AMAN BHARDWAJ
2019SIY7580

COL780 Computer Vision: Assignment 2

IMAGE MOSAIC CREATION.

CONTENT

I. INTRODUCTION [1]

In the field of ***photographic imaging***, a photographic mosaic, also known under the term *Photomosaic* (a portmanteau of photo and mosaic), is a picture (usually a photograph) that has been divided into (usually equal sized) tiled sections, each of which is replaced with another photograph that matches the target photo. When viewed at low magnifications, the individual pixels appear as the primary image, while close examination reveals that the image is in fact made up of many hundreds or *thousands of smaller images*. Most of the time they are a computer-created type of montage.

There are *two* kinds of mosaic, depending on how the matching is done. In the simpler kind, each part of the target image is averaged down to a single colour. Each of the library images is also reduced to a single colour. Each part of the target image is then replaced with one from the library where these colours are as similar as possible. In effect, the target image is reduced in resolution (by down-sampling), and then each of the resulting pixels is replaced with an image whose average colour matches that pixel.

In the more advanced kind of photographic mosaic, the target image is not down-sampled, and the matching is done by comparing each pixel in the rectangle to the corresponding pixel from each library image. The rectangle in the target is then replaced with the library image that minimizes the total difference. This requires much more computation than the simple kind, but the results can be much better since the pixel-by-pixel matching can preserve the resolution of the target image.

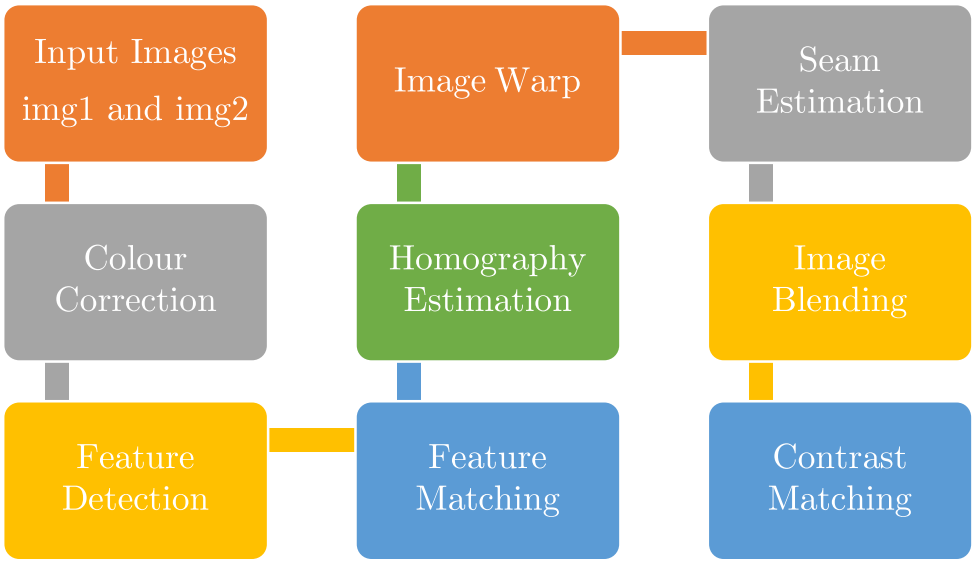


Figure: General *high-level* pipeline for creating Image Mosaics

II. METHODOLOGY EXPLAINED

Steps-by-step process followed explained with images:

1. **Import Image:** From the input folder 2 images imported in their original **RGB format** and named **img1** and **img2** respectively.


Fig: Img1 and Img2 imported.

2. **Color Correction:** There are times when images taken are **badly lit or too bright.** Also, they may have bad contrasts. So, feature detection and extraction become very difficult for these images. Therefore, they need to be corrected.
   This was done by using **CLAHE [2] (**Contrast Limited Adaptive Histogram Equalization) using *OpenCV* Library. It takes care of over-amplification of the contrast. CLAHE operates on small regions in the image, called tiles, rather than the entire image. The neighbouring tiles are then combined using bilinear interpolation to remove the artificial boundaries. This algorithm can be applied to improve the contrast of images.

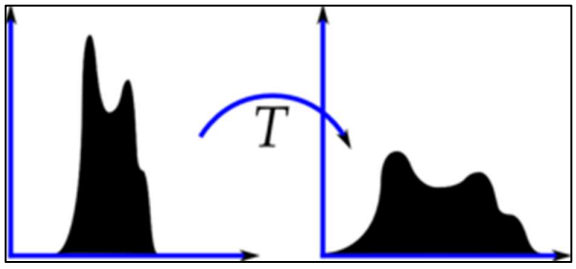


Fig: Histogram Equalization


Fig: Color corrected input images by CLAHE

3. Key point detection via Feature Descriptor:

In order to match the images, we need to extract features such that they are **invariant of scale and transformation operations** so that the features from both the images should be common irrespective of the scale and transformational differences between the images.

I have used **SIFT [3]** for this purpose. SIFT keypoints of objects are first extracted from a set of reference images and stored in a database. An object is recognized in a new image by individually comparing each feature from the new image to this database and finding candidate matching features based on Euclidean distance of their feature vectors.

By using SIFT I have extracted **a maximum of 500 keypoints** from each image in order to improve upon the computational time. Anyway, for Homography detection a minimum of only 4 good inlier points are required. (Figure shown along with step 4)

NOTE: Images should be converted to **grayscale** before passing into feature descriptor.

