CMPE 537 – Assignment 2

Oğuzhan Sevim

December 13, 2020

INTRODUCTION

In this assignment, we implement an image stitching method in order to create a single panoramic image by computing homography and blending multiple corresponding pictures. The major details of my implementation can be summarized as follows:

- **Selecting corresponding point pairs:** This is done by manual selection of corresponding point pairs after printing 2 images side by side.
- **Normalization:** After choosing the corresponding pairs, chosen points in each image is normalized by scaling and translation. Corresponding transformation matrices are kept to be reused in the upcoming wrapping step.
- **Homography Estimation:** This step is implemented by computing applying singular value decomposition on a matrix formed by normalized corresponding point pairs. The homography matrix is obtained from the eigenvector that corresponds to the smaller singular value. As a result, we obtain homography matrix $\mathbf{H}$ that takes $image-1$ to the frame of $image-2$.
- **Image Wrapping:** After finding homography matrix $\mathbf{H}$ and normalization matrices $\mathbf{T}_1$ and $\mathbf{T}_2$, each pixel of $image-1$ is wrapped onto frame of $image-2$ by using transformation $\mathbf{T}_2^{-1}\mathbf{H}\mathbf{T}_1$.

  In order to calculate RGB values of non-integer pixel locations, first, the boundaries of each wrapped image is found. Then, a grid formed within that boundaries, followed by a back transform to the original frame. In the original frame, each non-integer pixel value is interpolated by using the mean RGB intensities of 4 closest pixel values.

  It should also be stressed that in each frame transformation, the resulting homogenous coordinates are scaled such that the last element is 1.
- **Image Blending:** After wrapping $image-1$ onto frame of $image-2$, a single pixel intensity of the blended image is calculated by taking pixel-wise maximum value of 2 images. This step is simply implemented by *numpy.maximum* function.

Important note for bonus points: In all the above steps (except interpolation), vectorized implementations are used without any for loop.

PART 1: PARIS IMAGES

In this part, we stitch 3 Paris scene images. From left to right this images are given as *paris_a*, *paris_b*, and *paris_c*. The stitching is done by taking *paris_b* as base. After *paris_a* and *paris_c* are stitched on *paris_b*, the final blended image is obtained by stitching resulting 2 images together. Figure 1 shows the final image when 10 corresponding points are chosen for each stitching.

How to read saved corresponding points: For this part, the chosen corresponding pairs are saved in *paris_a_to_b.npy* and *paris_c_to_b.npy* files. For *paris_a_to_b.npy*, indices $[:,:,0]$ gives the corresponding points in *paris_a* where $[:,:,1]$ contains points of *paris_b*.

