Digital Image Processing

Computer Homework 3

Due date: 1404/02/28

Spatial Domain Techniques for Image Processing

1. Instructions

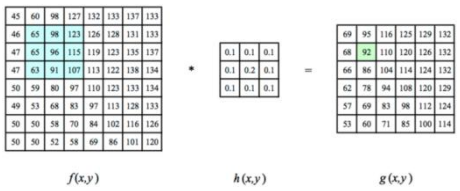
a) Submit the assignment as a **Jupyter Notebook** (.ipynb) containing:
   i. Executed results of the code (no empty cells).
   ii. Explanations in Markdown cells for each step of the assignment.

b) Important Notes:
   i. Attach all necessary files (including sample images) to ensure the notebook runs directly.
   ii. Keep each part of the code in separate cells for clarity.
   iii. Add proper documentation and comments in both Markdown and code cells for a better score.
   iv. Both Python and MATLAB can be submitted, but Python submissions will be awarded 10% more points.

2. Preliminaries

Spatial Filtering

Spatial filtering is a technique in image processing where the value of each pixel in an image is modified based on the values of neighboring pixels. It involves applying a filter kernel (or mask) over the image in a sliding-window fashion. Spatial filters can be used for various tasks such as: Smoothing ,Sharpening ,Edge detection and etc. These filters operate directly in the spatial domain, modifying pixel values based on their surroundings.



[1]

$$g(x,y) = \sum_{i=-k}^{k} \sum_{j=-k}^{k} f(x+i, y+j) \cdot h(i,j)$$

Histogram Manipulation

Histogram manipulation is the process of adjusting the distribution of pixel intensities in an image to improve visual quality or extract hidden information. The histogram of a grayscale image represents how frequently each intensity value (0–255) appears. Two common techniques are:

- **Histogram Stretching**: Expands the range of intensity values so that the darkest becomes closer to 0 and the brightest closer to 255. This increases contrast and can reveal hidden features in low or high intensity ranges.

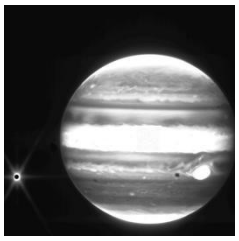
$$Xnew = \frac{Xinput - Xmin}{Xmax - Xmin} \times 255$$

- **Histogram Equalization**: Redistributes the pixel intensity values so that the histogram becomes more uniform. This enhances global contrast, especially in images where background and foreground are hard to distinguish.

3. Tasks & Implementation

Task 1: Histogram Stretching

An image of Jupiter taken by the James Webb Space Telescope has been provided. The supervisor has hidden certain information inside the image, concealing the main password. The objective of this assignment is to apply image processing techniques—such as histogram stretching and enhancement—to extract the hidden information and ultimately recover the password.



➢ Step 1: Load and Display Image

1. Load an image (Jupyter-James-Web.jpg [2])
2. Convert it to grayscale (if not already).

➢ Step 2: Bright Area Enhancement: A QR code is embedded in a very bright region of the image, which makes it difficult to see or scan. To improve its visibility, the histogram of pixel intensities corresponding to the bright area is stretched. This process enhances contrast in that region and reveals the hidden QR code. Histograms before and after stretching are plotted to illustrate the enhancement effect.

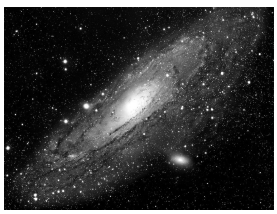
➢ Step 3: A secret key is hidden in a very dark region of the image, making it nearly invisible. To reveal the hidden content, the histogram of pixel intensities within the darker range is stretched. This increases the contrast in shadowed areas and helps make the hidden information more visible. Histograms before and after stretching are plotted to demonstrate the effectiveness of the enhancement.

➢ Step 4: Overall Enhancement: Propose a method to stretch the histogram so that both **QR Code** and other **Secret Pass** in the room can be easily seen. Implement your suggestion, display the resulting image, and plot the histograms before and after the transformations side by

➢ Step 5: QR Code Scanning and Password Retrieval (+10 point)

Now, scan the QR code embedded in the image. This will open a web page where you will be prompted to enter the secret key. If the secret key entered is correct, the password will be displayed. If the key is incorrect, a message will appear stating that the key is wrong. The secret key is: _ _ _ _ _ _.

