Assignment 1

General Info

- This work can be done anywhere where MATLAB and the related toolboxes are available.
- A **written report** is required. The report is free form but should include results, figures, and any code you wrote, as well as discussions of what you observe.
- Reports should be submitted as an electronic copy in **PDF** format on **Canvas** by the posted due date.
- Late submissions will not be accepted under any circumstance. If you run out of time, submit what you have.
- Tidy documentation and clear discussions of your work improves our ability to fairly mark your report. Make sure your report is well structured, organized, and clear. Your report has to follow the order of questions.
- Include all your relevant code right before the results & discussion of each part and not in an appendix.
- This assignment has 6 questions.

1. 2D Signal Representation, Display, and Manipulation

Here you will learn about the effects of different sampling strategies when geometrical transformation operations are applied to images.

(a) Generate a binary value image of a box (10×10 white box centered in a 30×30 image, with the background set to black). The idea behind this is to have a small number of pixels in the image, so that after performing some geometrical operation, say rotation, the distortions at the edges of the box become clearer.

(b) Rotate the image by 25 and by 45 degrees, respectively. How does the result look like? Do you see any visible artifacts?

(c) Try to improve the rotated image quality by applying the following interpolation methods: 'nearest neighbour', 'bilinear', and 'bicubic'. Display and discuss your results while comparing the different methods you tried. To see the difference, make sure that the matrix type is not logical or integer.

(d) For a grey-level image of your choice, resize the image to twice the original size using the three aforementioned interpolation methods. Discuss and compare your results.

Hint: Use a lower resolution image (e.g., MATLAB's "mri.tif") in order to see the difference between the techniques clearly.

2. Image Convolution

Here you will learn about convolution in the spatial domain and spatial filters.

Given the two spatial filters below:

$$h_1 = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 2 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

$$h_2 = \begin{bmatrix} 0 & +1 & 0 \\ 0 & 0 & 0 \\ 0 & -1 & 0 \end{bmatrix}$$

(a) Convolve the two filters h_1 and h_2 so as to create the filter $h_3 = h_1 * h_2$.

(b) What type of filter is h_3 (lowpass, highpass, etc)? Explain your answer.

(c) Load the image "mri.tif" and convolve it with h3. Display the original and convolved images and briefly describe what you see.

(d) Repeat (a), (b), and (c) but now with the spatial filters h_1 and h_2 given as:

$$h_1 = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 2 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

$$h_2 = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 2 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

3. Noise Reduction and Edge Detection

Here you will learn about some noise reduction and edge detection techniques.

Load the image "eight.tif" and display it, then:

(a) Add shot noise (salt & pepper noise) and display the result. Explain the effect of shot noise on the image with varying noise variance levels?

(b) Repeat step (a) but with white noise (Gaussian) instead. Remember to restore the original image first.

(c) Compare the results of a median filter acting on an image contaminated with shot noise as opposed to the image contaminated with white noise.

(d) Select a binary image and a grey-level image and display them. Determine the histograms of the two images and note how the two histograms look quite different.

(e) By adjusting the histogram of a grey-scale image, it is possible to transform it into a binary image. Apply this on the original "eight.tif" image.

(g) Load the image "eight.tif" again and smooth it with a Gaussian filter to eliminate noise.

(h) Use the *gradient* function to compute the 1st derivatives in the x and y-direction:

[dy,dx]=gradient(I);

(i) Plot the magnitude of the gradient image using *imshow, M=sqrt(dx^2+dy^2).*

(j) Now use the command *quiver(dy,dx)* to view the actual gradient vectors; you may want to zoom in.

(k) Create a binary edge image by thresholding the gradient magnitude image. Choose an appropriate threshold value that gives you few gaps in the edges.

(l) Using the same "eight.tif", for looking at the edges use the following spatial filters: Sobel filter (high-pass filter) and a Laplacian filter.

4. Two-Dimensional Fourier Transform

Here you will learn about the 2D Fourier Transform and the significance of the magnitude and phase spectrums.

(a) Construct a binary image containing a rectangle (e.g. 60×20 pixels) in the middle. The background should be black and the rectangle should be white. Display the binary image. Display and study the Fourier transform of the image and make sure that you understand the overall relation between the image form and the power spectrum.

Tips: Check how the logarithm operation enables a better visualization (display) of the Fourier spectrum. Also, the function '*fftshift*' can be useful for centering the spectrum.

(b) Implement and describe what happens in the Fourier domain when the object, i.e. the rectangle in (a), is rotated, translated, or made "thinner" (i.e. size decreased in one direction)? Display the results in both the spatial and Fourier domain before/after these operations.

(c) The Fourier transform $X(\omega_1, \omega_2)$ of an image $x(n_1,n_2)$ is in general complex valued and the unique representation of an image in the Fourier transform domain thus requires both the phase and the magnitude of the transform. It is, however, often desirable to synthesize or reconstruct a signal from partial Fourier domain information. The Fourier transform phase $\theta_x(\omega_1, \omega_2)$ alone often captures most of the intelligibility of the original signal (image) $x(n_1,n_2)$. The Fourier transform magnitude contributes less to the image information. For the image "lena512.bmp", illustrate the above statement by synthesizing the magnitude-only signal:

$$x_m(n_1,n_2) = F^{-1}[\,|\,X(\omega_1,\omega_2)\,|\,e^{j0}]$$

(d) Similar to (c), now synthesize the phase only signal: $x_p(n_1,n_2) = F^{-1}[1e^{j\,\theta x(\omega 1,\,\omega 2)}]$

Tip: You might need to normalize the results for appropriate display.

5. Filtering in the 2D Frequency Domain

Here you will explore the concept of frequency domain filtering in 2D.

(a) Calculate the Fourier transform of "lena512.bmp" and plot its magnitude spectrum.
(b) Calculate the cutoff frequencies of 3 ideal lowpass filters that remove 0.5%, 5.4%, and 8%, respectively, of the image power.
(c) Apply the lowpass filters you calculated in (b) to the original image (filter in the frequency domain) and reconstruct the filtered images back (using the inverse Fourier transform). Display the resulting images and discuss the results.

Tip: The MATLAB function '*fftshift*' can be useful for centering the spectrum. You might need to normalize or adjust the contrast (e.g. using the log function) of the results for appropriate display.

6. Laplacian Operator in the Frequency Domain

Here you will implement a 2D second order derivative operator (Laplacian) in the frequency domain.

(a) Show that the Laplacian spatial operator can be implemented in the frequency domain using the filter $H(u,v)=-4\pi^2(u^2+v^2)$; *that is,*

$$\Im[\nabla^2 f(x,y)] = -4\pi^2(u^2+v^2)F(u,v)$$

(b) Read the image "blurry_moon.tif" and implement the frequency domain filtering process expressed above.

(c) Generate a filter h = [0 1 0; 1 −4 1; 0 1 0]. Use this filter to process the moon image in the spatial domain and compare your results with (a). Comment on your findings.

Tips: The following MATLAB commands may be useful: *fft2, ifft2, fftshift, imresize, imrotate, imnoise, imagesc, imshow.*

End of assignment 1