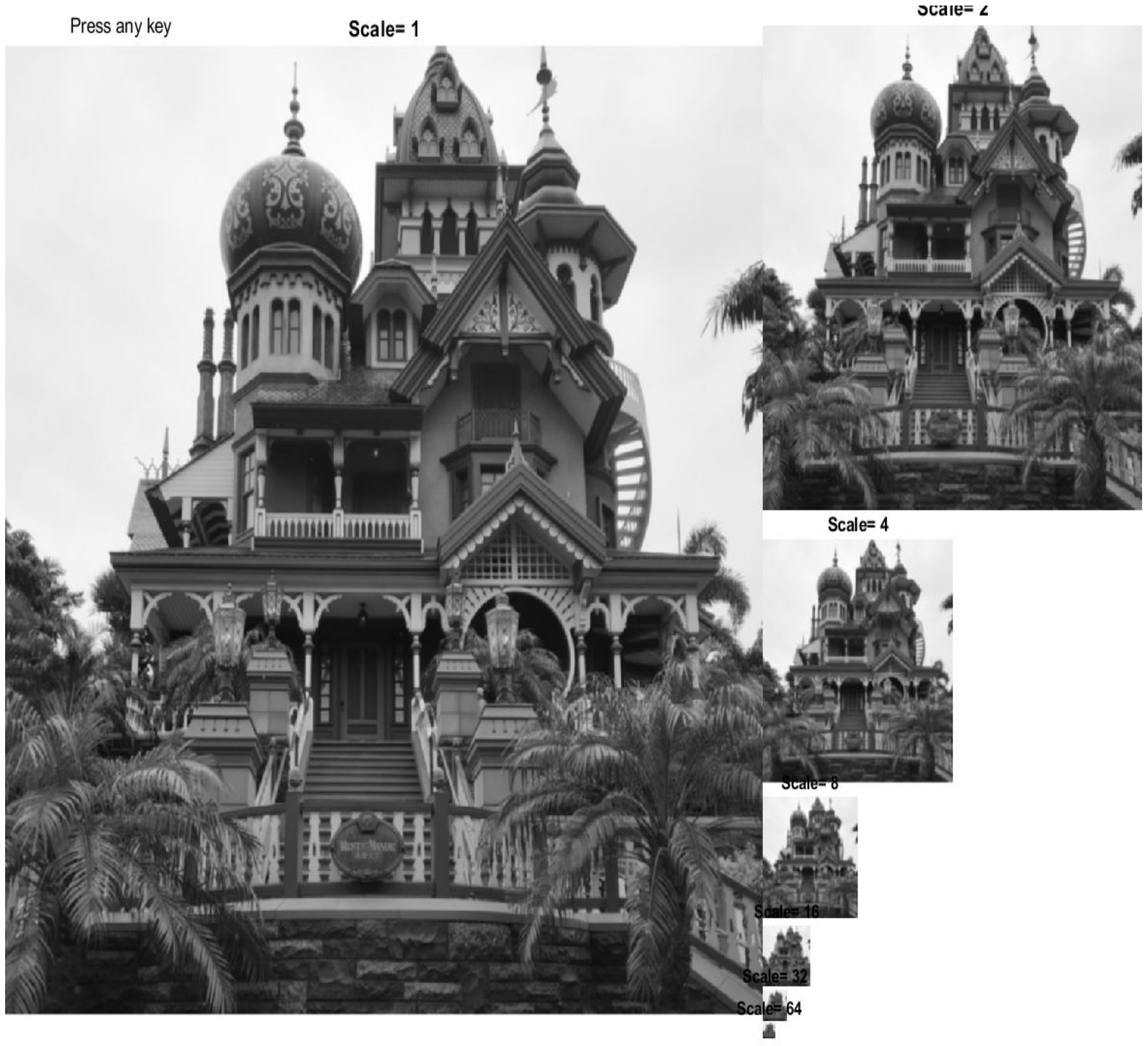
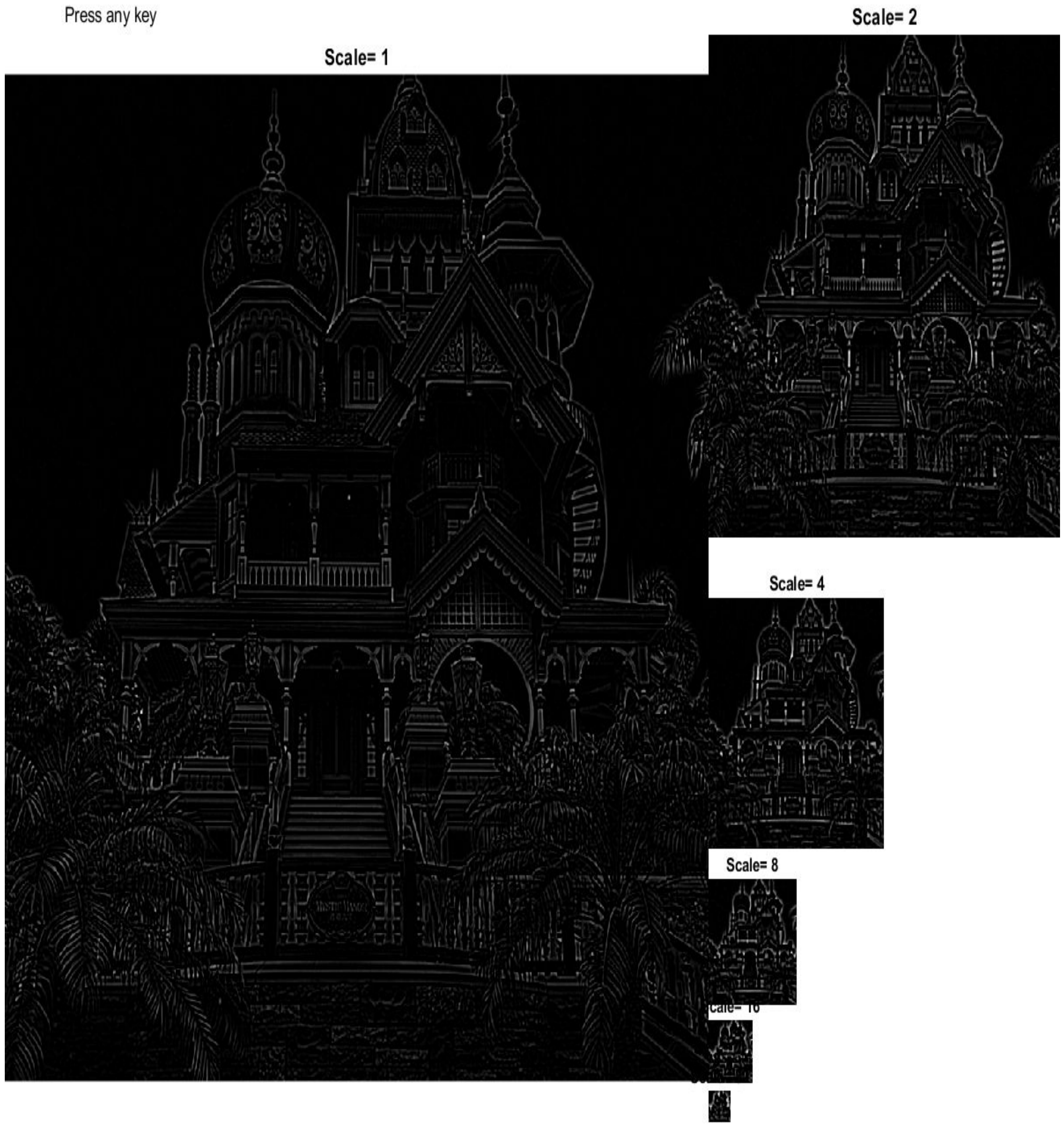
COMPUTER VISION ASSIGNMENT 2

Nishant Mishra
260903177

1. The task associated with this problem was to generate levels in Gaussian Pyramid for a given Image. A gaussian pyramid was generated by blurring the image with a gaussian filter with sigma successively increasing by a factor of 2 and downsampling the image by the same factor. 7 levels were generated with σ in range [1,64].



2. Using the gaussian pyramid constructed earlier , a Laplacian pyramid(Difference of Gaussian) pyramid was created, where for each level in the Laplacian pyramid, we took the difference of the image at the same level in Gaussian pyramid and an upsampled version of image in the next level in the Gaussian pyramid. Since there were 7 levels in the Gaussian pyramid, we get 6 levels in the Laplacian pyramid.



