Computer Engineering Dept.

Faculty of Engineering

Cairo University Digital Image Processing Lab

# Lab 4

## Pixel Brightness Transformation

**Objective:**

- Know the effect of Negative transformation.
- Know the effect of contrast enhancement.
- Know the effect of gamma correction.
- Understand and implement Histogram Equalization.

## Experiment 1: Negative Transformation [10 minutes]

- Read image ‘Picture1.png’
- Define function ‘***Negative’*** that apply Negative transformation for any image and any threshold. *What is the equation???*
- Call the function and apply it on the image ‘Picture1.png’
- Display the image before and after transformation in the same figure.

## Experiment 2: Contrast Enhancement [10 minutes]

- Read image ‘Picture2.png’
- Define function ‘***Contrast_enhancement’*** that Stretches the grey levels in the range 0 to 100 into the range 50 – 200 and leaves other levels non changed. What is the equation???
- hint: draw the spectrum of levels to know the equation, use for loops and if statements(or matrix conditional indexing which is faster but a little tricky) in the implementation.
- Call the function and apply it on the image ‘Picture2.png’
- Display the image before and after transformation in the same figure.

## Experiment 3: Gamma Correction [10 minutes]

- Read image ‘Picture2.png’
- Define function ‘***Gamma_Correction’*** that Stretches the grey levels according to the gamma equation. **(**$A' = c * A^{\gamma}$ **)** .

- Call the function with **$c$=1 and $\gamma = 3$ and $c$=1 and $\gamma = 0.5$**
- Display the images before and after transformation in the same figure.
- What is the effect of decreasing $\gamma$ **?**

## Experiment 4: Histogram Equalization [40 minutes]

- Write a function to perform histogram equalization to the image .
- Read, display the image and its histogram before and after equalization, and observe the effect on image contrast. Use the image pout, tire.

To perform equalization, code the following steps:

a. For an NxM image with G gray-levels, create an array H with a length G initialized by Zeros.
b. Calculate the histogram of gray-levels for the image.
c. Form the cumulative image histogram, H_c. The cumulative histogram tells you how many pixels have gray-levels less than or equal to the p-th gray level. The calculation is:

   H_c[0] = H[0] then: H_c[p] = H_c[p-1] + H[p]

d. Set the mapping between gray-levels as:

   q = T[p] = round((G – 1) * H_c[p] /(N * M))

e. Go through the images, pixel-by-pixel and write an output image with the gray-levels g_q using the mapping from step 4.

Useful New Functions and Attributes

| Name | Attribute or Function | Usage |
|---|---|---|

| | | |
|---|---|---|
| round | Function | Get round of number |
| np.zeros | Function | To generate matrix with a given shape all elements with zero values |
| range | Function | To generate range of indexes from low range to high range with an optional step |

To make the range of a gray-scale image uint8 (from 0 – 255), use astype(np.uint8).