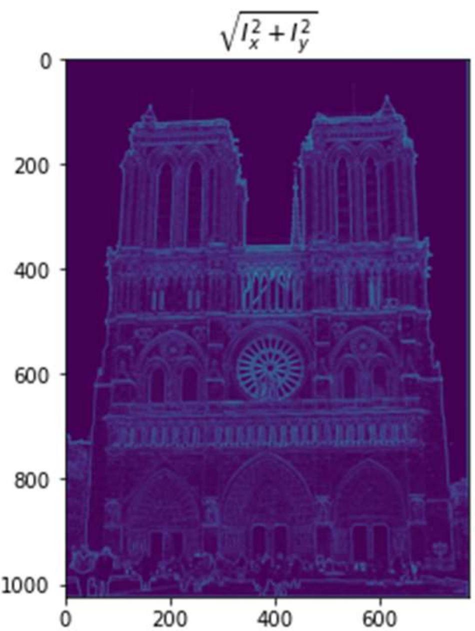
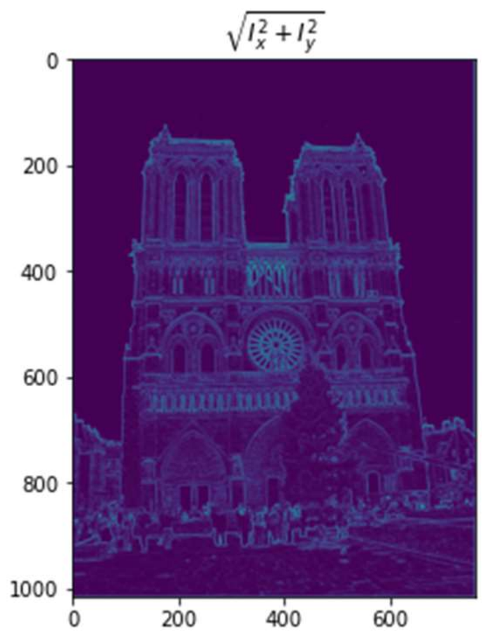
CS 4476/6476 Project 2

Cameron Potter
cgpotter@gatech.edu
cpotter8
903465425

Part 1: Harris corner detector

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

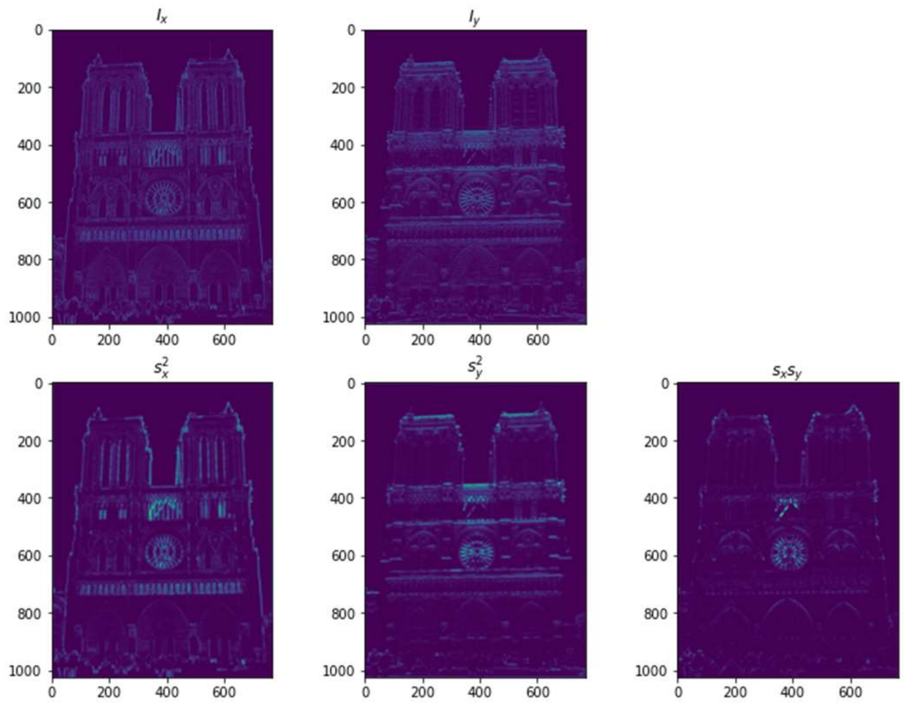




The areas near the middle circular area and directly above that have the highest values because the gradients have high magnitude. The gradient information helps detect corners, and clearly corners would have high gradients since they imply the image changes a lot in that area.