Task 2: Intensity Transformation



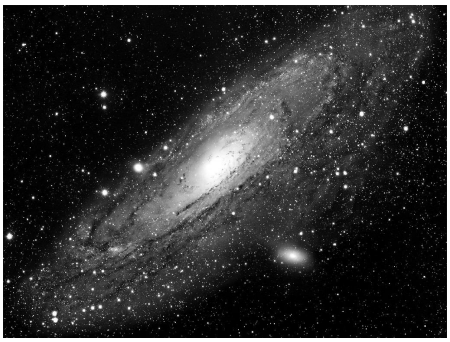
➢ Step 1: Load and Display Image

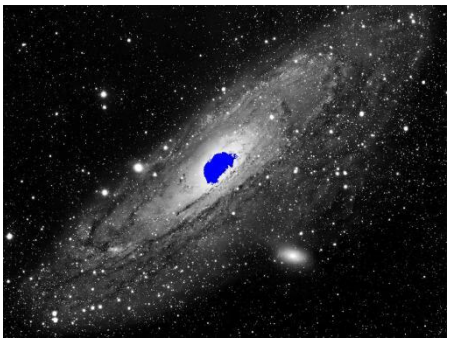
1. Load an image (Andromeda-galaxy.jpg [3])
2. Convert it to grayscale (if not already).

➢ Step 2: **Logarithmic Transformation**: Apply the logarithmic transformation to the image using different coefficients. Choose the coefficient that best visualizes image details. Discuss how this transformation enhances visualization. Plot the histogram of the transformed image.

➢ Step 3: **Power-Law Transformation**: Similarly, perform the power-law transformation with various coefficients to optimize image visualization. Explain how this transformation improves image detail visualization and plot the histogram of the transformed image.

➢ Step 4: **Detection**: Here, the goal is to identify the brightest spot in the image, corresponding to the region where the the core of Andromeda Galaxy is. Simple thresholding may not work due to stars in image. Propose and implement a method to detect this region despite stars. Then, overlay the detected region onto the original image for visualization. (Refer to an acceptable output image shown below for guidance).





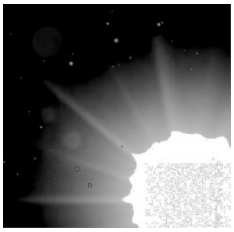
Task 3: Histogram Equalization

During the image capture of the night sky, an unexpected flash of intense light disrupted the scene, causing the moon to become barely visible or completely obscured. In this task, we aim to use adaptive histogram stretching techniques to reveal the hidden moon within the image.

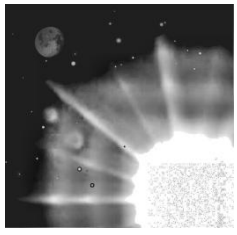
➢ Step 1: Explain the differences between simple histogram equalization and adaptive histogram equalization.

➢ Step 2: Load and Display Image

1. Load an image (moon.jpg)
2. Convert it to grayscale (if not already).



➢ Step 3: **Simple Histogram Equalization**: Apply simple histogram equalization to the image, display the processed image, and plot its histogram.



➢ Step 4: **Adaptive Histogram Equalization**: Apply adaptive histogram equalization to the image, display the processed image, and plot its histogram. Set the size of the sliding window to 7×7.

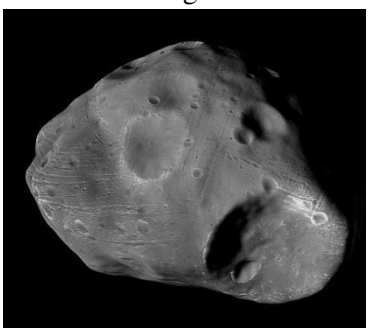
Task 4: Spatial Filtering – Noise Reduction

European Space Agency captures a breathtaking image of Mar's orbiter Phobos but the data is corrupted. Due to transmission interference and sensor limitations, the image is covered with unwanted **Impulse (salt and pepper)** noise and **Gaussian** noise.
Apply spatial filtering techniques to recover the original scene as clearly as possible.

