

# Relatório - Trabalho 04

## MO443 - Introdução ao Processamento de Imagem Digital

VINICIUS TEIXEIRA DE MELO - RA: 230223

Universidade Estadual de Campinas  
[viniciusteixeira@liv.ic.unicamp.br](mailto:viniciusteixeira@liv.ic.unicamp.br)

8 de Junho de 2019

### I. ESPECIFICAÇÃO DO PROBLEMA

O objetivo deste trabalho é aplicar técnicas de detecção de pontos de interesse para registrar um par de imagens e criar uma imagem panorâmica formada pela ligação entre as imagens após sua correspondência.

Os principais passos do processo de correspondência e geração da imagem panorâmica são:

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 descritor de imagem;
5. executar a técnica *RANSAC* (*RANdom SAmple Consensus*) para estimar a matriz de homografia (*cv2.findHomography*).
6. aplicar uma projeção de perspectiva (*cv2.warpPerspective*) para alinhar as imagens;
7. unir as imagens alinhadas e criar a imagem panorâmica;
8. desenhar retas entre pontos correspondentes no par de imagens.

No passo (2), é necessário explorar e comparar diferentes detectores de pontos de interesse e descritores, tais como *SIFT* (*Scale Invariant*

*Feature Transform*), *SURF* (*Speed Up Robust Feature*), *BRIEF* (*Robust Independent Elementary Features*) e *ORB* (*Oriented FAST, Rotated BRIEF*). No passo (4), uma correspondência será considerada se o limiar definido estiver acima de um valor especificado pelo usuário. No passo (5), o cálculo da matriz de homografia requer o uso de, no mínimo, 4 pontos de correspondência.

A fundamentação e resolução deste trabalho estão descritas nas seções seguintes.

### II. ENTRADA DE DADOS

O código fonte criado para a execução de todas as tarefas está no notebook **Trabalho 04.ipynb**. O código foi criado para aceitar imagens coloridas ou em escala de cinza em diversos formatos, como exemplo: *PNG* (*Portable Network Graphics*), *JPG* (*Joint Photographic Group*), entre outros.

Para executar o notebook, basta iniciar o ambiente *Jupyter Notebook*, abrir o notebook **Trabalho 04.ipynb** e executar as células em ordem. Todo o algoritmo foi implementado na linguagem Python na versão 3.6.

As imagens de entrada utilizadas nos testes dos algoritmos foram retiradas da página do prof. Hélio Pedrini: [Imagens](#). Na pasta **imgs/** estão as imagens coloridas utilizadas nos testes: **foto1A.jpg** (1024 x 683 pixels), **foto1B.jpg** (1024 x 683 pixels), **foto3A.jpg** (640 x 427 pixels) e **foto3B.jpg** (640 x 427 pixels).

**Figura 1:** foto1A.jpg**Figura 2:** foto1B.jpg**Figura 3:** foto3A.jpg**Figura 4:** foto3B.jpg

### III. DEPENDÊNCIAS E CÓDIGOS

As bibliotecas utilizadas neste trabalho foram:

Biblioteca	Versão
numpy	1.16.2
cv2	3.4.2
matplotlib	3.0.3
warnings	2.1

A leitura das imagens foi realizada utilizando uma função do **matplotlib** [3] chamada **pyplot.imread()**, a qual necessita apenas do caminho da imagem, já que queremos as informações das imagens assim como estão.

Foi criada uma função para plotar os resultados intermediários e finais utilizando também uma função do **matplotlib** chamada **pyplot.imshow()**. No passo (1) da descrição do trabalho foi utilizada uma função do **opencv** [1], denominada **cv2.cvtColor()**, para transformar a imagem lida inicialmente em uma imagem em escala de cinza, para isso foi necessário a utilização da constante **cv2.COLOR\_BGR2GRAY**.

Para a utilização dos descritores pedidos no trabalho, foram utilizadas as funções do próprio **opencv**: **SIFT** **cv2.xfeatures2d.SIFT\_create()**, **SURF** **cv2.xfeatures2d.SURF\_create()**, **BRIEF** **cv2.xfeatures2d.BriefDescriptorExtractor\_create()** e **ORB** **cv2.ORB\_create()**.

No passo (3) e (4), como é necessário calcular a distância entre os *keypoints* da imagens em questão e utilizar somente os pontos que possuem uma melhor correspondência, utilizamos a função do **opencv knnMatch()** para fazer esse cálculo e uma taxa de 0.8 para termos os pares de pontos que possuem correspondência válida.

