# HW2 - Panoramic Image Stiching

Wen-Jie Tseng (0556146)
Li-Yang Wang (0656619)
Yi-Chen Lee (0656611)
Computer Vision - 2018 Spring Semester

## 1. INTRODUCTION

We have learned how to manipulate and infer meaningful data from images in class. Also learned all concepts that are necessary for this project such as SIFT and projective transformation. This time we try to match images and create an automatic panoramic image stitching based on features.

## 2. IMPLEMENTATION DETAILS

For section 2.1-2.3, we demonstrate the result by using 2 images as source image: box image and box-in-scene image. The images are shown in figure 1.

### 2.1 Interest Points

To analyze whether an image is similar to another, detecting their interest points and treat them as the characteristic of the images is a common method. By comparing their interest points, we could tell if some parts of one image exist in another. To achieve our goal, we first initiate a SIFT operator by using function

```
cv2.xfeatures2d.SIFT_create()
```

After enabling the operator, we then use the function built in SIFT to compute the interest points. We demonstrate the result in figure 1 and figure 1:
In the picture the interest points, also known as key points, are displayed as blue circles. These points are regarded as key points since they are edges, corners, or blobs in the picture.


Figure 1. The Box-in-scene image and its key points. Clockwise, starts from upper left: 1) box, 2) box in the scene, 3) key points of box in the scene image, and 4) key points of box image.

## 2.2 Descriptors

After obtaining the interest points, we then move on to extract the features. As mentioned previously, interest points are used to compare input images. In order to compare if they are similar, we have to know "how does it look like" around the interest points; in other words, we have to extract the data around chosen points. In this assignment we used SIFT descriptor. SIFT descriptor takes data from a 16x16 pixel square with interest point being the center, and split it into 4x4 cells; from these cells the descriptor generates 16 8-bins histogram. Finally, after combining all histograms and doing L2-normalization, the result is a feature factor created regarding this particular interest point.

## 2.3 Feature Matching

For feature matching, we first pair up feature vectors. First we create a brute force matcher, and apply KNN match with K = 2 to the feature vectors. What we have now is a roughly-matched result, so further filtering and checking is needed. We filter out inappropriate matches by using David Lowe's test with ratio = 0.7. The result of matching the box image and the box-in-scene image is shown in figure 2:

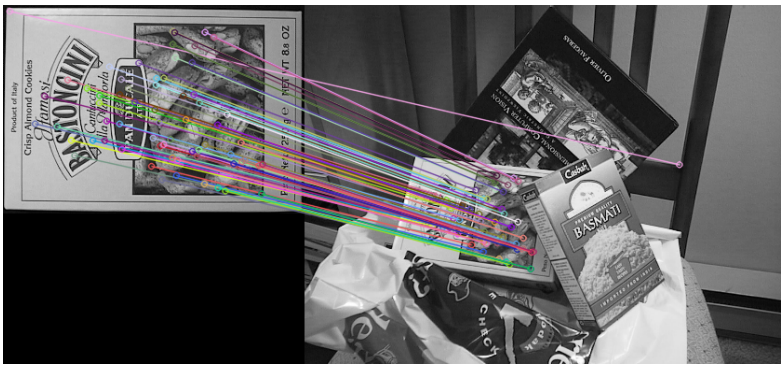


Figure 2. Matching 2 images

As shown, an inappropriate matching still exists in the picture(the one matching from the top-left corner of the box to the right of the chair). From the result we can see that further test should be applied.

## 2.4 Homography

First we match SIFT features between two given images. Then by using those SIFT feature pairs, a homography matrix is calculated via RANSAC. To estimate H, we start from the equation b = Ha. Through writing elements one by one in homogeneous coordinates, we get the following constraint:

$$b = Ha \Leftrightarrow \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} H_{11} & H_{12} & H_{13} \\ H_{21} & H_{22} & H_{23} \\ H_{31} & H_{32} & H_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ x_1 \end{bmatrix}$$

### 2.4.1 RANSAC

RANSAC will choose 4 random correspondences, calculate a homography, count the number of inliers, and keep the homography if it is better than any homography yet found.

## 2.5 Image Wrapping

After finding the homography matrix, the next thing we want to do is stiching one image on another. First we use the function in opencv library,

```
cv2.perspectiveTransform()
```

to get the relative perspective of second image. Next, we find the dimension of output image, calculate dimensions of match points, and create the output array after Affine transformation. We wrap the first image into the second image (the perspective transformed one) by

`cv2.warpPerspective()`

function. Thus we can obtain the result in 5. For blending the gap between two images, out idea was finding the overlapping part and do pyramid blending. But we haven't got enough time to implement it in this report. At last, we compute a cropping function to crop the black part.

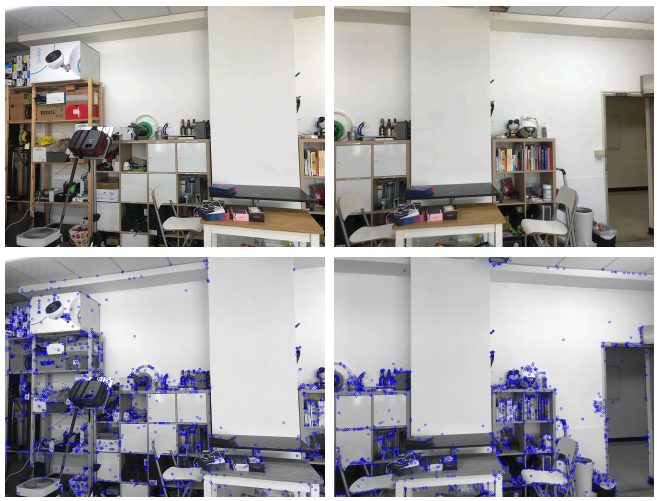

Figure 3. Using a lab scene as out input.

