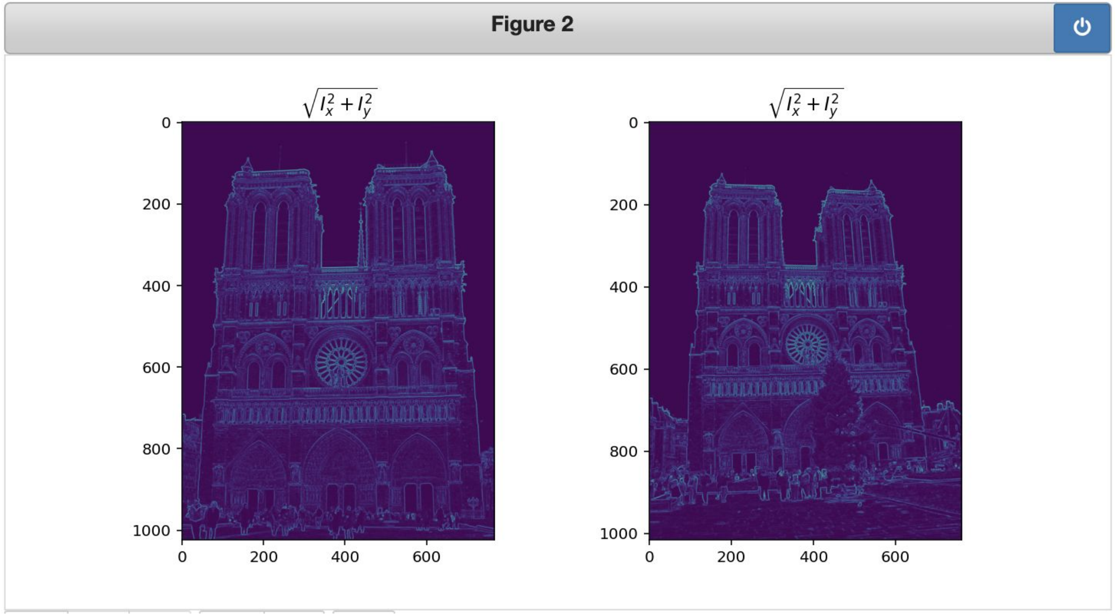
CS 5330 Programming Assignment 2

Luke Davidson
davidson.lu@northeastern.edu
Davidson.lu
002197966

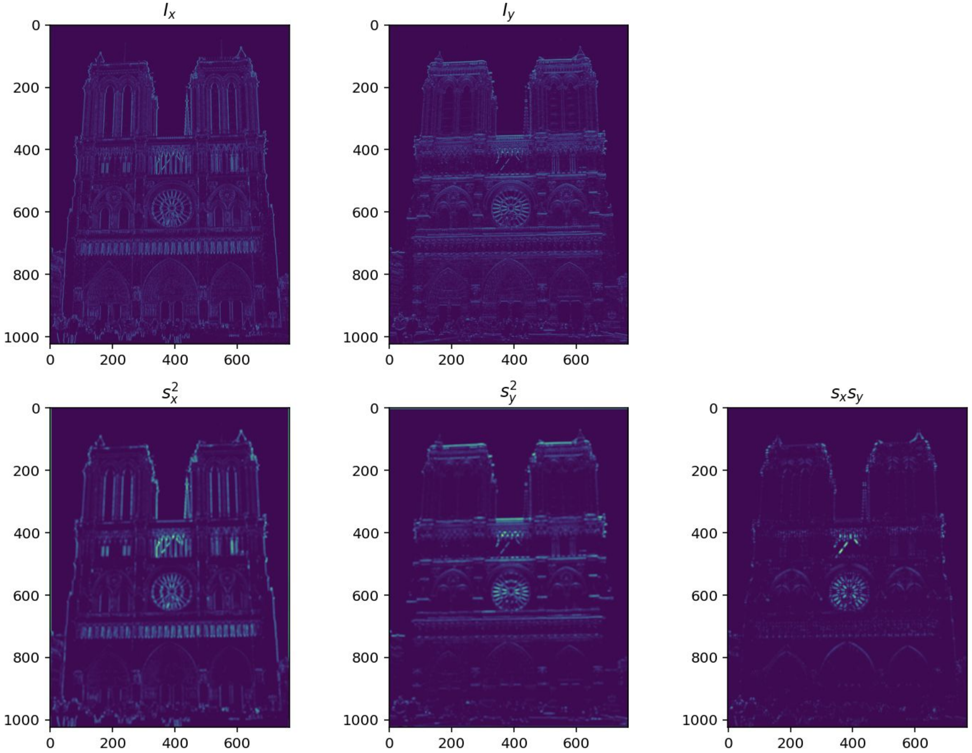
Part 1: Harris corner detector



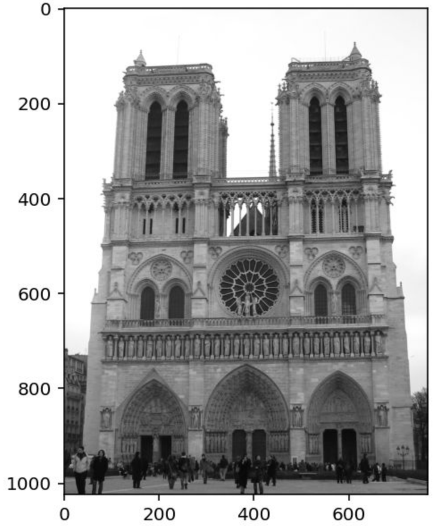
Which areas have the highest magnitude?

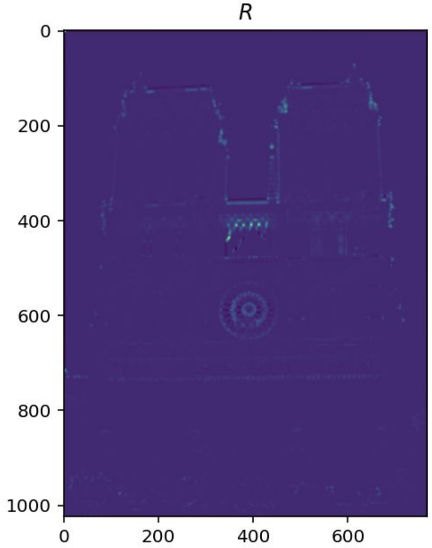
- The areas that have the highest magnitude are the edges in the image, as both Ix and Iy will be large values. A large Ix denotes a large change in the x direction (horizontal edge), and a large Iy denotes a large change in the y direction (vertical edge)

