Bilkent University

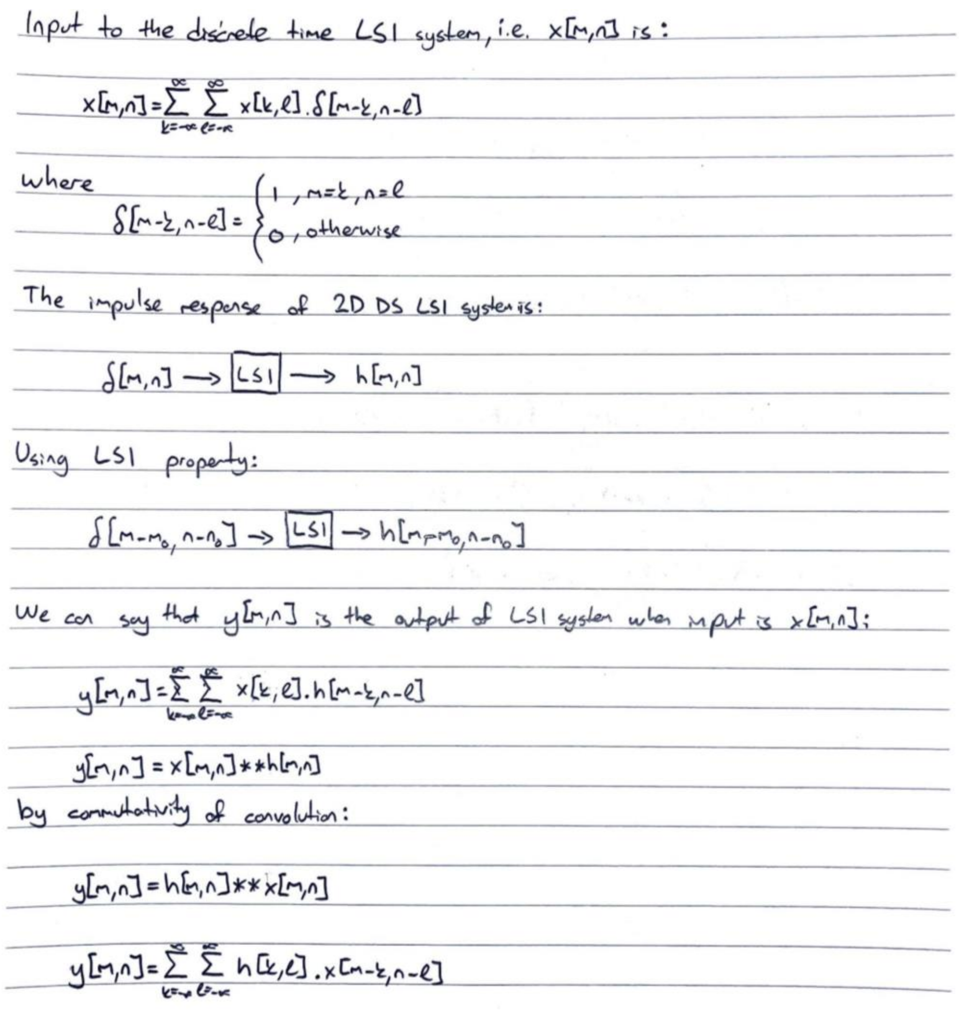
EEE321: Signals and Systems

Lab Assignment 4

Mehmet Emre Uncu - 22003884

Section - 01

Part 2:



Input to the discrete time LSI system, i.e. $x[m,n]$ is:

$$x[m,n]=\sum_{k=-\infty}^{\infty}\sum_{l=-\infty}^{\infty} x[k,l].\delta[m-k,n-l]$$

where

$$\delta[m-k,n-l]=\begin{cases}1, & m=k, n=l\\ 0, & \text{otherwise}\end{cases}$$

The impulse response of 2D DS LSI system is:

$$\delta[m,n]\longrightarrow\boxed{LSI}\longrightarrow h[m,n]$$

Using LSI property:

$$\delta[m-m_0,n-n_0]\rightarrow\boxed{LSI}\rightarrow h[m-m_0,n-n_0]$$

We can say that $y[m,n]$ is the output of LSI system when input is $x[m,n]$:

$$y[m,n]=\sum_{k=-\infty}^{\infty}\sum_{l=-\infty}^{\infty} x[k,l].h[m-k,n-l]$$

$$y[m,n]=x[m,n]**h[m,n]$$

by commutativity of convolution:

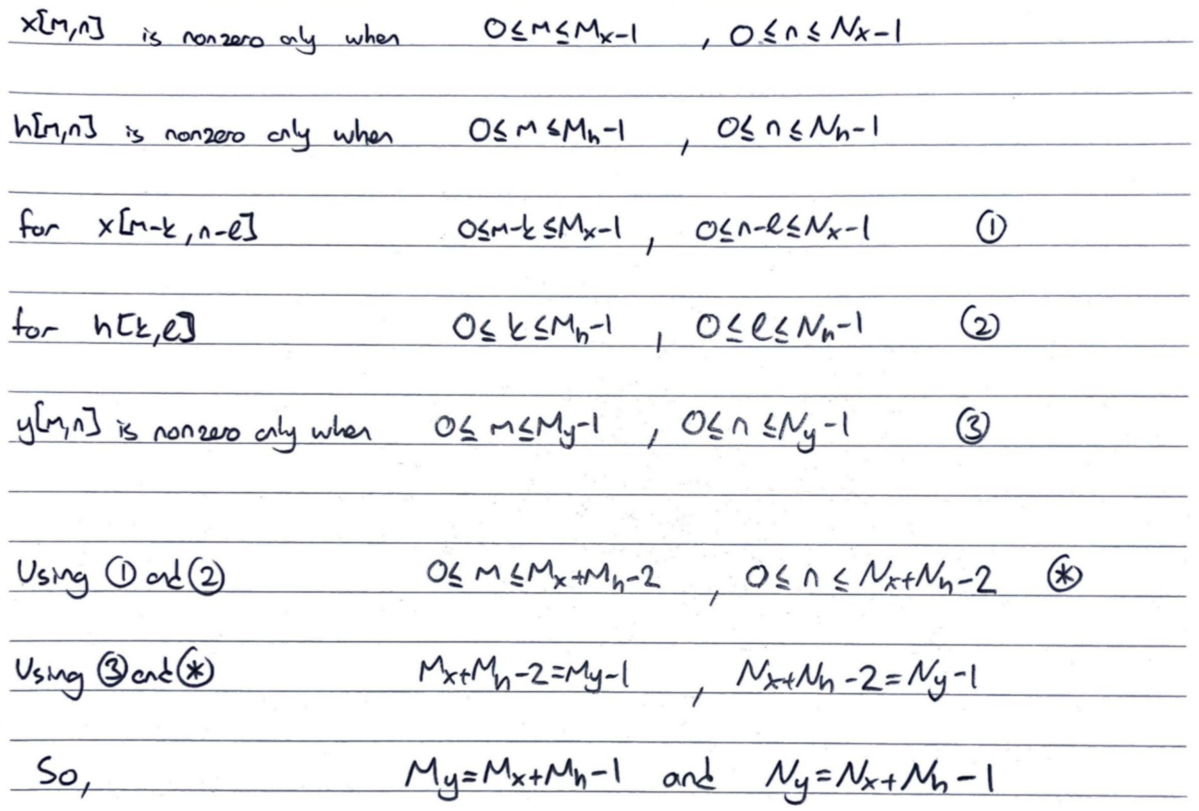
$$y[m,n]=h[m,n]**x[m,n]$$

$$y[m,n]=\sum_{k=-\infty}^{\infty}\sum_{l=-\infty}^{\infty} h[k,l].x[m-k,n-l]$$

Figure 1: Derivation of convolution for a 2D DS LSI system

Part 3:

In this part, it was asked to derive the matrix sizes of y[m, n], M_y, and N_y, in terms of M_x, N_x, and M_h, N_h, which are the matrix sizes for x[m, n] and h[m, n] respectively.



