Image Stitching Using Homography Techniques

Feature Extraction and Photometric Correction Methods Overview

Dmitry Demidov
dmitry.demidov@mbzuai.ac.ae

Muhammad Uzair Khattak
uzair.khattak@mbzuai.ac.ae

Umaima Rahman
umaima.rahman@mbzuai.ac.ae

Abstract

Image stitching is used in a wide range of Computer Vision applications, such as high-resolution photo-mosaics in digital maps, satellite imagery, video stitching, object insertion and image stabilisation to name a few. In this work we describe the major components involved in stitching two images to obtain a panorama. We explore the important components such as keypoint detection, local invariant descriptors, feature matching, homography estimation and perspective warping. Additionally, we present our implementation details where we explain our approaches used to solve this problem and also provide extensive experimentation for each of the considered tasks. Moreover to tackle the problem of illumination difference in the two images we also incorporate different blending and photometric correction techniques as an effort to procure a seamless stitching.

Contribution: We divided our work in three different parts. The first part involved Feature detection and matching, the second one included the part of estimating the homographies and image stitching whereas the last one included the photometric correction and image blending techniques to solve the illumination challenges. Muhammad Uzair Khattak worked on the first part, Umaima Rahman on the second whereas Dmitry Demidov worked on the third.

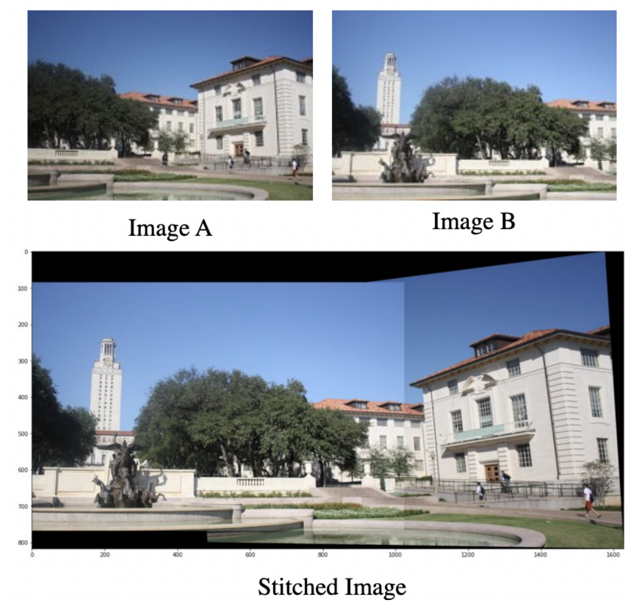

Figure 1. Image A (left) and Image B (right) to be stitched together

1. Introduction

Algorithms for aligning photos and stitching them together to create flawless photo-mosaics, often known as panoramas, are among the most commonly used in computer vision. Image stitching algorithms have improved throughout time, and today's digital maps and satellite photos are created using high-resolution photo-mosaics.

Image stitching using homography is a well researched area. [1] The generalised algorithm pipeline starts with the detection of common key-features between overlapping images, which provide point mappings between them. Using the point mapping correspondences, homography matrix is calculated using which, one image is fully mapped onto the second image, thus creating a mosaic or stitched version of two images. By extending the same idea between several images in a loop, complete panoramic mosaics are obtained. Finally, image blending and photo-metric corrections are applied which removes the seam-line effects in the overlapping part of images.

In our work, we study and implement the modules that are part of the image stitching algorithm. Firstly, we compare several methods for extracting and matching characteristics across image pairs of having overlapping scenes between them. Then we go over how to use RANSAC to match the photos and estimate homography matrix along with probabilistic models to verify the results. The rest of the study is devoted to aligning, stitching the matching photos together and carrying our experiments to overcome the discontinuous illumination effects using image blending and photo-metric corrections.

2. Background

2.1. Keypoints and feature descriptors detection

Keypoints or features are spatial locations in the image which are unique in nature and exhitbits several key properties such as repeatability, invaraince to photo-metric transformations, rotation and scale changes. These features are used as many important modules in various computer vision tasks, such as for image matching (finding corresponding point mappings between images) and image recognition. A feature can be anything in the image, which helps in recognizing the image e.g corners, edges, etc. Fig. 2 shows an image and its corresponding corner features mapped on it.

