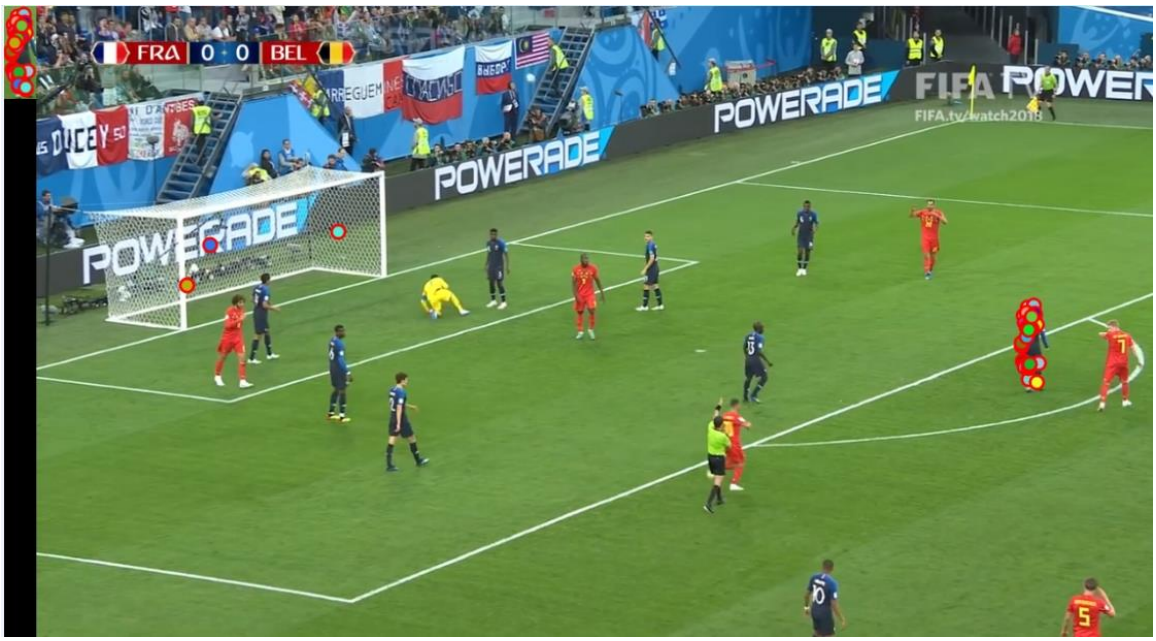

Local Feature Matching

PROJECT 2

Alaa Hesham | CIE 552

Introduction

In this project, I have created a local feature matching algorithm and attempt to match multiple views of real-world scenes. There are hundreds of papers in the computer vision literature addressing each stage. We will implement a simplified version of SIFT.

Manual

The only thing that you need to do is to uncomment the pair of images that you want to test project on, hit run button, and it will work just fine.

```
%    image1 = imread('../data/Notre Dame/921919841_a30df938f2_o.jpg');
%    image2 = imread('../data/Notre Dame/4191453057_c86028ce1f_o.jpg');
%    eval_file = '../data/Notre Dame/921919841_a30df938f2_o_to_4191453057_c86028ce1f_o.mat';

%  image2 = imread('../data/player/7-playerfull.PNG');
%  image1 = imread('../data/player/7player.PNG');

 %image1 = imread('../data/agra_fort.jpg');
 %image2 = imresize(image1,0.5);


% %This pair is relatively easy (still harder than Notre Dame, though)
% image1 = imread('../data/Mount Rushmore/9021235130_7c2acd9554_o.jpg');
% image2 = imread('../data/Mount Rushmore/9318872612_a255c874fb_o.jpg');
% eval_file = '../data/Mount Rushmore/9021235130_7c2acd9554_o_to_9318872612_a255c874fb_o.mat';

%This pair is relatively difficult
% image1 = imread('../data/Episcopal Gaudi/4386465943_8cf9776378_o.jpg');
% image2 = imread('../data/Episcopal Gaudi/3743214471_1b5bbfda98_o.jpg');
% eval_file = '../data/Episcopal Gaudi/4386465943_8cf9776378_o_to_3743214471_1b5bbfda98_o.mat';
```

Algorithm and implementation HIGHLIGHTS:

Harris corner detection to get interest points:

First convolve sobel filter with Gaussian filter, to apply smoothing while calculating derivatives, the normal step was to get sobel filter apply it on image to calculate derivatives and then apply Gaussian filter .However, since convolution is associative then we could apply Gaussian filter on sobel filter and the resultant filter will be applied on image since this computationally is cheaper, and has smaller running time.

