Digital Image Processing: 4th assignment

Tiago Moreira Trocoli da Cunha
RA: 226078
t226078@dac.unicamp.br

May 2019

1 Introduction

The fourth assignment involves image registration and explains step by step how a pair of scenes can be aligned into a single integrated image. In order to do that, it is required to find key points invariant to scale, rotation and translation between scenes, find their best descriptors through any distant measurement and select the best matches for each descriptor. The assignment also mentions four feature detector algorithms that could be used, that are, SIFT (Scale Invariant Feature Transform), SURF (Speed Up Robust Feature), BRIEF (Robust Independent Elementary Features) e ORB (Oriented FAST, Rotated BRIEF). It is also imperative to calculate the homography matrix to find the perspective transformation of the target images.

2 Program Specification

The code is written using Python 3+ in Ubuntu 18.04. I used the following libraries:

- **cv2** (opencv): used for loading and saving images, detect key points, find matches and find homography.
- **numpy**: used for matrix operations.

Input	foto1A, foto2A, foto3A, foto4A, foto1B, foto2B, foto3B and foto4B
Output of SIFT	foto1ASIFT_with_matches.jpg, foto1ASIFTPerspective.jpg. foto2ASIFT_with_matches.jpg, foto2ASIFTPerspective.jpg. foto3ASIFT_with_matches.jpg, foto3ASIFTPerspective.jpg. foto4ASIFT_with_matches.jpg, foto4ASIFTPerspective.jpg
Output of SURF	foto1ASURF_with_matches.jpg, foto1ASURFPerspective.jpg. foto2ASURF_with_matches.jpg, foto2ASURFPerspective.jpg. foto3ASURF_with_matches.jpg, foto3ASURFPerspective.jpg. foto4ASURF_with_matches.jpg, foto4ASURFPerspective.jpg

Figure 1: From left to right: foto1A, foto2A, foto3A and foto4A.

3 Code and Literature Review

The program deploys SIFT and SURF feature detectors alongside with other methods described below to matches key points between pair of images integrating them into one image with lines between these points and produces perspective images.

3.1 Scale Invariant Feature Transform

The Scale Invariant Feature Transform (SIFT) is a feature detector that detects key points invariant to scale and rotation. To achieve that, the algorithm applies an approximation of LoG (Lagrangian of Gaussian) scale-space filter with various σ values to detect different sizes of key points. With LoG, it is possible to find local maximum across the scale and space producing a list of (x, y, σ) meaning that there is a potencial key point in (x, y) with σ value.

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \text{ and } \begin{bmatrix} -1 & -1 & -1 \\ -1 & 4 & -1 \\ -1 & -1 & -1 \end{bmatrix},$$

used together with a Gaussian filter centered on $(0, 0)$ with a standard deviation σ, the LoG function has the form:

$$LoG(x, y) = -\frac{1}{\pi\sigma^4}(1 - \frac{x^2 + y^2}{2\sigma^2})e^{-\frac{x^2+y^2}{2\sigma^2}}.$$

The key points between two images are matched through a distant measurement. The distance could be Euclidean, Manhattan, Chebyshev or any other. A match happens when a key point of an image finds the closest key point of another. However, due to noises and other reasons, this method becomes more accurate if ratio of closest-distance to second-closest distance is taken.

The program uses Euclidean distance via the function $knnMatch()$ to find matches and ratio of closest-distance to second-closest distance. It applies $drawMatchesKnn()$ to connect key points with straight lines between two images and executes $findHomography()$ followed by $warpPerspective()$ to produce image perspectives.

The results of SIFT, $knnMatcher()$ with $k = 2$ and $ratio = 0.7$ are shown in Appendix section.

3.2 Speed Up Robust Feature

The SURF (Speed Up Robust Feature) is an improvement over SIFT which is also a patent local feature detector and descriptor. To detect key points it applies an integer approximation of the determinant of Hessian blob detector. It was first presented by Herbert Bay, et al., at the 2006 European Conference on Computer Vision and patented in United States.

SURF algorithm is composed of three main parts.

1. It deploys an approximation of Gaussian filter to blur the images. This approximation uses a kind of square-shaped filters that, according to the literature, run much faster then Guassian one. SURF detects blobs via Hessian matrix. The determinant of this matrix are calculated to measure local changes around the point and points are chosen where this determinant is maximal. So, given a point $p = (x, y)$ in an image I, the Hessian matrix $H(p, \sigma)$ at point p and scale σ is:

$$H(p, \sigma) = \begin{bmatrix} L_{xx}(p, \sigma) & L_{xy}(p, \sigma) \\ L_{yx}(p, \sigma) & L_{yy}(p, \sigma) \end{bmatrix}$$

   in which $L_{xx}(p, \sigma)$ is a convolution of the second-order derivative of guassian with an image I at point p

2. SURF finds key points with different scales by utilizing box filters of different sizes. The scale space is analyzed by increasing the filter size.

