BRIEF)". Disponível em: < https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_feature2d/py_orb/py_orb.html>. Acesso em: jun. 2019.

SciPy.org. (2019) "NumPy Reference". Disponível em: < https://docs.scipy.org/doc/numpy/reference/>. Acesso em: jun. 2019.