Assignment 2

Mohit Goyal
2015EE10111 IIT Delhi
ELL715 Assignment
Monica Agrawal

January 27, 2019

Note: The code is available at `https://github.com/mohit1997/DIP_Assignment1`

1 Problem 1

Take any 8 bit gray image of 100x100 size of your choice, add Gaussian noise of such that SNR is 0, 10, 20 30 dB respectively. Plot all the images. Apply smoothing operation using 5x5, 7x7 and 9x9 smoothing filters and evaluate the mean square error in all cases.

Solution:

We take a 512x512 image, and add the gaussian noise with variance calculated by the equation below,

$$SNR\ (in\ dB) = 10log(\frac{Power(Image)}{Power(Noise)}) \tag{1}$$

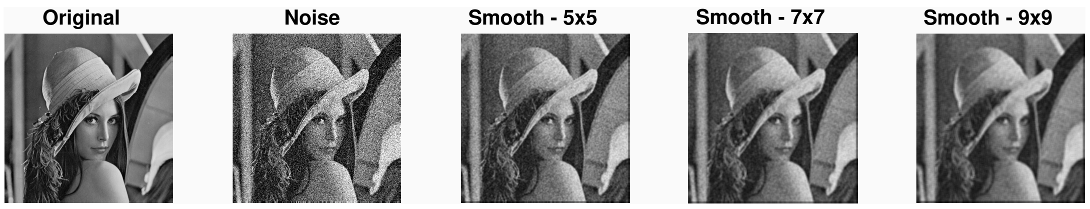

Figure 1: Image with a gaussian noise at SNR = 0dB followed by smoothing filters of various sizes.

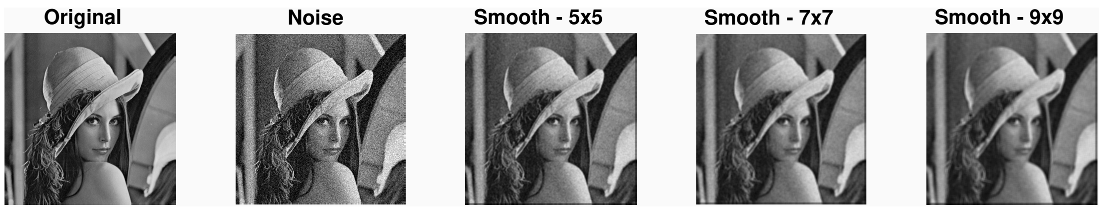

Figure 2: Image with a gaussian noise at SNR = 5dB followed by smoothing filters of various sizes.

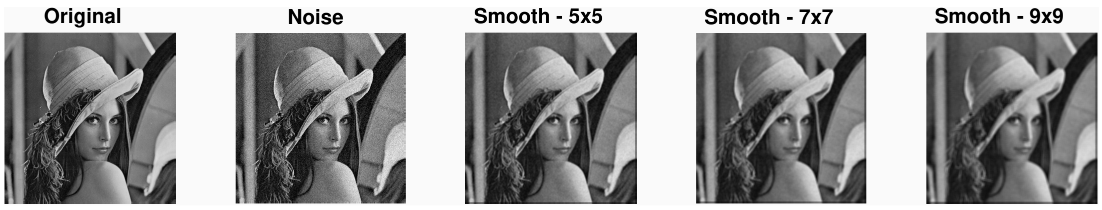

Figure 3: Image with a gaussian noise at SNR = 10dB followed by smoothing filters of various sizes.

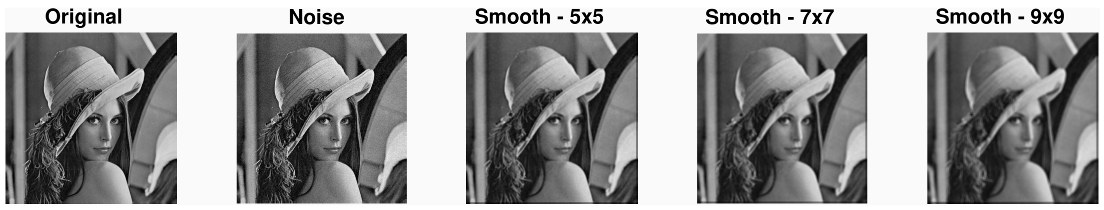

Figure 4: Image with a gaussian noise at SNR = 15dB followed by smoothing filters of various sizes.

