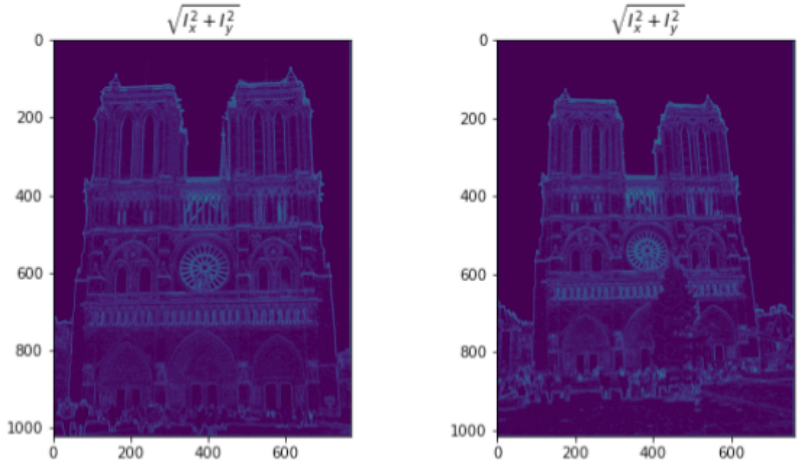
CS 6476 Project 2

Haoran Wang
Haoran.wang@gatech.edu
Hwang827
903543952

Part 1: Harris corner detector

[insert visualization of \sqrt(I_x^2 + I_y^2) for Notre Dame image pair from proj2.ipynb here]

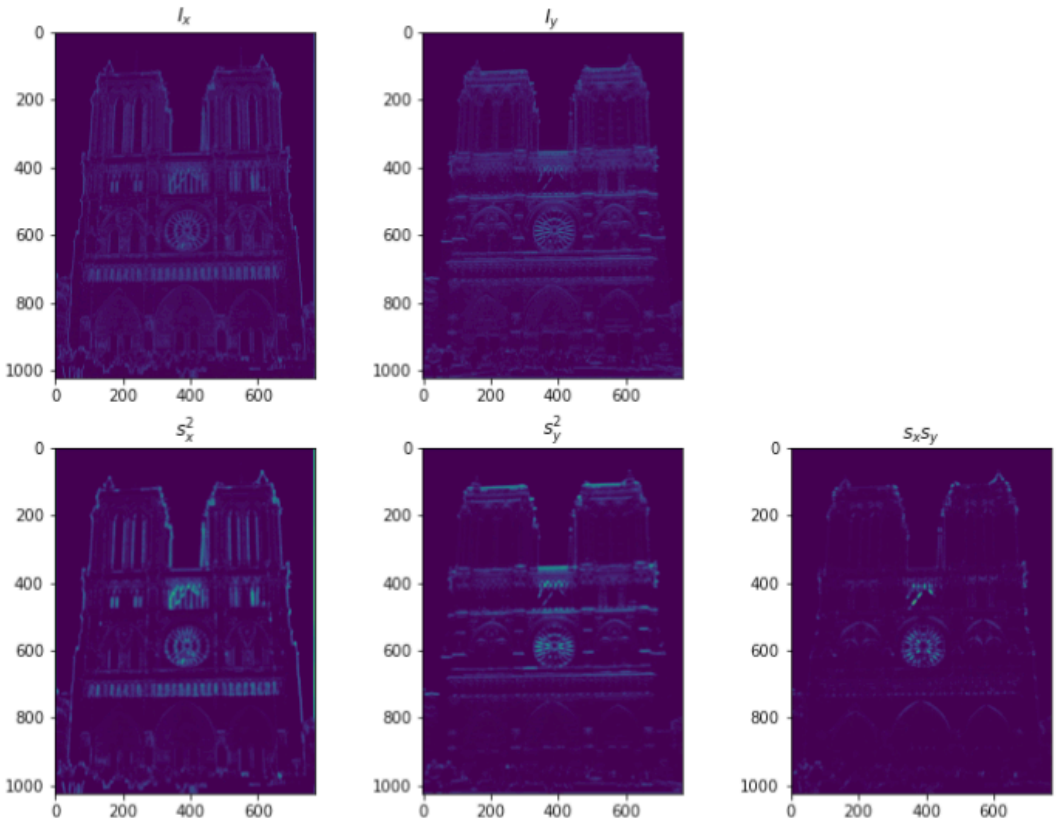


[Which areas have highest magnitude? Why?]

