HACETTEPE UNIVERSITY

Department of Computer Engineering

BBM415 Fundamentals of Image Processing Lab

Assignment – 3

Fall 2021-2022

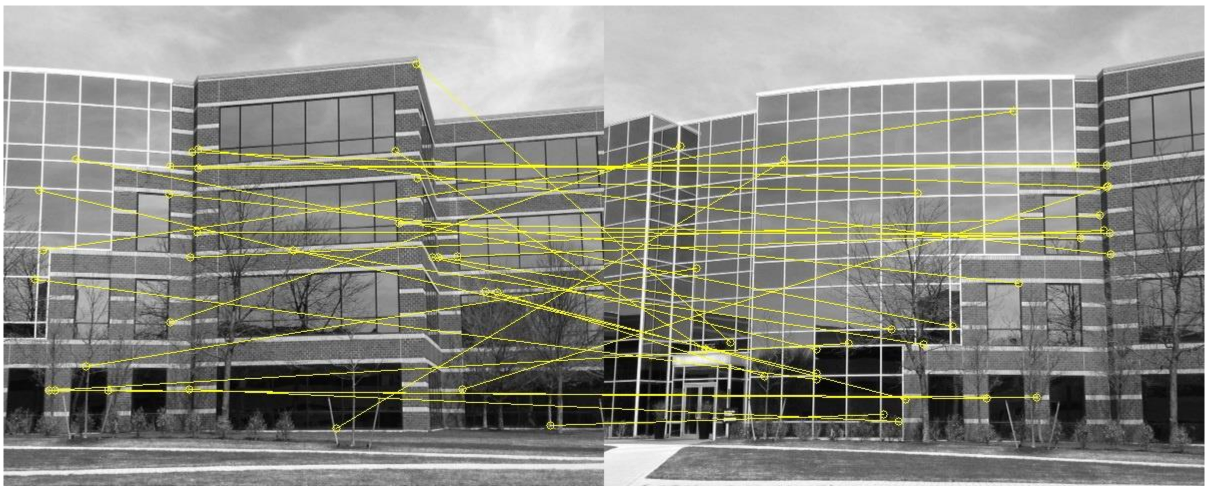
Due Date: 11:59 pm, 2022/01/01 Saturday

Student Name:	Student Number:
Muhammed Seyfullah Bilgin	21990322

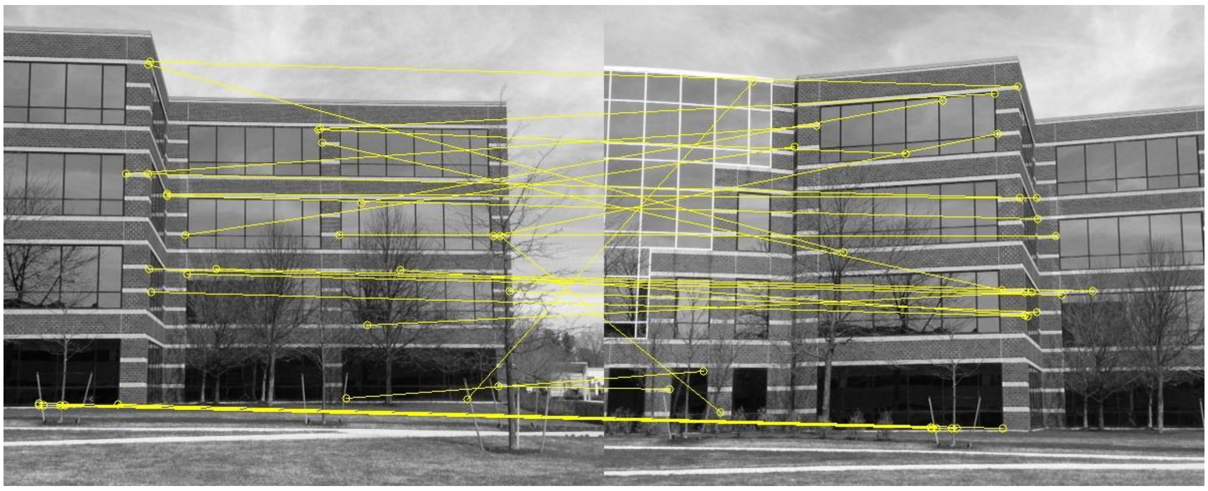
In this assignment, our goal is to generate a panoramic image by stitching multiple images. We will implement stitching to generate panoramic images. We assume images will have same view and there will be a listed images given in an order (left to right).

