DIP Course
MiniProject 4

Part 1: Wavelet & Edge Detection

We were trying to apply wavelet transformation to 2 different images. Our purpose was to extract approximation and detail coefficients of these images for 3 levels. However, when we tried to apply inverse wavelet transform and reconstruct the images. We noticed that the detail coefficients of these two images have been mixed up therefore the reconstructed images are not correct. Your job is to restore these two images into their original form.

1. Use monkey.tif and building.tif as the input images and show them.
2. Apply Haar Wavelet transformation on both images for 3 levels.
(Hint: Use “dwt2” Matlab function.)
3. Concatenate wavelet decompositions and show multiresolution image representations.
4. Find the mixed components and solve this problem.
5. Apply inverse discrete 2-D wavelet transform using the Haar wavelet and reconstruct the original images. (Hint: Use “idwt2” Matlab function.)
6. Replace approximation coefficients of original images with zero and reconstruct images using only the remaining wavelet coefficients to detect edges.

Part 2: Enhancement

I. Intensity transformation and Histogram Equalization

Intensity transformation is a technique for mapping an image's intensity values to a new range.
You can do this kind of adjustment with the **imadjust** function.

1. use Fig1.jpg as Input grayscale image and apply the transformation $s = cr^{\gamma}$ with $c = 1$ and $\gamma = 0.6$, 0.4 and 0.3.
2. display the adjusted images and its histogram.
3.Write a program to apply the histogram equalization algorithm on the original image.
As a minimum, your report should include the original image (Fig2.jpg), a plot of its histogram, a plot of the transformation function, the enhanced image, and a plot of its histogram.
(Note that all histograms should be in the range of 0-255)

II. Enhance Images with Spatial Filters (smoothing-sharpening)

Image Filtering is a technique of modifying or enhancing an image. It includes emphasizing certain features and involves techniques like smoothing, sharpening, edge enhancement.

1.implement **imfilter** Matlab function for filter with 3x3 window size by yourself.
2.Use **fspecial** Matlab function and create **'average'**, **'prewitt'** and **'sobel'** filter with 3x3 window size then apply to rice.png and show the result.
3. What happens if we use a filter with a bigger window for example 1/49 of a matrix 7x7 with all elements equal 1. (You can investigate that by **imfilter** Matlab function)

Part 3: Restoration

I. Noise Models & Noise Reduction

1. Use circuit.tif as the input image and show it.
2. Use **imnoise** Matlab function to get three noisy versions of the image:
I. Corrupted with Gaussian white noise with zero mean and variance of 0.06.
II. Corrupted with salt and pepper noise with 0.2 noise density.
III. Corrupted with Gaussian white noise with zero mean and variance of 0.06 then Salt and pepper noise with 0.2 noise density.

3. Use Arithmetic mean filter, Geometric mean filter and Contra-harmonic mean filter with 3× 3 window size to restore the corrupted images. (Mean filters)

4. Use Median filter, Min filter, Max filter and Mid-point filter with 3 × 3 window size to restore the corrupted images. (order-statistic filters)
(Hint: you can use the following Matlab commands: **colfilt**, **medfilt2**, **ordfilt2**, **fspecial**, **imfilter**)

5. Implement Alpha-trimmed mean filter with 3 × 3 window size and d = 2, 4, 6 to restore the corrupted images.

6. Show the restored images and calculate the peak signal-to-noise ratio (PSNR) for each restored image.
7. Compare your results.

II. Image Restoration with Inverse Filter & Wiener Filter

Use the color image "Lena" with the resolution of 256x256. (use "**imresize**" if needed)
1. Simulate a case of a linear motion across 20 pixels at an angle of 30 degree. Use the "**fspecial**" function in Matlab to generate the point spread function (i.e., h) of the motion blur filter. Apply the filter to the image and plot the blurred image.

2. Simulate the effect of the additive noise. Use “**randn**” function in Matlab to generate a noise with the normal distribution with zero mean and variance of 0.25. Add the simulated noise to the blurred image from (1). Plot the image with the additive noise.

3. Follow the Matlab example for "**dconvwnr**" command (in help of this command) to compute the autocorrelation functions of the image and the noise. Apply the Weiner Filter to the blurred image from (b) to get a restored image. Plot the restored image. Estimate the quality of the restored image by computing the SNR (in dB) between the restored image and the original image before blurring and adding noise.

4. If we don’t use the full knowledge about the autocorrelation functions, instead just use the noise-to signal power ratio (NSR) between the noise and the image, what will be the quality of the restored image? Note you don’t have to convert NSR to the dB unit.

Follow the Matlab example to compute the NSR between the noise and the image. Apply the Weiner filter with the NSR value and the blur filter only, but not the autocorrelation functions. Plot the restored image and compute the SNR (in dB) between the restored image and the original image before blurring and noise. Compare them to the results from part (3).