Assignment 1: Panoramic Image Stitching

EE5731 VISUAL COMPUTING

Name	Matric No.
Lim YangZheng	A0195048X

Part 1: 2D Convolution

2D convolution in image processing combines two signals (image and 2D-kernel) to produce an output signal. Here we study on three variants of kernels and understand how the effect is of different kernels on the same image.

```
%% Kernel Types

KERNEL_SOBEL_HORIZONTAL = [ -1, -2, -1
                             0,  0,  0
                             1,  2,  1];

KERNEL_SOBEL_VERTICAL = [ -1, 0, 1
                         -2, 0, 2
                         -1, 0, 1];

KERNEL_GAUSSIAN_BLUR = [ 0,  0,   0,   5,   0,  0, 0
                         0,  5,  18,  32,  18,  5, 0
                         0, 18,  64, 100,  64, 18, 0
                         5, 32, 100, 100, 100, 32, 5
                         0, 18,  64, 100,  64, 18, 0
                         0,  5,  18,  32,  18,  5, 0
                         0,  0,   0,   5,   0,  0, 0];

KERNEL_LAPLACIAN_GAUSSIAN = [ 0,  0, -1,  0,  0
                              0, -1, -2, -1,  0
                             -1, -2, 16, -2, -1
                              0, -1, -2, -1,  0
                              0,  0, -1,  0,  0];
```

```
HAAR_MASK_1 = [1;
               -1];

HAAR_MASK_2 = [1, -1];

HAAR_MASK_3 = [-1;
               1;
               -1];

HAAR_MASK_4 = [-1, 1, -1];

HAAR_MASK_5 = [1, -1;
               -1, 1];
```

Figure 1: Kernel Types

In our study, convolution is applied to a single channel. Hence a colour image is converted into a grayscale image before applying the convolution. Zero padding is also used to maintain the size of the output signal with the input signal. The kernel size can also be scaled up with an integer multiplier.

Figure 2: Input image used

From the results we can see that Sobel kernels and Haar type 1 and type 2 are like edge detectors, Gaussian kernel will have a blur effect to the image. Scaling up of the kernel will extract features that are larger.

Code Reference: Part1.m, Convolution2D.m;

