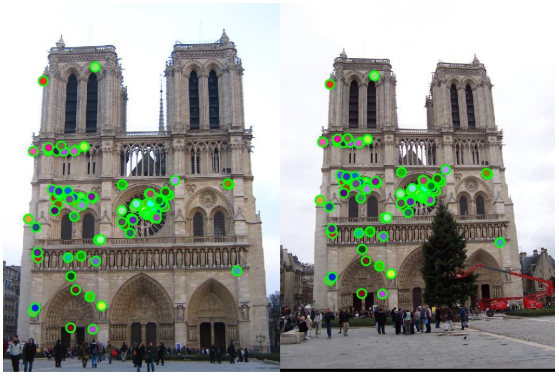


PROJECT 2

Local Feature Matching

ABSTRACT

Extracting features from different images and matching them with each other has many applications such as object recognition and robot navigation. In this project, Harris corner algorithm along with SIFT algorithm are implemented to accomplish the task of local feature matching.

Ahmed Wael - 201500862

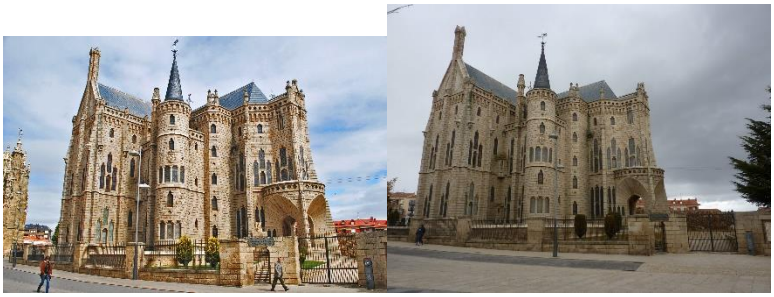
CIE552 – Computer Vision

Contents

1. Dataset

The dataset is simply 6 images that can be considered as 3 pairs of images as following:

1. Notre Dame de Paris
2. Mount Rushmore
3. Gaudi's Episcopal Palace



The dataset varies in difficulty as the first one is the easiest one, the second one is more difficult, and the last one is the hardest.

2. Harris corner

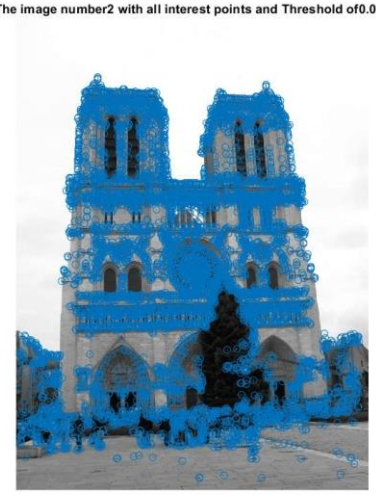
In order to extract the most important features of the image, Harris corner algorithm is used. This algorithm detects the corners -interest points- using the following procedure:

1. Algorithm

1. Compute the derivatives in x and y directions Ix ,Iy . This was done using sobel
   filters in both directions with gaussian filter.
2. Compute products of derivatives at every pixel Ix_2 ,Iy_2 , I_{xy} . This was done in a vectorized manner to speed up the calculations.
3. Compute the sums of the products of derivatives at each pixel S_{x2}, S_{y2}, S_{xy}. This was multiplying the output of step 2 with a gaussian filter.
4. Define at each pixel the Harris matrix. This was also done in a vectorized format.
5. Compute the response of the detector at each pixel. Once again, this was done using vectorized code. $R = Det(H) – k\ (Trace(H))^2$. k is set to 0.04 as suggested in SIFT algorithm paper.
6. Compute non-max suppression. Loop over all the points and do the following:
   a. Choose the current point R value.
   b. Get the 3*3 window values around the current point.
   c. If the current point R value is bigger than a specified threshold (hyperparameter to tune)
      i. If the current point R value is the biggest value in the window
         1. Concatenate the feature vectors in both x and y directions with the current row and column values.

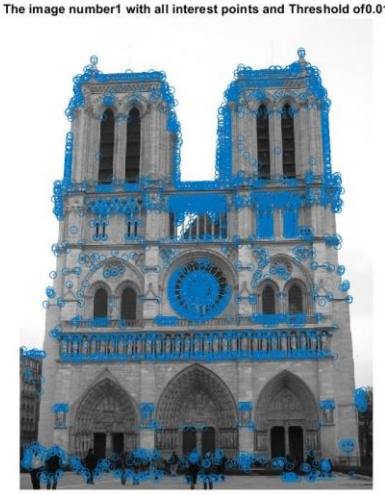
2. Results

The image number2 with all interest points and Threshold of0.002



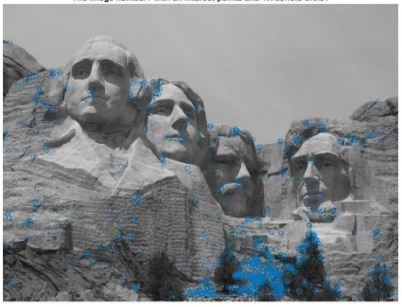
The image number1 with all interest points and Threshold of0.002