The edges has highest magnitude. Because the edges has higher gradients, either in x direction or y direction or both.

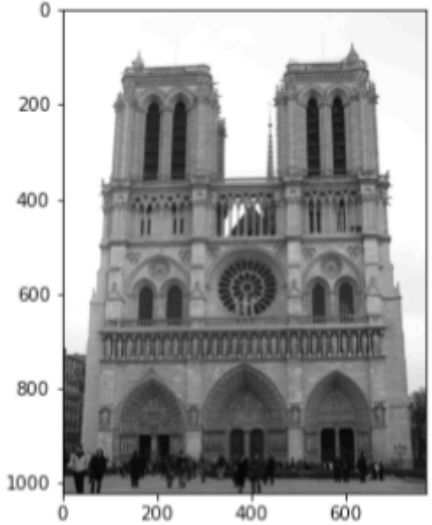
Part 1: Harris corner detector

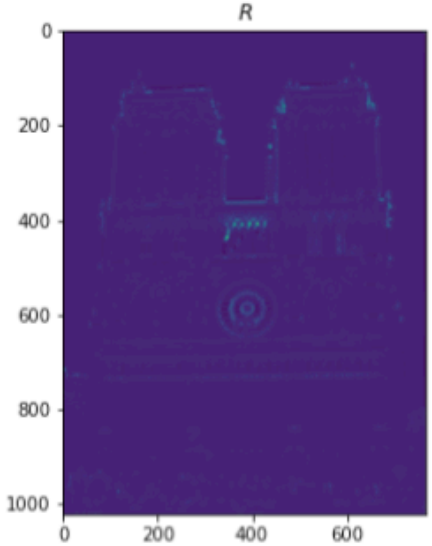
[insert visualization of I_x, I_y, s_x^2, s_y^2, s_xs_y for Notre Dame image pair from proj2.ipynb here]



Part 1: Harris corner detector

[insert visualization of corner response map of Notre Dame image from proj2.ipynb here]



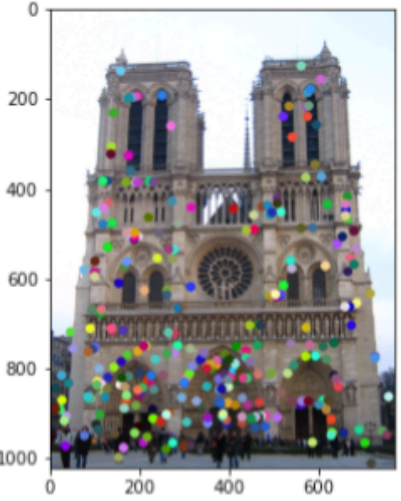


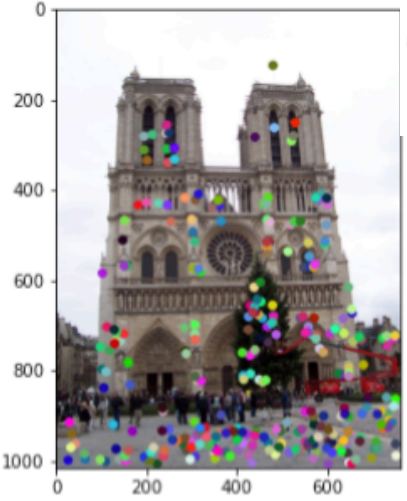
[Are gradient features invariant to both additive shifts (brightness) and multiplicative gain (contrast)? Why or why not? See Szeliski Figure 3.2]

Gradient features are invariant to additive shifts (brightness), but it is not invariant to multiplicative gain (contrast). As we can see from Figure 3.2, the shape of the plots of the channels doesn't change for additive shifts; but it becomes sharper for the multiplicative gain.

