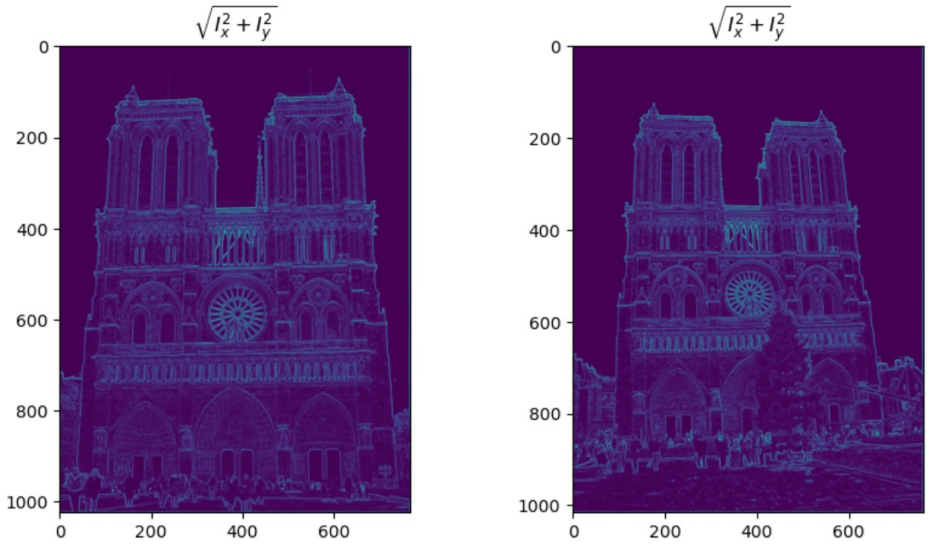
CS 4476/6476 Project 2

[Ruoran Jia]
[rjia41@gatech.edu]
[rjia41]
[903631225]

Part 1: Harris corner detector

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

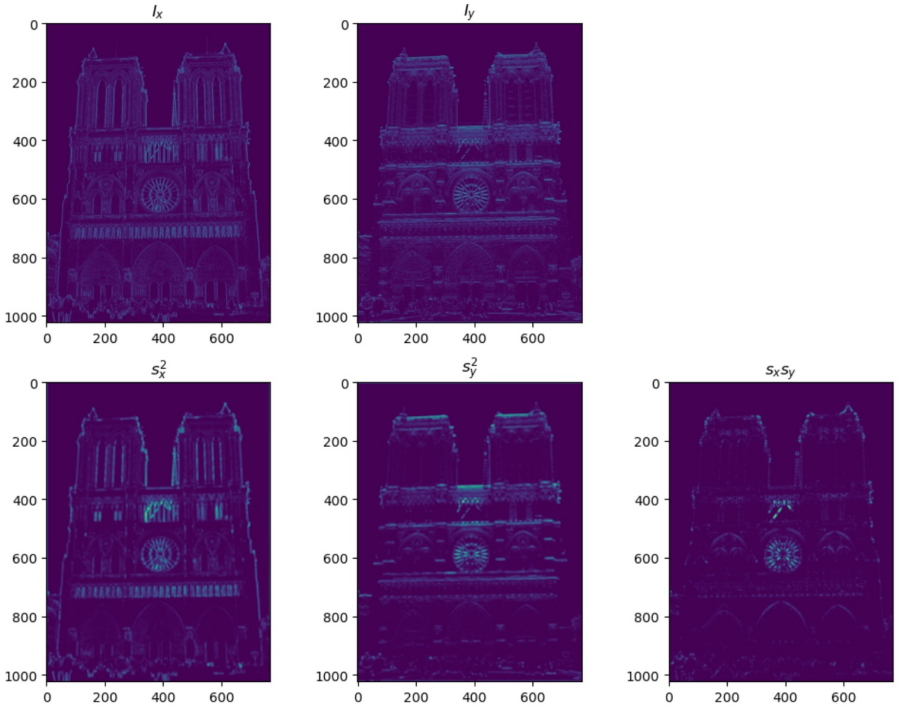


[Which areas have highest magnitude? Why?

The edges in the picture have the highest magnitude because they have higher gradient values in x or y direction, or in both directions.

