**School of Engineering and Applied Science (SEAS)**
**Ahmedabad University**

## BTech(ICT) Digital Signal Processing (Section 1)

## Laboratory Assignment-2

**Enrollment No: AU1841145** **Name: Samarth Shah**

AIM : Practice of concepts like Convolution, cross-correlation and auto-correlation.

1. Solution Problem-1

   (a) Approach: In each sub-questions, 2 sequence are given. From those input sequence, final output range can be calculated from their minimum and maximum ranges. Thus, the axis will be set and with own function convolution for each sub-questions will be done.

   (b) Matlab Script:

```
% Name : Samarth Shah
% Roll No: AU1841145
% Lab2 (Question_1) we have to and linear convolution using our own function as well as inbuilt function
close all ;
clear all ;
clc ;
series1_input_x =  [1,2,2,1] ; % first signal input
series2_input_h = [1,-1,2]; % second signal input
series1_range_x = [-1:2]; % first signal range
series2_range_h = [-2:0]; % first signal range
% Calculating the length of the ranges
series1_length = length ( series1_range_x ); % calculating the length of range
series2_length = length ( series2_range_h ); % calculating the length of range
% making the length same for both series
final_length1 =[ series1_input_x , zeros(1,series2_length) ]; % making length same
final_length2 =[ series2_input_h , zeros(1,series1_length) ]; % making length same
max_series2 = max ( series2_range_h ); % finding max of index2
min_series2 = min ( series2_range_h ); % finding min of index2
max_series1 = max ( series1_range_x ); % finding max of index1
min_series1 = min ( series1_range_x ); % finding min of index1
%getting output from the user-defined convolution function
output_convolution = linear_convolution(final_length1 ,final_length2 , series1_length , series2_length );
% getting output from the inbuilt convolution function
out_system_function = conv( series1_input_x , series2_input_h ); % inbuilt function
start_point = min_series2 + min_series1 ; % starting index
end_point = max_series2 + max_series1 ; % ending index
axis_range =[ start_point : end_point ];
subplot (211)
stem (axis_range ,output_convolution,'rs-') ; % Discrete signal
% Plotting Graphs
ylabel ('Signal');
xlabel ('Range');
title ('Linear Convolution using user-defined Function') ;
subplot (212)
stem (axis_range , out_system_function,'bs-' ); % plotting discrete graph
ylabel ('Signal');
xlabel ('Range');
% graph title
title ('Convolution using inbuilt function')
```

(c) Function:

```
function func_out = linear_convolution(series1_input_x ,series2_input_h ,
    series1_range_x , series2_range_h )
    for i = 1:series1_range_x+series2_range_h-1
        output(i) =0;
        for j = 1:series1_range_x
            if(i-j+1>0) % final range should be more than 0
                output(i)=output(i)+series1_input_x(j).*series2_input_h(i-j+1);
                %Convolution function
            end
        end
    end
    func_out = output; % output of function
end
```

(d) Simulation Output:

