Computer Vision

Assignment 2

Abdallah Yasser Ibrahim - 18015026

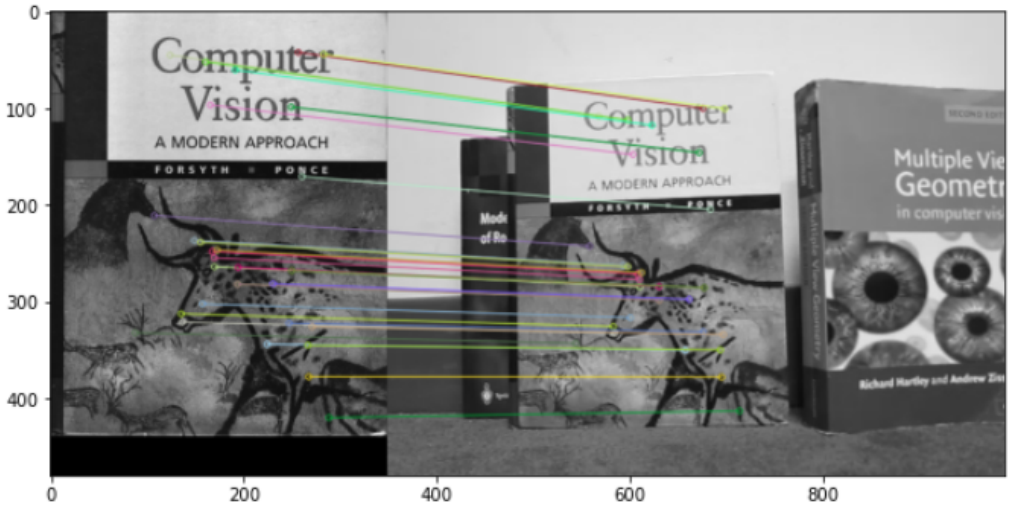
Ahmed Bahgat Elsherif - 18010078

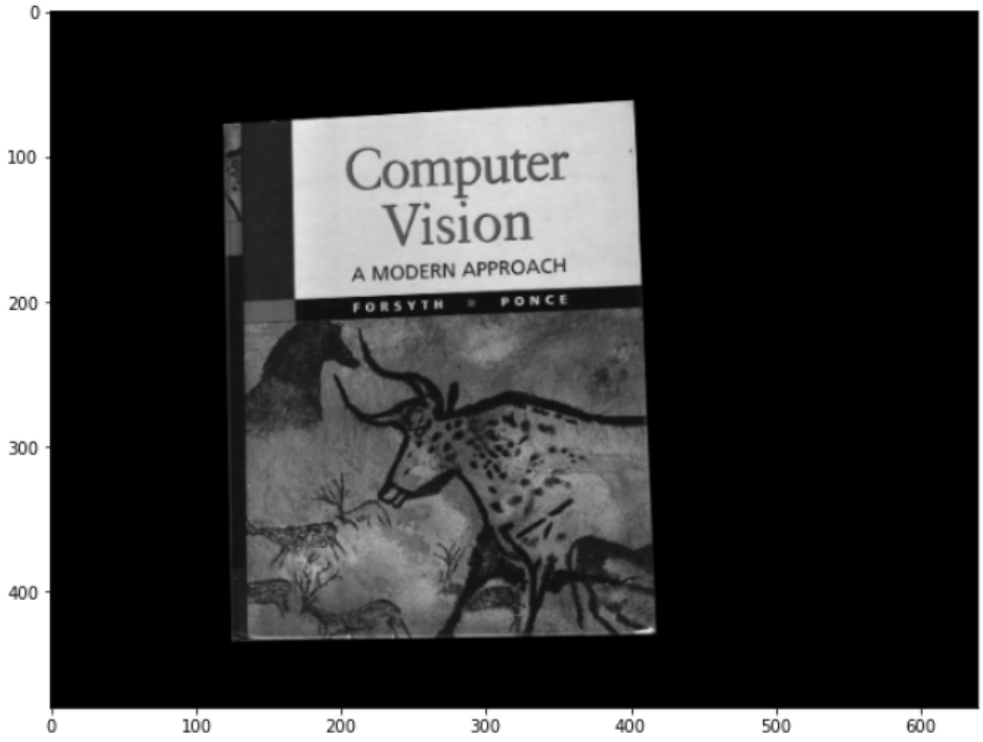
Mohamed Metwalli Noureldin - 18011587

Part 1: Augmented Reality with Planar Homographies

Getting Correspondences and Compute the Homography Paramete

The first stip is to find keypoints in each image using *SIFT* descriptor and then use the brute force matcher to get the correspondences.





