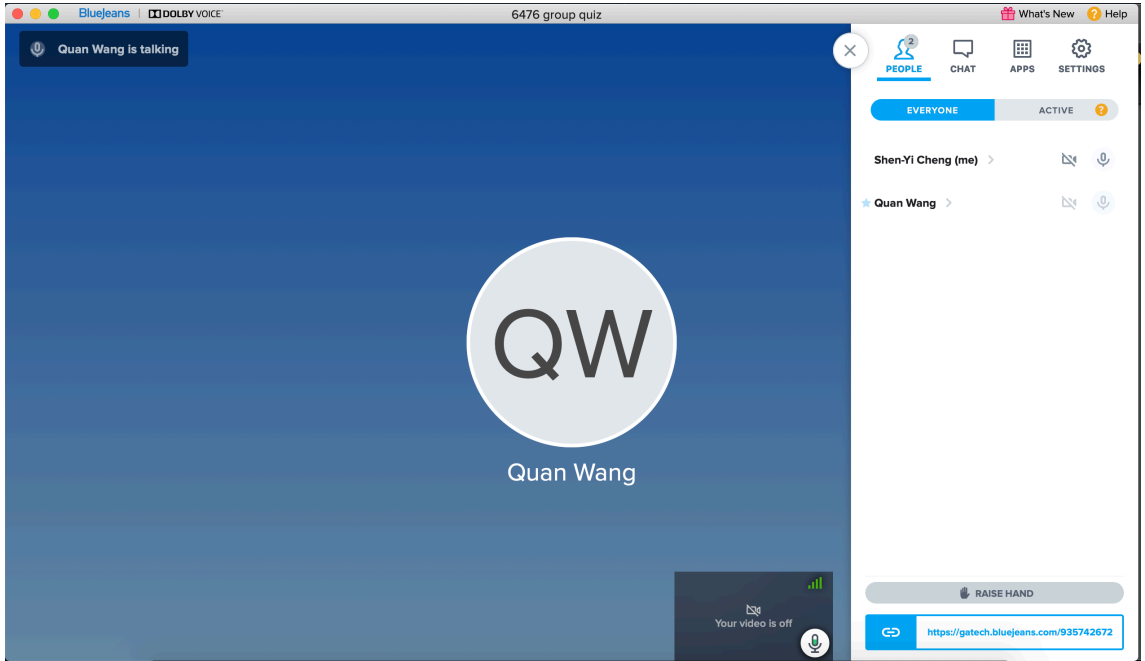
CS 6476 Project 3

<name> Shen-Yi Cheng
<GT username> scheng98
<GTID> 903514405

Gradescope Group Quiz Collaboration Photo

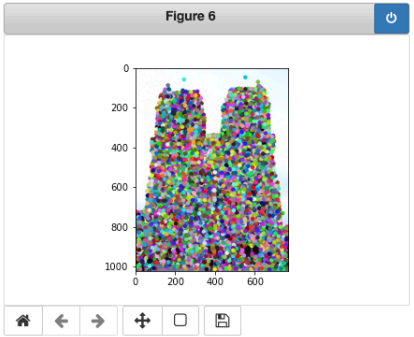
<Insert a picture showing that you and your group met to discuss the quiz and basic concepts of the project>



Part 1: HarrisNet

<insert visualization of Notre Dame interest points from proj3.ipynb here>

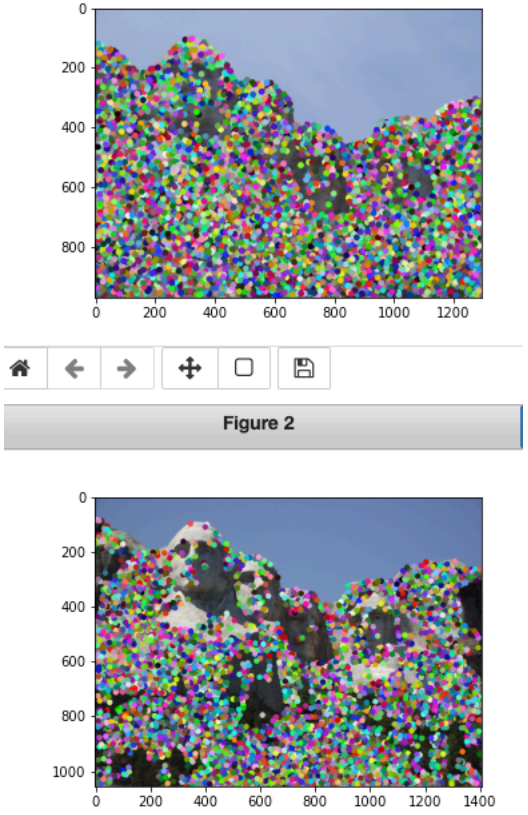
num_points: int = 4500




```
4380 corners in image 1, 4198 corners in image 2
```

