Project Report Lab(04)

Create a mosaic image by joining together a group of images.

1 Introduction

The given task is to produce a mosaic image from given dataset.According to task, I have stitched multiple images that have numerous keypoints features in-common and I tried to produce a mosaic image (as close as original scenes). The accomplish this task I had to follow different hierarchical steps to generate the results.
The main objective is to find the matching between multiple images, the process involves to find **key-points** and assigning **Descriptors** to those points that separates the points from one another, after this i had to find best matching among those points. Finally, **Affine Transformation** is needed to apply to adjust the different matrices.

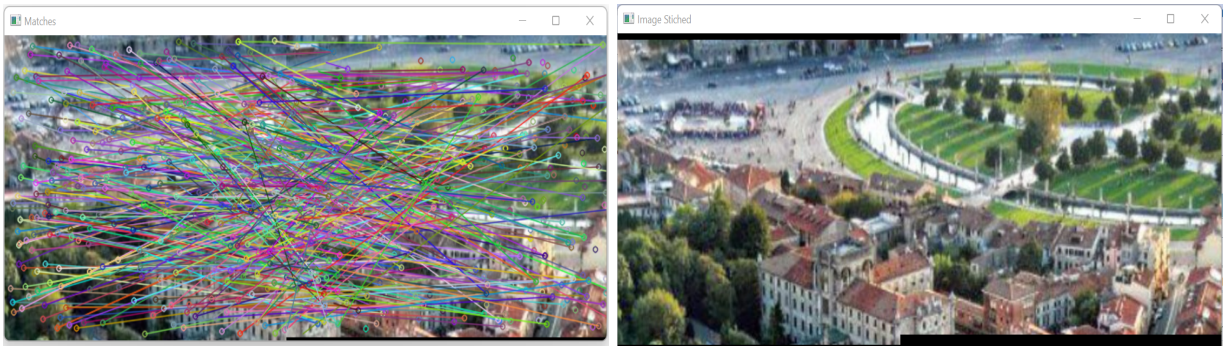
For this algorithm, I have tried two datasets **T1** and **RT1** from provided dataset.

2 Steps

For the project I have build two different functions **sift_des_stiching()** and **Affine_Rotation()** plus a **main()** function. Below are the description how this algorithm works.

2.1 sift_des_stiching()

Starting with **sift_des_stiching()** ,the most part of the project is done by this function. This function takes two images and provide the stitched image. This is done for each two pictures and the the result of two picture is again stitched with third picture, and I recall this function until all the images are stitched with each others.
I manually loaded the images from folder **T1** and passed the pair of images to this function, which takes the images and computes the keypoints and Descriptor using **SIFT** function and **dectectandcompute()** method, provided by opencv. After computing these matrices, I have used **BruteForce** function to match the keypoints between pair of images, which can be seen in the result below obtained from *T1* dataset.



(a) (b)

Figure 1: (a)Features Matching first Two images are combined (b) Combined Images

We can see from results that numerous numbers of wrong keypoints matches are found by sift, to overcome this, I have to compute some good points that better represents the regions that are more close to each other, this is done by setting computing the distance between each points with respect to others. Then these good points are filtered again by placing threshold which is three times to **min_dist**, distance should be less than this threshold, otherwise if this is increased then more keypoints are detected.After this I found the minimum horizontal value for each image and each of them is cropped using **croppImg1** in the form of Rectangle.Then By know the orientation using **Direction**, it tells us that the image is aligned or not and if it is not aligned, then it should adjust

accordingly. After performing all the operation this function returns a single stitched image using variable name **result**.
By looking at the following figure, we can see that after combining first two images, the result of those images are again stitched with third image, by performing the same operations.

