

EE 605 Digital Image Processing

Assignment 1 [10 points]

Shanmuganathan Raman

August 23, 2023

1 General Instructions

Please read the following instructions carefully towards your assignment submission.

1. Implement the assignment in python from scratch without using any advanced libraries. You can use numpy, scipy, pandas, matplotlib, opencv or PIL (only for reading, writing, displaying, or transforming images) and other related libraries for preliminary tasks on images.
2. You need to show the result on grayscale images only.
3. Submit only a single Jupyter Notebook or Colab link in the classroom with both code and report (your observations and/or comments in the Markdown).
4. Diligently cite all the sources that you may have referred for completing this assignment. Copying and/or altering the exiting code repositories is strictly **NOT allowed**. Discussion among peers is allowed without exchanging codes. Any sort of malpractice will be have serious repercussions.
5. You need to show the results on the images provided in the 'Dataset.zip' folder.

2 Non Local Means (NLM) Denoising [5 marks]

NLM is a method for image denoising. This method is based on a simple principle: replacing the color of a pixel with an average of the colors of similar pixels. But the most similar pixels to a given pixel have no reason to be close at all. It is therefore licit to scan a vast portion of the image in search of all the pixels that really resemble the pixel one wants to denoise. You will have to implement and demonstrate the qualitative and quantitative performance over all the images provided here. Use Root Mean Square Error (RMSE) and Peak Signal to Noise Ratio (PSNR) for quantitative evaluation (you can use built-in function for these metrics).

2.1 Algorithm Overview

The denoising of a grayscale image (I) and a certain pixel p is performed as follows.

$$\hat{I}(p) = \frac{1}{C(p)} \sum_{q \in B(p,r)} I(q)w(p,q) \quad (1)$$

Here, $C(p) = \sum_{q \in B(p,r)} w(p,q)$ and $B(p,r)$ is the *research window* indicating a neighbourhood centered at p with the size $(2r+1) \times (2r+1)$ pixels.

The weights $w(p,q)$ are computed using the exponential kernel as follows.

$$w(p,q) = e^{-\frac{\max(d^2 - 2\sigma^2, 0)}{h^2}} \quad (2)$$



Figure 1: Example of NLM denoising results. From left to right: original image, noisy image ($\sigma = 26$), denoised image, and difference image.

Here σ denotes the standard deviation of the additive zero-mean Gaussian noise and h is the filtering parameter set depending on the value of σ . The squared Euclidean distance $d^2 = d^2(B(p, f), B(q, f))$ of $(2f + 1) \times (2f + 1)$ *comparison windows* centered respectively at p and q is given by

$$d^2(B(p, f), B(q, f)) = \frac{1}{(2f + 1)^2} \sum_{j \in B(0, f)} (I(p + j) - I(q + j))^2 \quad (3)$$

The weight of the reference pixel p in the average is set to the maximum of the weights in the neighborhood $B(p, r)$. This setting avoids the excessive weighting of the reference point in the average. A sample result on a color image is shown in Figure 1.

2.2 Implementation Instructions

1. For each image in the dataset, convert the RGB image to grayscale and create three noisy versions of the image by adding additive white Gaussian noise with mean 0 and standard deviation $\sigma = \{15, 45, 80\}$.
2. Denoise the image for every $\sigma \in \{15, 45, 80\}$ and obtain the corresponding optimal value of $h = k\sigma$ with $0.01 \leq k \leq 0.65$ and optimal size of *comparison window* $(2f + 1) \in \{3, 5, 7, 9\}$. Report your observations as to what happen when you choose a higher or a lower value of h and $(2f + 1)$.
3. Please note that the *research window* $(2r + 1) \times (2r + 1)$ is limited to a square neighborhood of fixed size because of computation restrictions and is a 21×21 window for small and moderate values of σ and 35×35 for large values of σ due to the necessity of finding more similar pixels to reduce further the noise.
4. Collate your results over grayscale images as shown in Figure 1. You may consider resizing the images (lower dimension no less than 256) if computation takes more time.

3 Histogram Matching [5 Marks]

In the case of images, histograms are the statistical way to represent the pixel intensity distribution. Two images can have similar intensity histograms, but they can differ spatially, i.e., how the intensities are distributed. For this assignment, you need to perform Histogram Matching between two grayscale images, A and B. Histogram matching transforms the source image (say A) histogram distribution to the target image (say B) histogram distribution. In order to match the histogram of images A and B, you need to first equalize the histogram of both images. Then, you need to map each pixel of A to B using the equalized histograms and later modify each pixel of A based on B. Histogram equalization is a method to transform the skewed distribution of image intensities into an approximately uniform distribution over the $[0, L]$, where L is the maximum possible intensity of an image, which in our case is 255 (8-bit, Grayscale).

Note: We have provided an image dataset along with a pre-defined pair in Table 1 over which you have to perform histogram matching. The sample result is shown in Fig. 2. **DO NOT** use any libraries for histogram matching. You can refer to the opencv/PIL/scipy documentation for understanding.

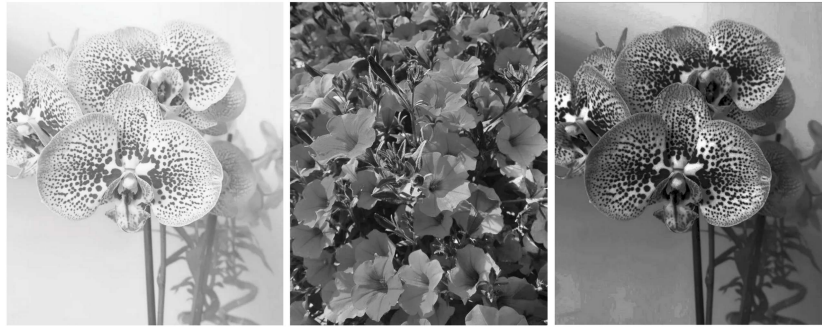


Figure 2: Sample result showing left image histogram matching the center image histogram. The result is shown on the right.



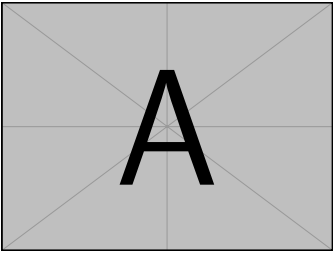


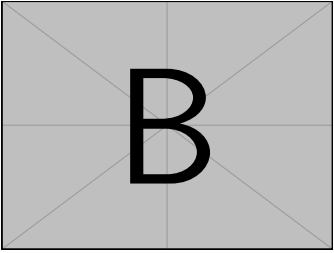


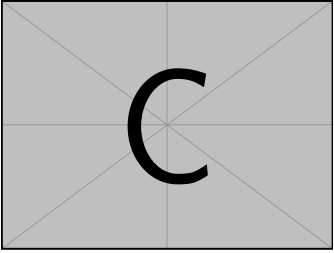


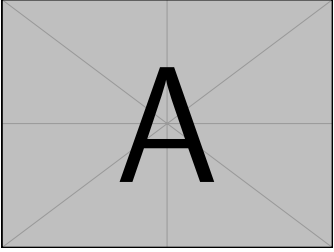
Source Image	Target Image	Matched Histogram
		
		
		
		

Table 1: Pair of images over which you have to show histogram matching. Replace the A, B, C, and A with the matched histogram image after applying your algorithm.