No passo (5) é necessário o cálculo da matriz de homografia, como já sugerido na descrição, utilizamos a função **cv2.findHomography()** com a técnica **RANSAC** (**cv2.RANSAC**) e um limiar de 4.0. Para o cálculo da projeção de perspectiva foi utilizada a função **cv2.warpPerspective()** e, por fim, foram dese-

nhadas as retas entre cada par de pontos que possuem uma maior similaridade.

## IV. FUNDAMENTAÇÃO

### i. Descritores

Dada um imagem qualquer, pontos de interesse de um objeto pertencente a imagem podem ser extraídos para que se tenha uma descrição de características desse objeto. A seguir estão as descrições dos descritores utilizados nesse trabalho.

#### i.1 SIFT

O descritor SIFT (*Scale-invariant feature transform*) é um algoritmo de detecção de características em Visão Computacional para detectar e descrever características locais em imagens. Um ponto de interesse SIFT é uma região de imagem circular com uma orientação. Ele é descrito por um quadro geométrico de quatro parâmetros: o centro do ponto-chave coordenado  $x$  e  $y$ , sua escala (o raio da região) e sua orientação (um ângulo expresso em radianos). O detector SIFT usa como estruturas de imagem de pontos chave que se assemelham a "blobs". Ao procurar por blobs em várias escalas e posições, o detector SIFT é invariante (ou, mais precisamente, covariante) para translação, rotações e redimensionamento da imagem.

#### i.2 SURF

O descritor SURF (*Speeded up robust features*) é um detector e descritor de características locais patenteado. Pode ser usado para tarefas como reconhecimento de objetos, registro de imagens, classificação ou reconstrução 3D.

Para detectar pontos de interesse, o SURF usa uma aproximação inteira do determinante do detector de *blob Hessian*, que pode ser calculado com 3 operações inteiras usando uma imagem integral pré-calculada. Seu descritor de característica é baseado na soma da resposta da *wavelet Haar* em torno do ponto de interesse. Estes também podem ser calculados com o auxílio da imagem integral.

#### i.3 BRIEF

O descritor BRIEF (*Binary Robust Independent Elementary Features*) fornece um atalho para encontrar as cadeias binárias diretamente, sem encontrar descritores. Dado uma imagem, ele seleciona um conjunto de pares de localizações  $n_d(x, y)$  em um único caminho. Então, algumas comparações de intensidade de *pixels* são realizadas nesses pares de localizações. Por exemplo, suponha dois pares de localizações  $p$  e  $q$ , se  $I(p) < I(q)$ , então o resultado é 1, se não, o resultado é 0. Isso é aplicado para todos os pares de localizações para obter uma string binária  $n_d$ -dimensional. E, assim, realizar os cálculos dos pontos de interesse.

#### i.4 ORB

O descritor ORB (*Oriented FAST and rotated BRIEF*) é um detector rápido de características locais. É baseado no detector de pontos de interesse FAST e no descritor visual BRIEF. Seu objetivo é fornecer uma alternativa rápida e eficiente para o SIFT.

### ii. RANSAC

RANSAC é uma abreviatura de "*RAndom SAmple Consensus*". É um método iterativo para estimar os parâmetros de um modelo matemático [4].

Dado um modelo que requer um mínimo de  $n$  pontos para instanciar seus parâmetros livres, e um conjunto de pontos  $P$ , tal que o numero de pontos em  $P$  é maior que  $n$ , seleciona randomicamente um subconjunto  $S_1$  de  $n$  pontos de  $P$  e instancia o modelo. Usa-se o modelo instanciado  $M_1$  para determinar o subconjunto  $S_{1*}$  de pontos em  $P$  que estão dentro de um erro tolerável de  $M_1$ . O conjunto  $S_{1*}$  é chamado de conjunto consenso de  $S_1$ .

Se o número de elementos de  $S_{1*}$  é maior que certo limite  $t$ , que é função da estimativa do número de erros grosseiros em  $P$ , usar  $S_{1*}$  para computar (possivelmente com Mínimos Quadrados) um novo modelo  $M_{1*}$ .

Se o número de elementos de  $S_{1*}$  é menor do que  $t$ , seleciona-se randomicamente um

novo subconjunto  $S_2$  e repete-se o processo acima. Se, após um número pré determinado de tentativas, nenhum conjunto consenso com  $t$  ou mais membros tiver sido encontrado, ou soluciona-se o modelo com o maior conjunto consenso, ou termina-se com falha.

## V. SAÍDA DE DADOS

As imagens resultantes foram salvas dentro da pasta **resultados/** utilizando uma função da biblioteca **matplotlib** chamada **pyplot.imsave()** [3].

O formato dos nomes de saída estão da seguinte forma: as imagens que representam as projeções de perspectiva possuem os nomes "descritor\_persp\_inter\_nomedafoto.jpeg"; as imagens que representam as junções das projeções de perspectiva possuem os nomes "descritor\_persp\_nomedafoto.jpeg"; e, por fim, as imagens que representam as projeções de perspectiva com as retas de correspondências desenhadas possuem os nomes "descritor\_vis\_nomedafoto.jpeg".