3. Now using the laplacian pyramid we had to obtain scale space extrema location. We found local extremas in each level of the laplacian pyramid by taking a local area and comparing the intensities in that local region for the same scale as well as the adjacent(next and previous) levels in the pyramid as shown in figure 0. Two local neighbourhood sizes(3*3,5*5) were tried. 3*3 gives 267 points Also a threshold was specified in order to select only strong extremas, i.e minimum difference in intensity>threshold. Different values of threshold produced different results as has been summarised by the figures below

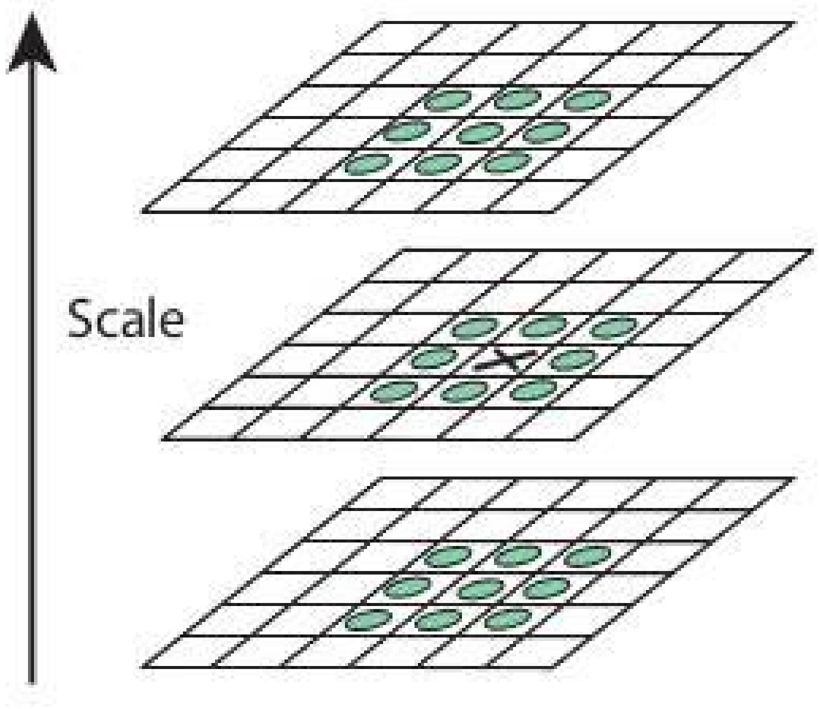


Figure 0 Strategy for finding scale space extremas for 3*3 local neighbourhoods

Blue(Sigma=2),Green(Sigma=4),Yellow(Sigma=8),Red(Sigma=16) were used to denote extremas at different scales

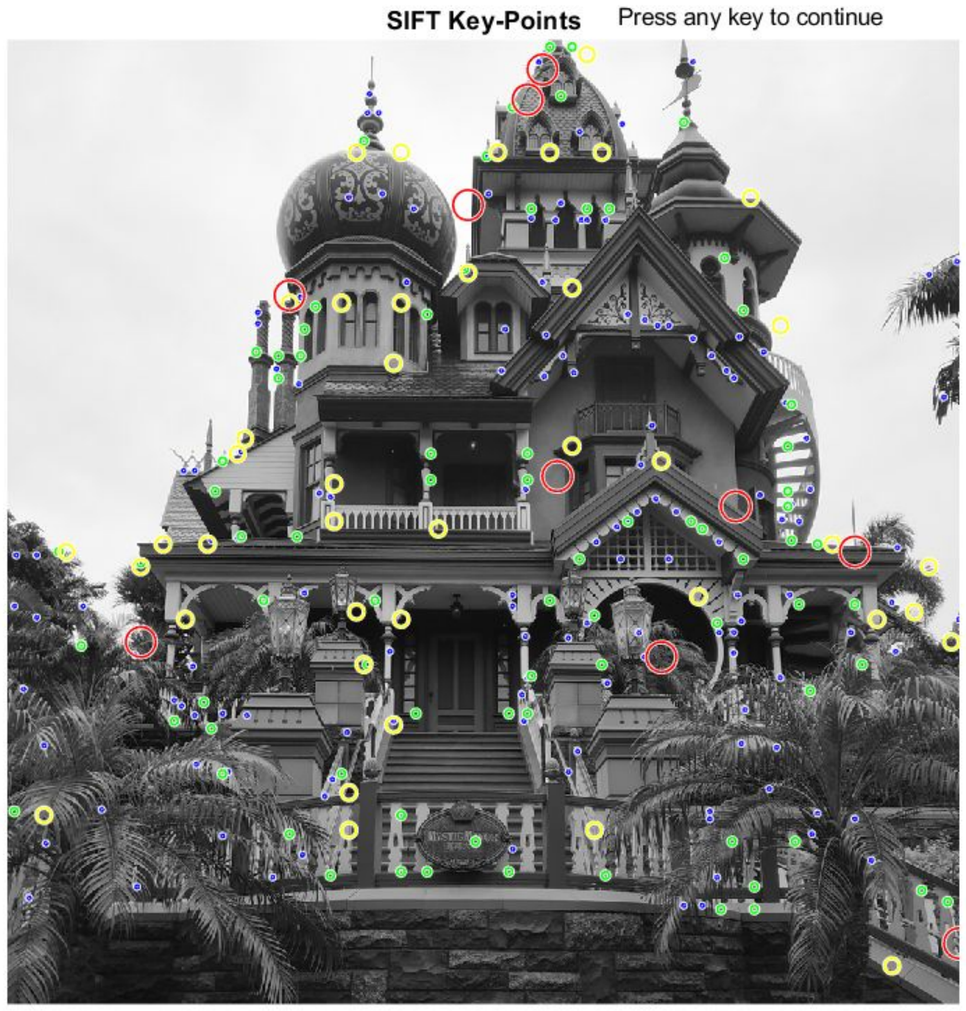


Figure1 Keypoints on the image with neighbourhood=3*3 and threshold=3
Number of keypoints=267

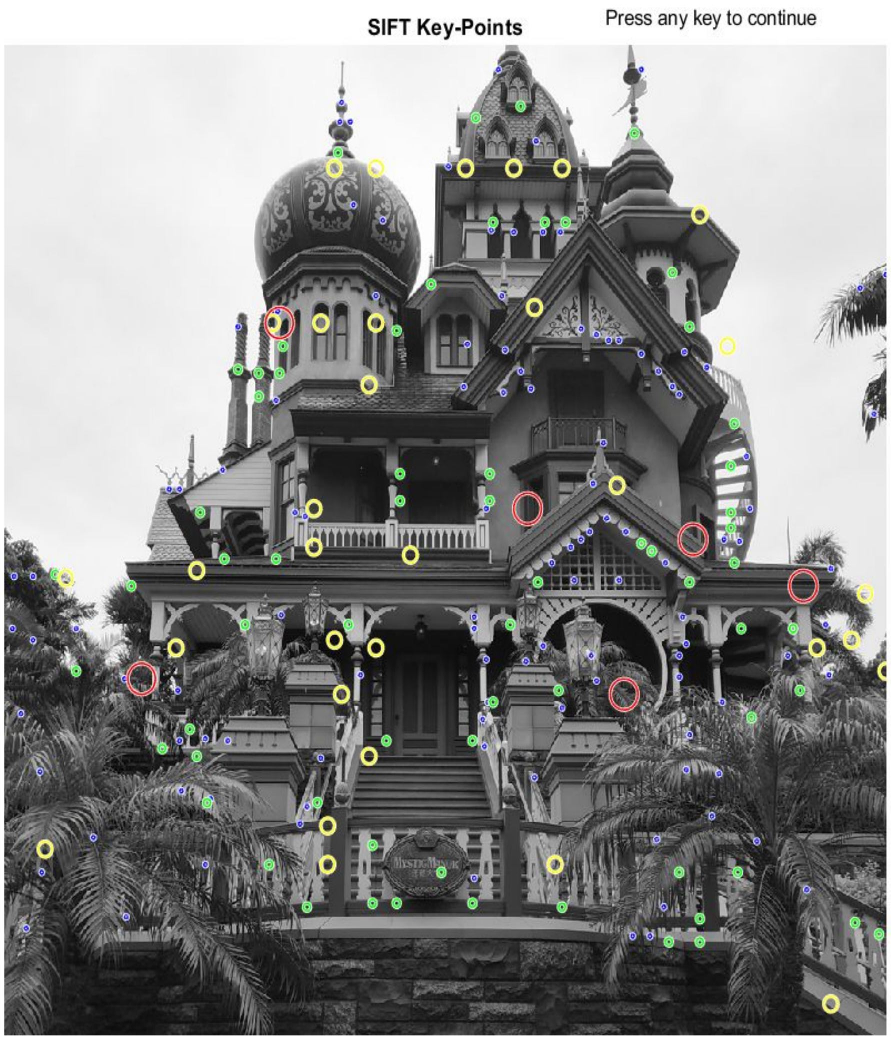


Figure2 Keypoints on the image with neighbourhood=5*5 and threshold=3
Number of keypoints=218