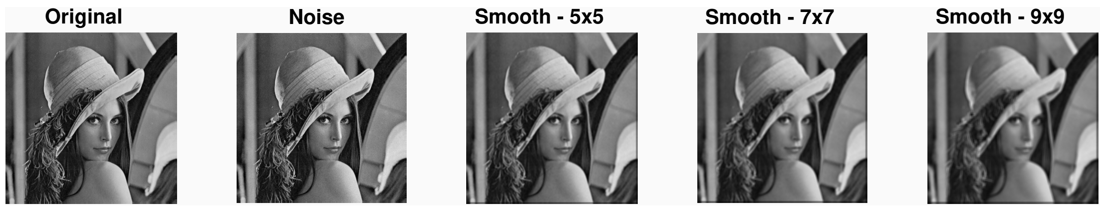

Figure 5: Image with a gaussian noise at SNR = 20dB followed by smoothing filters of various sizes.

The filter, to smooth the image, is taken to be a low pass filter which does an average over a patch of the corresponding size, when convoluted with the image. It helps in removing the noise by making its variance smaller (inversely proportional to size of kernel). One would notice that at higher SNRs, a smoothing kernel will increase the mean squared error.

Below is the variation of mean squared error at various SNRs and smoothing kernel sizes.

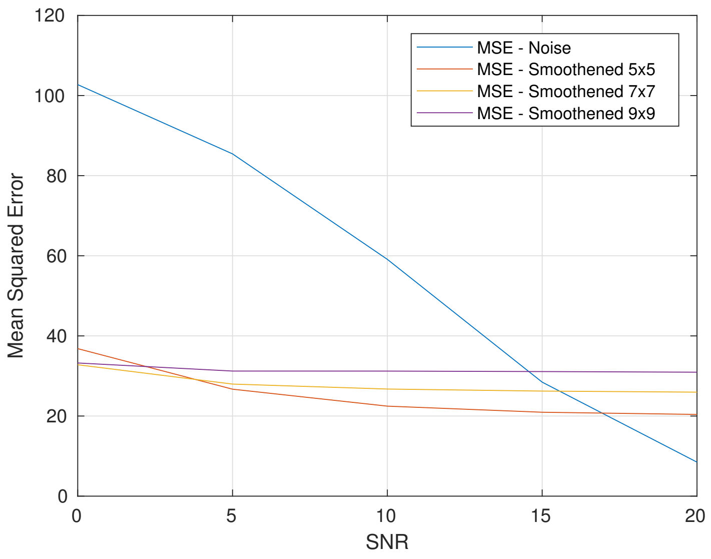

Figure 6: Variation of Mean Squared Error with SNR in dB.

One should note that the mean squared error with the original image increases with the increase in kernel size. This is because SNR is not too low giving rise to this trend. One should note that the performance of 9x9 kernel is better tha 5x5 at 0dB. Similar trend would be seen at lower SNR values.

2 Problem 2

Image denoted as $f(x, y)$, is transformed to image $g(x, y)$. $g(x, y)$ is 3 times larger along y-axis and 2 times larger along x-axis than $f(x, y)$. Also $g(x, y)$ is at 6 units horizontal and 7 units vertical distance from $f(x, y)$. Write a code to do this. Show $f(x, y)$ and $g(x, y)$, Compute $h(x, y)$ third image by rotating pixels of image 2 g(x,y) by 75 degrees counter clockwise. Write a code to do this. Show $f(x, y)$, $g(x, y)$ and $h(x, y)$.

Solution: Doing affine transformations can digitally lead to holes in the image. These holes need to be filled with interpolation methods. We use bilinear interpolation for scaling the image and nearest neighbour interpolation during rotation. *Note: Translations would not require any interpolations techniques.*

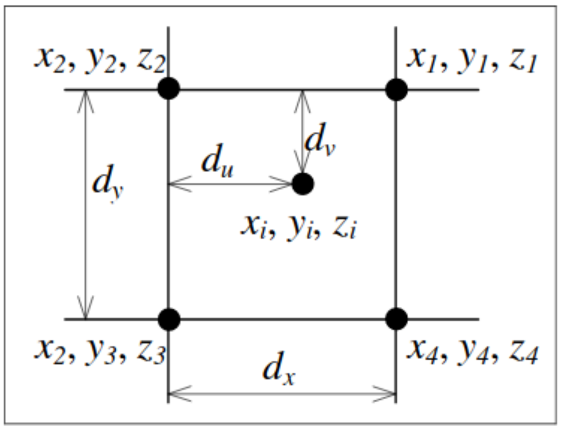

Figure 7: Illustration for bilinear interpolation

Briefly bilinear interpolations can be explained as (Fig.7), consider evaluating the pixel located at x_i, y_i with intensity z_i. We find the nearest pixels denoted by x_j, y_j, z_j, where $j \in \{1, 2, 3, 4\}$. The distances of these coordinates can be referred to from the Figure 7. The weights are then calculated as:

$$u = \frac{x_i - x_1}{x_1 - x_2} \tag{2}$$

$$v = \frac{y_i - y1}{y_1 - y_2} \tag{3}$$

To find the resulting intensity at x_i, y_i,

$$z_i = u.(1 - v).z_1 + (1 - u).(1 - v).z_2 + (1 - u).(v).z_3 + u.v.z_4 \tag{4}$$

The following figure shows the transformations that image goes through,

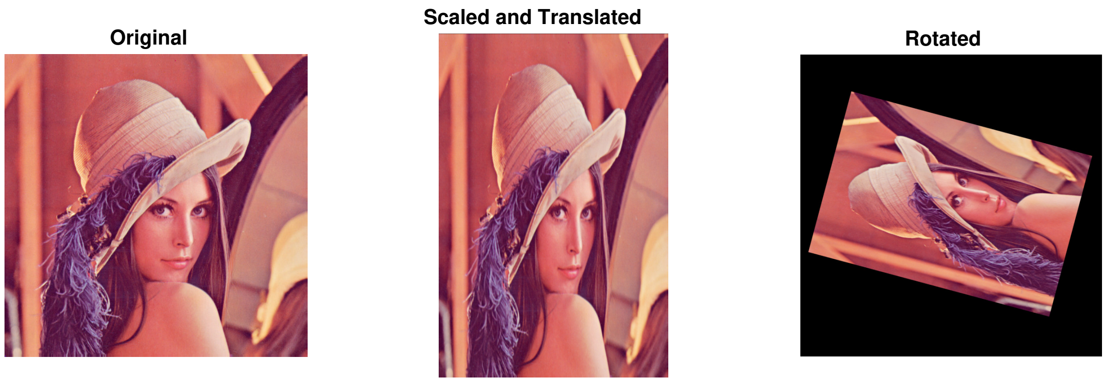

Figure 8: Left: Original Image $f(x, y)$, Middle: Scaled and Translated $g(x, y)$, Right: Rotated anticlockwise $h(x, y)$

3 Problem 3

Take an 8-bit gray scale image and perform the following operations using MATLAB

1. −ve of the image, log and antilog of the image.
2. Apply Gamma correction for gamma=0.4, 2.5, 10, 25 and 100.
3. 2,3,4 power of image.
4. Plot Bit-planes of image(show all the 8-plane images).
5. Plot the histogram of original image and apply Histogram equalization and plot the resulted image.
6. Apply a transformation that highlights range [120,200] but preserves all other levels.

Solutions:

1. **Part 1**:

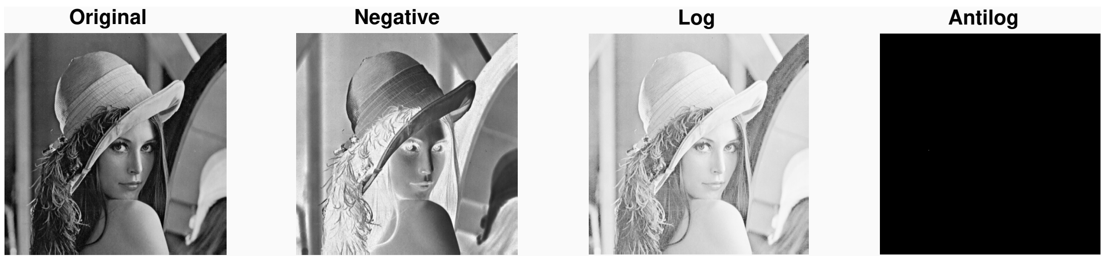

Figure 9: Negative of an Image, its log (normalized) and antilog (normalized).

2. **Part 2**:

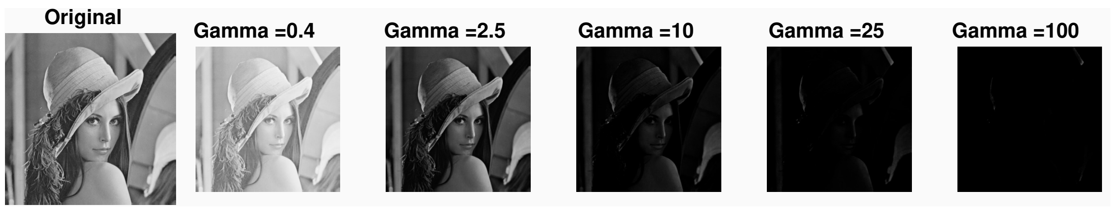

Figure 10: Gamma Correction to change luminosity of an image.