< insert visualization of Rushmore interest points from proj3.ipynb here >

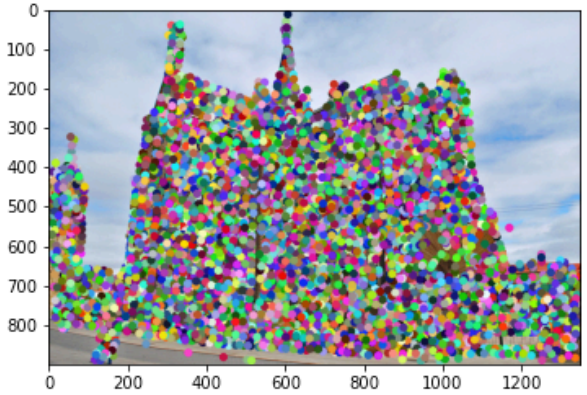
num_points: int = 4500

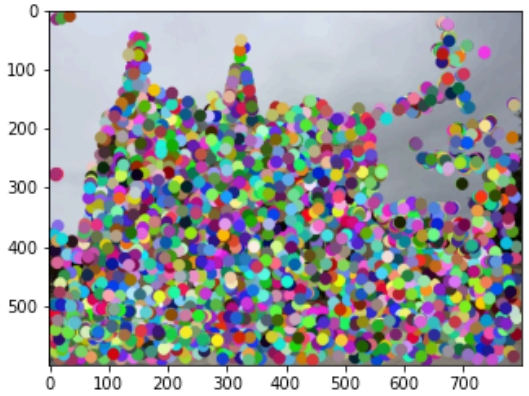


Part 1: HarrisNet

< insert visualization of Gaudi interest points from proj3.ipynb here >

num_points: int = 4500





Part 1: HarrisNet

<Describe how the HarrisNet you implemented mirrors the original harris corner detector process. (First describe Harris) What does each layer do? How are the operations we perform equivalent?)>

HarrisNet:

1st Layer: RBG to grayscale [n, 1, h, w] from Harris Corner Detection Algorithm step 1
2nd Layer: Creating image gradient [n, 2, h, w] from Harris Corner Detection Algorithm step 2
3rd Layer: Create Ix^2, Iy^2 and Ixy^2 [n, 3, h, w] from Harris Corner Detection Algorithm step 2
4th Layer: Create second moment M by gaussian filters [n, 3, h, w] from Harris Corner Detection Algorithm step 3
5th Layer: Calculate Harris corner score R = det(M) – alpha(trace(M))^2 [n, 1, h, w] from Harris Corner Detection Algorithm step 4
6th Layer: Calculate non-maximum suppression(NMS) from Harris Corner Detection Algorithm step 5
Last steep: Locate the interesting points on original image.

Harris Corner Detection Algorithm:

1. **Color to grayscale**
2. **Spatial derivative calculation**
3. **Structure tensor setup**
4. **Harris response calculation**
5. **Non-maxumum suppression**

Part 2: SiftNet

<Describe how the SiftNet you implemented mirrors the Sift Process. (First describe Sift) What does each layer do? How are the operations we perform equivalent?)>

SIFTNet:
1st Layer: RBG to grayscale [n, 1, h, w]
2nd Layer: Creating image gradient [n, 2, h, w]
3rd Layer: Angle cosines and image gradient magnitude[n, 3, h, w] from from Scale Invariant Feature Transform step1
4th Layer: Pass histogram layer to get 8 bin orientation histogram[n, 3, h, w] from from Scale Invariant Feature Transform step1
5th Layer: 8 bin histogram and accumulate over 4x4 subgrids from from Scale Invariant Feature Transform step3
Last steep: Locate potential recognition result on original image. from from Scale Invariant Feature Transform step4

Scale Invariant Feature Transform
1. **Scale-space peak selection**
2. **Key point localization**
3. **Orientation assignment**
4. **Key point descriptor**
5. **Key point matching**

Part 2: SiftNet

- <Explain what we would have to do make our version of Sift rotationally invariant (conceptually)>

By using 8 angle orientation to create a tensor for each feature. It can prevent feature rotation and makes our net detect the same feature but different orientation.

- <Explain what we would have to do to make our version of SIFT scale invariant (conceptually)>

SIFT descriptors are scale invariant because the descriptors are extracted relative to the key point detection scales, that is, a descriptor's actual window size is 16*scale x 16 *scale not 16x16.