4. Feature Matching:

Once we have extracted all the features from both the images. We need to perform match of features between the two images. This could be done by various methods for example **Euclidian Distance Calculation, K-Nearest Neighbours** etc. I have used here **FLANN** (Fast Library for Approximate Nearest Neighbours) based matching because it is faster than L2 Norm Matching and KNN Matching algorithms and provide enough accuracy for our task in hand.

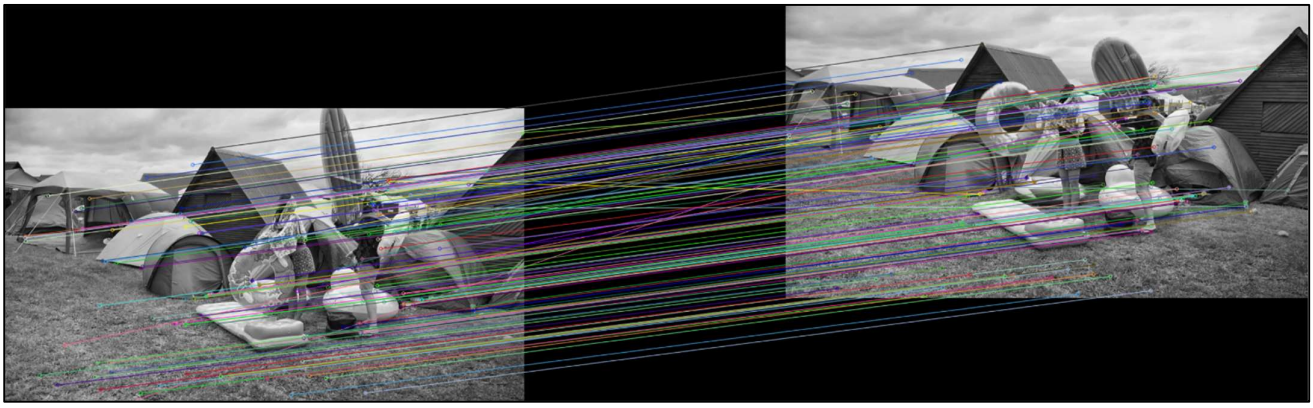


Fig: Visualization of Keypoints detected by SIFT and Features matched between two images by FLANN based Matcher.

5. Estimate Homography ($H_{(3x3)}$ matrix):

Before we blend the images, we need to identify the **Homography [4]** between the two images. It is a transformation (3×3 matrix) that maps the points in one image to the corresponding points in the other image.

But finding Homography is prone to outliers (non-parallel lines in above images) i.e. the features which got matched with a wrong feature of image 2. So, we need an algorithm which is robust to outliers of the image.

I have used **RanSaC [5]** (Random sample consensus) for H estimation.

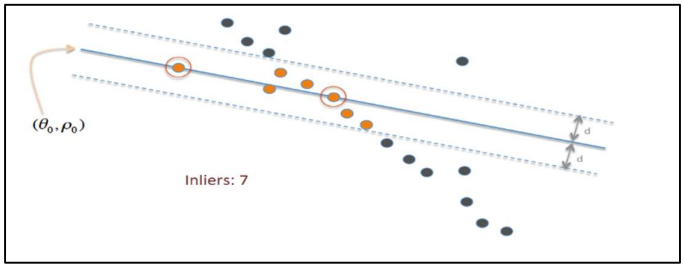


Fig: RanSaC visualization along with inliers and outliers.

$$\text{Estimated H} = \begin{matrix} +0.932 & -0.080 & +256.7 \\ -0.021 & +0.954 & +207.3 \\ -5e^{-05} & -1e^{-05} & 1.000 \end{matrix}$$

6. Perspective Warp:

Based on the H calculated, do a perspective warp of img2 keeping img1 as same in order to align img2 in same orientation as img1 reversing the effect of scaling and other transformations.

Fig: Perspective warp of Img2 based on H Martix in order to align features along the Img1.

7. Mask Creation:

Calculate binary masks for Img1, Img2 and Common Overlap Region of both the images.

Fig: Binary Masks of Img1 Img2(warped) and mask of the images overlap region

8. Energy Map creation:

Both the images are converted to grayscale in order to get the intensity of all the pixels and then the difference in the intensity is calculated between both the images. This makes the region where high frequency objects are there in the images as high intensity and rest low frequency portions becomes darker.

This signifies that seam can be carved through darker regions so that it doesn't cut any objects. So, ghosting effect can be avoided.