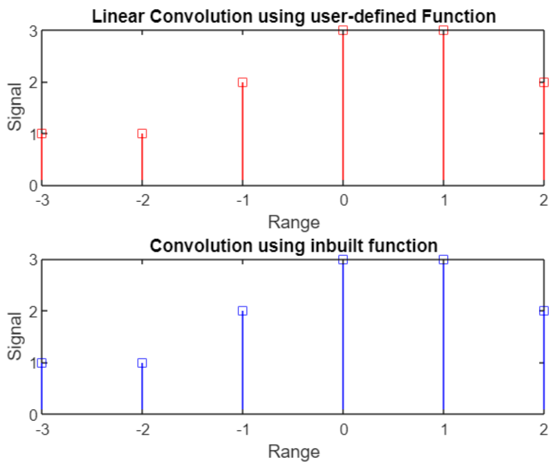



**Figure: 1st Sequence**

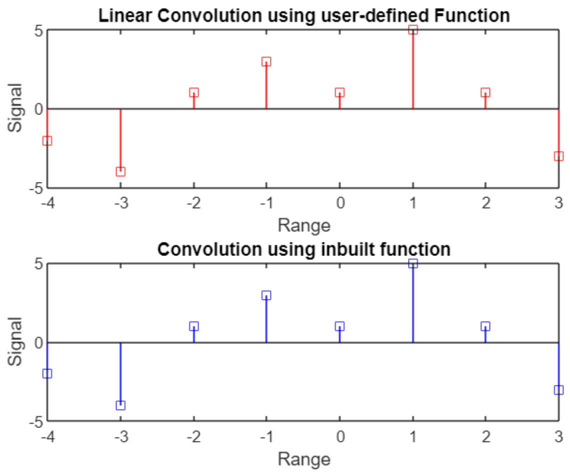



**Figure: 2nd Sequence**

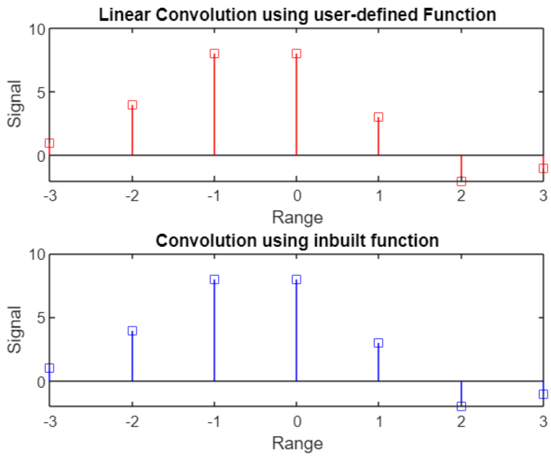



**Figure: 3rd Sequence**

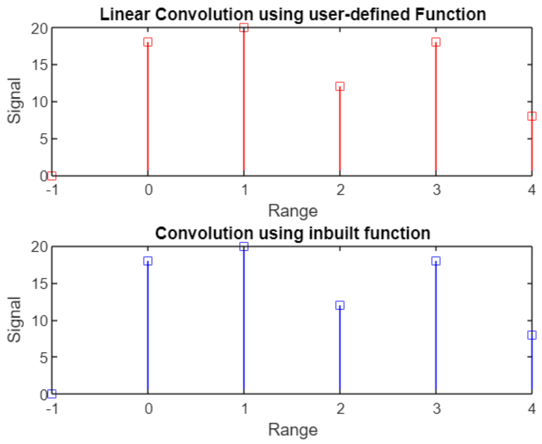



**Figure: 4th Sequence**

2. Solution Problem-2

   (a) Approach: In this question, the methodology is same as the 1st question. For cross-correlation, second sequence will be flipped and will be convoluted with the first sequence. Setting up the axis will be same as it is done in the 1st question. xcorr function will be applied for cross-correlation using inbuilt function.

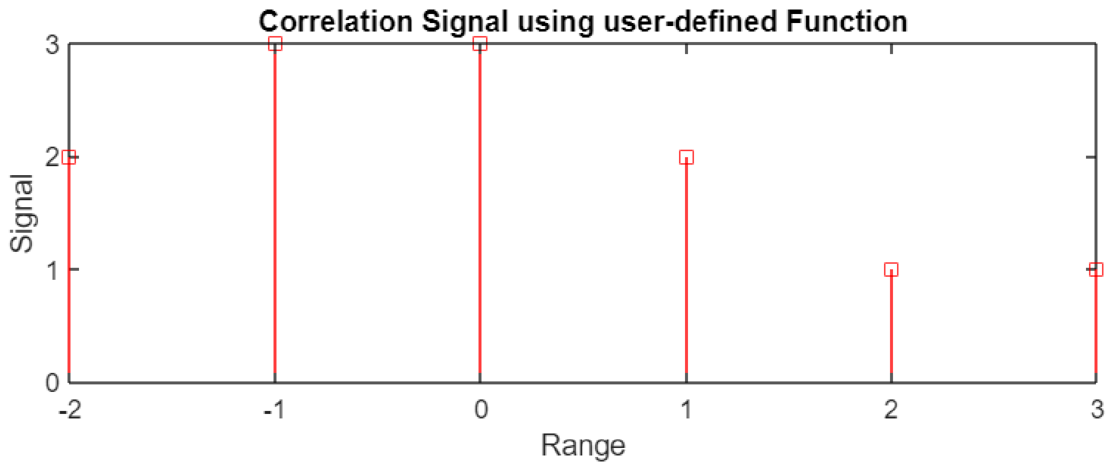
   (b) Matlab Script:

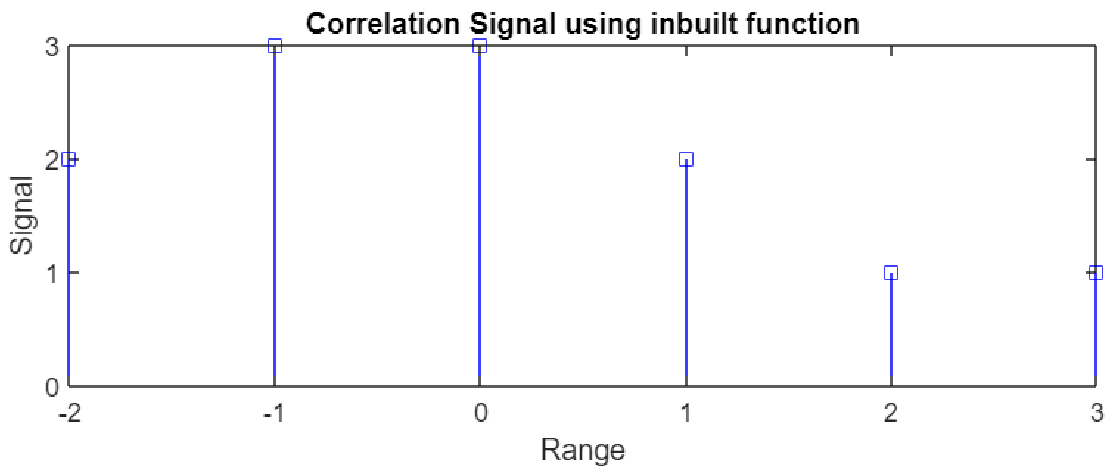
```
% Name : Samarth Shah
% Roll No: AU1841145
% Lab2 (Question_2) we have to do cross-correlation using our own function as well
    as inbuilt function
clc ;
close all ;
clear ;
series1_input_x =  [1,2,2,1] ; % first signal input
series2_input_h = [1,-1,2]; % second signal input
series1_range_x = [-1:2]; % first signal range
series2_range_h = [-2:0]; % first signal range
% Calculating the length of the ranges
series1_length = length ( series1_range_x ); % calculating the length of range
series2_length = length ( series2_range_h ); % calculating the length of range
flipped_series_2 = flip ( series2_input_h ); % Flipping second sequence
%getting output from the user-defined correlation function
final_length1 =[ series1_input_x , zeros(1 , series2_length ) ]; % making length
    same
final_length2 =[ flipped_series_2 , zeros(1 , series1_length ) ]; % making second
    input length same
max_series2 = max ( series2_range_h ); % finding max of index2
min_series2 = min ( series2_range_h ); % finding min of index2
max_series1 = max ( series1_range_x ); % finding max of index1
min_series1 = min ( series1_range_x ); % finding min of index1
output_convolution = linear_convolution (final_length1 ,final_length2 ,
    series1_length , series2_length  );
%getting output from the inbuilt correlation function
out_system_function = xcorr ( series1_input_x , series2_input_h ); % cross
    correlation inbuilt function
start_point = min ( min_series2 , min_series1 ); % Starting index
end_point = min ( min_series2 , min_series1 )+ series1_length + series2_length -2;
     % Ending index
axis_range =[ start_point : end_point ]; % range
subplot (211)
stem (axis_range ,output_convolution,'rs-') ; % discrete plot
% Graphs
ylabel ('Signal');
xlabel ('Range');
title ('Correlation Signal using user-defined Function') ;
subplot (212)
stem (axis_range , out_system_function (2: length ( out_system_function )),'bs-' )
    ; % discrete plot
ylabel ('Signal');
xlabel ('Range');
title ('Correlation Signal using inbuilt function') ;
```