Part 1: Harris corner detector



Part 1: Harris corner detector

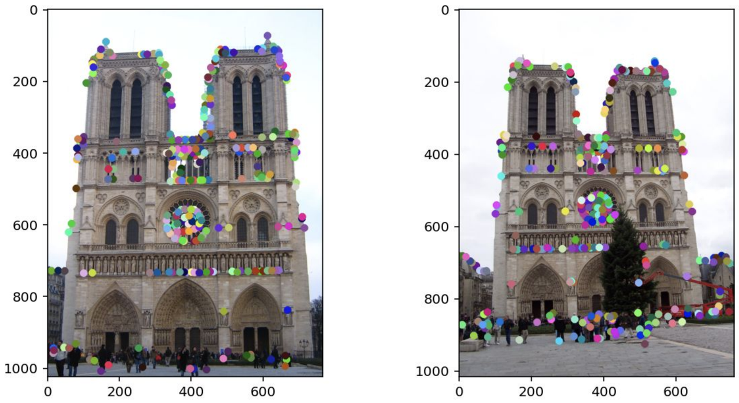




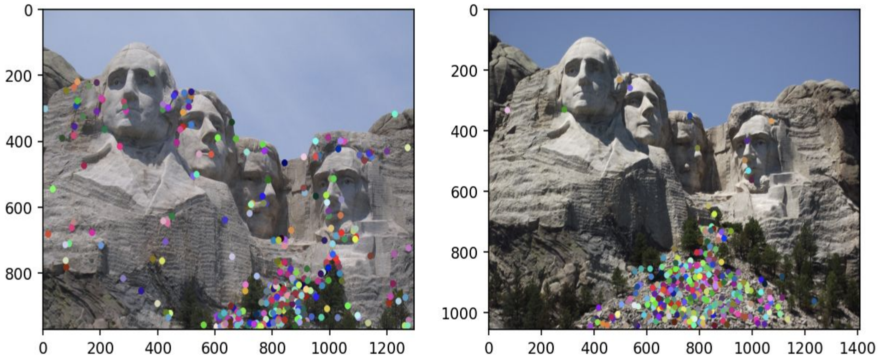
Are gradient features invariant to both additive shifts (brightness) and multiplicative gain (contrast)? Why or why not?

- Gradient features are invariant to both shifts and gains because when taking a gradient, a universal change in the global data field won't have much of an affect on a gradient. All values that the gradient is based on will change at the same rate, so the value should remain relatively similar.

Part 1: Harris corner detector

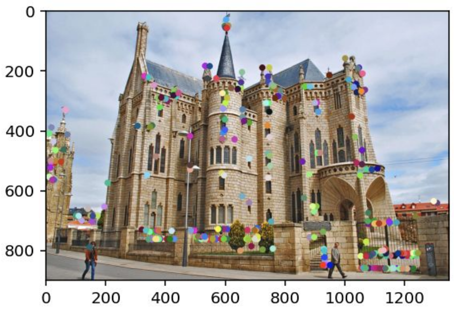


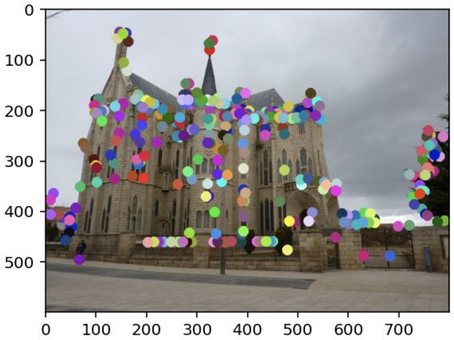
Notre Dame



Mount Rushmore

Part 1: Harris corner detector





Gaudi

What are the advantages and disadvantages of using maxpool for non-maximum suppression?

- Advantages include being able to identify one interest point in an area with multiple global high values; flexibility in terms of selecting window size; and providing limited shift invariance in an image
- Disadvantages include in low local regions, max values are still kept and may throw off results, filtering is required; image discrepancies can result in false max/min values

Part 1: Harris corner detector

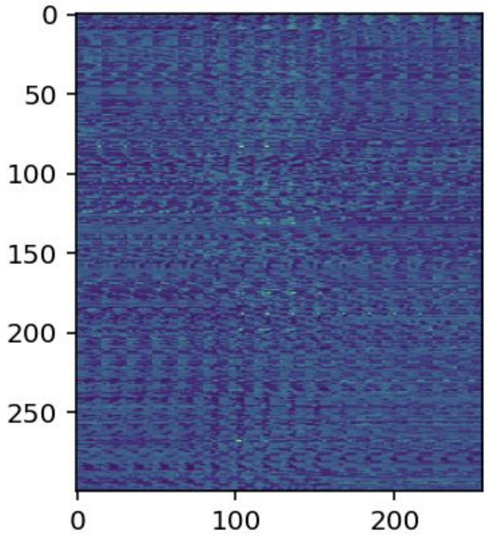
What is your intuition behind what makes the Harris corner detector effective?

- The Harris corner detector is an effective way to detect image changes within an input image because it assigns a unique score to a pixel based on the exact environment it is located in, not an approximation. The gradients of a pixel are calculated based off the surrounding pixel values, and stored to that specific pixel. This allows there to be a large data set of exact data that can be evaluated and scored.

