

State University of New York at Buffalo

CSE 473/573 Summer 2018 Programming Assignment #1

Date: Wednesday June 06th, 2018; Due: Monday Jun 18, 2018 at 3:00PM

General Instruction: Use Python as the programming language. Submit your code and a report in PDF document showing results. Submission is through UBLearn

Problem (1) (1D and 2D Convolution on Images) 50%

Sobel filter is used in image processing and computer vision, particularly within edge detection algorithms where it creates an image emphasizing edges. It computes gradient in **x** and **y** directions

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} * I \quad G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} * I \quad G = \sqrt{G_x^2 + G_y^2}$$

- (a) Perform 2D convolution on grayscale Image **lena_gray.jpg** with filters specified above to obtain gradient images G_x and G_y . the three images G_x , G_y and G in your Include report. (20%)

Sobel filter kernels are linearly separable

$$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} \begin{bmatrix} -1 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} = \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix} \begin{bmatrix} 1 & 2 & 1 \end{bmatrix}$$

- (b) Perform 1D convolution on grayscale Image **lena_gray.jpg** with 1D-filters specified above to obtain gradient images G_x and G_y . Include these two images in your report. Verify the result after 1D convolutions is same as the one obtained from 2D convolution from (a) (20%)
- (c) Given an $M \times N$ Image and a $P \times Q$ filter, Report the computational complexity of performing 2D convolution vs using separable filters with 1D convolution. (5%)
- (d) Create a separable filter of 100×100 dimension of your choice and apply it on Lena **lena_gray.jpg** using both 2D convolution function and 1D convolutions. Compare the execution time of these two routines using Python time libraries and report the execution times (Note: with just 3×3 filter, the speedup observed will be negligible)

Problem (2) **(Histogram Equalization)** 50%

Histogram equalization is a technique for adjusting image intensities to enhance contrast. Capture or pick an Image of your choice, convert it to **grayscale** and perform histogram equalization using the algorithm below (excerpt from Milan & Sonka):

Algorithm 5.1: Histogram equalization

1. For an $N \times M$ image of G gray-levels (often 256), create an array H of length G initialized with 0 values.
2. Form the image histogram: Scan every pixel and increment the relevant member of H —if pixel p has intensity g_p , perform

$$H[g_p] = H[g_p] + 1 .$$

3. Form the cumulative image histogram H_c :

$$\begin{aligned} H_c[0] &= H[0] , \\ H_c[p] &= H_c[p-1] + H[p] , \quad p = 1, 2, \dots, G-1 . \end{aligned}$$

4. Set

$$T[p] = \text{round} \left(\frac{G-1}{NM} H_c[p] \right) .$$

(This step obviously lends itself to more efficient implementation by constructing a look-up table of the multiples of $(G-1)/NM$, and making comparisons with the values in H_c , which are monotonically increasing.)

5. Rescan the image and write an output image with gray-levels g_q , setting

$$g_q = T[g_p] .$$

Plot the histogram, cumulative histogram of the original image, Transformation function in step 4 and histogram of image obtained after step 5 (4 plots in total). Also show both original image and enhanced image side by side for comparison in your report