x[m,n] is nonzero only when $0 \leq m \leq M_x-1$, $0 \leq n \leq N_x-1$

h[m,n] is nonzero only when $0 \leq m \leq M_h-1$, $0 \leq n \leq N_h-1$

for x[m-k, n-l] $0 \leq m-k \leq M_x-1$, $0 \leq n-l \leq N_x-1$ ①

for h[k,l] $0 \leq k \leq M_h-1$, $0 \leq l \leq N_h-1$ ②

y[m,n] is nonzero only when $0 \leq m \leq M_y-1$, $0 \leq n \leq N_y-1$ ③

Using ① and ② $0 \leq m \leq M_x+M_h-2$, $0 \leq n \leq N_x+N_h-2$ ⊛

Using ③ and ⊛ $M_x+M_h-2=M_y-1$, $N_x+N_h-2=N_y-1$

So, $M_y=M_x+M_h-1$ and $N_y=N_x+N_h-1$

Figure 2: Derivations of M_y and N_y in terms of M_x, M_h and N_x, N_h

The DSLSI2D(h, x) function is as follows;

```
function [y] = DSLSI2D(h, x)
    [M_h, N_h] = size(h);
    [M_x, N_x] = size(x);
    M_y = M_x + M_h - 1;
    N_y = N_x + N_h - 1;
    y = zeros(M_y, N_y);
    for k = 0:M_h - 1
        for l = 0:N_h - 1
        y(k+1:k+M_x, l+1:l+N_x) = y(k+1:k+M_x, l+1:l+N_x) + h(k+1, l+1) * x;
        end
    end
end
```

The expected result was obtained when the function was tested with the values in the lab manual.

Part 4:

In this part, a picture was denoised with the low-pass filter obtained using the sinc function. The effects on the picture of the operation performed with three different B values selected between 0 and 1 are as follows, where B is a free parameter that determines the filter's bandwidth.

