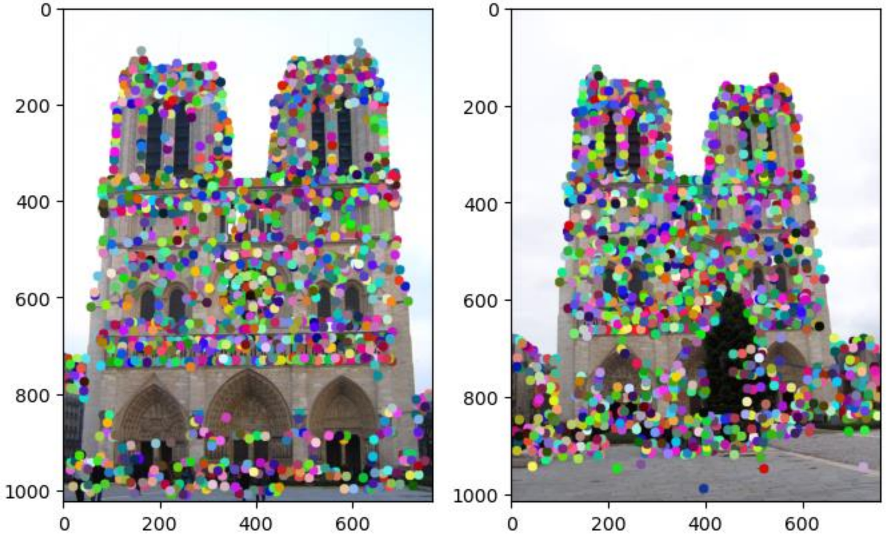
CS 4476 PS2

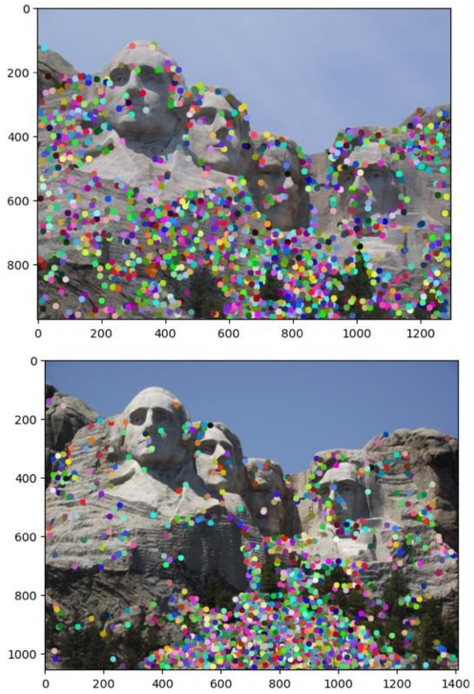
Dong Kyun Lim
dlim70@gatech.edu
dlim70
903590612

1.1: Harris Corner Detector

<insert visualization of Notre Dame interest points from ps2.ipynb here> [2.5 pts]