(a)

(b) (c)

(d)

Figure 2: (a)First three images are combined (b) Image 4 and 5 are combined (c) Image 6 (d) Result of 45 are combined with image 6

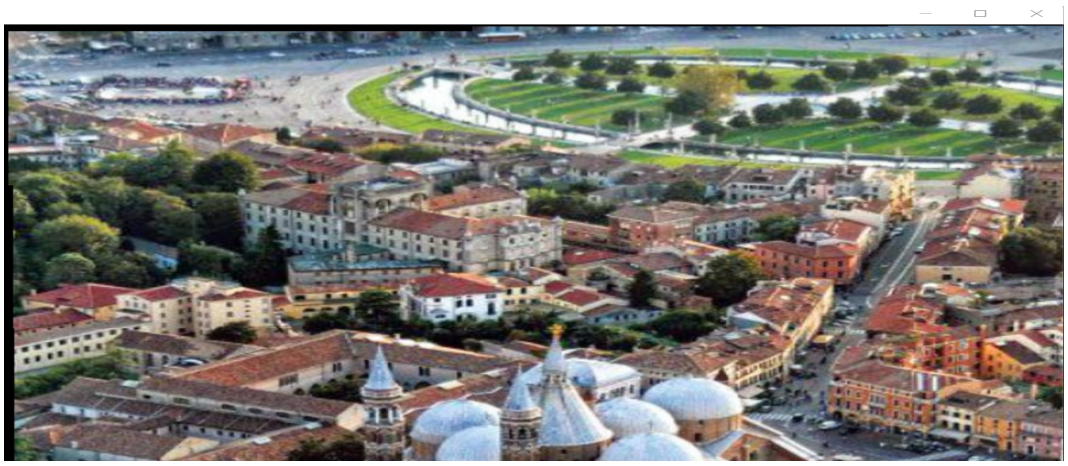
2.2 Affine_Rotation()

This function takes two parameters, **Image** and **Angle** and retruns the rotated image. Using **Point2f**, which make us applicable to select the point from where the image needs to rotate and with the help of **getRotationMatrix2D**, it allows to store the rotated images to apply affine transformation. This Affine transformation is carried out by using **warpAffine**,which is available in opencv library, which perform transformation on the rotated images to set the points accordingly. This can be seen in the following figures.

(a)

(b)

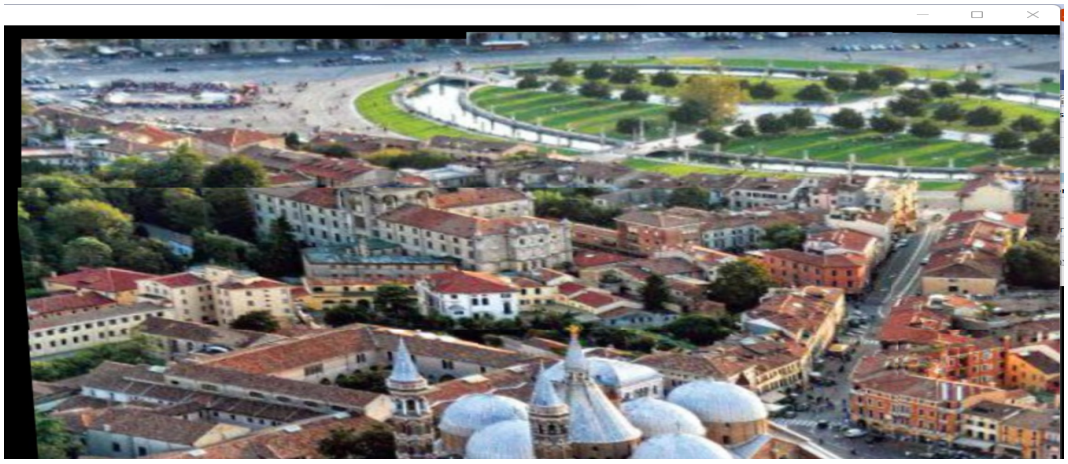
(c)



(d)

Figure 3: (a) Image 7&8 (b) Image 7&8 plus 9 (c) Combined and Rotated (d) Final Result

Regarding the dataset **RT1**, I applied the same algorithm, which performs the same steps, as mentioned above, and produces the output accordingly, it performs well also on *RT1* but shows some overlapping. The obtained result is shown in the figure below.



(a)

Figure 4: (a) Image

To sum up whole procedure, Initiating from inputting the pictures, the algorithm starts working with two images as input and produce a single stitched image, using this output, it takes another image detect feature points and descriptors, and combined third image with the previous output and generate another stitched image. This process is repeated until last image and provide the entire image as a result.

3 Resources

1. https://github.com/vineet0814/
2. https://opencv.org/
3. https://stackoverflow.com/