INTRODUCTION TO DIGITAL IMAGE PROCESSING

361.1.4751

EXERCISE 1 - BASIC IMAGE OPERATIONS

Submission Date: 17.01.2024

For any question regarding this assignment please refer to the course forum on the moodle web site, for personal questions **only** please email schorya@post.bgu.ac.il

1 Histogram Manipulation

1.1 Reading the Image

1. Read the image named *picasso.jpg* and transform it into a gray scale image of type double using *double(rgb2gray())* function.
2. Display the image using the *imagesc()* function, determine the colormap to grayscale using *colormap()* and add a colorbar.
3. Write your own function named *dip_GN_imread(file_name)* that will return normalized gray scale image. Read the image using *imread()* function, transform it into a gray scale of type double using *double(rgb2gray())* function. Normalize the image between $[0, 1]$ using

$$\frac{Img - min(Img(:))}{max(Img(:)) - min(Img(:))} \tag{1}$$

   We will use this function from section 1.3

1.2 Histogram Construction

Use the above image named *picasso.jpg* with *double(rgb2gray())* for the following sections:

1. Write your own function named *dip_histogram(img,nbins)* that will return the histogram of the image '*img*' using '*nbins*' bins.
2. Display the generated histogram using 256 bins. Compare your result to MATLAB *imhist()* function (use a quantitative measurement). Explain the results.

Note: Here, you can use the *imhist()* function only for checking your answer.

1.3 Brightness

From now on, use the normalized gray scale image version of *picasso.jpg* using the *dip_ GN_ imread(file_ name)* function.

1. Write your own function named *adjust_ brightness(img,action,parameter)* in which '*action*' could get either *'mul'* for multiplication or '*add*' for addition. Adjust the brightness of *'img'* using the *'parameter'*. The output of the function will be the modified image. The output of the function will be the modified image. Make sure the output image stay in the [0, 1] range.

2. Display the original gray scale image together with **one** adjusted image of increased or decreased brightness. Explain the results.

1.4 Contrast

1. Write your own function named *adjust_ contrast(img,range_ low,range_ high)* that will change the contrast of the image '*img*' and in which the *range_ low,range_ high* parameters will determine the new dynamic range of modified image. The output of the function will be the modified image. You should use linear mapping.

2. Calculate the modified image for a new dynamic ranges of $[0.45, 0.9]$ and $[0.4, 0.5]$ and display the images and corresponding histograms. Explain the effect of each new range.

1.5 Quantization

Quantize the original gray scale image using 4bit and 1bit. Explain the results.

1.6 Histogram Equalization

1. Read the image named *dog.jpg* ,transform it into a gray scale image of type double and normalize it between $[0, 1]$ using *dip_ GN_ imread(file_ name)*.

2. Use the MATLAB function *histeq()* to apply the histogram equalization on the image .

3. Display the new image and the corresponding histogram.

4. Why histogram equalization fail in enhance the image?

1.7 Histogram Matching - Optional Section

1. Take an image using your camera/phone/computer, read the image and transform it into a gray scale image of type double and normalize it between $[0, 1]$ using *dip_ GN_ imread(file_ name)* function.

2. Read the image named *city.jpg* and transform it into a gray scale image of type double and normalize it between $[0,1]$ using *dip_ GN_ imread(file_ name)*.
3. Read the image named *face.jpg* ,cast it into a double type and normalize it between $[0,1]$.
4. Display all the three images and their corresponding histograms.
5. Use the MATLAB function *imhistmatch()* to match the histogram of your image to the histogram of *face.jpg* and *city.jpg*.
6. Display the new images and their corresponding histograms.
7. Explain the results. In your explanation, consider the quality of the new images.

2 Spatial Filters and Noise

2.1 Read the Image

Read the image named *dog.jpg* and transform it into a gray scale normalized image in the range $[0,1]$ using *dip_ GN_ imread(file_ name)*. Use this image from now on.

2.2 Mean vs Median Filter

1. Write a function named *mean_filter(img,k)* that will apply a 2-D *k-by-k* mean filter on the image '*img*'. Make sure that the size of the output image is the same as the input image. Find a method to address the boundaries and explain how you implemented it.
2. **Optional Subsection** - Write a function named *median_filter(img,k)* that will apply a 2-D *k-by-k* median filter on the image '*img*'. Make sure that the size of the output image is the same as the input image. Find a method to deal with the boundaries.
3. Filter the *dog.jpg* image using the functions above for k=3, 9, display the results and explain the results (refer to median vs. meanand and the kernel size efect). If you have not implemented subsection 2.2.2, utilize the built-in median filter function

2.3 Gaussian Filter

1. Write a function named *dip_ gaussian_filter(img, k, sigma)* that will apply a 2-D *k-by-k* Gaussian filter on the image '*img*'. The smoothing kernel should be with covariance matrix of $\begin{bmatrix} sigma & 0 \\ 0 & sigma \end{bmatrix}$.
   Hint: use *meshgrid()* function in MATLAB to create the grid and apply the Gaussian formula on the grid to create your kernel.

2. Display the filtered images using *(k, sigma) = (3, 0.2), (3, 1.7), (9, 0.2)* . Briefly explain your results.

3. Subtract the original image from one of the filtered images. Display the new image using *'imshow(out_ img, [])'*. Explain what you see.

2.4 Noise Filtering

1. Create 2 new images by adding 2 different kinds of noises to the original image using *imnoise()* function. The noises are: *'salt & pepper', 'gaussian'*.

2. Apply the implemented filters on each of the noisy images. Use kernel sizes of 3x3 and 9x9 for mean, median and gaussian filters. **Display only the best filter result for each noisy image**.

3. Briefly explain your results. What is the effect of each filter on each noise? Which is the best filter for any given noise? what kernel size work the best? What are the pros and cons of each filter?

3 Bonus Question

Choose a noisy image (you can "trash" a clean image or start with noisy one) as you wish and try to fix it using the histogram manipulation and filters. You are more then welcome to use other ways. Display the initial image together with the modified image in the document. **Be creative! One of the images will be chosen by the course staff and it's authors will receive 0.5 bonus point to the final grade.** The staff will judge by the visual result, originality and the code. Explain!