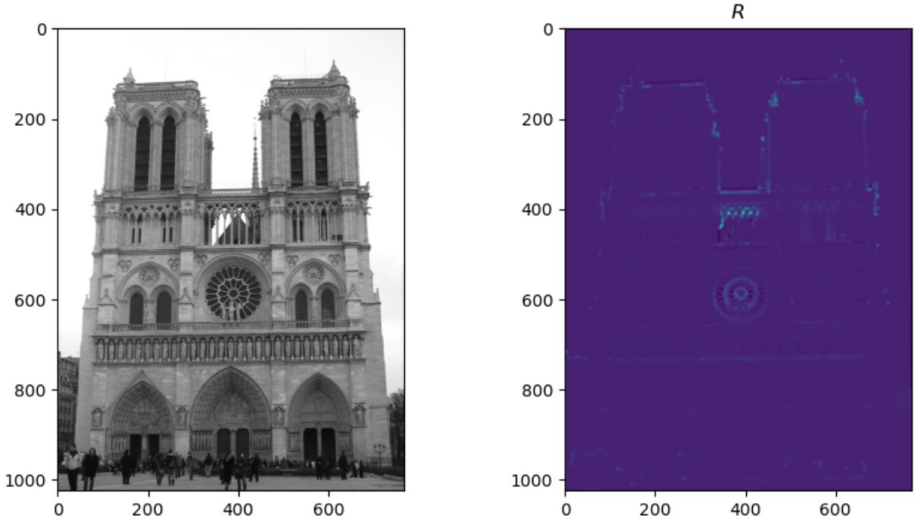
Part 1: Harris corner detector

[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]

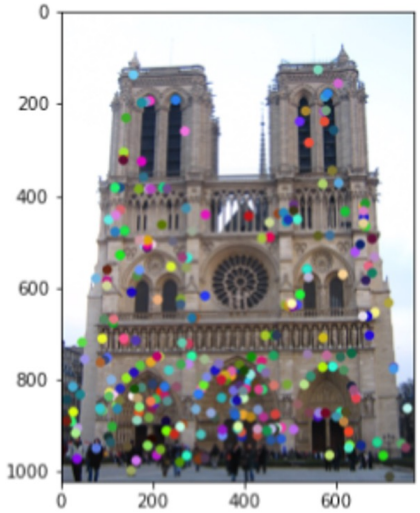


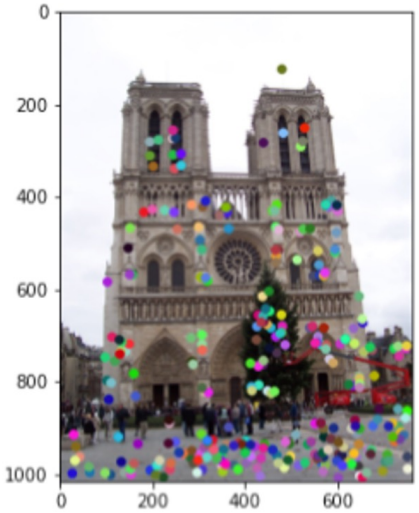
[Are gradient features invariant to both additive shifts (brightness) and multiplicative gain (contrast)? Why or why not? See Szeliski Figure 3.2]

Gradient features are invariant to additive shifts(brightness), but is not invariant to multiplicative gain (contrast).This is because when we look at the Szeliski Figure 3.2, we can easily notice that the shape of the plots of the channels doesn’t change for additive shifts, but changes (become sharper) for the multiplicative gain.

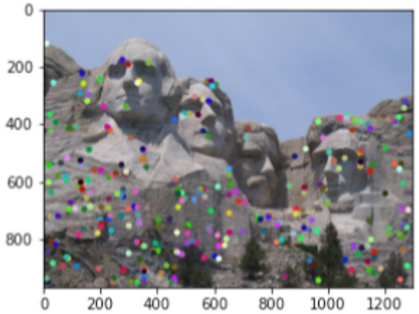
Part 1: Harris corner detector

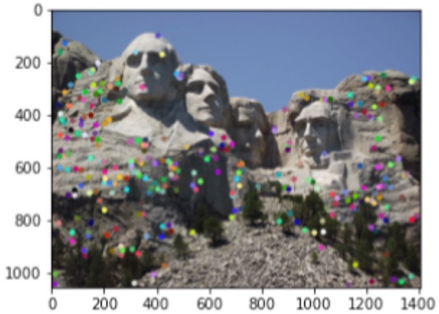
[insert visualization of Notre Dame interest points from proj2.ipynb here]





