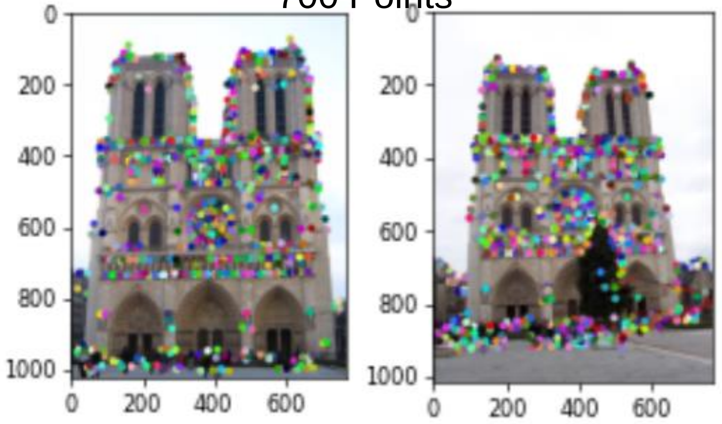
# CS 6476 Project 3

Dong Jae Lee
dlee640
902987855

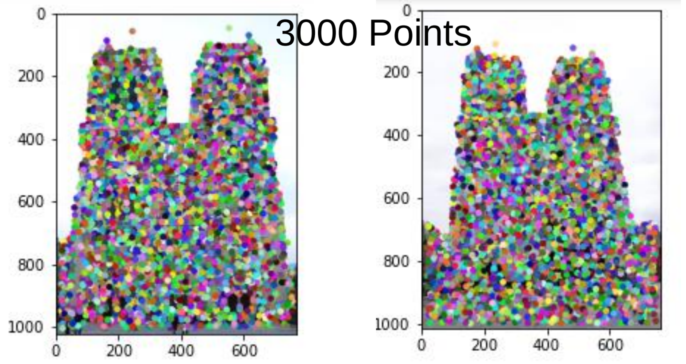
# Gradescope Group Quiz Collaboration Photo