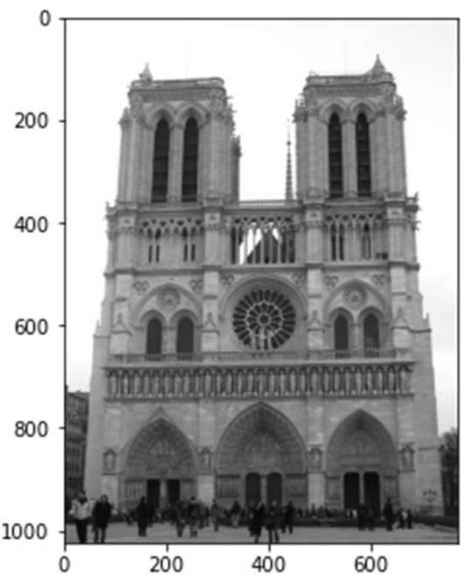
Part 1: Harris corner detector

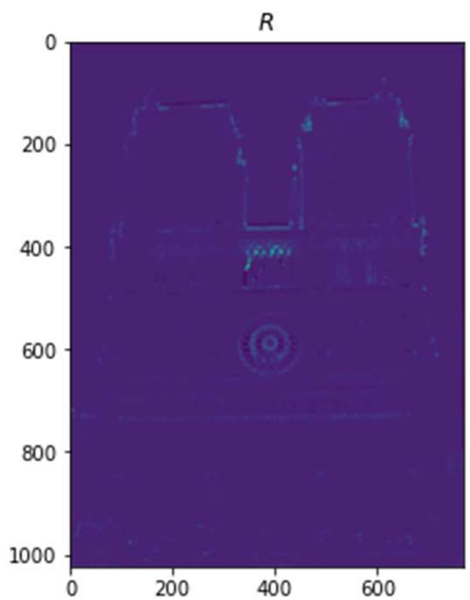
[insert visualization of I_x, I_y, s_x^2, s_y^2, $s_x s_y$ for Notre Dame image pair from proj2.ipynb here]



Part 1: Harris corner detector

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





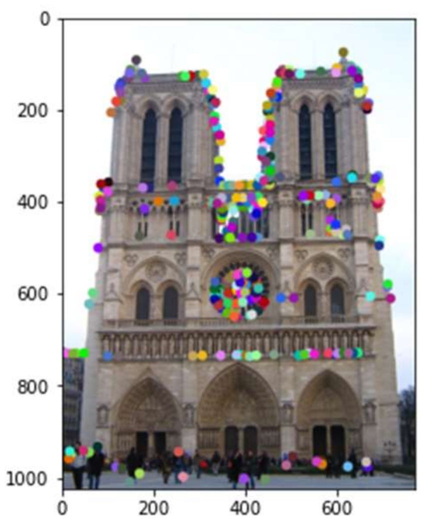
[Are gradient features invariant to both additive shifts (brightness) and multiplicative gain (contrast)? Why or why not? See Szeliski Figure 3.2]

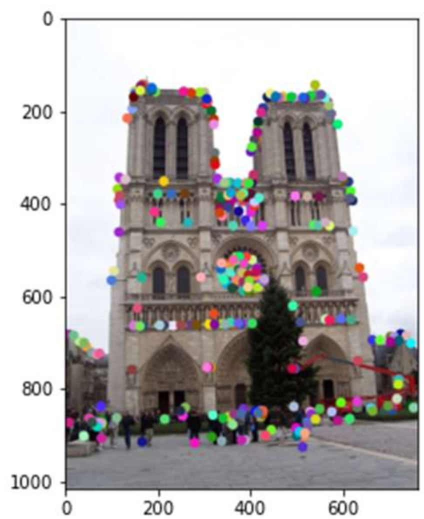
They are invariant to additive shifts (gradient of F(x) + c equals gradient of F(x)), but only partially invariant to multiplicative gain (gradient of aF(x) is proportional to the gradient of F(x), but only equal if a=1 or the multiplicative identity in the given vector space).

