

# AI 6121- Computer Vision Assignment 2

Done by: Liu Yao Hong	G2203319C Email:liuy0220@e.ntu.edu.sg
Patricia Njo	G2205032J Email:patricia003@e.ntu.edu.sg
Lam See Hwee	G2102084K Email :seehwee001 @e.ntu.edu.sg

## 1. Introduction

A panorama image is a large image which is naturally created by image stitching, where stitching multiple small images or mosaicing are performed(Wang, 2020).In this modern age, this image-stitching technique is widely adopted to produce satellite photos and digital maps.

However, the challenge of image stitching is the inability to know how each image will fit to create a flawless image, just like piecing the jigsaw puzzle together, where we need to find the correct features to piece together. Therefore, it uses mathematical concepts or tools like Homography to provide a better understanding of the transformation between each image with different perspectives and create composite images. Apart from having Homography process , our team has developed an algorithm that pieces together with a Harris Corner detector to detect and scale invariant feature transformation (SIFT) as a descriptor to match the feature points to the given images.

Random Sample Consensus (RANSAC) has been used as well for robust homography estimation. Wrapping and linear blending have been implemented for image alignment. This process was applied to the several images that were provided. Figure 1 illustrates the standard process of image stitching.

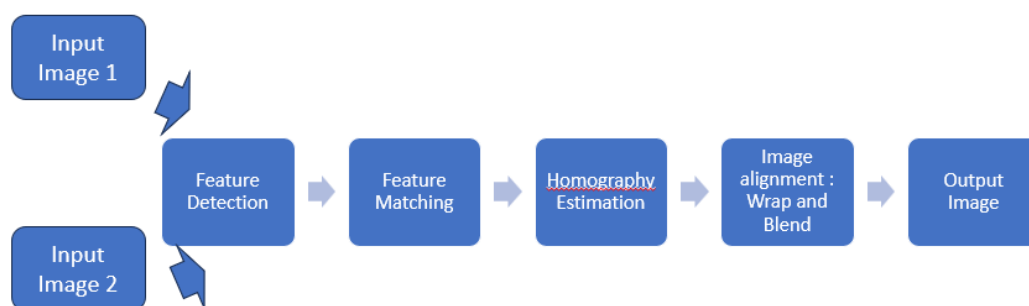


Figure 1 : Standard Processes of Image stitching

## 2. The Standard Processes of Image Stitching

Below is our team's design algorithm for stitching to create panorama images on the given 3 pairs of images.

### a. Feature Detection : Harris Corner

Under the feature detection, our team has adopted the Harris Corner Detection. Harris Corner Detection is a method of feature detection where the image's content is gathered based on specific features. Specifically, Harris Corner gathers the image's content based on the corners and edges[7].

In this context, corners are typically defined as the points in 2 common edge directions within the local region or the junction of 2 edges that can handle the change in image contrast[7]. They are classified as an important feature within the images because they are considered the interest points invariant to the brightness, transformation and rotation. Furthermore, they reduce the data processed in the computer vision, such as image stitching.

#### Formula of Harris Corner[8]

The function  $E(u, v)$  calculates the difference in intensity when displacing  $(u, v)$  in all directions.

$$E(u, v) = \sum_{x,y} \underbrace{w(x, y)}_{\text{window function}} \left[ \underbrace{I(\underline{x+u}, \underline{y+v})}_{\text{shifted intensity}} - \underbrace{I(x, y)}_{\text{intensity}} \right]^2$$

Applying the Taylor function, we can maximize the function  $E(u, v)$  for corner detection.

$$E(u, v) \approx \begin{bmatrix} u & v \end{bmatrix} M \begin{bmatrix} u \\ v \end{bmatrix} \quad (1)$$

Finally, the equation  $R$  determines whether or not an image contains corners.

$$R = \det(M) - k(\text{trace}(M))^2 \quad (2)$$

In our assignment, we have decided to implement Harris Corner and below are the steps.

#### Steps in Implementing Harris Corner in code

1. Convert the coloured image to grayscale
2. Detect the corners with Harris Corner. In our case, we used OpenCV's function `cv.cornerHarris()`
3. Threshold on the value of  $R$  (i.e.  $R > \text{threshold}$ ) to compute non-maximum suppression
4. Draw Harris Corner key points on the image pairs

## Harris Corner Detection on Image pairs 03\_01 and Image pairs 03\_02



Figure 1. Harris Corner Detection

Our team decided on using a threshold instead of an adaptive non-maximal suppression technique. There are a few reasons for using a threshold. Firstly, they are fast in computation. Secondly, they can be more sensitive to weak corners, and lastly, the robustness in handling noisy data. Furthermore, the threshold plays a role in measuring the substantial corners, while filtering out the false positives.

In implementing the Harris Corner Detection, our team used the commonly adopted threshold value of 0.01. A high threshold value detects strong corners only, while a low threshold value will include weak corners. We decided that the threshold value of 0.01 was neither too high nor too low for the image pairs we were working with.

Our team understands that, in principle, using a smaller threshold will result in more noise. However, it allows more weak corners to be detected, which is often missed by a higher threshold. Weak corners are considered valuable points which consistently provide subtle information that represents the whole scene, especially in our panorama image stitching process assignment task.

Based on Figure 2, the red points in the image are the corner points our team is interested in. Adopting the threshold value of 0.01 allows us to cater to different images without requiring us to perform individual fine-tuning to obtain optimal results. Therefore, our team has retained the threshold used in Harris Corner = 0.01 to build a general algorithm that caters to all image pairs.

### **b. Feature Matching : Scale Invariant Feature Transform(SIFT)**

The Scale Invariant Feature Transform is also known as SIFT. It is a popular feature descriptor introduced by David G. Lowe in 2004[1] for its robustness, flexibility and uniqueness. It boasts scale and rotation invariance, allowing it to detect and match features across varying scales and orientations adeptly. Furthermore, SIFT features are widely adopted in computer vision fields in image stitching, object recognition, 3D reconstruction, and pedestrian detection. As being described as a robust feature, SIFT can change the lighting conditions and exhibits resilience to noise and blur. They are also excellent in multi-scale analysis, where they can detect and describe features of different sizes.

Our team has taken into consideration a few pros in implementing SIFT into our methodology. Based on Karam et al., SIFT has outperformed both the SURF and ORB when dealing with distorted images, even though the computational cost memory requirements are higher than the other 2[2]. Based on the literature review of Karam et al., our team have decided to develop a more general algorithm that allows us to handle both the distorted and no distorted images after reviewing this literature on the effect of the images being output[2].

In the SIFT algorithm, the descriptor is its main feature; its roles are characterizing the local features detected by the detector, deriving the orientation and direction of the local patterns around each key point, and encapsulating this into the information vectors. Therefore, in our assignment context, once the features have been detected with our feature detection Harris corner detector, the SIFT descriptor is applied to generate a feature vector for each identified corner point.

Then, it will calculate the direction parameter and assign orientations to the feature points. The main concept is to gather the gradient directions and magnitudes in the proximity of each key point. From Figure 3, the most prominent orientation within that region is determined and then assigned to the key point.

All the subsequent computations are carried out relative to this orientation. This is to ensure the rotation invariance. The extent of the "orientation collection region" around the key point depends on its scale. This means that with larger scales, it corresponds to a more significant collection of regions.

In addition, the formulas 3 and 4 as illustration shown is used to compute the Gradient magnitudes and orientations.

Formula 3 and 4

$$m(x, y) = \sqrt{(L(x+1, y) - L(x-1, y))^2 + (L(x, y+1) - L(x, y-1))^2} \quad (3)$$

$$\theta(x, y) = \tan^{-1} \left( \frac{L(x, y+1) - L(x, y-1)}{L(x+1, y) - L(x-1, y)} \right) \quad (4)$$

Each pixel's gradient is also computed to get a feature map. SIFT takes a 16x16 pixel image batch around each keypoint and grids the image batch in many 4x4 small windows. The directions of each window are calculated, and the 128-dimensional vectors are computed, as shown in Figure 1.

Finally, using SIFT to concretizes the direction vector of each window to a histogram as a fingerprint, as plotted in Figure 2.

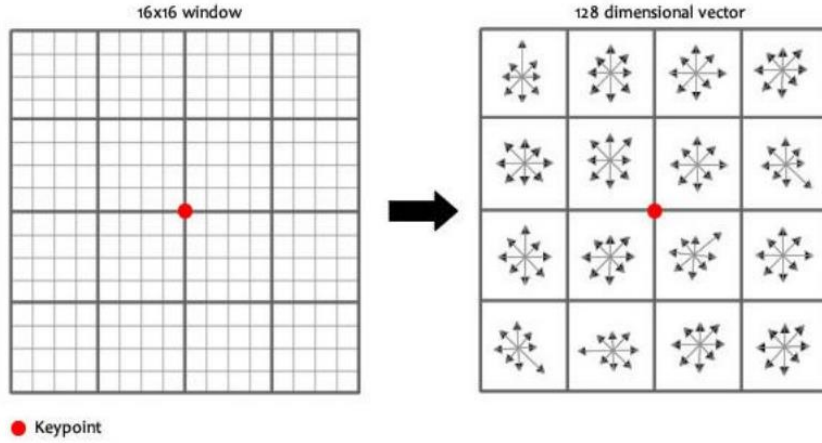


Figure 1 SIFT Dimensional vector



Figure 2 the histogram of the direction vector

### Feature Matching:

Under the feature matching stage, once the unique feature points have been detected and described, the primary objective of feature matching is to establish the corresponding points in the source(reference) image that match those in the target image. The evaluation metric is the Sum of Squared Differences (SSD) to gauge the similarity between feature points, as indicated in equation (2). In the context of feature matching, SSD measures how well two feature points align. There is a more substantial match if the SSD values are comparatively lower.

$$SSD(f1, f2) = \sum (f1[i] - f2[i])^2 \quad (5)$$

As shown in equation (2), the SSD formula is applied where 'f1' represents the feature vector from the reference image, and 'f2' represents the feature vector extracted from the source image. Computation based on the distance allows the algorithm to find the two closest feature points in the second image for every feature point in the first image.

$$\text{The ratio}(\text{nearest SSD}/2\text{nd nearest SSD}) < \text{threshold} \quad (6)$$

The ratio test is implemented to the top two nearest feature points to measure the reliability of a match on the SSD distance. This ratio is used to validate the distance between the nearest feature point and the nearest feature point which falls below a predefined threshold. The default of the threshold in this algorithm is set to 0.5. It is critical as it helps to filter out ambiguous or erroneous matches.