Compute the Homography Parameters

In the next step, we wrote a function that takes a set of corresponding image points and computes the associated (3*3) homography matrix *H*. This matrix transforms any point *p* in one view to its corresponding homogeneous coordinates in the second view, *p'*, such that p' = Hp.

Calculate the Book Coordinates

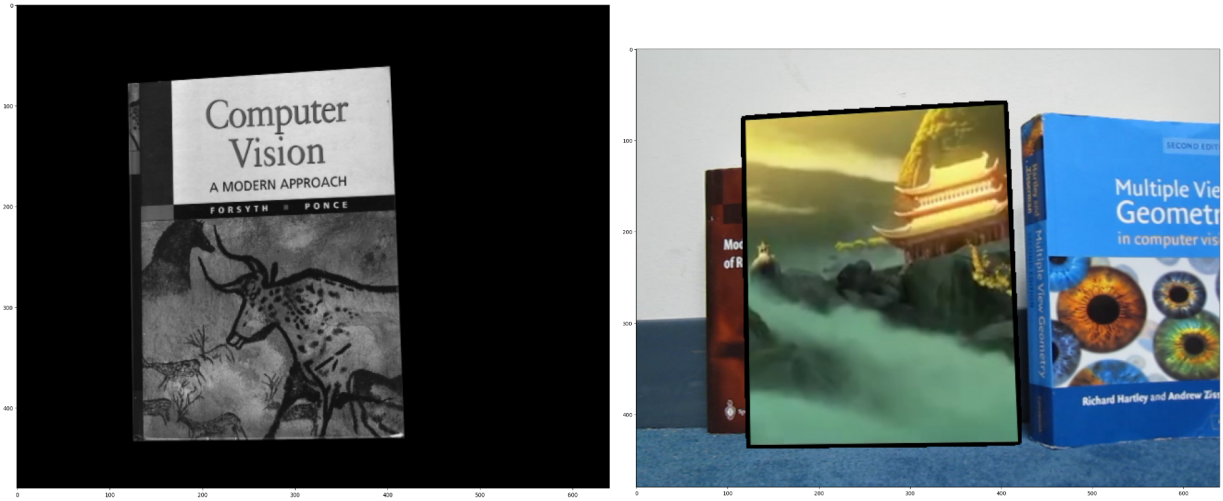
Then, we detected the four corners of the book in the video. This was done by mapping the four corners of the book image (cover) to the first frame in the book video using the previously calculated homography matrix.

Crop AR Video Frame

Then, we cropped each frame such that only its central region is used in the final output.

Overlay the First Frame of the Two Videos

Next, we replace the computer vision book in the first video frame with the first cropped frame of the movie video to make the AR Effect. .



Creating AR Application

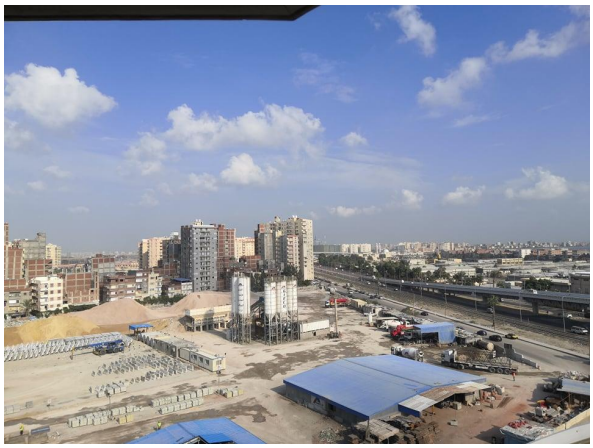
Finally, we get the new location of the book in the following frame to overlay each cropped video frame to its corresponding frame of the book video, and repeating the cropping step again.

