Information Technology University – Punjab

Computer Engineering Department

CE204L – Signals and System

Spring 2021

BSCE-19

Group A Project Report

Aroosa Batool Bsce19009

Faizan Asghar Bsce19021

Mirza Mukkaram Baig Bsce19029

Muhammad Hamza Hayee Bsce19036

Instructor:

Dr. Adnan Siddique

Lab Engineer:

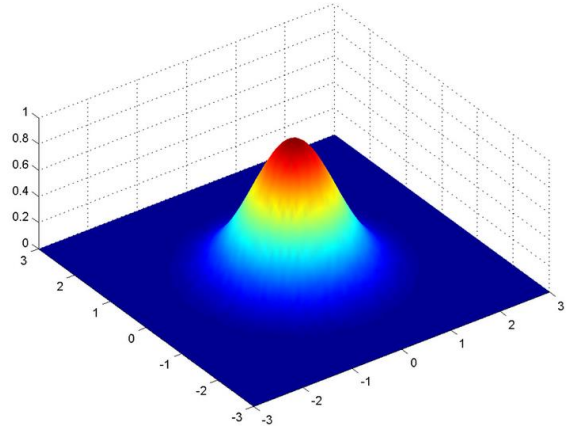
Twahhaa Ahmed

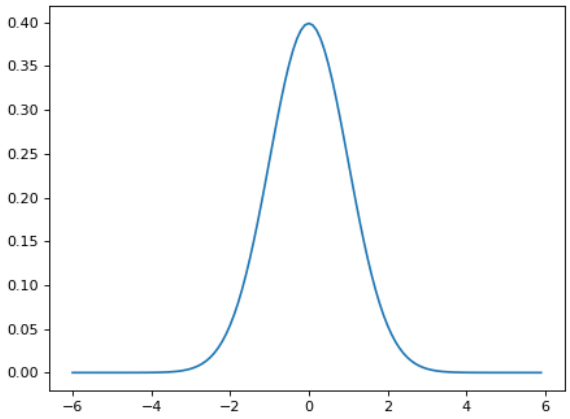
Gaussian Noise and Denoising in Grey scale Imaging

Abstract
Images pick up noise due to acquisition and transmission. This project proposes the idea and algorithm to identify noise and its reduction. A NxN matrix will be decided which will be used as an impulse function of the image. The noisy image will convolute with the matrix to identify the noise by calculating its kernel and comparing it with the decided threshold. The stride will move to until every pixel of the image is visited.

Methodology:
Image is just a matrix of numbers. Each number is representing a unique contrast in color. In this project we figured out the reasons of the imaging catching gaussian noise and how to reduce the noise. To get to the induction of gaussian noise and its reduction first let us give an overview of gaussian noise.

What is Gaussian Noise and How it is induced in pictures?
When current starts flowing in the electrical components the electrons collide with each other. Although these electrons have insignificant mass but due to their enormous velocity, they achieve some amount of momentum. When they collide with other electrons in their flow heat is dissipated. This heat subjects to a little more resistance than ideal situation. This excessive heat results in the changing of the contrast values. This alteration of values is Noise. This noise is modeled as Gaussian Curve.





The Gaussian model suggest that the greatest probability of picking up noise is in the middle of object, and the probability decreases radially. Greater the radius lowers the probability. The Gaussian model is given by the given formula:

$$G(x, y; \partial) = -\frac{1}{2\pi\partial^2} e^{-\frac{x^2+y^2}{2\partial^2}}$$

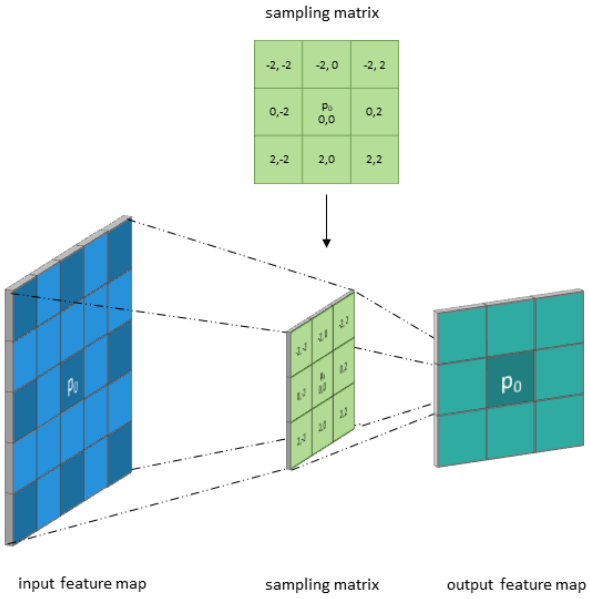
How to reduce the Noise?