Experiment with Imagepairs\_03\_01 and imagepairs\_03\_02

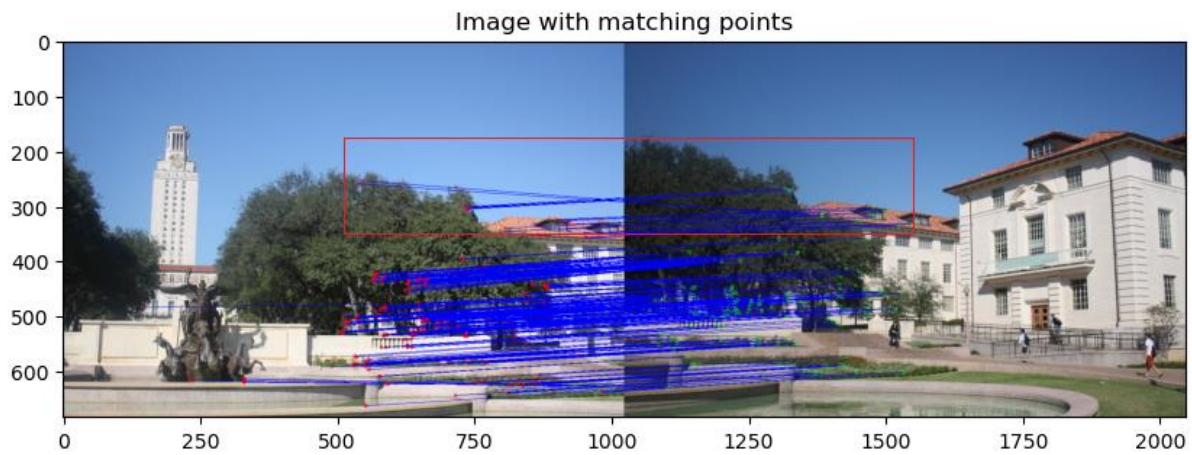


Figure 3 Keypoint match without applying threshold

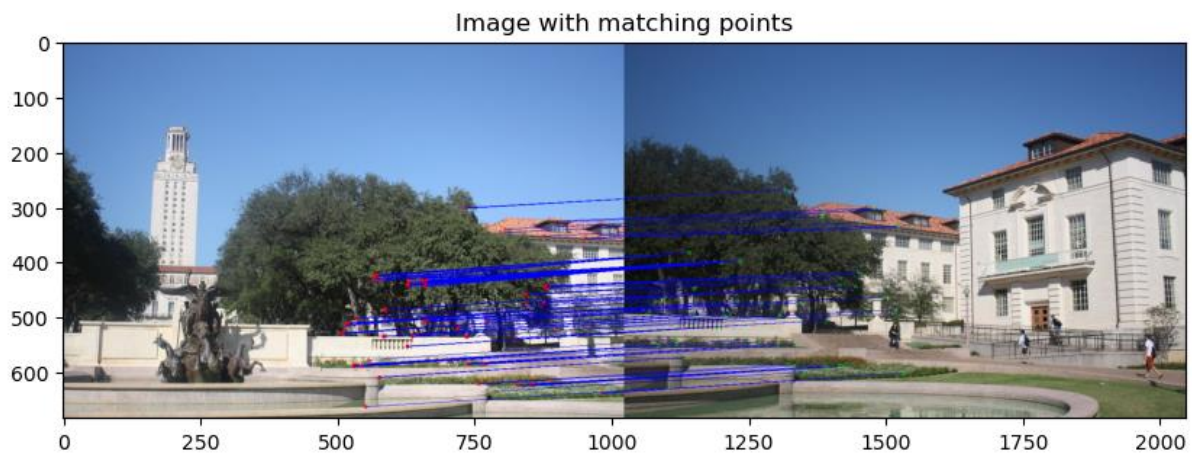


Figure 4 Keypoint match with threshold

Based on our observation gathered from the results, some key points matching is inaccurate; for example, in Figure 3 as an illustration, those points in the red rectangle and the left side key points indicate the top of the tree but match the right edge. Therefore, Figure 4 shows the importance of the inclusion of the threshold, which helps filter out those in accurate matching even though it lost some of the key point matching.

### c. Homography Estimation : Random Sample Consensus (RANSAC)

This section will examine the Homography Estimation using Random Sample Consensus (RANSAC). As described previously in the introduction section, homography is a concept that relies on formulas. It uses projective geometry with a  $3 \times 3$  matrix to represent an invertible transformation[6].

Furthermore, it reflects the matching key point between the matching images[6]. The changes in geometry will eventually affect images as the images are taken from different viewpoints[6]. The following are classifications of the geometry change: translation, rotation, scaling, and distortion. Therefore, the images taken from different viewpoints will be adjusted and aligned during image stitching to create a coordinated single system under homography. While the homography estimation uses the matching key point to find the best key points and becomes an input into the RANSAC algorithm. The RANSAC algorithm has been frequently introduced to calculate the image's homography matrix.



## Random sample consensus (RANSAC)

Random sample consensus(RANSAC) is also an image registration that uses statistical approaches to estimate the randomly selected sets of corresponding points iteratively [4]. It also finds the key point between the consecutive frames and filters out the outliers robustly, resulting from incorrect feature matching while picking the best matches from the inliers. Using the RANSAC allows the reduction of the occurrences of overfitting and requires fewer computation resources. Below, Figure 5 shows the Homography matrix for the image 03\_01 and image 03\_02 pairs.

### Steps on how RANSAC Works [3]

1. S denotes the randomly selected samples and will be labelled as the minimum sample to fit into the model. In our algorithm, we have selected 4 samples.
2. Calculate the count of the H data points (inliers) that fit the model.
3. In this transformation, it is used to find the inliers and outliers
4. The steps above are repeated with N no of times.
5. Select the model with the most significant number of M inliers. Alternatively, based on the probability, it can be represented by formula 7 :

$$p(\mathbf{M} \text{ is correct}) = 1 - (1 - (p_i)^r)n \quad (7)$$

### Experiment with Imagepairs 03\_01 and imagepairs 03\_02

Estimated Homography Matrix:

```
[[ 1.24730257e+00 -9.51774711e-02 -5.36401506e+02]
 [ 1.53239822e-01  1.16386760e+00 -1.45947136e+02]
 [ 2.54816599e-04 -1.81276852e-05  1.00000000e+00]]
```

Total Inlier Matches: 266  
Total Outlier Matches: 179

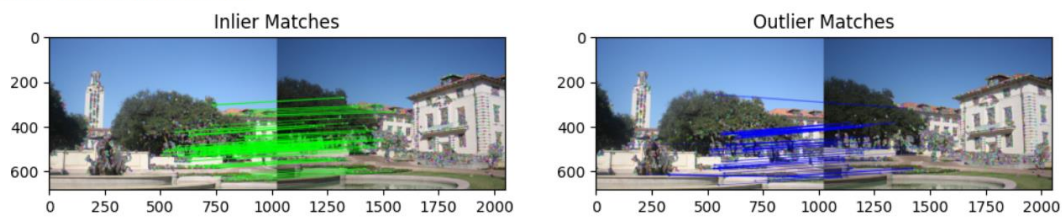


Figure 5 RANSAC Inlier and outlier plot

Based on Figure 5, the top image shows the result of matching key points with the find homography functions. The results are being plotted to give us an overview of the inlier, outlier and noise points. The 2 images in Figure 5 (Bottom images) represent the plot for the inlier point with suitable matches illustrated in green, while the outlier point is drawn in blue.

Based on the image results gathered from imagepair03\_01 and imagepair03\_02, there are 266 inlier matches and 179 outlier matches. In this experimental, the error between points which are being examined and predicted is determined by the threshold, which is used as a measurement. Therefore, different threshold values ranging from 0.6, 1, 4, 7 and 10 are used to see the effect and examine how they would affect the quality of the images.

From this observation, with a smaller threshold value used, the inlier detected will be lesser, and the outlier will increase. The threshold has made it stricter in picking the best points that fall within and vice versa; however, there is no significant impact on the final output images based on the range of experimental values. Therefore, we can conclude that the threshold of the image may be dynamic and depends on each image. In our algorithm, the threshold=4 is a general baseline, as it is difficult to adjust and determine for each image's threshold.

In addition, the ablation studies are also performed on the iteration. Our team tested to include the iteration, although it did not achieve a good visual on the image and resulted in it being more skewed. Therefore, in our algorithm, we have removed the iteration to be included in the RANSAC functions.

In conclusion, our team has built on the observation of the RANSAC function, where it will adopt baseline threshold = 4 without using iteration with direct use from the keypoint matches to find the best-estimated points. It can be portrayed in the final image, where the images are perfectly blended.