3. SURF describes image by fixing a reproducible orientation based on information from a circular region around the interest point. Then it constructs a square region aligned to the selected orientation, and extract the descriptor from it.

.

The results of SURF, $knnMatcher()$ with $k = 2$ and $ratio = 0.7$ are shown in Appendix section.

4 Comments

In this fourth assignment, I had to take a look at SIFT and SURF and other things embedded in them. Most of the information written in this report was taken from Wikipedia and Opencv website as well as partially the code to apply all functions required to do the homework. I had to downgrade opencv because the nearest one were blocking SIFT and SURF due to patents issues.

5 Appendix

5.1 Appendix A.1

Integrated images with matches resulting from SIFT algorithm.

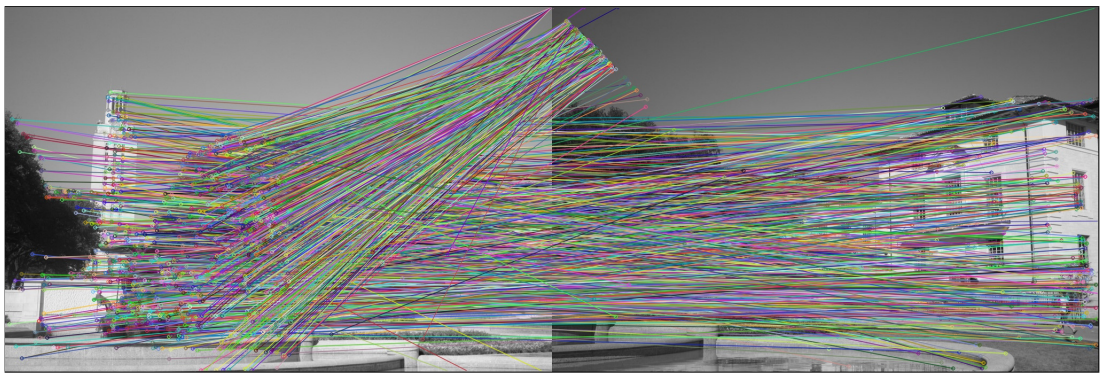

Figure 2: Pair of images, foto1A and foto1B, showing their matched key points.

Figure 3: Pair of images,foto2A and foto2B, showing their matched key points.

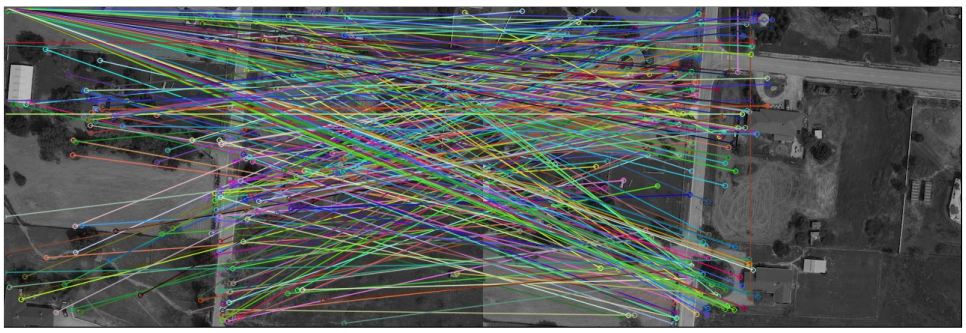

Figure 4: Pair of images, foto3A and foto3B, showing their matched key points.

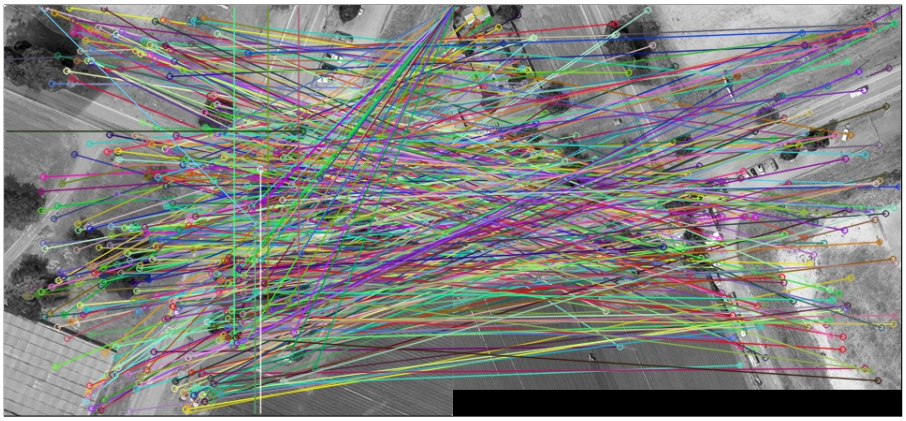

Figure 5: Pair of images, foto4A and foto4B, showing their matched key points.

5.2 Appendix A.2

Perspective images resulting from SIFT algorithm.

Figure 6: Perspective image of the pair foto1A and foto1B.

Figure 7: Perspective image of the pair foto2A and foto2B.

Figure 8: Perspective image of the pair foto3A and foto3B.

