Computer Vision Homework 2, Part 1 Questions

Siddharth Iyer

1 Q1

When answering this question, it's important to consider that high and low pass filters filter by frequency, which is what can lead to increased/decreased edge intensity and noise, respectively. With a low-pass filter, we will see a reduction in edge contrast due to the loss of high-frequency components. Low-frequency components will include smaller gradients, which will make edges less pronounced. The resulting reduction of the gradient products will result in smaller eigenvalues, as the eigenvalues are calculated from the determinant/trace of the M matrix. With a high-pass filter, we will see an increase in edge contrast and noise. This results in more pronounced edges and corners, due to the inclusion of larger gradients. The steeper gradients will then result in larger eigenvalues.

2 Q2

Algorithm 1 SIFT Descriptor Calculation

```
procedure SIFT_DESCRIPTOR(x, y)
    descriptor ← np.zeros(128, 1)
    patch_start_x ← x − 8
    patch_end_x ← patch_start_x + 15
    patch_start_y ← y − 8
    patch_end_y ← patch_start_y + 15
    for i ← 0 to 3 do
        for j ← 0 to 3 do
            tmp_histogram[orientations[i][j]] += (mag[i][j])
            descriptor.append(tmp_histogram)
        end for
    end for
    np.normalize(descriptor)
    return descriptor
end procedure
```

In SIFT Feature Description, we begin with an interest point. From there, we determine our patchby centering a 16x16 pixel region surrounding the interest point. After this patch is

determined, we divide this patch into a grid of 4x4 squares. Each square will be 16/4 = 4 pixels wide and will be used to compute histograms of gradient orientation. The histogram divides the 0 to 2 π range of potential orientations into 8 bins. These 8 bins will be appended for the 16 squares within the patch, giving us 8 * 16 = 128 orientations to describe our interest point.

3 Q3

The main difference in Geometric Interpretations between Euclidean Distance and Cosine Similarity is in the specific metric each method uses to compare features. Euclidean distance computes the root of the difference of the square of the nth dimension value from some feature 1 and some feature 2. This summation and squaring is impacted by the magnitude of the gradient and is thus a good choice for a geometric interpretation when dealing with features where magnitude differences are relevant. Cosine difference, however, ignores magnitude entirely and computes the cosine of the angle between two vectors. The magnitude of the vectors can be scaled however the user wants, but this will not change the angle at which they meet. This metric rather measures the orientation of the vectors in relation to each other. Naturally, this interpretation is more suited when dealing with features with patterns, where relative differences between features are compared to determine closeness.

In feature descriptor matching, an effective strategy involves employing Nearest Neighbors along with a Ratio Test, as practiced in this lab. This method shines by comparing distances to the two nearest neighbors, allowing for a precise filtering of candidates through magnitude thresholds to minimize false positives effectively. The Ratio Test plays a crucial role here, as it uses magnitude as a key metric for comparison, ensuring that only the most accurate matches are considered. This approach smartly narrows down potential matches, leveraging the precision of magnitude-based comparisons to enhance the reliability of the matching process. Since magnitude is essential for this matching approach, this is a great option to use given Euclidean distance as your comparison metric