ECE6360
Computer Vision and Image Reconstruction
Project3
Sharanya Ramakrishna
Tsung-Sheng Huang

Overview

In this project, we are going to implement the SFIT, SURF and MSER to detect the key points in the picture, then we use the key points to estimate the geometric transform between the images. Finally, we construct the panorama.

Scale-Invariant Feature Transform

First, what is detector and descriptor? Detector is used to detect the key points of the feature and the descriptor will generate the features. For SIFT, the descriptor will generate a 128-dimention feature for each key point. For SURF, it will have 64-dimention feature.

The implementation of SIFT:

```
[~, features, points] = sift(image);
```

The function returns the features (generated from descriptors) and the points (key points generated from detectors).

The following is the step of detecting the key points in the image.

1. Apply the LoG (Laplacian of Gaussian) to the image with different scales (sigma)
2. Subsample the image e.g. 128X128 to 64X64
3. Locate the key points

In the first step, we are calculating the LoG of the image which can be achieved by applying the DoG (Difference of Gaussian) to speed up the calculation. The example is shown in Figure 1. In Figure 1, we can see we apply the Gaussian filter to the image using different scale, then subtract them to obtain the DoG. After this, we subsample the image then apply step one again. The first octave means the image without sampling, the second octave will be the image after one subsampling and so on.

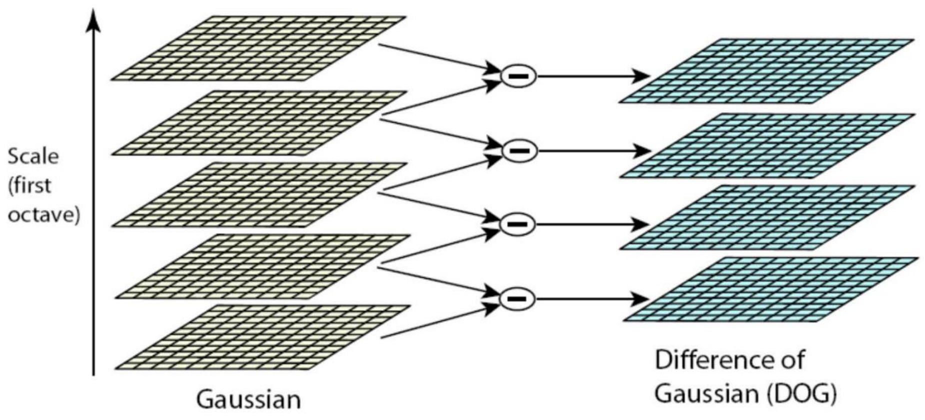


Figure 1 Example of DoG. [1]

After applying DoG on different octaves, next step is to locate the key points. Let's say we have a point called X, and we compare this point with 26 pixels in current and adjacent scales in a 3X3 neighborhood. This point will be selected as a key point if X is smaller of bigger than all of the 26 pixels. The example is shown in Figure 2.

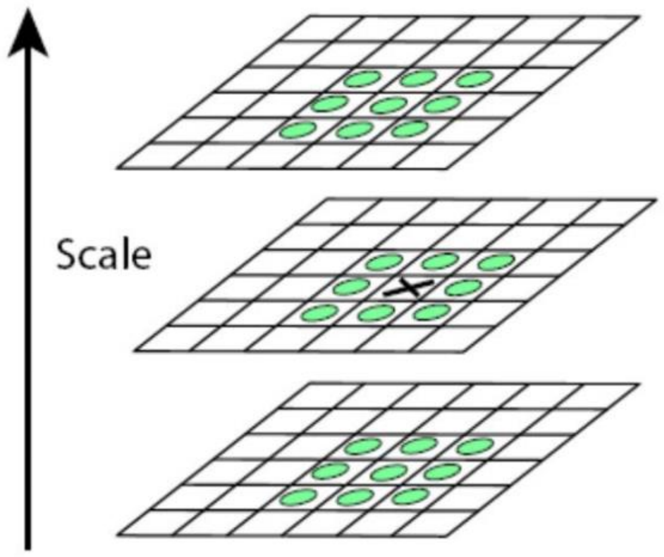


Figure 2 Example of locating the key points [1]

After the detection of the key points, we are going to describe the features of each key point. In SIFT, it uses the HoG (Histogram of Gradient), so it calculates the gradient of and the magnitude of the gradients of all the points around the key point in a 16X16 neighborhood. Then it divides the 16X16 kernel into 4 8X8 blocks. Each 8X8 block will generate 4 HoG (8 bins). And it concatenates all the HoG as a long 128-dimension vector (8*4*4=128). A result of SIFT is shown in Figure 3.