3. **Part 3**:

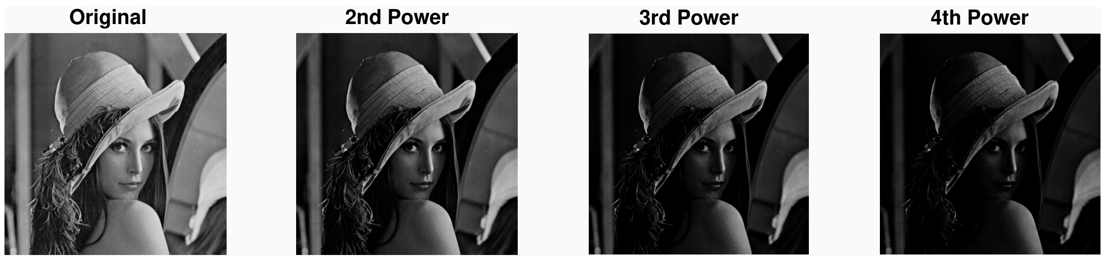

Figure 11: Powers of an Image (2, 3, and 4)

4. **Part 4**:

   It can be inferred from the above images that the least significant planes contribute more to the information content/details of an image. The most significant bit plane looks mostly like a gaussian noise. On the other hand, the last bitplane is mostly alike to the original image.

