CS282 PA2 Report
Image Stitching

Marwin B. Alejo (2020-20221)

Date of Submission: Mar 15, 2022
Due Date: Mar 29, 2022

Figure 1: UT Tower Left

Figure 2: UT Tower Right

1. INTRODUCTION

This activity is among the activities of CS282 also known as Computer Vision with the aim of studying and implementing into code the Harris algorithm for feature detection in the context of Image Stitching. Image Stitching or Mosaicking is the process in Computer Vision that combines multiple images with overlapping fields of view to outcome a segmented panorama or high-resolution image.

The fundamental algorithm for image stitching may be summarized into four steps: (1) detection of image keypoints using Harris Detector or Gradient Difference, (2) extraction of local descriptors using Harris Detector or SIFT (Scale-Invariant Feature Transform), (3) homography computation from the matches feature arrays using RANSAC algorithm, and (4) application of warping transformation using the computed homography arrays to stith the images. Furthermore, this activity implemented this simple image stitching algorithm to stitch the overlapping images of the University of Texas - Tower as shown in figures 1 and 2 using Python programming language and OpenCV 4.5.5.64 on Google Colab. Also, to provide a comparative result, this paper show the output difference of using Harris Detector and SIFT for stitching images.

The following sections of these report is architected as follows: section 2 discusses the used theory of the activity and section 3 discusses the results of the activity.

2. THEORY

Features are an essential building block of Computer Vision for the task of feature detection and matching. Features are a piece of information about the content of an image as such properties. These properties are either be edges, points, or objects in an image which also determine the keypoints and descriptors of an image. The keypoint of an image contain the patch 2D position such as scale and orientation of an image. The keypoints of an image are also described by the appearance of patches of pixels surrounding the keypoint location hence, the descriptors. These keypoints may be estimated using several algorithms like SURF, FAST, BRIEF, and ORB. Moreover, this activity focuses on only discussing the most used feature detectors which are the (1) Harris Corner Detector and (2) SIFT.

2.1 Harris Corner Detector

Harris corner detector is a commonly used Computer Vision algorithm for extracting corners and inferring features of an image. It is the most used feature (corner) detection algorithm due to its accurate and efficient results in utilizing points of corners. Corners are points that have a dominant gradient in more than two directions which also serve as a connection between edges. These features allow the machine to determine whether the image features are invariant to translation, rotation, and illumination.

To further understand how Harris Corner Detector works, consider a grayscale image with an intensity I(x,y) and windowing function w(x,y). The (x,y) points state the position of the pixel with intensity I in the image which have the same context as with the windowing function w. If the pixel is enclosed in a moving window both the intensity I could be expressed as I(x+u, y+v) where u and v are the added values to the pointer position. Equation 1 as shown below allows the machine to search for the features like pointers in an entire image. Through Taylor series expansion, Equation 1 could also be expressed in a matrix form as shown in Equation 2. By simplification and applying the representation of pointers in a moving window function, Equation 2 could be simplified further as shown in Equation 3. Moreover, to determine a corner in an image through the Harris Corner Detection, Equation should be considered.

$$E(u,v) = \sum_{x,y}[I(x+u, y+v) - I(x,y)]^2 \tag{1}$$

$$E(u,v) \approx [u,v](\sum_{x,y} w(x,y)[\begin{bmatrix} I_x^2 & I_xI_y \\ I_xI_y & I_x^2 \end{bmatrix}])\begin{pmatrix} u \\ v \end{pmatrix} \tag{2}$$

$$E(u,v) \approx [u,v]M\begin{pmatrix} u \\ v \end{pmatrix} \tag{3}$$

$$R = det(M) - k(trace(M))^2 \tag{4}$$

By setting a threshold to R as used in Equation 4 might yield to several points in a small neighborhood considering that their values are above R. These R's are considered as corners. Furthermore, it is important to determine the point of local maxima from these cluster of points if it will be used for feature matching.

2.2 SIFT Feature Detector

Although the algorithm of Harris Corner Detection is efficient and accurate for detecting features from images regardless of their rotation orientation, it is still inefficient in terms of "scaling". Hence, SIFT was introduced. Once the keypoints are detected in SIFT, it tries to construct the descriptor for each and takes account for its neighborhood pixels. Among of the advantages of SIFT are: (1) distinctive to features, (2) quantity – it can predict smaller details, (3) efficient by computation, and (4) robust to occlusions on local features.

2.3 Feature Matching

Feature matching in this activity was performed using brute-force k-nearest neighbors. It is a simple feature matcher between two pair of images where it will take the descriptor of one feature in the first image and is matched with all other features of the second image by computing their geometric distances. The nearest or closest distance to the reference feature will be considered for the output. A good example of this is shown in figure 3.