The image number1 with all interest points and Threshold of0.01

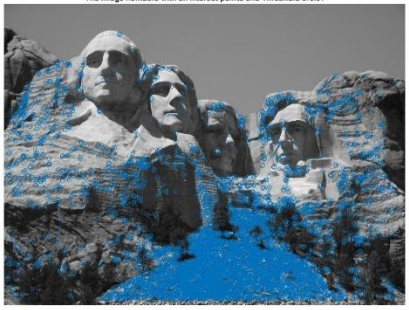


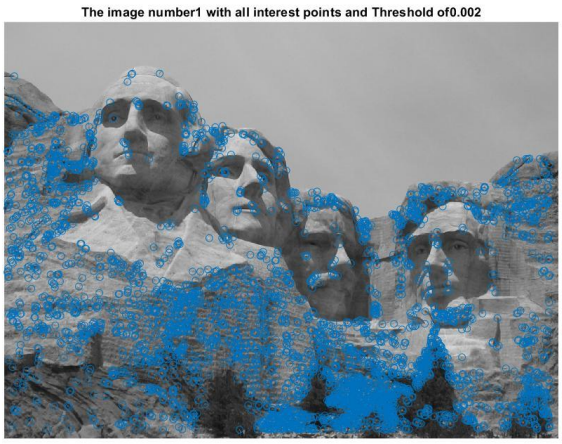
The image number2 with all interest points and Threshold of0.01

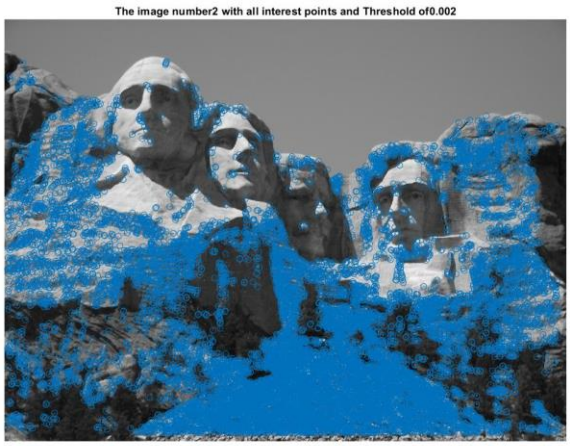
The image number1 with all interest points and Threshold of0.01



The image number2 with all interest points and Threshold of0.01







3. Discussion

It’s clear that increasing the threshold makes the algorithm captures less interest points. This is intuitive because when the threshold is big, less points will satisfy the condition to be considered as an interest point.

Getting more interest points would increase the chance of this interest point to be an accepted feature. This will increase the accuracy of the algorithm at the end. However, getting more interest points would require more memory and time. After experimentation, from 2,000 to 15,000 points are enough to extract the features with SIFT algorithm.

3. SIFT Descriptor

1. Algorithm

1. Compute the image orientation using sobel and gauss filters.
2. Compute the dominant orientation as follows:
    a. Loop over all interest points and do the following:
        i. Create a window of 256 values around the key point.
        ii. Compute the histogram for 36 values using quantization.
        iii. Subtract the dominant orientation (the most frequent orientation value) from the key point orientation.
3. Quantize the directions into only 8 directions.
4. Compute the feature vector as follows:
    a. Start from 7 values before the current key
    b. Start from 7 values before the current key value in both row and column.
    c. Loop for 16 times (the window size) as follows:
        i. If the iteration number is divisible by 4:
            1. Increase the start row value by 4.
            2. Increase the end row value by 3.
            3. Reset the start column value.
            4. Reset the end column value.
        ii. If the iteration number is not divisible by 4:
            1. Increase the start column value by 4.
            2. Increase the end column value by 3.
    d. Compute the window to be the indices of the image orientation after quantization with the start and end values for both columns and rows.
    e. Compute the histogram for the 8 bins.
    f. Normalize the feature vector to 1.
    g. Clip all the values in the feature vector bigger than 0.2 to solve the illumination problem.
    h. Normalize the feature vector to 1 again.

5. Repeat step 4 for all the key points.

2. Results

The SIFT descriptor output a matrix k*n where k is number of interest points and n is the feature vector for each interest point.

