ROBOTICS ALGORITHM

LAB 3 REPORT

UBIT Name and Person No.

sdhande

50357082

1) Introduction:

In this lab we will be implementing the image processing algorithms to detect the features and get the frame alignment. We will be using 6 photos from Prokudin-Gorskii photo collection for the image processing and frame alignment.

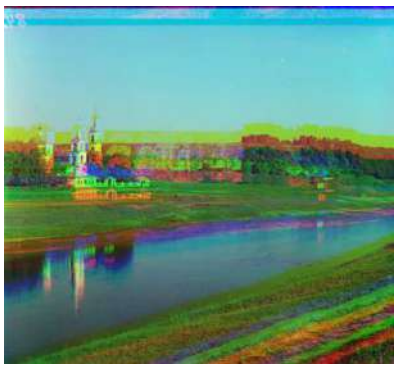
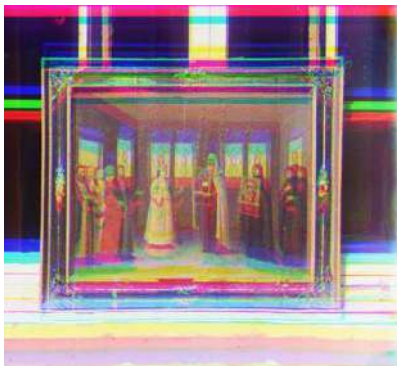
2) Objective:

Lab3 has 3 parts as below:

1) The 6 images given have three separate glass plate images, one each for each color channel (R, G and B). The first task is to take each take each of the six images and align the three plate images as three-color channel images and save the resultant color image.
2) The 2nd task is to find the correct alignment of the images using Sum of Squared Differences (SSD) also known as L2 norm.
3) 3rd task is to find the correct alignment of the images using another metrics known as normalized cross-correlation (NCC).
4) The 4th task is to get the correct image alignment using the feature detection and align the images based on the best fit features. For this task we will run the Harris corner detector method and use this response to get correct alignment by running RANSAC algorithm on the detected features.

3) Implementation:

1) We first crop all the images into 321:350 size and combine the given 3 separate R,G,B channels to get one single image. Though the image is not properly aligned and looks a little blurry. The output result of all images is shown below.



2) Sum of Squared Differences:

i. After getting the above results we now implement the SSD method for alignment of the images, using the R,G,B channel of all 6 images.

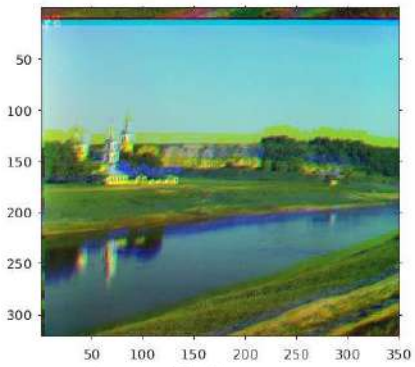
ii. SSD is one of the image matching metrics which gives the measure of match between two images pixel by pixel by getting the difference between Image1 and Imag2. Image1 in our case is the blue channel image and the image 2 will be red and green channel image.

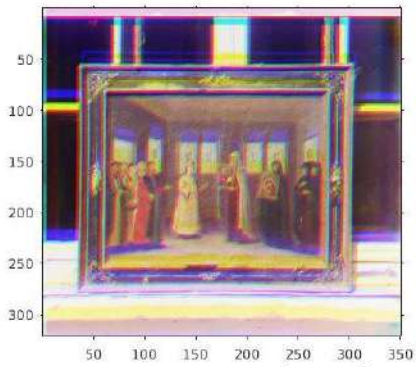
iii. SSD calculates the difference between two images and squares it pixel by pixel and finally sums it all to get a location of minimum value in the image matrices. SSD is given by formula:

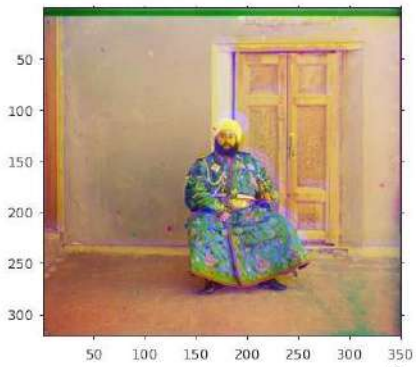
$$SSD = sum(sum\ (Image1 - Image2)^2)$$

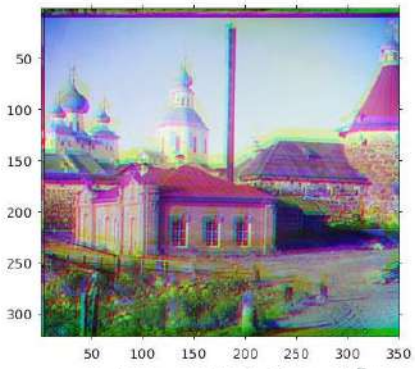
iv. Here we use the window of range[5.0:10.0]and range[-3.0:2.0], the window we get here is little different as due to the unusual cropping of the image. Tried a lot of combinations of different window size and choosed the best one which gives approximately 95% accurate alignment of all the images.

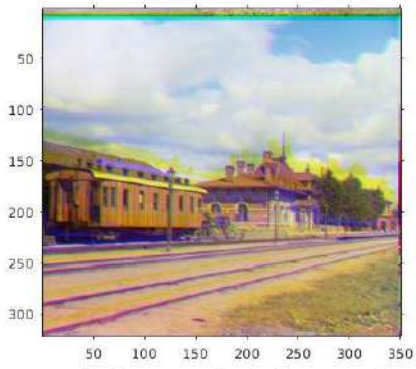
v. The result after applying SSD to all images are as below, the displacement vector/Alignment shift we get for red and green channel images is mentioned below the images. The final displacement vector is use to shift the red and green channels which aligns with the blue channel image and then finally we concatenate all 3 channels to get a perfect alignment.











3) Normalized Cross-Correlation (NCC):

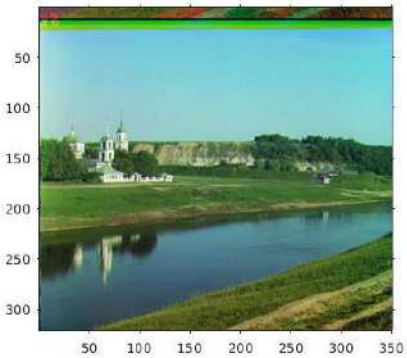
i. NCC is another image alignment method which have a good robustness and gives alignment between two images with high accuracy.

ii. NCC have more complex computation then the previous SSD method. This method have a simple concept of aligning images. It takes in the magnitude of Image 1 (blue channel) and the image 2 (red and green channel), and divides it with the Image1 and Image2. After dividing we sum the results and then do the dot product of the sum(Image1/|Image1|) and sum(Image2/|Image2|).

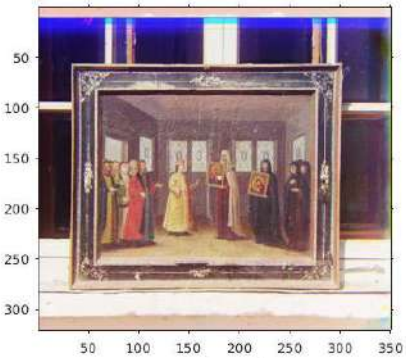
NCC= (sum(Image 1/|Image1| . Image 2/|Image 2|))

iii. Here we use a window of range [-20:30] to check the Image2 whether it matches with the Image 1. It searches for the maximum value in the image matrices unlike the SSD method where it finds the minimum value in the image matrix.

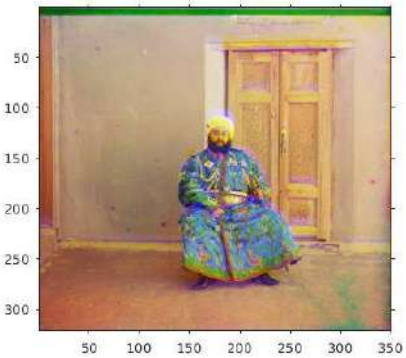
iv. Finally, we get the displacement vector for both red and green channel using which we finally shift the red and green channels which aligns with blue channel and then we concatenate all 3 channels to get the perfect alignment of all the images. The displacement vector of red and green are given below the respective images.