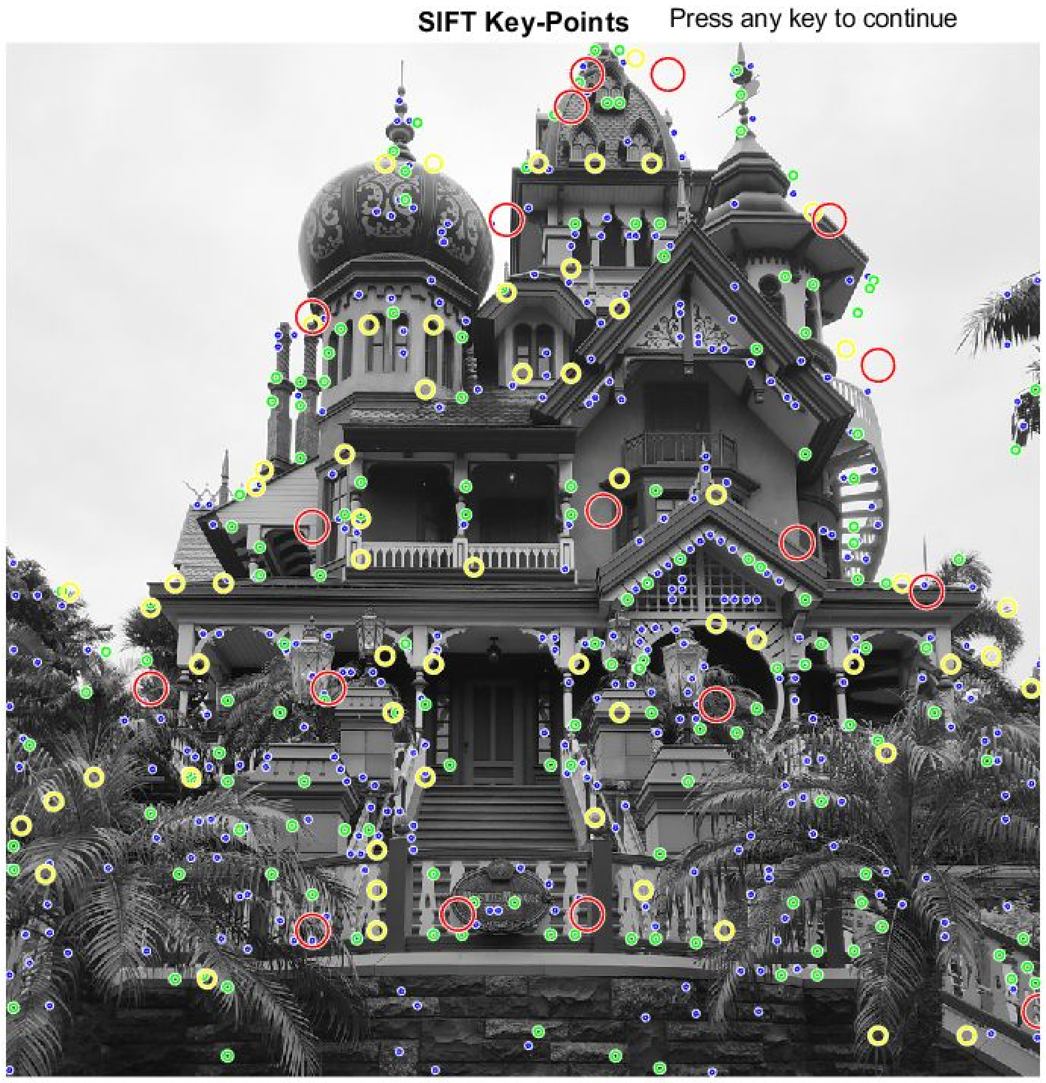


Figure3 Keypoints on the image with neighbourhood=3*3 and threshold=2
Number of keypoints=563