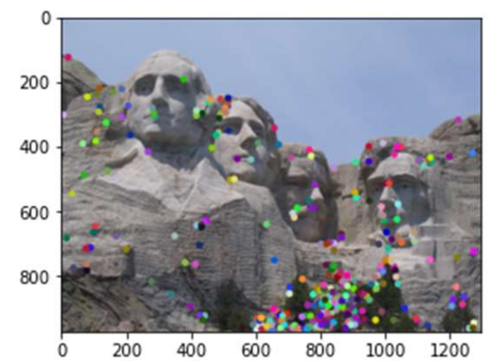
Part 1: Harris corner detector

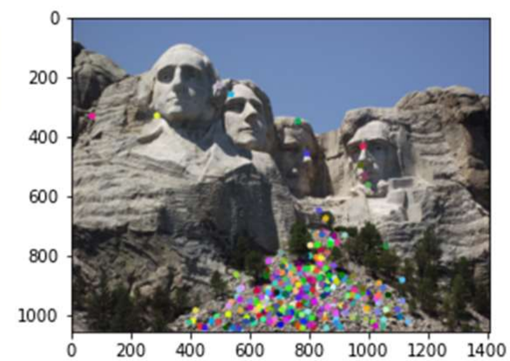
[insert visualization of Notre Dame interest points from proj2.ipynb here]

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



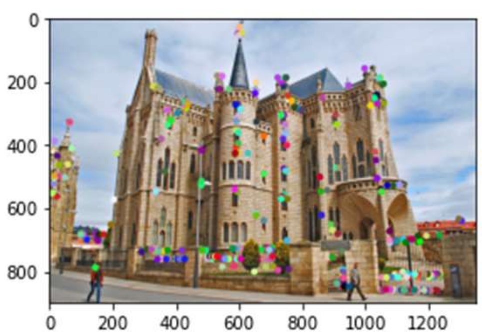


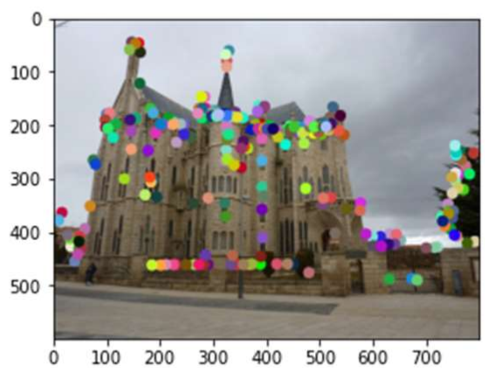




Part 1: Harris corner detector

[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 is an efficient way to do NMS with comparable accuracy to alternatives. A disadvantage is potentially losing information, especially if the kernel is too big. Also, results are less accurate than other methods, especially dependent on when NMS is done.

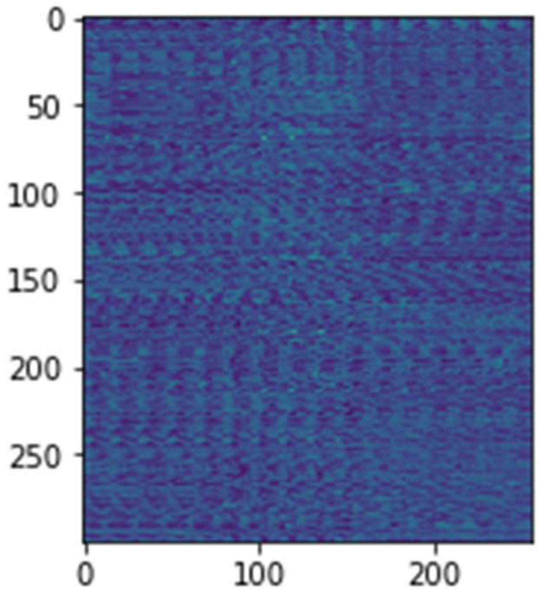
Part 1: Harris corner detector

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

Since corners are both rotation and translation invariant, they are useful tools for detecting features. Computing the vertical and horizontal derivatives allows for changes in both directions to matter. The NMS ensures the features are comers versus just edges.