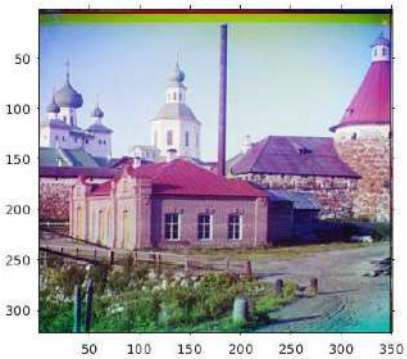
R[21,-1] G[11,1]



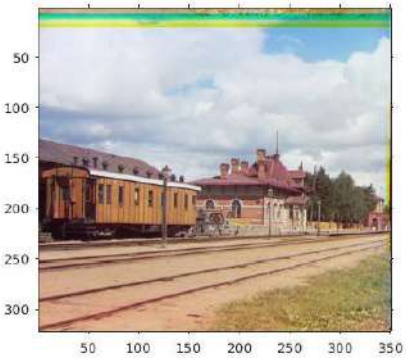
R[22,-2] G[10,0]



R[9,-4] G[8,-2]



R[18,-1] G[6,-1]



R[20,-4] G[9,-1]

R[26,-1] G(10,-1)

4) Feature detection by Harris Corner Detector and Image Alignment using RANSAC Algorithm:

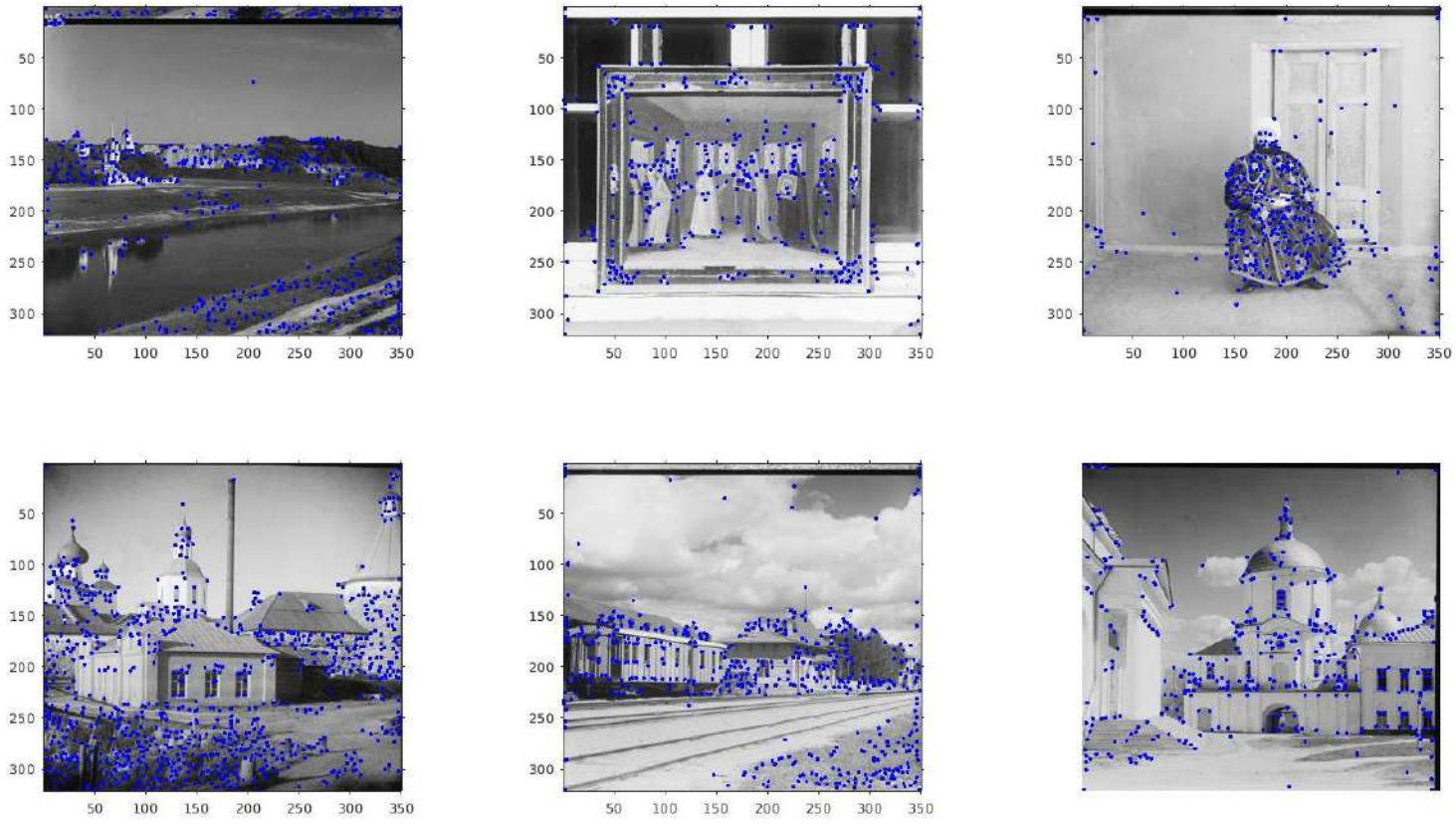
i. Harris Corner detector is the operator use to find the corners or specific features in the image. This method is widely use due to its simple computation and fast processing of the images and easy to work on any computer.