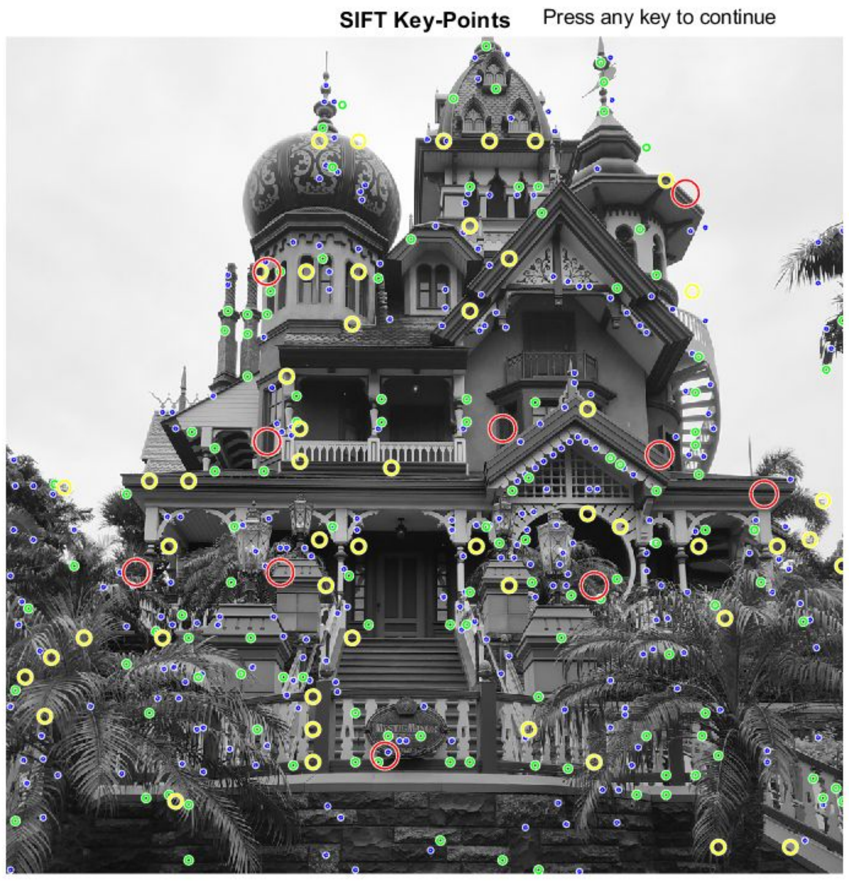


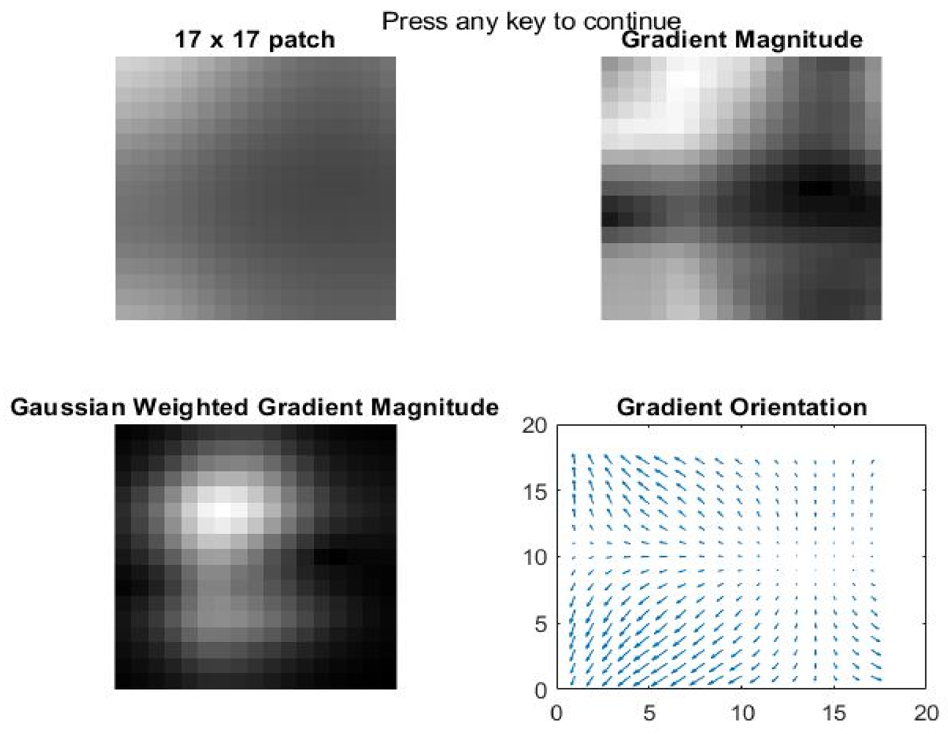
Figure4 Keypoints on the image with neighbourhood=5*5 and threshold=2
Number of keypoints=444

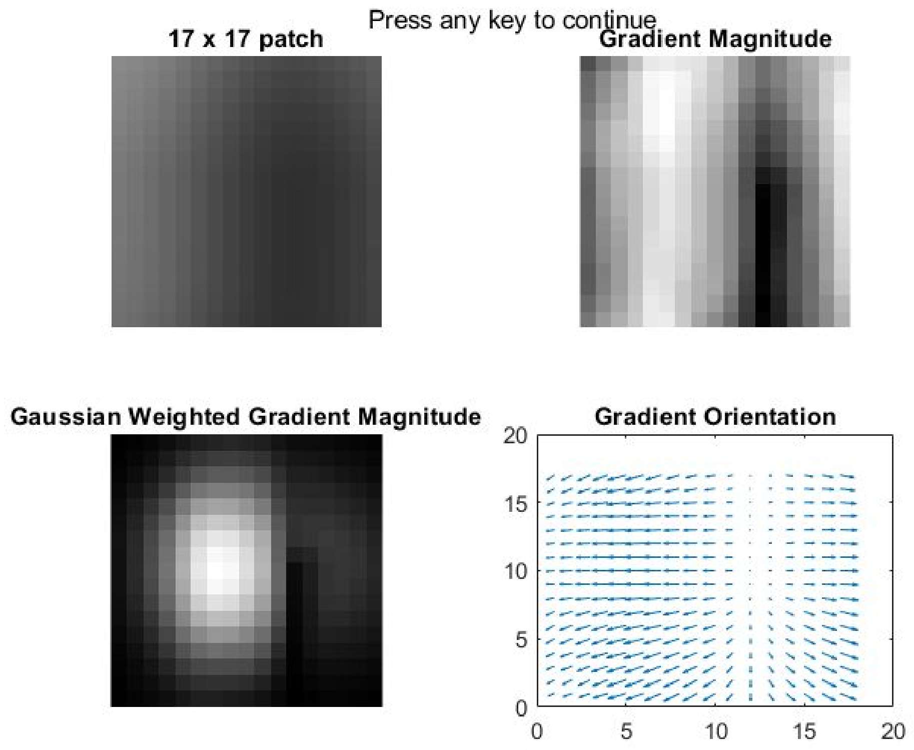
4. Once the keypoint locations were found, sift feature descriptors were to be calculated. For this, a patch of size 17*17 was taken around every keypoint location. Then gradient magnitude and direction were calculated. If G_x and G_y are the gradients in x and y direction respectively, then

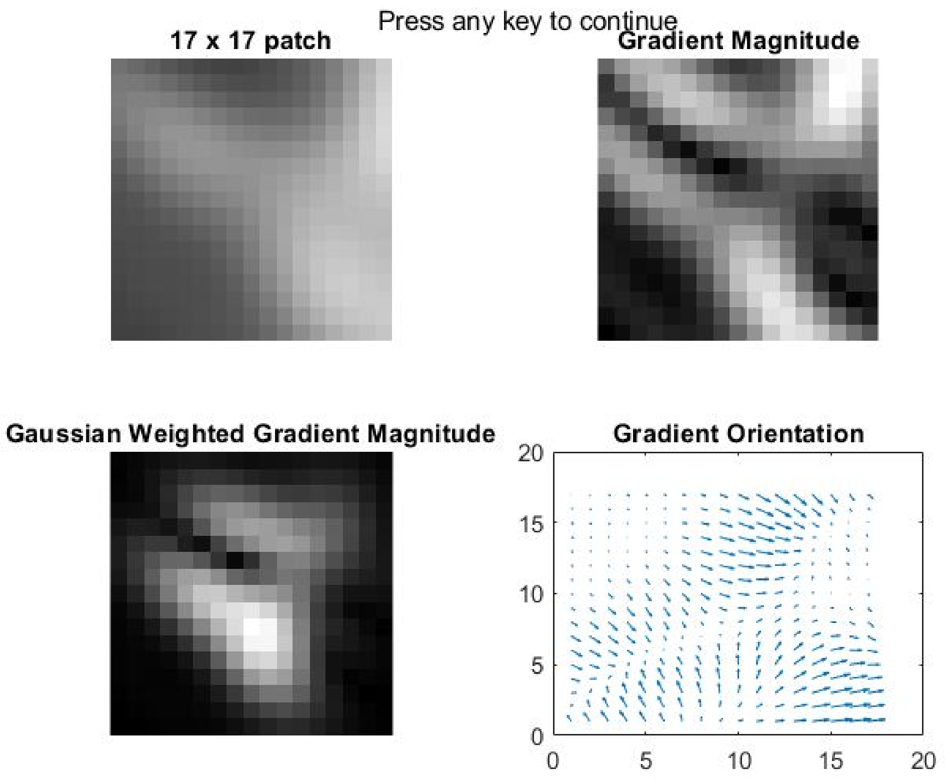
$$||G|| = \sqrt{(G_x)^2+(G_y)^2}$$

And $$\arg(G)=\tan^{-1}(G_y/G_x)$$

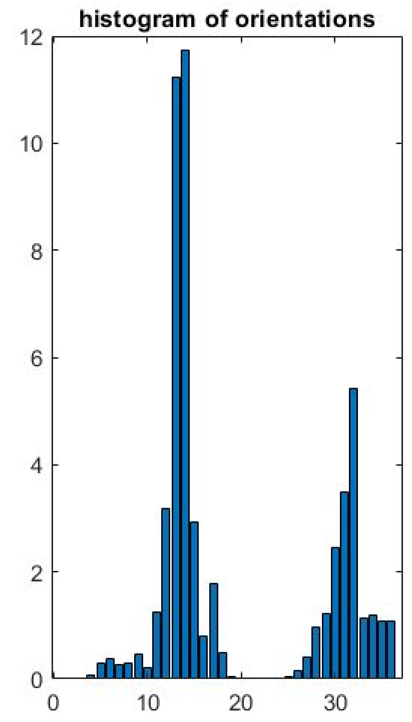
Then a gaussian mask of the same size as the neighbourhood i.e 17*17 was taken with $\sigma = (N-1)/4$ =4 and was used as a weighting mask over the Gradient magnitude matrix. MATLAB functions imgradient() was used for computing the gradient. The various matrices for three random patches are shown below

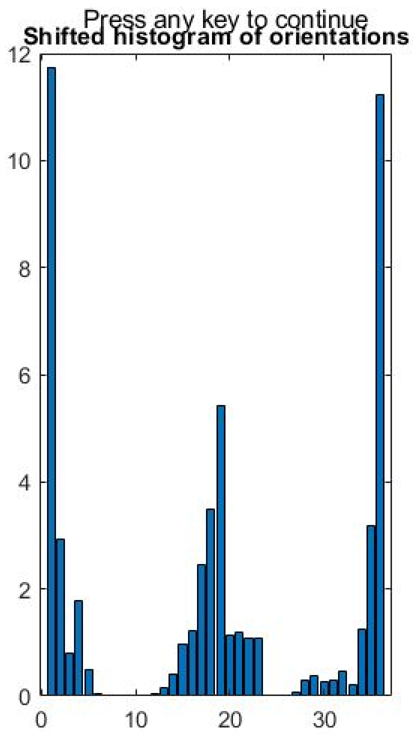

Press any key to continue
17 x 17 patch
Gradient Magnitude
Gaussian Weighted Gradient Magnitude
Gradient Orientation
0
5
10
15
20