Similarly, using these keypoints, its corresponding descriptors are obtained that are the vector representation of these keypoints. These descriptors encode the keypoint information and embed it with other properties which make it more robust. In our work, we experiment with such feature descriptors and use them in our image stitching algorithm for finding the point correspondences between two images.

In the literature, various kinds of advanced feature detectors and descriptors have been proposed over the years which tends to be more invariant and robust to different types of transformations compared to others. We investigate following important feature detectors and descriptors:

The Harris corner detector [2] is a simple and efficient corner and feature detector. It finds corners as an intersection along the edges and analyses the intensity from multiple directions. The main limitation of Harris corners are that they are not scale invariant, which means if the scale of image is varied, the same corners will not be detected.

Scale Invariant Feature Transform (SIFT) [3], proposes features and descriptors which are scale invariant and thus are better than Harris corner features. SIFT uses different scales of an image and find keypoint extremes in each scale after filtering it with Gaussian filters, which helps to make it scale invariant.

Speeded-Up Robust Features (SURF) Although SIFT features are robust to nearly every kind of transformations, they require more compute and its calculation is slow. SURF [4] approximates the Gaussian filtering process in SIFT using integral images, which are faster.

Features from Accelerated Segment Test (FAST) [5] proposes image features which are suitable for real-time processing. They use features from image without any filtering or multi-scale approach, rather feature candidates are compared within its neighbour.

Scale-Invariant Center-Surround Detectors (CenSurE) [6] proposed robust scale invariant features, which uses all scales for selecting the extremas. Their method enables to detect SIFT like features in real-time. This is achieved by using simplified bi-kernels as center-surround filters.

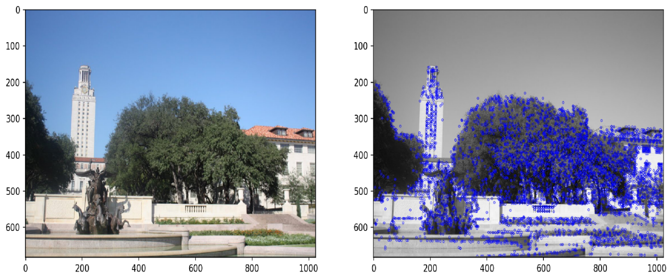

Figure 2. An image (left) and its corresponding features mapped on top of it (right)

We also explore a variant of CenSurE called STAR keypoints which uses only overlapping square filters to approach the center-surround filters. This selection further reduces the compute complexity.

Binary Robust Independent Elementary Features (BRIEF) [7] is a feature descriptor which focuses on reducing the memory requirement for storing the features. It converts the descriptors from floating point representation into binary settings, thus effectively reducing the memory requirements.

Oriented FAST and Rotated BRIEF (ORB) [8] uses FAST features along with BRIEF descriptors. ORB was proposed to be alternative to SIFT and SURF.

For our image stitching algorithm, we experiment and compare some of these feature detectors and show their respective capabilities.

2.2. Feature descriptor matching

Feature matching is the method to compare similarity of feature descriptors of one image with another. In the image stitching pipeline, once features for both images are obtained, they are matched together and their best correspondence points are calculated.

There are different methods to assign matches among feature descriptors of two images. One popular approach is a naive feature matching technique called brute force (BF) matcher, in which every feature descriptor from first image is matched with every feature descriptor of second image. The matching is performed by calculating the distance of descriptor vectors for the two images. However, there may be numerous closely-matching features on the local scale, resulting in false positives. To resolve this, we incorporate the test ratio proposed by Lowe [3] to remove the bad matches. This test ratio is the distance ratio between the first and second-best matches (of one image) for a given keypoint descriptor (of the second image). The mathematical expression for this ratio test is given as:

$$\text{if } \frac{\text{First match distance}}{\text{Second match distance}} < 0.75 \rightarrow \text{Goodmatch}$$

Only those matches are retained which fulfil the above

Figure 3. Visualization of the results (keypoints) obtained from different Keypoint detectors

Lowe's Test criteria. We explore these techniques in the image stitching algorithm which helps to find the best feature descriptor matches between the two images. Subsequently, these correspondences are used to find homography, which we explain in the next section.