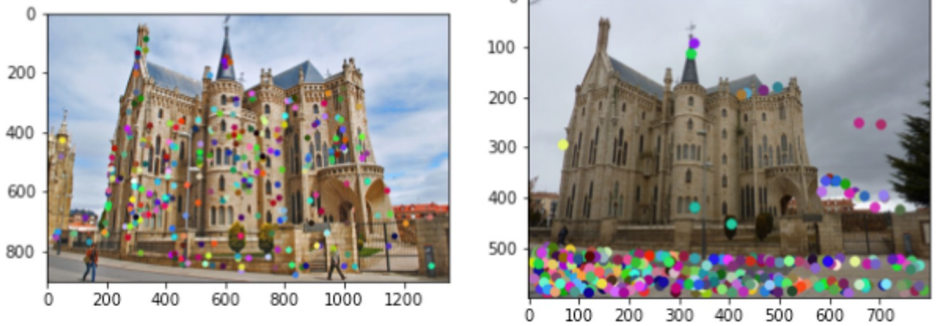
[insert visualization of Mt. Rushmore interest points from proj2.ipynb here]





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)?]

Advantages: Maxpooling can help us get a reasonable number of interest points using spatial compression, saving the computation resources for us. Maxpooling also provides local maximum of a window.
Disadvantages: Maxpooling will get the interest point even though the local maximum is lower than other interest points. On the other hand, sometimes when an interest point has higher value than other interest points, maxpooling might not give us the interest point but not its local maximum.

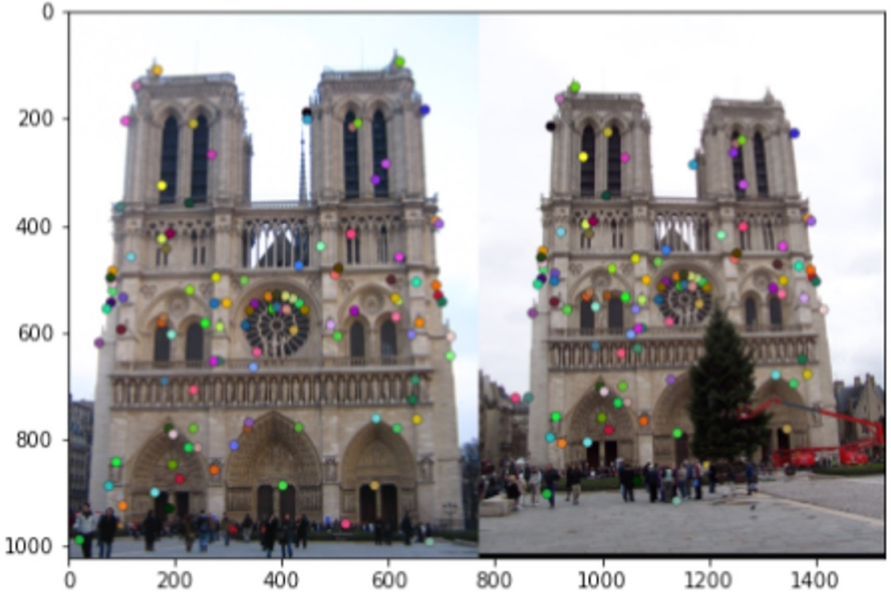
Part 1: Harris corner detector

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

My intuition behind Harris corner detector is that sliding a small window over an image causes gradient change in different directions. This can be used to detect corners as shifting the window in windows that contain corners will result in a large change in gradients, while in windows that don’t contain corners, the change will be small.

Part 2: Normalized patch feature descriptor

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

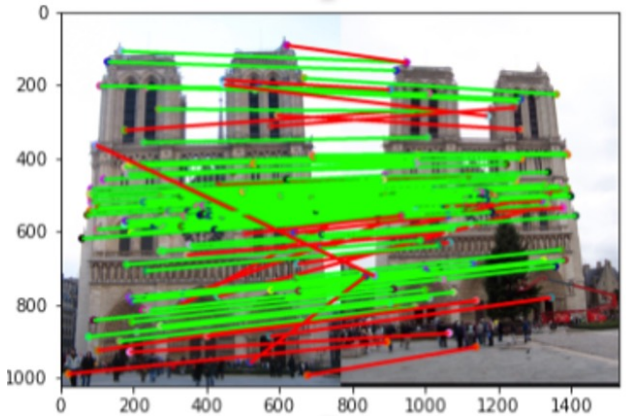


[Why aren't normalized patches a very good descriptor?]

Because as you can see, the local appearance of features will change in orientation and scale, and sometimes this will lead to deformations. Hence, normalized patches are not invariant to spatial shifts, additive shifts, and multiplicative gain.