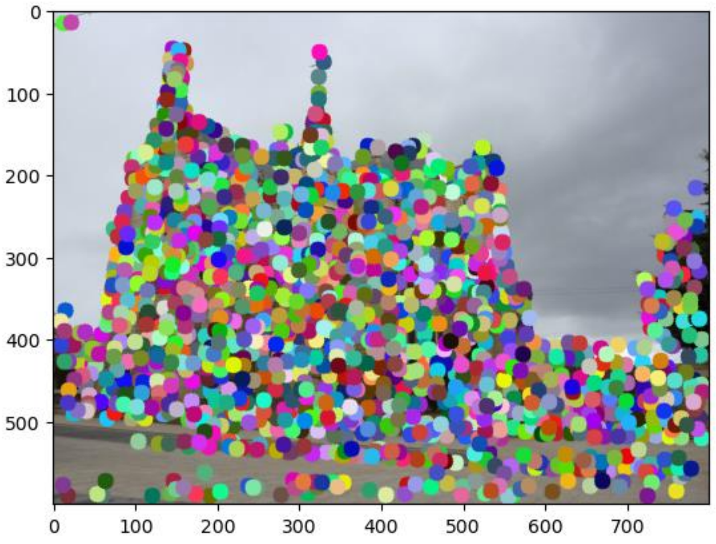


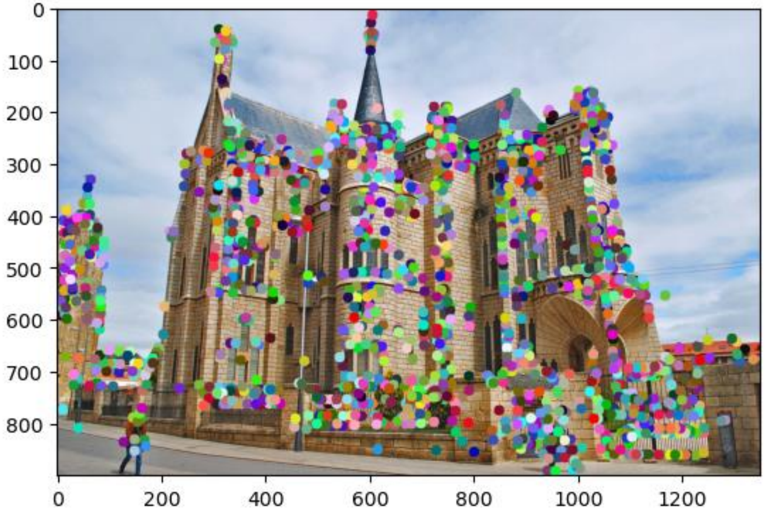
< insert visualization of Rushmore interest points from ps2.ipynb here > [2.5 pts]



1.1: Harris Corner Detector

< insert visualization of Gaudi interest points from ps2.ipynb here > [2.5 pts]





1.1: Harris Corner Detector

- Briefly describe how the Harris corner detector works. [1 pt]

The Harris corner detector is a corner detection algorithm where it identifies points where the changes in intensity are strong. First, it computes gradients, forms a second-moment matrix and then finally calculate a corner response function. Pixels that have high response values are likely corners of the image.

- What does the second_moments() helper function do? [1 pt]

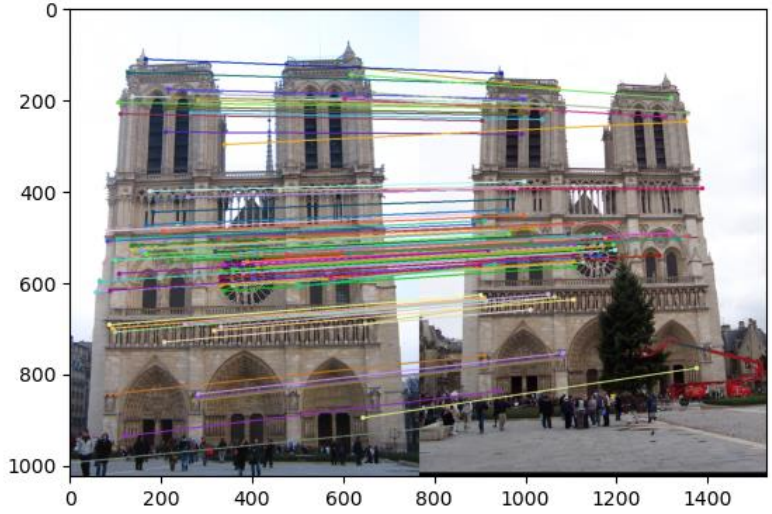
The function smooths squared gradient terms with a Gaussian filter which computes the entries for second-moment matrix. This captures how image intensity varies in both x and y directions.

- What does the corner_response() helper function do? [1 pt]

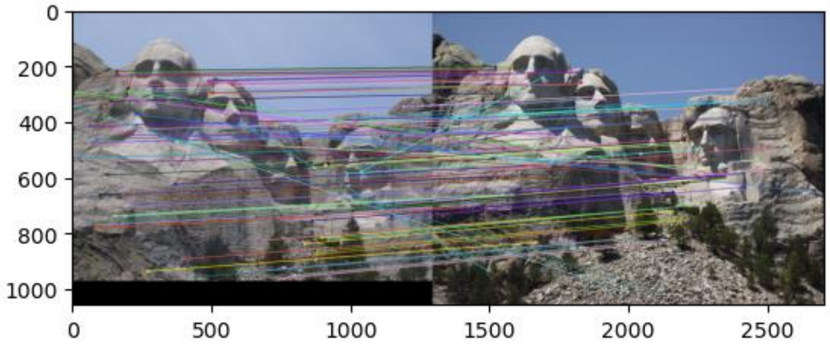
It calculates the Harris response and produces a high score for corners, near zero for flat regions, negative for edges.

1.3: Feature Matching

<insert feature matching visualization of Notre Dame from ps2.ipynb> [2.5 pts]

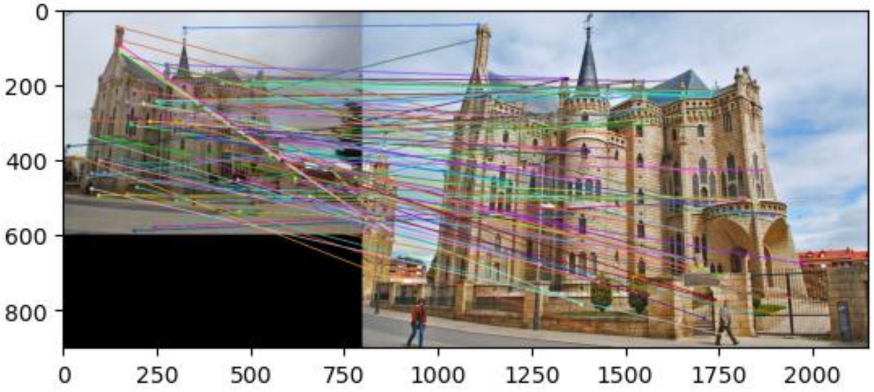


<insert feature matching visualization of Rushmore from ps2.ipynb > [2.5 pts]