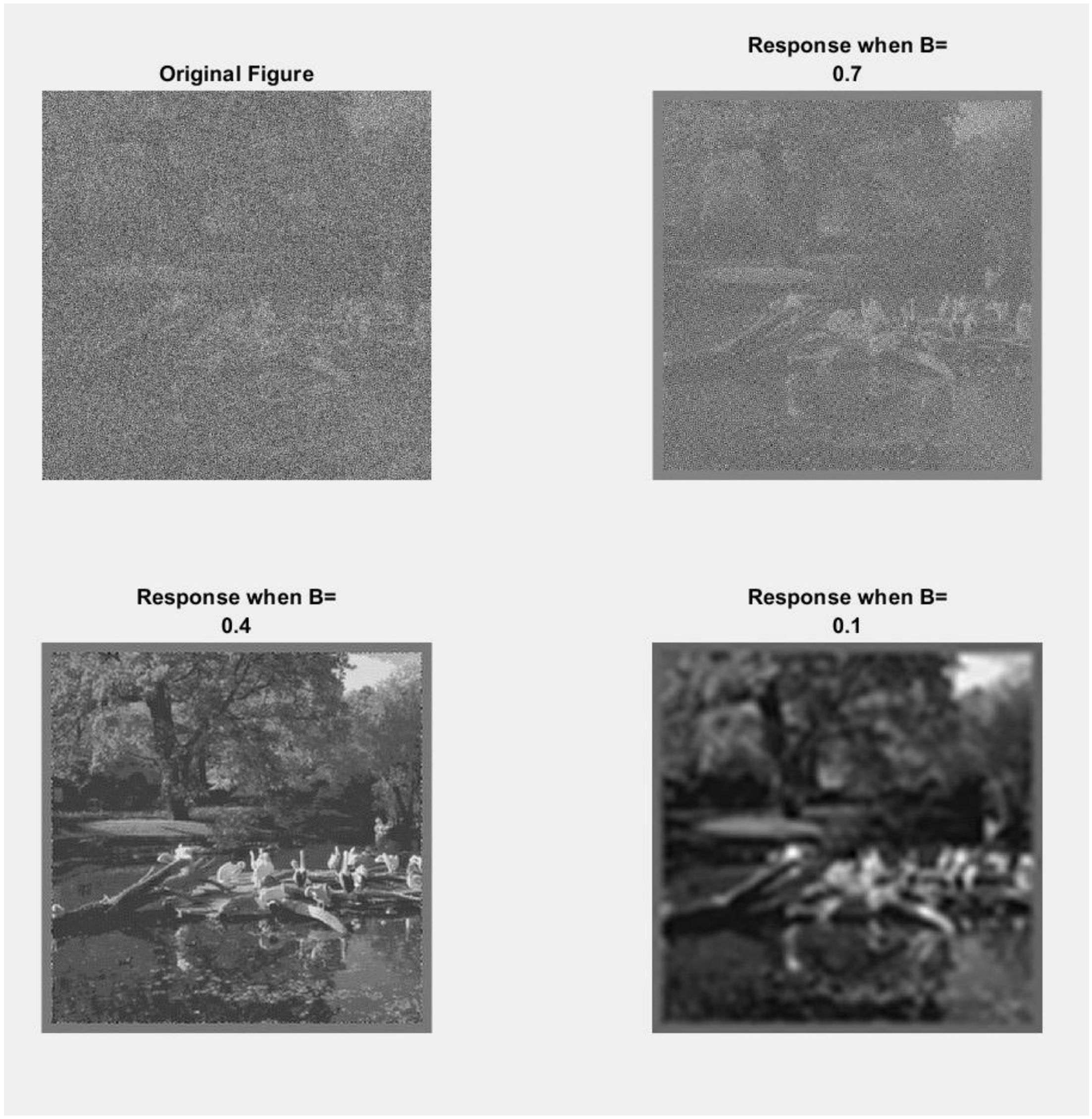


Figure 3: Denoising with different bandwidth values

The least noisy picture was obtained when B = 0.4 when the figures were examined. At the other two B values, that is, when the bandwidth is very wide and very narrow, the picture is seen to be blurrier. Additionally, when different b values were tried out of curiosity, the clearest image was obtained when B = 0.3. Both high-frequency noise is eliminated, and low-frequency data is preserved at this value.

Part 5:

In this part, it was asked to make edge detection in three different ways. The results obtained are shown in the figure below with the original image.