#### **d. Image Alignment : Wrap and Blending**

In this section, our team will discuss image alignment, which uses wrap and blending techniques as the final stage illustrated in the standard process in the image stitching flows in the introduction. Under the wrap technique, the image region has been cut out and stitched based on the best matches gathered from the RANSAC result in the previous sections. Let us assume there are two images: One is the reference image, while the other is the transformed image and in the transformed image, all transformation will take place.

Furthermore, in our assignment, the location of the image used as reference and transform is as follows: the right position is the reference, while the left position is the transformed image. It is essential to define which are a reference and transform them to achieve a correct position reflected in a stitched image.

Blending is the final step that our team has implemented. Our team has chosen linear blending, commonly used in image stitching for multiple images in the panorama. This process not only assists in obtaining the natural looks by blending the 2 images stitched together in the final output but also assists in resolving distortion or misalignment of the images.

Linear blending is a methodology which combines images after image wrapping. This allows the transition between the images to be smooth. The final pixel values are calculated in the blending process based on the average weights assigned in the overlapping regions. The image weights are denoted as alpha masks, which will determine the contribution of the overlapping area in the output images. The last step is to blend and stitch the images and provide more appealing natural images.

Step of image blending (refer to Figure 9):

1. Create the mask of the transformed left-side image
2. Create the mask of warped image



3. Calculating the mask of those areas on the right side image does not intersect the left side image.
4. Create the mask of the right side image.
5. Calculate the mask of those areas at the right side image intersecting the left side image.
6. Calculate the mask of the alpha
7. Apply the linear blending to the warped image via the alpha mask

The formula of the linear blending is denoted as :

---


$$Q(i, j) = \alpha * P1(i, j) + (1 - \alpha) * P2(i, j) \quad (8)$$


---

However, the blending of images is not always necessary. For example, in images pair1 and pair 4, the left side image does not have much deformation in reference image space after wrapping. The images are perfectly stitched without any contrast or luma problems.



Figure6 Blended image pair4



Figure7 Blended image pair4 with applying gaussian smooth to alpha

The linear blending is applied, resulting in black lines between 2 images(Refer to Figure 6). To resolve the problem, our team has implemented Gaussian Smooth to smooth the alpha to get Figure 7. Figure 5 is the same as the image, including the warp function.

However, we also observe that adding the Gaussian Smooth to alpha causes much higher computation during the experiment than the warp method, and applying blending resulted in wasted computation resources. Therefore, our team has selected linear blending to generate the image which is illustrated in Figure 8 shows the implementation of our algorithm.

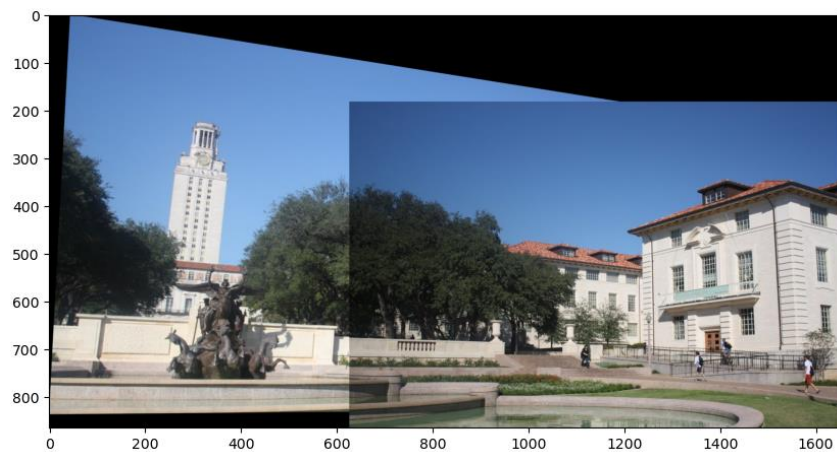


Figure 8. Image pair3 wrapping

image masks

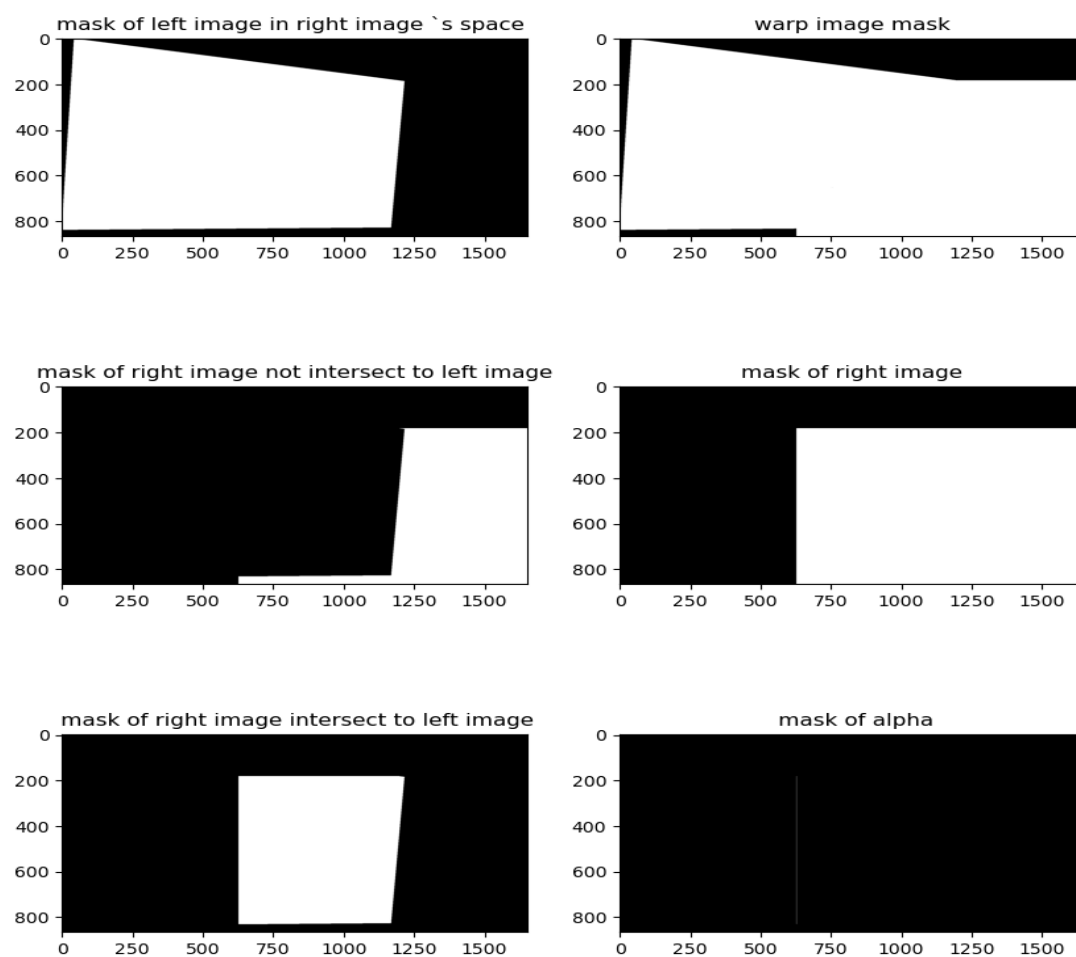


Figure 9 image pair3 masks

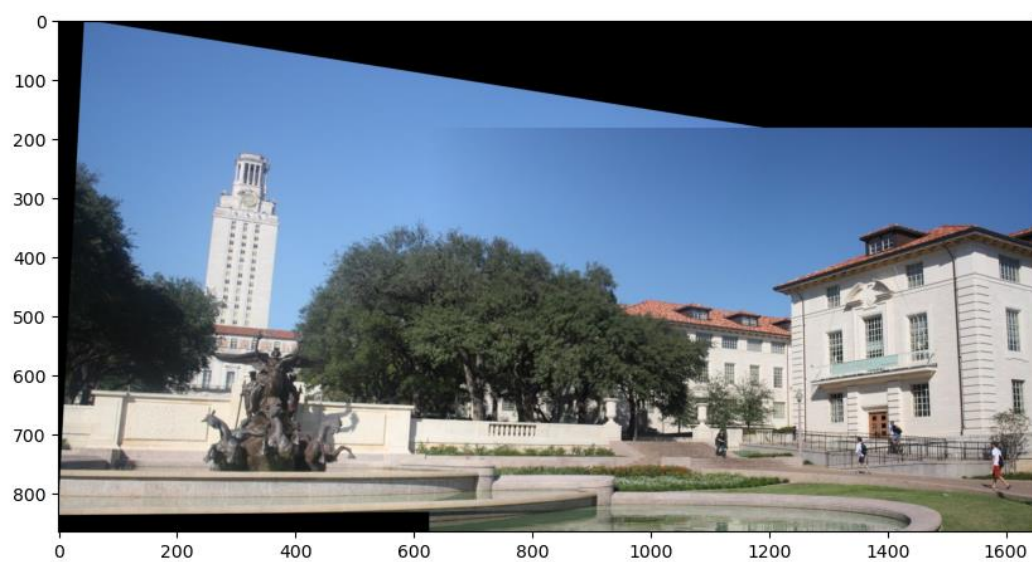


Figure 10 image pair3 blending

### 3. Conclusion

In conclusion, this image stitching assignment has allowed the team to reflect during the design process. We observed that the traditional way of performing image stitching may result in alignment issues in the region where overlapping and non-overlapping occur. This resulted in the image distortion as well.

The techniques like RANSAC and Linear Blending are being introduced where the threshold and alpha calculation are being assisted in to ensure the final output image is smooth and natural in appearance while reducing the occurrence of misalignment. In addition, the parameters used in each feature are independent while aiming to achieve optimal results.

Therefore, integrating these techniques and algorithms allows our team to stitch the image and create seamlessly panorama images successfully.