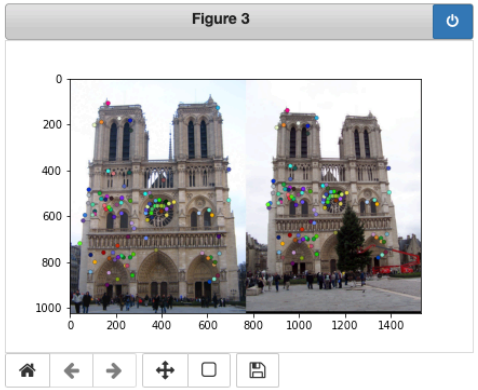
For example, if the scale of an object is doubled then the actual descriptor window relative to the current scaled object becomes 32x32. The dimensionality of the descriptor remains 128 because the orientation binning cells in which sum pooling occurs will also proportionately scale from 4x4 to 8x8.

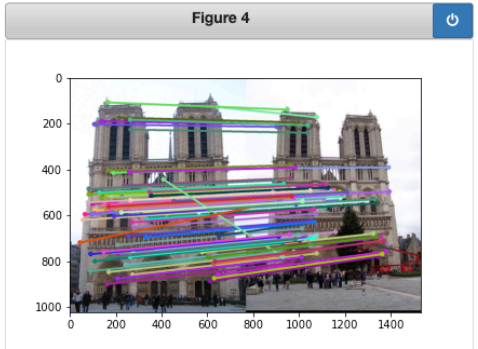
Part 2: SiftNet

- <What would happen if instead of using 16 subgrids, we only used 4 (dividing the window into 4 grids total for our descriptor)>
- In my opinion, I think we will still get some descriptosr but maybe the result do not have too significant difference between each one because their local information are too less to generate different parameter to compute.
- <What could we do to make our histograms in this project more descriptive?>
- Try have mor angles and get more parameter. But the trade-off is it might compute pretty slow.

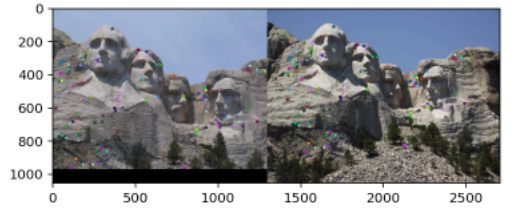
Part 3: Feature Matching

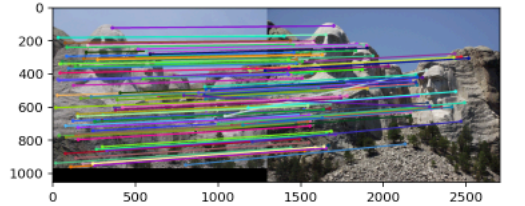
<insert feature matching visualization of Notre Dame from proj3.ipynb>





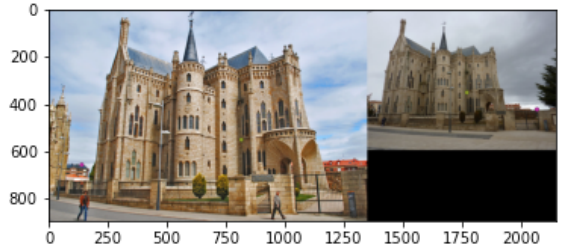
<insert feature matching visualization of Rushmore from proj3.ipynb >

