UNIVERSITY OF CALIFORNIA, SANTA BARBARA
Department of Electrical and Computer Engineering

ECE 178 **Image Processing** Fall 2020

Homework Assignment #2

(Due on Wednesday 10/21/2020 by 6 pm)

Problem # 1. Histogram operations in the perfect world of probability theory:

a) Suppose an input continuous image has intensity PDF of

$$f_X(x) = \cos(x)u(x)u(\frac{\pi}{2} - x)$$

where $u(\cdot)$ is the unit step function. First, convince yourself that this PDF is zero outside the interval $[0, \pi/2]$ and draw a little plot of it for yourself. (Just to make sure everyone is on the same page). Now, find the histogram equalizing transformation $y = g_a(x)$, i.e., such that the output image has intensity PDF

$$f_Y(y) = u(y)u(1 - y),$$

noting, of course that this PDF is zero outside the unit interval. Finally, as sanity check, verify that g_a maps correctly the end points of the input interval to the end points of the output interval.

b) Next, consider another "desired" output PDF,

$$f_Z(z) = \frac{1}{z}u(z - 1)u(e - z),$$

where e is the usual mathematical constant used as base for natural logarithms, etc. Find the transformation $z = g_b(x)$ that would map the input image (whose PDF was $f_X(x)$ to an output image with intensity PDF as desired ($f_Z(z)$. Again, for sanity check, verify that the end points of the input interval are mapped correctly.

Problem # 2. Some "practical" histogram operations with pencil and paper:

a) Consider the following 2-bit per pixel image:

$$f = \begin{matrix} 1 & 0 & 0 & 1 & 2 & 3 \\ 2 & 1 & 1 & 0 & 3 & 1 \\ 3 & 2 & 2 & 2 & 2 & 1 \end{matrix}$$

Calculate the normalized histogram (level probabilities) and plot the corresponding PDF (should consist of a bunch of delta functions). Then calculate and plot the CDF.

b) Perform histogram equalization and determine the equalized image. (For uniformity let's use the convention that 0.5 is rounded up to 1)

c) Next consider the following histogram of a 3-bit per pixel image that consists of 4096 pixels.

ℓ_k	0	1	2	3	4	5	6	7
n_k	91	300	800	900	80	275	650	1000

Perform histogram equalization for $L = 8$ (3-bit per pixel) output, and then equalize for $L = 4$ (2-bit per pixel) output.

d) Finally, perform a transformation, given the histogram of (c) to approximate the desired histogram:

ℓ_k	0	1	2	3	4	5	6	7
n_k	1000	650	275	80	900	800	300	91

Note that it might be safer (until you have enough practice) to plot a staircase function so that you can carefully determine its inverse.

Problem # 3. *A hands-on problem.* In this problem, we are going to understand the limitations of simple linear stretching and then explore histogram equalization.

a) Apply your linear stretching code from HW1 on the provided image *low_contrast.bmp* and obtain the resulting image and its histogram. How does that compare to the original image and histogram? Why does linear stretching fail to produce a stretched histogram in this case?

b) *Histogram equalization* : **Write your own code** to perform histogram equalization as seen in class to get an "approximately flat" histogram for the resulting image. Perform equalization on *low_contrast.bmp*. How does the equalization output image compare to the original image and to the image obtained by linear stretching?

c) *Histogram transfer* : Given the image pair *image1.bmp* and *image2.bmp*, **write your own code** to perform histogram transformation such that the histogram of the first image is transformed to approximate the histogram of the second. Observe the output image and its histogram and compare it with the histogram of the second image.

You are not allowed to use any sort of built-in function from Python or MATLAB that would perform histogram equalization or transformation. Please write your own code. Note that it is, however, allowed to use built-in functions to display the histogram of an image, and you are not required to write your own code for that purpose. Submit a single .zip named as YourLastName_HW2.zip containing your answers for problem 1 and 2, and your code and output images + histograms for problem 3.