2.3. Homography Estimation

Homography is a transformation between two image planes. It maps the points of an image in one plane to the corresponding points in the image of another plane. For a an image point P, the 3x3 transformation matrix H transforms this point P to P' using P' = HP where H is represented in the homogeneous coordinates system as follows:

$$H = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix}$$

Let us assume P = $\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$ and P' = $\begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$ in homogeneous coordinates. Then we can write

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

By eliminating c, we can formulate the above equation in the form $Ah = 0$, where,

$$A = \begin{bmatrix} \text{-x} & -y & -1 & 0 & 0 & 0 & ux & uy & u \\ 0 & 0 & 0 & -x & -y & -1 & vx & vy & v \end{bmatrix}$$

$$h = \begin{bmatrix} \text{h1} & h2 & h3 & h4 & h5 & h6 & h7 & h8 & h9 \end{bmatrix}^T$$

Knowing A, we can find the values of h by solving the above equation. [9]. It is to be noted that we need at least four such point correspondences in order to estimate the homography matrix.

For a given pair of images we find the homography matrix and the inlier features between them. However, in real life scenarios there are outlier matches between the two images. Therefore, using all the keypoints for homography estimation may yield undesirable results. As a result, a robust method RANSAC (random sample consensus) only uses a subset of inliers in the data for estimating the model parameters. This algorithm can handle practically any ratio of outliers given a threshold to distinguish inliers from outliers. [1] This threshold can vary from 1.0 to 10.0 based upon the resolution of the image. In this project, we use OpenCV's builtin cv2.findHomography function to get the homography and inlier candidates.

2.4. Image stitching

The process of image stitching involves combining multiple images that have an overlapping field of view and results in what is called a segmented panorama. Image stitching comprises of two important steps of image registration and blending. Image registration is the process of aligning images in a way that minimises the sum of the absolute differences between the overlapping pixels of two images. Once the alignment is done the two images are blended in a way that blurs the line of separation and makes the stitch seamless. In this stage we use the results of the feature matching to find overlapping features of the two images. We try to stitch the two images only if enough feature matches were found between them.

2.5. Photmetric Correction

When different images are stitched together, the stitched pixels sometimes produce artefacts. This may happen due to various reasons, such as changed lighting conditions, varying light exposure levels, vignette effects, and viewpoint angles etc. The human vision has the ability to perceive objects as the same colour from both eyes even under changing illumination. A similar behaviour can be achieved for stitched images if they have different levels of illumination. These image processing techniques adjusting and balancing brightness disparities of photographs, often called photometric correction, are considered in this section.

2.5.1 Image Blending Algorithm

Image blending (or center weighting) is a mathematically simple approach assuming that the pixel values in the blended regions are weighted average of two images in the overlapping area.

The image blending basically can be considered as simply weighted images addition, where different weights are given to images so that it controls the amount of blending or transparency. Mathematically it is done by this equation:

$$g(x) = (1 - \alpha) * img_1 + \alpha * img_2.$$

By varying α from 0 to 1, we can adjust the transition between one image to another.

2.5.2 White Patch Algorithm

White patch is a white balancing technique that assumes that the maximum response in an image is caused by perfect reflectance (a hypothetical white patch) [10]. In practice, each image channel is normalised by its respective maximum, $R_{max}, G_{max} \& B_{max}$, and then these channels are multiplied by 255. By doing this, we ensure that the maximum pixel value in each channel is a constant. This reduces the impact of illumination or light source in perceived colour of the image. Mathematically, a white-balanced image can be calculated by the following expression:

$$\begin{bmatrix} R' \\ G' \\ B' \end{bmatrix} = \begin{bmatrix} a/R_{max} & 0 & 0 \\ 0 & a/G_{max} & 0 \\ 0 & 0 & a/B_{max} \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

Therefore, the factors for each channel are as follows:

$$R_{factor} = a/R_{max}, G_{factor} = a/G_{max}, B_{factor} = a/B_{max}$$

where, $a = 255$. We may alternatively set $a = G_{max}$ for normalising to the green channel's maximum colour.

2.5.3 Histogram Equalisation Algorithm

Histogram equalisation (HE) is a statistical approach of image processing aiming to spread out intensity values of an image. It is used in order to improve the contrast of an image making dark portion darker and bright portion brighter.