## VI. RESULTADOS E DISCUÇÕES

Os resultados obtidos com a aplicação do passo a passo definido no problema estão a seguir:



**Figura 5:** *sift\_persp\_inter\_foto1.jpeg*

A Figura 5 apresenta o resultado do passo (6), o qual é necessário aplicar uma projeção de perspectiva para alinhar as imagens. Essa figura mostra como ficou o resultado da imagem que foi transformada com base nos pontos que foram obtidos com o descritor *SIFT*. Podemos destacar que a imagem sofre uma transformação não linear, com uma pequena rotação para

que pudesse ficar se assemelhar a continuidade da imagem 1.



**Figura 6:** *surf\_persp\_inter\_foto1.jpeg*

A Figura 6 apresenta também o resultado obtido após o passo (6), porém, com o descritor *SURF*. Podemos destacar que os resultados foram parecidos com a imagem anterior e que a imagem resultante sofreu, praticamente, as mesmas transformações.



**Figura 7:** *brief\_persp\_inter\_foto1.jpeg*

A Figura 7 apresenta o resultado obtido após o passo (6), mas utilizando os pontos de interesse obtidos com o descritor *BRIEF*. Assim como as duas imagens anteriores, os resultados foram bem semelhantes e sofreram, visualmente, as mesmas transformações.



**Figura 8:** *orb\_persp\_inter\_foto1.jpeg*

A Figura 8 mostra o resultado obtido após o passo (6) também, porém, utilizando os pontos de interesse resultantes da aplicação do descritor *ORB*. Podemos observar que a imagem não se assemelha as anteriores, o que é causado

pelo fato de o descritor *ORB* ter resultado em uma quantidade consideravelmente diferente de pontos de interesse dos descritores aplicados anteriormente. Assim, essa figura sofreu transformações diferentes, o que vai ficar claro na execução do passo (7).



**Figura 9:** *sift\_persp\_foto1.jpeg*

A Figura 9 mostra o resultado da aplicação do passo (7), o qual é necessário unir as imagens alinhadas e criar a imagem panorâmica a partir disso. Podemos observar que o resultado foi satisfatório, as imagens se alinharam muito bem na “linha de intersecção” e formaram uma imagem panorâmica assim como é descrito no passo (7).



**Figura 10:** *surf\_persp\_foto1.jpeg*

A Figura 10 apresenta o resultado obtido após a execução do passo (7), mas com a utilização do descritor *SURF*. Podemos destacar que assim como visto nos resultados do passo (6), os descritores *SIFT* e *SURF* descrevam pontos semelhantes e, por tanto, as transformações aplicadas nas imagens também foram parecidas, o que resultou em imagens panorâmicas semelhantes também.



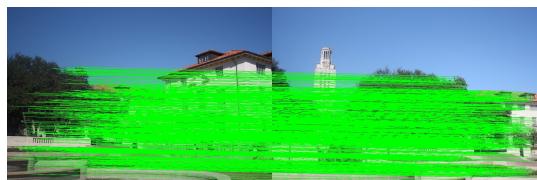
**Figura 11:** *brief\_persp\_foto1.jpeg*

A Figura 11 mostra também o resultado obtido após a execução do passo (7), porém, com a utilização do descritor *BRIEF* para descrever os pontos de interesse nas imagens. Assim como nas duas imagens anteriores, o resultado da imagem panorâmica foi bem parecido, o que reforça a questão dos descritores terem escolhidos pontos de interesse semelhantes.



**Figura 12:** *orb\_persp\_foto1.jpeg*

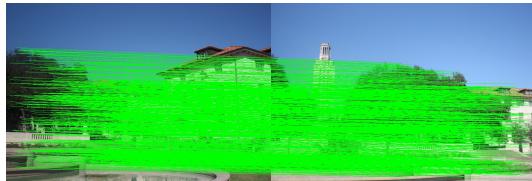
Como última figura do passo (7), a Figura 12 apresenta o resultado da imagem panorâmica utilizando o descritor *ORB* para definir os pontos de interesse. Podemos ver algumas falhas em comparação com as desmais imagens panorâmicas, isso se deve ao fato da grande diferença de detecção de pontos de interesse entre o descritor *ORB* e os demais descritores, como será evidenciado a seguir.