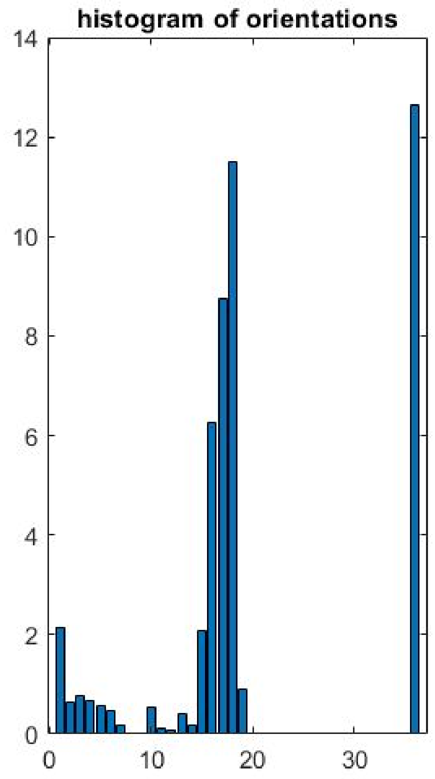

Press any key to continue
17 x 17 patch
Gradient Magnitude
Gaussian Weighted Gradient Magnitude
Gradient Orientation
0
5
10
15
20

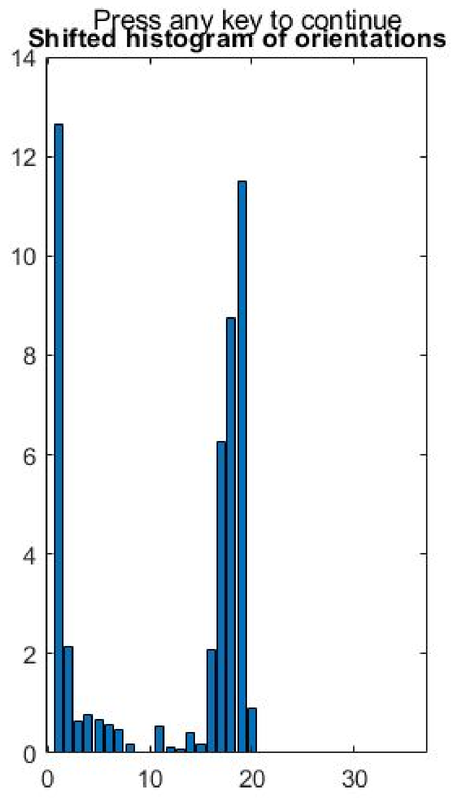


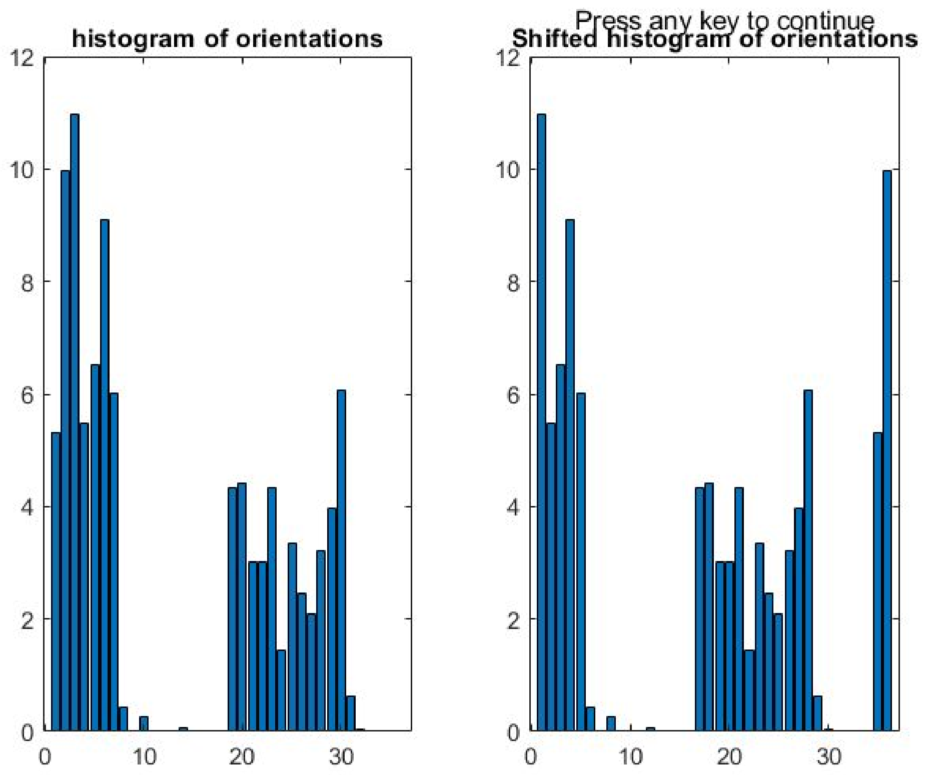
5. Now these gradients were used for computing feature histogram or the histogram of orientations of gradients. The histogram had 36 bins for every angle in 10 degree intervals. Now for each patch a 36 dimensional feature vector was to be constructed such that,for each point in the neighbourhood that had orientation in a bins range, the corresponding Gaussian Weighted gradient magnitude was added to that bin in order to construct the sift vector for a given keypoint location. Once this histogram was calculated, the principle orientation of the vector was identified as the bin having the largest value or peak and the sift vector was shifted counter clockwise such the principal orientation became the first entry in the histogram vector, this is what makes the sift vectors rotationally invariant as we derive canonical representations of the keypoints irrespective of the rotation. The original and shifted histogram vectors are shown for the three random points.


histogram of orientations
0
10
20
30
0
2
4
6
8
10
12


Press any key to continue
Shifted histogram of orientations
0
10
20
30
0
2
4
6
8
10
12

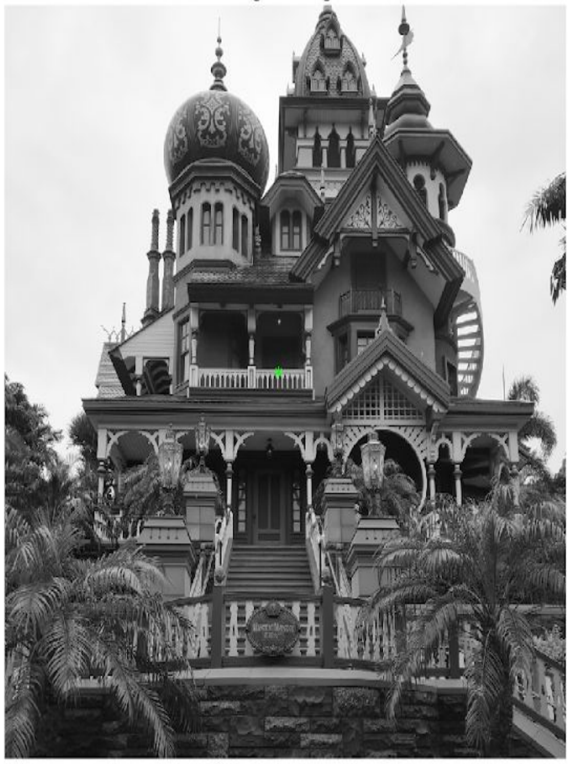

histogram of orientations
0
10
20
30
0
2
4
6
8
10
12
14


Press any key to continue
Shifted histogram of orientations
0
10
20
30
0
2
4
6
8
10
12
14

