CSE 107: Lab 04: Histogram Equalization.

<your name>
LAB: T 10:30-1:20pm
Yuxin Tian
November 7, 2022

Abstract:

In a few sentences, describe the purpose of this lab. (Do not mention specific Python functions. This description should be at a very high level.)

Results:

Figure 1 contains the dark image and figure 2 contains the histogram of the dark image. Figure 3 contains the equalized dark image and figure 4 contains the histogram of the equalized dark image. The following table contains the mean and standard deviations of the dark image and the equalized version of the dark image.

	Mean pixel value	Standard deviation of the pixel values
Dark image	82.816910	60.353374
Equalized dark image	128.906438	73.687541

Figure 1. The dark image.

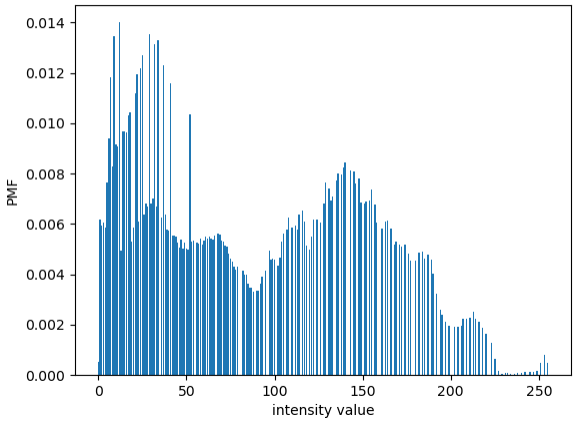

Figure 2. The histogram of the dark image.

Figure 3. The equalized version of the dark image.

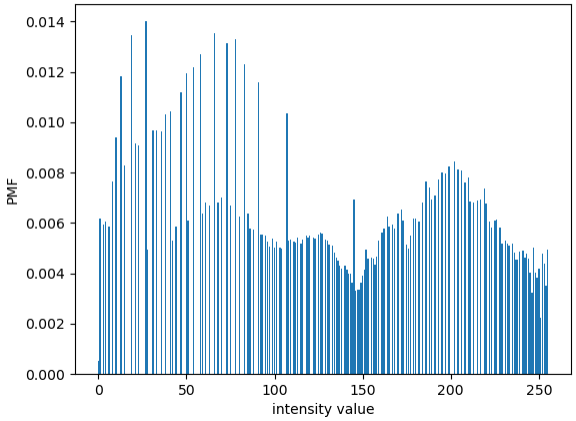

Figure 4. The histogram of the equalized version of the dark image.

<repeat the above for the light image>

Questions:

<Your answers to the assignment questions>

```
# Import pillow
from PIL import Image, ImageOps

# Import numpy
import numpy as np
from numpy import asarray

################################################################################
# Perform histogram equalization on the dark image.
################################################################################

# Read the dark image from file.
dark_im = Image.open('Lab_04_image1_dark.tif')

# Show the image.
dark_im.show()

# Create numpy matrix to access the pixel values.
# NOTE THAT WE WE ARE CREATING A FLOAT32 ARRAY SINCE WE WILL BE DOING
# FLOATING POINT OPERATIONS IN THIS LAB.
dark_im_pixels = asarray(dark_im, dtype=np.float32)

# Import compute_histogram from My_HE_functions.
from My_HE_functions import compute_histogram

# Compute the histogram of the dark image.
dark_hist = compute_histogram( dark_im_pixels )

# Import plot_histogram from My_HE_functions.
from My_HE_functions import plot_histogram

# Plot the histogram for the dark image.
plot_histogram( dark_hist )

print('Dark image has mean = %f and standard deviation = %f' % \
    (np.mean(dark_im_pixels), np.std(dark_im_pixels)))

# Import equalize from My_HE_functions.
from My_HE_functions import equalize

# Apply histogram equalization to the dark image.
equalized_dark_im_pixels = equalize( dark_im_pixels );

# Create an image from numpy matrix equalized_dark_image_pixels.
equalized_dark_image = Image.fromarray(np.uint8(equalized_dark_im_pixels.round()))

# Show the equalized image.
equalized_dark_image.show()

# Save the equalized image.
equalized_dark_image.save('equalized_dark_image.tif');

# Compute the histogram of the equalized dark image.
equalized_dark_hist = compute_histogram( equalized_dark_im_pixels )

# Plot the histogram for the equalized dark image.
plot_histogram( equalized_dark_hist )

print('Equalized dark image has mean = %f and standard deviation = %f' % \
    (np.mean(equalized_dark_im_pixels), np.std(equalized_dark_im_pixels)))

################################################################################
# Perform histogram equalization on the light image.
################################################################################
```

test_HistogramEqualization.py

```
# Read the light image from file.
light_im = Image.open('Lab_04_image2_light.tif')

# Show the image.
light_im.show()

# Create numpy matrix to access the pixel values.
# NOTE THAT WE WE ARE CREATING A FLOAT32 ARRAY SINCE WE WILL BE DOING
# FLOATING POINT OPERATIONS IN THIS LAB.
light_im_pixels = asarray(light_im, dtype=np.float32)

# Compute the histogram of the light image.
light_hist = compute_histogram( light_im_pixels )

# Plot the histogram for the light image.
plot_histogram( light_hist )

print('\nLight image has mean = %f and standard deviation = %f' % \
    (np.mean(light_im_pixels), np.std(light_im_pixels)))

# Apply histogram equalization to the light image.
equalized_light_im_pixels = equalize( light_im_pixels );

# Create an image from numpy matrix equalized_light_image_pixels.
equalized_light_image = Image.fromarray(np.uint8(equalized_light_im_pixels.round()))

# Show the equalized image.
equalized_light_image.show()

# Save the equalized image.
equalized_light_image.save('equalized_light_image.tif');

# Compute the histogram of the equalized light image.
equalized_light_hist = compute_histogram( equalized_light_im_pixels )

# Plot the histogram for the equalized light image.
plot_histogram( equalized_light_hist )

print('Equalized light image has mean = %f and standard deviation = %f' % \
    (np.mean(equalized_light_im_pixels), np.std(equalized_light_im_pixels)))
```

```
# MyHEFunctions.py

# Import numpy
import numpy as np

def compute_histogram( image_pixels ):

<your function header>

<your implementation>


def equalize( in_image_pixels ):

<your function header>

<your implementation>


def plot_histogram( hist ):
    # plot_histgram  Plots the length 256 numpy vector representing the normalized
    # histogram of a grayscale image.
    #
    # Syntax:
    #   plot_histogram( hist )
    #
    # Input:
    #   hist = The length 256 histogram vector..
    #
    # Output:
    #   none
    #
    # History:
    #   S. Newsam     10/23/2022   created

    # Import plotting functions from matplotlib.
    import matplotlib.pyplot as plt

    plt.bar( range(256), hist )

    plt.xlabel('intensity value');

    plt.ylabel('PMF');

    plt.show()
```