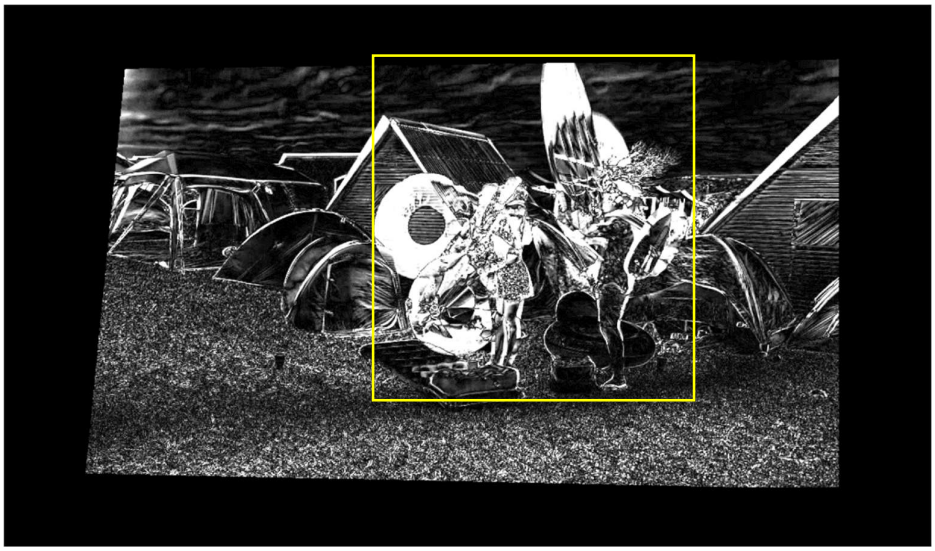


Fig: Energy map of both the mosaic. In the yellow box ghosting effect can be seen.

9. Graph Cut [6]:

Graph cut seam estimation is performed in order to find a seam which passes through low intensity regions so that **a minimal/no seam** is visible. Also, the seam passing through objects can be avoided. This prevents ghosting effect.

It is calculated by **Max-flow Min-cut algorithm**. I have used **PyMaxFlow** Library for the same.

Fig: (left)Min-Cut Mask for the overlap region of images. (middle) Min cut mask applied to the Img1 and the inverted min-cut mask applied to the Img2.

10. Image Blending:

This could be done by various methods.

I have used **direct blending, graph cut blend and pyramid blending.**

Along the seam estimated by graph cut. Blend the overlapping regions based on the left and right overlap. And keep rest of the image mosaic as it is. The difference between direct blending and Graph cut blend can be clearly seen in below images.

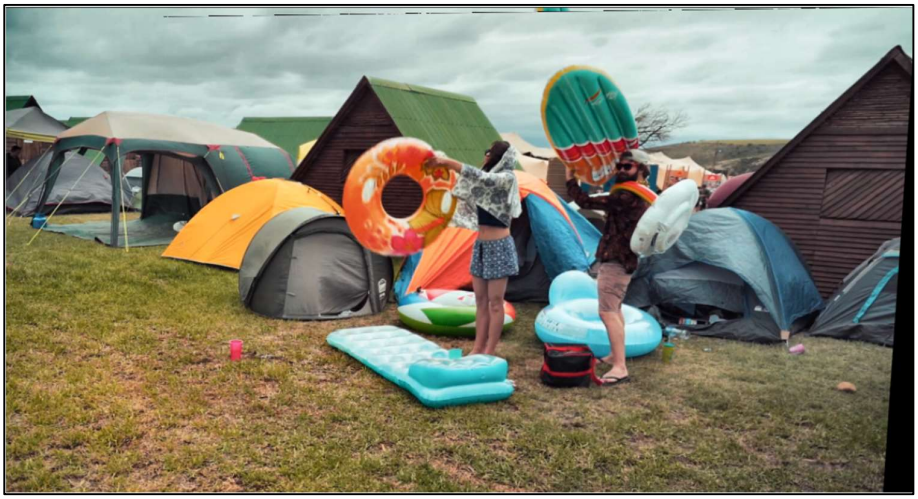


Fig: Image Blended after graph cut seam estimation.

11.Pyramid Blending [7] (Bonus):

It is used to create **seamless blending** of images. It provides a smooth transition of pixel intensities across the seam.

Observation: But after performing pyramid blending the image becomes **slightly blurred & brighter.**

Fig: As you can see that the seam has almost disappeared, or very subtle seam is visible after pyramid blending.