One important limitation of the HE algorithm is that it can be only applied to one channel, which is useful for grey-scale images, where each pixel is represented by the intensity value (brightness). However this is not the case for the given input RGB images, where each channel represents the intensity of the related colour, and not the intensity (brightness) of a point in an image.

A solution is to first separate the brightness of the image from the colour and then run the HE algorithm on the former. For this task we considered such colour spaces that encode brightness and colour separately as YCbCr, YUV, and HSV. Next, when the colour space is changed, we can run HE on the specific channel that this channel represents brightness, and then convert the image back to the original RGB colour space. For HSV the intensity channel is V channel, for YCbCr and YUV is Y channel. The results for a sample image are shown in Fig. 9.

3. Implementation and Results

3.1. Keypoint features and descriptors

For detecting keypoints of the images, we use and compare several major keypoint detectors which includes SIFT, SURF, FAST and STAR (CenSurE) detectors.

Firstly, we select two images which are to be stitched. Such a pair is shown in Fig. 1. Both images are converted into grey scale before applying the feature detector method. Image pair should have at least 15-30% overlap [9] between them to enable extracting reasonable number of common keypoints (which will be later used for the matching). For each image, we independently detect the keypoints and its corresponding keypoint descriptors.

In the figure 3, keypoints of different feature detectors are visualized on its corresponding images.

We observe that the SIFT and SURF keypoints mapped on the image are very high in number, as compared to other types of keypoints. We can see that, most of the detected keypoints for SIFT are at corners, and shape outlines of different objects. The number of keypoints for the SURF detector are also high in number, but we can see there are many false positive detections as well, even though in the presence of non-maximum suppression (NMS). This response is likely to occur as in SURF, it approximates the Gaussian filtering process with box filters (via integral images).

On the other hand, the number of STAR keypoints are lesser in number. Its keypoints also tends to be on the borders of various objects in the image. For the FAST detector, we observe that its number of keypoints are less but they correspond to the strong and unique corners and features of the image.

The quantitative results of number of detected keypoints for each method is shown in figure 4. From the set of these detected keypoints, its corresponding vector descriptors are generated. SIFT and SURF descriptors are used for its respective SIFT and SURF keypoints. While for FAST and STAR keypoints, we use the BREIF descriptor method.

3.2. Feature matching

Once the set of keypoints and its descriptors for both images are obtained. We match features between the two images by adopting the method of brute-force (BF) matcher. More specifically, we use kth-nearest neighbour method in which for each descriptor (of first image), we keep its two nearest descriptors (from second image). For the similarity metric, we use L2 distance norm (euclidean distance)

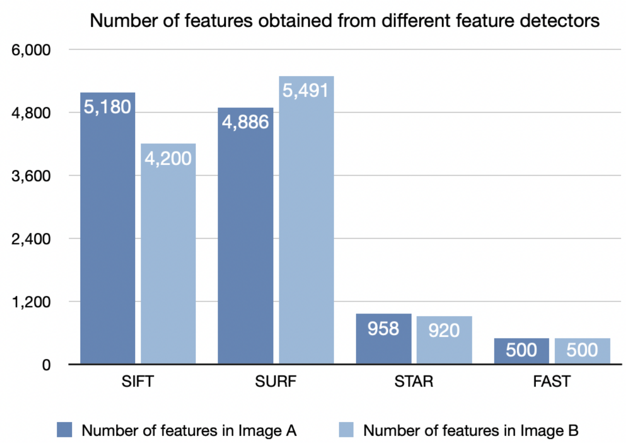

Figure 4. Number of features obtained from different feature detectors for both the images.

for matching of SIFT and SURF descriptors. For the STAR and FAST descriptors, we use the Hamming distance metric as its calculation is very fast which aids these real-time feature detectors.

To remove the false-positive matches between the two images, we incorporate the Lowe's test ratio on the raw initial matching pairs. The number of final pruned matched features depend upon the ratio of Lowe's test ratio and initial number of raw matches. Table 1 shows the comparison between number of matches before and after applying this ratio. We see that, for test ratio value of 0.75, the final processed matches decreases by a large extent. Thus we end up with only those matches which are true positives.