1) Convolution Visualisation Results

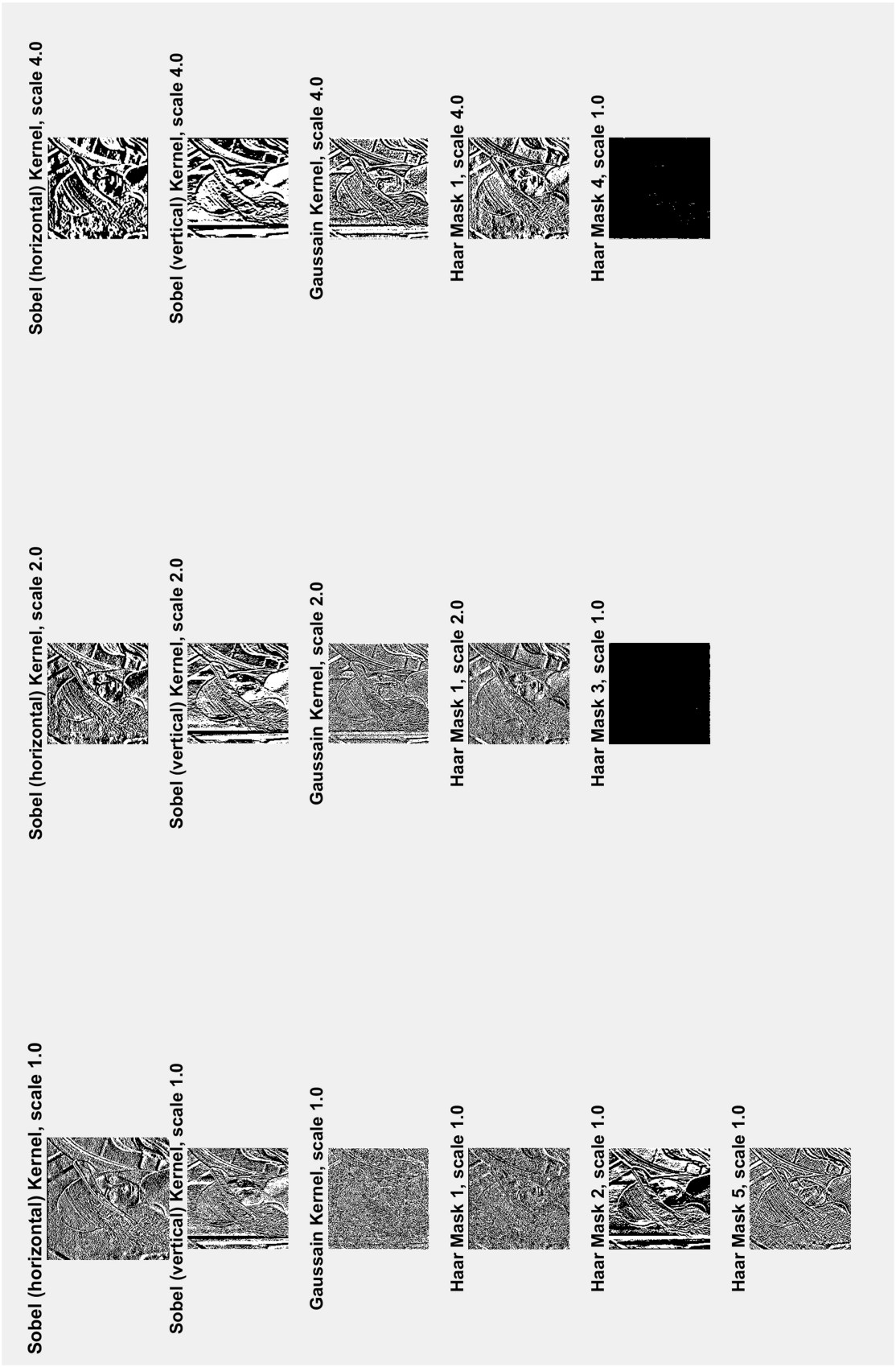

Figure 3: Convolution results

Part 2: SIFT Features and Descriptors

To extract SIFT features and descriptors, an open source library *VLFeat* is used (http://www.vlfeat.org/api/sift_8h.html). Figure 3 shows 50 random SIFT features obtained from the image. The yellow circle shows the centre coordinate (x, y) of the frame and the green boxes shows the orientation descriptor. Different sizes of the green boxes indicate the scale of the descriptor found.