I have used SIFT feature to detect keypoints and extract local invariant descriptors

Middle - Left

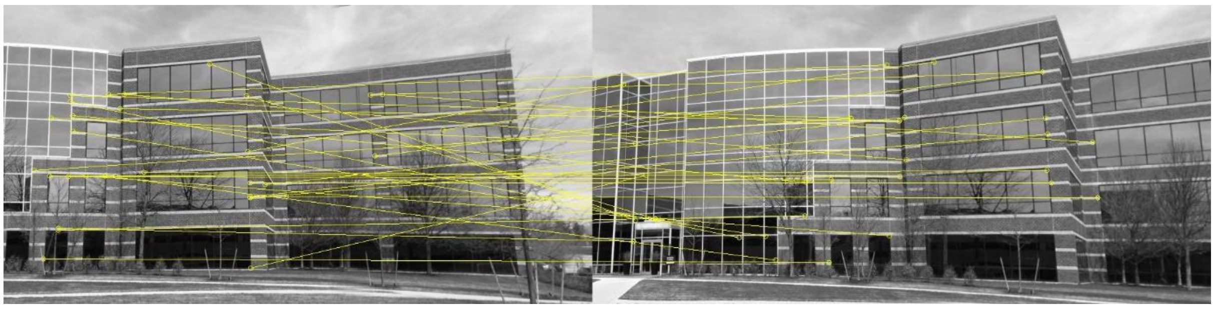


Right – Middle



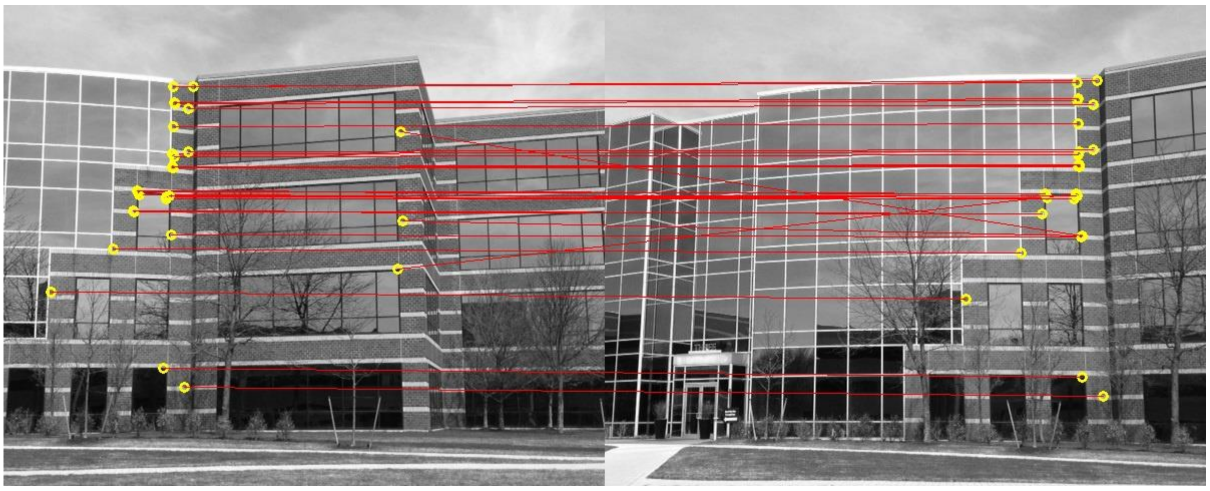
Right Part – Left Part

Right ∪ Middle – Middle ∪ Left

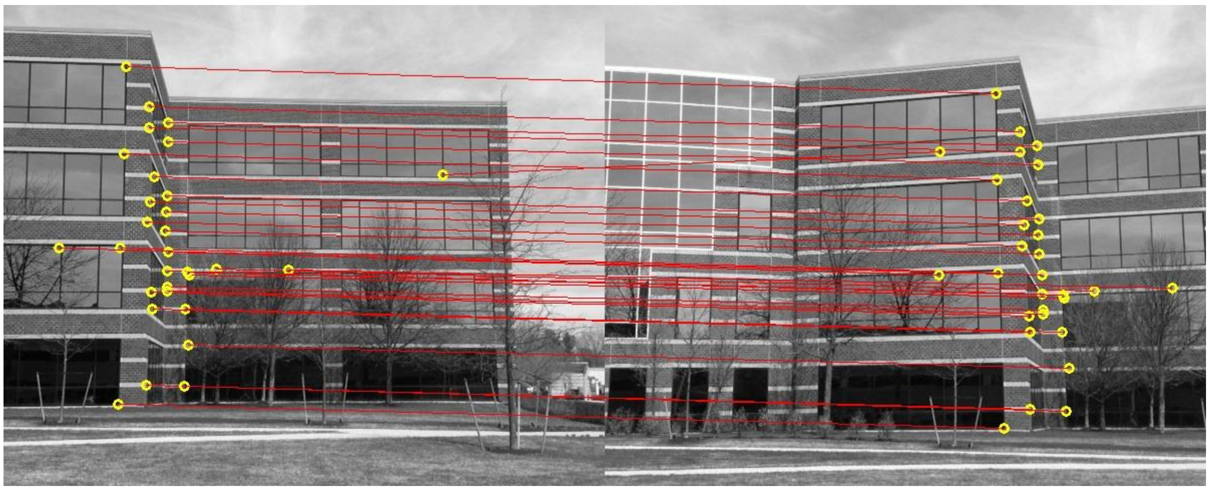


As seen as in the pictures, there are many line that is wrongly aligned. However, if we eliminate weaker ones, we will get strongest ones the following pictures.