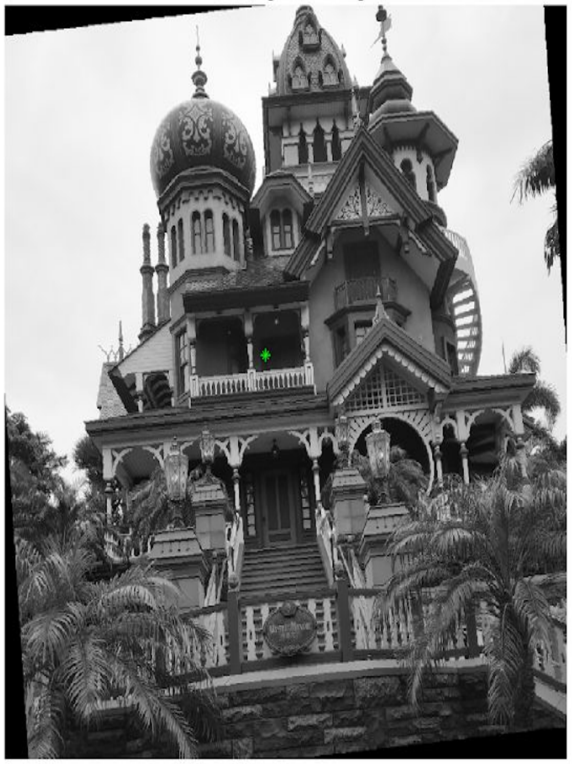


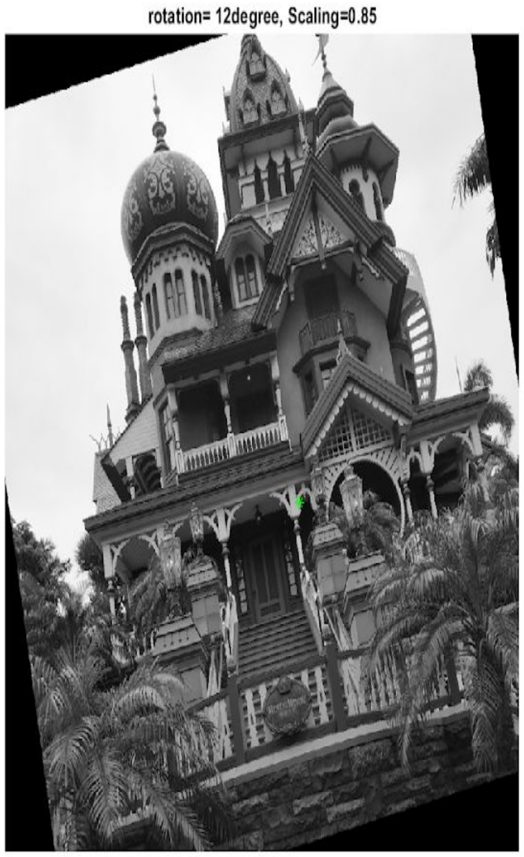
6. Now in order to test the robustness of sift vectors, the images were transformed in order to match features. An arbitrary point near the center of the image was taken and the image was rotated about the point and then it was scaled. Two instances are shown below, the center was offset by 12pts from (512,512) to (500,500)

Original Image



rotation= 5degree, Scaling=1.05





7. Once we have transformed the image, now a brute force feature matching was carried out between the original and transformed image. In order to make the process simpler and computationally less expensive, a smaller region of interest was taken from the original image and the transformed image, the ROI taken was of size 512 around the center the offset center. The two images were then passed through the whole pipeline of steps above to calculate the sift feature vectors. Once these histogram vectors were calculated for both the image, a similarity function was implemented to match the vectors. This similarity vector is the Bhattacharya Coefficient given by

$$\sum_i \frac{\sqrt{H_1(i) \cdot H_2(i)}}{\sqrt{\sum_i H_1(i) \cdot \sum_i H_2(i)}}$$

Where H_1 and H_2 are two given histograms.
Higher the coefficient, higher is the match between two histograms. Now while matching a lot of noisy matches were recorded because of recurrent geometrical structure of the image. Hence in order to avoid this , for each keypoint in the original image a local neighbourhood around it was chosen in the transformed image in order to focus the feature matching. The results obtained are shown below for the ROIs in two transformormed images.

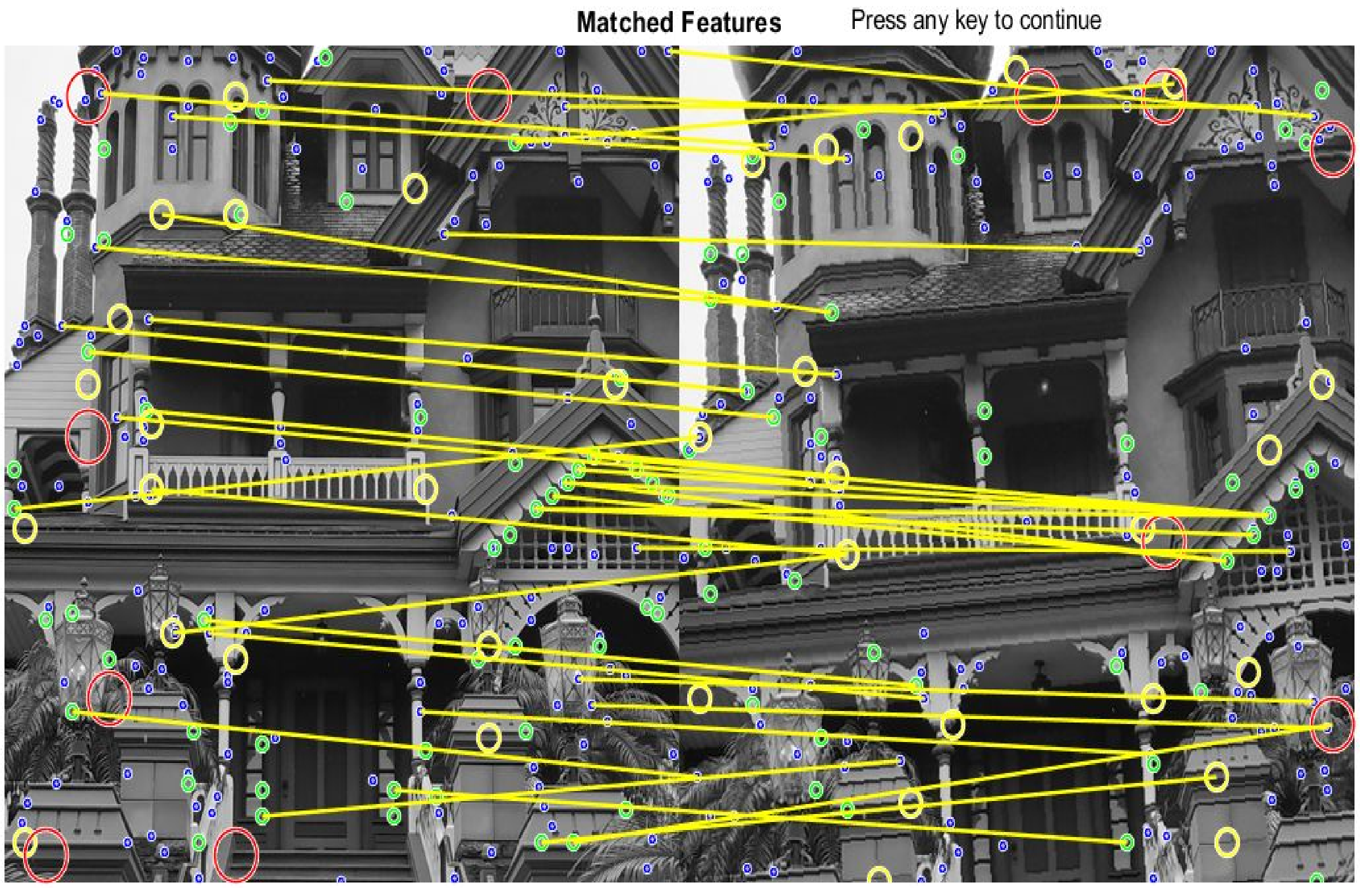


Figure Image showing matched features with Bhattacharya Coefficient>0.95 for rotation=5 and scaling=1.05