ii. In our case we use the Harris corner response to detect the corners in our R,G,B channel images. We first calculate the derivatives in x, y and xy axis and then compute the variations in the image using the below formula which results into the detection of corners in the image which we called as a feature.

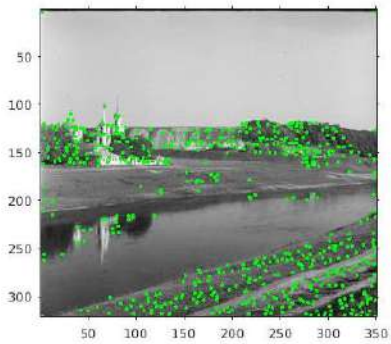
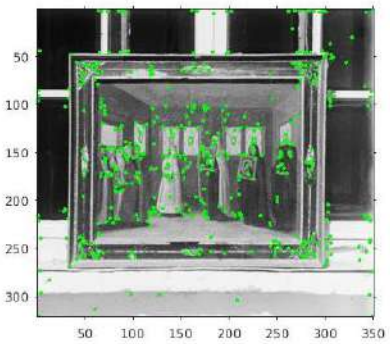
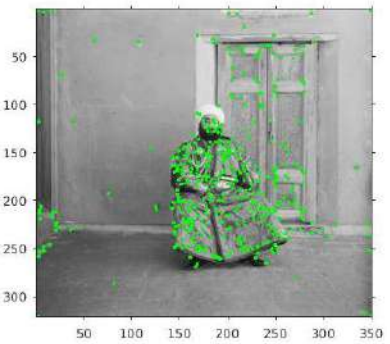
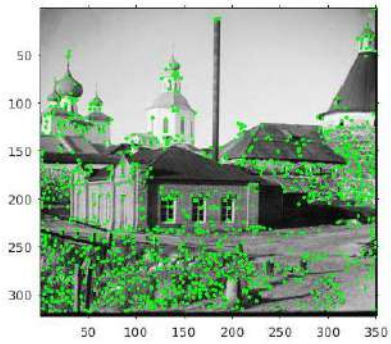
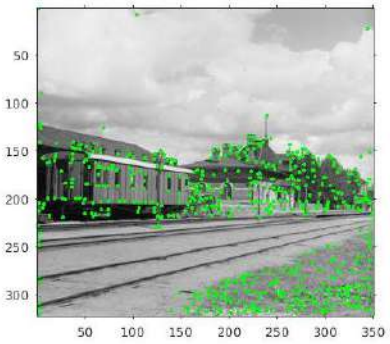
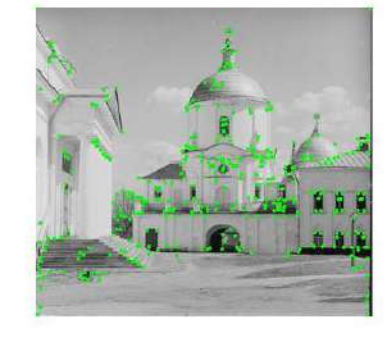
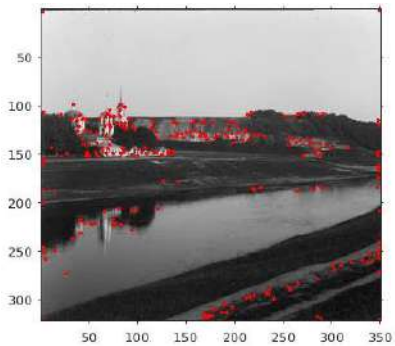
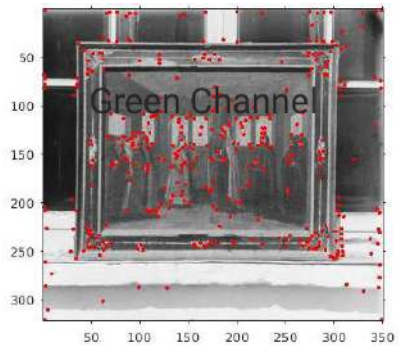
$$R = \det(M) - \text{constant } k * (\text{trace } (M)^2)$$