Middle - Left

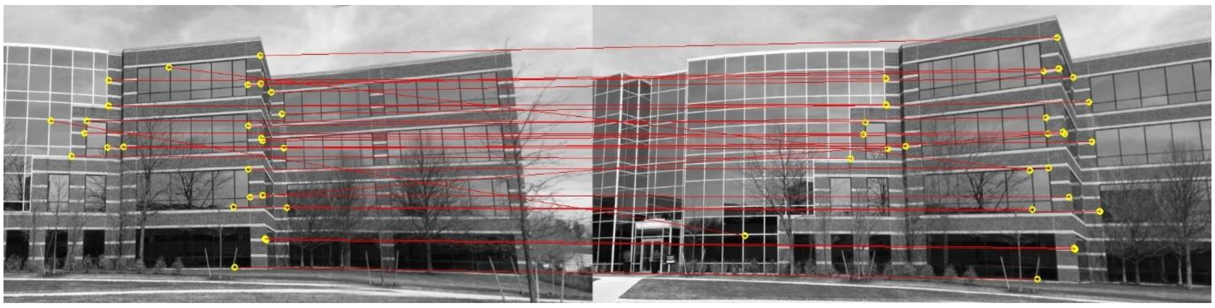


Right – Middle



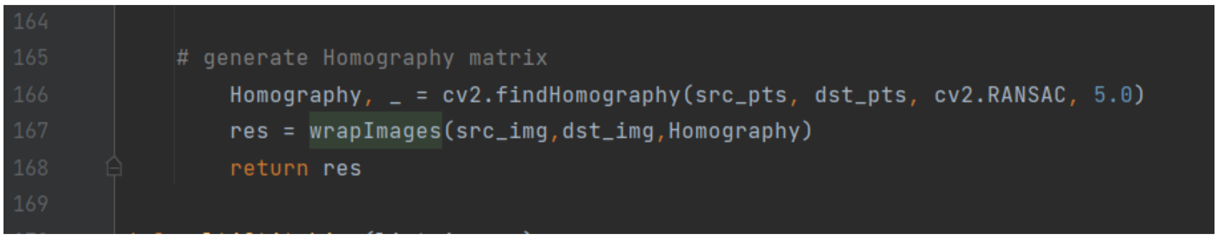
Right Part – Left Part

Right ∪ Middle – Middle ∪ Left



As a result, we have matched the images one by one thanks to SIFT method. I have used brute force algorithm to match keypoints. There can be used different matching algorithms.