III. RESULTS: Validation Set

1: Images

1: Graph Cut

1: Post Pyramid Blending

2: Images

2: Graph Cut

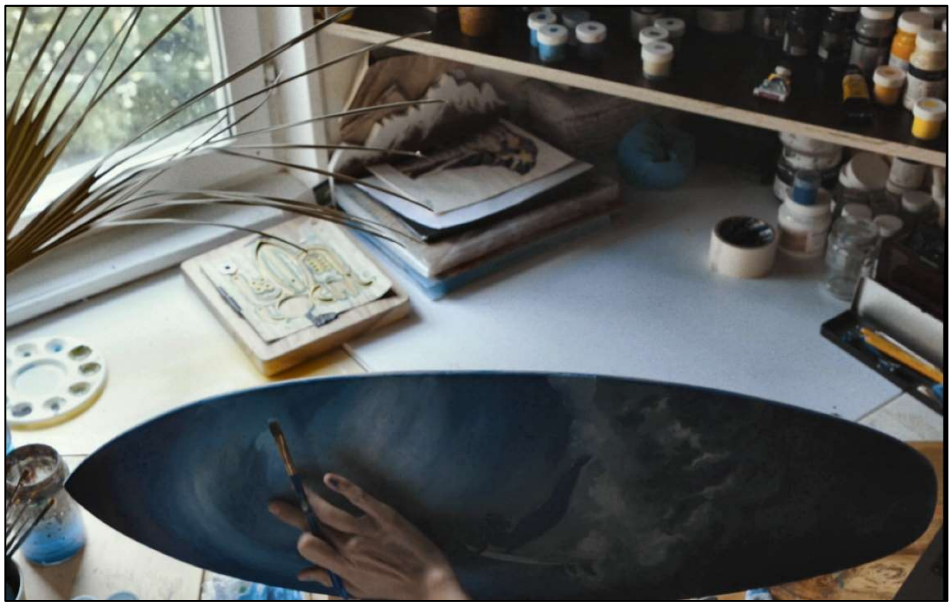
2: Post Pyramid Blending

3: Images

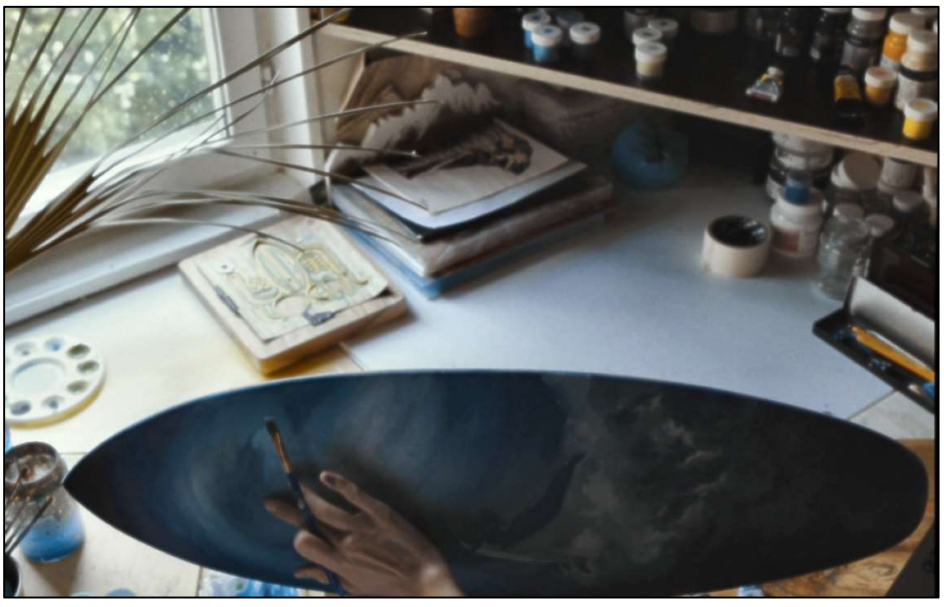

3: Graph Cut (left). Post Pyramid Blending (right)