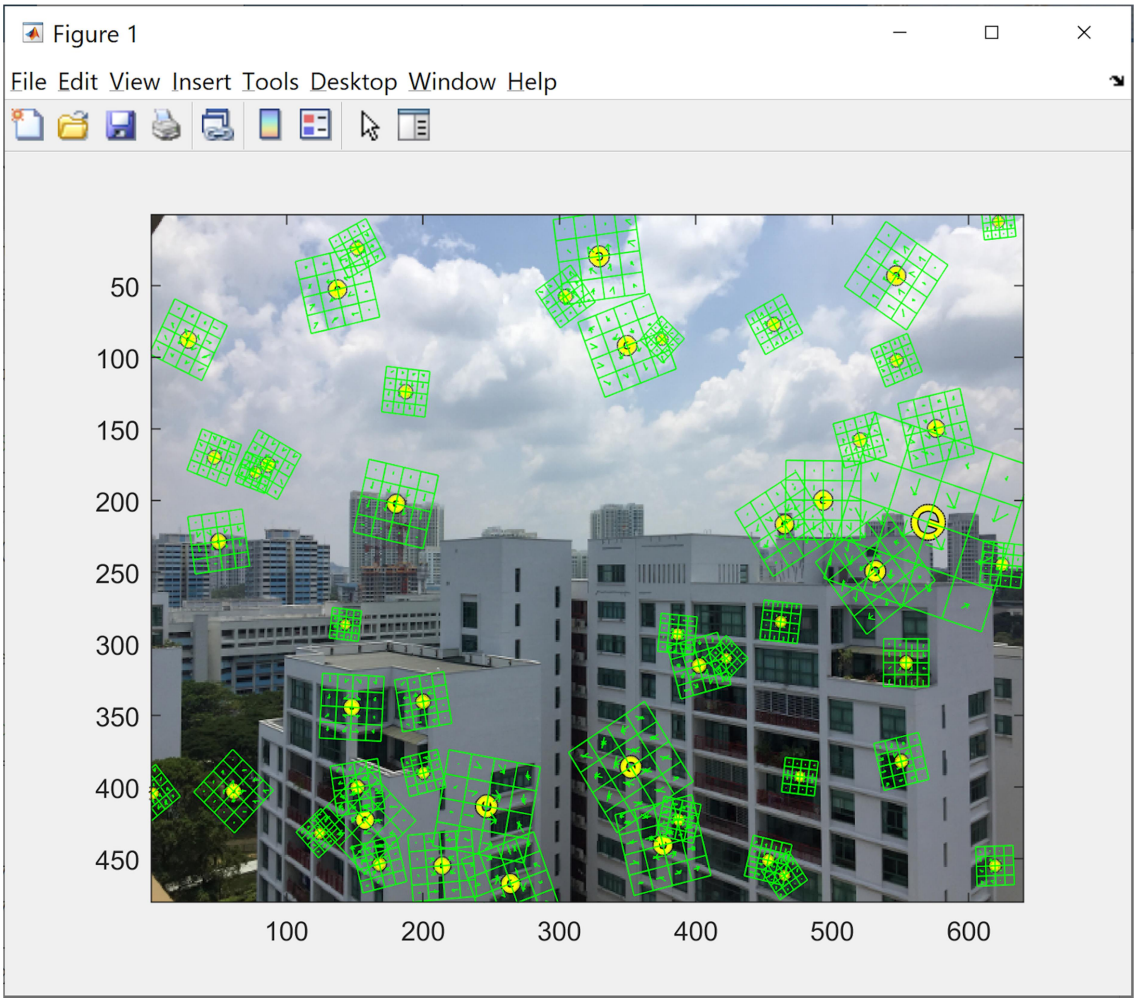

Figure 4 : SIFT features

Code Reference: Part2.m

Part 3: Homography

In this section, we are dealing with homography where one image is converted into another image of a different perspective. In order to convert Image 1 to the coordinates in Image 2, we first find the coordinates of the four corners of the chessboard in both image 1 and 2 from the user inputs (marked with 'X'). With four points the homography is able to be solve in a closed-form system.

$$\because\ x_1 = H\,x_2 \qquad \therefore\ x_2 = x_2\,H^{-1}$$

To get the homography of image 2 to image 1, the inverse homography, H^{-1}, is applied to image 2.