The following Steps are:

1. Compute x and y derivatives of image
$$I_x = G_\sigma^x * I \quad I_y = G_\sigma^y * I$$
2. Compute products of derivatives at every pixel
$$I_{x2} = I_x.I_x \quad I_{y2} = I_y.I_y \quad I_{xy} = I_x.I_y$$
3. Compute the sums of the products of derivatives at each pixel
$$S_{x2} = G_{\sigma\prime} * I_{x2} \quad S_{y2} = G_{\sigma\prime} * I_{y2} \quad S_{xy} = G_{\sigma\prime} * I_{xy}$$
4. Define at each pixel (x, y) the matrix
$$H(x,y) = \begin{bmatrix} S_{x2}(x,y) & S_{xy}(x,y) \\ S_{xy}(x,y) & S_{y2}(x,y) \end{bmatrix}$$
5. Compute the response of the detector at each pixel
$$R = Det(H) - k(Trace(H))^2$$
6. Threshold on value of R. Compute nonmax suppression.

Sift to get features:

In SIFT, I have done the following:

1 –Calculated the gradient of an image patch surrounding the interest point detected earlier, it is 16x16 patch.

2- Using matlab's histcounts function to bin these gradient directions.

3-I have created histogram bins for 4x4 pixels from the 16x16 image patch, and have binned them into 8 bins each representing 45 degrees.

*As a result, a hisogram of 8 values for each of the 4x4 pixel bunch. Thus I have 4*4*8 = 128 values to describe each interest point. Hence my feature vector's length is 128.*

knnsearch using k-d tree:

To organize points into k-dimentional space, I have used k-d tree which is a space-partitioning data structure.

The steps could be summarized as the following:

Using Matlab's built in function ***k-d tree searcher and*** by applying knnsearch to get two nearest neighbours , and then threshold ratio of nearest neighbout distance to second nearest neighbor .

Results :

In order to get high accuracy , I made feature matching exclusive to first top hundred matches