### Reference

- [1] David G Lowe, "Distinctive Image Features from Scale-Invariant Keypoints", International Journal of Computer Vision, vol.50, No. 2, 2004, pp.91-110
- [2] Karami, E., Prasad, S. D., & Shehata, M. (2017). Image matching using SIFT, SURF, BRIEF and ORB: performance Comparison for distorted images. arXiv (Cornell University). <https://arxiv.org/pdf/1710.02726.pdf>
- [3] Lowe, M. B. (2007). Automatic Panoramic Image Stitching using Invariant Features. International Journal of Computer Vision, 59-73. doi:<https://doi.org/10.1007/s11263-006-0002-3>
- [4] Monnin, D. a. (2010). An Effective Rigidity Constraint for Improving RANSAC in Homography Estimation. (J. a. Blanc-Talon, Ed.) Berlin: Springer Berlin Heidelberg. Retrieved Oct 18, 2023, from [https://link.springer.com/remotexs.ntu.edu.sg/chapter/10.1007/978-3-642-17691-3\\_19](https://link.springer.com/remotexs.ntu.edu.sg/chapter/10.1007/978-3-642-17691-3_19)
- [5] Wang, Z. a. (2020, 08). Review on image-stitching techniques. Multimedia Systems. doi:10.1007/s00530-020-00651-y
- [6] Premsingh, P. (2023, May 21). *Image Stitching using OpenCV — A Step-by-Step Tutorial*. Retrieved from <https://medium.com/@paulsonpremsingh7/image-stitching-using-opencv-a-step-by-step-tutorial-9214aa4255ec>
- [7] Sánchez, J. a. (2018). An Analysis and Implementation of the Harris Corner Detector. *Image Processing On Line*. doi:10.5201/ipol.2018.229
- [8] Wikipedia. (2023, July 23). *Harris\_corner\_detector*. Retrieved from Wikipedia,: [https://en.wikipedia.org/wiki/Harris\\_corner\\_detector](https://en.wikipedia.org/wiki/Harris_corner_detector)



## 4. Appendix

Below is the image for the other 3 pairs for each section of the reports.

### A. Harris Corner:





First image with Harris Corner detections	Second image with Harris Corner detections
<p>Image pairs 01_01:</p> 	<p>Image pairs 01_02:</p> 
<p>Image pairs 02_01:</p> 	<p>Image pairs 02_02:</p> 

Image pairs 04\_01:



Image pairs 04\_02:



## B. SIFT:

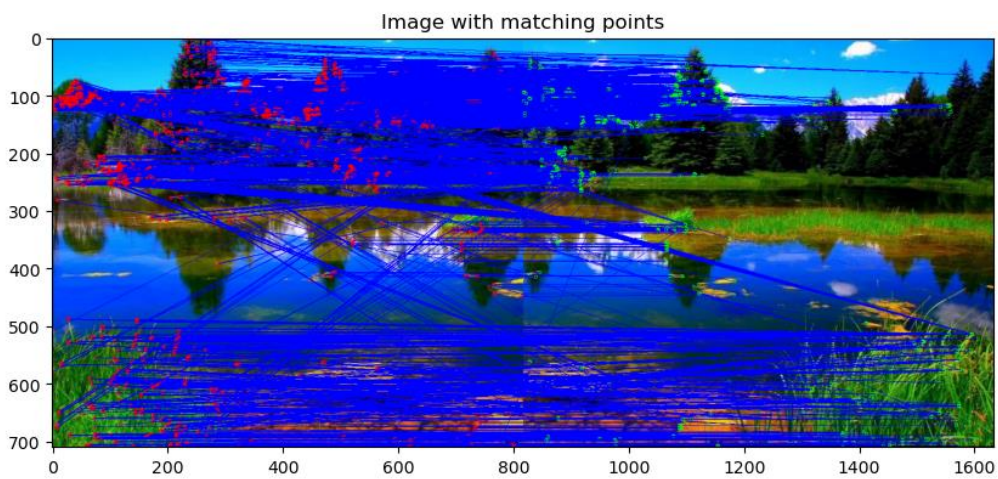


Image pair1 without apply threshold

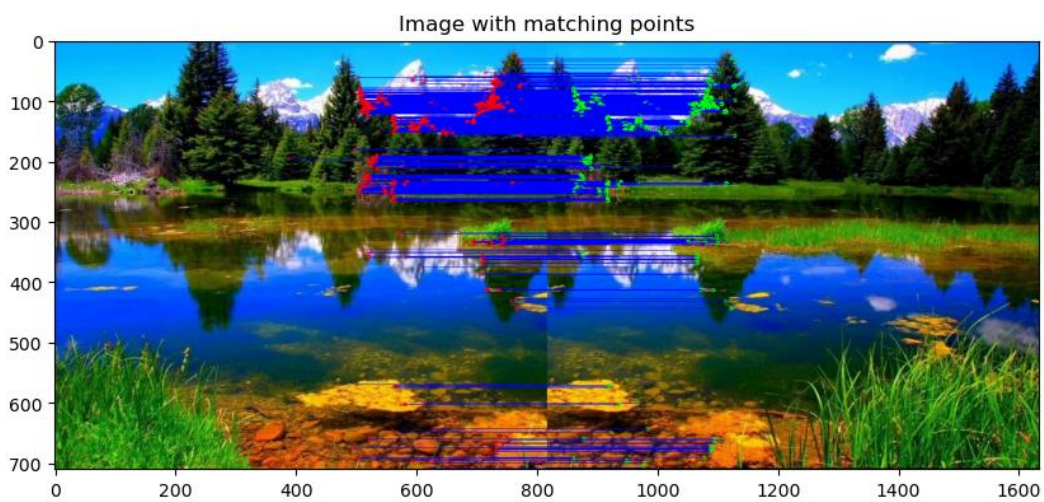


Image pair1 with apply threshold



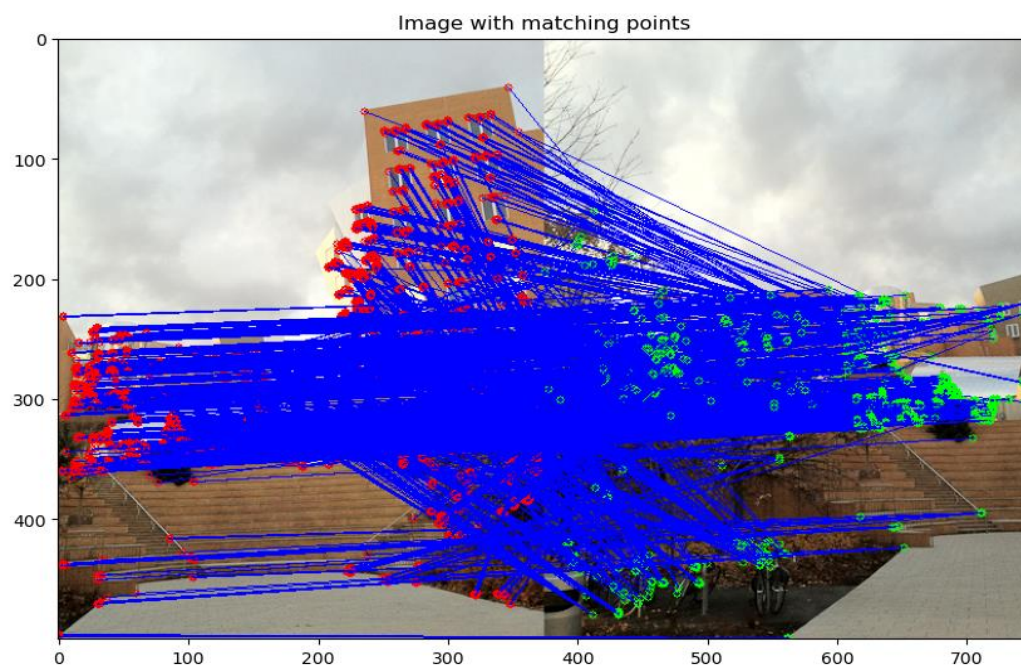


Image pair2 without apply threshold

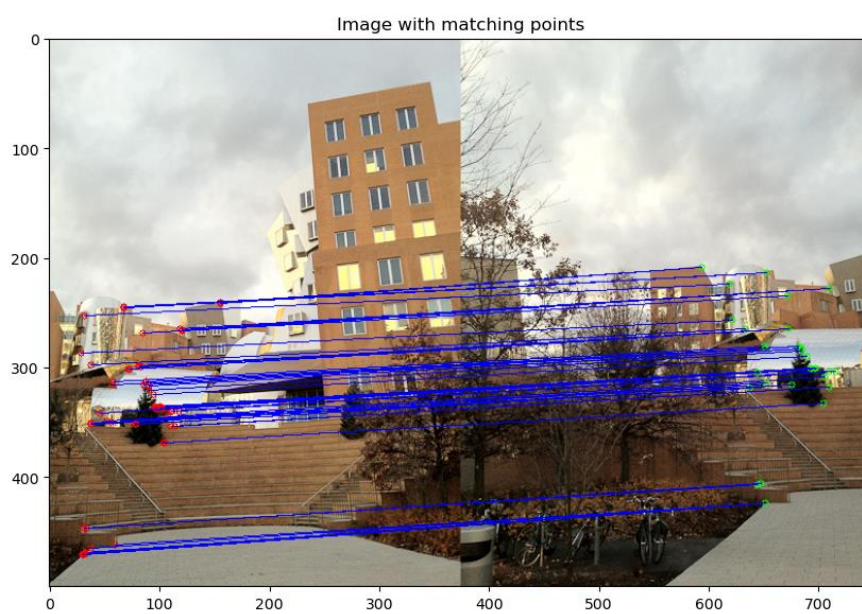


Image pair2 with apply threshold

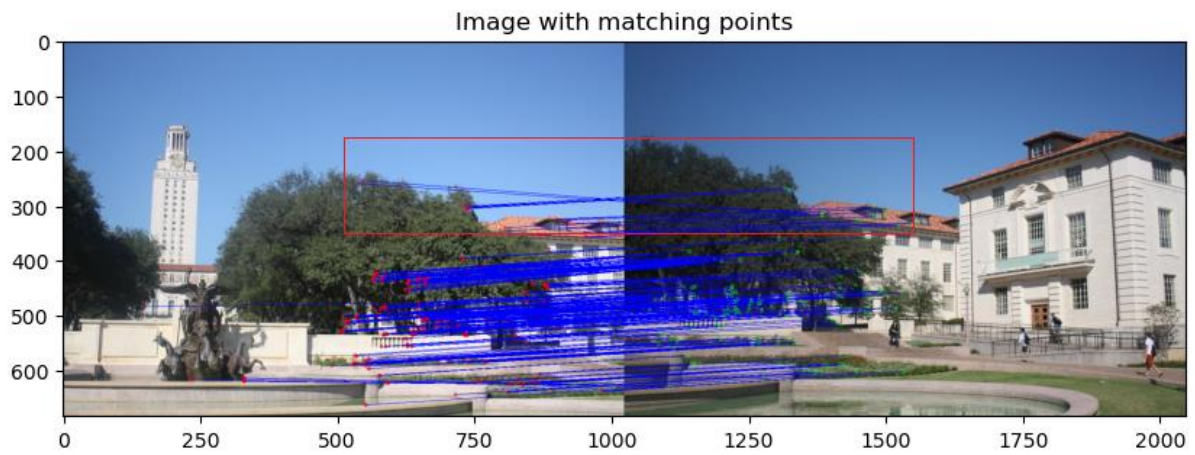


Image pair3 without apply threshold

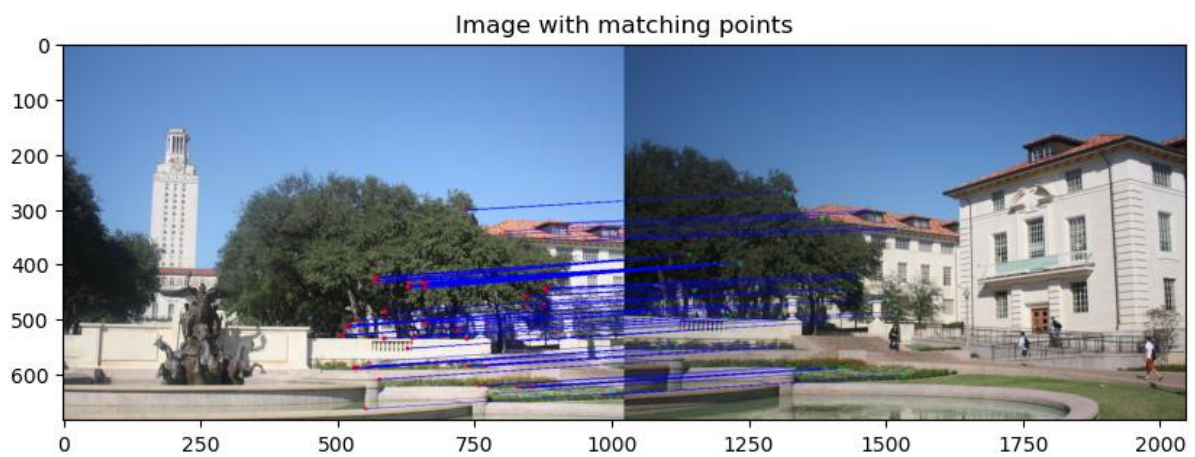


Image pair3 with apply threshold

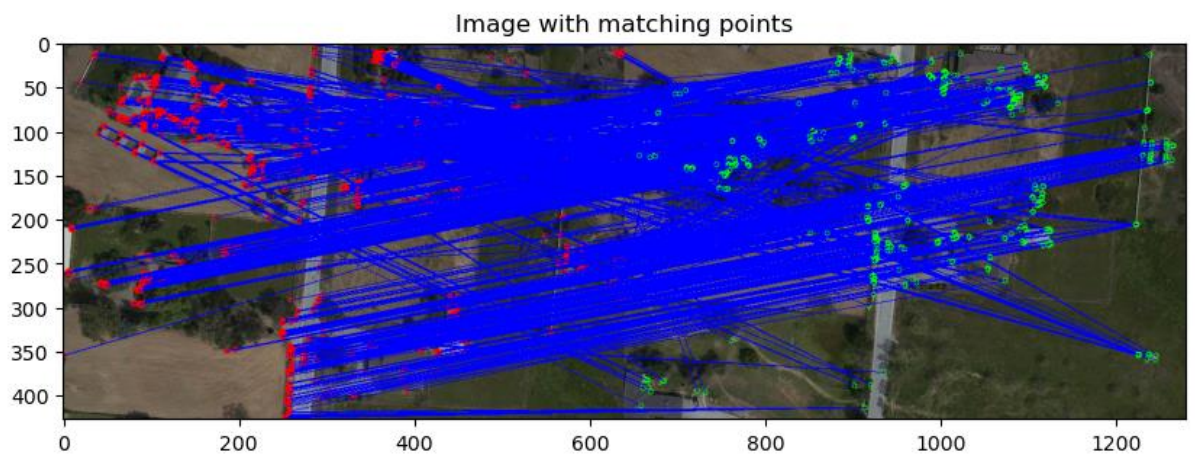


Image pair4 without apply threshold



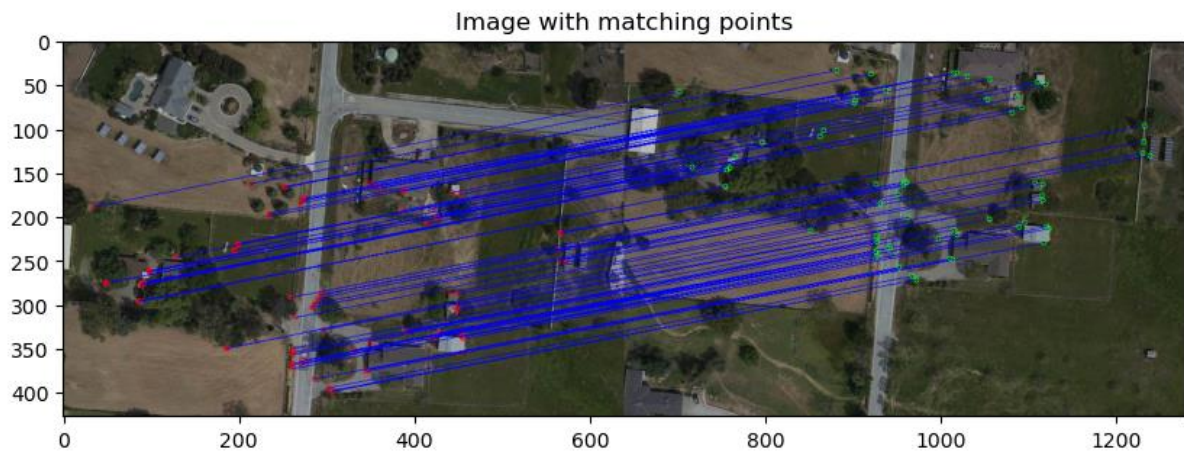


Image pair4 with apply threshold

### C. RANSAC

Below is the inlier and outlier plot of using RANSA

```
Estimated Homography Matrix:
[[ 9.99735342e-01  3.04747060e-04 -4.62892118e+02]
 [ 4.73567394e-05  1.00027071e+00 -7.86761266e-02]
 [-2.27204283e-07  6.87855368e-07  1.00000000e+00]]
```

Total Inlier Matches: 1106  
Total Outlier Matches: 29

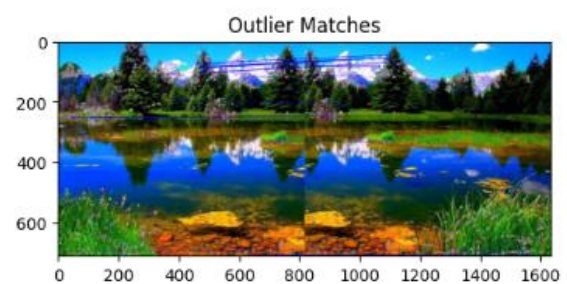
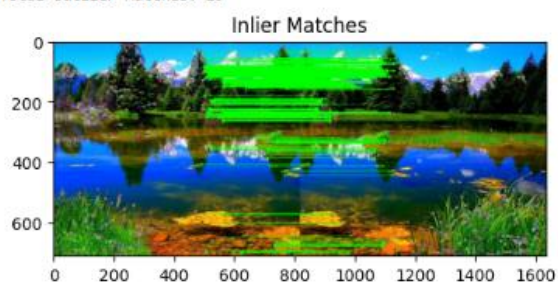


Image pair 1 plot with inlier and outlier

```
Estimated Homography Matrix:
[[ 1.00078392e+00 -4.83627451e-04  4.63030486e+02]
 [ 2.72971930e-04  9.99880126e-01 -1.24320292e-02]
 [ 1.14192431e-06 -4.26780447e-07  1.00000000e+00]]
```