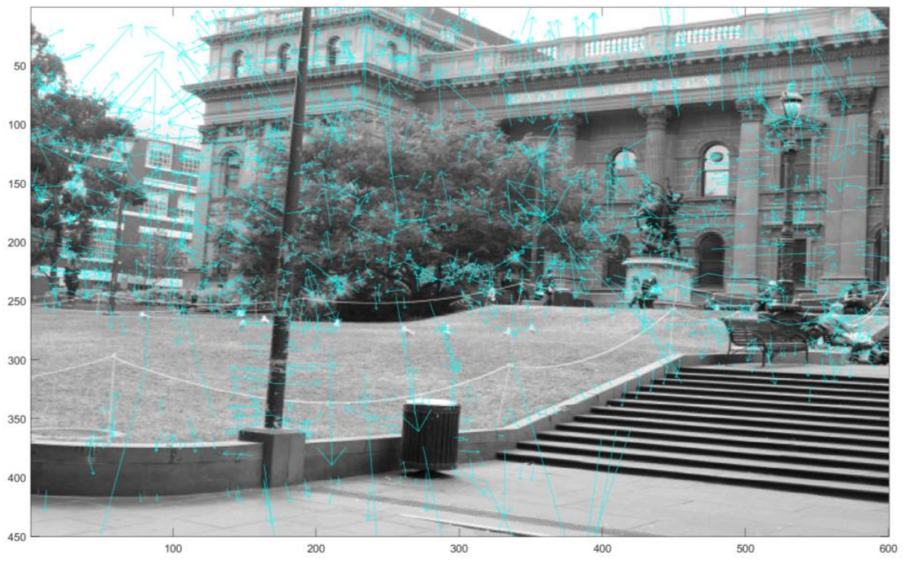


Figure 3 Result of SIFT

PCA-Scale Invariant Feature Transform (PCA-SIFT)

The detector of PCA-SIFT and SIFT is the same. The difference between PCA-SIFT and SIFT is their descriptor. The descriptor of SIFT uses HoG, but PCA-SIFT doesn't. In PCA-SIFT, it extracts a 41X41 patch at the given scale, which is centered over the key point. Then rotated the patch to align the dominant gradient generated by the detector.

PCA-SIFT contains following steps: (1) pre-compute an eigenspace to express the gradient images of local patches (41X41 patch). (2) Compute its local gradient. (3) Project the gradient image vector using the eigenspace to derive a compact feature vector. [2]

Speeded Up Robust Features (SURF)

In the detection step, instead of using the DoG, SURF uses the box filter to approximate the Gaussian second order derivative filter. Also, instead of subsampling the image, it just increases the size of the box filter to convolve with the image. It also implements the integral image to speed up the algorithm.

For the descriptor, it calculates the wavelet response centered at the key points. Then the dominant orientation of the key point is estimated by calculating the sum of all responses within a sliding orientation window covering an angle of π /3 [3]. The horizontal and vertical responses are calculated to generate the gradient. The integral image is also used to calculate the wavelet response.

Maximally Stable External Regions:

MSER is a blob detection technique and it is affine invariant and intensity extrema-based method. It is based on the idea of taking regions that stay nearly the same in different thresholds. The algorithm to find MSER regions is as follows:

1. Detect extremal points (detect corner points using Harris detector)
2. Start from a local intensity extremum point.
3. Go in every direction until the point of extremum of a function f and connected points form boundaries for each region.
4. Compute geometric transforms upto second order (i.e covariance matrix for 2D image) which gives the ellipses.
5. Finally replace the region with the ellipse.
6. All points create some irregularly shaped region and have corresponding regions in their affine transforms.

