# IC152: Assignment 8 Image Histograms and Line Of Fit

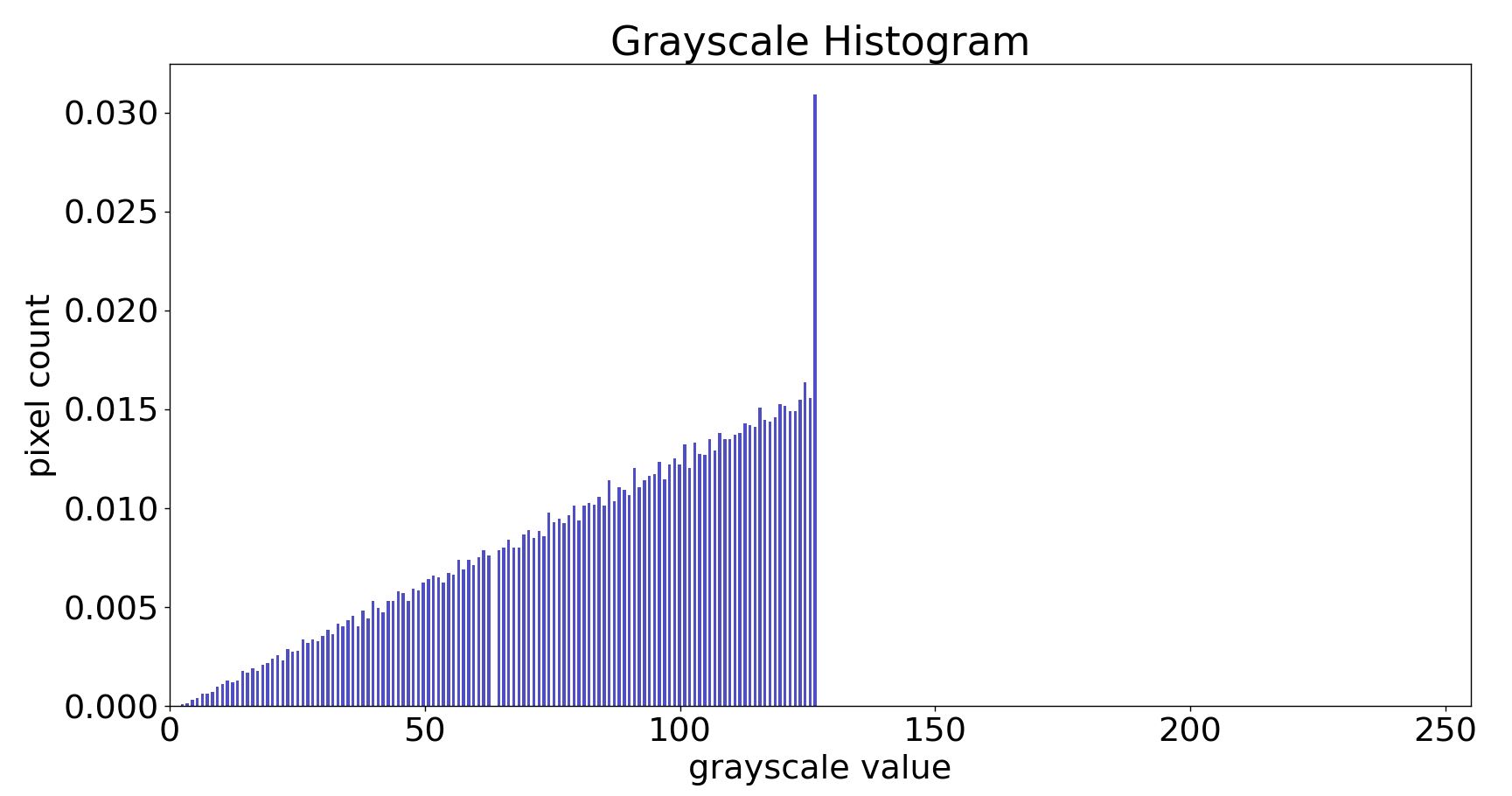
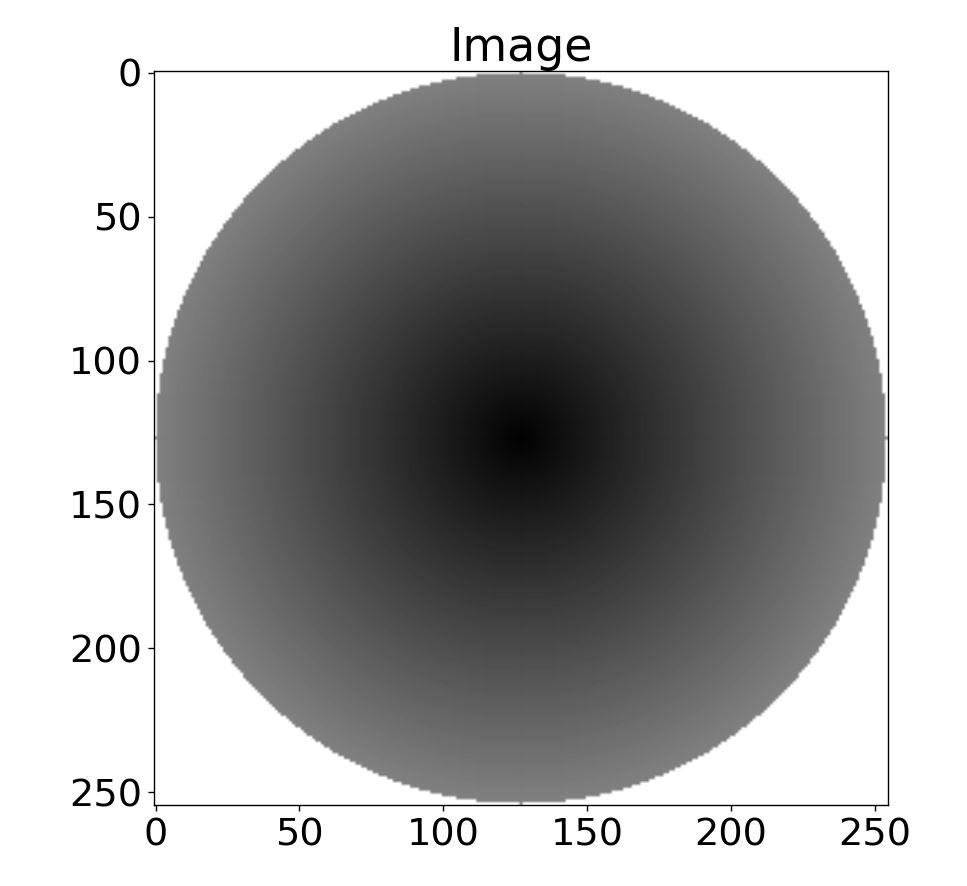
Submission instructions are the same as previous assignments. -2 marks for each missing algorithm on pen and paper.