4: Images

4: Graph Cut



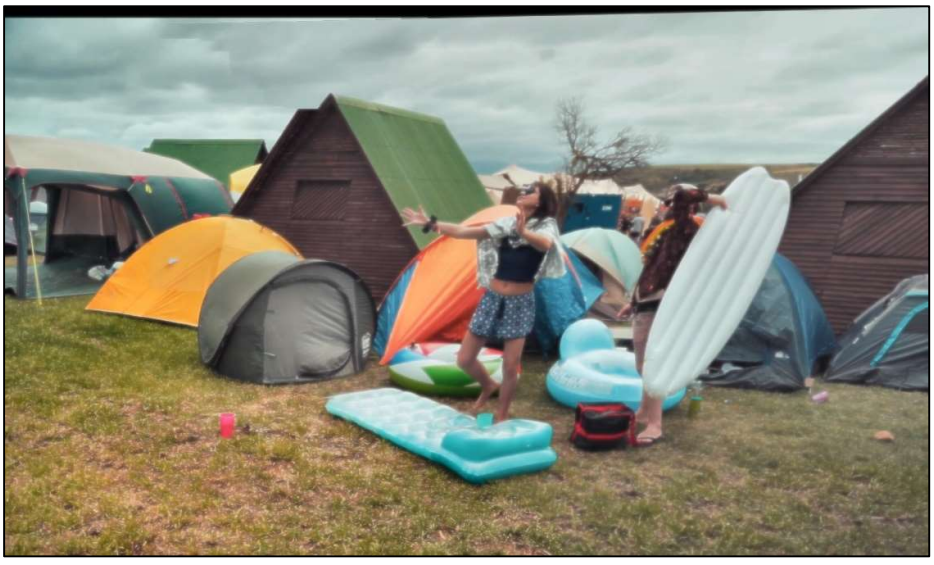
4: Post Pyramid Blending



5: Images

5: Graph Cut

5: Post Pyramid Blending



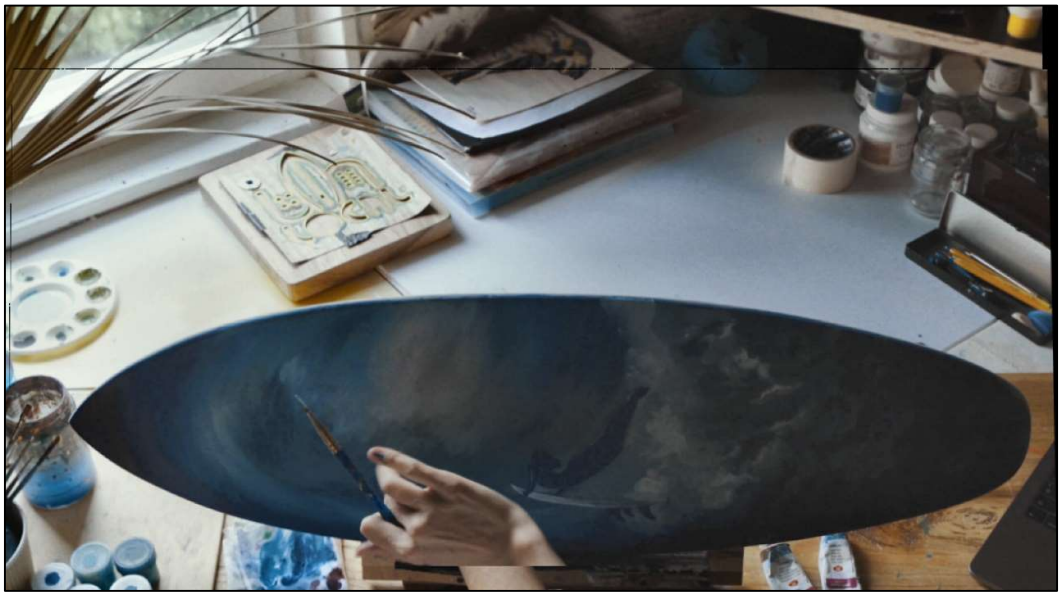
6: Images

6: Graph Cut

6: Post Pyramid Blending

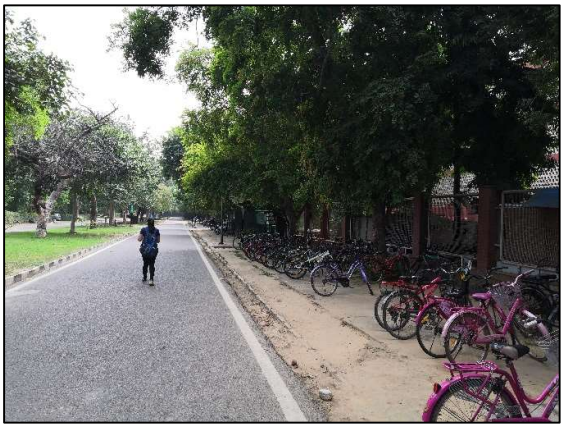
7: Images

7: Graph Cut



7: Post Pyramid Blending

8: Images



8: Graph Cut

8: Post Pyramid Blending

IV. RESULTS: Own Set: Good Mosaics

Good Mosaic 1: Images

Good Mosaic 1: Graph Cut Results

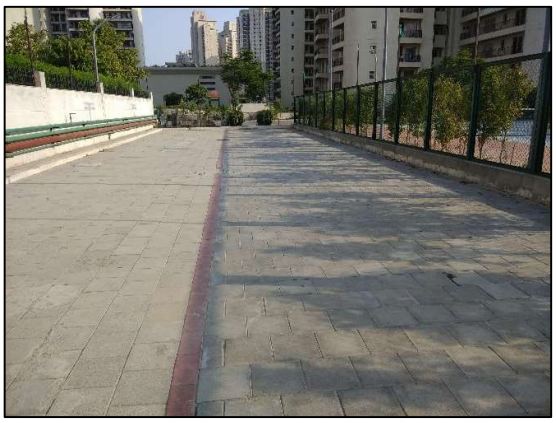
Good Mosaic 1: Post Pyramid Blending Results

