Hao Le  
A15547504  
ECE 172A Winter 2022 HW3

*Academic Integrity Policy: Integrity of scholarship is essential for an academic community. The University expects that both faculty and students will honor this principle and in so doing protect the validity of University intellectual work. For students, this means that all academic work will be done by the individual to whom it is assigned, without unauthorized aid of any kind. By including this in my report, I agree to abide by the Academic Integrity Policy.*

1. **Histograms and Adaptive Histogram Equalization**

i.

Text

Description automatically generated with low confidence

ii.

Text

Description automatically generated

iii.

Plotting function was made:

Text

Description automatically generated

Main function calls to read image and plot histograms:

Text

Description automatically generated

Chart, bar chart, histogram

Description automatically generated

Chart, histogram

Description automatically generated

iv.

Main function calls to flip image and plot histograms:

**Text

Description automatically generated**

Chart, bar chart, histogram

Description automatically generated**Chart, histogram

Description automatically generated**

Histograms are invariant to the vertical flip.

v.

Main function calls to double the red channel, show the image, and plot the histograms:

A screenshot of a computer

Description automatically generated with medium confidence

A couple deer in a forest

Description automatically generated with low confidence

Chart, histogram

Description automatically generated

Chart, histogram

Description automatically generated

The histograms make sense since 1. Red makes up the mid-range of luminosity, hence we see a boost in that range for the gray plot and 2. We see that the red channel’s normalized histogram is more even, meaning that we are getting more amount of red across its value spectrum from 0 to 255. The reason why the normalized frequency did not stay the same is because some values after being doubled go over 255, thus clipping at 255 – this explains why we see the trough at the high range “fill up” and even out.

Text

Description automatically generated

Main function calls to load beach.png, apply premade histogram equalization function, and apply own adaptive HE function with varying window sizes. Also each output image was saved:

Text

Description automatically generated



Original



After applying SciKit-Image’s HE function

Image after applying my own AHE function:





Window size: 129

Window size: 65

Window size: 33

A group of people in a small house

Description automatically generated with low confidence

After applying both HE and AHE, the contrast of the image is significantly boosted – we see this especially in the texture of the ground and the house. Also, we see dark areas such as the window where the people are and underneath the house get brightened up – for AHE with window size 129, we can actually make out their faces, and even notice the fourth person who was barely seen in the original image.

HE in this case does not greatly distort the original image to the point where it is not aesthetically pleasing anymore. The sky is clipped greatly, but other than that, the brightness boost is subtle. It still fails in brightening up the window area where the people subjects are.

AHE performs best with the window size of 129, or ideally bigger, since it does not create over contrasted areas due to the window size being small, thus creating large-gapped quantization of pixel intensities. It still suffers in clipping of the faces, and the sky is also clipped to darkness because of how even it is in the original image – the algorithm automatically turns this area of all white (pixel value of 1) to 0 since the rank is always 0. Overall, AHE does a good job at boosting the texture of the house and the ground, giving the image more even luminosity range.

Although in this case AHE performs better than HE in the beach image, images where there are large bright areas of even value can cause AHE to inadvertently darken those areas unnaturally. Moreover, images with already good contrast to begin with can get over contrasted by AHE, especially with smaller window sizes. Those images are better with the subtle equalization that HE offers to get more luminosity range.

1. **Filtering**
2. The mural can be found at Price Center

2.

Chart, histogram

Description automatically generated

A picture containing graphical user interface

Description automatically generated

There is a disproportionate occurrence of 0 (black) and 255 (white) pixels, compared to the histogram of the original image, implying the image is corrupted with salt and pepper noise. Moreover, we see higher frequencies at the high range beside from 255, pointing to the presence of some Gaussian noise with a mean near that range.

3.

**Table

Description automatically generated with low confidence**

Like mural\_noise1.jpg, there is disproportionate occurrence of both white and black pixels. However, we do not see any perturbance for the other values. Thus, there is only salt and pepper noise.

4.

Both the mean and median filter implementations reused a good deal of the AHE function due to the sliding kernel functionality. Moreover, computation was improved by replacing the two nested for loops to iterate over the pixels of the kernel with numpy slicing. The images were padded symmetrically.