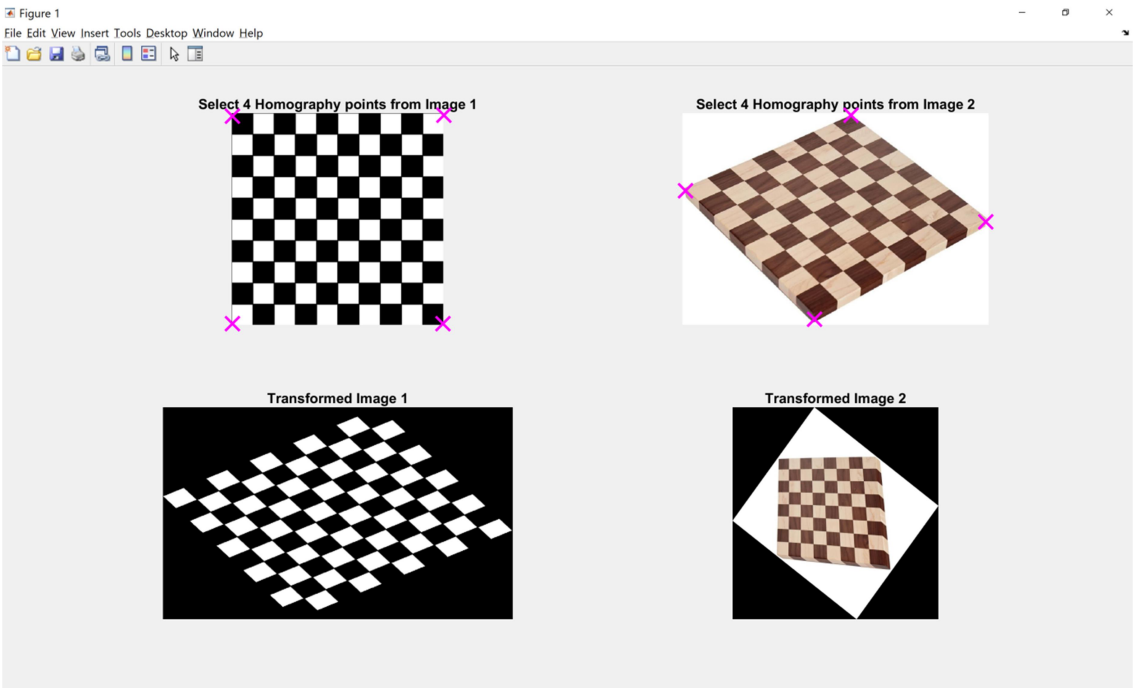

Figure 5: Homography transformation

Code Reference: Part3.m ; GetHomography.m; GetAndDrawPoints.m

Part 4: Manual Homography + Stitching

In this section, we are trying to stitch two images into one. This is similar to Part 3 where we find four points in both images and transform Image 2 to Image 1 perspective. Then a big canvas is ready to fit both Image 1 and transformed Image 2. Image 1 is first placed on the canvas and then Image T2 is overlapped onto image 1.

As the four points selected have different depth information viewed in both images, we are not able to get a flat image when both images are stitched together. The shapes of the four points are different which cause the distortion of the image.

