Homework 4 Writeup

Instructions

- Describe any interesting decisions you made to write your algorithm.
- Show and discuss the results of your algorithm.
- Feel free to include code snippets, images, and equations.

Design Choices

Get Interesting Points

Parameters

There are several parameters related to this part of task. Final value was determined from best practice.

- alpha starting from value 0.05 and 0.06 (introduced in the textbook), find the best value that leads to higher accuracy(Top 100 confidnce accuracy). keypoint threshold and local window size are fixed to 0.01 and 8 respectively. Final value for alpha was determined to 0.03.

$$R = det(A) - \alpha \times trace(A)^2 \tag{1}$$

alpha	accuracy(Notre Dame)	accuraccy(Mount RushMore)
0.00	83.00	97.00
0.01	84.00	96.00
0.02	84.00	96.00
0.03	84.00	95.00
0.04	84.00	94.00
0.05	83.00	92.00
0.06	81.00	95.00
0.07	80.00	97.00

Table 1: Top 100 confidence accuracy by alpha.

- keypoint threshold Threshold that filters out whether there is a possibility of being a keypoint. alpha and local window size was determined to 0.03 and 8 respectively. Final value for keypoint threshold was determined to 0.005.

```
keypoint_mask = matrix_R > KEYPOINT_THRESHOLD
```

keypoint threshold	accuracy(Notre Dame)	accuraccy(Mount RushMore)
0.00	77.00	94.00
0.005	87.00	96.00
0.01	84.00	95.00
0.015	80.00	97.00

Table 2: Top 100 confidence accuracy by keypoint threshold.

- local window size

  local window size determines how many keypoints will be preserved. In general, as many keypoints are preserved, accuracy increases, however, it takes much more time in calculating distances. Alpha and keypoint threshold was fixed to 0.03 and 0.005. When the window size was smaller than 4, it took very long time due to the lack of memory. Final value for window size was determined to 4.

Window size	accuracy(Notre Dame)	accuraccy(Mount RushMore)	Total Running Time(s)
2	-	-	-
4	98.00	99.00	149.02
8	87.00	96.00	66.78
16	76.00	98.00	24.43

Table 3: Top 100 confidence accuracy and elapsed time (image generating time was excluded) by window size.

Get Descriptors

Match Features

Distance Metrics

There were two main ways to get the distance of features. *Euclidean distances* and *Cosine Similarity*. Two method showed difference in accuracy and running time. Generally, Euclidean distance were more accurate, but it took much more time than Cosine Similarity. Since I achieved high accuracy with cosine similarity, I adopted it as my final model.

Clamping

In order to get more sophisticated distance value, I clamped magnitude value that are too big compared to other features. When Clamp thershold is 1.0, it means no clamping. Final value for clamp threshold was determined to 0.2.

```
features1[features1 >= CLAMP_THRESHOLD] = CLAMP_THRESHOLD
features2[features2 >= CLAMP_THRESHOLD] = CLAMP_THRESHOLD
```

Clamp Threshold	accuracy(Notre Dame)	accuraccy(Mount RushMore)
0.1	84.00	97.00
0.2	98.00	99.00
0.4	90.00	97.00
0.6	89.00	96.00
1.0	88.00	96.00

Table 4: Top 100 confidence accuracy by clam threshold.

A Result

Get Interesting Points

Figure1 shows how interesting points are extracted. In the middle of Figure1, we can see edge gradients are eliminated, and only the corner preserved. And finally, only the local maximum points are preserved as in right image.

Figure 1: *Left:* Original Image. *Middle:* Result of Corner Detection *Right:* Local Maximum Points

Get Descriptors

Figure3 shows how the gradient of eight directions are extracted. Figure2 is the window for original image. Each of these gradients will be summed by 4*4 blocks.

Match Features

Figure4 shows the Improvement of keypoint matching compared to the starter code.

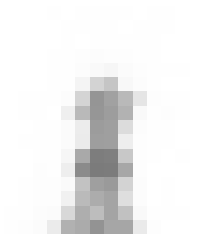

Figure 2: Original keypoint image

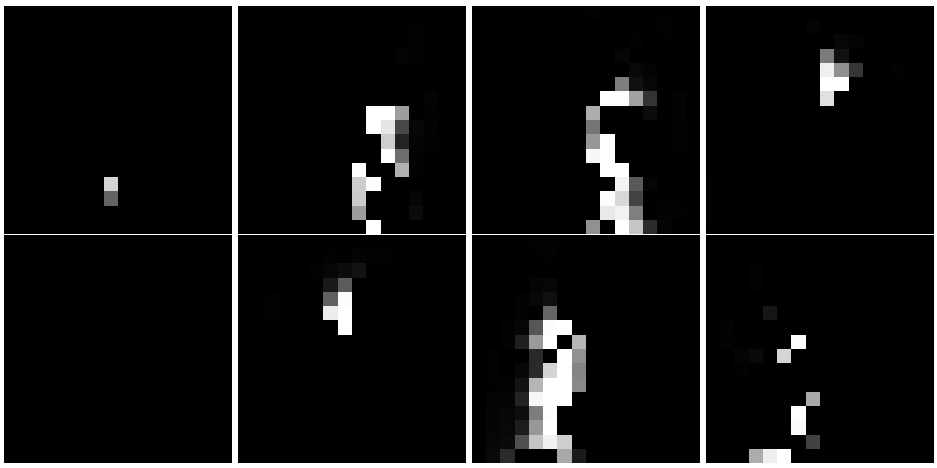

Figure 3: image gradient for eight directions

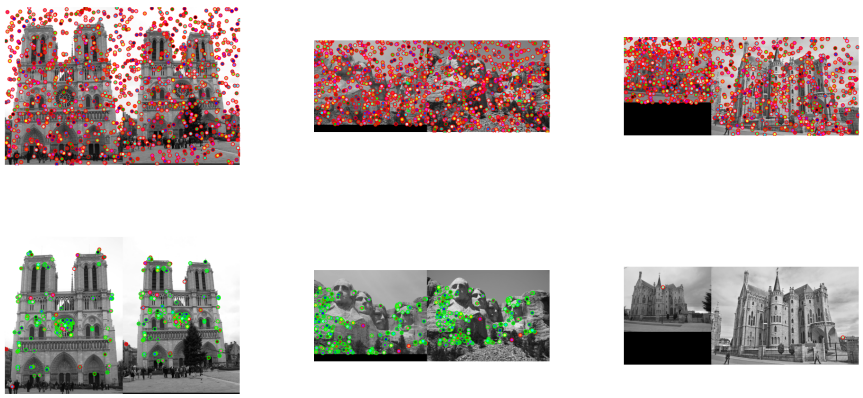

Figure 4: keypoint matching result compared to the starter code. UP: Result by starter code *Down:* Result by my model