- Harris corner detector is also scale invariant, so the results can be analyzed at different scales/sizes of the input image. Different features become more or less prevalent at certain scales.

- Harris corner detector is also effective because it is a good mathematical way to score pixels and analyze what the score means. Different gradients relate to different values of R, which can be analyzed for more than just corners. It makes understanding the image much easier in a numerical way.

Part 2: Normalized patch feature descriptor



Visual Normalized Patch
Descriptor for Notre Dame

Why aren't normalized patches a very good descriptor?

- Normalized patches aren't a very good descriptor to match two features because they are extremely sensitive to rotations and translations. Even just small displacements or rotations in an image can cause values in an image feature window to change rapidly, causing the patch to not get paired correctly.

Part 3: Feature matching

```
int64
match_features_ratio_test: "Correct"
int64
105 matches from 2464 corners
```

```
int64
match_features_ratio_test: "Correct"
int64
111 matches from 2329 corners
```

Visual feature matching Notre Dame

Visual feature matching Mount Rushmore

Part 3: Feature matching

```
int64
match_features_ratio_test: "Correct"
int64
13 matches from 2463 corners
```

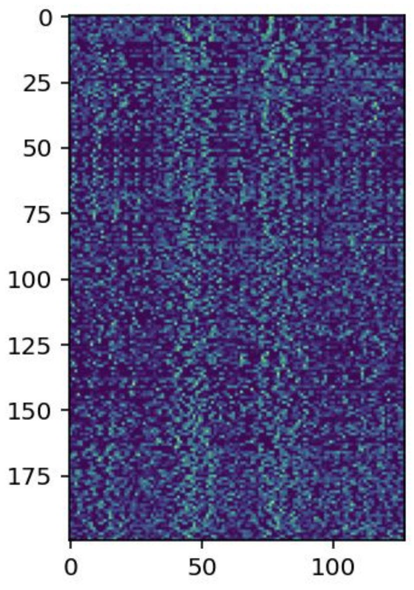
Visual feature matching Gaudi

Describe your implementation of feature matching:

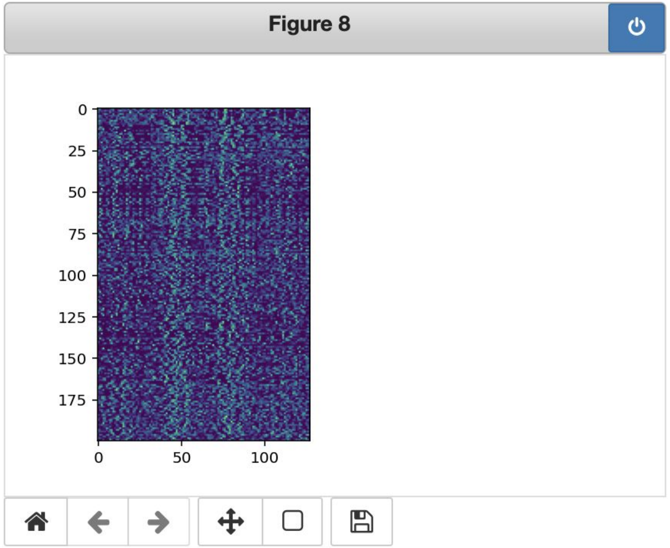
- In part 3, our goal was to determine the closest matching point in one image to another. I began by calculating the distance from one interest point in one image to every other in the second.
- Next, I sorted those distances in descending order and took the 2 smallest as “closest” and “second closest” to their corresponding point in image 1. I computed their NNDR ratio and only kept values indexes in which the NNDR was less than 0.8, showing a strong closest coordination.

