Assignment2
pmorpari-sarafm

Part 1:

Running the code :

- ./a2 part1 “input1.img”
- ./a2 part1 lincoln.png
- The input images should be in the folder “./images/Part1”
- The output images can be seen in the folder “./OutputImages/Part1””

In part 1 of the assignment was about homographic image transformation. In part 1.1 the transformed image of given homographic matrix had to be computed, which was simple multiplication of co-ordinates to get coordinates.

$$x' = \frac{h_{00}x + h_{01}y + h_{02}}{h_{20}x + h_{21}y + h_{22}} \text{ and } y' = \frac{h_{10}x + h_{11}y + h_{12}}{h_{20}x + h_{21}y + h_{22}}.$$

Img source: Computer Vision:
Algorithms and Applications by Richard Szeliski
September 3,

Using the technique from part 1, we calculate the part 2 of the image, we tried to reverse the process and get the ***transformation matrix*** given 4 corresponding co-ordinates from two images.

Part 3 was projection of an image to billboard using the technique to find ***transformation matrix*** for the given image.

```
[[sarafm@silo pmorpari-sarafm-a2]$ ./a2 part1 lincoln.png
>>>>>>>>>>>>>>Transformation matrix>>>>>>>>>>>>>>>
1.80258 1.55276 -776.398
-0.809994       2.13295 -107.91
0.000579313     0.000757831     1
```

Assignment2
pmorpari-sarafm

Synthetic-billboard3.png

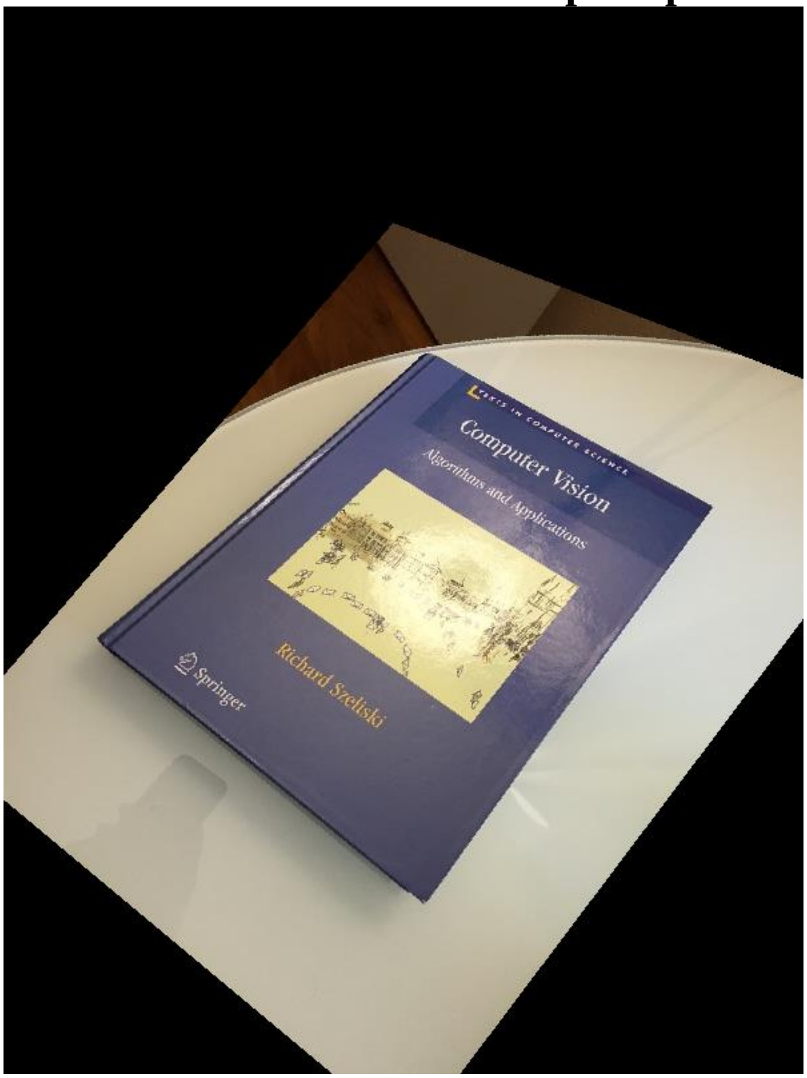
Synthetic-billboard2.png

Assignment2
pmorpari-sarafm

Synthetic-billboard1.png

Lincoln_warped.png

Assignment2
pmorpari-sarafm



Book_result.png

Assignment2
pmorpari-sarafm

Part2:

Running the code :

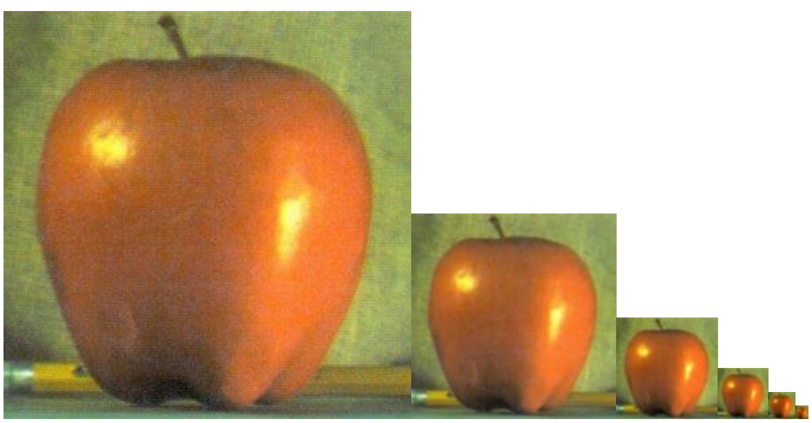
- **.**/a2 part2 input1.jpg input2.jpg Mask.jpg
- E.g. ./a2 part2 apple.jpg orange.jpg mask.jpg
- ./a2 part2 new1.jpg new2.jpg mask2.jpg
- The input images should be in the folder "./images/Part2"
- The output images can be seen in the folder "./OutputImages/Part2""

In this we first calculate the Gaussian pyramid upto depth 6 for the two input images to be blended and the mask in the following way:

G0=originalImage
G1= DownscaleBy2(ConvolutionBy Gaussian(G0))
G2= DownscaleBy2(ConvolutionBy Gaussian(G1))
G3= DownscaleBy2(ConvolutionBy Gaussian(G2))
G4= DownscaleBy2(ConvolutionBy Gaussian(G3))
G5= DownscaleBy2(ConvolutionBy Gaussian(G4))

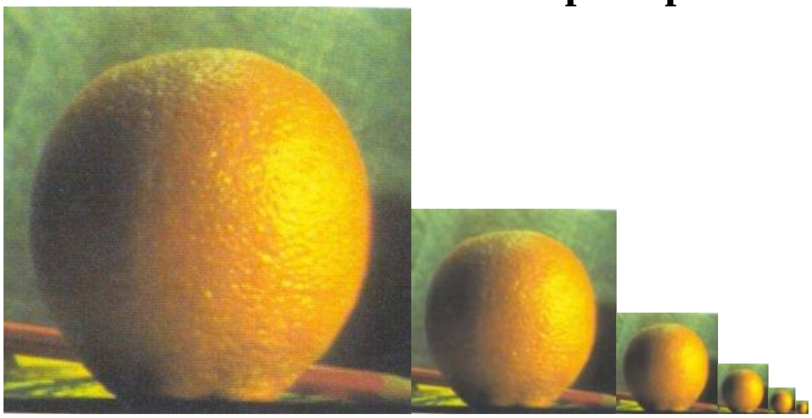
Gaussian pyramid:

Example 1:



apple.jpg

Assignment2
pmorpari-sarafm



orange.jpg

mask.jpg

Assignment2
pmorpari-sarafm

Example2:

new1.jpg

new2.jpg

Assignment2
pmorpari-sarafm

mask2.jpg

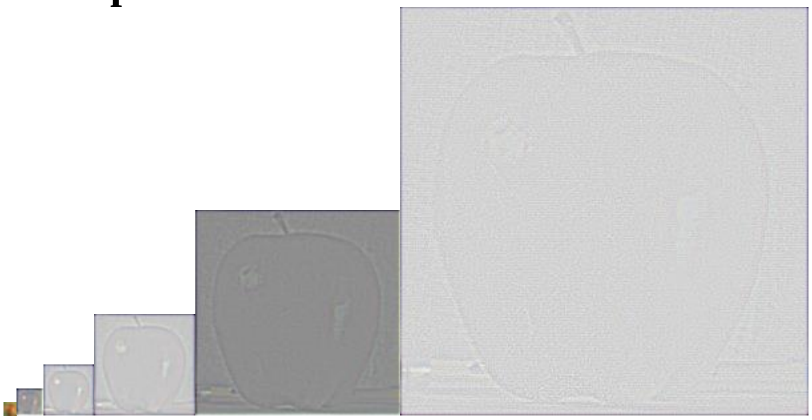
Now we calculate the laplacian pyramid upto depth 6 for both the images to be blended in the following way:

L0=G5
L1=G4-4*(ConvolveByGaussian(UpscaleBy2(G5)))
L2=G3-4*(ConvolveByGaussian(UpscaleBy2(G4)))
L3=G2-4*(ConvolveByGaussian(UpscaleBy2(G3)))
L4=G1-4*(ConvolveByGaussian(UpscaleBy2(G2)))
L5=G0-4*(ConvolveByGaussian(UpscaleBy2(G1)))

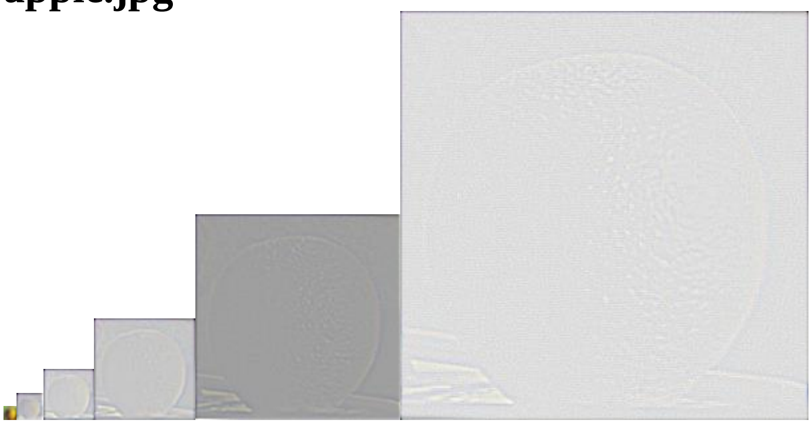
Assignment2
pmorpari-sarafm

Laplacian Pyramids:

Example1:



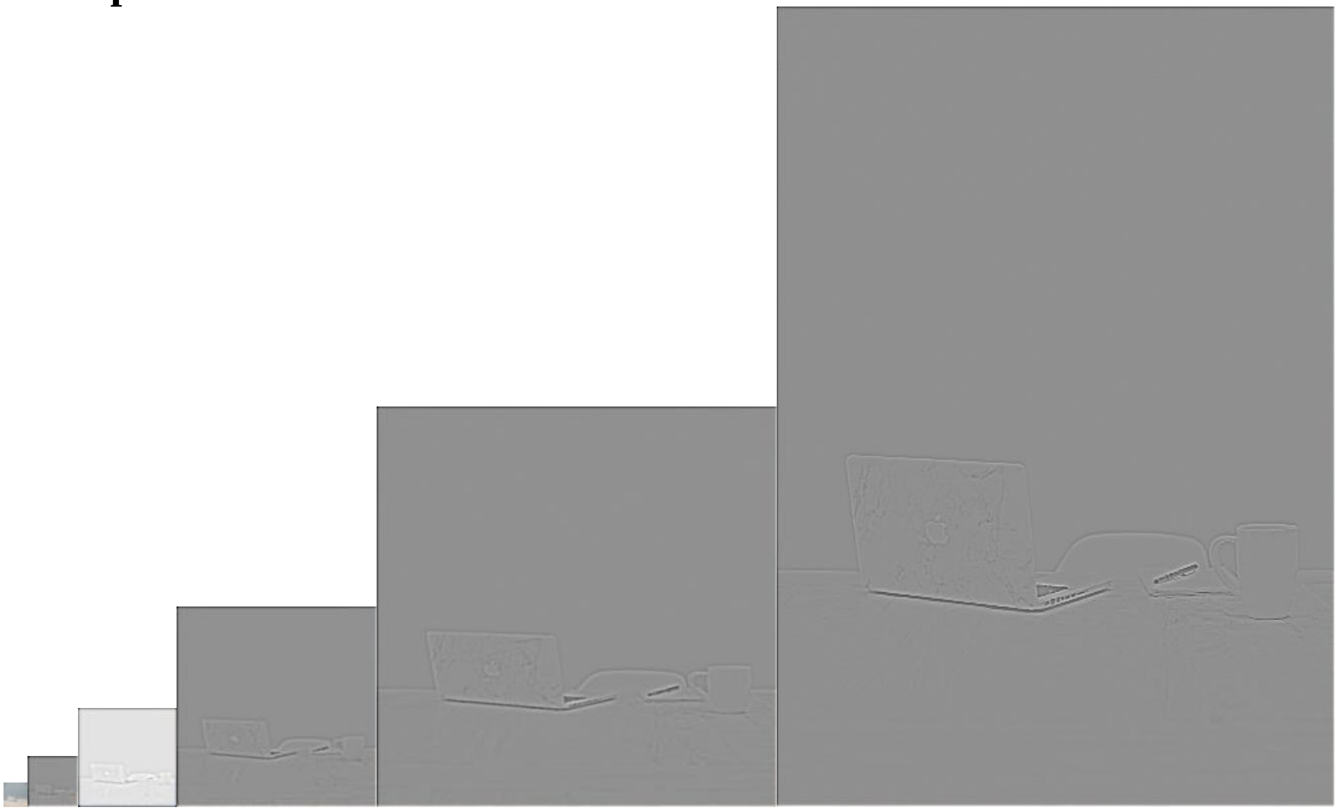
apple.jpg



orange.jpg

Assignment2
pmorpari-sarafm

Example2:



new1.jpg

new2.jpg

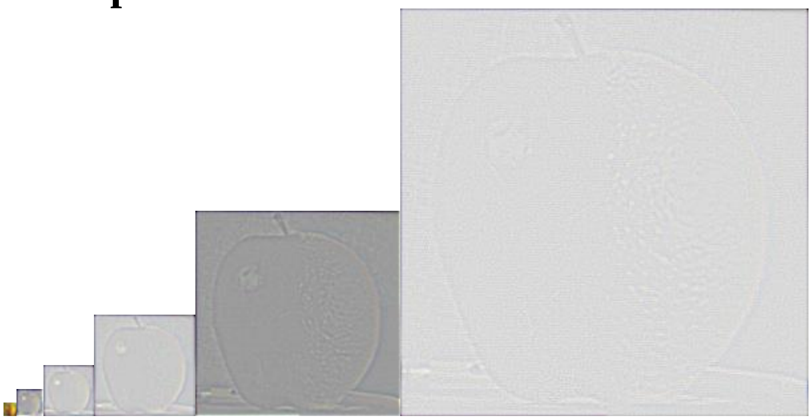
Assignment2
pmorpari-sarafm

Now we calculate the blended laplacian pyramid upto depth 6 for both the images weighted by their masks in the following way:

LB_i=L_image2_i*Mask_i+ (1-Mask_i)*L_image1_i

Blended Laplacian Pyramid:

Example1:



Example2:

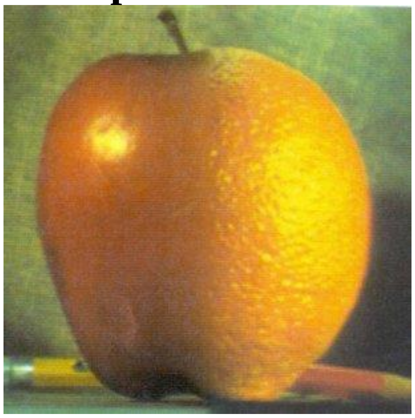
Assignment2
pmorpari-sarafm

Now we reconstruct the final blended image in the following recursive way upto depth 6:

BlendedlaplacianAt(i+1) = 4*ConvolveByGaussian(UpscaleBy2(BlendedlaplacianAt(i))) +BlendedlaplacianAt(i+1)

The Image we get at the final level is our final blended image.

Example1:



Blended.jpg

Example2:

Assignment2
pmorpari-sarafm

Part 3:

Execution:

- ./a2 part2 "input1.png" "input2.png"
- E.g. ./a2 part3 eiffel_18.jpg eiffel_19.jpg
- E.g. ./a2 part3 bigben_2.jpg bigben_3.jpg
- The input images should be in the folder "./images/part3"
- The output images would be in the folder "./OutputImages/part3"

This part was about finding similarities between given images.