# Part 1: HarrisNet

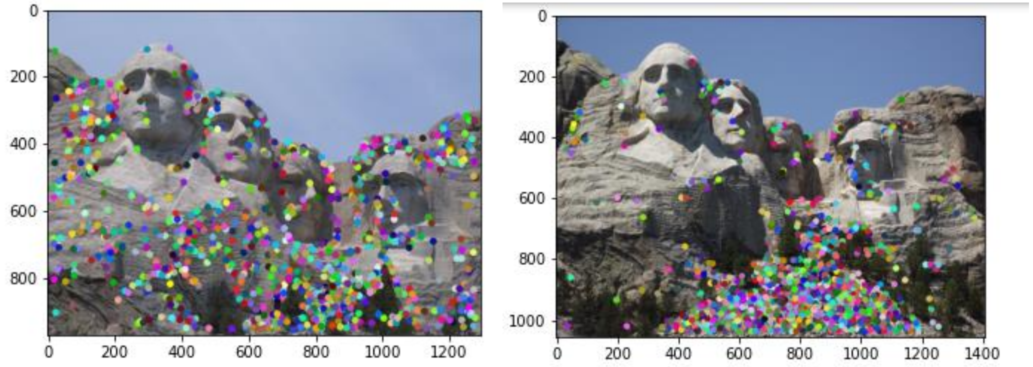
700 Points

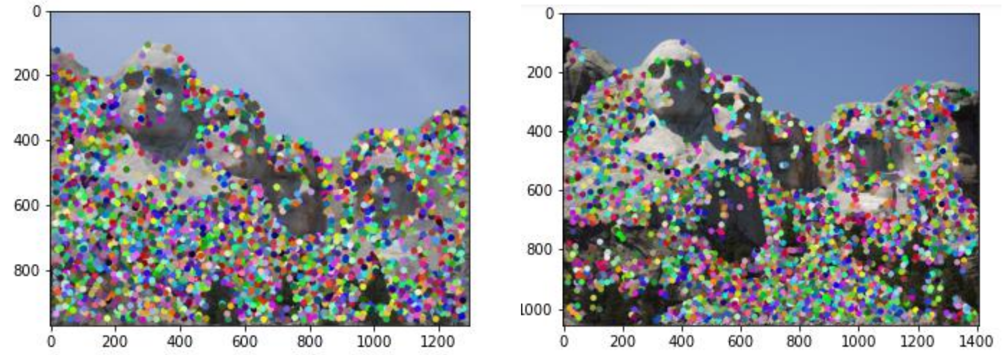


3000 Points



1000 Points

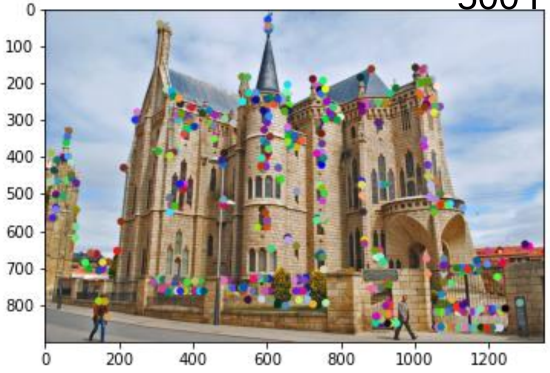


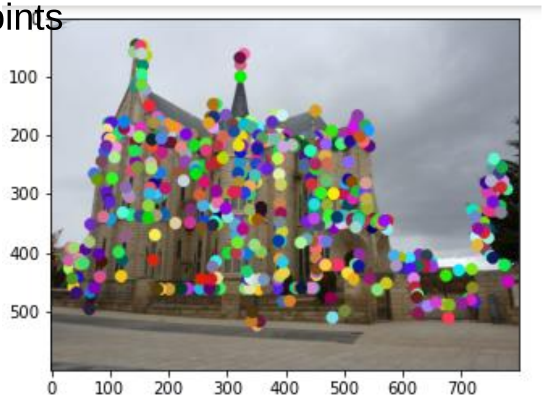


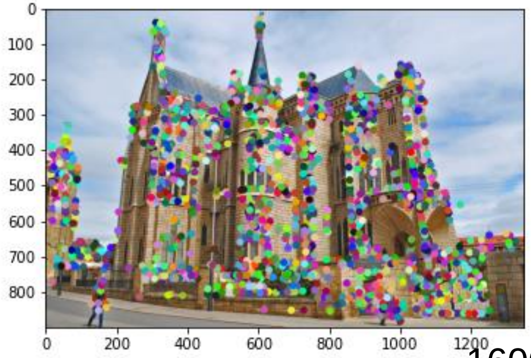
3000 Points

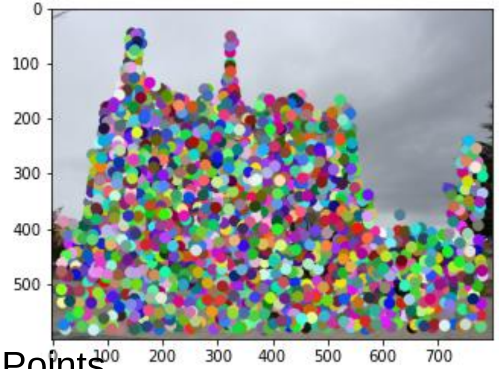
# Part 1: HarrisNet

500 Points









1600 Points

## Part 1: HarrisNet

<Describe how the HarrisNet you implemented mirrors the original harris corner detector process. (First describe Harris) What does each layer do? How are the operations we perform equivalent?)>

The fundamentals of Harris corner detector is using eigenvalue pair to detect where the corner is. This works because the property of eigenvalue pair changes depending on flat region, edge or corner. The detector takes 1. image derivates, 2. square them, 3. apply gaussian filter, 4. obtain cornerness function then finish it off with 5. non-maxima suppression. The HarrisNet I implemented mirrors the original since ImageGradientsLayer() reflects 1 – (finds Ix, Iy of image), ChannelProductLayer() reflects 2 – (finds Ixx, Ixy, Iyy of the image), SecondMomentMatrixLayer() reflects3 – (finds convolution of Ixx, Iyy, Ixy using Conv2d with Gaussian filter), CornerResponseLayer() reflects 4 – (uses Sxx, Sxy, Syy to calculate determinant and trace to calculate R) and NMSLayer() reflects 5 – (performs non maxima suppression using maxpooling & binarizing then multiplying with corner response score).