The Video Link :

https://drive.google.com/file/d/1sJyI6admDmY8-6v_ADvTZamH5QGPMa9O/view?usp=share_link

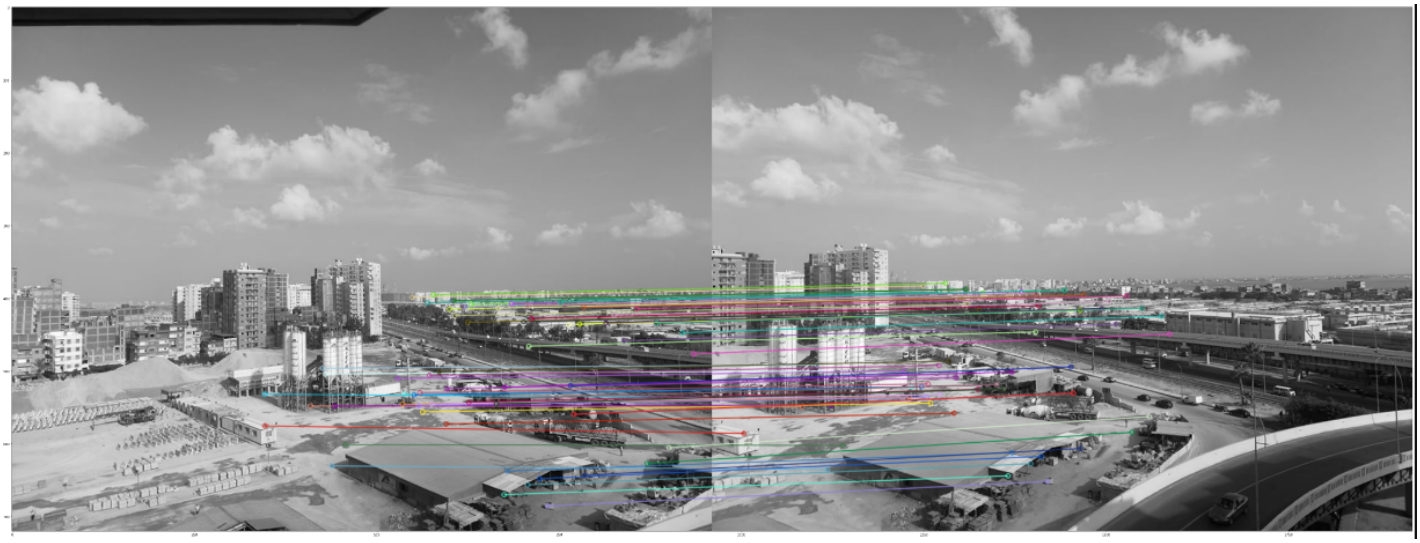
Part 2: Image Mosaics

We used these two images to apply the image stetching using image warping and homographies to automatically create an image mosaic.




Getting Correspondences and Compute the Homography Paramete

The first stip is to find keypoints in each image using *SIFT* descriptor and then use the brute force matcher to get the correspondences.