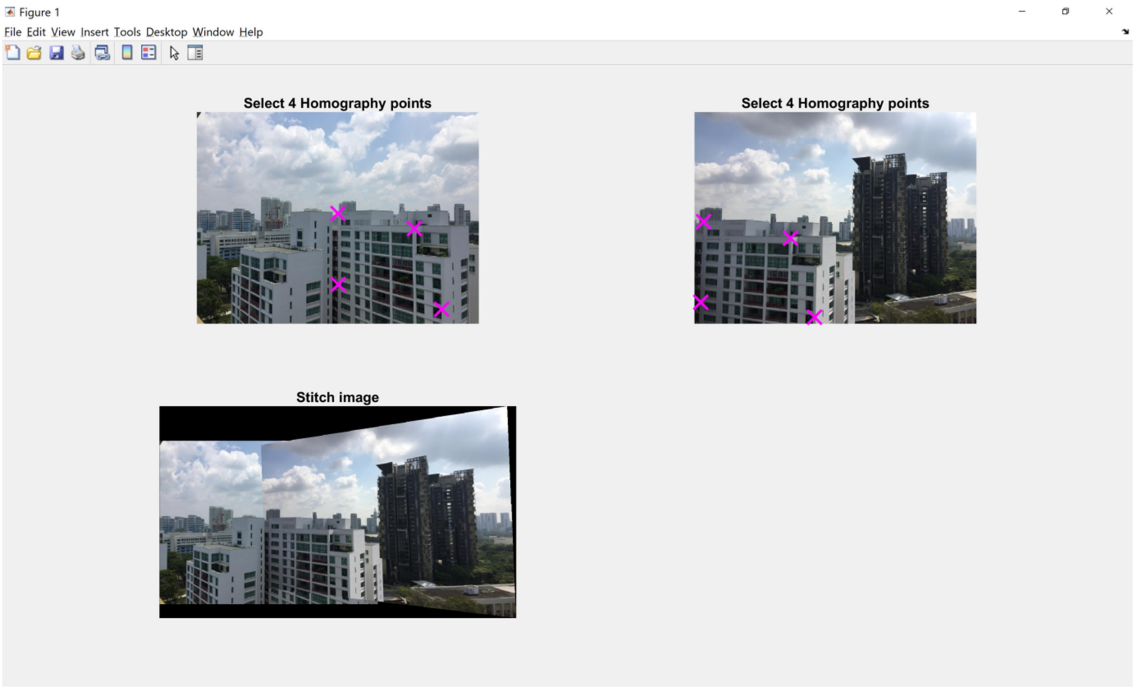

Figure 6 : Stitching of two images with user inputs

Code Reference: Part4.m ; GetHomography.m; GetAndDrawPoints.m

Part 5: Homography + RANSAC

Now we are able to stitch two images with the help from user. In order to eliminate user inputs, we use SIFT detector to find the points that matches between the two images. To obtain keypoints that matches between the two images, a cross check is done for the best match features in between image 1 to image 2 and image 2 to image 1. A keypoint is considered as a matching pair if exists in both matches.

$$Best_i = \min_j \sqrt{\sum_k | v_i[k] - v_j[k] |^2}$$

$where,$
$i = index\ of\ keypoint\ in\ Image\ 1;$
$j = index\ of\ keypoint\ in\ Image\ 2;$
$v_i = descriptor\ of\ keypoint\ i;$
$k = index\ of\ descriptor$

However not all the SIFT features matched can be used to compute the homography, some keypoints are features that look alike by chances. In order to eliminate those points, RANSAC is introduced to filter out the noise and the inliers obtain are the points that can be used to compute the homography.

The RANSAC parameters that used for this study are iteration (i = 35), random keypoints pairs (n = 5), L2 norm distance (d = 15).

After obtaining the points, the homography can be obtained with SVD as the problem is now an over-determined system.