Part 2: Normalized patch feature descriptor

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

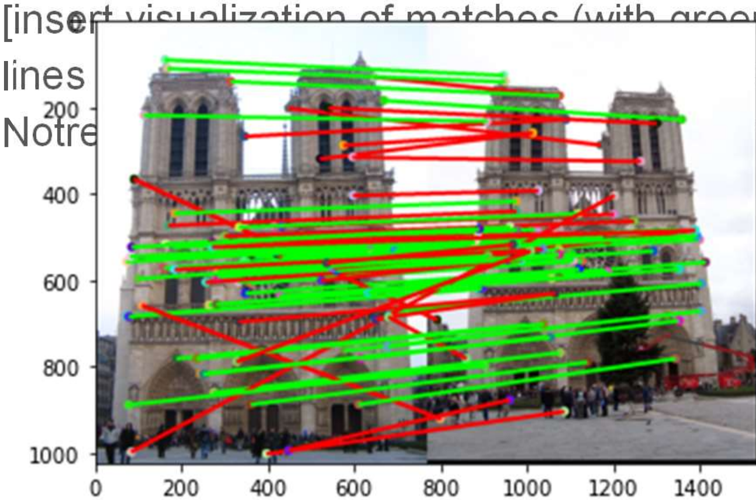


[Why aren't normalized patches a very good descriptor?]

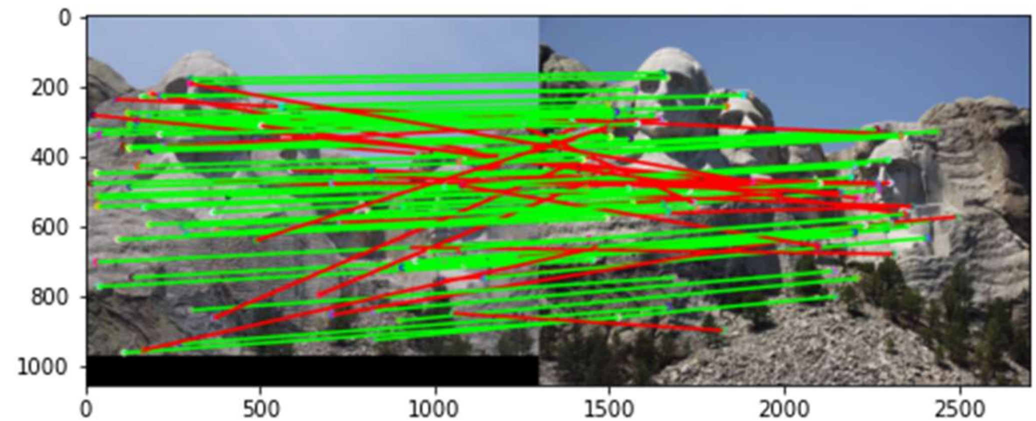
They are very sensitive to small shifts and rotations. Hence, when trying to use them as descriptors, they will not produce reliable outputs when these mappings are present.