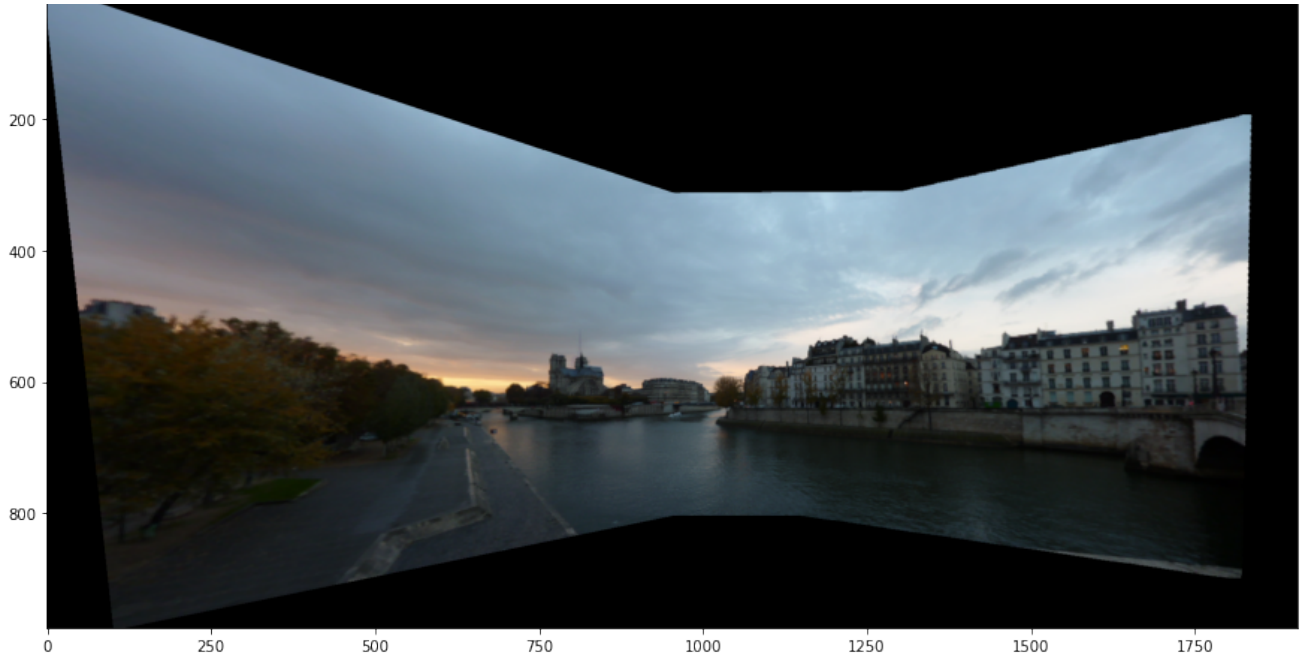


Figure 1: Result of Paris images when 10 corresponding point pairs are used.

EFFECTS OF CORRESPONDING POINTS

In this section, we have examined the effects of different corresponding points (number, selection, noise and normalization). In order to see these effects, we Paris image set is used. The results of different changes are given in the following sections.

Number of Pairs

In this part, we experiment with different number of corresponding point pairs. While 4 pairs are sufficient in theory, the more point pairs result in better image stitching. Figure 1 and 2 shows result of Paris images when number of pairs are chosen 10 and 5 respectively.

If we assume perfect selection, choosing any number (more than 4) of pairs don't make any difference. However, we make inevitable mistakes in manual selection process. When, the number of pairs is increased, the effects of these mistakes tend to be less systematic and somehow cancel each other. If we look at white buildings on the right hand side of stitched images, we see that stitching in Figure 1 is more superior than stitching in Figure 2.

Figure 2: Result of Paris images when 5 corresponding point pairs are used.

Point Selection

Choosing wrong point pairs results in wrong estimation of homography matrix. When we make 5 intentionally wrong choice (on top of 10 good pairs), a problem occurs in my wrapping function such that I can't generate any result. When I add only 2 wrong pairs on the 10 correct ones, a resulting image can sometimes be generated. With 2 wrong and 10 good point pairs, an example stitching of *paris_a* on *paris_b* is shown in Figure 3.

Please note that the wrong point pairs are generated randomly by discrete uniform distribution.

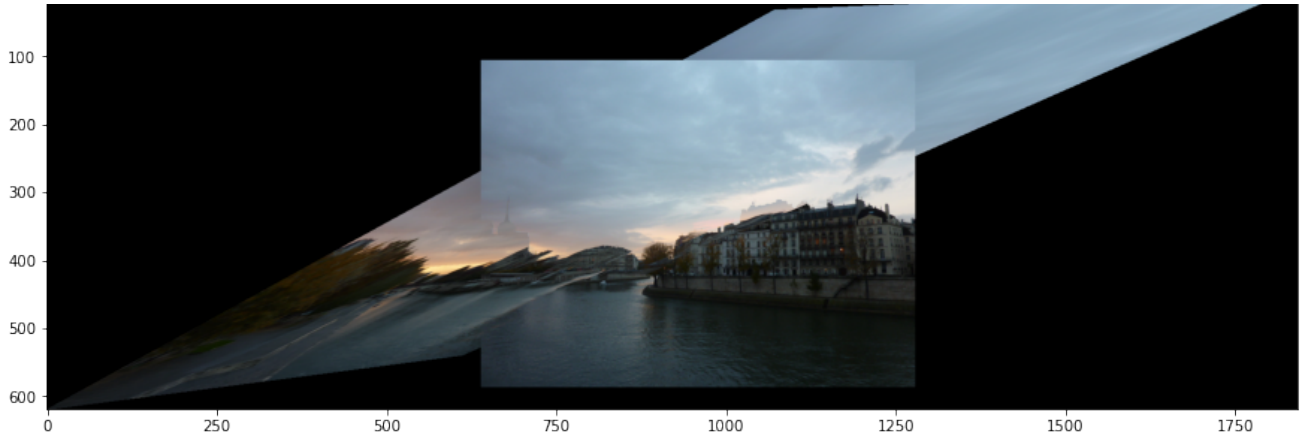
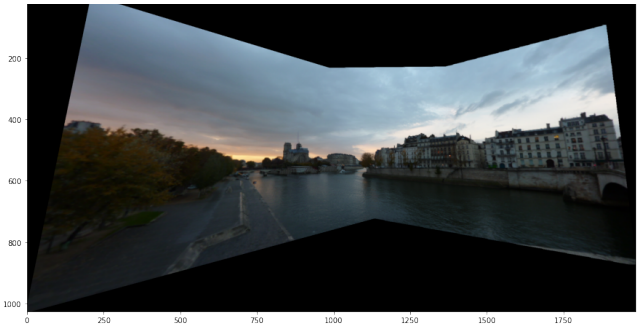