## Part 2: SiftNet

<Describe how the SiftNet you implemented mirrors the Sift Process. (First describe Sift) What does each layer do? How are the operations we perform equivalent?)>

SIFT stands for scale invariant feature transform. The algorithm transforms an image into a large collection of feature vectors, each of which is invariant to image translation, scaling and rotation. It has also proven to be partially invariant to illumination changes and local geometric distortion. SIFT consists of 4 main steps: 1. scale-space extreme detection, 2. keypoint localization, 3. orientation assignment, 4. keypoint descriptor, and 5. keypoint matching. In the project, SIFTOrientationLayer takes care of preparation for orientation assignment. HistogramLayer takes care of scale-space extrema detection and keypoint localization by creating a weighted histogram pixelwise over the entire image. By doing this, you can find the associated 8-dim histogram representing particular pixel’s contribution to the direction it most aligns with. The SubGridAccumulationLayer takes care of keypoint descriptor step by accumulating feature histograms over 4x4 subgrids. The keypoint matching step I have used is nearest neighbor distance ratio method. This method can vary for different purposes.

## Part 2: SiftNet

- <Explain what we would have to do make our version of Sift rotationally invariant (conceptually)>

To make the version rotationally invariant, we need to find the main orientation of the descriptor and assign that angle to the keypoint so that when comparing with other image, we can find the difference in dominant and other keypoint orientation. Also, it would have to assume no change in viewpoint.

- <Explain what we would have to do to make our version of SIFT scale invariant (conceptually)>

To make the version scale invariant, we need to create scale space by creating a sequence of further convolutions with increasing standard deviations. This allows keypoints to be extracted at different scales and blur levels, allowing the descriptors to be invariant to image scaling & minor changes in perspective.

## Part 2: SiftNet

- <What would happen if instead of using 16 subgrids, we only used 4 (dividing the window into 4 grids total for our descriptor)>

Having 4 subgrids instead of 16 would mean that vector would have 4 subgrids*8bins = 32 elements. Since there are smaller number of subgrids(histograms) to characterize the image of the same size, small changes would be harder to detect from feature comparison.

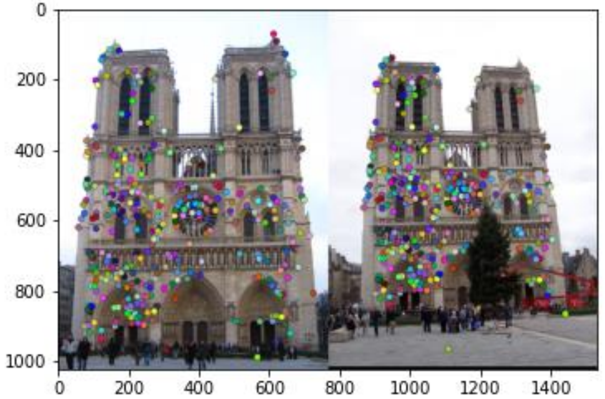
- <What could we do to make our histograms in this project more descriptive?>

