Digital Image Processing

Computer Homework 4

Due date: 1404/03/11

Machine Vision & Image Processing Lab.

K. N. Toosi University of Technology

Image Restoration 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

Image Degradation Model

The image degradation process is modeled as:

$$g(x,y) = h(x,y) * f(x,y) + \eta(x,y)$$

g(x,y) is the degraded image formed by convolving the original image f(x,y) with the degradation function h(x,y) (the PSF, modeling blur or motion), and adding noise η(x,y); * denotes 2D convolution.
or in frequency domain:

$$G(u,v) = H(u,v) \cdot F(u,v) + N(u,v)$$

G(u,v) is the Fourier transform of the degraded image, F(u,v) is the transform of the original image, H(u,v) is the transform of the degradation function (PSF), and N(u,v) is the transform of the additive noise.

Motion Blur

Motion blur occurs when there is relative motion between the camera and the object during the exposure time. The mathematical model of motion blur:

$$H(u,v) = \frac{T}{\pi(ua+vb)} \sin\left[\pi(ua+vb)\right] e^{-j\pi(ua+vb)}$$

T is the exposure time, a and b are the motion blur parameters representing velocity components in the x and y directions, and u,v are the frequency variables in the Fourier domain.

Inverse Filtering

Inverse filtering attempts to recover the original image by dividing the degraded image's Fourier transform by the degradation function's Fourier transform

$$F(u,v) = \frac{G(u,v)}{H(u,v)}$$

where G(u,v) is the Fourier transform of the degraded image, H(u,v) is the Fourier transform of the degradation function (PSF), and F(u,v) is the Fourier transform of the original image.

Wiener Filtering

Wiener filtering improves upon inverse filtering by adding a term to account for noise and prevent amplification of high-frequency noise:

$$H_w(u,v) = \frac{H^*(u,v)|H(u,v)|^2}{|H(u,v)|^2 + K}$$

where $H^*(u,v)$ is the complex conjugate of the degradation function's Fourier transform, $|H(u,v)|^2$ is the power spectrum of H(u,v), and K is a constant that represents the noise-to-signal power ratio. The filtered image is obtained by multiplying the degraded image's Fourier transform by $H_w(u,v)$.

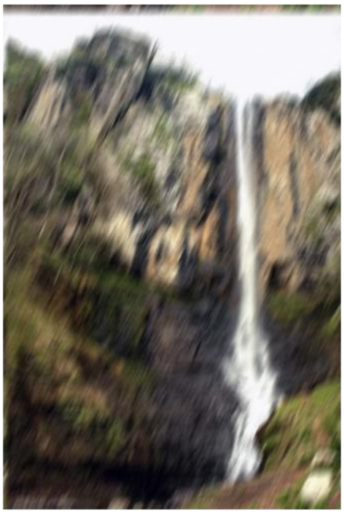
3. Tasks & Implementation

Task 1: Image Obliteration

Simulate linear motion blur of 25 pixels at an angle of 25 degrees in Python. Use a suitable method (like *scipy.signal.convolve2d*) to create the point spread function (PSF) for the motion blur. Apply the motion blur filter to an image, and then display both the original and the blurred images.

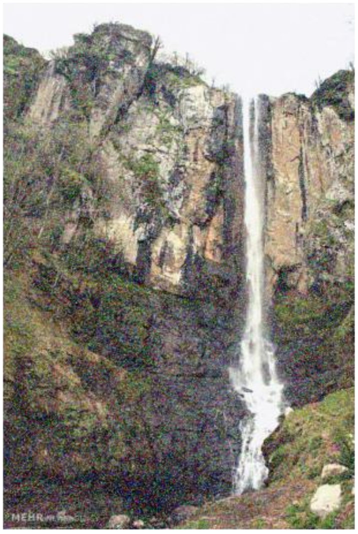
- Step 1: Load Image and apply linear motion blur
  1. Load an image (Iran_Gilan_Laton_Waterfall.jpg [1])
  2. Apply linear motion blur

(a) original image

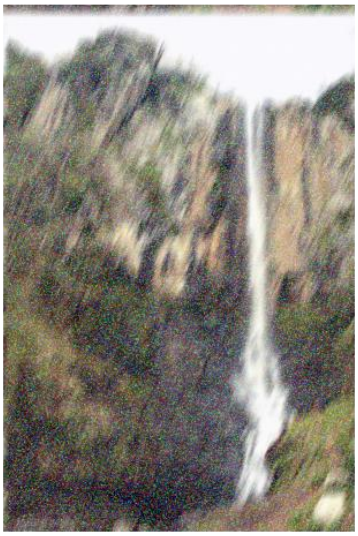


(b) motion blurred image

- Step 2: Simulate the effect of additive Gaussian noise on the previously blurred image. In Python, generate noise with a normal distribution having a mean of 0 and a variance of 0.2. Add this noise to the blurred image obtained in step (I), and display the resulting noisy image.