Good Mosaic 2: Images

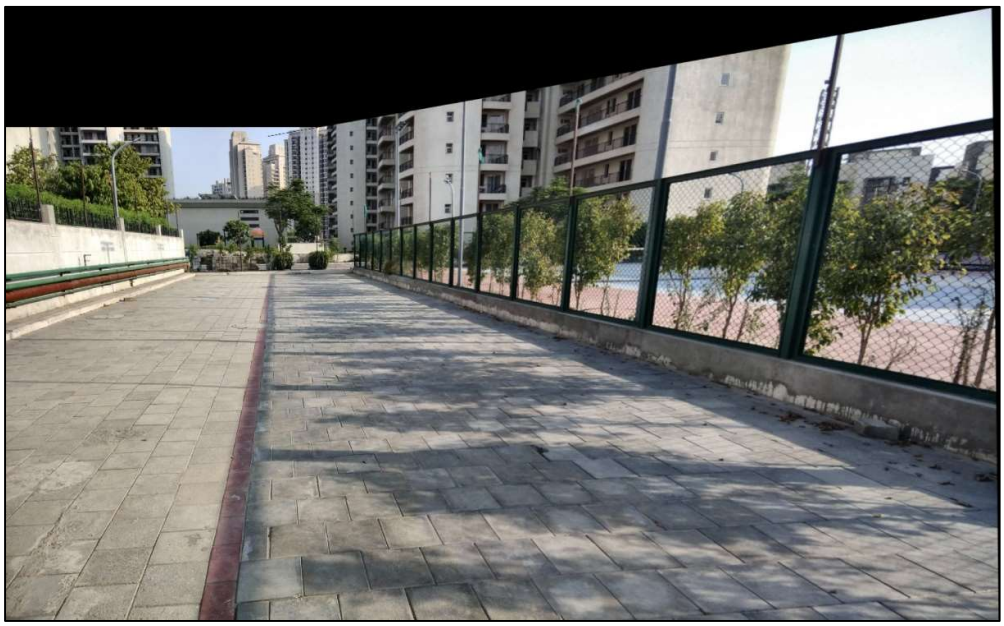
Good Mosaic 2: Graph Cut Results

Good Mosaic 2: Post Pyramid Blending Results

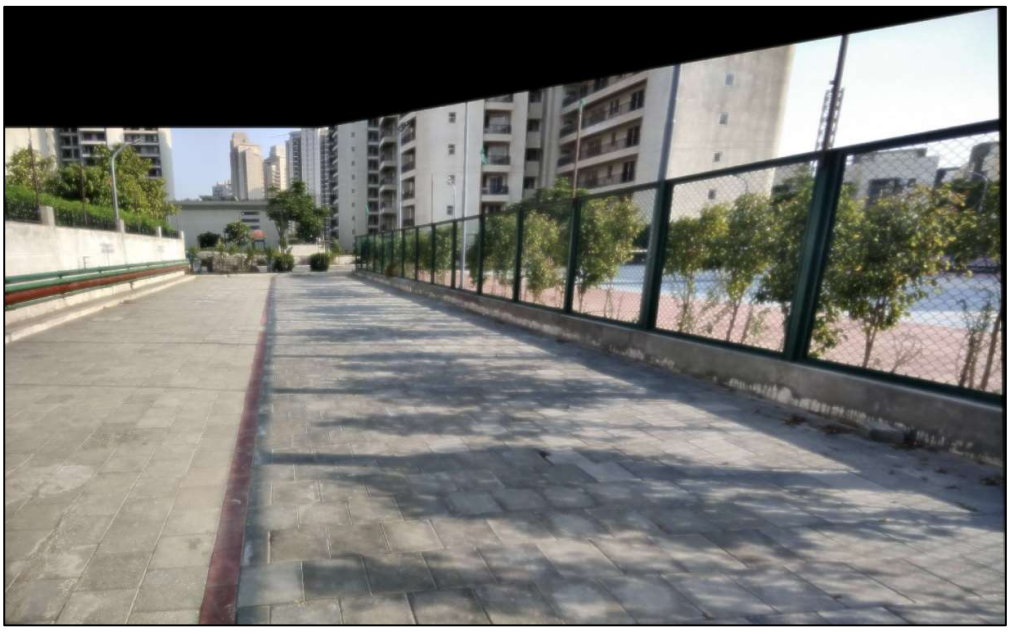
Good Mosaic 3: Images



Good Mosaic 3: Graph Cut Results



Good Mosaic 3: Post Pyramid Blending Results

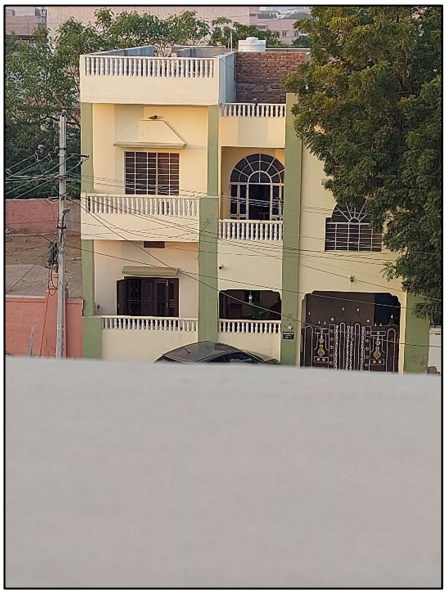


Good Mosaic 4: Images

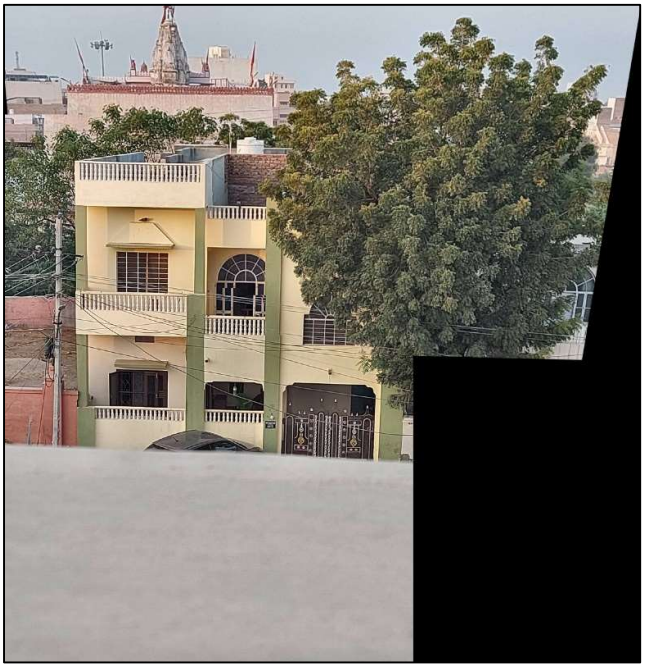
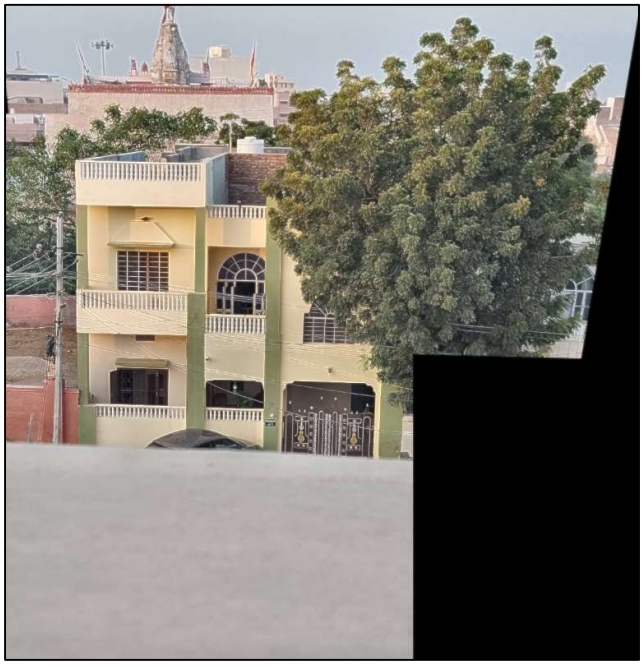
Good Mosaic 4: Graph Cut Results