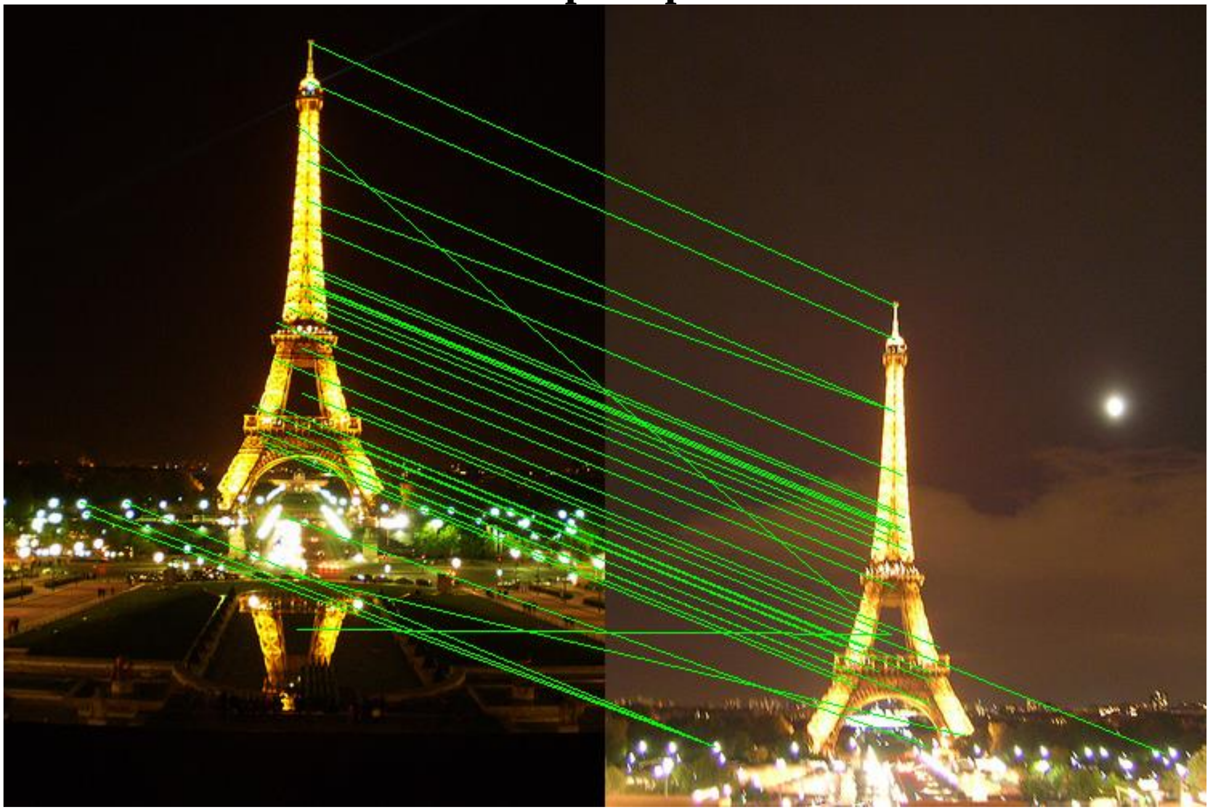
The given two images were scanned for sift points and then scanned for given. The nearest corresponding point was compared to second nearest point and made to par the threshold in order to qualify for a pair. The corresponding points were matched.

Based on the points derived from part 3.1, we calculated the outliers and inliers from those points using RANSAC. RANSAC in short is random trial and error method to test hypotheses and find the best among them. This will be useful to get the best homography matrix in order to transform the image correctly.