I have used Ransac algorithm to estimate a homography matrix by using matched images. Ransac is an iterative method to estimate parameters of a mathematical model to detect same points in different clusters.

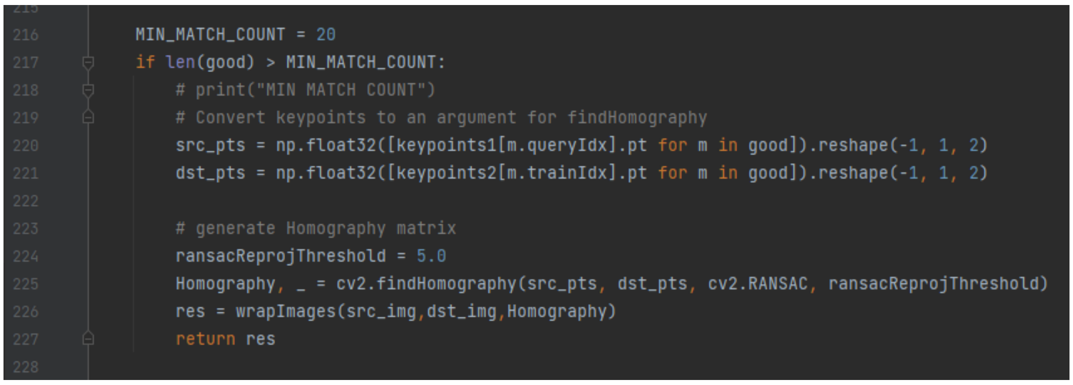


```
164
165         # generate Homography matrix
166             Homography, _ = cv2.findHomography(src_pts, dst_pts, cv2.RANSAC, 5.0)
167             res = wrapImages(src_img,dst_img,Homography)
168             return res
169
```

Thanks to ransac algorithm, homography matrix is created in line 166. We will use homography matrix to merge images.

When merging images, I have used image blending. Image blending makes a copy of an image and transfer each source pixel’s values into a pixel in the target image. Another important issue is borders of two images in overlapped parts. I try to blur and soften these parts inside blending function. There was another method as known as feathering or center weighting for image blending. It s basically applys weighted average of two overlapping images. If the quality of images is low, this method may not be suitable. However, I think it produces excellent outputs for our input images.

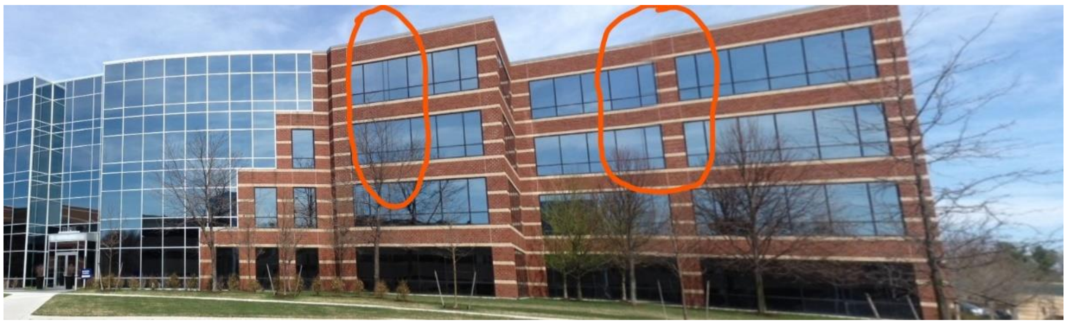
To improve the results, most important thing is taken images. High quality images and images have larger overlapped areas can produce better results due to number of keypoints. If we are not able to change inputs, we have to detect significant keypoints. I think, the most importing thing is to find better keypoints and blending with appropriate blurring method.