This model pdf suggest that the noise is most likely to be picked up in the center of the of sub-window

or image, that is under observation. The sub-window will calculate the convolution of pixel in NxN window and an average will be calculated to normalize the pixel value, if the
value comes out to be greater than the decided threshold value then the middle value is being replaced. Consider the matrix M as the picture under observation. The following figures shows the pictorial representation of movement of the stride (3 shown in red, blue and green).

$$M = \begin{pmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1m} \\ a_{21} & a_{22} & a_{23} & \dots & a_{2m} \\ a_{31} & a_{32} & a_{33} & \dots & a_{3m} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & \dots & a_{nm} \end{pmatrix} \quad M = \begin{pmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1m} \\ a_{21} & a_{22} & a_{23} & \dots & a_{2m} \\ a_{31} & a_{32} & a_{33} & \dots & a_{3m} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & \dots & a_{nm} \end{pmatrix}$$

As seen in the image above there is a problem that arises. When the red stride applies convolution, the value replacement starts from a_{22}. There for the denoising is applied to a fraction of the image. Take a look at the picture below:



Therefor we actually face a problem of information loss. The output image has a lower resolution image than the input image.
To counter the problem Image Padding was applied. Image Padding creates a border of constant value (most likely 0) in the input image. Take a look in the image below.

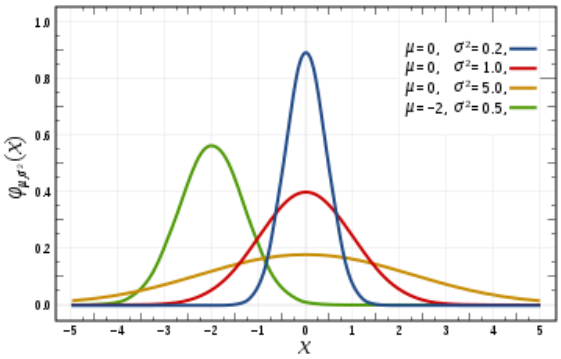
0	0	0	0	0	0
0	35	19	25	6	0
0	13	22	16	53	0
0	4	3	7	10	0
0	9	8	1	3	0
0	0	0	0	0	0

By applying padding the convolution is ensured to be applied on the actual image.

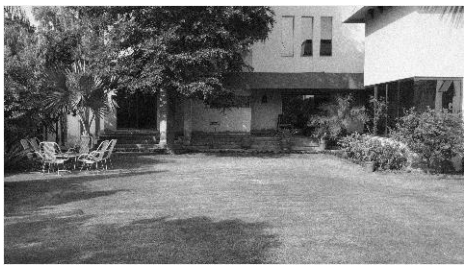
Kernel:
The kernel used in this project is Gaussian Kernel. The exponential part of the equation stated above decides the values of the kernel. The peak value of the kernel is in the middle where as the value decreases radially. The kernel is then divided with the sum of each number of the kernel. This ensures that the area under/summation of the kernel is always unity. This is extremely important, otherwise the Axiom 1 (probability lies between 0 and 1) of probability theory will be violated and the model would not work.

$$\frac{1}{273}$$

1	4	7	4	1
4	16	26	16	4
7	26	41	26	7
4	16	26	16	4
1	4	7	4	1



Kernel Size:
In this project we have used a kernel of 3x3 and a 5x5 kernel. When a 3x3 kernel is applied less noise is being removed compared to the 5x5 kernel but experienced sharper edges in 3x3 kernel as compared to the 5x5 kernel.
Results concluded that choosing a greater sized kernel will resulting in blurring of the output image. Since we are using the gaussian kernel, we need to have a middle/center point to be replaced. This factor prevents us from using any even number matrix.