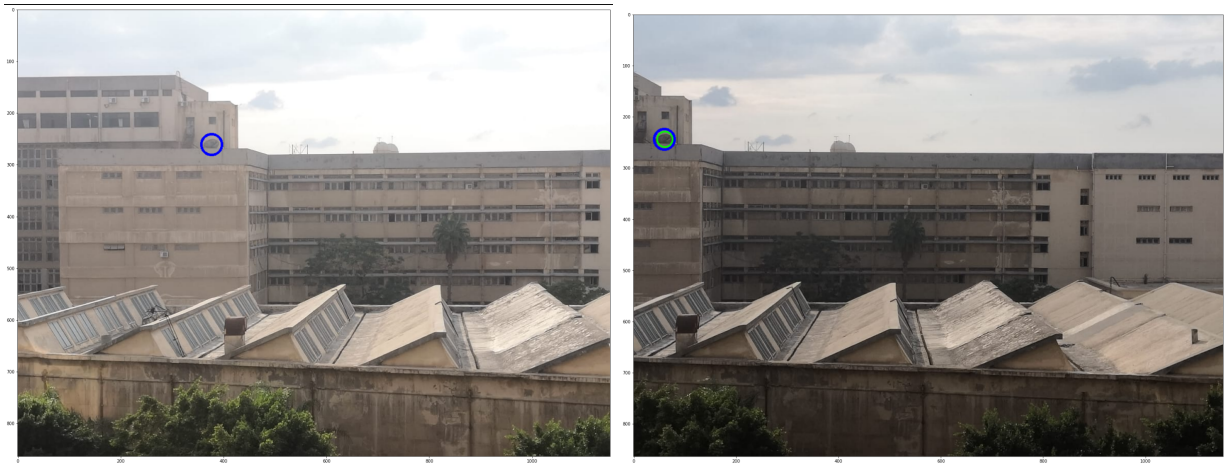


Compute the Homography Parameters

In the next step, we wrote a function that takes a set of corresponding image points and computes the associated (3*3) homography matrix *H*. This matrix transforms any point *p* in one view to its corresponding homogeneous coordinates in the second view, *p'*, such that p' = Hp.

Test the Homography Matrix

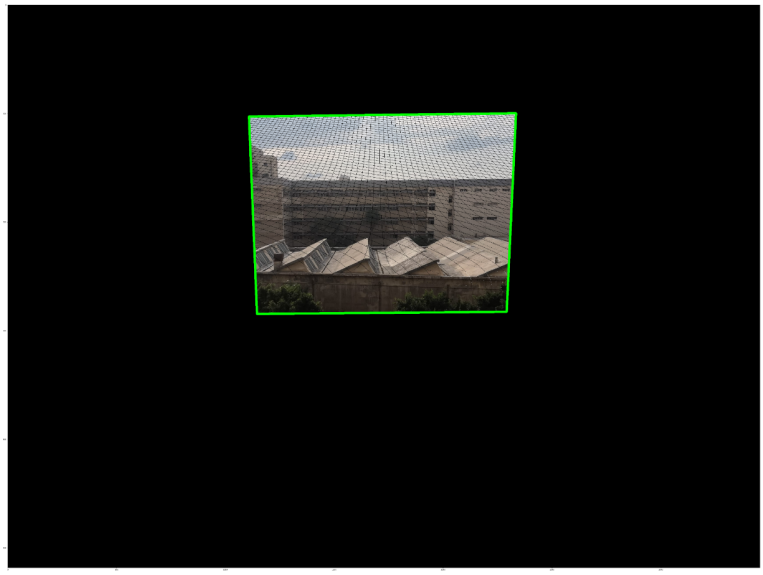
Then, we verified that the homography matrix we compute is correct by mapping the clicked image points from one view to the other, and displaying them on top of each respective image.



Warping Between Image Planes

Next, we wrote a function that can take the recovered homography matrix and an image, and return a new image that is the warp of the input image using *H.* We implemented the forward warping, and the inverse warping.

To avoid holes in the output, we used the inverse warp to warp the points from the source image into the reference frame of the destination, and computing the bounding box in that new reference frame. Then we sampled all points in that destination bounding box from the proper coordinates in the source image (linear interpolation).