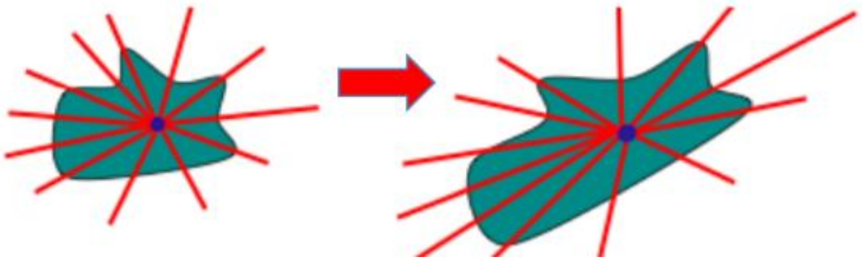


Figure 4 Corresponding points in affine transform

7. A margin is defined for the number of thresholds for which the region is stable.

MSER processing is as follows :

1. All pixels below a given threshold are white and all those above are black.
2. If we are given an image with a particular threshold, the image will initially be all black after which white spots will begin to appear at local axtrima points and will slowly merge.
3. The set of all connected components in the image are local extrimums.

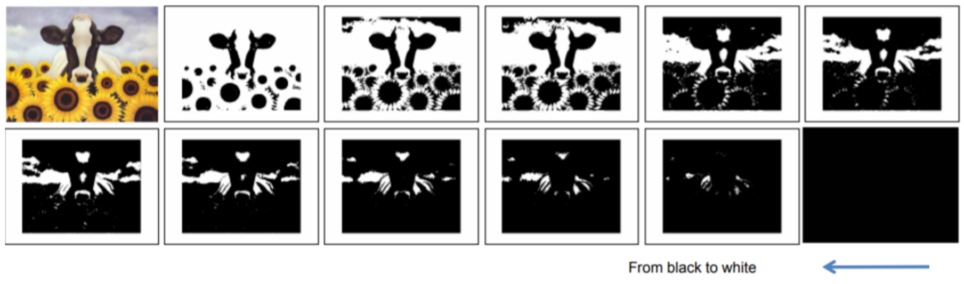
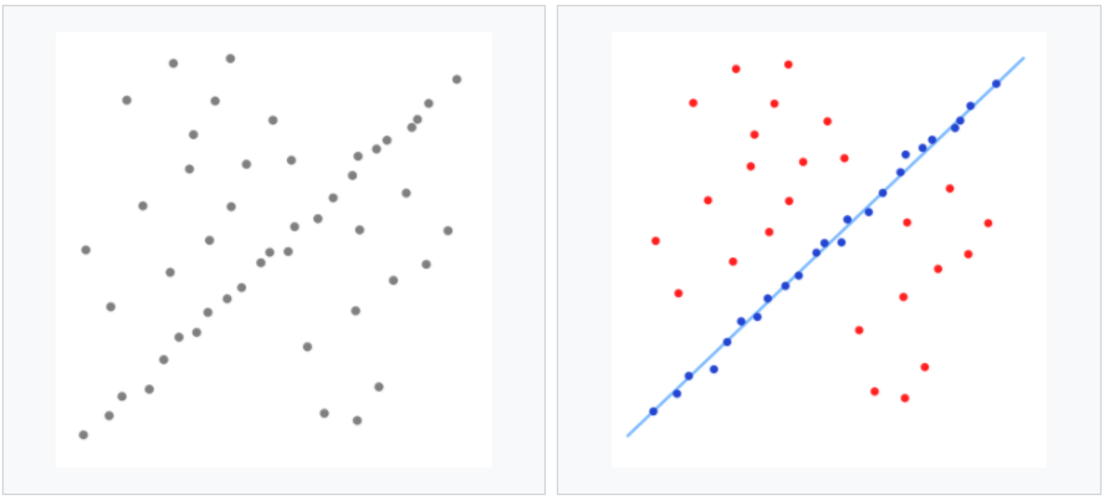


Figure 5 : Region detection in steps

Random Sample Consensus (RANSAC)

RANSAC is a method used when we need to find parameters of a mathematical model from a set of data where the outliers should have no influence on the estimation. Hence RANSAC is an outlier detection method.



A data set with many outliers for which a line has to be fitted.

Fitted line with RANSAC; outliers have no influence on the result.

Figure 6 RANSAC for outlier detection

Algorithm for RANSAC:

1. After finding matched points, If image1 matched points = u_0 , image2 matched points = u_1 , fit the data to $u_0^T F u_1 = 0$, to find the fundamental matrix F.
2. If all points in the two images are exactly matched, $u_0^T F u_1$ should be 0.
3. Check the residue for each point using $u_0^T F u_1 - 0$.
4. If a point has large residue, it is discarded as an outlier.
5.