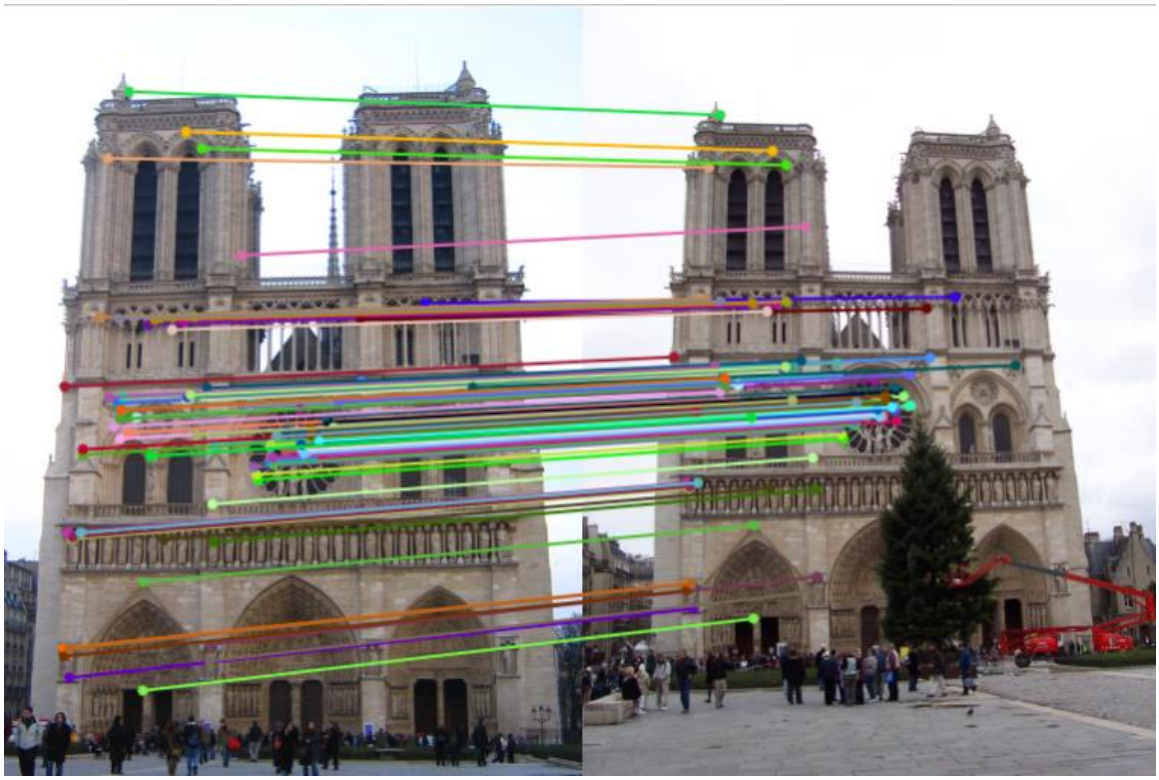

Figure 1 first hundred match

The accuracy is **100%** when I use only first **hundred** match

When I use all points (good matches and bad matches), the accuracy is **31.15%**

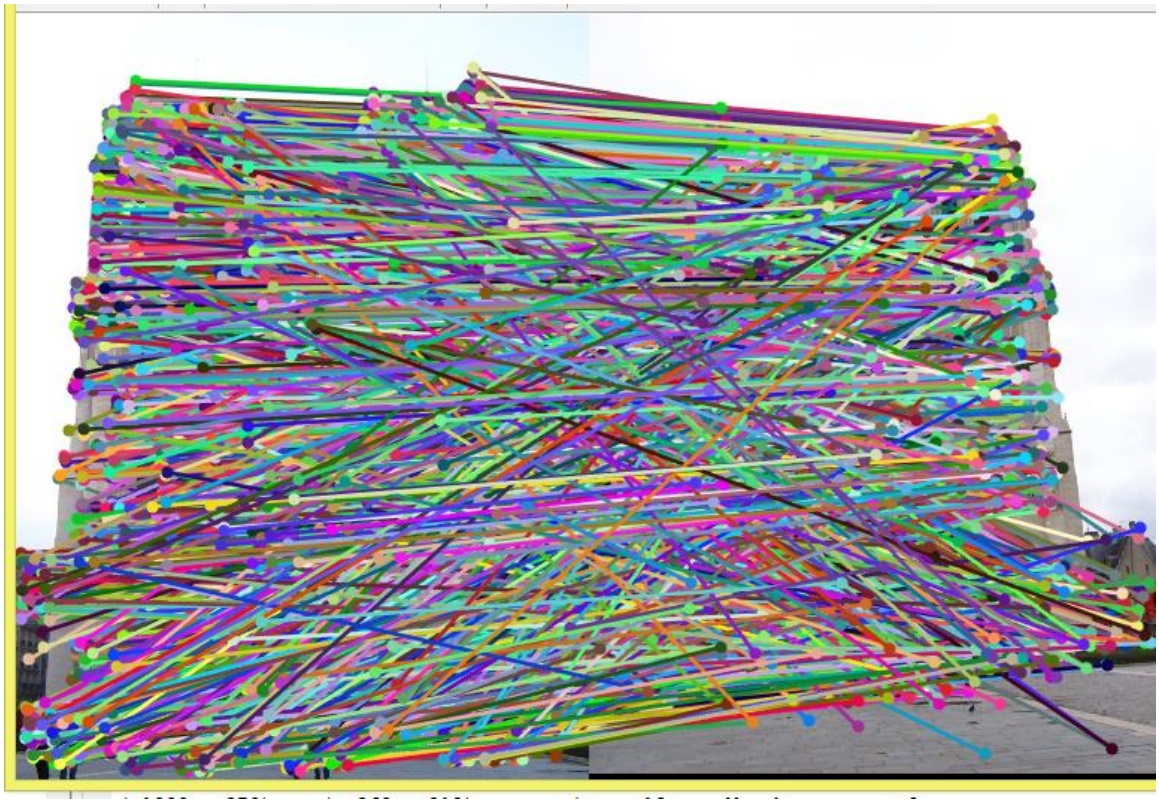

Figure 2 all matches are considered