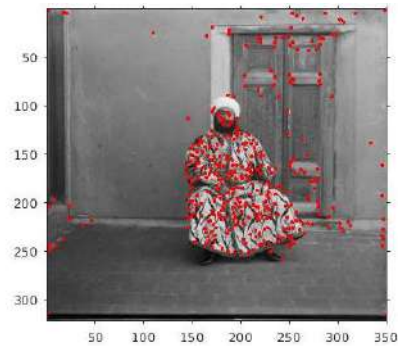
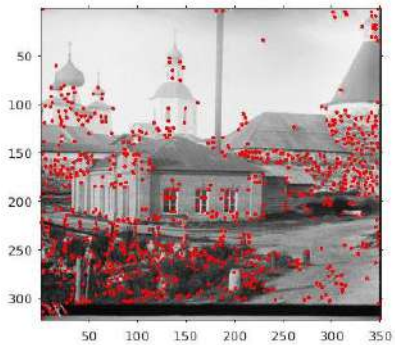
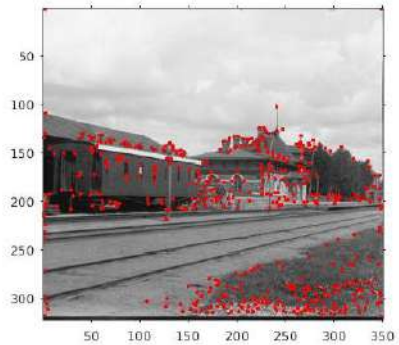
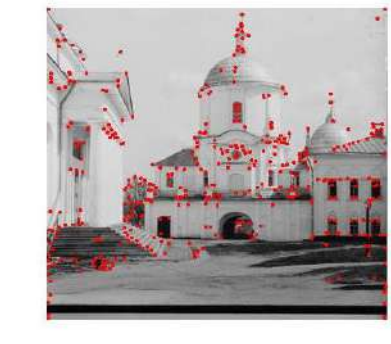
where, $M = \begin{bmatrix} Ix^2 & Ixy \\ Ixy & Iy^2 \end{bmatrix}$

iii. Below are the detected features (corners) for the R,G,B channel images.



Blue Channel

Red Channel

iv. Now we use the detected features to get the Alignment of the Images using the RANSAC algorithm. In this algorithm it will randomly pick the feature in image 1 (B channel) and tries to match with the random another feature in the image 2 (R & G channel). Assuming it aligns with the feature in image 2, we will calculate the pixel shift for this alignment and apply the same pixel shift to every feature in Image 1, and search for a corresponding feature in image2 within a threshold (a small window (range (-2:2))).

v. Now, if we find the feature in that window, we will count it as an inlier. We run this algorithm for several times and pick the best alignment (highest number of inliers). We get the final alignment of R and G channel which we use to shift the respective channels and then finally concatenate the all the 3 channels to get the final image.

vi. In our case the RANSAC Algorithm doesn't work that good as the other Image alignment method does, which were used previously. The resulting images after applying RANSAC algorithm is as below.