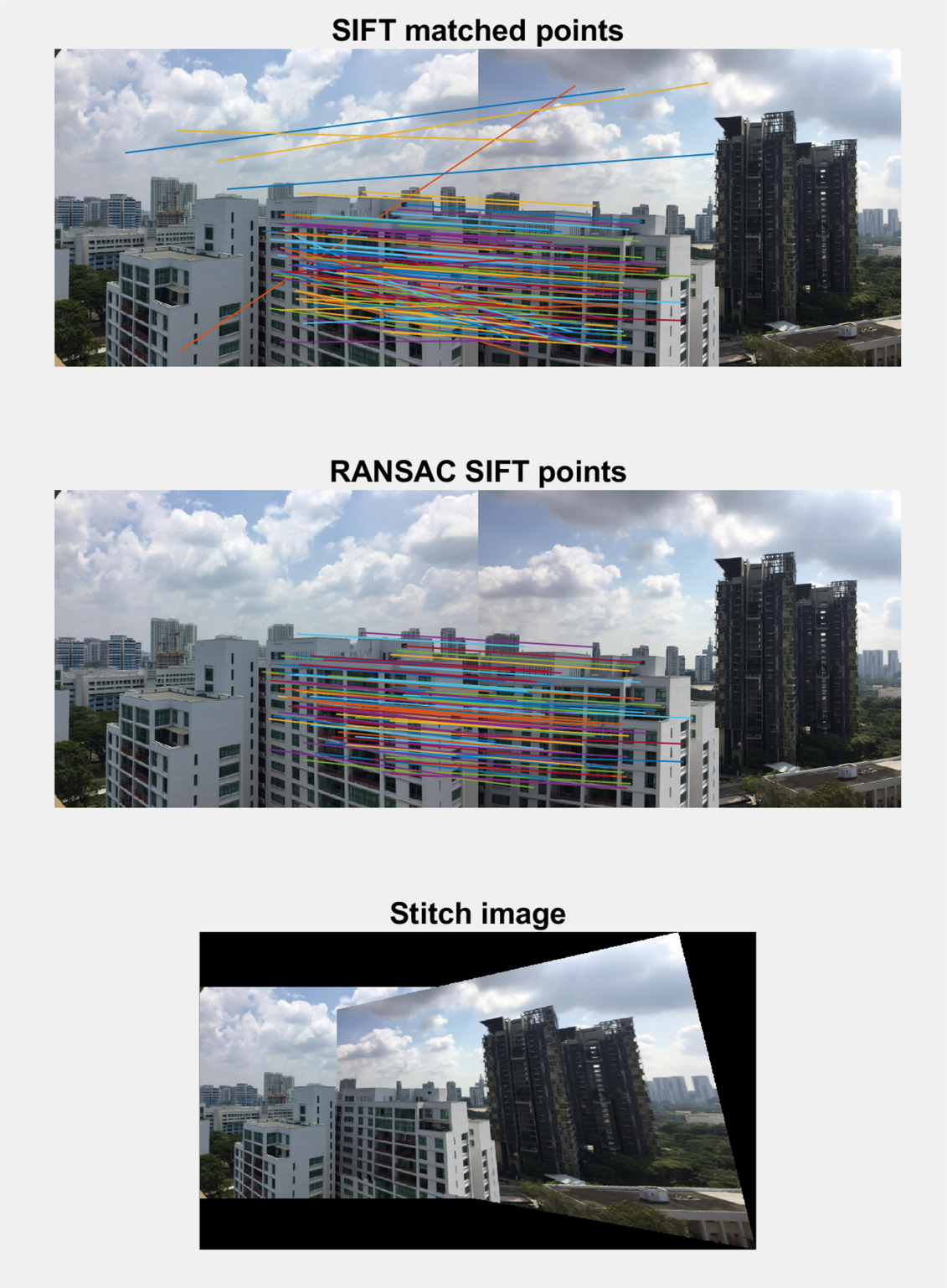

Figure 7: SIFT with RANSAC for Image Stitching

Code Reference: Part5.m ; KeypointMatching.m; GetHomographySVD.m; FindHomographyRANSAC.m

Part 6: Basic Panoramic Image

Finally, we are able to form a panoramic image with a list of ordered images as shown in Figure 8 containing of a total 5 images. Firstly, the homography, H_n, between two images are calculated as shown in Part 5. However as we would like to transform all the images into the same global coordinates, the affine transformation for all homography is applied to a single homography.

In our example, the image that is in the middle will be selected as the global coordinate so that the result image would be viewed from the middle. Therefore all of the images will be transform in respect to the middle image as a result shown in Figure 8.

Code Reference: Part6.m ; KeypointMatching.m; GetHomographySVD.m; FindHomographyRANSAC.m