(c) Function:

```
function func_out = linear_convolution(series1_input_x ,series2_input_h ,
    series1_range_x , series2_range_h )
    for i = 1:series1_range_x+series2_range_h-1
        output(i) =0;
        for j = 1:series1_range_x
            if(i-j+1>0) % final range should be more than 0
                output(i)=output(i)+series1_input_x(j).*series2_input_h(i-j+1);
                %Convolution function
            end
        end
    end
    func_out = output; % output of function
end
```

(d) Simulation Output:



3. Solution Problem-3

(a) Approach: Here, both the sequence will be same as it is auto-correlation. It is done using user defined function and linear convolution. xcorr function will be applied for auto-correlation using inbuilt function.

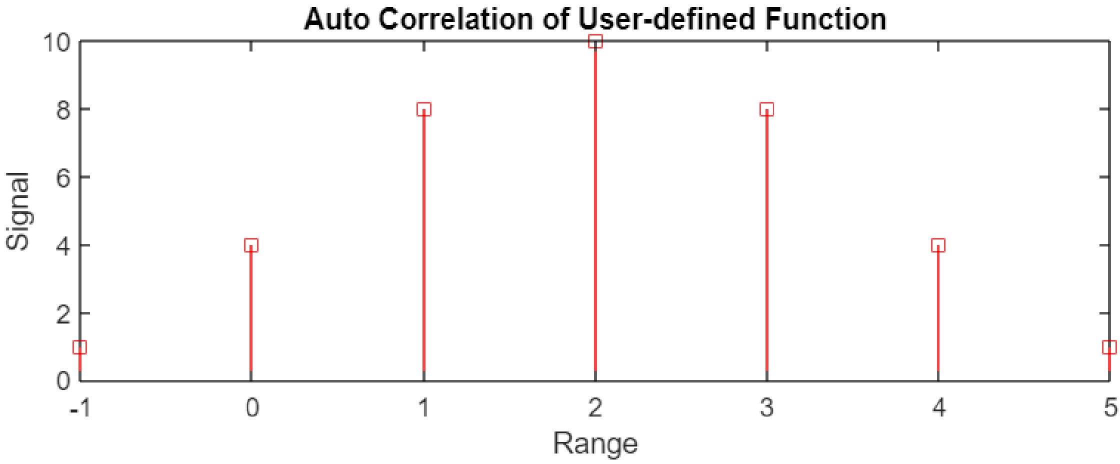
(b) Matlab Script:

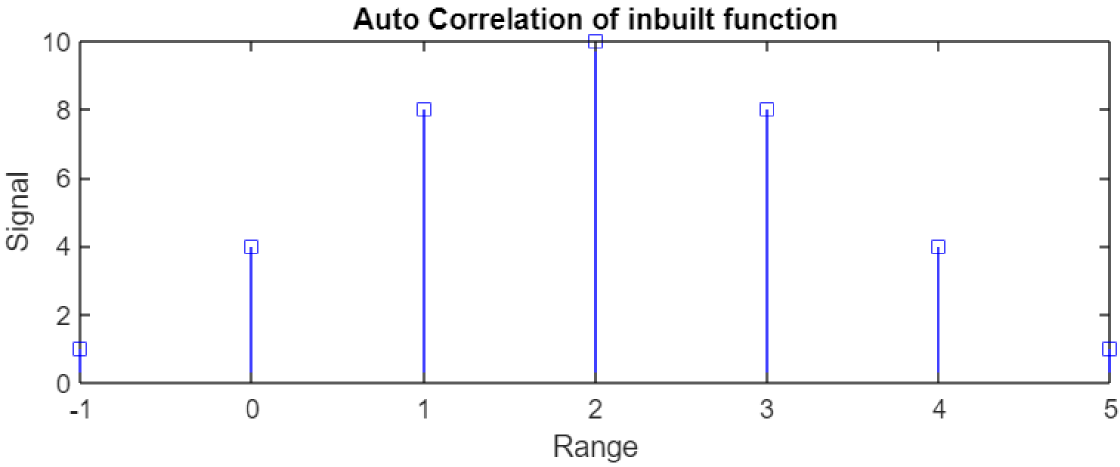
```
% Name : Samarth Shah
% Roll No: AU1841145
% Lab2 (Question_3) we have to do Auto-correlation using our own function as well
    as inbuilt function
clc ;
close all ;
clear all ;
series1_input_x =  [1,2,2,1] ; %  % first signal input
series1_range_x = [-1:2]; % first signal range
min_series1 = min ( series1_range_x ) ; % finding minimum index
range_number = ( min_series1 * -1) + 1;
range_series1 = length ( series1_input_x ); % range of first index
length_series =2* range_series1 -1; % length of output
%User Defined function
out_user_defined = AutoCorrelation( series1_input_x ) ; % Autocorrelation function
%Inbuilt function
output_inbuilt = xcorr ( series1_input_x ) ; % inbuilt function xcorr has one
    input so it will automatically second input as first input also so this will
    be autocorrelation
axis = ( -( range_number -1) ) :(length_series -1 -( range_number -1) ); %x axis
    of output
subplot (211)
stem (axis , out_user_defined ,'rs-'); % output
xlabel ('Range')
ylabel ('Signal')
hold on ;
title ('Auto Correlation of User-defined Function')
subplot (212)
stem (axis , output_inbuilt ,'bs-'); % output
xlabel ('Range')
ylabel ('Signal')
hold on ;
title ('Auto Correlation of inbuilt function')
```

(c) Function:

```
function out = AutoCorrelation( sequence )
    length_series = length ( sequence ); % length of series
    total_terms =2* length_series -1; %l = m+n -1 terms size of output
    seq_1= zeros (1 , total_terms ) ; % length same first all zero
    seq_2= zeros (1 , total_terms ) ; % length same first all zero
    flipped_seq = fliplr ( sequence ); % flipping sequence
    seq_1 (1: length_series )= sequence ;
    seq_2 (1: length_series )= flipped_seq ;
    % performing linear convolution
    for i =1: total_terms % total length
        output(i ) =0; % initializing to zero
        for j =1: i
            output(i )=output( i)+seq_1( j)*seq_2(i -j +1) ;
        end
    end
    out = output ;
end
```

(d) Simulation Output:



4. Solution Problem-4

(a) Approach: In this question, both of the sub questions have infinite ranges. Because of that, a range [-20:20] was predefined. Both the sub question were done using linear convolution function. That function was user-defined.