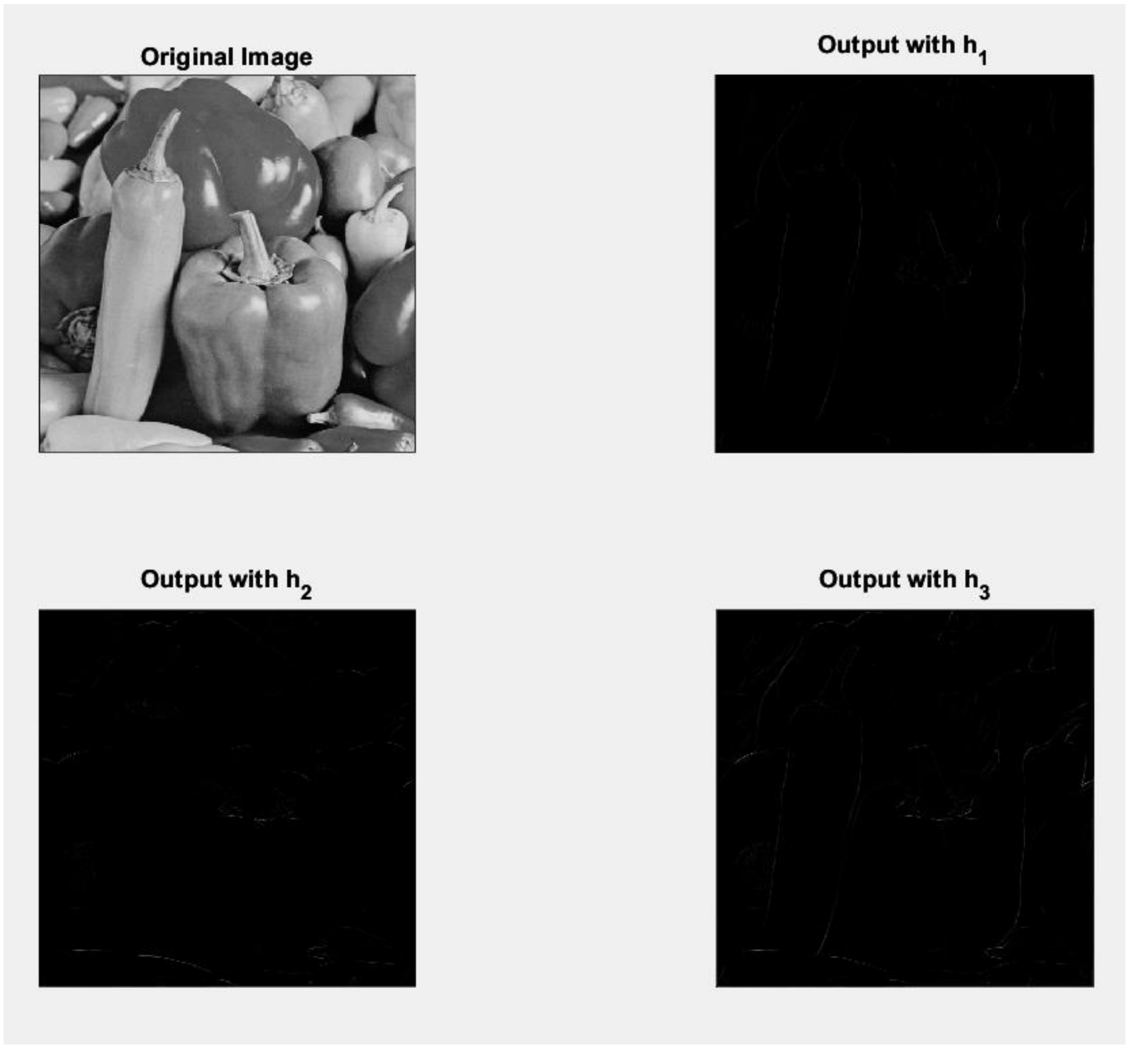


Figure 4: Comparison of three different edge detections with the original image

When the image at the top right is examined, vertical lines are emphasized; that is, vertical edge detection is made. In the picture below left, this time, the horizontal lines are emphasized; that is, horizontal edge detection is made. Finally, both vertical and horizontal edges were detected in the image below right, and an image more similar to the original was obtained.

Part 6:

In this part, a certain object in an image was detected. An attempt was made to detect the face of football player number 23 in the picture of the Turkish national football team. To do this, the matching filter method was used, that is, a 2D DS LSI system whose impulse response is an inverted face of football player number 23. The inverted face was passed through this system as input, and when the absolute value of the output was taken, the picture at the top right in the figure below was obtained. In this picture, which is created by combining values between 1 and 0, 1 being white and zero being black, white dots can be seen on the faces of the football players and in some different places. From this, it can be deduced that the white areas are places similar to the input, that is, the face of football player number 23.