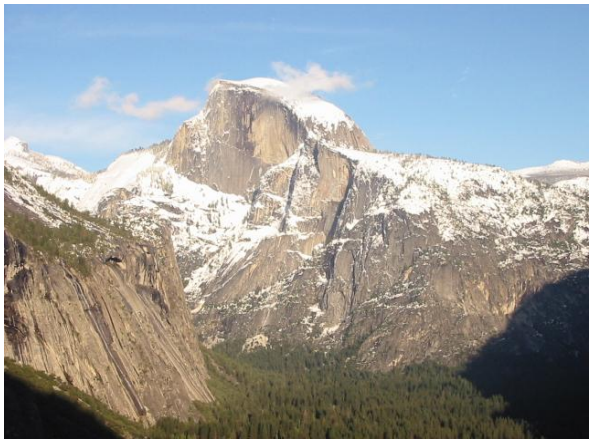
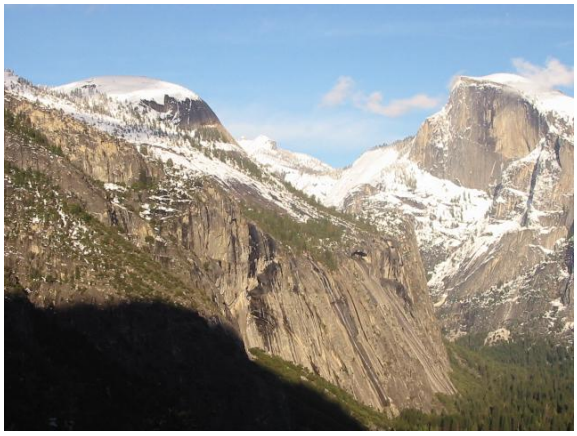
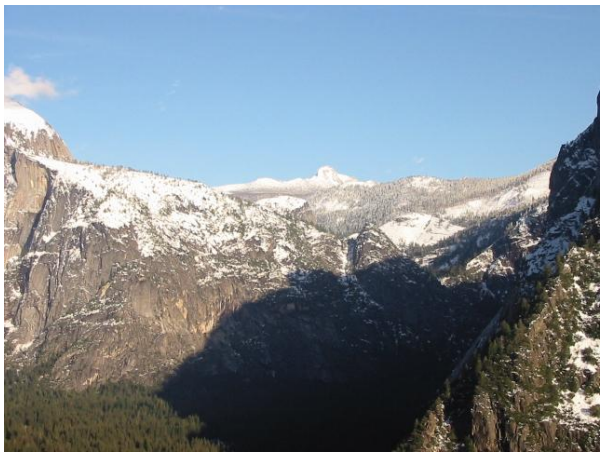
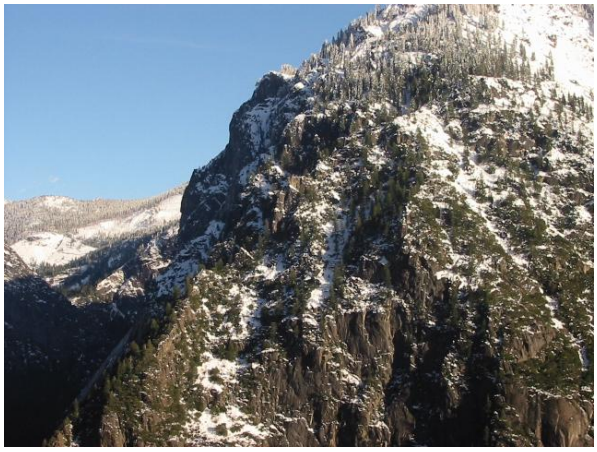
Create the Output Mosaic

Finally, Once we have the source image warped into the destination images frame of reference, we created a merged image showing the mosaic. We created a new image large enough to hold both registered views; then, overlaid one view onto the other, and left it black wherever no data is available.

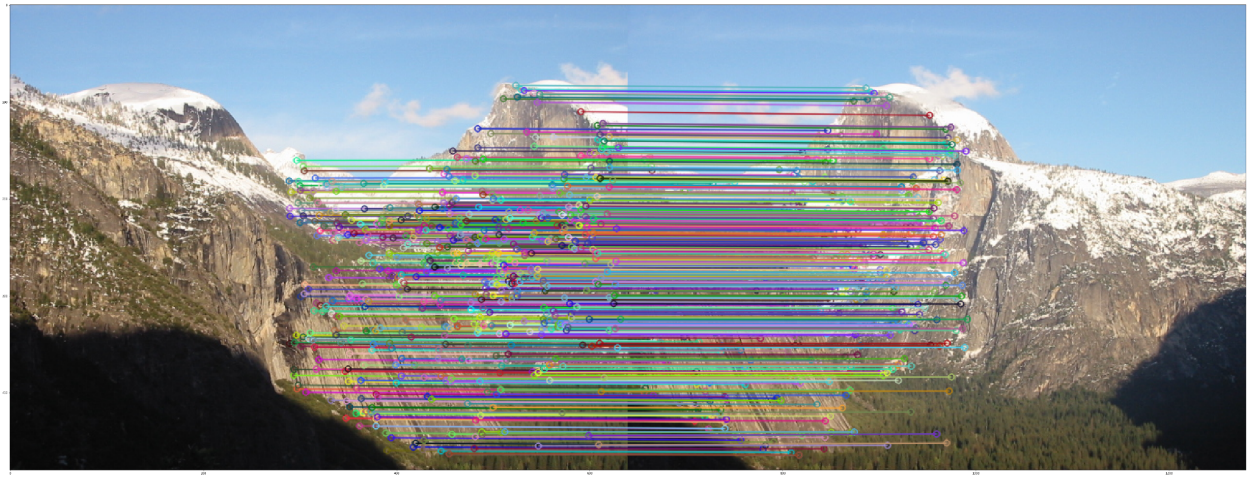
Here is the result :