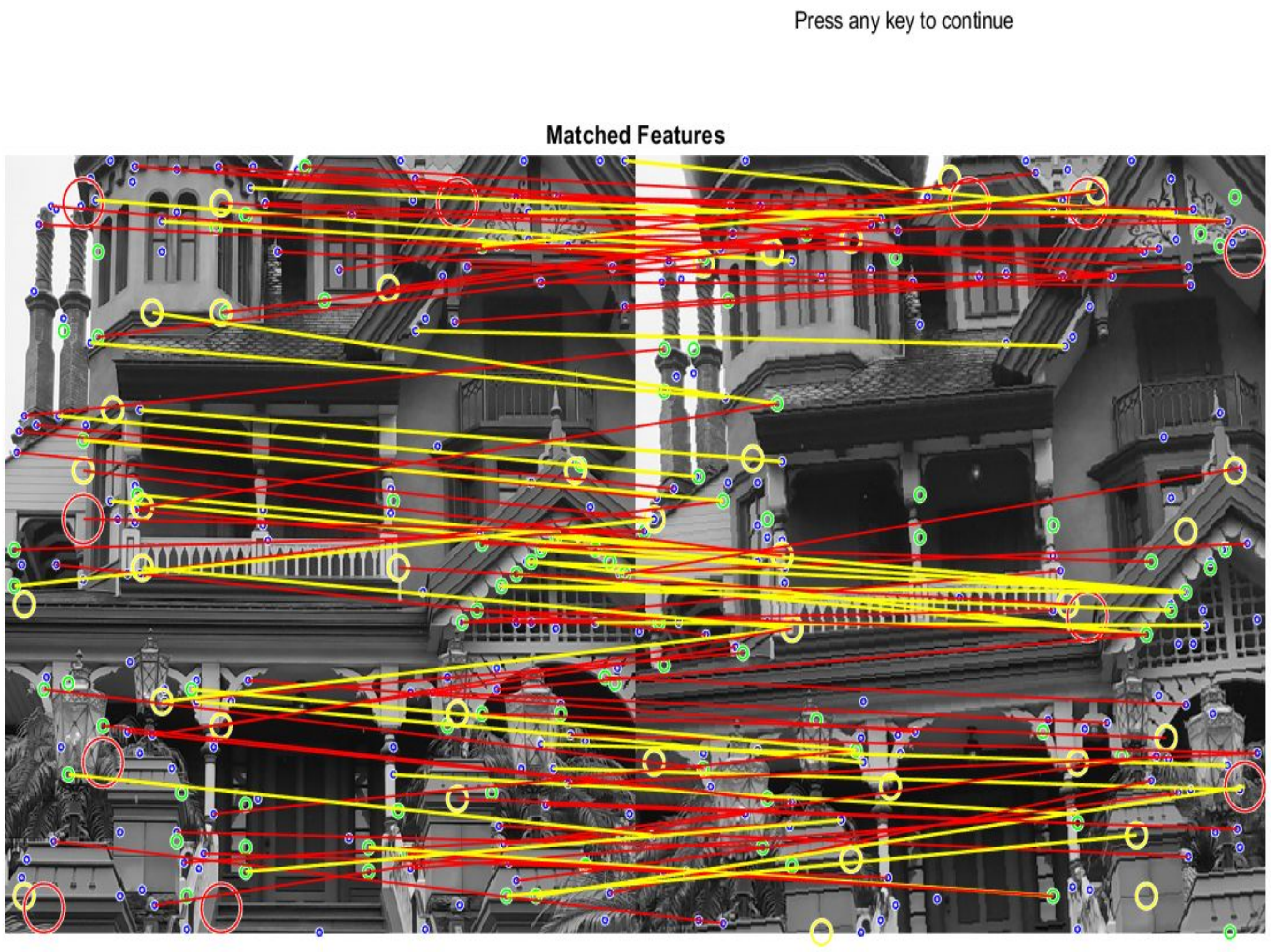


Figure Image showing feature matches for bhattacharya coefficient 0.9 to 0.95 in thin red lines and for bhattacharya coefficient >0.95 in yellow thick line for rotation=5 degree and scaling =1.05

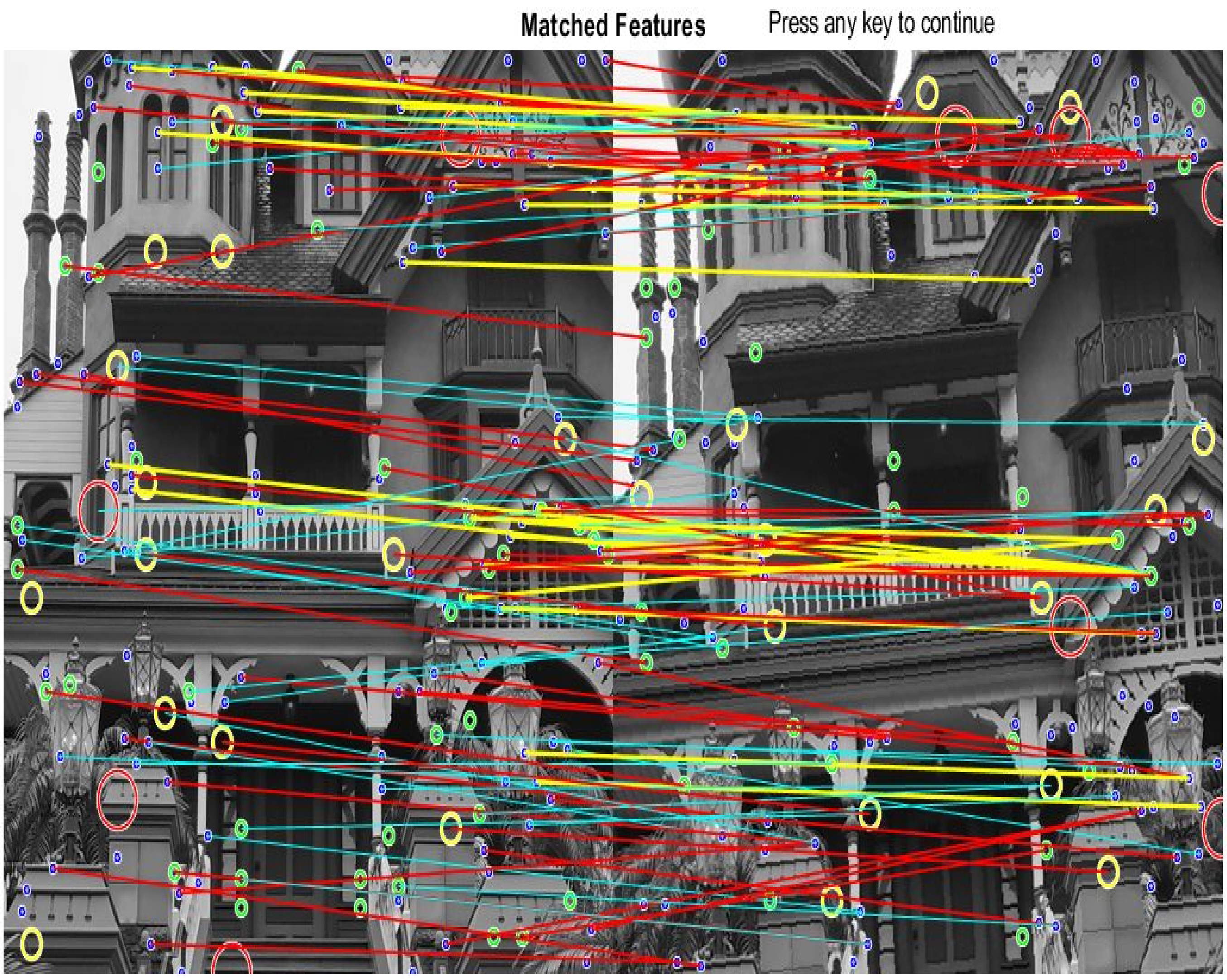


Figure Image showing feature matches for bhattacharya coefficient 0.85 to 0.95 in light blue lines, 0.9 to 0.95 in thin red lines and for bhattacharya coefficient >0.95 in yellow thick line for rotation=5 degree and scaling =1.05