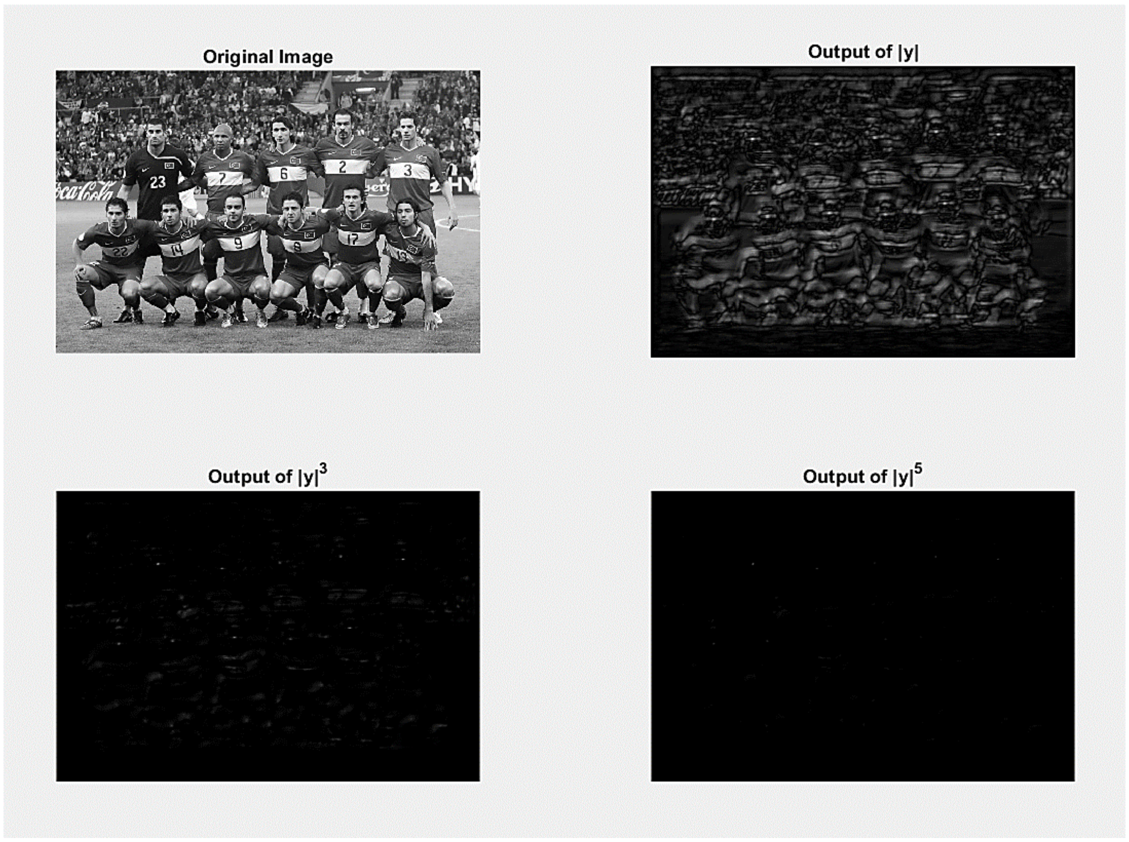


Figure 5: Outputs and Original Image for Part 6

Afterward, the absolute value of the output was taken to the third and fifth powers to make the white dots more obvious. Since the values in the picture are between 1 and 0, the difference between the small and large values will increase when taken to the third power. Thus, the white spots on the football players' faces became more visible. The difference became very large when the output was taken to the fifth power. The white dot on the face of football player number 23 became easily visible, so the face was detected without any confusion in the image.

MATLAB Code:

```
%% PART 3

x = [1 0 2; -1 3 1; -2 4 0];
h = [1 -1; 0 2];
y = DSLSI2D(h, x);


%% PART 4 IMAGE DENOISING

x   = ReadMyImage('Part4.bmp');
M_h = 30 + 0;
N_h = 30 + 0;
subplot(2, 2, 1);
DisplayMyImage(x);
title("Original Figure");
for i = 1:3
    B = 1 - i * 0.3;
    for m = 0:M_h - 1
        for n = 0:N_h - 1
            h(m + 1, n + 1) = sinc(B * (m - (M_h - 1) / 2)) * sinc(B * (n - (N_h - 1) /
2));
        end
    end
    y = DSLSI2D(x, h);
    subplot(2, 2, i + 1);
    DisplayMyImage(y);
    title(["Response when B=" B]);
end


%% PART 5

x   = ReadMyImage('Part5.bmp');
tiledlayout(2, 2);
nexttile;
DisplayMyImage(x)
title("Original Image");

h_1 = [0.5 -0.5];
y_1 = DSLSI2D(h_1, x);
s_1 = y_1 .^ 2;
nexttile;
DisplayMyImage(s_1)
title("Output with h_1");

h_2 = [0.5; -0.5];
y_2 = DSLSI2D(h_2, x);
s_2 = y_2 .^ 2;
nexttile;
```

```
DisplayMyImage(s_2)
title("Output with h_2");

h_3 = 0.5 * h_1 + 0.5 * h_2;
y_3 = DSLSI2D(h_3, x);
s_3 = y_3 .^ 2;
nexttile;
DisplayMyImage(s_3)
title("Output with h_3");


%% PART 6

x = ReadMyImage("Part6x.bmp");
h = ReadMyImage("Part6h.bmp");
y = DSLSI2D(h, x);

tiledlayout(2, 2);
nexttile;
DisplayMyImage(x)
title("Original Image");

nexttile;
DisplayMyImage(abs(y));
title("Output of |y|");

nexttile;
DisplayMyImage(abs(y).^3);
title("Output of |y|^3");

nexttile;
DisplayMyImage(abs(y).^5);
title("Output of |y|^5");


%% FUNCTIONS

function [y] = DSLSI2D(h, x)
    [M_h, N_h] = size(h);
    [M_x, N_x] = size(x);
    M_y = M_x + M_h - 1;
    N_y = N_x + N_h - 1;
    y = zeros(M_y, N_y);
    for k = 0:M_h - 1
        for l = 0:N_h - 1
        y(k + 1:k + M_x, l + 1:l + N_x) = y(k + 1:k + M_x, l + 1:l + N_x) + h(k + 1, l
+ 1) * x;
        end
    end
end

function [x] = ReadMyImage(string)
    x = double((rgb2gray(imread(string))));
    x = x - min(min(x));
    x = x / max(max(x));
    x = x - 0.5;
end

function []= DisplayMyImage(Image)
    Image = Image - min(min(Image));
    figure(1);
    imshow(uint8(255 * Image / max(max(abs(Image)))));
end
```