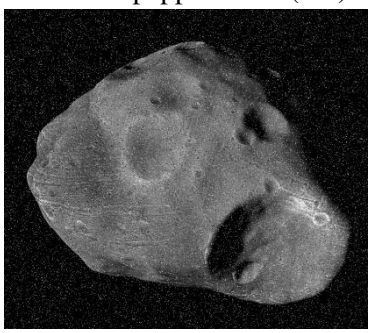
➢ Step 1: Load and Display Images

1. Load an image (Phobos-Orbiter-Mars.jpg, salt_pepper_noise.jpg, gaussian_noise.jpg, both_noises.jpg)
2. Convert it to grayscale (if not already).

Original

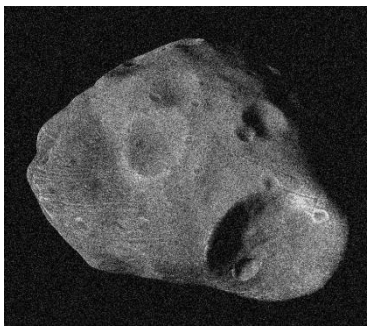


Salt-and-pepper noise (6%)



[4]

Gaussian noise (μ = 0, σ = 50)



Both

➢ Step 2: **Computing SNR**: calculate the error value in each noisy region using the SNR metric. The general formula is:.

$$SNR(x,y) = 10log_{10}\left(\frac{\Sigma_{i,j}\left(x(i,j)\right)^2}{\Sigma_{i,j}\left(x(i,j)-y(i,j)\right)^2}\right)$$

Where x is the noise-free image and y is the noisy or noise-reduced image. Note that you need to calculate the SNR metric separately for each noisy region. (A higher SNR indicates better image quality after filtering compared to the reference.)

➢ Step 3: **Noise Reduction**: Now we want to reduce noise in the image by applying spatial filters. Apply the median filter, Gaussian filter, and averaging filter (with a reasonable size) to the image, then calculate the error value for each noisy region using the SNR metric. Finally, compare the obtained images visually and with the error metric in a table and state which filter performs better for which type of noise.

➢ Step 4: **Deep learning Based Noise Reduction (+10 point)**: Apply a pre-trained deep learning Denoising model (like **DnCNN**) to the noisy images and compare the result with traditional spatial filters.

Task 5: Spatial Filtering – Edge Detection

The James Webb Space Telescope (JWST) is a powerful space telescope designed to observe the universe in infrared light. It has a large mirror and special instruments that allow it to study distant stars, galaxies, and planets. The telescope's sun shield, shaped like a pentagon, keeps its instruments cool by blocking heat from the Sun. JWST is built to explore the early universe and help scientists learn more about how stars and planets form.


[5]

➢ Step 1: Load and Display Images

1. Load an image (James-Webb.jpg)
2. Convert it to grayscale (if not already).

➢ Step 2: **Spatial Filtering**: Apply the following filters to the image and discuss the functionality of each and their differences:

$$[1, -1],\ [1, 0],\ [1, 0, -1],\ \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix},\ \text{and}\ \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

➢ Step 3: **Compare with other method (+10 point)**: Apply the other edge detection methods such as DFT, DWT and Canny and Sobel Edge Detection.

Task 6: Spatial Filtering – Bilateral Filter

In class, we usually begin with basic filters like the mean filter, which works by replacing each pixel with the average of its surrounding pixels in a certain window. This process smooths the image and helps reduce noise, but it treats all neighboring pixels the same—each one contributes equally, regardless of how far it is or how different its brightness is from the center pixel.

To improve on this, we use the bilateral filter, which is a more advanced version of the mean filter. Unlike the mean filter, the bilateral filter gives different weights to neighboring pixels based on two things: how close they are (spatial distance) and how similar they are in intensity (brightness). Pixels that are near and have similar intensities to the center pixel have more influence, while distant or very different pixels contribute less.