Homography Matrix:

In computer vision any two images in the same planar surface in space are related by homography. The basic algorithm for a homography matrix is as follows:

1. Suppose we have some corresponding points in two images, we can create a matrix p such that,

$$p_i = \begin{bmatrix} -x_i & -y_i & -1 & 0 & 0 & 0 & x_i x_i' & y_i x_i' & x_i' \\ 0 & 0 & 0 & -x_i & -y_i & -1 & x_i y_i' & y_i y_i' & y_i' \end{bmatrix}$$

2. We can then stack them to calculate PH=0

   Ex, suppose we have 4 corresponding points between 2 images,

$$PH = \begin{bmatrix} -x_1 & -y_1 & -1 & 0 & 0 & 0 & x_1x_1' & y_1x_1' & x_1' \\ 0 & 0 & 0 & -x_1 & -y_1 & -1 & x_1y_1' & y_1y_1' & y_1' \\ -x_2 & -y_2 & -1 & 0 & 0 & 0 & x_2x_2' & y_2x_2' & x_2' \\ 0 & 0 & 0 & -x_2 & -y_2 & -1 & x_2y_2' & y_2y_2' & y_2' \\ -x_3 & -y_3 & -1 & 0 & 0 & 0 & x_3x_3' & y_3x_3' & x_3' \\ 0 & 0 & 0 & -x_3 & -y_3 & -1 & x_3y_3' & y_3y_3' & y_3' \\ -x_4 & -y_4 & -1 & 0 & 0 & 0 & x_4x_4' & y_4x_4' & x_4' \\ 0 & 0 & 0 & -x_4 & -y_4 & -1 & x_4y_4' & y_4y_4' & y_4' \end{bmatrix} \begin{bmatrix} h1 \\ h2 \\ h3 \\ h4 \\ h5 \\ h6 \\ h7 \\ h8 \\ h9 \end{bmatrix} = 0$$

3. We can then find the new co-ordinates in the transformed plane as

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Implementation

In Implementation part, we first use SIFT, SURF and MSER to find the key points in the image, then we use the function “extractFeatures” to find the features of each key points. Next, we use function “matchFeatures” to match the features between two images. After this step, we will have the index of the points of each method. Last, we can estimate the transform matrix between two images. In the function “estimateGeometricTransform”, it has MSAC (M-estimator Sample Consensus) algorithm to remove the outliers. However, we have to find the centered image in order to get a nicer panorama.

Implementation of SURF

Figure 7 shows the key points by using SURF. The size of the circle is related to the scale of Gaussian filter.

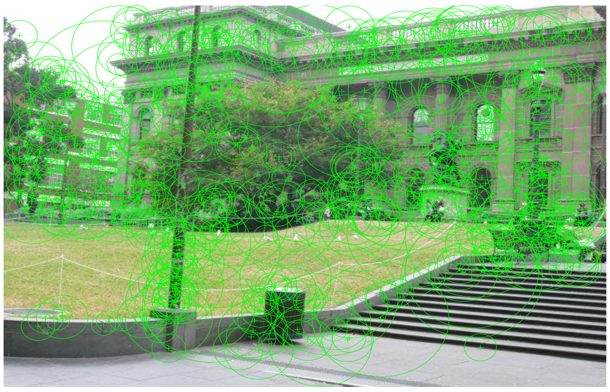


Figure 7 Key points detection using SURF

Second step, we find the key points in the second image and match it with image one. The result is shown in Figure 8.

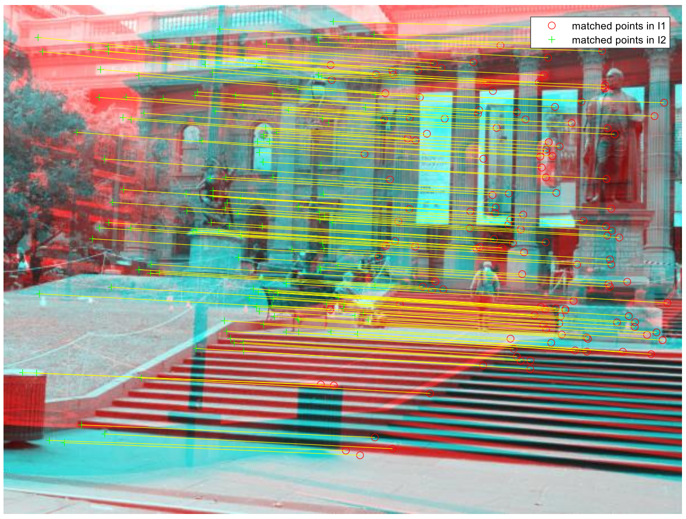


Figure 8 Result of Matched point

Finally, we can estimate the geometric transform matrix to generate our panorama. Figure 9 shows the result of panorama using SURF. The result might be different from each run.