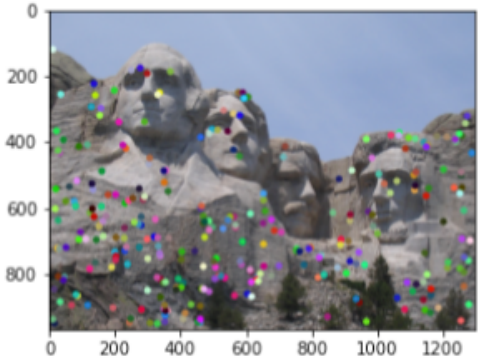
Part 1: Harris corner detector

[insert visualization of Notre Dame interest points from proj2.ipynb here]



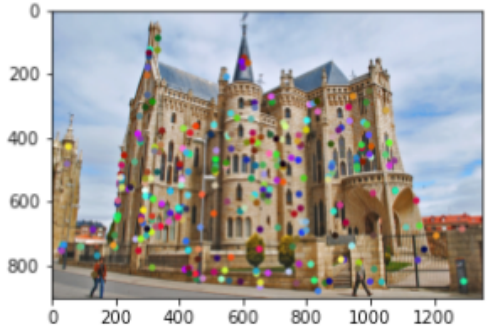


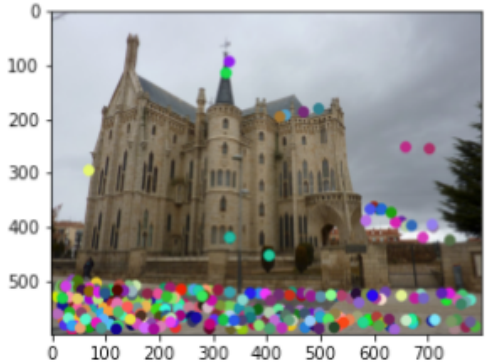
[insert visualization of Mt. Rushmore interest points from proj2.ipynb here]



Part 1: Harris corner detector

[insert visualization of Gaudi interest points from proj2.ipynb here]





[What are the advantages and disadvantages of using maxpooling for non-maximum suppression (NMS)?]

Maxpooling performs spatial compression for our detector to get a reasonable number of interest points, which saves computation resources. Maxpooling provides the local maximum of a window, however, it would still pick out the interest point even if the local maximum is lower than other and it may not give us the interest point even if the point has higher value than other interest points but not its local maximum.

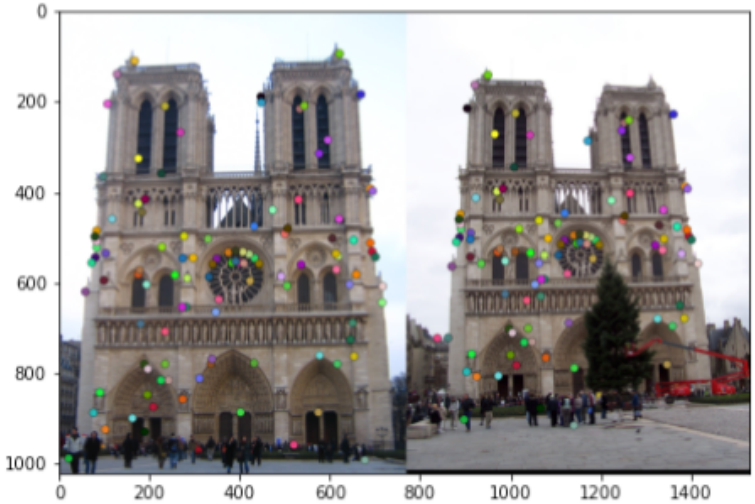
Part 1: Harris corner detector

[What is your intuition behind what makes the Harris corner detector effective?]