3. Discussion

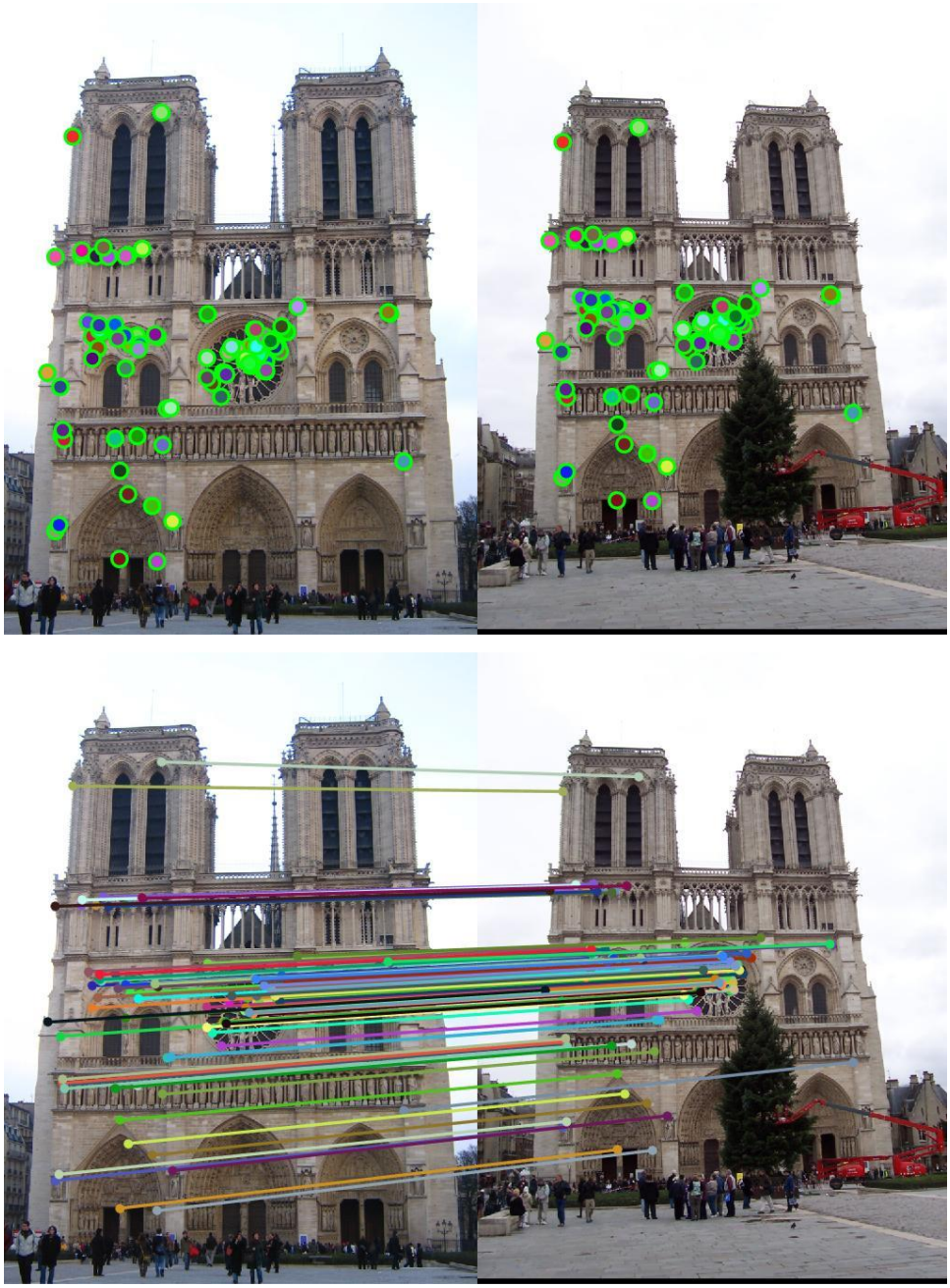
SIFT descriptor is orientation invariant as the dominant orientation was calculated. Also, it is illumination invariant as it's normalized and clipped.

4. Features Matching

1. Algorithm

1. Number of features is the minimum number of features out of the two feature matrices.
2. Use MATLAB build-in function (**bonus**) to get nearest two neighbors for each interest point in the first feature matrix from the second feature matrix.
3. Compute the nearest neighbor distance ration (NNDR) by using element-wise division between the first nearest neighbor vector and the second nearest neighbor vector.
4. Loop over the number of features and do the following:
    a. If the current value of the NNDR vector is less than a specific threshold (0.8):
        i. Add this value to the good matches vector.
        ii. Add the current index of feature1 matrix to the matches first column.
        iii. Add the current index of the nearest neighbor matrix to the matches second column.
5. Sort the good matches so the most confident matches are using in calculating the accuracy.

4. Results



Uniqueness: Pre-merge: 100 Post-merges: 99

99 total good matches, 0 total bad matches.

100.00% precision

99.00% accuracy (top 100)

Uniqueness: Pre-merge: 100 Post-merges: 100

99 total good matches, 1 total bad match.

99.00% precision

99.00% accuracy (top 100)

Uniqueness: Pre-merge: 100 Post-merges: 99

2 total good matches, 97 total bad matches.

2.02% precision

2.00% accuracy (top 100)

5. Discussion

As we can see, the algorithm is superb and achieves 100% accuracy and precision for all kind of images. However, in extreme cases such as the third pair, the two images have different scales. Thus, the algorithm fails as it needs to be scale invariant. The original algorithm is scale invariant, however in our implementation this is not a requirement.

5. Conclusion

SIFT algorithm which was introduced back in 1999 and enhanced in 2004 is considered the most famous and robust algorithm for feature extraction. This is because it is invariant to any changes in the images such as orientation, scaling, illumination or any other thing.

In this project, Harris corner algorithm was implemented to get the interest points which was then described by SIFT algorithm and matched at the end using NNDR.