Finally, the pruned matches are obtained and those matches which do not pass the ratio test are ignored. Fig. 5 shows the visualisations of the matching between the features of both images for SIFT descriptors. We observe similar trends for other types of descriptors, which are shown in Fig. 10. It can be observed that most of the features are correctly matched between the two images.

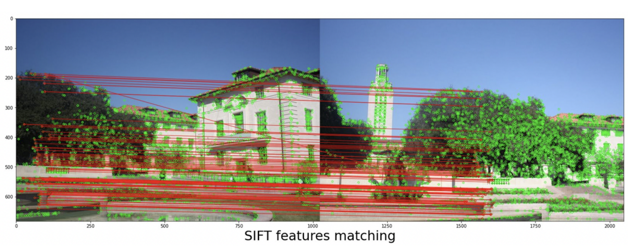

Figure 5. Feature matching of SIFT descriptors between the image pair for stitching. The red points and lines shows the **inlier** points which has passed the ratio-test, while the green features are those **outliers** which were not matched in the descriptor matching method

Detector Type	Number of matches	
	before ratio test	after the ratio test
SIFT	5180	931
SURF	4886	1054
STAR (BRIEF)	958	239
FAST (ORB)	500	105

Table 1. The difference in the number of matched features once we apply the Lowe's ratio test. The number of matches significantly reduces once the ratio test is applied. The results are written for the image pair shown in 1

Finally, the pruned matched features, which passes the ratio test are considered to be used for input to Homography matrix calculation.

3.3. Robust recoveries of homography

Once we find the location of features of one image in the other, it is sufficient to find the overlapping features between the two images. To obtain the homography matrix we use the same formulation as mentioned above in section 2.3. This method robustly estimates the homography that fits all corresponding points in the best possible way. The method used to calculate the homography matrix takes the keypoints of the two images along with the number of matches and the projection threshold as the input and finds the corresponding homography matrix. To make it robust to outliers and overcome errors while matching, we use the RANSAC algorithm. [11].

3.4. Stitching the images

Once we have obtained the homography matrix, the task is to transform the source image using the homography and add it to the destination image. For mapping of the source image, we use OpenCV's [12] warpPerspective function, which applies homography matrix on all pixels of the source image.

However, before using this function we need to translate the source and the destination images in a way that the entire image is visible. And hence we also need to translate the homography matrix. We pre-process the image to obtain the coordinates of the maximum (positive) and minimum (negative) pixel values which are outside the original image pixel location after the homography matrix is multiplied with it. Once we have these extreme points, we pad the original image before finding the warped image using the warpPerspective function of OpenCV which takes an image and the homography matrix as input. Then based on the homography matrix it warps the source image to the destination.

After warping the source image and combining the destination, the resultant stitched image is shown in Fig. 1.

The difference of using warpPerspective alone and using it with our customised function is highlighted in Fig. 6.

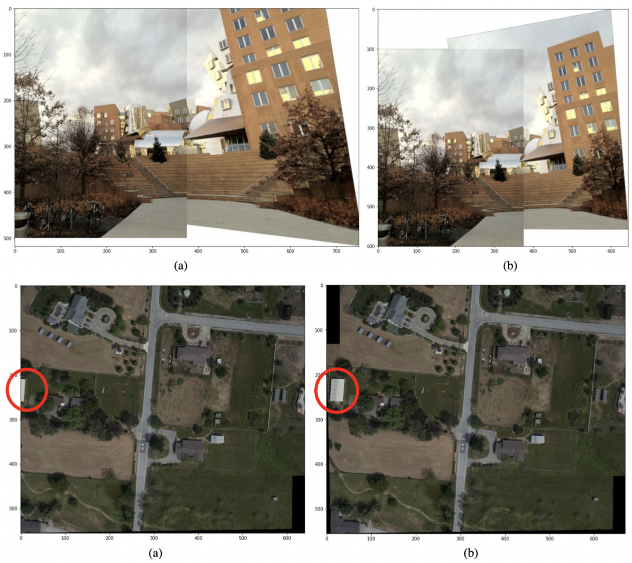

Figure 6. For two different pair of images, we show (a) image stitching without padding the images(b) image stitching after padding the images.