The intuition behind the Harris corner detector is that it uses a small window to slide across the image, where gradients in different directions will change along the windows. In windows that contain corners, shifting of the window in any directions will cause large changes in the gradients, while in windows that do not contain corners, the change will be small.

Part 2: Normalized patch feature descriptor

[insert visualization of normalized patch descriptor from proj2.ipynb here]

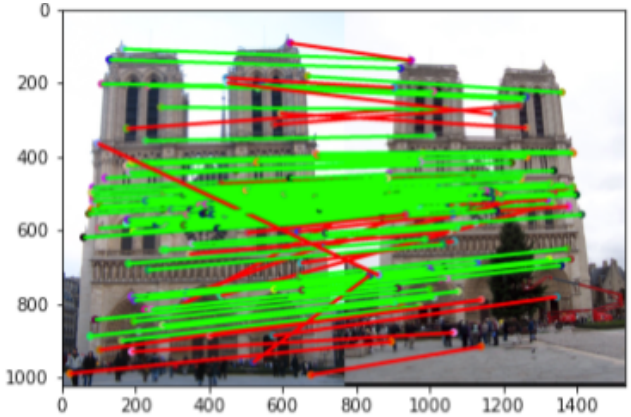


[Why aren't normalized patches a very good descriptor?]

Because according to Szeliski 7.1.2, the local appearance of features will change in orientation and scale, and sometimes even undergo affine deformations, which means normalized patches are not invariant to brightness change, contrast change, or small spatial shifts.

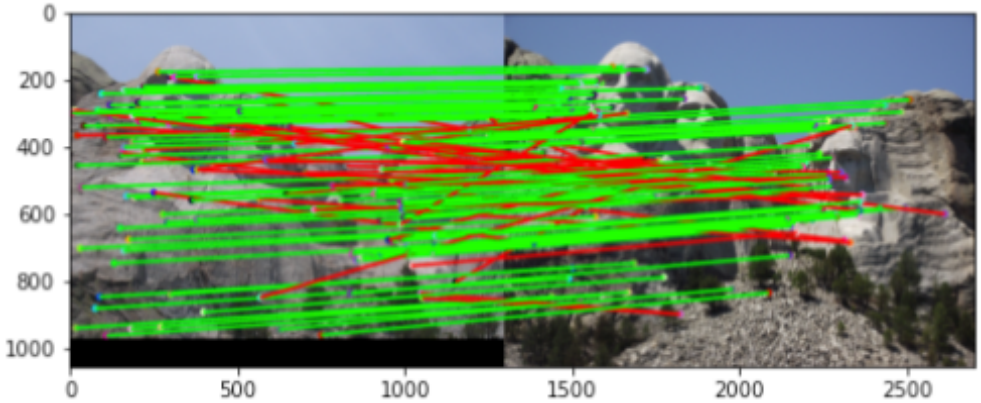
Part 3: Feature matching

[insert visualization of matches (with green/red lines for correct/incorrect correspondences) for Notre Dame image pair from proj2.ipynb here]



matches (out of 100): [109/100]
Accuracy: [0.761468]