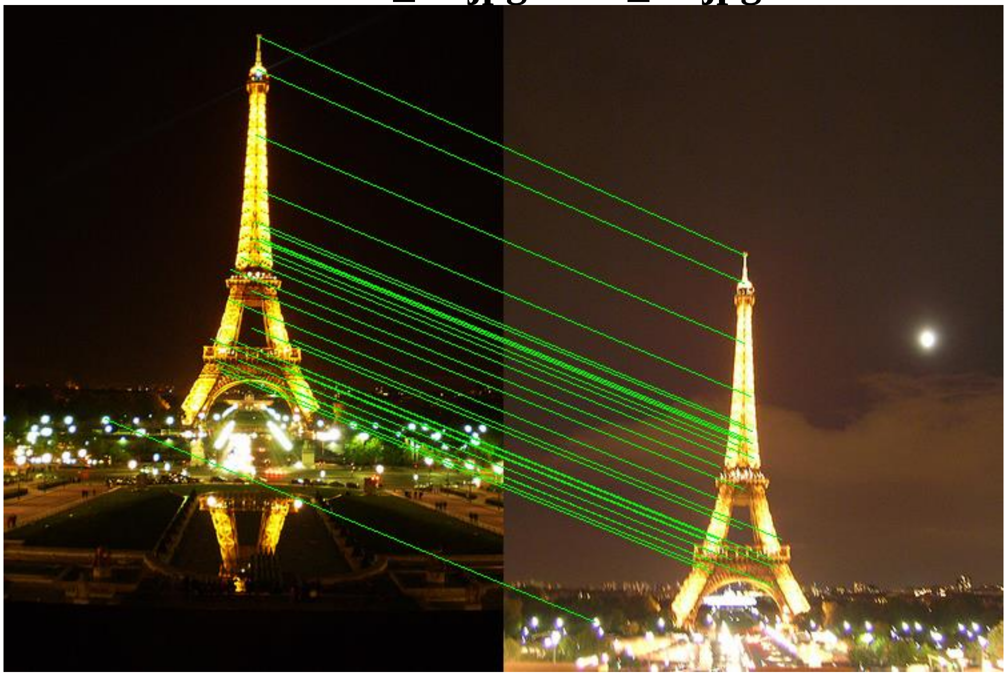
Tests & Playing with threshold:
SIFT Descriptor: optimal solution with minimized outliers was obtained when d1/d2 was < 0.7

RANSAC: Outliers were reduced with the points were found to be in proximity of +-4 pixels.

Assignment2
pmorpari-sarafm



Sift matches for eiffel_18.jpg eiffel_19.jpg



Ransac matches for eiffel_18.jpg eiffel_19.jpg

Assignment2
pmorpari-sarafm

Sift matches for bigben_2.jpg bigben_3.jpg

Ransac matches for bigben_2.jpg bigben_3.jpg

Part4:

Execution:

- ./a2 part4 roof_1.jpg roof_2.jpg roof_3.jpg roof_4.jpg roof_5.jpg
- ./a2 part4 Hill_1.jpg Hill_2.jpg Hill_3.jpg
- ./a2 part4 books_1.jpg books_2.jpg books_3.jpg
- ./a2 part4 books_1.png books_2.png books_3.png
- The input images should be in the folder "./images/part4"
- The output images would be in the folder "./OutputImages/part4"

In this we take we are taking N images in sequence from left to right to create a combined panaroma image.

Assignment2
pmorpari-sarafm

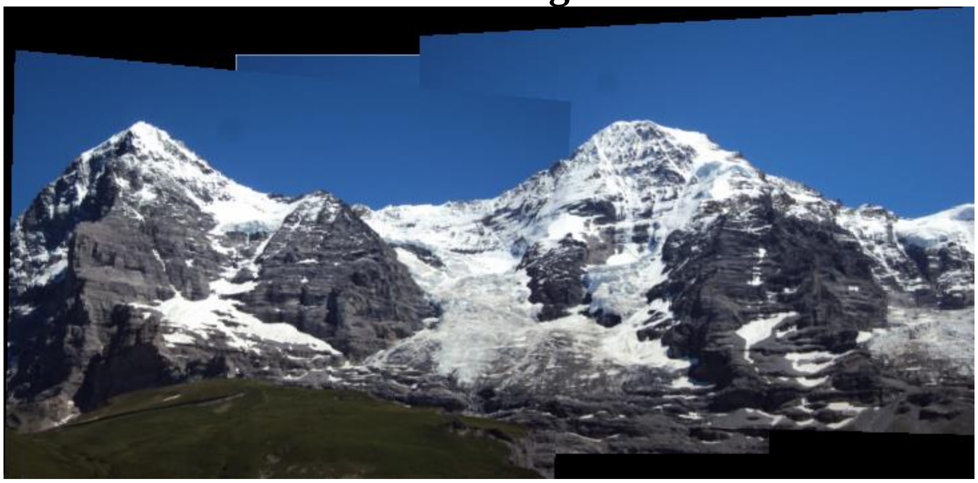
We perform the following steps in this:

1. First, we took the center image and performed padding around the image.
2. **Center to right**
   - We calculated the Ransac of Right Image with respect to the padded center image.
   - After getting the homography matrix using ransac we performed projection of the right image on the center image as performed in part1 billboard example.
   - No, we take the combined image and perform the same steps with respect to the next right image until the right most image is combined.
3. Considering the combined image obtained in the previous step as center, we perform similar steps and move towards left.
4. Removed the unwanted padding around the image for a better result.