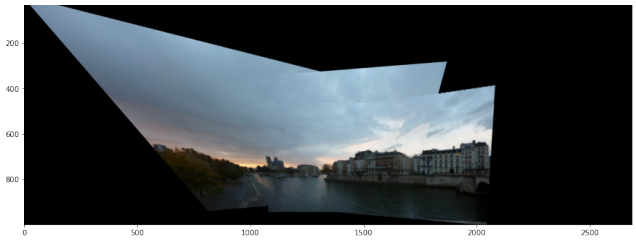
Figure 3: Result of Paris images when 10 good and 2 wrong corresponding point pairs are used.

Noisy Points

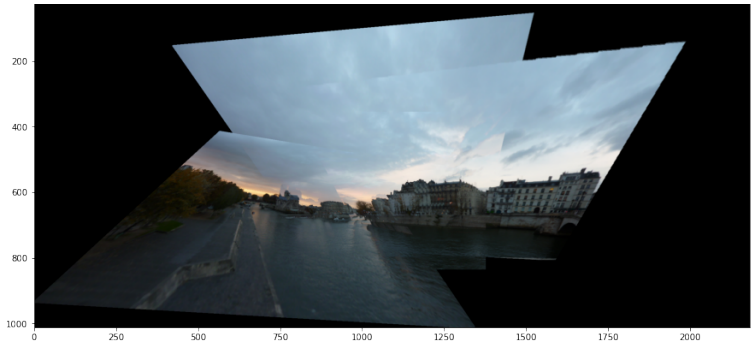
The effect of noise on the corresponding points is experimented by adding zero-mean Gaussian noise on top of each corresponding pairs. Following 3 variance values are used for this purpose: $\sigma^2 = 2$, $\sigma^2 = 5$, and $\sigma^2 = 8$. These results are illustrated in Figure 4. As the variance of Gaussian noise increases, the resulting image gets more distorted.



(a) $\sigma^2 = 2$



(b) $\sigma^2 = 5$



(c) $\sigma^2 = 8$

Figure 4: Stitching result when zero-mean Gaussian noise with different variances is added to pixel values of corresponding points.

Normalization

The effects of normalization is examined in this section. Figure 1 shows the stitching result when normalization is applied on 10 corresponding pairs. When the normalization is cancelled with the same corresponding point pairs, the result becomes as shown in Figure 5. There seems to be no eye-detectable difference between two results.

Figure 5: Result of Paris images when no normalization is applied on corresponding point pairs. The result may be compared with Figure 1 which has the same corresponding points.

PART 2: CMPE Building

In this part, we experiment on stitching 5 images of CMPE building of Bogazici University. In order to stitch 5 images, following 3 approaches have been implemented: left-to-right, middle-out, and first-out-then-middle. The results for these tasks (in the same order) are shown in Figures 6, 7, and 8.

Figure 6: Result of CMPE building images when left-to-right stitching is applied with 8 corresponding point pairs.