Part 3: Feature matching

[insert visualization of matches (with green/red
lines … s) for
Notre … here]



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

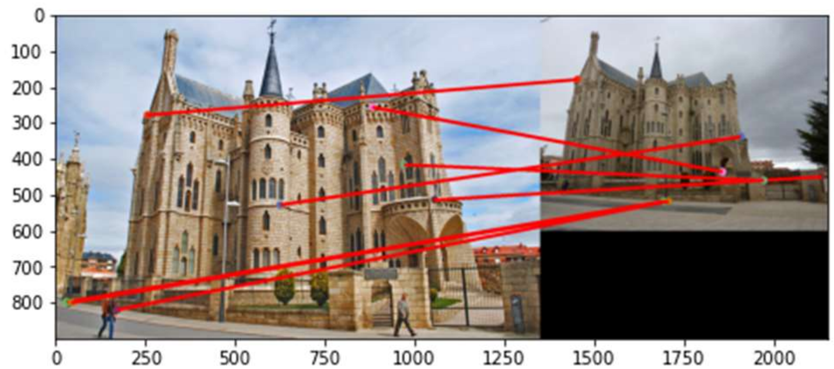


matches (out of 100): 88
Accuracy: 0.59

matches: 100
Accuracy: 0.71

Part 3: Feature matching

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

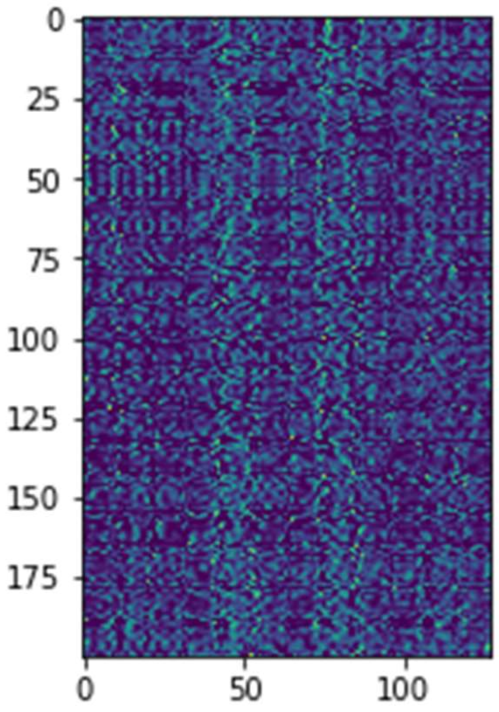


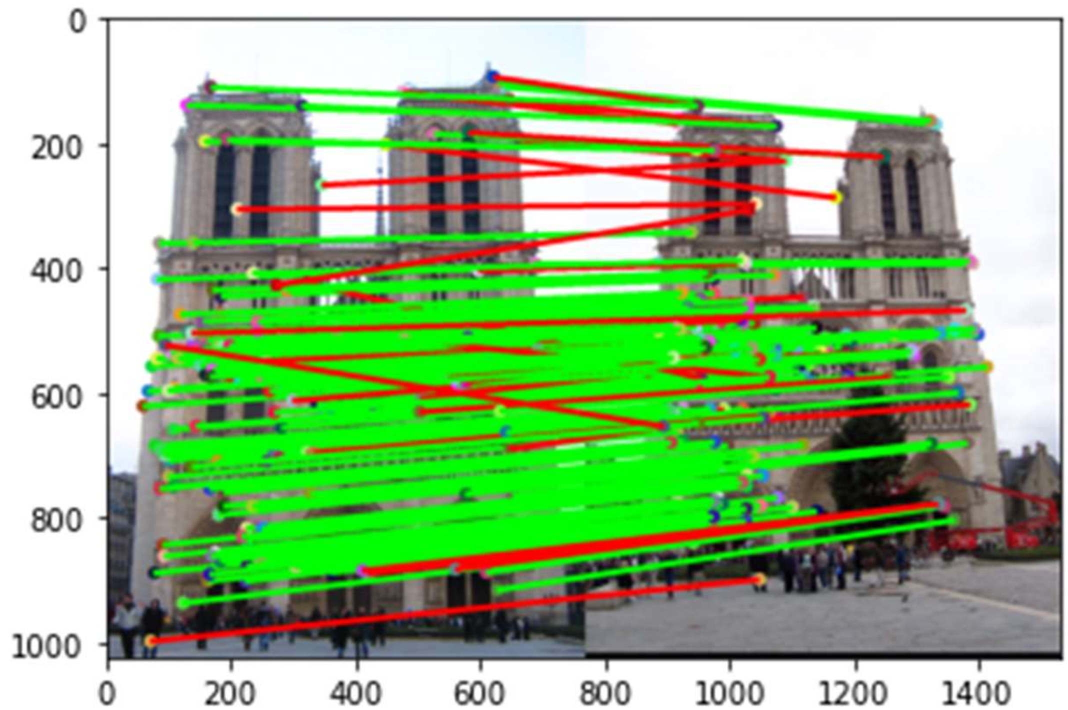
matches: 8
Accuracy: 0

We use the nearest-neighbor distance ratio feature matching. We establish a cutoff ratio, here is 0.8. Then, we compute the Euclidean distances between each pair of features. Then, we take the smallest distance and divide it by the second smallest distance for each feature. If this ratio is less than our cutoff, we define it as a match.

Part 4: SIFT feature descriptor

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



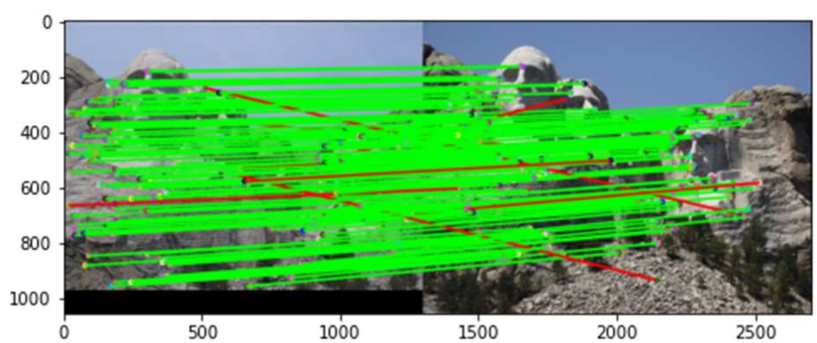


matches (out of 100): 176

Accuracy: 0.823864

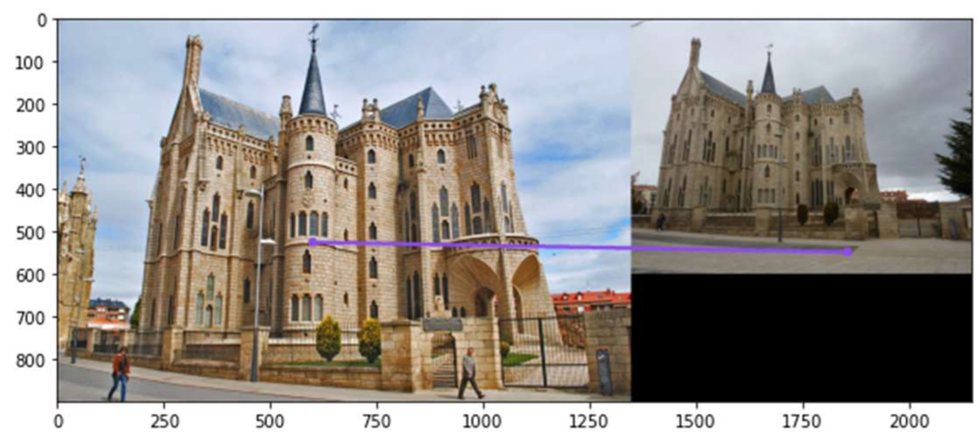
Part 4: SIFT feature descriptor

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



matches: 166
Accuracy: 0.945783

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



matches: 1
Accuracy: 0

Part 4: SIFT feature descriptor

This function returns the 128-d SIFT features computed at each of the input points. First, we compute the image gradients, and then their magnitudes and orientations. Then, we get the feature vector for each set of interest points. This is done by getting the gradient histogram from a window around the interest point.

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

The key issue with normalized patches was that they were very sensitive to small shifts and rotations. SIFT features are invariant to shifts, giving more accuracy.

Part 4: SIFT feature descriptor

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

The mapping between these pairs of images has much greater shift, rotational, and multiplicative translations than the Notre Dame pair. Hence, the latter two translations will create more invariance, leading to worse results. A key issue is the scale in the Gaudi images, as the SIFT feature detector is not scale invariant.