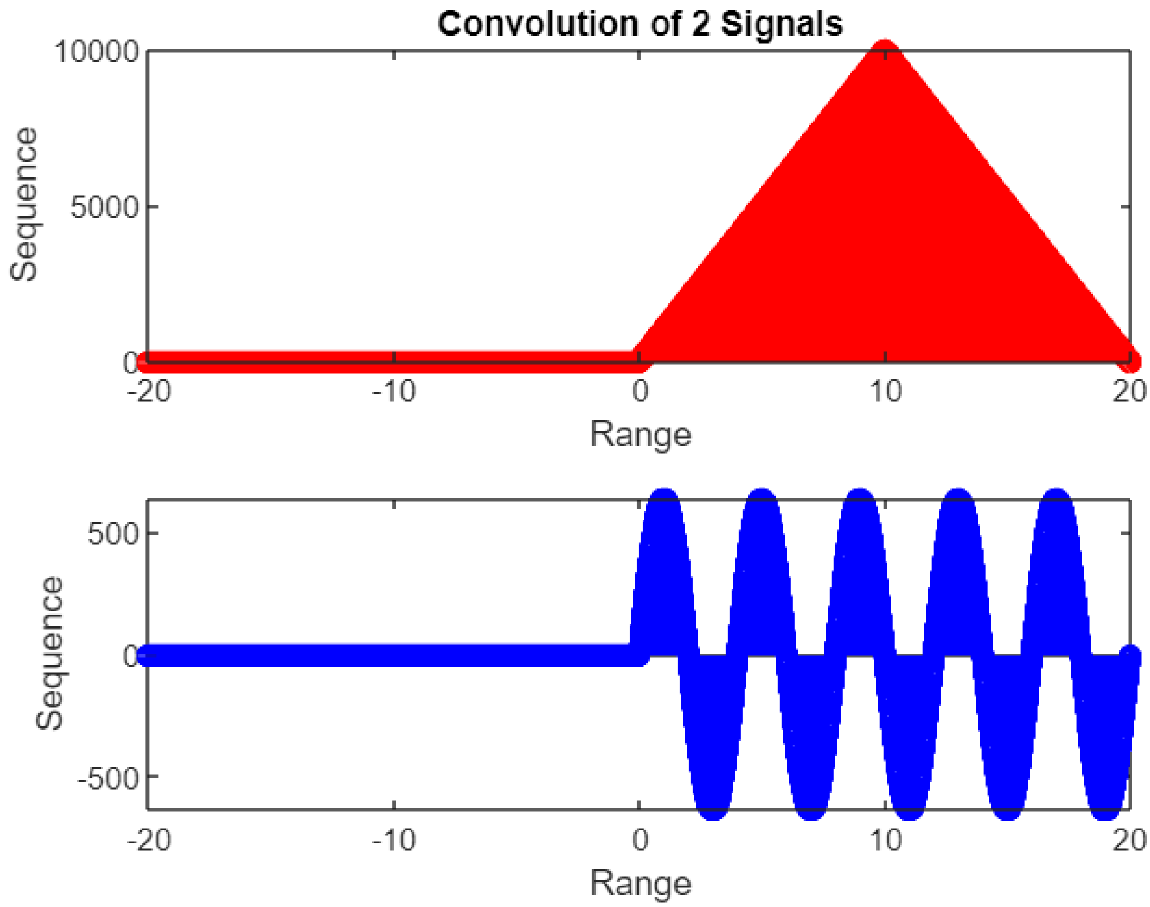
(b) Matlab Script:

```
% Name : Samarth Shah
% Roll No: AU1841145
% Lab2 (Question_4) Find the linear convolution for following infinite length sequences and Plot required outputs.
clc ;
close all ;
clear all ;
%For first sub-question
seq_range=[-10:0.001:10]; %defining the appropriate range
unit_step=[seq_range>=0]; %Unit step Function
output=updated_convolution(unit_step,unit_step,seq_range,seq_range); % User Defined Function
range__output=-20:0.001:20; %Output Range
%Plotting graphs
subplot(211);
stem(range__output,output,'r'); %Plotting the graph
title('Convolution of 2 Signals');
xlabel('Range');
ylabel('Output Function');
%For Second Sub-Question
function2=cos((2.*seq_range.*pi)/4).*unit_step; %Equation of 2nd Sub question
output2=updated_convolution(function2,unit_step,seq_range,seq_range);
subplot(212);
stem(range__output,output2,'b');%Plotting the graph
xlabel('Range');
ylabel('Output Function');
```