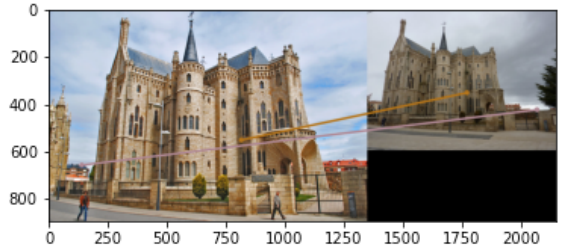




Part 3: Feature Matching

<insert feature matching visualization of Gaudi from proj3.ipynb >



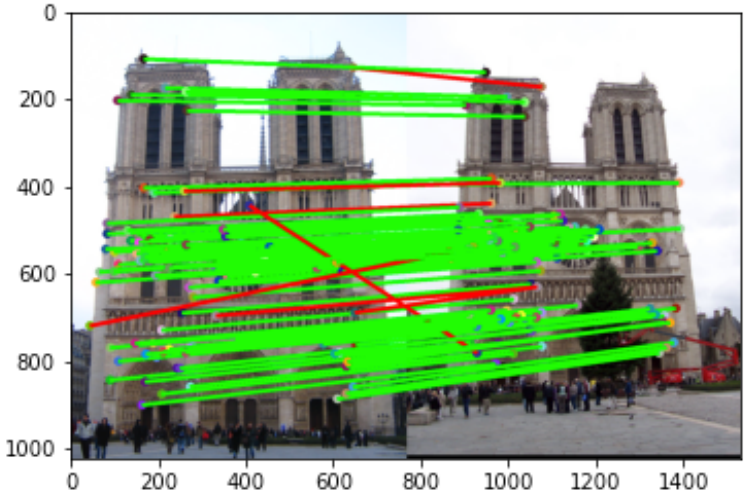


<Describe your implementation of feature matching.>

1. Get the distance between all points.
2. Sort them and extract the two lowest value to get a ratio.
3. Discard the candidate points which ratio are too high
4. Get and return the matches and confidences

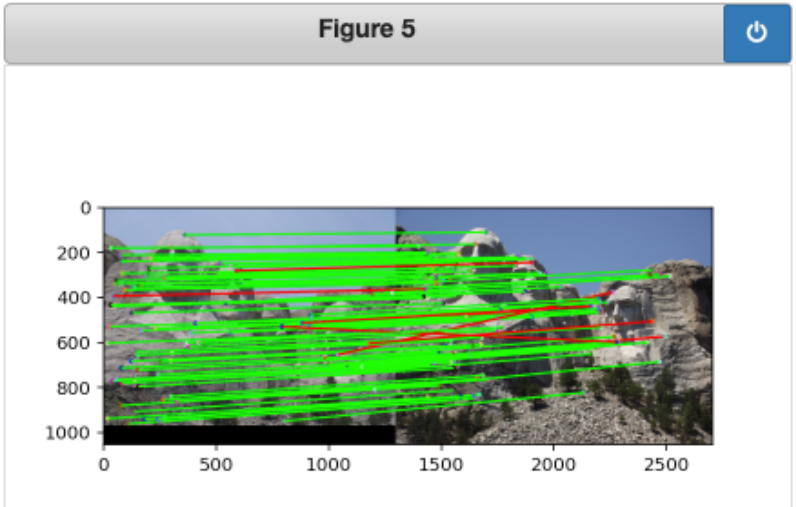
Results: Ground Truth Comparison

<Insert visualization of ground truth comparison with Notre Dame from proj3.ipynb here>



<Insert visualization of ground truth comparison with Rushmore from proj3.ipynb here>

```
You found 100/100 required matches
Accuracy = 0.920000
```

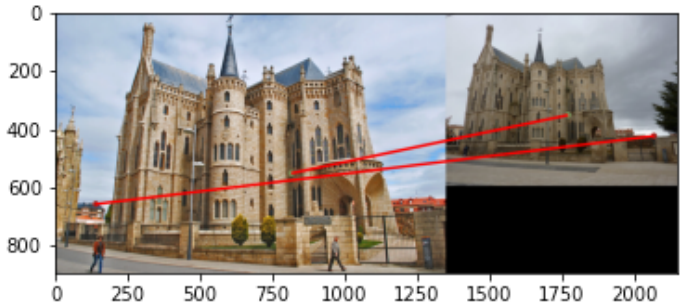


Results: Ground Truth Comparison

<Insert visualization of ground truth comparison with Gaudi from proj3.ipynb here>

<Insert numerical performances on each image pair here.>

```
You found 2/100 required matches
Accuracy = 0.000000
```



	Notre Dame	Rushmore	Gaudi
matches	100/100	100/100	2/100
accuracy	93/100	92/100	0/100

Results: Discussion

- <Discuss the results. Why is the performance on some of the image pairs much better than the others?>
- According to my result, accuracy of Notre Dame and Rushmore are close but Gaudi is pretty low. It could caused by the brightness of two Gaudi image are different and one is sunny on is cloudy. It may cause the descriptor to mismatch or false-recognition.
- <What sort of things could be done to improve performance on the Gaudi image pair?>
- Maybe we can make all the image have the similar background color, exposure, brilliance and contrast before put into network.

Conclusions

<Describe what you have learned in this project. Feel free to include any challenges you ran into.>

I learned how to implement each layer by myself(Of course I read a lot on pizzia and interent, also keep bothering TAs). I guess I ran into problems in every sections. I feel that there is a gap between course material and project content. I am not implying they are bad. I mean both are really useful and interesting to me. However, I need more time to understand what the concept in them and connect them together. Moreover, there is also a gap between deep learning and me. I am new to deep learning. So, there are many things I am learning during using. All of them are really time-consuming and I really enjoy trying to work them out.

Thanks all the TAs especially Haoxin Ma. He really helped me a lot.

Extra Credit: Sift Parameter variations

Extra Credit: Custom Image Pairs