This approach helps reduce noise while keeping edges and details intact, something the mean filter can't do well. The idea behind this method was explored in earlier research by Yaroslavsky (1985), Aurich and Weule (1995), Smith and Brady (1997), and Tomasi and Manduchi (1998), who formally defined the bilateral filter, often noted as BF[·].

$$BF[I]_{\mathbf{p}} = \frac{1}{W_{\mathbf{p}}} \sum_{\mathbf{q} \in \mathcal{S}} G_{\sigma_s}(\|\mathbf{p} - \mathbf{q}\|)\, G_{\sigma_r}(|I_{\mathbf{p}} - I_{\mathbf{q}}|)\, I_{\mathbf{q}},$$

where normalization factor $W_{\mathbf{p}}$ ensures pixel weights sum to 1.0:

$$W_{\mathbf{p}} = \sum_{\mathbf{q} \in \mathcal{S}} G_{\sigma_s}(\|\mathbf{p} - \mathbf{q}\|)\, G_{\sigma_r}(|I_{\mathbf{p}} - I_{\mathbf{q}}|).$$

And G is the Gaussian distribution function:

$$G_\sigma(x) = \frac{1}{2\pi\sigma^2}\ \exp\left(-\frac{x^2}{2\sigma^2}\right)$$

[6]

➢ Step 1: Load and Display Images
1. Load an image (Carina-Nebula.jpg)
2. Convert it to grayscale (if not already).

➢ Step 2: **Intuition:** What do the two functions (the blue and the red boxes) in the bilateral filter kernel represent, and how do you tune the coefficients and based on image criteria?

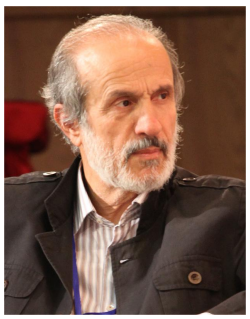
➢ Step 3: **Generating the Noisy Image:** Add Gaussian noise with zero mean and a specified standard deviation to the image.

➢ Step 4: **Simple Mean Filter:** Implement a simple local mean filter to denoise the image, and display the output image.

➢ Step 5: **Bilateral Filter:** Implement a bilateral filter to denoise the image, and display the output image.

➢ Step 6: **Explanation:** Explain the differences between the results obtained from the simple local mean filter and the bilateral filter. Focus on aspects such as noise reduction effectiveness, preservation of image details and edges, and computational complexity.

Dedicated to **Dr. Reza Mansouri** and **Dr. Mehdi Golshani**, for their lifelong commitment to advancing physics, astronomy, and scientific thought in Iran, and to the memory of **Dr. Firouz Naderi** (1946–2023), whose work at NASA inspired generations to look beyond the stars.



Best of Luck

For further information, do not hesitate to contact me at a.najafi@email.kntu.ac.ir.

4. Appendix

a) **Mean Squared Error (MSE)** The average squared difference between the value observed in a statistical study and the values predicted from a model. When comparing observations with predicted values, it is necessary to square the differences as some data values will be greater than the prediction (and so their differences will be positive) and others will be less (and so their differences will be negative). Given that observations are as likely to be greater than the predicted values as they are to be less, the differences would add to zero. Squaring these differences eliminates this situation [8].

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

Mean Error Squared

[9]

b) **Structural Similarity Index (SSIM)** Is a perceptual metric that quantifies image quality degradation caused by processing such as data compression or by losses in data transmission. SSIM actually measures the perceptual difference between two similar images. It cannot judge which of the two is better: that must be inferred from knowing which is the "original" and which has been subjected to additional processing such as data compression [10].

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)}$$

$$Where:$$

$$\mu_x = \frac{1}{N}\sum_{i=1}^{N} x_i, \quad ,\mu_y = \frac{1}{N}\sum_{i=1}^{N} y_i$$

$$\sigma_x^2 = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \mu_x)^2, \quad \sigma_y^2 = \frac{1}{N-1}\sum_{i=1}^{N}(y_i - \mu_y)^2$$

$$\sigma_{xy} = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \mu_x)(y_i - \mu_y)$$

c) **Peak Signal-to-Noise Ratio (PSNR)** is an expression for the ratio between the maximum possible value (power) of a signal and the power of distorting noise that affects the quality of its representation. Because many signals have a very wide dynamic range, (ratio between the largest and smallest possible values of a changeable quantity) the PSNR is usually expressed in terms of the logarithmic decibel scale.

$$\text{PSNR} = 10\log_{10}\left(\frac{L^2}{\text{MSE}}\right)$$

$$L = maximum\ possible\ pixel\ value (e.g.,\ 255\ for\ an\ 8-bit\ image).$$