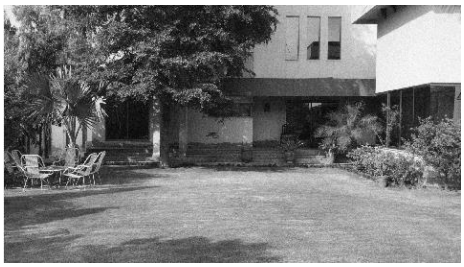
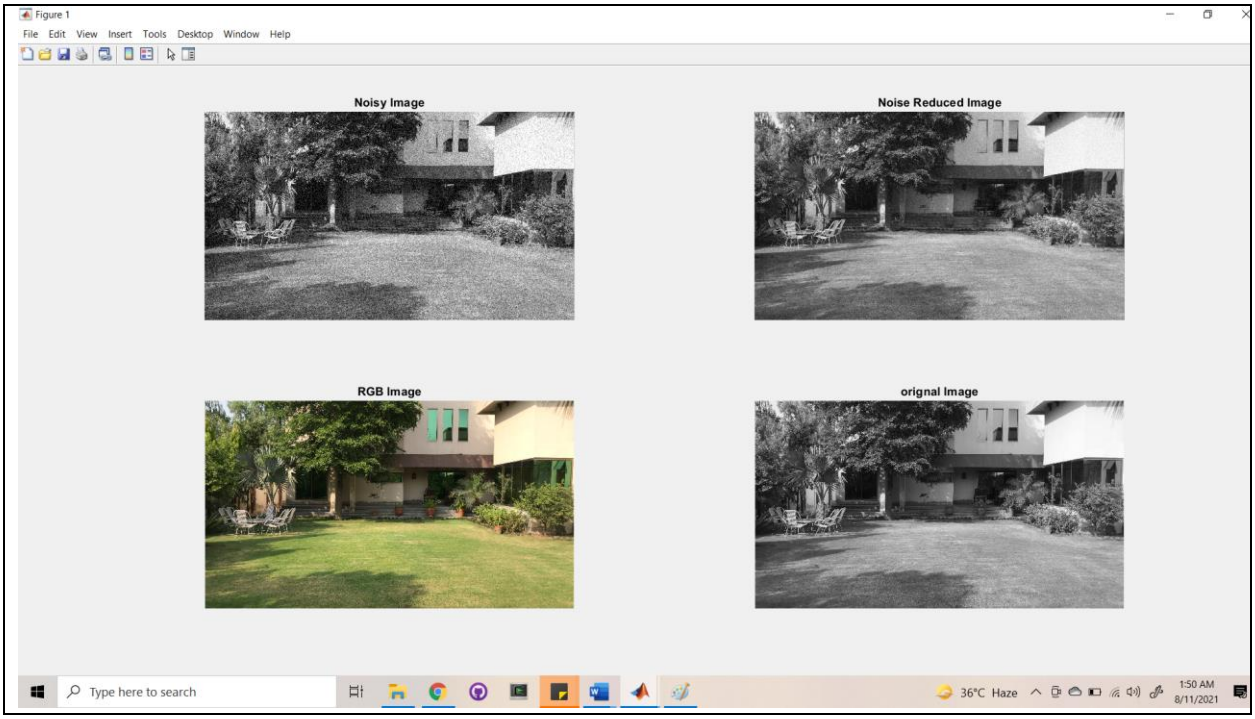


Image Data Conversion:

MATLAB opens the image in unsigned integer 8-bit mode. That means each number of image matrix ranges between 0 to 255. On applying convolution in the very mode, we would have faced problems of loss of information. This is because when multiplication is applied during convolution, we cannot guarantee that after convolution the value that is to be replaced will remain under the range of 0 to 255. If the number exceeds the range than we do not have any shade to represent that number. To solve the problem, we converted the unsigned integer values to floating point data mode. Floating point data mode has a range of 0 to 1. Every single number can be represented between the very range. This ensures that no value goes out of bound after convolution.

Input and Output

The following images are our inputs and results of our project.



MATLAB Code:

```
img = imread("EKWO8531.JPG");%open Image

grayImg = rgb2gray(img); %turn the RGB image to Gray Scale Image
% img2 = rgb2gray(img);
```

```
origGray = grayImg;
imshow(img); %Show Image

grayImg = im2double(grayImg);%Turn the unit8 Gray Image data to floating data Gray Image
% NgrayImg = imnoise (grayImg, 'salt & pepper');
NgrayImg = imnoise (grayImg, 'gaussian'); %Add Gaussian Nosie to the Image
imshow(NgrayImg); %Show Image
%imshow(NgI);
sigma = 1;
kernel = zeros(3,3);
w = 0;
for i = 1:3
    for j = 1:3
        sq_dis = (i-2)^2 + (j-2)^2;
        kernel(i,j) = exp((-1*sq_dis)/ 2*(sigma^2));
        w = w + kernel(i,j);
    end
end
kernel = kernel/w;
[m,n] = size(NgrayImg);
NgI = padarray(NgrayImg, [1 1], 0, 'both'); %imaging pading with 0
for i=1:m
    for j=1:n
        temp = NgI(i:i+2, j:j+2);
        convolution = temp.*kernel;
        output(i,j)=sum(convolution(:));
    end
end
subplot(2,2,1)
imshow(NgI);
title("Noisy Image");
subplot(2,2,2)
imshow(output);
title("Noise Reduced Image");
subplot(2,2,3)
imshow(img);
title("RGB Image");
subplot(2,2,4)
imshow(origGray);
title("orignal Image");
```