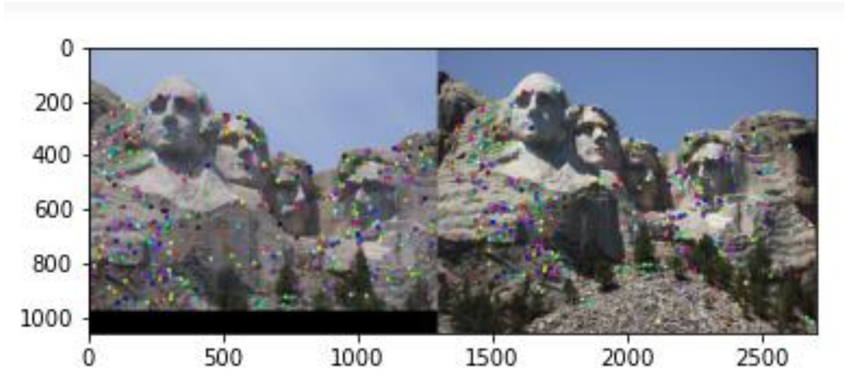
Increase the number of angle bins. Threshold histograms to reduce the influence of large gradients. Use circular normalized patches divided into concentric rings of equal width to make the algorithm more robust to rotation…etc

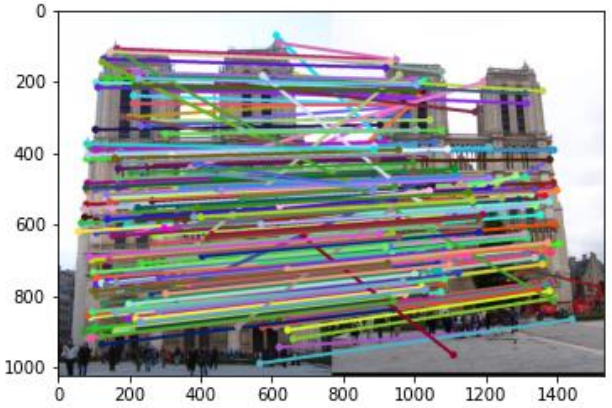
# Part 3: Feature Matching

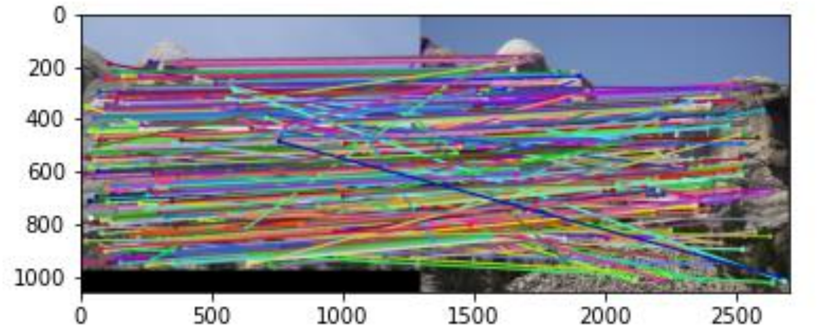
<i otre

D

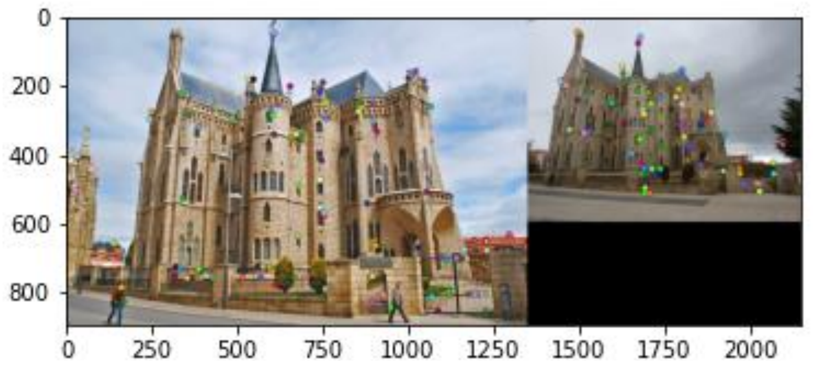


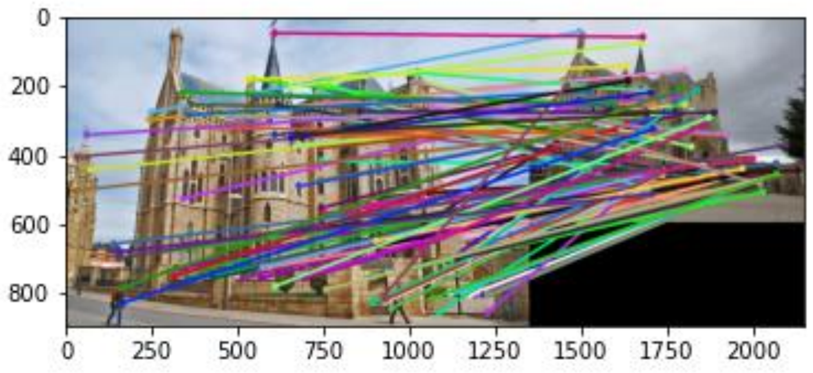