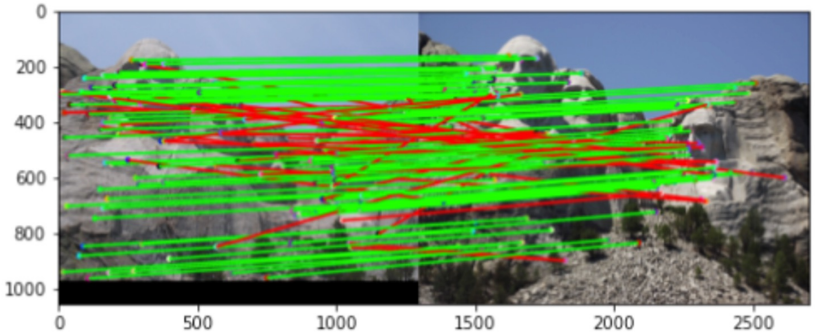
Part 3: Feature matching

[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): 108/100
Accuracy: 0.76

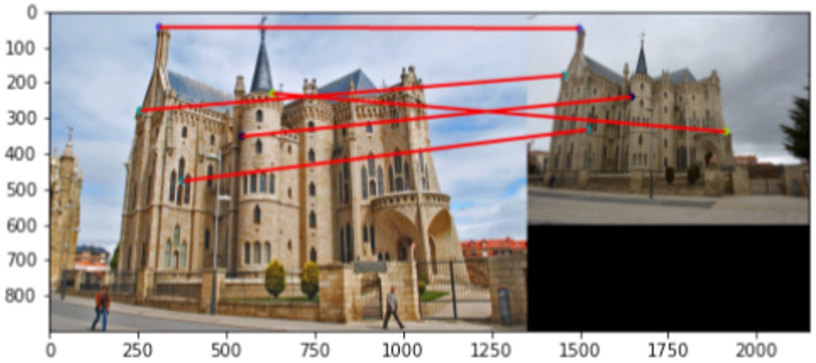
[insert visualization of matches for Mt. Rushmore image pair from proj2.ipynb here]



matches: 116/100
Accuracy: 0.733

Part 3: Feature matching

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



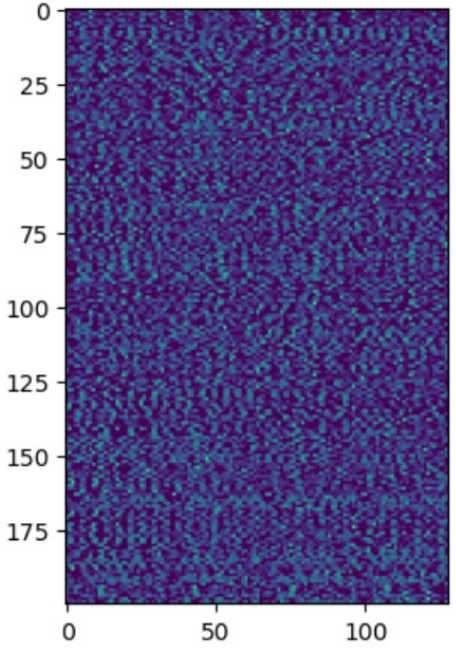
matches: 5/100
Accuracy: 0

[Describe your implementation of feature matching here]

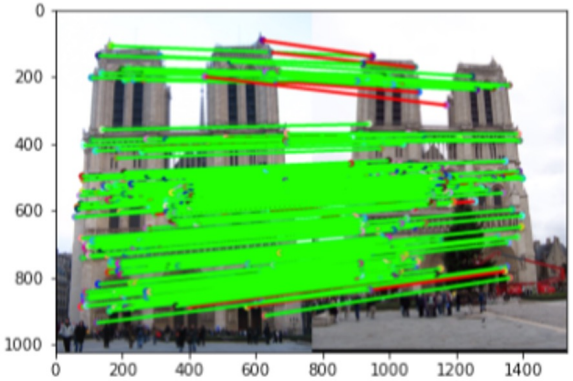
First, I use Euclidean distance to compute pairwaise distances between the two image, which are represented by two sets of features. Then, I use the nearest-neighbor distance ratio test through calculating the ratio of the top two nearest neighbors for each point in feature 1 set, and then I will compare it to a threshold (0.81 according to the paper referred in code). If the ratio is lower than the threshold, I will consider the nearest neighbor point in feature 2 as a match for the point in feature 1.