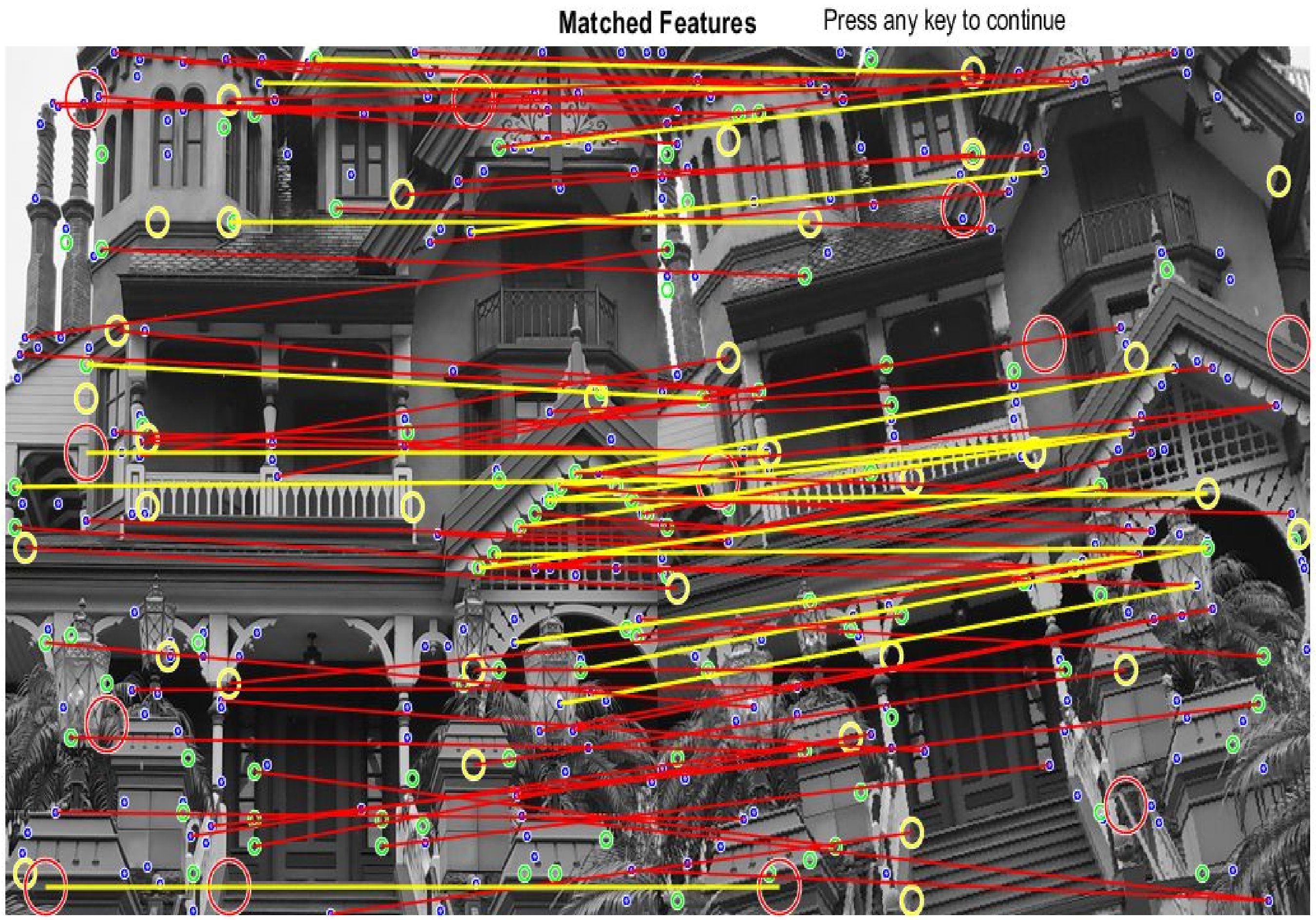


Figure Image showing feature matches for bhattacharya coefficient 0.9 to 0.95 in thin red lines and for bhattacharya coefficient >0.95 in yellow thick line for rotation=12 degree and scaling =0.85

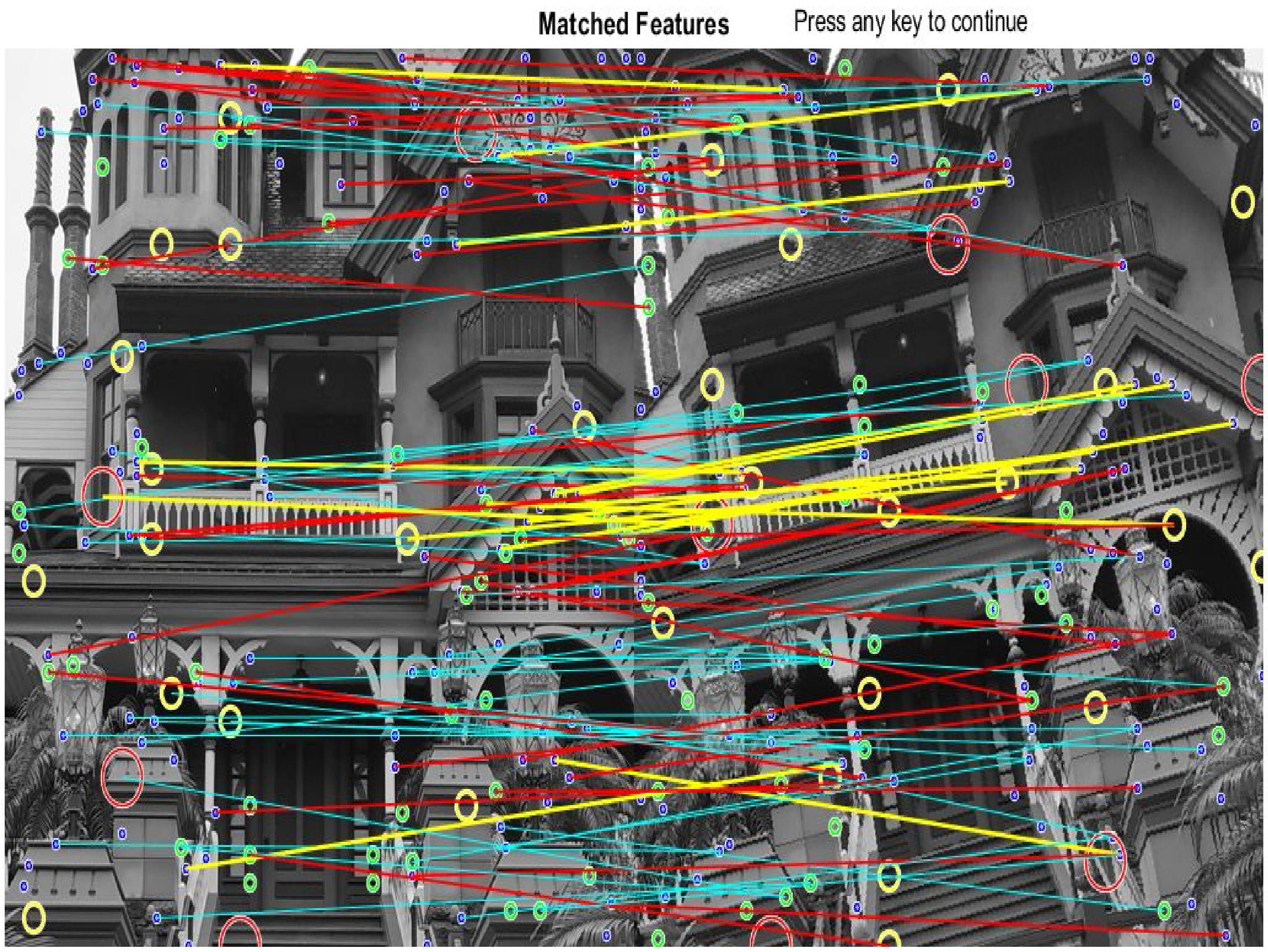


Figure Image showing feature matches for bhattacharya coefficient 0.85 to 0.95 in light blue lines, 0.9 to 0.95 in thin red lines and for bhattacharya coefficient >0.95 in yellow thick line for rotation=12 degree and scaling =0.85

8. The above feature matching technique is very primitive and brute force. It can however be made much more robust by a number of strategies.
   The matching is carried out in isolation, i.e all the feature vectors are matched independently. A much more robust way of implementing would be to calculate pairwise or groupwise distance estimates of keypoints that lie spatially close to each other. This is because the relative positions of two points varies linearly or says same when transformations like scaling or rotation is applied. Carrying out feature matching this way will help alleviate the problem of noisy matches to distant yet similar keypoints on the

image for which we had to limit the neighbourhood around a keypoint, thus reducing cases of false matches.
But carrying out this kind of feature matching assumes that the keypoints in both the original and transformed images remains the same or pairs of keypoints co-occur in the Image and no non linear transformation is introduced in the image, the external Environment such as lighting and noise in case of both the images remains same.