(c) Function:

```
function output=updated_convolution(seq_1,seq_2,range_1,range_2)
    length_sequence_1=length(range_1); %finding the length of seq-1
    length_sequence_2=length(range_2);%finding the length of seq-2
    final_range=length_sequence_2+length_sequence_1-1;
    seq_1=[seq_1,zeros(1,length_sequence_2)]; %making same length of sequence
    seq_2=[seq_2,zeros(1,length_sequence_1)];%making same length of sequence
    for i=1: final_range
        %i goes from 1 to length of final range
        output(i)=0;
        %initializing the value of output
        for j=1: length_sequence_1
            if(i-j+1>0)
                output(i)=output(i)+seq_1(j).*seq_2(i-j+1); %convolution function
            end
        end
    end
end
```

(d) Simulation Output:



5. Problem- Analysis of Sobel Edge detection

   (a) Approach: Sobel edge detection uses two types of gradients, in the X-Direction and in the Y-Direction. In order to detect the edges, first derivative is taken. This computes an approximation of the gradient of an image. Consider, the original image taken here is I. So, Horizontal changes would be convoluting I with Gx and Vertical changes would be convoluting I with Gy. Gx and Gy are 3*3 Kernels. Sobel edge detection is done using Matlab platform. In the first program, basic kernels were considered for the edge detection.

$$\text{Gx} = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix} \quad \text{Gy} = \begin{pmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{pmatrix}$$

   Here's the output of that program,

**Figure: Original Image**

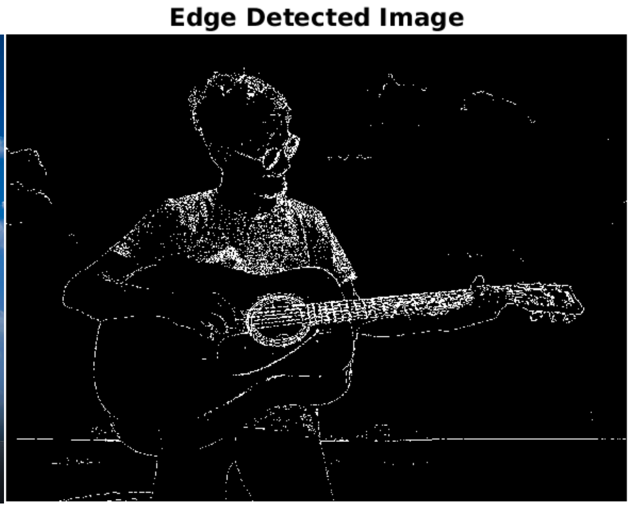



**Figure: Edge detected Image**

(b) Now, The kernels were modified by increasing the weights by 5 of the side edges. We can see there is an increment in the sharpness w.r.t. 1st example.

$$\mathrm{Gx} = \begin{pmatrix} -1 & 0 & 1 \\ -5 & 0 & 5 \\ -1 & 0 & 1 \end{pmatrix} \quad \mathrm{Gy} = \begin{pmatrix} -1 & -5 & -1 \\ 0 & 0 & 0 \\ 1 & 5 & 1 \end{pmatrix}$$