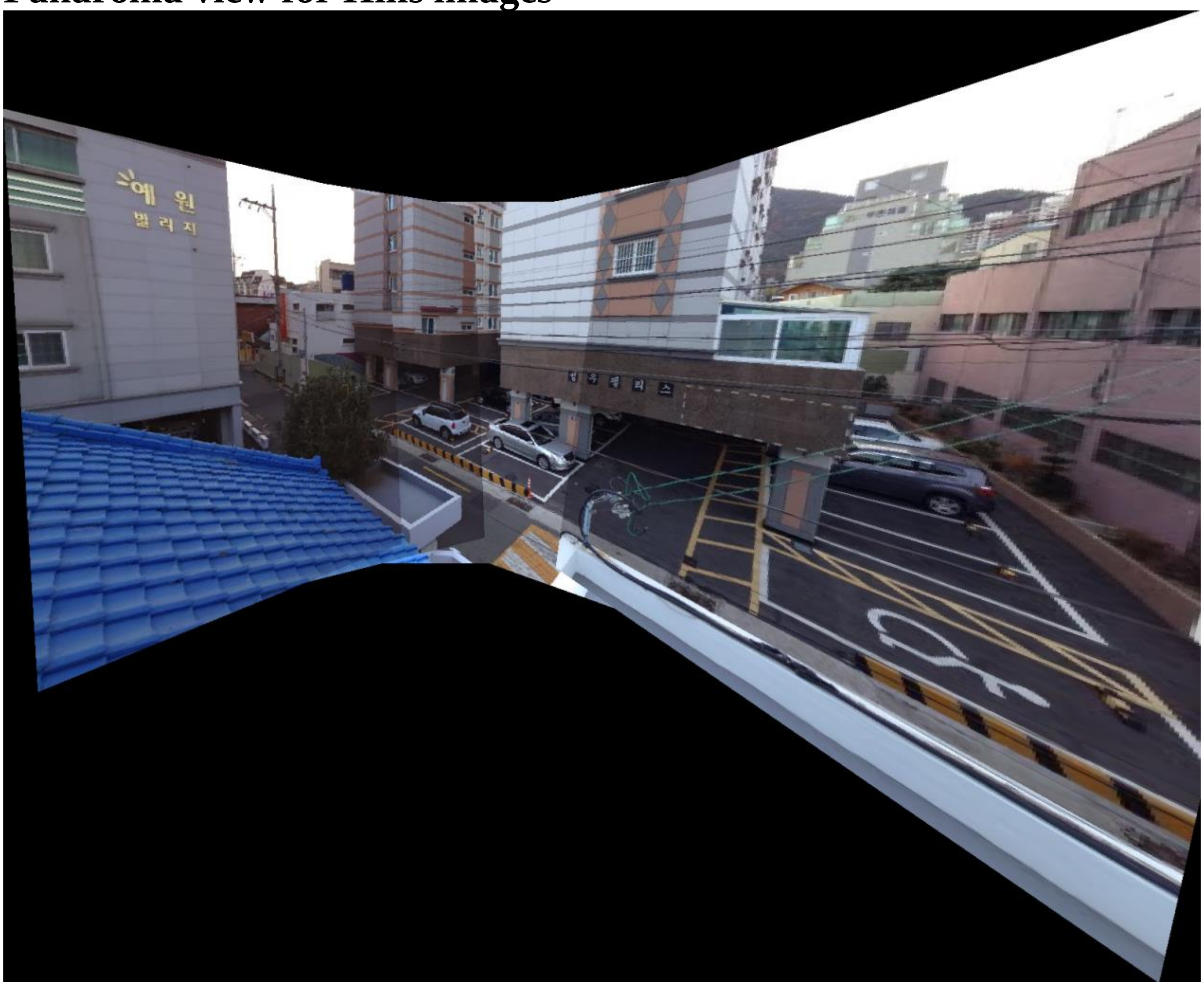
We further tried averaging pixel values of overlapped portion but got unsatisfactory results for the given examples.