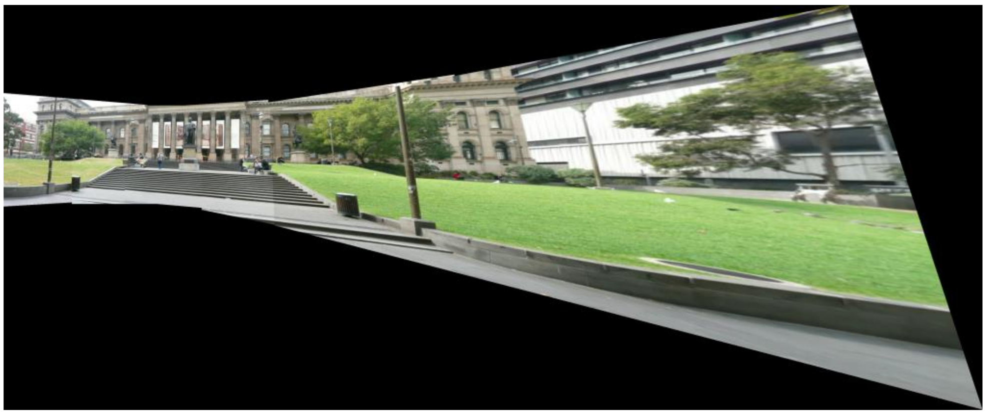


Figure 9 Panaroma using SURF

MSER implementation :

detectMSERfeatures outputs an object called “regions” which describes the regions that have the same second order moments as detected ellipses. The image shown below is a visualization of MSER regions described by pixel lists stored inside the returned regions object.