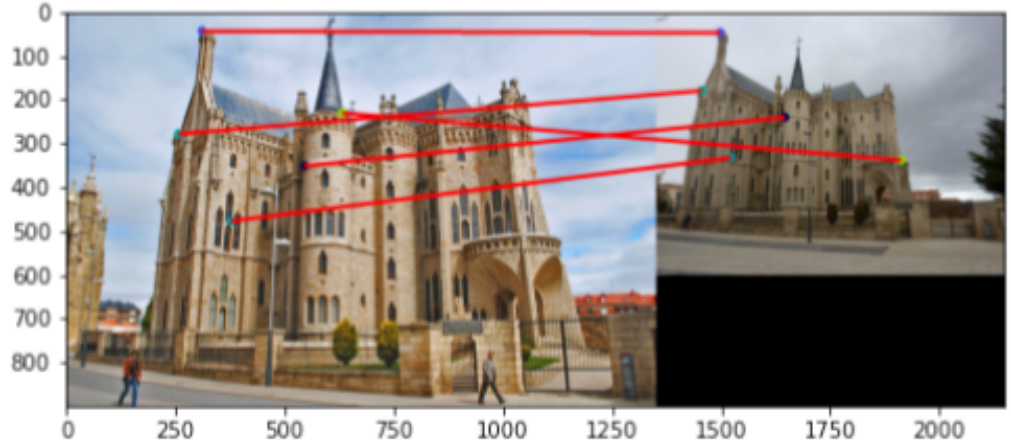
[insert visualization of matches for Mt. Rushmore image pair from proj2.ipynb here]



matches: [116/100]
Accuracy: [0.732759]

Part 3: Feature matching

[insert visualization of matches for Gaudi image pair from proj2.ipynb here]



matches: [5/100]

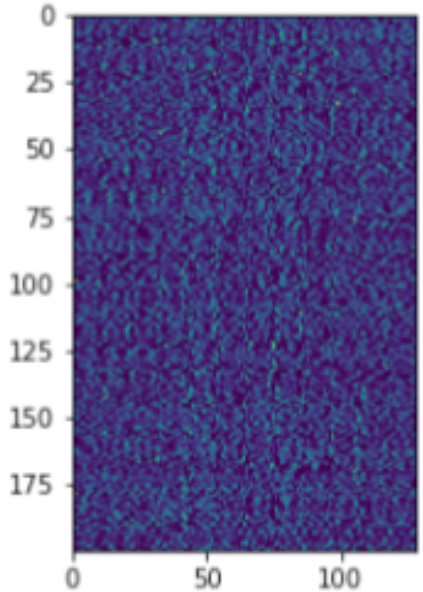
Accuracy: [0.000000]

[Describe your implementation of feature matching here]

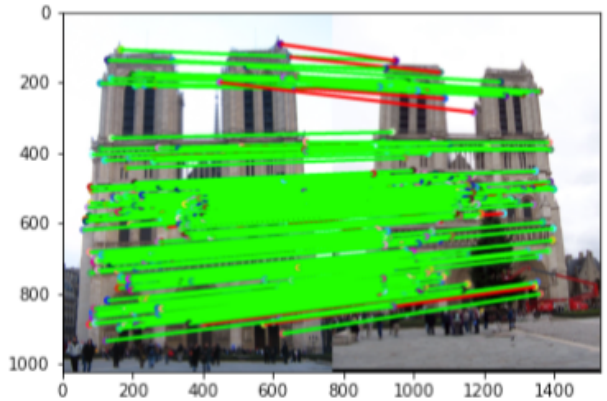
To implement feature matching, I used Euclidean distance to compute the pairwise distances between the two sets of features, then the Nearest-neighbor distance ratio test is performed, by first calculating the ratio of the top two nearest neighbors for each points in feature 1 and then compare it to a threshold (0.8, according to the paper), if the ratio is lower than the threshold, I deem the nearest neighbor point in feature 2 a match for the point in feature 1.