Here's the output of that program,

Figure: Original Image

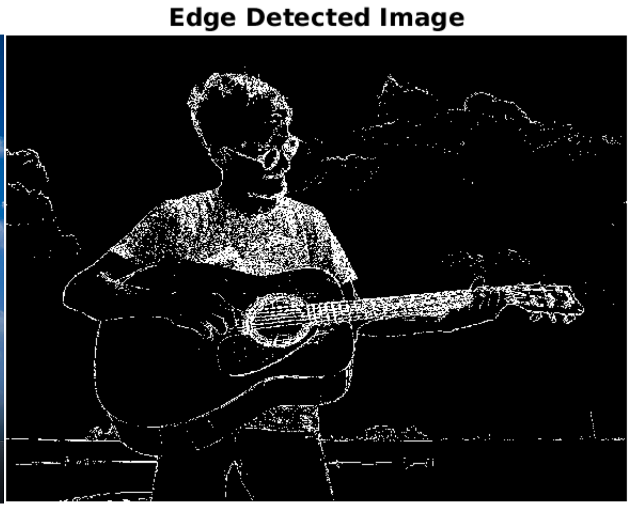



Figure: Edge detected Image

(c) Again, The kernels were modified by increasing the weights by 1 of the corner edges. We can see there is an increment in the sharpness w.r.t. 1st and 2nd example.

$$\mathrm{Gx} = \begin{pmatrix} -2 & 0 & 2 \\ -5 & 0 & 5 \\ -2 & 0 & 2 \end{pmatrix} \quad \mathrm{Gy} = \begin{pmatrix} -2 & -5 & -2 \\ 0 & 0 & 0 \\ 2 & 5 & 2 \end{pmatrix}$$

Figure: Original Image

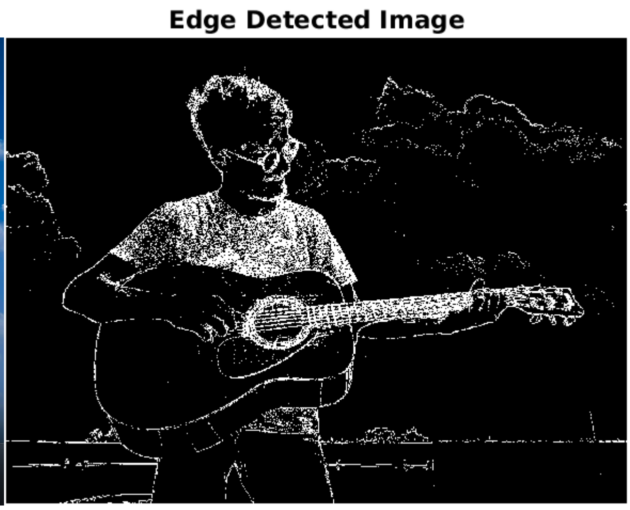



Figure: Edge detected Image

(d) Corners of both kernels were replaced with zero. We can see there is dullness in the image w.r.t. all the images.

$$\mathrm{Gx} = \begin{pmatrix} -0 & 0 & 0 \\ -5 & 0 & 5 \\ -0 & 0 & 0 \end{pmatrix} \quad \mathrm{Gy} = \begin{pmatrix} -0 & -5 & -0 \\ 0 & 0 & 0 \\ 0 & 5 & 0 \end{pmatrix}$$

**Figure: Original Image**

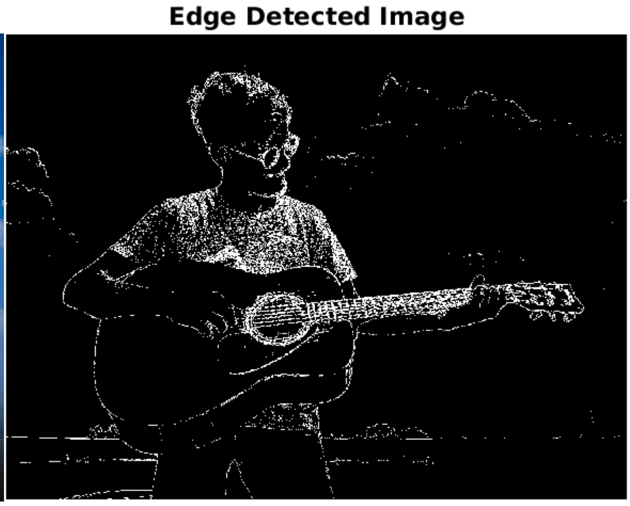



**Figure: Edge detected Image**