Total Inlier Matches: 41  
Total Outlier Matches: 16

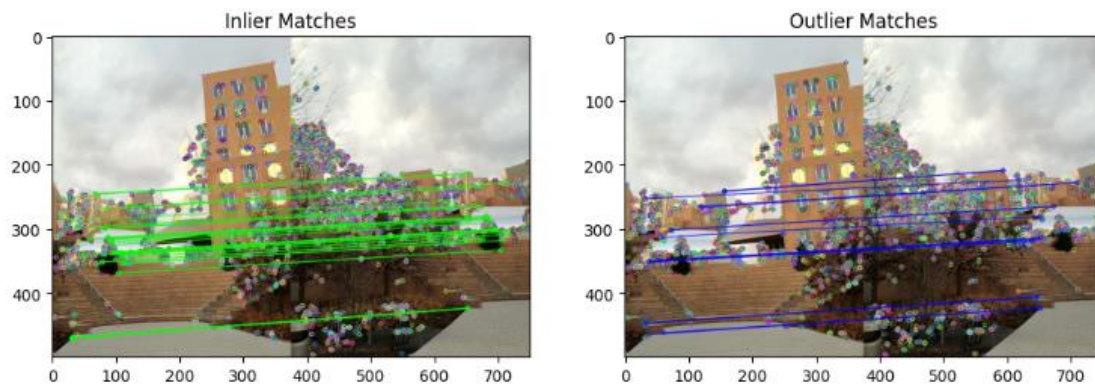


Image pair 2 plot with inlier and outlier

Estimated Homography Matrix:

```
[ [ 9.94738185e-01 -9.34562911e-03 3.00447964e+01]  
  [ 1.94288900e-02 9.90146697e-01 -1.31261547e+02]  
  [ 1.04520289e-06 -1.44018120e-05 1.00000000e+00]]
```

Total Inlier Matches: 252  
Total Outlier Matches: 16

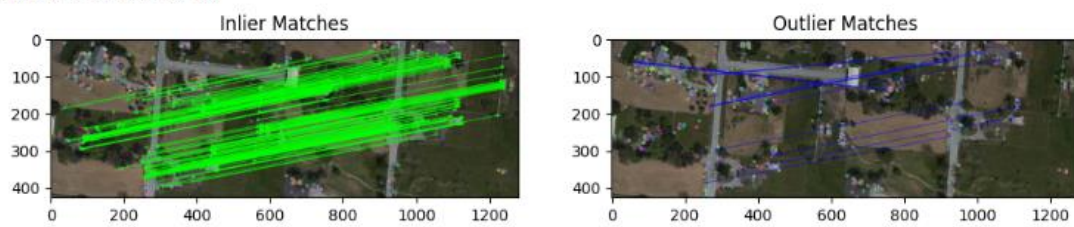


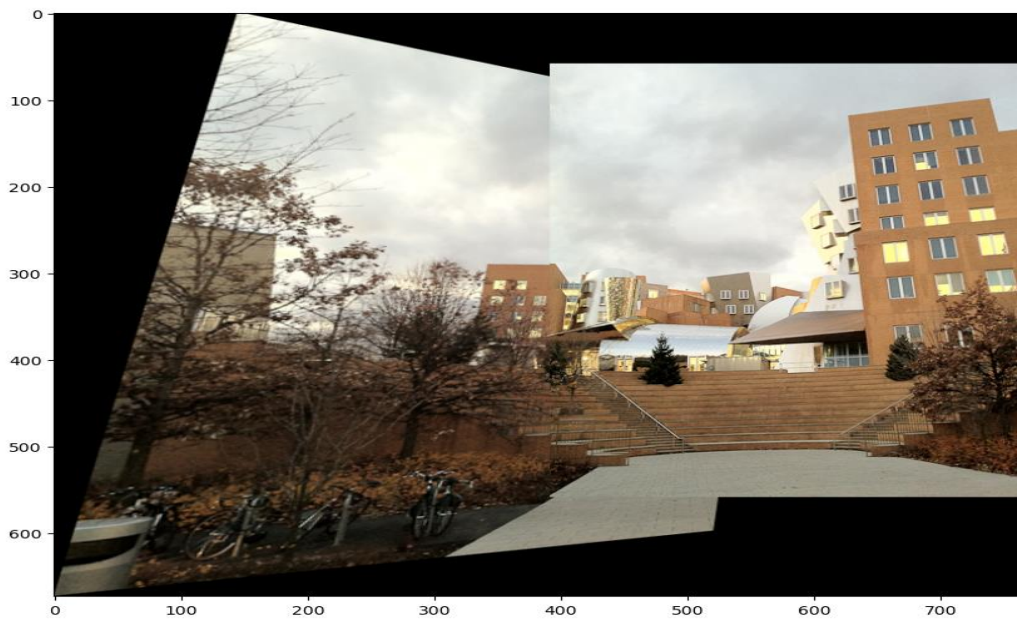
Image pair 4 plot with inlier and outlier

## D. Warp and Blending

Image Warp:

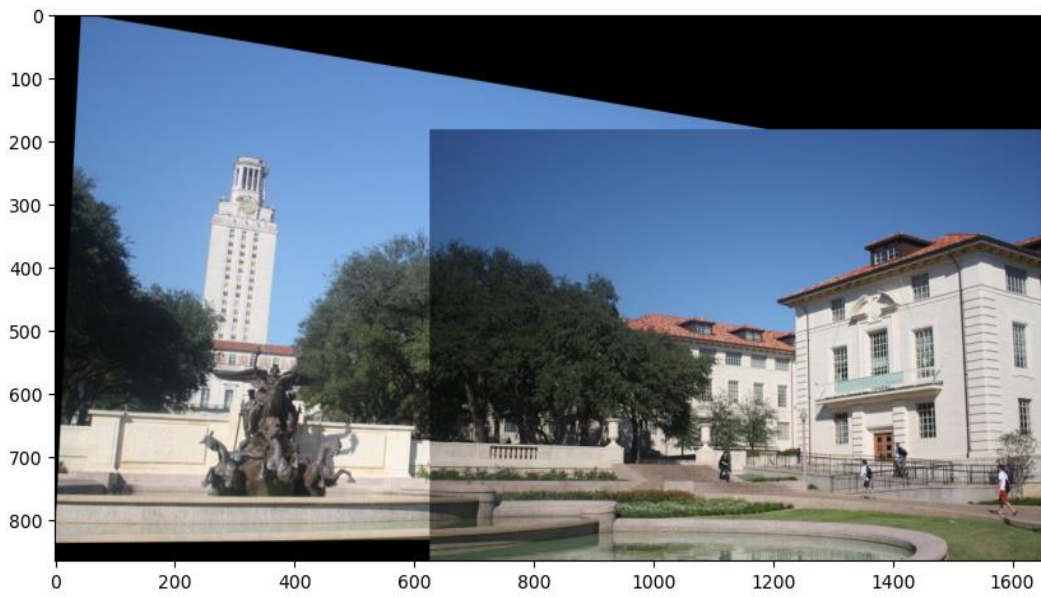


Warp Image pair 1



Warp image pair 2





Warp image pair 3

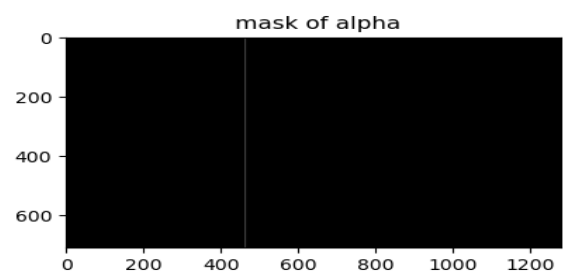
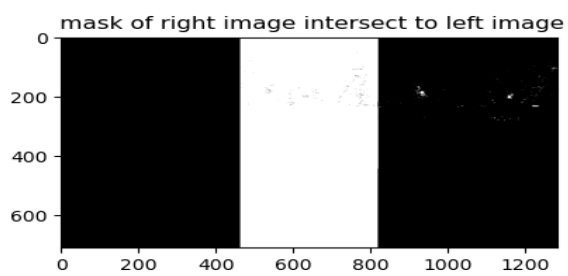
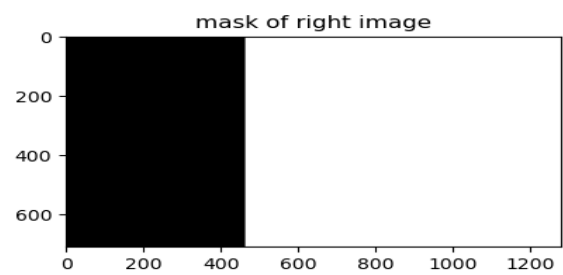
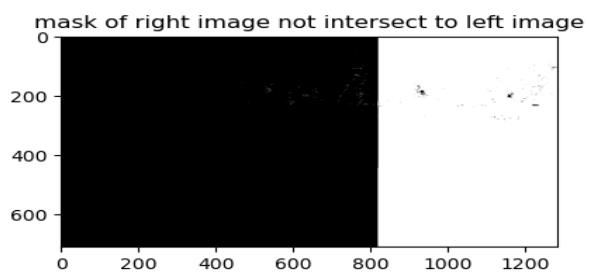
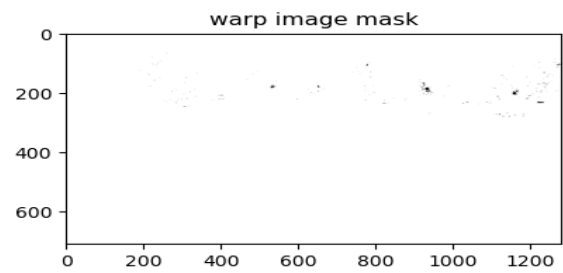
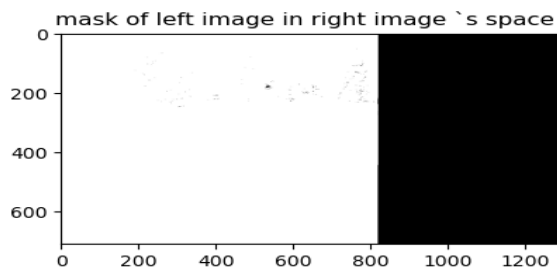