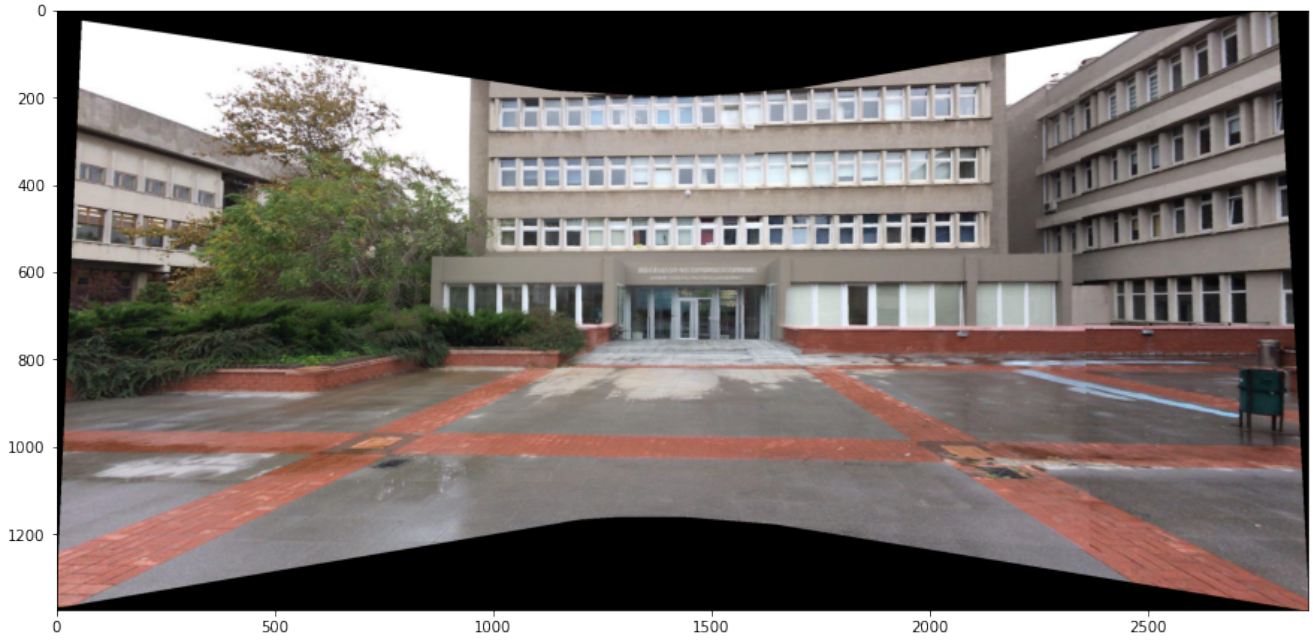


Figure 7: Result of CMPE building images when middle-out stitching is applied with 8 corresponding point pairs.

Figure 8: Result of CMPE building images when first-out-then-middle stitching is applied with 8 point pairs.

In CMPE bulding images, middle-out and first-out-then-middle stitchings gives almost the same results which are better than left-to-right method.

PART 3: North Campus

In this part, we have repeated PART 2 with another set of 5 images that is taken from north campus of Bogazici University. Unfortunately, I couldn't stitch the images by using left-to-right method. As shown in Figure 9, after stitching left_2, left_1, and middle, the blended image becomes too wide to work further on. The other stitching results are illustrated in Figures 10 and 11.

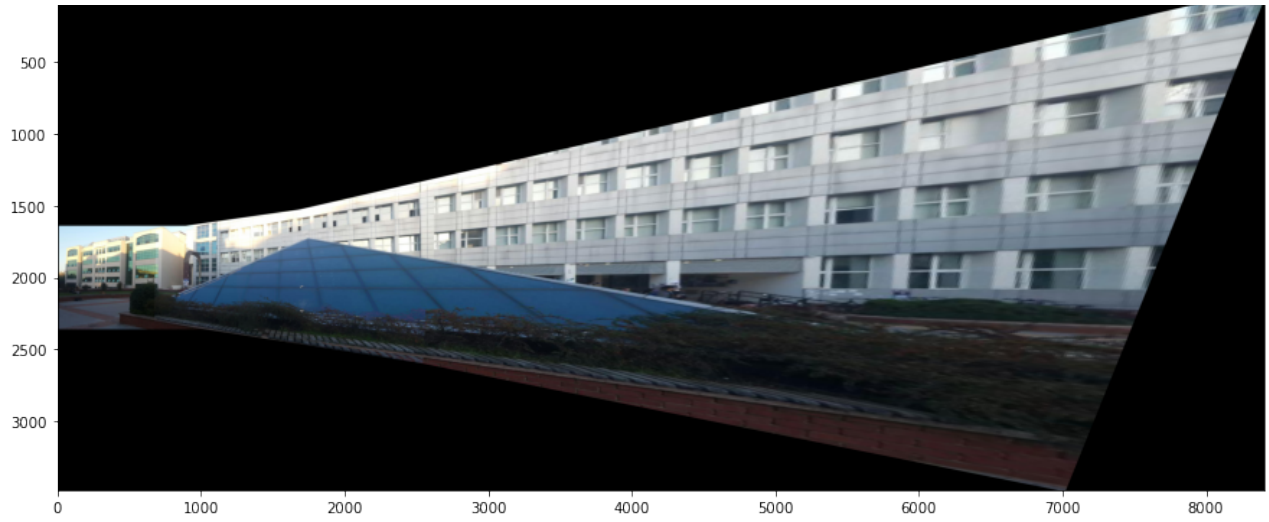


Figure 9: Result of North Campus images when left-to-right stitching is applied (FAILED after 3 images).