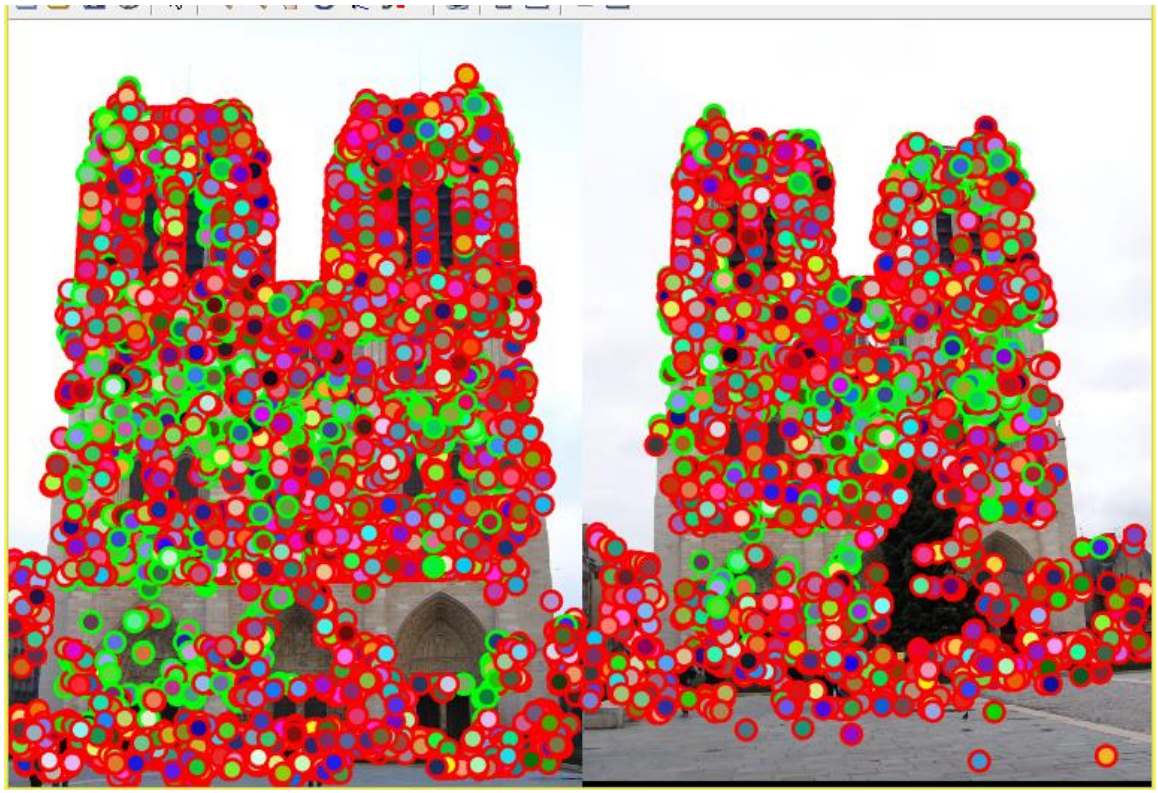

Figure 3 All matches are considered

Accuracy when first top hundred matches are used is 99%

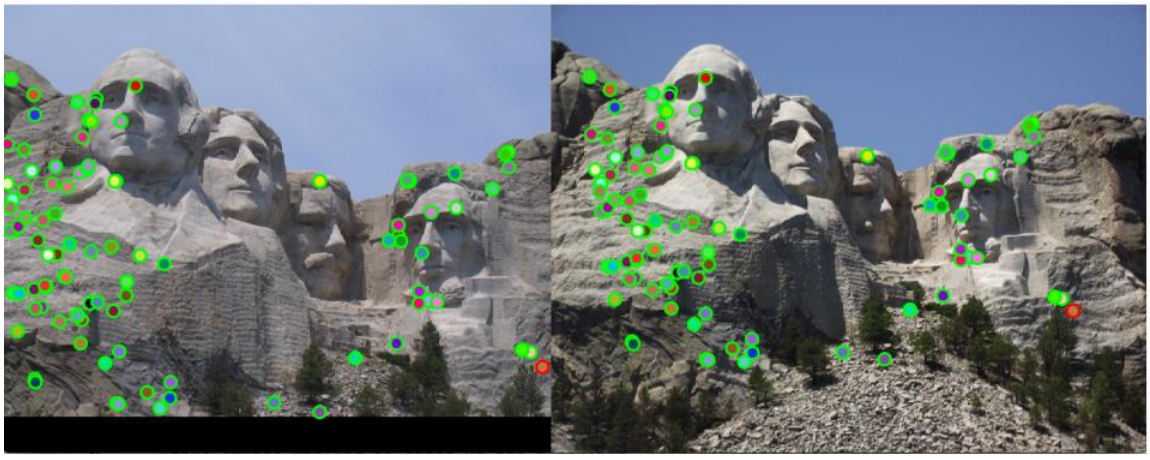

Figure 4 first top hundred matches

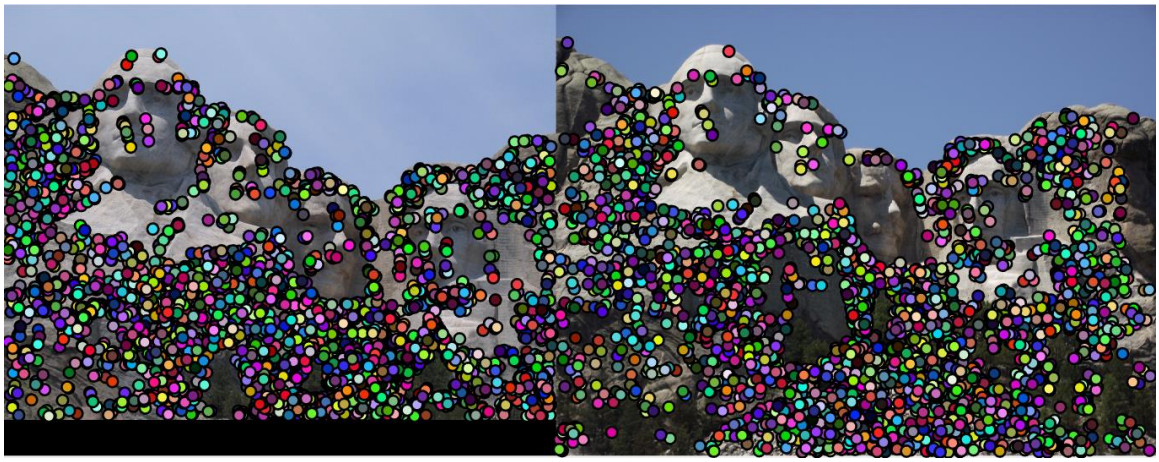

Figure 5 all matches are used