Warp image pair4



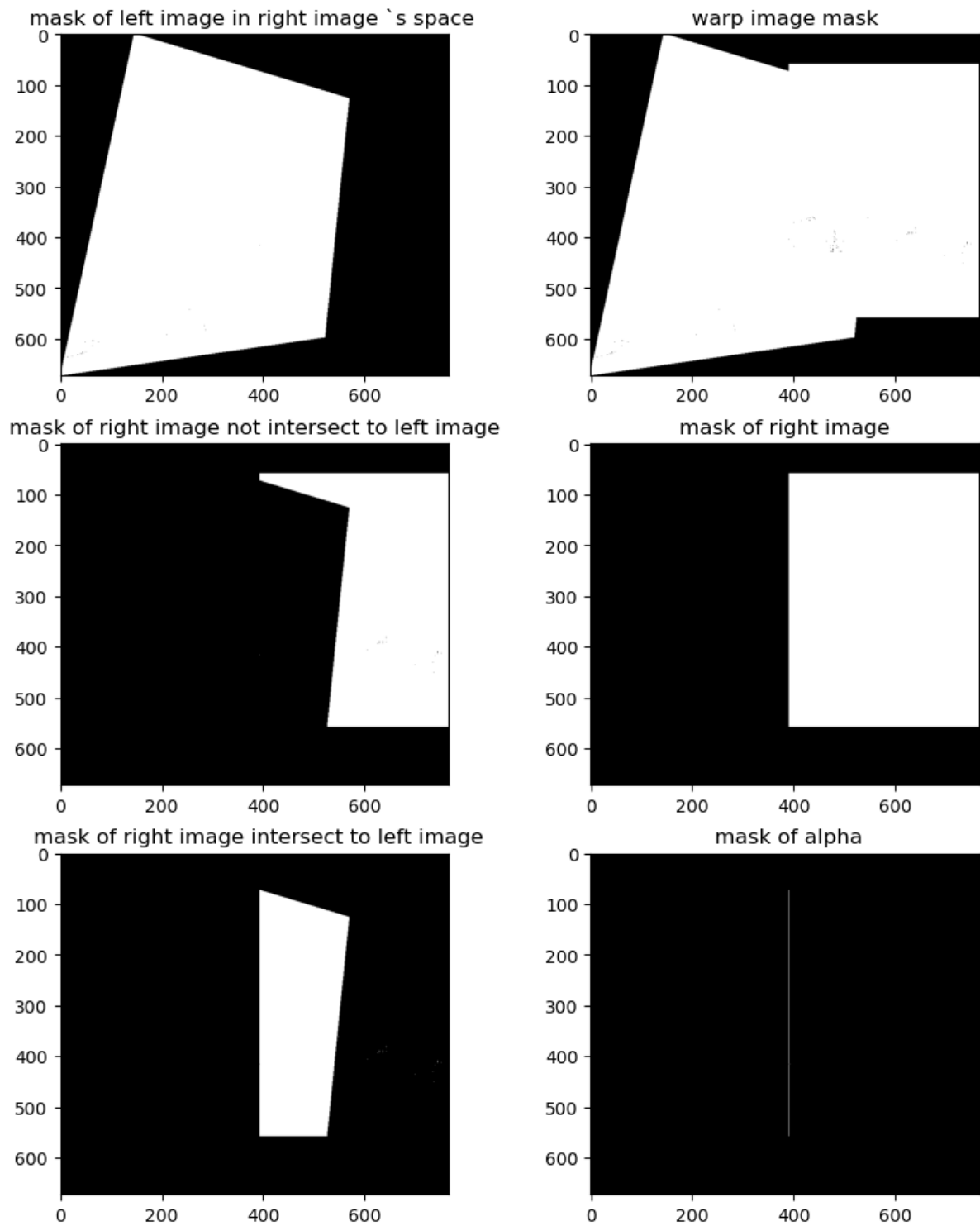
Mask:

image masks



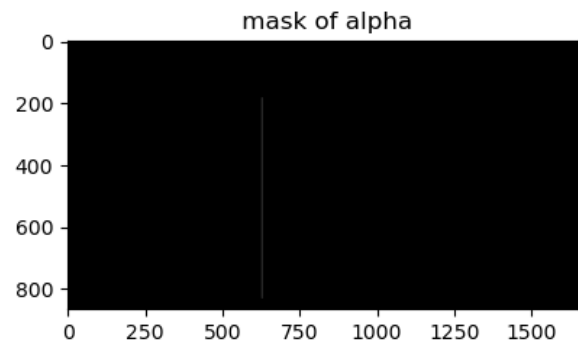
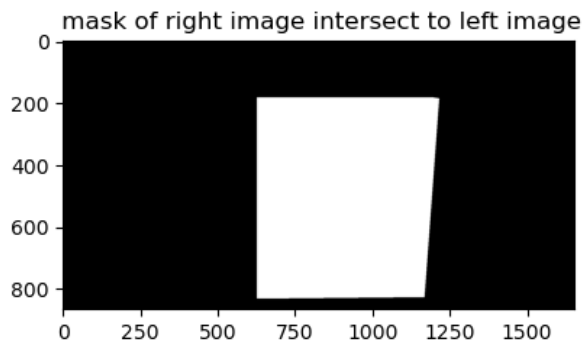
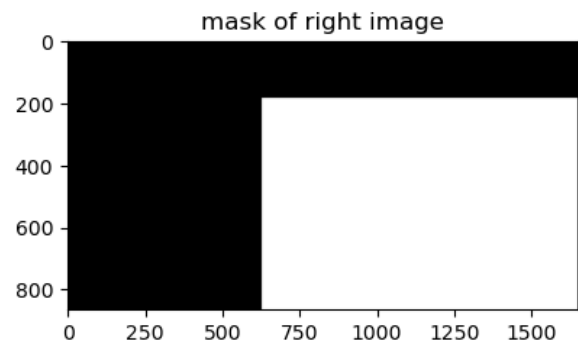
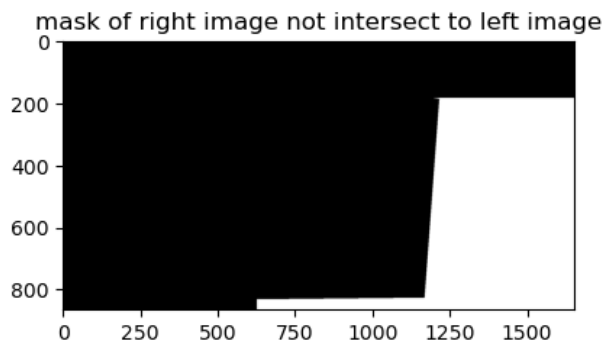
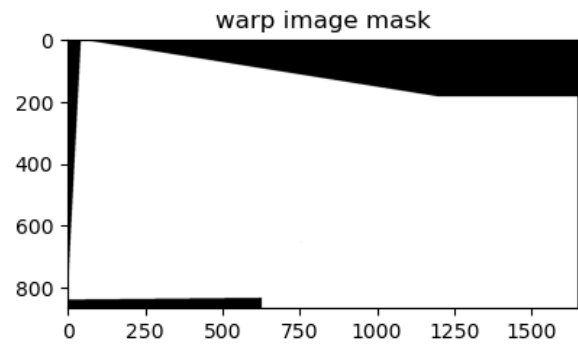
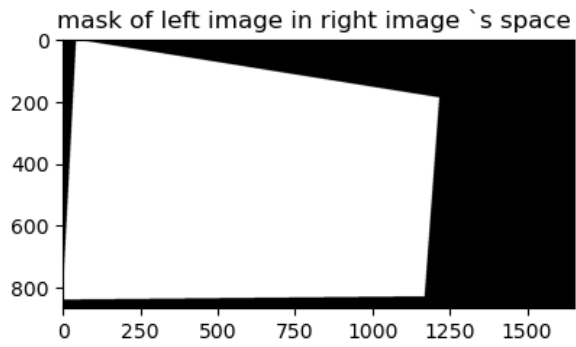
Masks of image pair1