Text

Description automatically generated

A computer screen capture

Description automatically generated with medium confidence

The main script is as follows, and performs four experiments on both noisy mural images. It saves the filtered images, and logs the computation times:

Text

Description automatically generated

**Text

Description automatically generated**

**Mural\_noise1**

Mean 5x5

A group of people standing in a room

Description automatically generated with medium confidence**Chart, histogram

Description automatically generated**

Mean 21x21

 Chart, histogram

Description automatically generated

Chart, histogram

Description automatically generatedMedian 5x5

A group of people standing in a room

Description automatically generated with medium confidence

Chart, histogram

Description automatically generatedMedian 21x21



**Mural\_noise2**

Mean 5x5

A group of people standing in a room

Description automatically generated with medium confidence Chart, histogram

Description automatically generated

Mean 21x21

 Chart, histogram

Description automatically generated

Median 5x5

A group of people standing in a room

Description automatically generated with medium confidence Chart, histogram

Description automatically generated

Chart, histogram

Description automatically generatedMedian 21x21



5.

From the log, 5x5 median filter performs the fastest, followed by 21x21 media, 5x5 mean, and 21x21 mean.

6.

I would use median 21x21 filter because the salt and pepper noise seem to go away, with the side effects of the whole mural being more contrasted, which aesthetically looks better anyways, and the details of the people and objects fading – the image is unsharpened. The mean 21x21 filter also does a good and similar job, though does not have as strong of a contrasting effect.

7.

I would use median 21x21 filter for the same reasons in 6, though I can picture using the 5x5 median filter for both since the noise adds an artist “grain” effect to it.

8.

This expression assumes that the i and j’s are indexing the corresponding absolute coordinates of the original image when the sliding window is over a certain portion i.e. i and j need not be “offset” for template image, but need to for original image depending on where the top left corner of the template is.

This expression is evaluating the square error of the template vs. the original. A match would be minimizing this mean squared error.

9.

We can minimize the whole expression by minimizing what is being summed:

So to minimize the original expression, we want to maximize *2fi*

Code for both 10 and 11:

A screenshot of a computer

Description automatically generated

10.

**A group of people standing in a room

Description automatically generated with medium confidence**

The matching function thinks that the template matches where the candle is. This is a bright spot on the image. Since we are trying to maximize 2fi, the ideal multiplier with the template is the highest value, which is 255. Thus, when the sliding template passes by that area, it will get multiplied by the all 255 values of the candle, outputting the highest match values there.

11.

A group of people standing in a room

Description automatically generated with medium confidence

Now the matching function is actually working. Since it is normalized to the intensity value of both the template and the windowed area of the original image, this scales back the output match value to be relative to the intensities of the window. Since values can only go from 0 to 255, really bright areas of the original image will cause the match value to clip and 255-out. By normalizing, this keeps the match value in a reasonable range, and puts more emphasis on the matching of features.

1. **Canny Edge Detector**
2. A GaussianSmoothing function is made to perform the sliding kernel operation over the image with the Gaussian weights, followed by normalization:

Text

Description automatically generated

The main script is as follows:

Text

Description automatically generated

It reads the lane.png image and performs the Gaussian smoothing filter and shows the results, along with other operations that will be discussed in the proceeding problems.

The output image:

A bus driving down a road

Description automatically generated with medium confidence

The image is free of the original grid-like noise, and the edges are well-preserved, an advantage of the Gaussian filter over a mean filter.

1. A Sobel function is made:

Text

Description automatically generated

This will return three images:

* Magnitude image where each pixel value represents the magnitude of the gradient vector
* Angle image where each pixel value represents the angle of the gradient vector, limited to a range of (-pi/2, pi/2) by nature of arctan
* Magnitude-angle image that is similar to the angle image, only showing the edges more cleanly

The main script is as follows to show the three images:

Text

Description automatically generated

A train on the railway tracks

Description automatically generated with low confidence

Chart

Description automatically generated with low confidenceA black and white photo of a train track with trees on the side