Part 4: SIFT feature descriptor

```
"Correct"
SIFT took 4.914172887802124 sec.
```



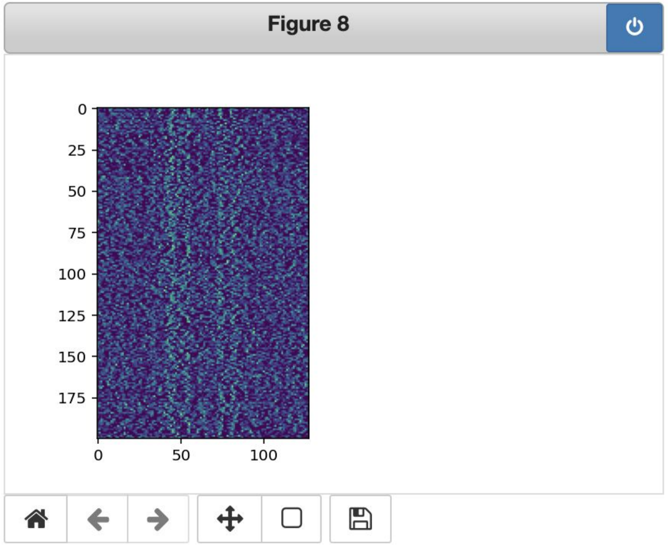

```
"Correct"
SIFT took 4.138158082962036 sec.
```



Visual SIFT matches Notre Dame

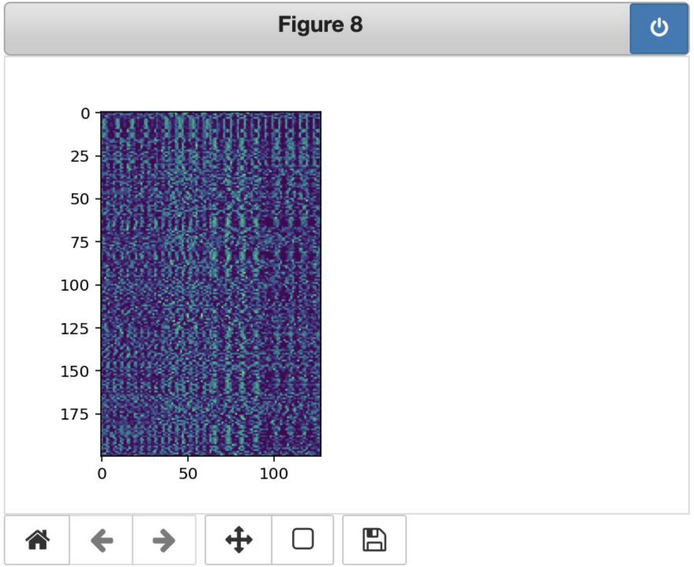
Part 4: SIFT feature descriptor

"Correct"
SIFT took 4.1467461585998535 sec.



Visual SIFT matches Mount Rushmore

"Correct"
SIFT took 3.7387430667877197 sec.



Visual SIFT matches Gaudi

Part 4: SIFT feature descriptor

Describe your implementation of SIFT feature descriptors:

1. The first step in SIFT is to determine the orientation and magnitudes of the gradient of each pixel, to determine the rate of change in direction and mag of the pixel.
2. Next, we create local feature windows around a given point to determine the total change in direction around a given point based on the orientations and sum of magnitudes. This will give us a final total direction and magnitude of a given pixel (interest point)
3. Taking multiple local bins at a time, for example in groups of 4x4 pixels, help make our results more accurate by pooling those pixels and then analyzing the total local window.

Why are SIFT features better descriptors than the normalized patches?

- SIFT features better describe the orientation and magnitude of pixels compared to their surrounding. Normalized patches compare hard-coded values of pixels that are not analyzed to their surrounding pixels in magnitude or orientation, so there is a greater chance for error in comparison.

Conclusion

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

- Our version of SIFT features aren't scale invariant because we didn't take in to account a difference of Gaussian (DoG) approach at the problem. Taking multiple scales of the same image and taking the difference between those scaled images would help analyze the difference between scales, in other words, maximize the scale invariance of images.
- Our version of SIFT features aren't totally rotation invariant because there is error in the ways in which we bin our data. We need to analyze a larger local region of data in order to determine the exact rotational analysis of the image.