## 3. RESULTS

We try to stitch a series of images from a lab. The results are shown in Figure 3, 4, and 5. There are still some parts of black frame should be cropped. There is still a portion of pixel should be processed by blending, to make panoramic image more smooth.

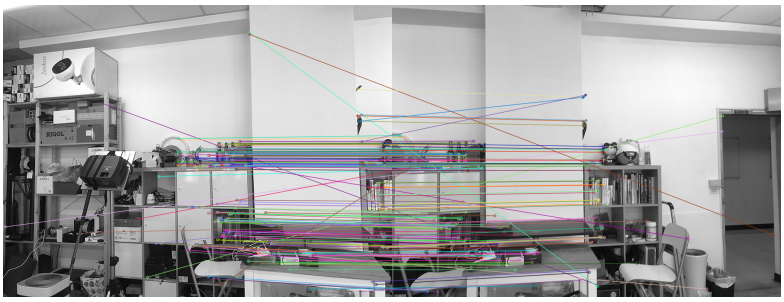

Figure 4. Matches of two lab scenes.

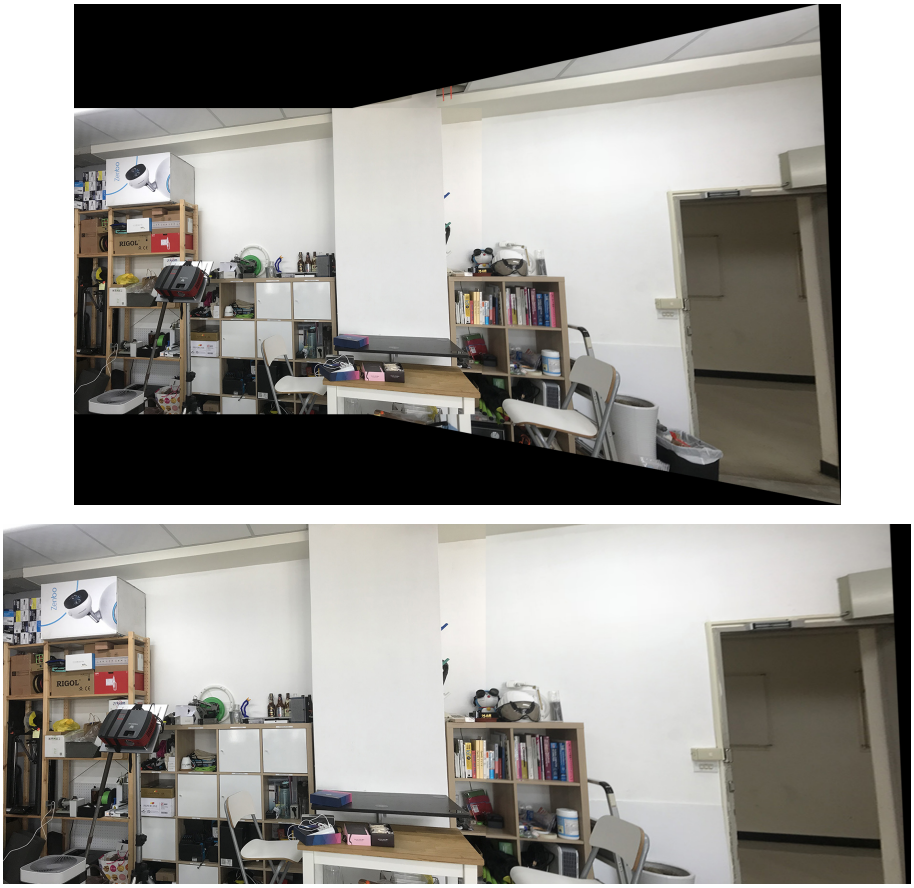

Figure 5. Final stitching result. The one below is a cropped one.

## 4. DISCUSSION

In this homework, we spent many time on RANSAC but cannot produce the correct homography matrix. We actually used the function provided in opencv library. Our homography matrix will turn into an identity matrix after the RANSAC iteration and cannot provide transformation for two input images. After calling function to get the correct homography, we met another problem in image stitching. In Figure 6, the stitched image suffered from strongly distortion. Although we can stitch many images, but the result was poor. We consider the reason may be no blending and too many black parts in the first stitched image pair.

We are expecting ourselves to update the stitching problem and implement blending to generate a better panoramic image before demo. We found some more interesting steps, for example, conducting cylindrical project to each image before stitching, can make the 360 panoramic image more realistic.

Figure 6. A distorted example.

## 5. WORK ASSIGNMENT PLAN

1. Interest point detect and feature matching: Li-Yang Wang
2. Finding homography with RANSAC: Yi-Chen Lee
3. Image stiching and blending: Wen-Jie Tseng

## REFERENCES


1. CHTseng's blog
2. Feature mapping-opencv document
3. Image source of box images
4. A very nice blog about image stiching.
5. Homography,opencv,calculation principle