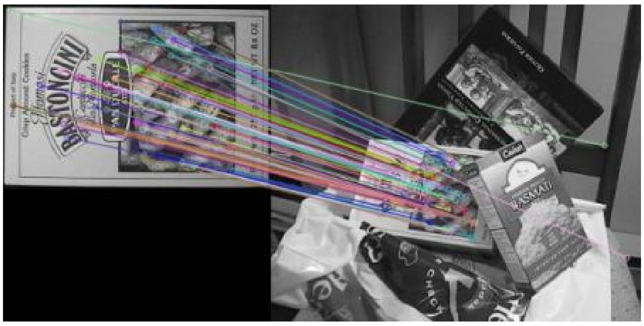

Figure 3: Feature Matching on boxes prints.

2.4 Perspective Transform

Perspective transformation allows the transformation of 3D objects into 2D spaces without altering its center projection. This is usually represented as a 3x3 matrix in a homogeneous coordinate.

2.5 Homography Estimation and Image Blending

Feature Matching alone will not eliminate outliers in the matched arrays hence, instances of having mistaken matches is possible. To resolve this issue, filtering of these wrong matches is necessary the use of RANSAC or Random Sample Consensus is necessary to separate points between inliers and outliers. The output of RANSAC is homography matrix of the images which will be placed on the canvas one-by-one, from left to right.

3. RESULTS AND DISCUSSION

This activity implemented the above-discussed theory into code using Python programming language and OpenCV with other necessary libraries on Google Colab. The following figures show these outputs leading to the stitching of the left and right images of UT Tower. Figures 4-7 show the results of detecting features (corner) using the Harris Corner Detection algorithm with while figure 8 shows the stitched image. Figure 9 on the other hand show the match features of the two images using SIFT with figure 10 showing the stitched image using SIFT as well.

4. GENERALIZATION

Overall, this activity studied the Harris Corner Detection algorithm and optionally the SIFT on the context of Image Stitching. The two methods differ only with the process of feature detection and extraction while the way they compute the arrays for stitching appears to be the same as evident with the result. Their only notable difference is that the Harris Corner Detection algorithm is rotation-oriented while SIFT is both rotation and scale-oriented.

Figure 4: Detected Features on UT Tower Left image.

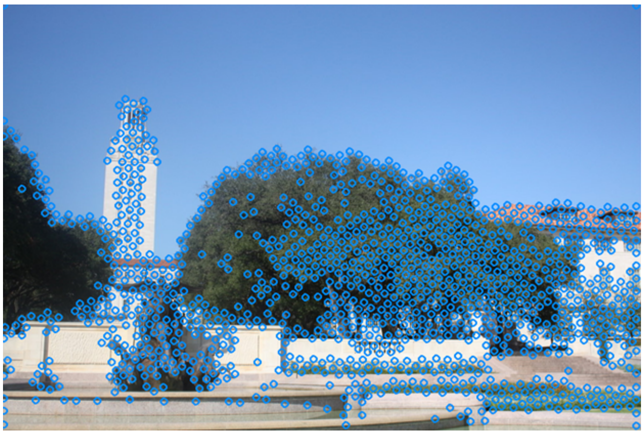

Figure 5: Detected Features on UT Tower Left image.

Figure 6: Detected Features on UT Tower Right image.

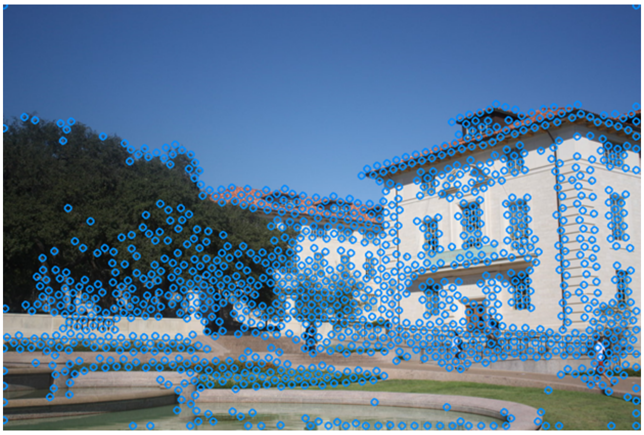

Figure 7: Detected Features on UT Tower Right image.

Figure 8: Stitched Image using Harris Corner Detector.

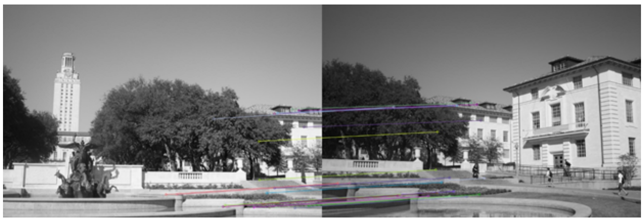

Figure 9: Detected and Matched Features on UT Tower Left and Right images using SIFT.

Figure 10: Stitched Image using SIFT.