(c) noisy original image



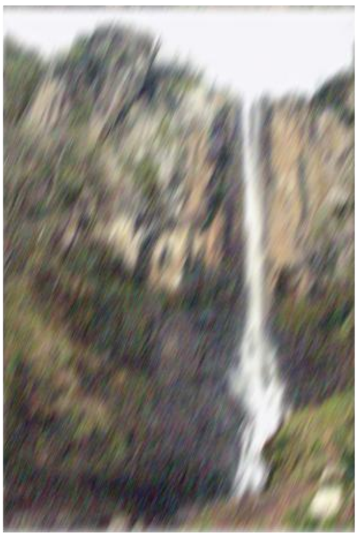
(d)noisy blurred image

Task 2: Image Restoration

➢ Step 1: Use the inverse filtering method to restore image (b) (noiseless blurred image). The result will look like:

(e) Result of applying inverse filtering to the blurred image (b)

➢ Step 2: Simulate the restoration of a motion-blurred image corrupted by additive noise using **Wiener filtering**. First, estimate or compute the autocorrelation of the noisy blurred image (d). Then, apply the Wiener filter to the noisy blurred image (d) to obtain a restored version. Next apply Inverse Filtering to image (d) Finally, display the resulting restored images and **interpret** the results.



(f) result of Wiener filtering

(g) result of inverse filtering

Task 3: Image Verification

➢ Step 1: Use the **Least Squares Error (LSE)** approach to restore the image shown in (d). Display the resulting image after restoration.

➢ Step 2: Estimate the quality of the restored image (e) by calculating the **Signal-to-Noise Ratio (SNR)** in decibels (dB), comparing it to the original image before blur and noise were applied.

➢ Step 3: Similarly, assess the quality of the restored images f and g by computing the SNR (in dB) between each restored image and the clean, original image.

Task 4: Deep Learning-Based Image Restoration (Extra Credit)

Apply a Deep Learning-Based Image Restoration Method to restore the image (d), which has been blurred and corrupted with noise. Evaluate the performance and compare the output with traditional image restoration techniques using Signal-to-Noise Ratio (SNR).

➢ Suggested Pre-trained Models:

DnCNN (Denoising Convolutional Neural Network) A well-known deep neural network designed for removing noise from images. Pre-trained models are available in both PyTorch and TensorFlow.
GitHub: https://github.com/cszn/DnCNN

ESRGAN (Enhanced Super-Resolution GAN) A state-of-the-art model for super-resolution and general image enhancement, including restoration of blurry images.Pre-trained PyTorch models available.
GitHub: https://github.com/xinntao/ESRGAN

➢ Available APIs for Image Restoration:

DeepAI Image Restoration API An accessible API that restores degraded or noisy images using deep learning.
Link: https://deepai.org/machine-learning-model/restore-image

Hugging Face Transformers & Diffusers for Image Restoration A hosts several pre-trained models for tasks like denoising, deblurring, and super-resolution under its transformers, diffusers, and image-processing libraries.
Explore available models and APIs: https://huggingface.co/models?pipeline_tag=image-to-image

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

Where Is the Persian Gulf?

The **Persian Gulf** is a vital body of water located in the southwest of Asia, bordered by Iran to the north and northeast and several Arab countries to the south and west. It serves as one of the most strategically important and resource-rich regions in the world, connecting to the Arabian Sea through the Strait of Hormuz. Historically, this gulf has played a central role in trade, culture, and diplomacy for thousands of years.

It is important to emphasize the historically and geographically accurate name of the *Persian Gulf*. This term has been consistently used for over **two millennia** in historical texts, maps, and official records dating back to the Achaemenid Empire and classical antiquity. Despite some recent political attempts to rename it as the "~~Arabian Gulf~~," such usage is not supported by historical evidence. The *Persian Gulf* remains the internationally recognized name, as affirmed by the United Nations and major global atlases. Preserving the correct terminology respects both historical integrity and scholarly .

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].

$$\mathrm{MSE} = \overset{\text{Mean}}{\boxed{\frac{1}{n}\sum_{i=1}^{n}}}\left(\overset{\text{Error}}{\boxed{Y_i - \hat{Y}_i}}\right)^{\overset{\text{Squared}}{\boxed{2}}}$$

[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.

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

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