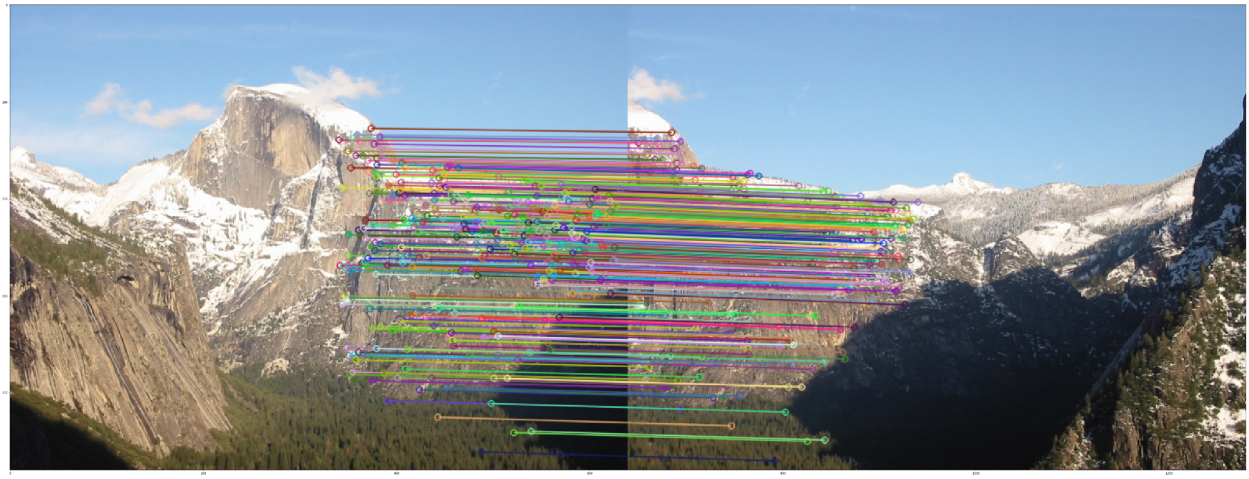
The Bonus Part :

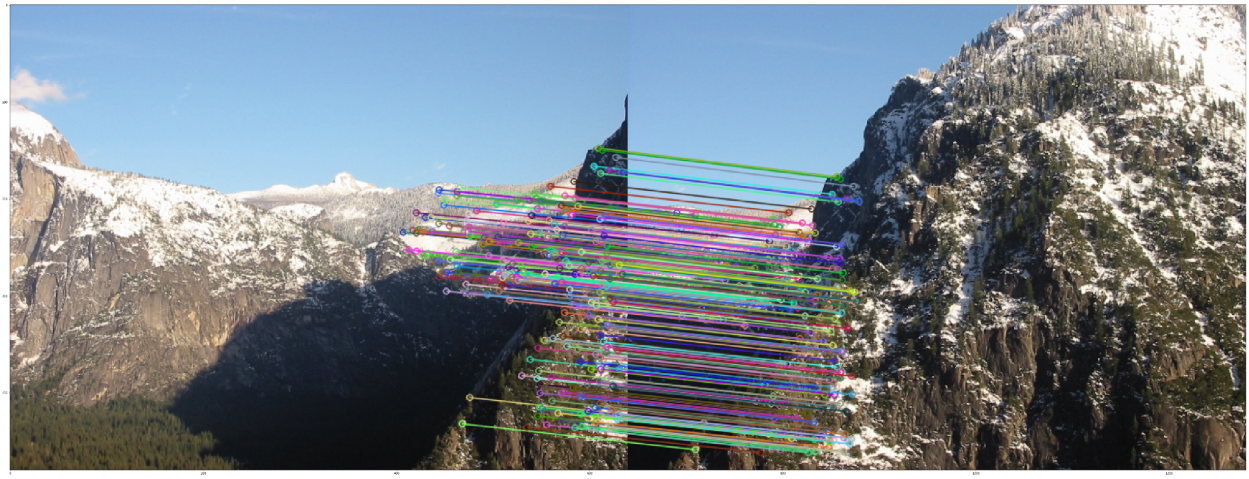
We used these four images to apply the image stetching.

Stitching every 2 images together :







Stitching the output with the third image creating the final output:

The result