1. Run problem1.py and try to understand it from comments. Don’t forget to check the file “problem1Op.png” created by the code.

After you analyze, write the algorithm for drawing a circle on pen and paper (Hint: you have to loop over the width and height of the image and modify some pixels based on their location).

* Slides for plotting are shared on LMS (if you wish to go deeper), however you do not need to worry about plotting in this assignment as code for plotting is shared in the python files/helper codes.
* You can assume plotting functions to be available while writing algorithms.
* Ignore the constant scaling factor on the y-axis if needed throughout the assignment.

1. First write the algorithm with pen and paper, and then modify the script shared in question 1 to create the following figure and histogram (note that bar at value 255 should be carefully removed from histogram). Save a python file with name problem2.py.



Do not worry about the height of the last bar (shown above) at value near int(255/2). Only requirement for the histogram is that it should be increasing in range [0, int(255/2)).

1. Now read the 'problem3Input.jpg' image using the “helper code” given below and create the histogram of:
   1. Its pixel values
   2. difference between each pixel with a pixel left to it (loop for column number from 1 to end). What do you observe? Make necessary corrections if the output histogram does not have negative values on x-axis, also reasoning why it happened. Share with the TA/instructor for evaluation.

# helper code starts:

# install pillow using:

# pip3 install pillow

# if above does not work and you are on windows, try:-

# py -m pip3 install pillow

# importing Image and ImageOps from PIL

from PIL import Image, ImageOps

# creating image object:

imInp = Image.open('problem1cInput.jpg')

# converting imInp to grayscale:

imGrayscale = ImageOps.grayscale(imInp)

# converting image to numpy array:

import numpy as np

pixels2D = np.array(imGrayscale)

print(pixels2D.shape)

# showing/saving image

#imGrayscale.show()

imGrayscale.save('problem3Output.jpg')

# helper code ends.

Save your python file as problem1c.py, and running it should save two images with names problem1ci.png and problem1cii.png.

4. Run the script with the name “problem4.py”. Consider the equation: y = mx + c where we want to learn m and c, for the line that fits the given data. Now for each point (xi, yi ) in the data, we can minimize the sum (mx1 + c - y1)2 + (mx2 + c - y2)2  + … + (mxn + c - yn)2  with respect to m and c, by taking the corresponding derivatives.

* Take pen and paper and solve for m and c (you will get interesting equations in terms of mean and standard deviation/variance of xi’s and yi’s, check online what they mean if you don’t know).
* Write the code to find m and c, and add the code to plot the line that fits the given data.

**Bonus Problem:** Modify the code in the previous question to minimize the sum of square of perpendicular distances instead of vertical.