For the first pair of images the difference in the result is apparent, however, for the second pair of images we have highlighted the difference in red circles. Similarly, the result of the image stitching for all the other samples are presnet in 11.

3.5. Photometric Correction

3.5.1 Image Blending Algorithm

As can be seen in Fig. 7, the algorithm produces good results when all the images to be stitched were taken at the same daytime and from the same place. However, sometimes the quality of this approach suffers in the case when images are not fully overlapped, which causes unnatural and sharp colour gradients.

Figure 7. A side-effect of applying Image Blending Algorithm

3.5.2 White Patch Algorithm

We implemented the white patch algorithm by first extracting the three image channels, and then computing the max of each channels. We set a to 255, and used it to calculate the gain factors for each channel. All the max values where calculated using the max function from the NumPy library. We then multiplied the gain factors with the respective channels and stacked them back using the NumPy stack function. The results of this algorithm can be found in Fig. 8

Figure 8. White balancing using white patch algorithm.

3.5.3 Histogram Equalisation Algorithm

According to our experiments, Histogram Equalisation works well in some cases, however its results are highly image-dependent, and sometimes it produces unnatural colours and artifacts, due to ignoring outliers and the location of a pixel. The results for a sample image are shown in Fig. 9.

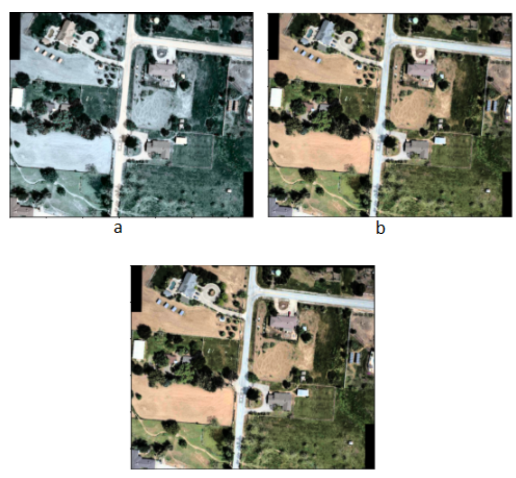

Figure 9. Results of applying Histogram Equalisation algorithm. (a) HE based on YCbCr, (b) HE based on YUV, (c) HE based on HSV

4. Conclusion

In this project we explored different types of feature detectors to obtain the features in the two images and find the matching pair of features between them. For image alignment and stitching we applied the homography matrix on the keypoints detected to obtain the warped image. The two images are then combined together after removing the intersecting parts between them. Our approach successfully captures the complete information present in the two images, when they are stitched. However, when there is a significant change in exposure, the color difference is visible. We believe this is due to the usage of mean brightness values to make color modifications prior to blending.

References

[1] Matthew Brown and David G Lowe. Automatic panoramic image stitching using invariant features. *International journal of computer vision*, 74(1):59–73, 2007.

[2] Christopher G. Harris and M. J. Stephens. A combined corner and edge detector. In *Alvey Vision Conference*, 1988.

[3] Tony Lindeberg. *Scale Invariant Feature Transform*, volume 7. 05 2012.

[4] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In Aleš Leonardis, Horst Bischof, and Axel Pinz, editors, *Computer Vision – ECCV 2006*, pages 404–417, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[5] Deepa Viswanathan. Features from accelerated segment test (fast). 2011.

[6] Motilal Agrawal, Kurt Konolige, and Morten Blas. Censure: Center surround extremas for realtime feature detection and matching. volume 5305, pages 102–115, 10 2008.

[7] Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua. Brief: Binary robust independent elementary features. In Kostas Daniilidis, Petros Maragos, and Nikos Paragios, editors, *Computer Vision – ECCV 2010*, pages 778–792, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[8] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: an efficient alternative to sift or surf. pages 2564–2571, 11 2011.

[9] Pratik Kinage. Image stitching.

[10] Alessandro Rizzi, Carlo Gatta, and Daniele Marini. Color correction between gray world and white patch. In *Human Vision and Electronic Imaging VII*, volume 4662, pages 367–375. International Society for Optics and Photonics, 2002.

[11] Jana Kosecka. Model Fitting, RANSAC. CV-702 lectures, 2022.

