COSE490 Special Topic in Computer Science (Digital Image Processing, Fall 2022)
Instructor: Prof. Won-Ki Jeong
Due date: Oct 2, 2022, 11:59 pm.

Assignment 1: Histogram Processing (100 pts)

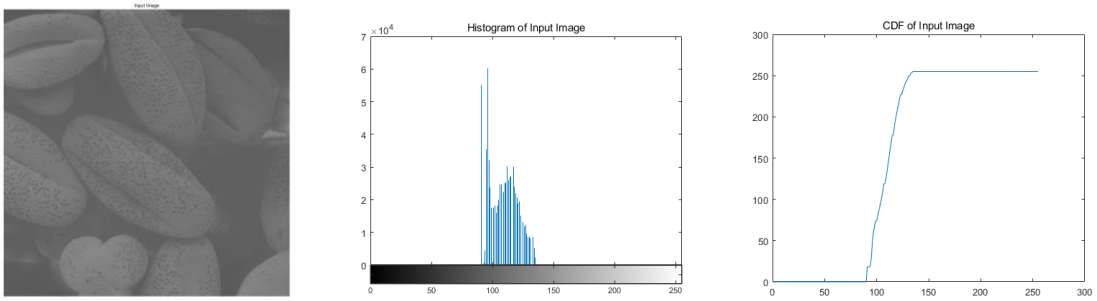
In this assignment, you will implement histogram equalization methods as described below. The provided main_assign_1.m is the main driver code that calls three external functions such as `myCDF.m`, `myHE.m`, and `myAHE.m`. You are not allowed to modify the driver code. In addition, you should implement each function on your own without using MATLAB-provided functions; which means you are not allowed to use the functions such as `cumsum`, `histeq`, `adapthisteq`.

1. myCDF.m : Cumulative distribution function (CDF) (30 pts)

As we discussed in class, the cumulative distribution function is defined as follows:

$$F_X(x) = P(X \leq x) = \int_{-\infty}^{x} p_X(t)\,\mathrm{d}t$$

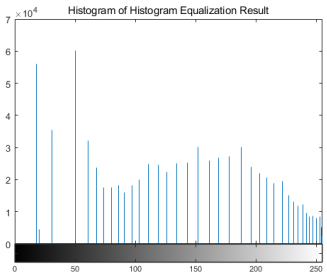
Since we have 8bit-pixel gray scale images, x ranges from 0 to 255. First, you need to compute the probability p(x) for every intensity value x. Then, you can compute the sum of p from 0 to x to compute F(x). The below is the example of histogram (middle) and CDF (right) for the input image (left):

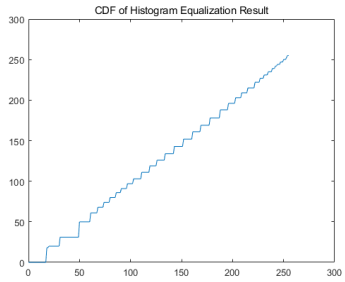


Note that MATLAB already provides `cumsum()` for computing CDF. The goal of this assignment is to implement the function on your own to understand the algorithm better. Therefore, as mentioned earlier, you are not allowed to use `cumsum()`. However, you may use it to debug your code during development, i.e., checking the correctness of `myCDF()`. Make sure MATLAB-provided functions are not in your final submission.

2. myHE().m : Histogram equalization (20 pts)

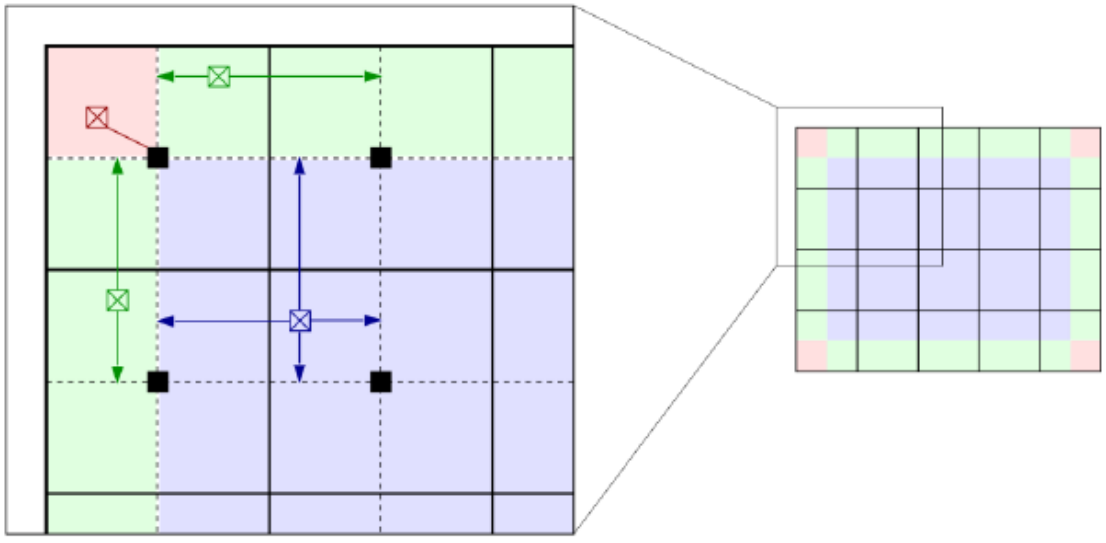
Once you compute the CDF, then you can transform every pixel intensity using CDF. The below is the result after histogram equalization.





3. myAHE.m : Adaptive histogram equalization (50 pts)

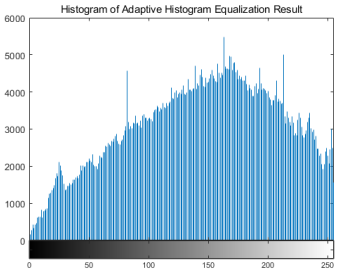
Once you have myCDF and myHE functions ready, then you can implement adaptive histogram equalization easily. What you need to do is split the input image into tiles, and compute intensity mapping function for each tile. Then, for every pixel in the input image, compute four different histogram equalized value and interpolate them based on the distance (use linear interpolation). If the current pixel is near the corner or boundary of the image, then you can use one or two nearby tiles, respectively (refer the figure below). Review the lecture note and follow the algorithm we discussed in the class.

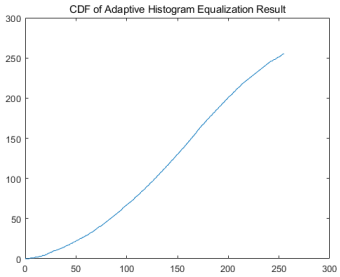


Per-tile histogram computation and interpolation

The below is the result after adaptive histogram equalization using 10*10 splits of the

input image (total 100 tiles).





4. Submission

You need to write a short report explaining the results (test with your own images, try with different tile size for adaptive histogram equalization). Submit the report (pdf) along with `myCDF.m`, `myHE.m`, and `myAHE.m` files via blackboard. Do NOT submit `main_assign_1.m` files.

Good luck and have fun!