Good Mosaic 4: Post Pyramid Blending Results

Good Mosaic 5: Images




Good Mosaic 5: Graph Cut Results & Post Pyramid Blending Results

Good Mosaic 6: Images


Good Mosaic 6: Graph Cut Results

Good Mosaic 6: Post Pyramid Blending Results

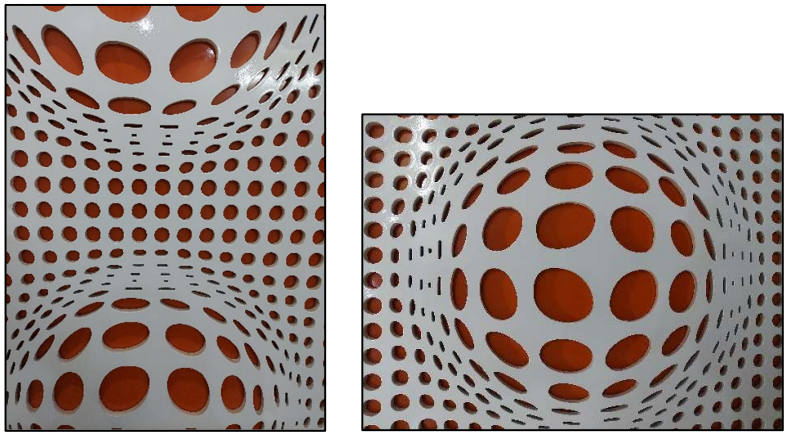
Good Mosaic 7: Images

Good Mosaic 7: Graph Cut Results

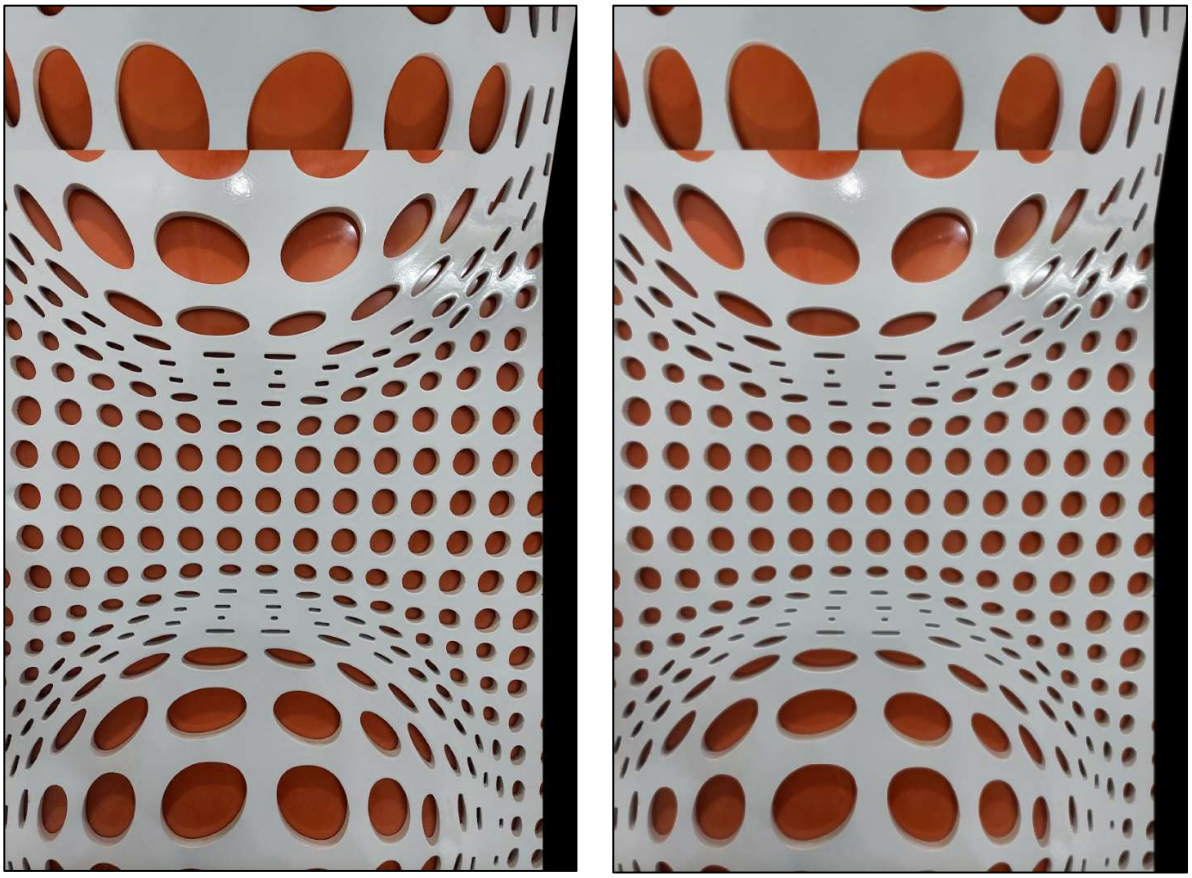
Good Mosaic 7: Post Pyramid Blending Results

V. RESULTS: Own Set: Failed Mosaics

Failed Mosaic 1: Images



Failed Mosaic 1: Graph Cut Results & Post Pyramid Blending Results



Reason of Failure: Lots of symmetrical points. Therefore, most of the feature matched were outliers. So, the Homography estimation was incorrect.