**Figura 13:** *sift\_vis\_foto1.jpeg*

Na Figura 13 podemos ver o resultado do passo (8), o qual é necessário desenhar as retas entre os pontos correspondentes no par

de imagens em questão. Podemos destacar que o descritor *SIFT* conseguiu detectar uma grande quantidade de pontos de interesse nas duas imagens e, com isso, obteve um ótimo resultado quando aplicado nessas imagens de entrada. Vale destacar também que, como os pontos de interesse não estão concentrados em uma mesma região, pode-se obter melhores resultados no passo de geração da projeção em perspectiva.



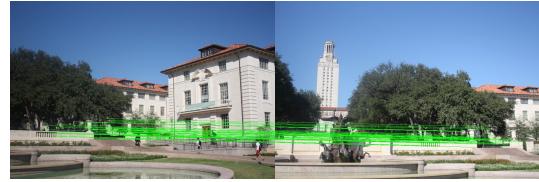
**Figura 14:** *surf\_vis\_foto1.jpeg*

A Figura 14 apresenta também o resultado do passo (8), porém, utilizando o descritor *SURF* para obter os pontos de interesse. Assim como na imagem referente ao descritor *SIFT*, essa figura mostra que o descritor *SURF* conseguiu detectar uma grande quantidade de pontos de interesse e em diferentes regiões das imagens. Podemos dizer que esses dois descritores se mostraram equivalentes quando aplicados a essas imagens de entrada.



**Figura 15:** *brief\_vis\_foto1.jpeg*

A Figura 15 apresenta o resultado do passo (8) utilizando como descritor o *BRIEF*. Podemos observar que foi detectado uma quantidade bem menor de pontos de interesse, quando comparado ao *SIFT* e *SURF*, porém os pontos de interesse detectados são bem diversificados em regiões da imagem, o que faz com que o resultado seja equivalente aos descritores anteriores.



**Figura 16:** *orb\_vis\_foto1.jpeg*

A Figura 16 apresenta o resultado do passo (8), mas utilizando o descritor *ORB* para detectar os pontos de interesse. Podemos observar que foi o descritor que obteve a menor quantidade de pontos de interesse e todos eles em uma mesma região da imagem, o que prejudica quando é necessário fazer a projeção de perspectiva do passo (6). Com isso, podemos ver que o resultado obtido com esse descritor é inferior quando comparado aos demais.

## VII. CONCLUSÃO

Podemos concluir que os resultados obtidos com as aplicações das técnicas de descrição de imagens e definição dos pontos de interesses nas imagens para a criação da imagem panorâmica e o traçado das retas entre os pontos de interesse foram satisfatórios quanto a especificação do trabalho e se mostrou razoável para este objetivo, assim, consolida ainda mais os conceitos vistos em sala de aula.

## REFERÊNCIAS

- [1] Welcome to opencv documentation! <https://docs.opencv.org/2.4/index.html>  
Acesso em: 08/06/2019.
- [2] Pedrini, Hélio, and William Robson Schwartz. Análise de imagens digitais: princípios, algoritmos e aplicações. Thomson Learning, 2008.
- [3] Matplotlib Version 3.0.3 <https://matplotlib.org/contents.html>  
Acesso em: 08/06/2019.
- [4] Wikipedia: RANSAC <https://pt.wikipedia.org/wiki/RANSAC>  
Acesso em: 08/06/2019.

## ANEXOS



**Figura 17:** *sift\_persp\_inter\_foto2.jpeg*



**Figura 21:** *sift\_persp\_foto2.jpeg*



**Figura 18:** *surf\_persp\_inter\_foto2.jpeg*



**Figura 22:** *surf\_persp\_foto2.jpeg*



**Figura 19:** *brief\_persp\_inter\_foto2.jpeg*



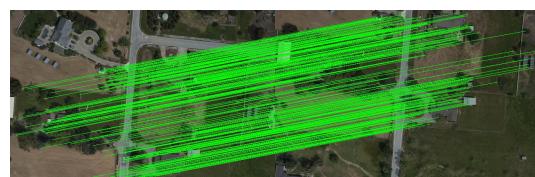
**Figura 23:** *brief\_persp\_foto2.jpeg*



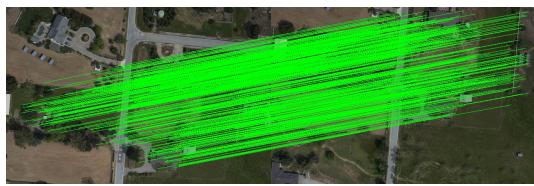
**Figura 20:** *orb\_persp\_inter\_foto2.jpeg*



**Figura 24:** *orb\_persp\_foto2.jpeg*



**Figura 25:** *sift\_vis\_foto2.jpeg*



**Figura 26:** *surf\_vis\_foto2.jpeg*



**Figura 27:** *brief\_vis\_foto2.jpeg*



**Figura 28:** *orb\_vis\_foto2.jpeg*