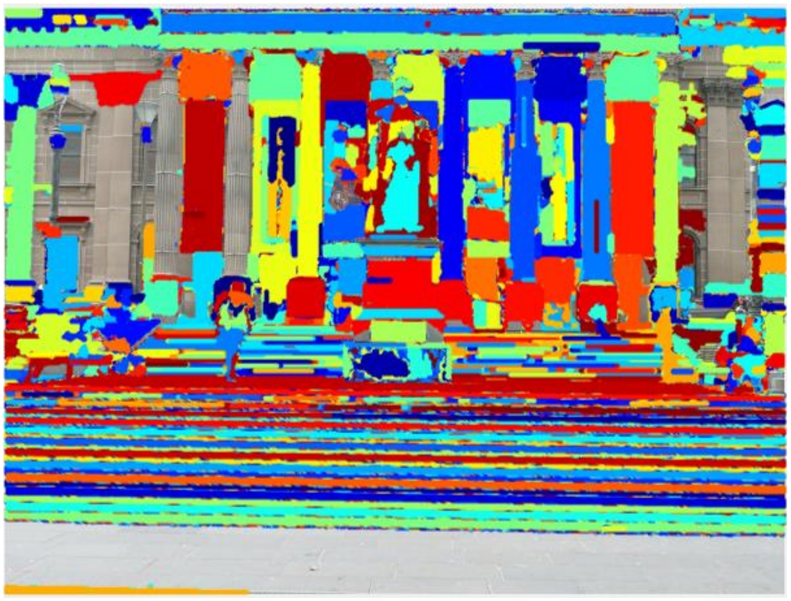


Figure 10 : region detection using pixel list

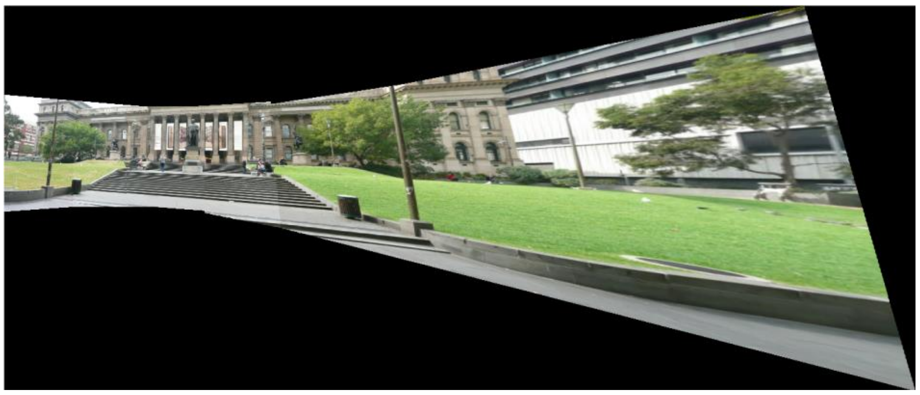


Figure 11 : Panaroma using MSER

Homography matrix implementation :

The panaroma obtained using homography matrix implementation is as below

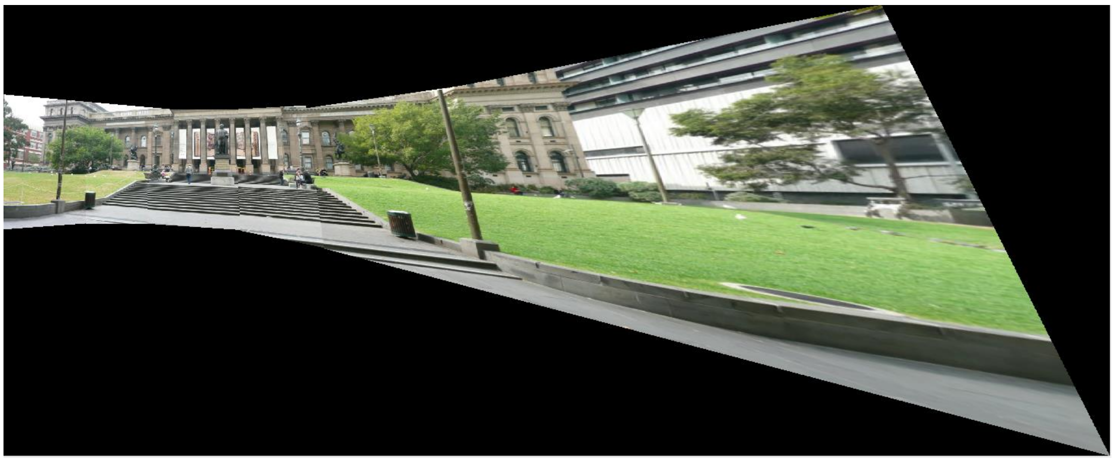


Figure 12 Panaroma using Homography matrix

Implementation of SIFT

The steps of generating the panorama using SIFT are basically the same as SURF, but the location (x,y) of key points using SIFT implementation needs to be swapped. The figures of key points and the matched points are shown below.