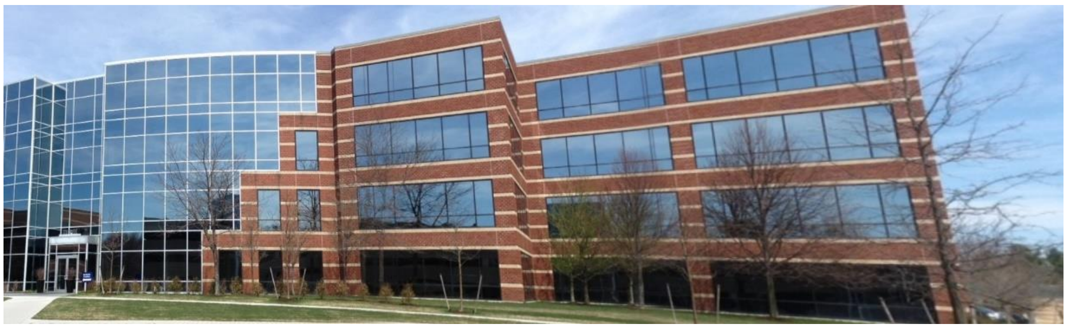


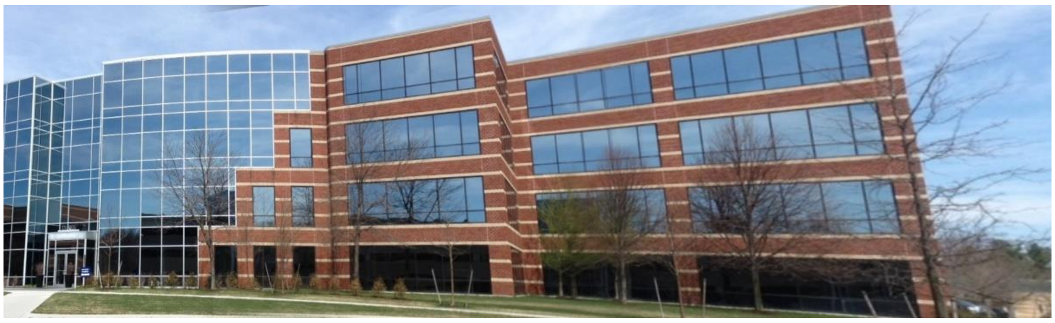
```
    MIN_MATCH_COUNT = 20
    if len(good) > MIN_MATCH_COUNT:
        # print("MIN MATCH COUNT")
        # Convert keypoints to an argument for findHomography
        src_pts = np.float32([keypoints1[m.queryIdx].pt for m in good]).reshape(-1, 1, 2)
        dst_pts = np.float32([keypoints2[m.trainIdx].pt for m in good]).reshape(-1, 1, 2)

        # generate Homography matrix
        ransacReprojThreshold = 5.0
        Homography, _ = cv2.findHomography(src_pts, dst_pts, cv2.RANSAC, ransacReprojThreshold)
        res = wrapImages(src_img,dst_img,Homography)
        return res
```

To find better keypoints, I have observed *limit, MIN_MATCH_COUNT and ransacReprojThreshold* values. I tried to find the best values by distinguishing between their outputs. According to the input images, these values can be changeable.

I have changed bluring parameter to reach best output. These are some of parameters I tried.


P = 16


P = 8


P = 2

I cropped black regions after getting wrapped images. I think, the last one is better because overlapped parts can noticeable in the first two images. Last image seems more natural.

References

1- https://becominghuman.ai/image-blending-using-laplacian-pyramids-2f8e9982077f
2- https://towardsdatascience.com/image-panorama-stitching-with-opencv-2402bde6b46c
3- https://www.pyimagesearch.com/2016/01/11/opencv-panorama-stitching/
4- https://en.wikipedia.org/wiki/Random_sample_consensus
5- http://graphics.cs.cmu.edu/courses/15-463/2010_spring/Lectures/blending.pdf
6- https://medium.com/featurepreneur/blending-images-using-opencv-bfc9ab3697b7