# Part 3: Feature Matching

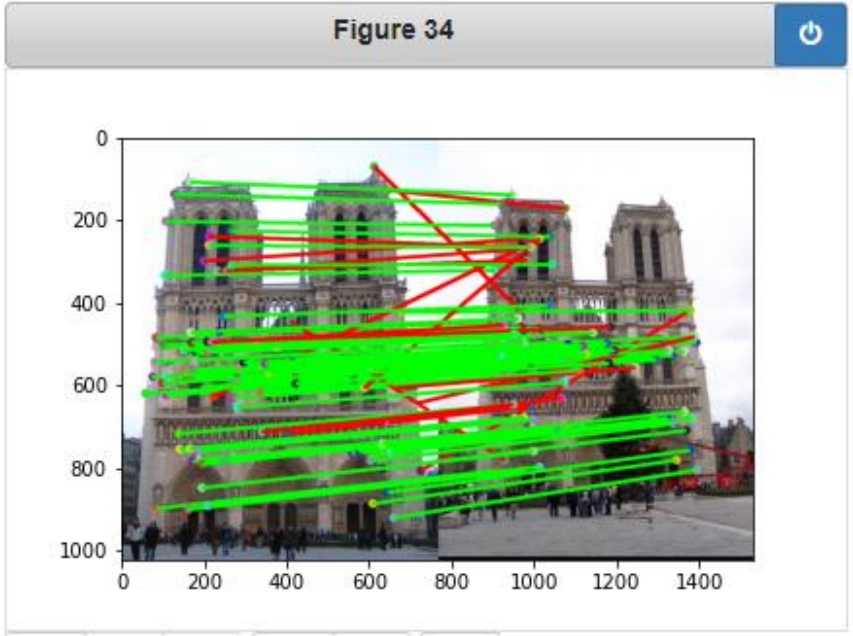




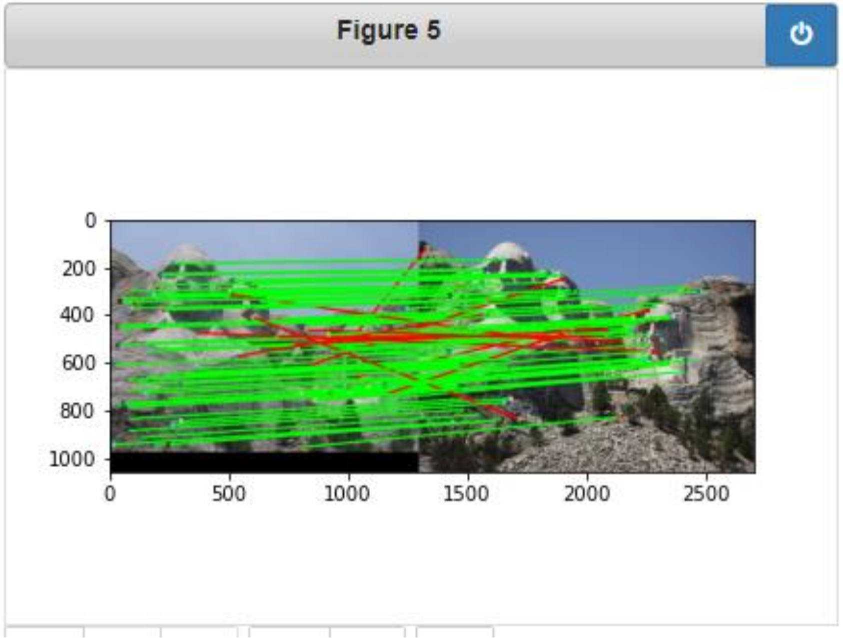
<Describe your implementation of feature matching.>
The feature matching method I used was nearest neighbor distance ratio test. First I calculated the Euclidean distance between feature points. Then, I divide closest distance by second closest distance for each points from feature1. If below certain threshold, I assume that the set of features are ‘good enough’ to use, then match the point that is closest to that of feature1.