[12] Pratik Kinage. opencv perspective, 2021.

A. Appendix

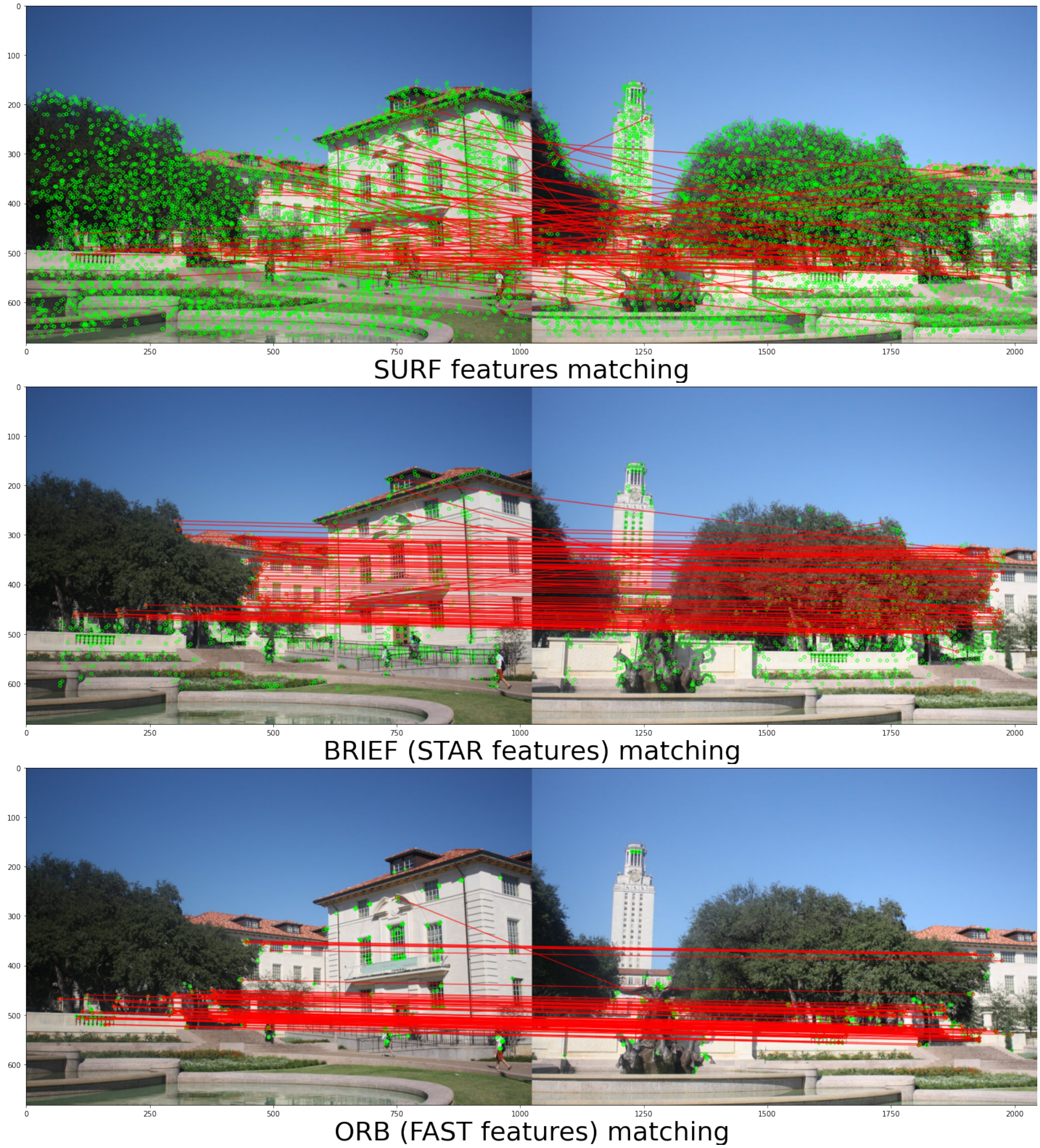

Figure 10. Visualisation of feature descriptor matching for various key point detectors.

Figure 11. Visualisation of original pair of images along with the stitched image can be seen for the four different images provided as samples.