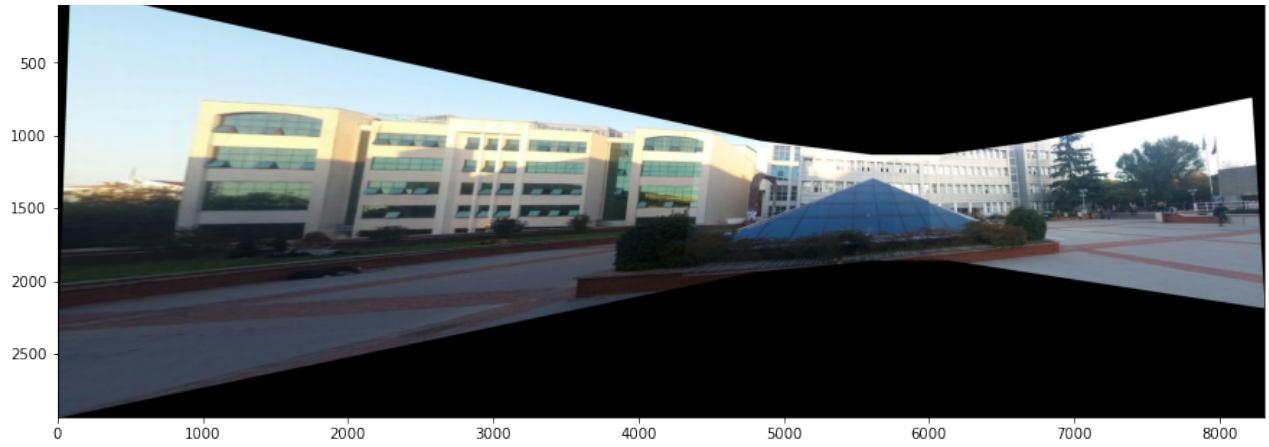


Figure 10: Result of North Campus images when middle-out stitching is applied with 8 corresponding point pairs.

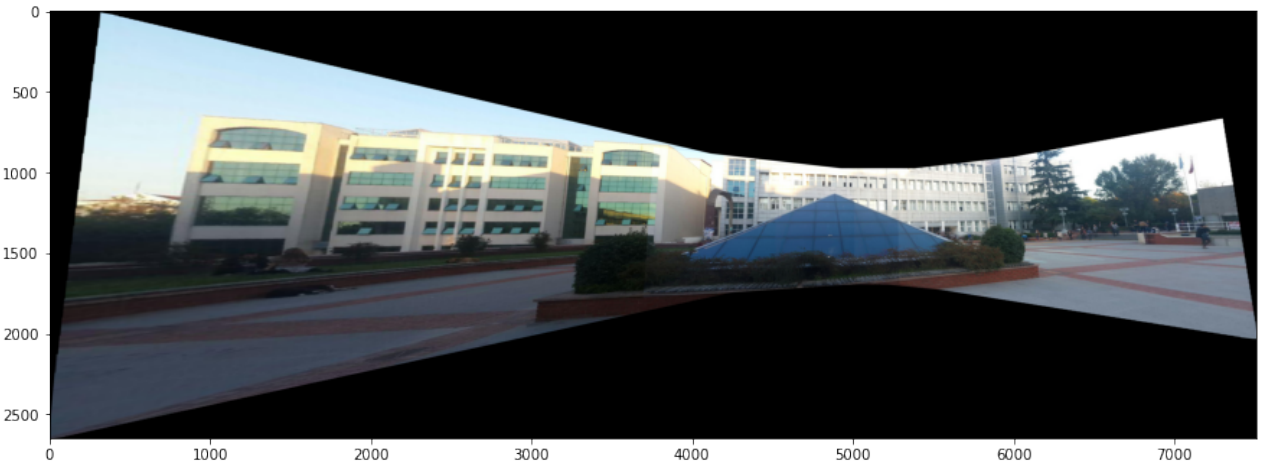


Figure 11: Result of North Campus images when first-out-then-middle stitching is applied with 8 point pairs.