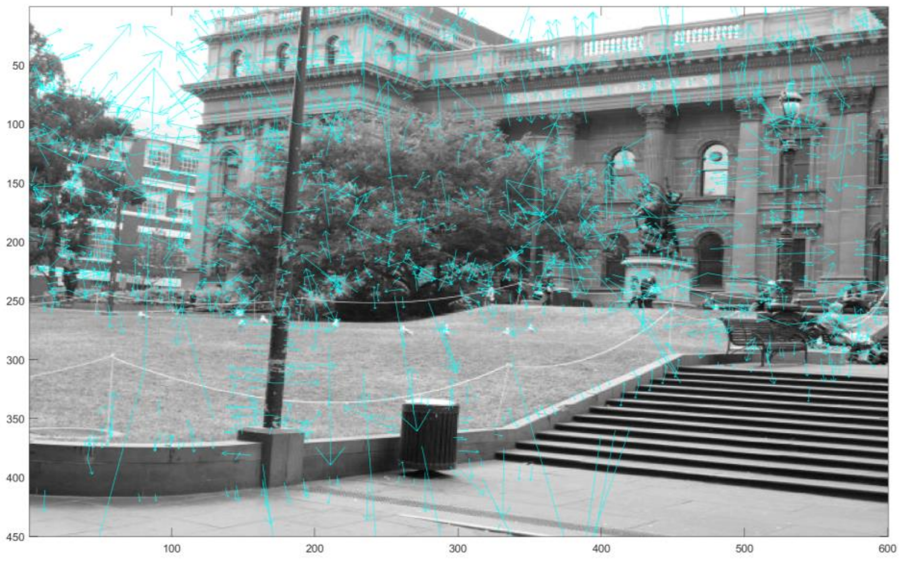


Figure 13. Key points generation using SIFT

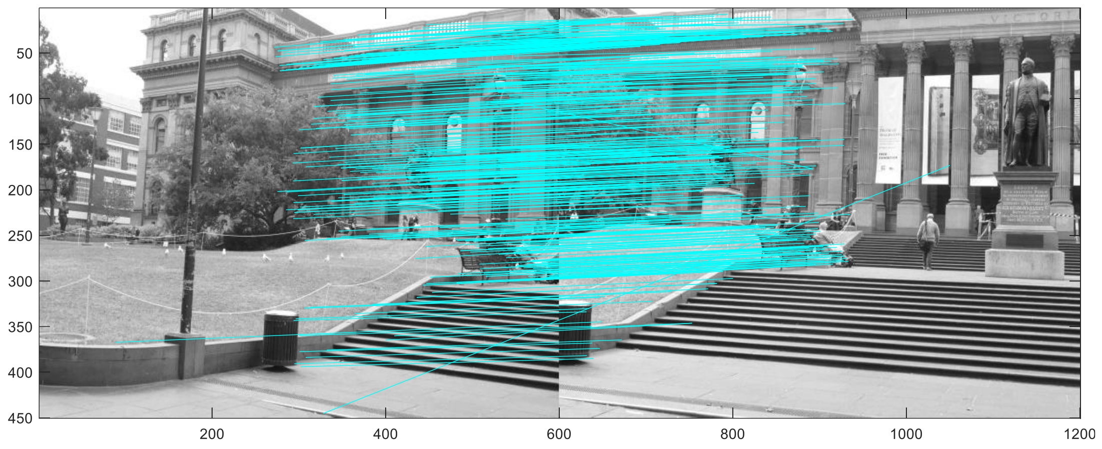


Figure 14. Matched points between two images

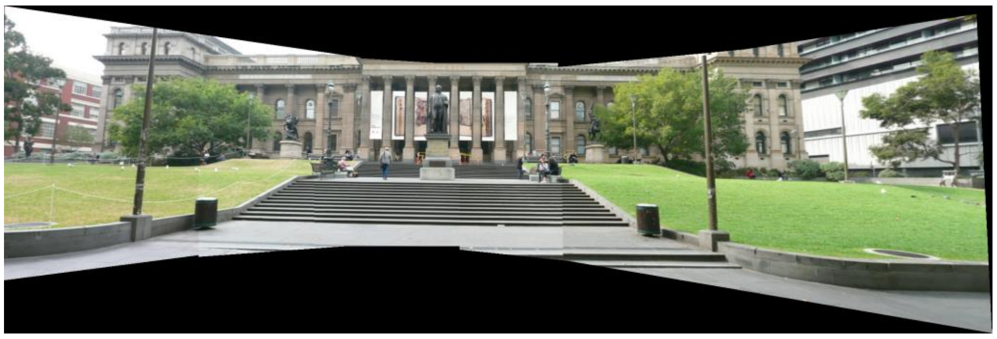


Figure 15. Panorama using SIFT

References

[1] http://crcv.ucf.edu/courses/CAP5415/Fall2012/Lecture-5-SIFT.pdf

[2] Ke, Yan, and Rahul Sukthankar. "PCA-SIFT: A more distinctive representation for local image descriptors." *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on. Vol. 2. IEEE, 2004.*

[3] Bay, Herbert, et al. "Speeded-up robust features (SURF)." *Computer vision and image understanding 110.3 (2008): 346-359.*

[4] https://www.mathworks.com/help/vision/examples/feature-based-panoramic-image-stitching.html

[5] https://www.youtube.com/watch?v=NKxXGsZdDp8&list=PL7v9EfkjLswLfjcI-qia-Z-e3ntl9l6vp&index=54

[6] https://stackoverflow.com/questions/1532168/what-are-the-second-moments-of-a-region

[7] http://www.micc.unifi.it/delbimbo/wp-content/uploads/2011/03/slide_corso/A34%20MSER.pdf

[8] https://www.mathworks.com/help/vision/ref/detectmserfeatures.html

[9] https://math.stackexchange.com/questions/494238/how-to-compute-homography-matrix-h-from-corresponding-points-2d-2d-planar-homog
[10] https://en.wikipedia.org/wiki/Homography_(computer_vision)