Figure 9: Perspective image of the pair foto4A and foto4B.

5.3 Appendix B.1

Integrated images with matches resulting from SURF algorithm.

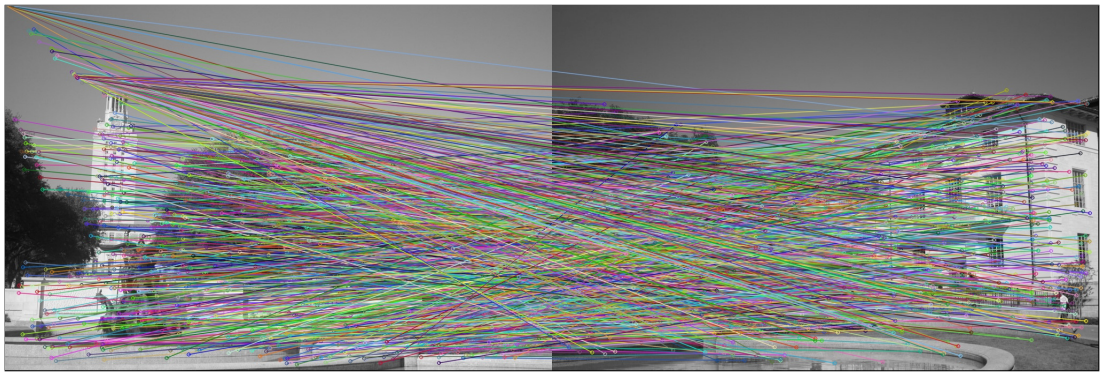

Figure 10: Pair of images, foto1A and foto1B, showing their matched key points.

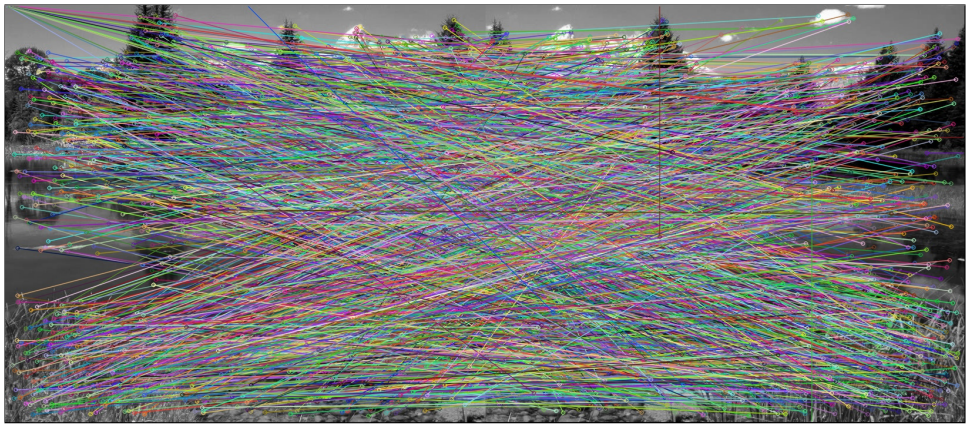

Figure 11: Pair of images,foto2A and foto2B, showing their matched key points.

Figure 12: Pair of images, foto3A and foto3B, showing their matched key points.

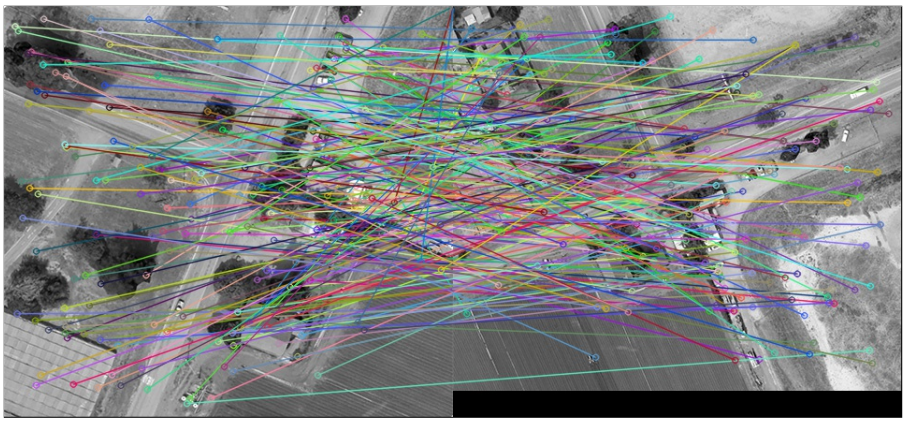

Figure 13: Pair of images, foto4A and foto4B, showing their matched key points.

5.4 Appendix B.2

Perspective images resulting from SURF algorithm.

Figure 14: Perspective image of the pair foto1A and foto1B.

Figure 15: Perspective image of the pair foto2A and foto2B.

Figure 16: Perspective image of the pair foto3A and foto3B.

Figure 17: Perspective image of the pair foto4A and foto4B.