Description automatically generated with low confidence

Magnitude-Angle

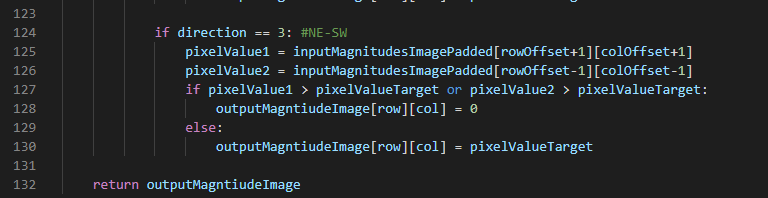
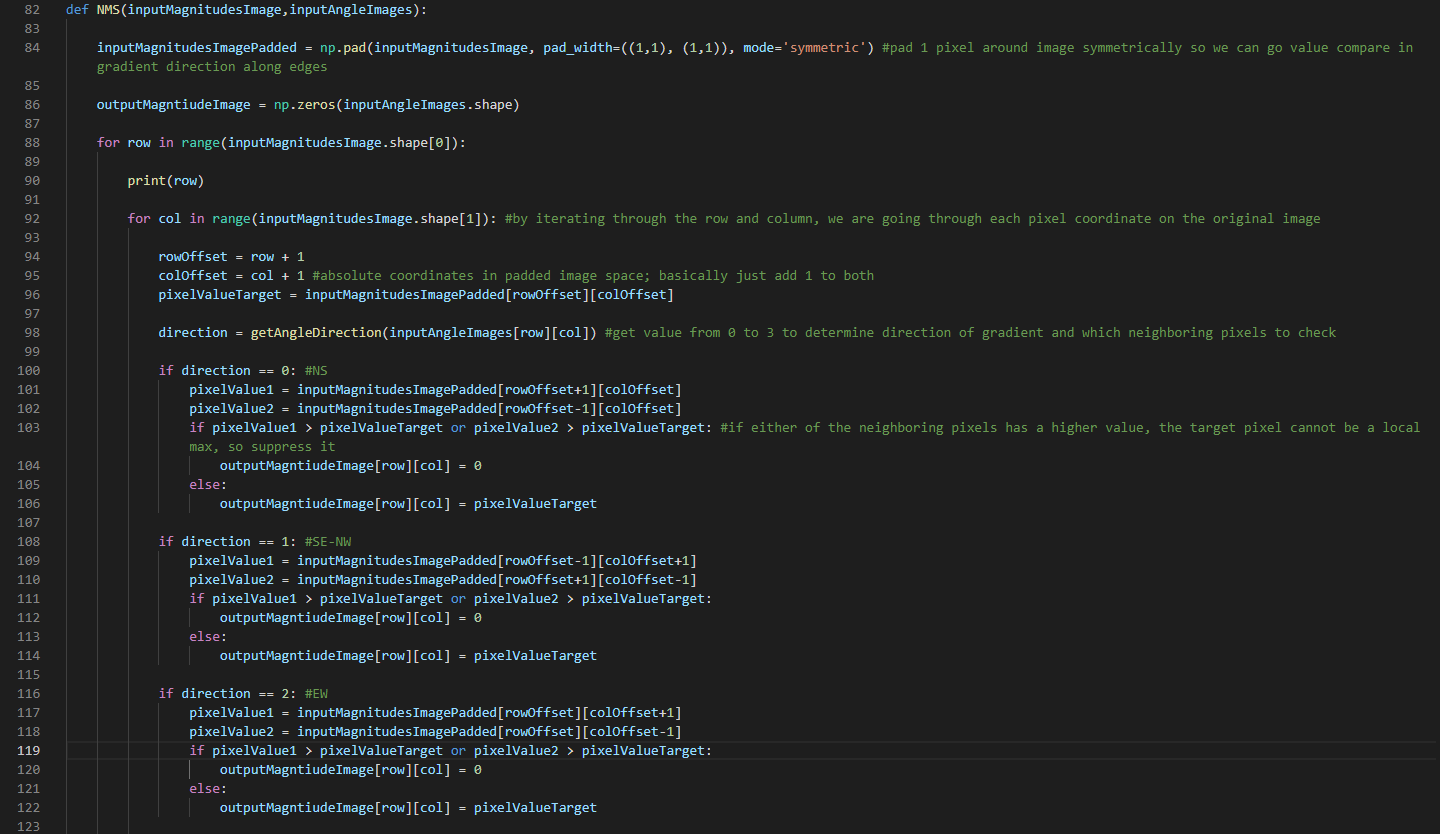
Angle

Magnitude

From the output, the Sobel operation is working by extracting the prominent edges in the original image. Though the angle image is noisy and unintuitive, the magnitude-angle image does a good job of showing the range of angles for each edge. For example, we can see that the straight lines of the road all take on a single color, implying that the angle stays consistent throughout which is expected. We can also see on the bus wheel the range of colors being cycled which is also expected for a circular shape.

3.

A non-maximum suppression function is made:



The function uses a getAngleDirection function which takes in the angle output of arctan, and outputs which direction of the 8-count neighboring directions the angle belongs to:

Text

Description automatically generated

NMS will compare the neighboring pixels, in the direction of the gradient, of each pixel in the magnitude image, and determine whether the target pixel is a local maximum in value. If not, then suppress the pixel by setting the value to 0. The result is that we get 1 pixel-wide, clean edges:



A train track with trees on the side

Description automatically generated with low confidence

4.

Now the NMS image is thresholded with a pixel value of 110 i.e., any pixel value in the image that is lower than or equal to 110 gets set to 0:



A picture containing diagram

Description automatically generated

The result is cleaner than the NMS image because we removed the faint edges on the road and especially the trees. Plus, the threshold value picked preserved the outline of the road which is critical for autonomous driving vision.