1.3: Feature Matching

<insert feature matching visualization of Gaudi from ps2.ipynb > [2.5 pts]

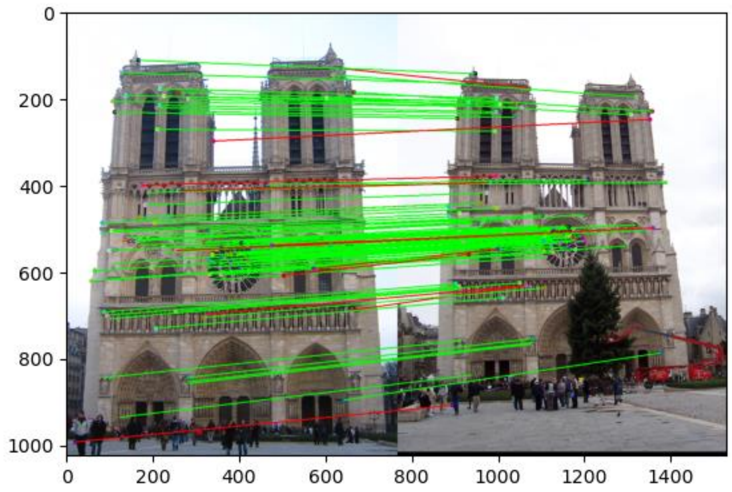


<Describe your implementation of feature matching.> [1.5 pts]

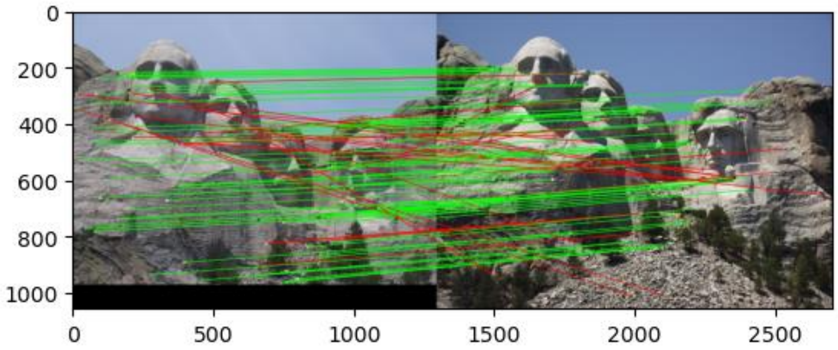
I used NNDR which is computing the Euclidean distances between each feature of the first image to all features in the second image and finding nearest neighbors. Then I calculated the ratio between the closest and the 2nd closest distance. The match was only accepted when it's below the threshold that I set up to ensure more reliable matches.

Results: Ground Truth Comparison

<Insert visualization of ground truth comparison with Notre Dame from ps2.ipynb here> [1 pt]

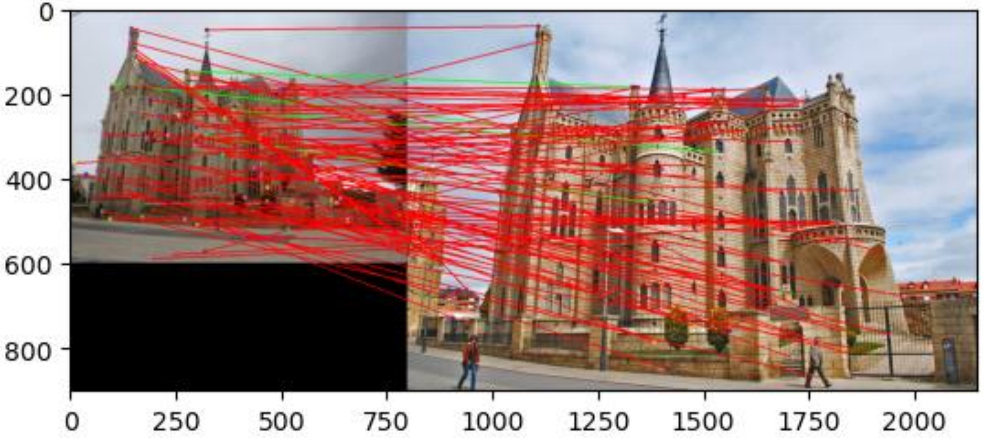


<Insert visualization of ground truth comparison with Rushmore from ps2.ipynb here> [1 pt]