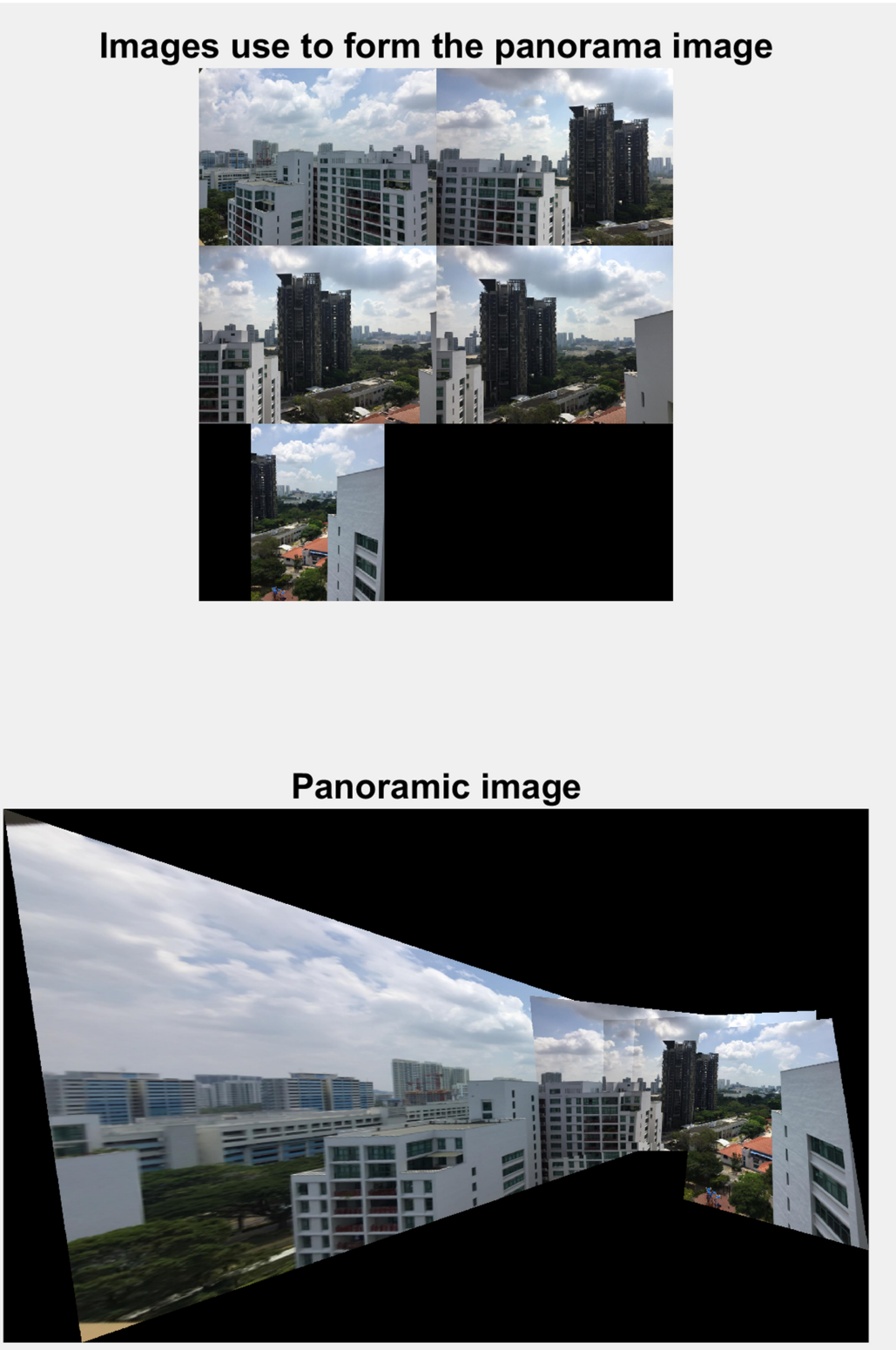

Figure 8: Image Stitching of Ordered Images

References

Feature Based Panoramic Image Stitching. (n.d.). Retrieved from https://www.mathworks.com/help/vision/examples/feature-based-panoramic-image-stitching.html

Opsahl, T. (n.d.). *Estimating homographies from feature correspondences*. Retrieved from https://www.uio.no/studier/emner/matnat/its/nedlagte-emner/UNIK4690/v16/forelesninger/lecture_4_3-estimating-homographies-from-feature-correspondences.pdf

Te VLFeat Authors. (n.d.). *VLFeat SIFT detector and descriptor*. Retrieved from http://www.vlfeat.org/overview/sift.html