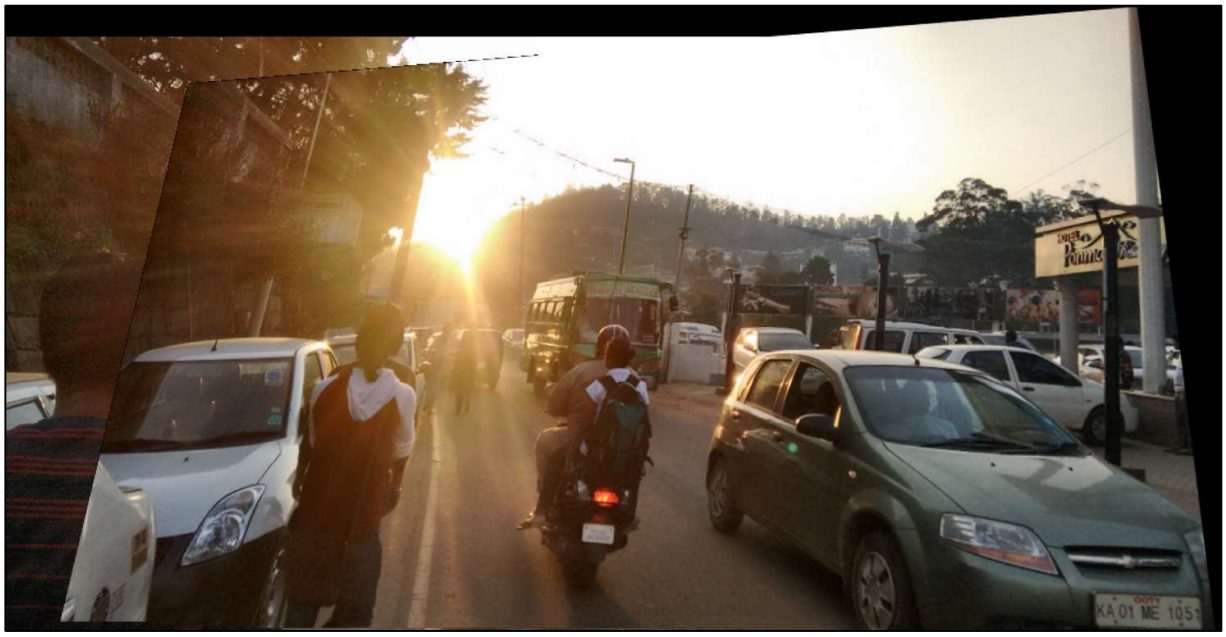
Failed Mosaic 2: Images


Failed Mosaic 2: Graph Cut Results & Post Pyramid Blending Results


Reason of Failure: Camera was rotated almost more 80 Degrees while the images were taken. Also, there are a good number of outliers. So, Homography estimation was inaccurate.

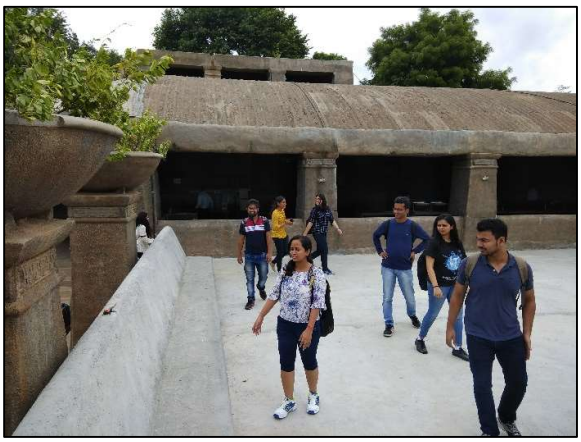
Failed Mosaic 3: Images

Failed Mosaic 3: Graph Cut Results & Post Pyramid Blending Results



Reason of Failure: Image is not clear because of high intensity sun rays. Also, lots of motion objects are there in the same frame which have zoomed in and out in the two given images. Therefore, the matches were not good for estimation of Homography.

Failed Mosaic 4: Images

Failed Mosaic 4: Graph Cut Results & Post Pyramid Blending Results

Reason of Failure: a lot of people are in motion on high intensity floor. Therefore, the graph cut in not good.

Failed Mosaic 5: Images

Failed Mosaic 5: Graph Cut Results & Post Pyramid Blending Results

Reason of Failure: In second, picture the leaves and stems are out of focus as the focus is on sun. But on the first image the focus in of branches. Therefore, the features were not crisp to be matched. Therefore, Homography estimation is inaccurate because of bad features match.

REFERENCES

[1] https://en.wikipedia.org/wiki/Photographic_mosaic

[2] https://docs.opencv.org/master/d5/daf/tutorial_py_histogram_equalization.html

[3] https://en.wikipedia.org/wiki/Scale-invariant_feature_transform

[4] https://en.wikipedia.org/wiki/Homography_(computer_vision)

[5] https://en.wikipedia.org/wiki/Random_sample_consensus

[6] https://www.cc.gatech.edu/cpl/projects/graphcuttextures/

[7] http://docs.opencv.org/3.0-beta/doc/py_tutorials/py_imgproc/py_pyramids/py_pyramids.html

****