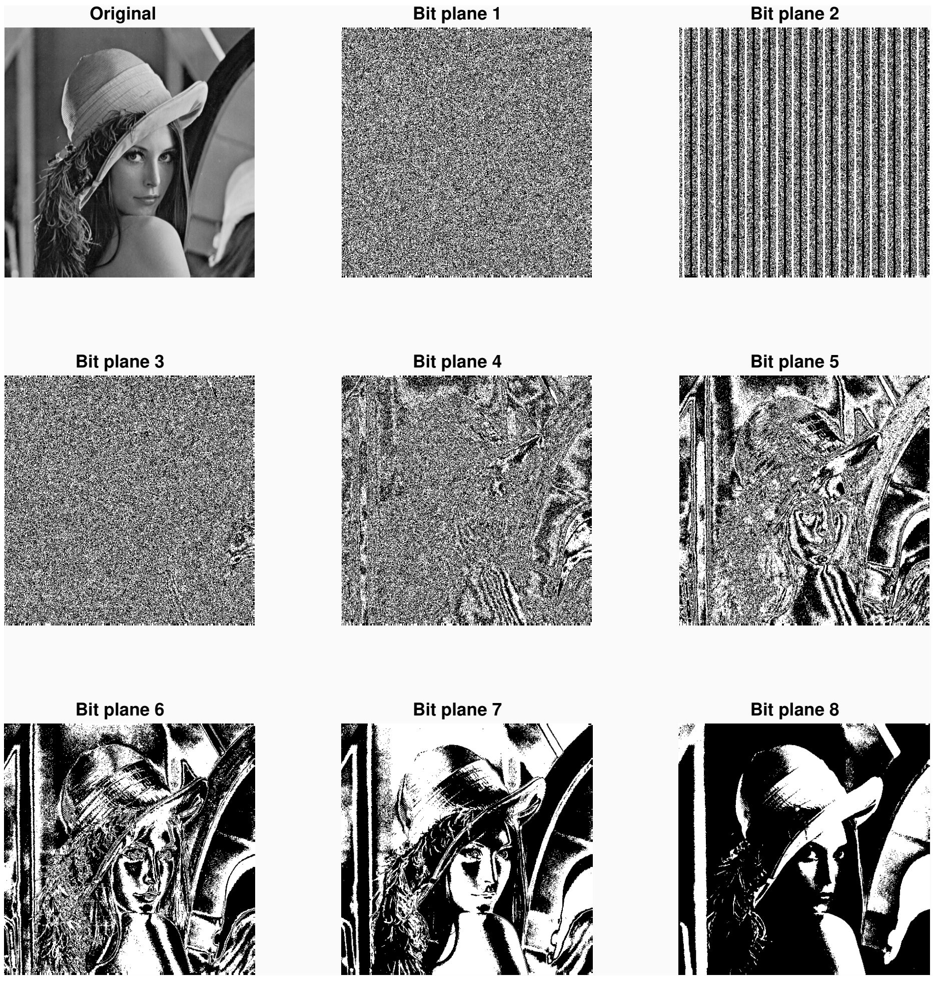

Figure 12: Bit Planes of an 8 bit image

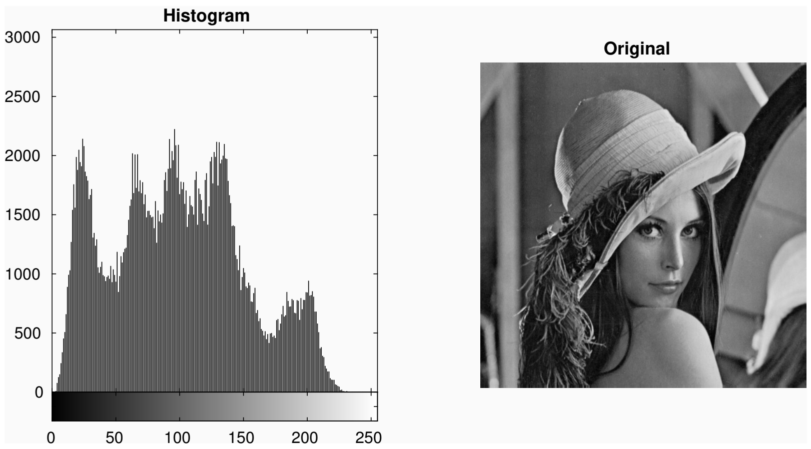

Figure 13: Histogram of the original image and the image itself.

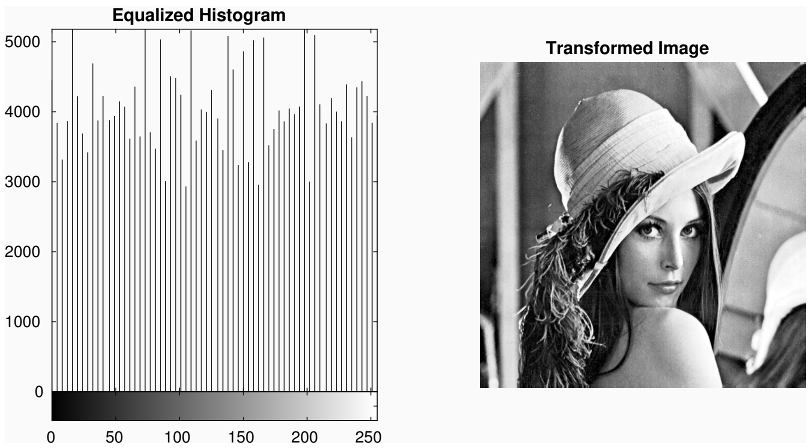

Figure 14: Equalized Histogram of the original image and the transformed image.

5. **Part 5**: We use histogram equalization in the image to improve the contrast.
6. **Part 6:** We adjust the contrast in the range [120, 200], and then keep the other intensities same. To achieve this, we first adjust the contrast in the restricted range, zeroing all other intensities which were outside this range. We then restore those original intensities, by superposing the original image only for those pixels whose intensities were outside this range.

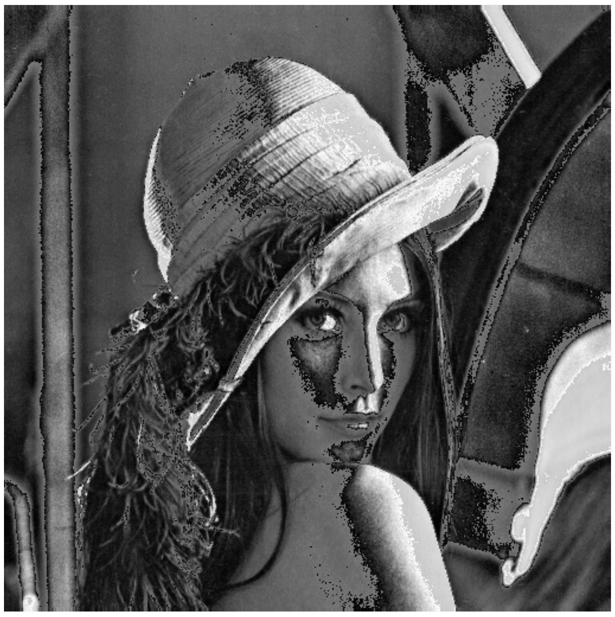

Figure 15: Highlighting the pixels in range [120, 200] and preserving other intensities.