Part 5: SIFT Descriptor Exploration

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

Slightly decreasing window size led to slightly better performance on some images, but generally performed worse. Increasing the window size led to worse 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?

Decreasing the number of local cells led to worse performance. Increasing it also led to worse performance, but at a much greater decrease.

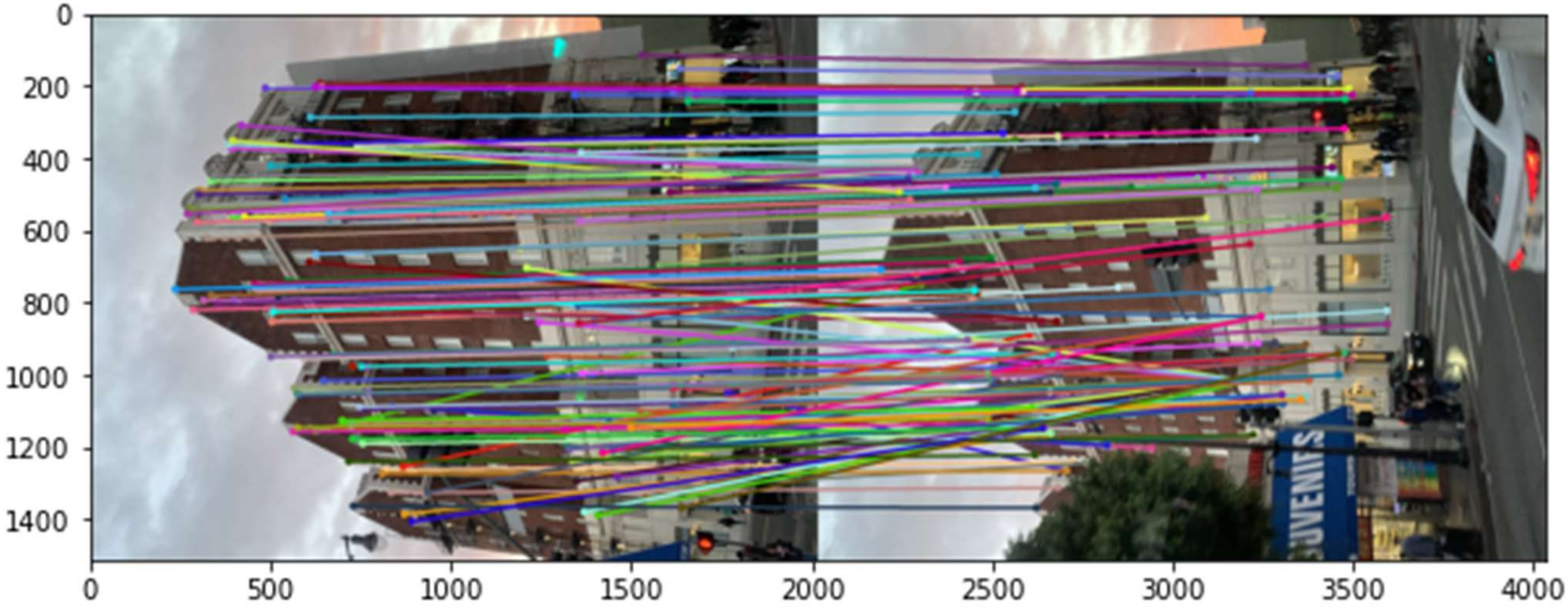
Part 5: SIFT Descriptor Exploration

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

Increasing the number of bins increased accuracy. Decreasing it worsened 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]?

The pipeline seemed to work very well. The two images are notably similar without strong rotational or multiplicative translations. Hence, SIFT is expected to work well. The only part that I expected to cause issues is the tree in the second picture, which is not present in the first. Also, the cars moving may have also caused some issues, but the pipeline surprisingly did not have any issues with them.

Conclusion

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

We did not use any gaussian blur, which helps with scale-invariance. Since we utilize the orientations of gradients, the pipeline is not rotation invariant. If we subtracted the interest points orientations from the rest of the orientation, the pipeline could produce better results with rotations.