Assignment2
pmorpari-sarafm

Panaroma view for books images

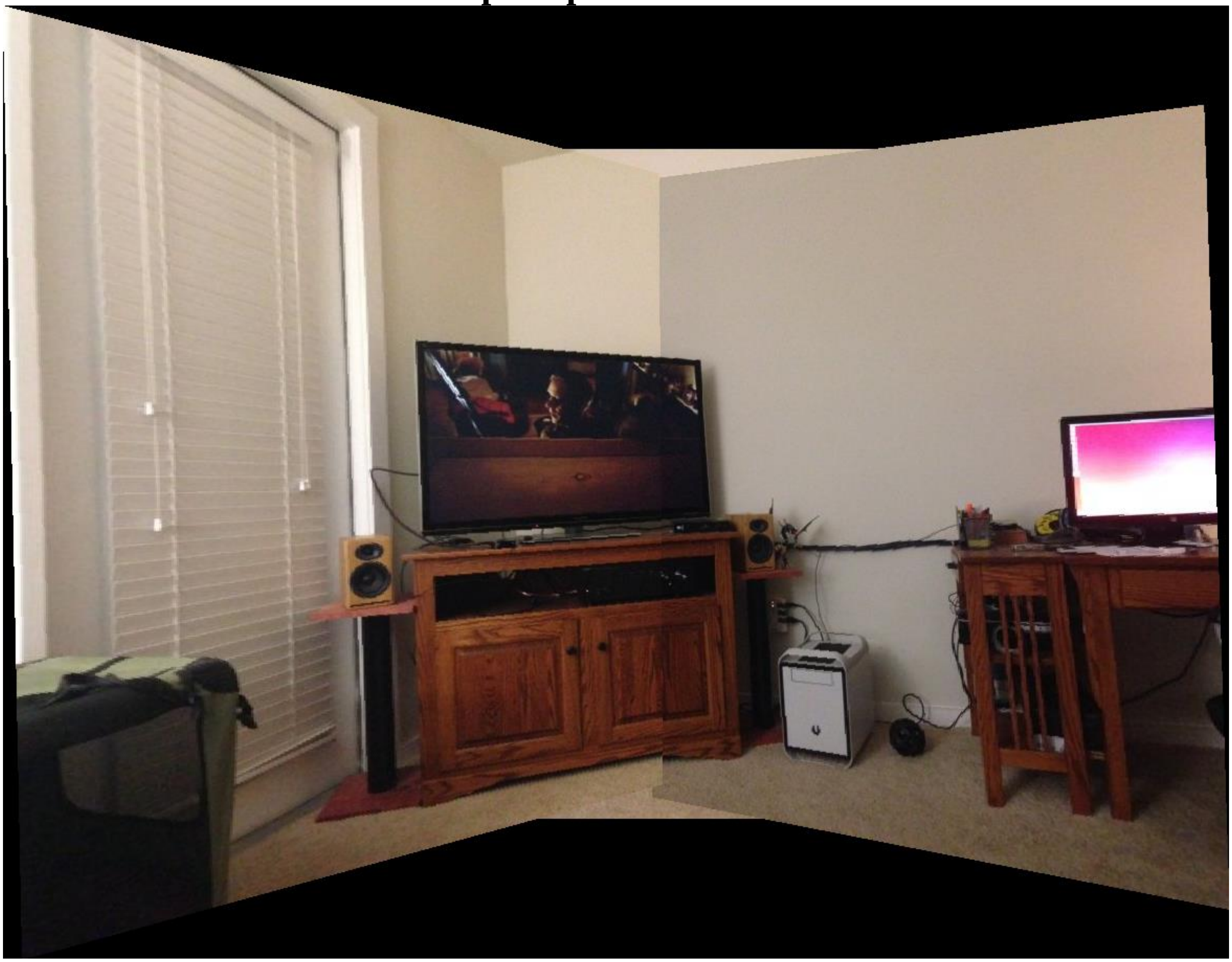


Panaroma view for Hills images



Panaroma view for roof images

Assignment2
pmorpari-sarafm



Panaraoma view for Panaroma imageset