## image masks



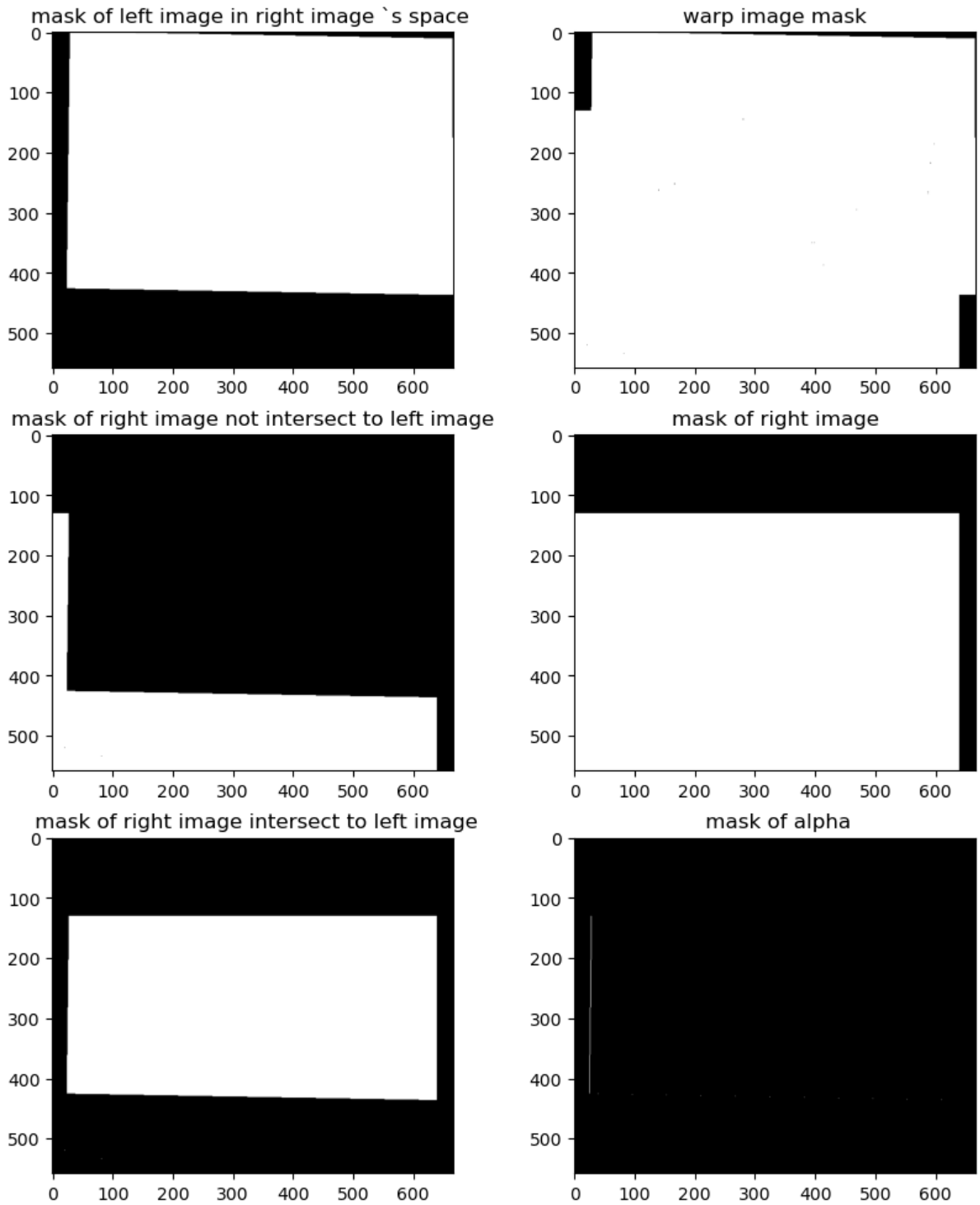
Masks of image pair2

## image masks



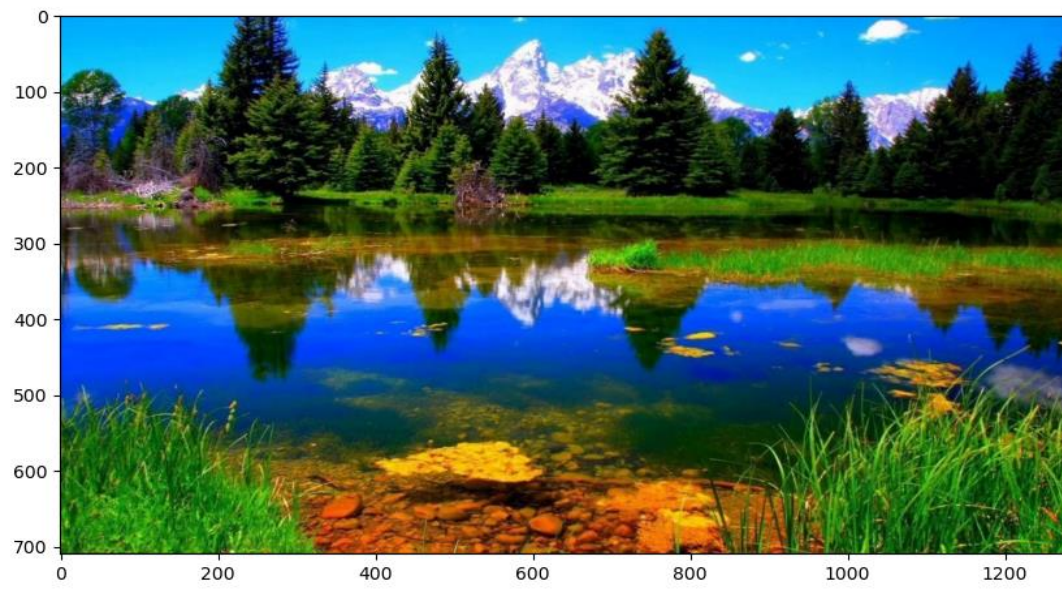
Masks of image pair3

## image masks

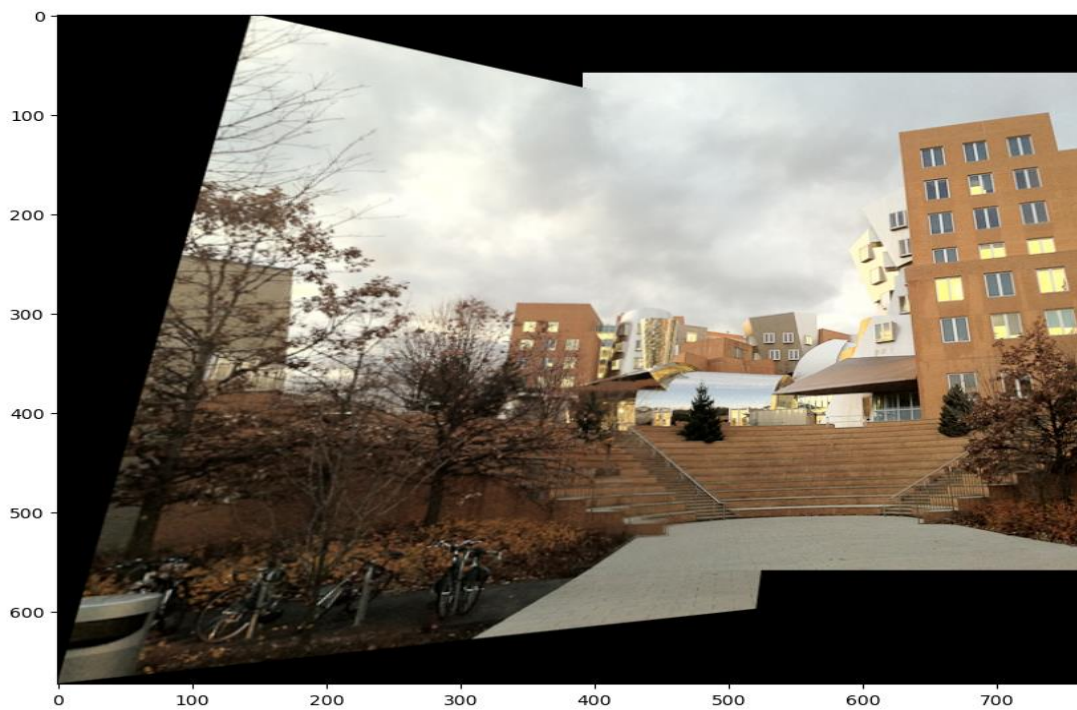


Masks of image pair4

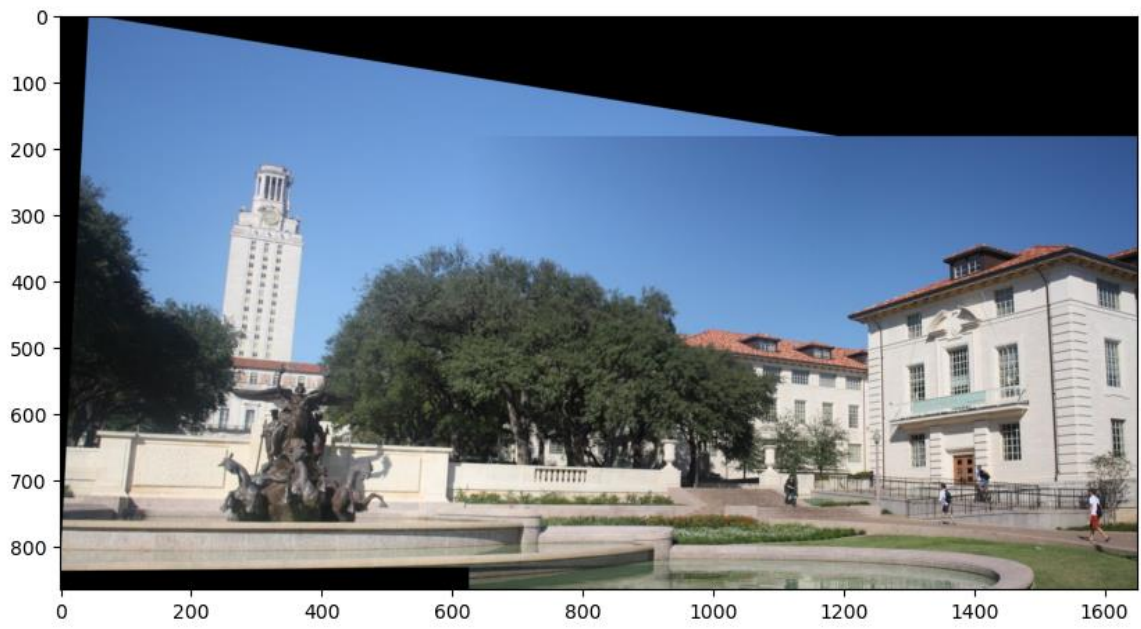
Blended image:



Blended image pair1



Blended image pair2



Blended image pair3



Blended image pair4





Blended image pair4 with applying gaussian blur to alph