## Results: Ground Truth Comparison

```
You found 100/100 required matches
Accuracy = 0.800000
```

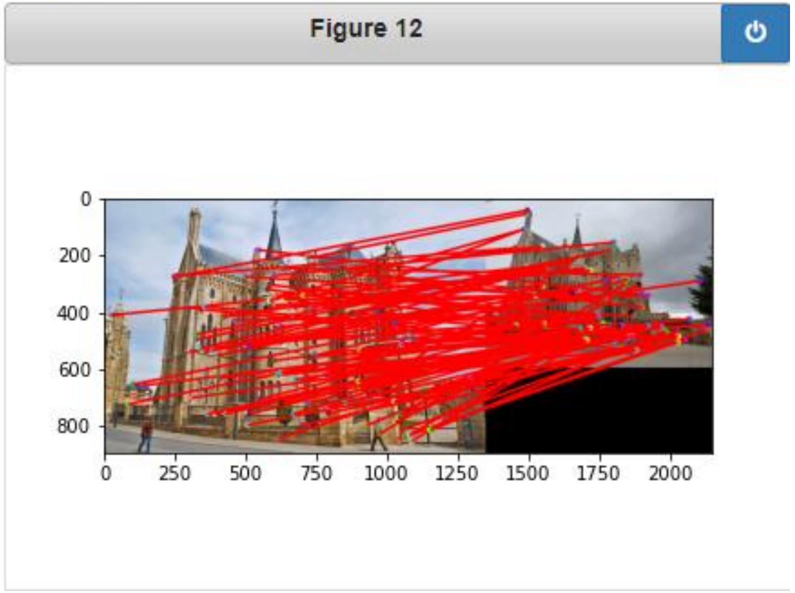


```
You found 100/100 required matches
Accuracy = 0.870000
```



# Results: Ground Truth Comparison

```
You found 100/100 required matches
Accuracy = 0.000000
```



<Insert numerical performances on each image pair here.>

They are attached with the picture.
Notredame: 100/100 matches, 80%
Rushmore: 100/100 matches, 87%
Gaudi: 100/100 matches, 0% ☹

## Results: Discussion

- <Discuss the results. Why is the performance on some of the image pairs much better than the others?>

The performance for rushmore and notredame was much better than that of the gaudi image pair. The gaudi image pair differed greatly in the image size, and the convolution we have implemented to find scale space was limited to work only in image pairs with slight difference.

- <What sort of things could be done to improve performance on the Gaudi image pair?>

Finding larger scale space with varying standard deviation would improve the performance. We can also allow the gradient to contribute to multiple bins like the original SIFT so that histogram can contain more info.

## Conclusions

<Describe what you have learned in this project. Feel free to include any challenges you ran into.>

I had no previous knowledge of corner detection, feature extraction and matching. Learning new concepts along with learning how to implement these algorithms using CNN was definetely a challenge since I have never done anything like it before. Tuning the hyperparameters for feature matching taught me how threshold can influence the number of matches & accuracy.

## Extra Credit: Sift Parameter variations

## Extra Credit: Custom Image Pairs