Results: Ground Truth Comparison

<Insert visualization of ground truth comparison with Gaudi from ps2.ipynb here> [1 pt]



<Insert numerical performances on each image pair here. Also discuss what happens when you change the 4x4 subgrid to 2x2, 5x5, 7x7, 15x15 etc?> [2.5 pts]

Notre Dame: 100/100 matches, 0.88 accuracy
Rushmore: 100/100 matches, 0.76 accuracy
Gaudi: 100/100 matches, 0.04 accuracy
Threshold: 0.9

Higher sub-grid sizes can capture finer details but more sensitive to small misalignments. Smaller sub-grid sizes are more robust but is less distinctive

1.4(a): Hyperparameter Tuning part 1 [Extra credit]

<Insert images of the ground truth correspondence and their corresponding accuracies for varying sigma in the second moments [3, 6, 10, 30] > [0.5 pts]

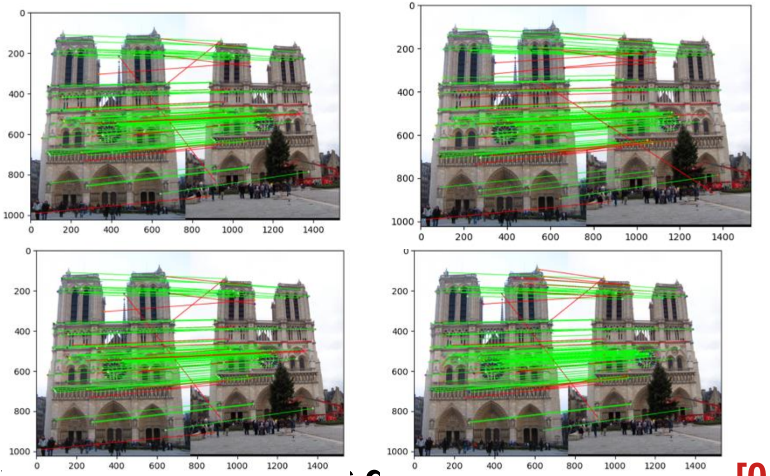
Image order (left to right, top to bottom)

Sigma = 3, 100/100 matches, 0.82 accuracy

Sigma = 6, 100/100 matches, 0.76 accuracy

Sigma = 10, 100/100 matches, 0.81 accuracy

Sigma = 20, 100/100 matches, 0.80 accuracy



When changing the values for large sigma (>20), why are the accuracies generally the same? [0.5 pts]

When sigma becomes too large, the blur oversmooths local gradients that additional increases in sigma barely affect the second moment matrices.

1.4(a): Hyperparameter Tuning part 2 [Extra credit]

<Insert images of the ground truth correspondence and their corresponding accuracies for varying feature width in the SIFT [8, 16, 24, 32] > [0.5 pts]

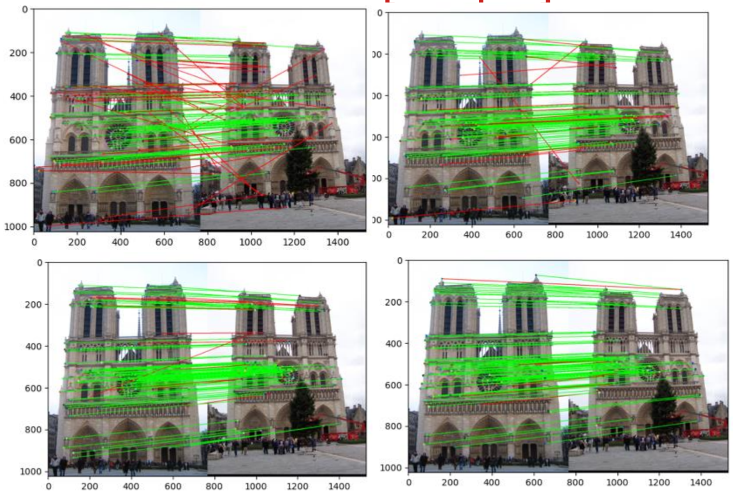


Image order (left to right, top to bottom). Assume sigma=10

Feature width = 8, 100/100 matches, 0.56 accuracy

Feature width = 16, 100/100 matches, 0.81 accuracy

Feature width = 24, 100/100 matches, 0.88 accuracy

Feature width = 32, 100/100 matches, 0.92 accuracy

What is the significance of changing the feature width in SIFT? [0.5 pts]

Increasing the width gives more contextual information making descriptors more robust and distinctive which is proved by increased accuracy with higher feature width

1.4(c): Accelerated Matching [Extra credit]

<Insert Runtime/Accuracy of your faster matching implementation. What did you try and why is it faster?> [1 pts]

Using KDTree quickly searches for nearest neighbors instead of brute-force calculation of pair-wise distance.

100/100 matches

0.81 accuracy