When all matched are used , accuracy is 31.71%

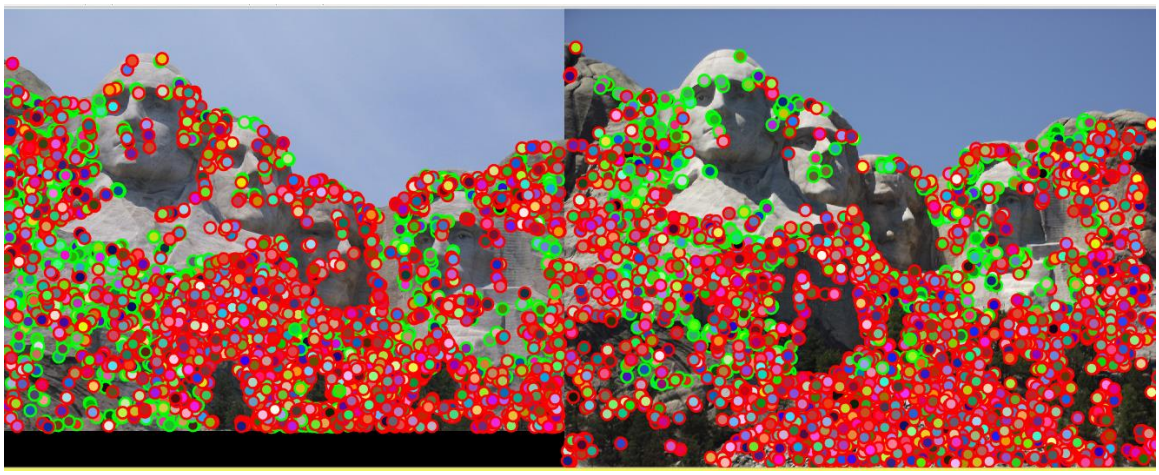

Figure 6 all matches are considered

For this image pair , accuracy was the worst .It achieved 5% on top 100 match , and 1.77% when all matches are considered .

Figure 7 Top 100 matches

Figure 8 when all matches are considered