Part 4: SIFT feature descriptor

[insert visualization of SIFT feature descriptor from proj2.ipynb here]



[insert visualization of matches (with green/red lines for correct/incorrect correspondences) for Notre Dame image pair from proj2.ipynb here]

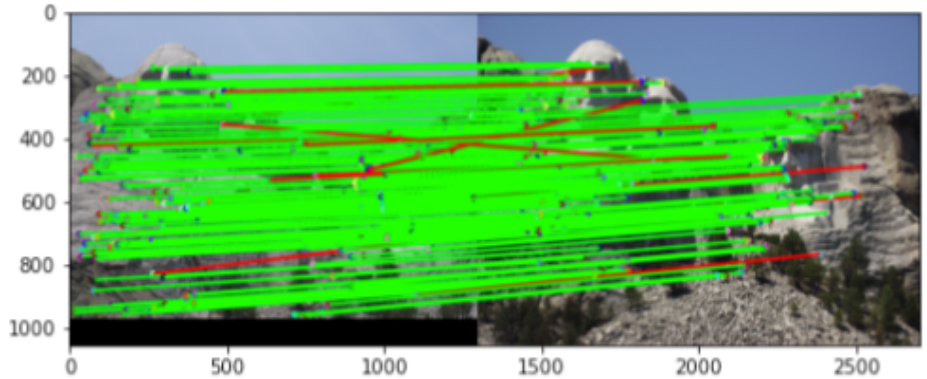


matches (out of 100): [210/100]

Accuracy: [0.923810]

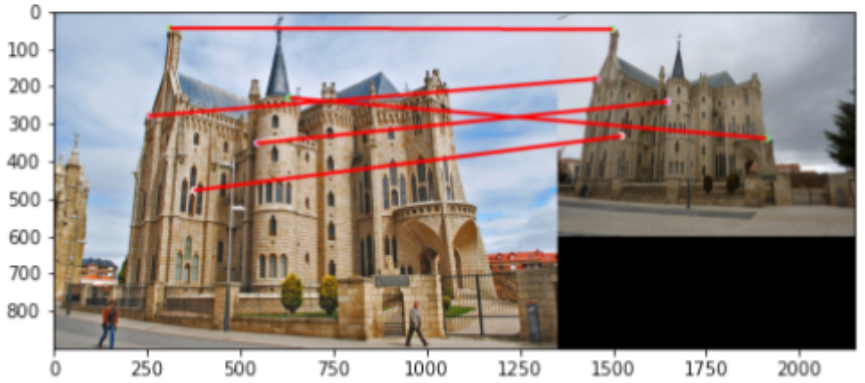
Part 4: SIFT feature descriptor

[insert visualization of matches for Mt. Rushmore image pair from proj2.ipynb here]



matches: [197/100]
Accuracy: [0.928934]

[insert visualization of matches for Gaudi image pair from proj2.ipynb here]



matches: [5/100]
Accuracy: [0.000000]

Part 4: SIFT feature descriptor

[Describe your implementation of SIFT feature descriptors here]
To implement SIFT feature descriptors, I first compute the gradient of the image. Then, I compute the magnitude and orientation of the gradients at each pixel. Then, for each interest points, I compute SIFT feature vector. To do this, I used a 16x16 window centered at the interest point to compute a 128x1 vector of gradient histogram by dividing the window to 16 4x4 grids and each compute a vector of local distribution of gradients in 8 directions. I normalize the 128x1 vector and take the square root. This forms my SIFT feature descriptor.

[Why are SIFT features better descriptors than the normalized patches?]
SIFT stands for Scale invariant feature transform. According to Lowe, the author of SIFT, it transforms an image into a large collection of local feature vectors, each of which is invariant to image translation, scaling, and rotation, and partially invariant to illumination changes and affine or 3D projection.

Conclusion

[Why aren't our version of SIFT features rotation- or scale-invariant? What would you have to do to make them so?]

Because we simply use the normalized orientation distribution histogram of the key points. According to Lowe, to achieve rotation invariance, we have to select key locations at maxima and minima of a difference of Gaussian function applied in scale space, then we can find the main orientation of the descriptor and assign that angle to the key points. Then to achieve scale invariant, we can use the generated scale space to calculate the Difference of Gaussians and the use them to calculate Laplacian of Gaussian approximations which are scale invariant.