Phenomena such as inefficient or excessive lighting, motion blur, and sudden scene changes can degrade the quality of recorded digital images. These phenomena often result in obtaining "*noisy*" images. In order to enhance images and make them more suitable for further processings, some preprocessing operations are usually needed. In this lab, you will implement some of these operations in grayscale images.

Important Note: You should complete the lab until the end of the lab hours and submit all your codes to SUCourse as a single zip file. Deadline for in-lab code submission to SUCourse is **15:30**.

Things to do:

Your functions must be as generic as possible, i.e., don't make any assumptions about the size, the type and the colors of the images. Your functions must convert the image to grayscale if it is colored and you must employ the row and column numbers of the images as variables.

Point Image Processing: Point image processing is used to produce a new image by transforming individual pixel values (u). An example of point image processing operations is linear transformation. General linear transformation $g(u)$ is represented by the following equation:

$$g(u) = b(u + a) \tag{1}$$

In the first part of the lab, you will implement special cases of this formula through linear and conditional scaling on the image shown below.

Figure 1: Original image.

- Linear scaling is an example of a linear point transformation in which the pixel values (u) of images are linearly scaled between u_{min} and u_{max} using the function $g(u)$, with parameters $a = -u_{min}$, $b = \frac{G_{max}}{(u_{max} - u_{min})}$ and $G_{max} = 255$.

  Now write a function which takes an image as input and returns the "**linearly scaled version**" of it. u_{min} and u_{max} of the returned image should be 0 and 255, respectively. Your function's name should be "**lab1linscale.m**". Plot histograms of the original and the transformed images using Matlab's built-in "imhist" function.

- Conditional scaling is another example of a linear point transformation in which an image I is mapped into image I_{new} such that I_{new} has the same mean and variance as a reference image I_{ref}. In this case parameters a and b in the $g(u)$, Equation (1), are given as

$$a = \mu_{I_{ref}} \frac{\sigma_I}{\sigma_{I_{ref}}} - \mu_I, \qquad b = \frac{\sigma_{I_{ref}}}{\sigma_I} \tag{2}$$

where $\mu_{I_{ref}}$, μ_I are mean values, and $\sigma_{I_{ref}}$, σ_I are standard deviations of images I_{ref} and I respectively. The mean and the standard deviation of an image can be computed using MATLAB's built-in "mean2" and "std2" functions.

Now write a function which takes "**two images**", current image I and reference image I_{ref}, as inputs and returns I_{new}, the "**conditionally scaled version**" of the current image. Map the current image into the returned image such that the resultant image has the same mean and standard deviation as the reference image. The current image is given in Figure 1 and the reference image is shown in Figure 2. Plot current, reference and output images, and compute mean and standard deviation for each. Your function's name should be "**lab1condscale.m**".

Figure 2: Reference image.

- Histogram equalization is another example of a point processing operation in which the contrast of an image is adjusted using the image's histogram.

Apply histogram equalization to the original image, given in Figure 1, using MATLAB's built-in "**histeq**" function, and obtain histograms of both the original and resulting images using "**imhist**" function.

Spatial Filtering: Local mean filter (Box filter) is an example of spatial filtering methods used to eliminate undesirable random variations in intensity values of an image, called noise. Box filter attenuates noise in the acquired images by convolving it with a sliding window W_P of size $(2k+1)\times(2k+1)$ centered at P. Box filter is realized by replacing each pixel of an image with the average of its neighborhood as follows:

$$\mu_{W_p(I)} = \frac{1}{(2k+1)^2} \sum_{i=-k}^{+k} \sum_{j=-k}^{+k} I(x+i, y+j) \tag{3}$$

$$I_{new}(p) = \mu_{W_p(I)} \tag{4}$$

Now write a function which takes an image I, given in Figure 3, and a number W (for window size) as inputs and returns I_{new}, the "**local mean (box filter) filtered version**". Your function's name should be "lab1locbox.m".

Figure 3: Original image.

Sharpening: The aim of the sharpening operation is to produce an enhanced image I_{new} by increasing the contrast of the given image I along edges, without adding too much noise within homogeneous regions in the image. Sharpening can be implemented as follows

$$I_{new}(p) = I(p) + \lambda\Big[I(p) - S(p)\Big] \tag{5}$$

where S is the smoothed version of the given image I and $\lambda > 0$ is a scaling factor which controls the influence of the correction signal.

Now write a function which takes an image I, given in Figure 4, and a constant λ as inputs and returns I_{new}, the "**sharpened**" version of the image. Use a box filter to smooth the image. Your function name should be "lab1sharpen.m".

Figure 4: Original image.

Post Lab

Post lab reports must include brief explanations of the functions that you implemented in this lab.

- Provide resulting images by utilizing all these functions with different parameters such as window size. Discuss your results.

Deadline for post-lab report submission to SUCourse: **24 October 2022, 23:55**.