Part 4: SIFT feature descriptor

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



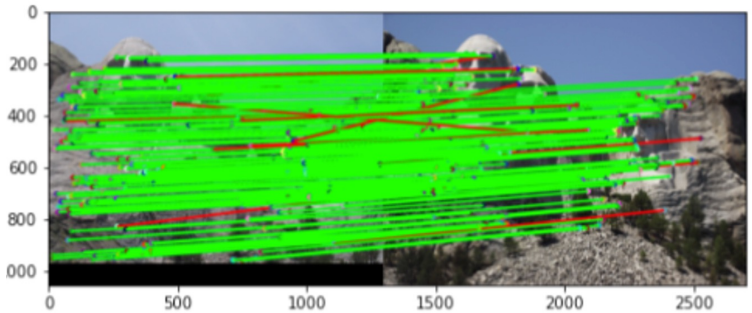
[insert visualization of matches (with green/red lines for correct/incorrect correspondences) for Notre Dame image pair from proj2.ipynb here]



matches (out of 100): [138/100]
Accuracy: [0.839]

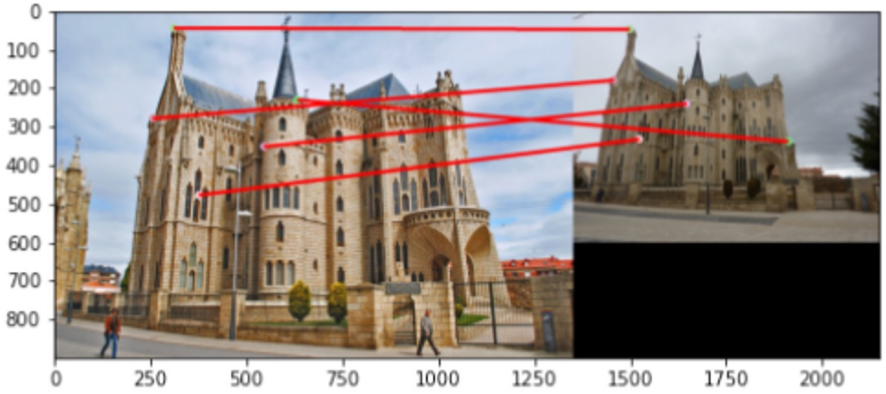
Part 4: SIFT feature descriptor

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



matches: [121/100]
Accuracy: [0.819]

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



matches: 5/100
Accuracy: 0

Part 4: SIFT feature descriptor

[Describe your implementation of SIFT feature descriptors here]

I first compute the gradient of the image and then calculate the magnitude and orientation of the gradients. Then, I compute SIFT feature vector for each interest point. I use 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 compute a vector of distribution of gradients in 8 directions for each. I then normalize the 128x1 vector and take the sqrt of it. This is finallly my SIFT feature descriptor.

[Why are SIFT features better descriptors than the normalized patches?]

SIFT means Scale invariant feature transform. SIFT featrues better descriptors because it transforms an image to a large collection of local feature vectors, each is invariant to image translation, scaling, and rotation, and partially invariant to illumination changes and affine or 3d projection.

Part 4: SIFT feature descriptor

[Why does our SIFT implementation perform worse on the given Gaudi image pair than the Notre Dame image and Mt. Rushmore pairs?]

This is because my SIFT implementation is not scale invariant. The Notre Dame pair images were taken from very similar viewpoints and with very similar illumination. Performance went down a little bit for the Mount Rushmore pair because their illumination is inconsistent. And, the performance is much more worse on the Episcopal Gaudi pair because the pair is both different from illumination and scale.

Part 5: SIFT Descriptor Exploration

Describe the effects of changing window size around features. Did different values have better performance?

Part 5: SIFT Descriptor Exploration

Describe the effects of changing the number of local cells in a window around a feature? Did different values have better performance?

Part 5: SIFT Descriptor Exploration

Describe the effects of changing number of orientations (bins) per histogram. Did different values have better performance?

Part 5: SIFT Descriptor Exploration

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

Part 5: SIFT Descriptor Exploration

[Discuss why you think your SIFT pipeline worked well or poorly for the given building. Are there any characteristics that make it difficult to correctly match features]?

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 merely utilize the histogram of the key points' normalized orientation distribution. The paper states that in order to feature rotation invariant, we first pick important locations at the maximum and minimum of the difference of the Gaussian function applied in scale space. After that, we can compute the descriptor's orientation and assign each important point a specific angle. Then, to achieve scale invariant, we can compute the difference of Gaussians using the scale space that was constructed, and then utilize them to compute the scale-invariant Laplacian of Gaussian approximations.