BAPATLA ENGINEERING COLLEGE (AUTONOMOUS)

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

14ECL703 : SIGNAL & IMAGE PROCESSING LAB MANUAL

2018-19

PREPARED BY

D SUNEEL VARMA, M.Tech
ASSISTANT PROFESSOR
DEPARTMENT OF ECE

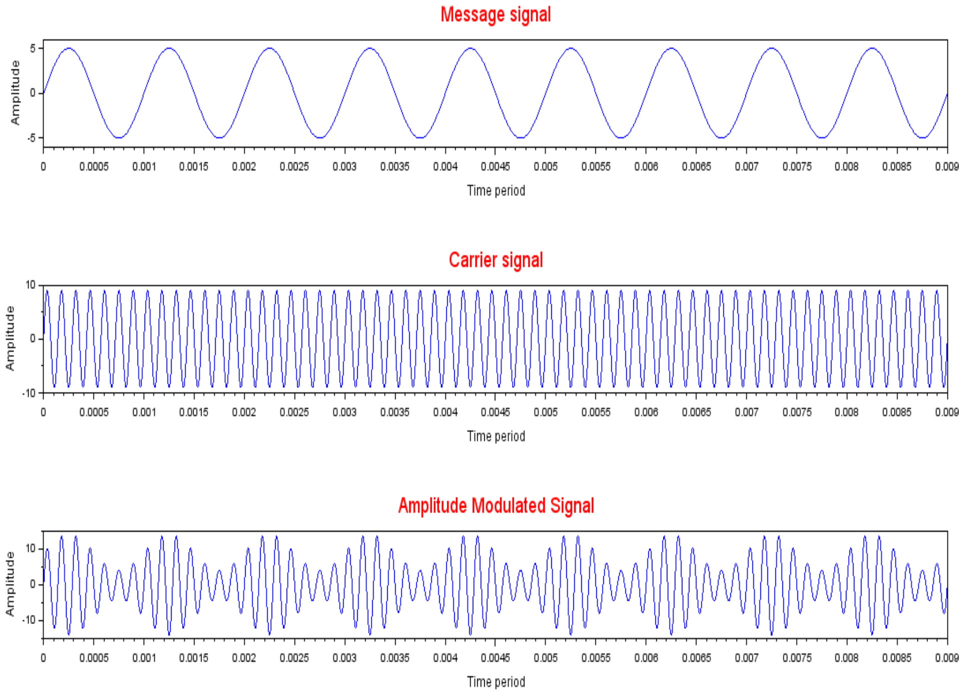
SD IMRAN BASHA, M.Tech
ASSISTANT PROFESSOR
DEPARTMENT OF ECE

LIST OF PROGRAMS

1. AMPLITUDE MODULATION

```
clc;
clear();
xdel(winsid());
Am=input('Enter Modulating signal amplitude value  : ')
Fm=input('Enter frequency value  : ')

Ac=input('Enter carrier signal amplitude value  : ')
Fc=input('Enter frequency value  : ')

n=input('enter no. of cycles  : ')
t=(0:(1/(1000*Fc)):n/Fm)

Vm = Am*sin(((2*%pi)*Fm)*t)
subplot(311)
plot(t,Vm)
title('Message signal','color','red','fontsize',4)
xlabel('Time period','fontsize',2)
ylabel('Amplitude','fontsize',2)

Vc = Ac*sin(((2*%pi)*Fc)*t)
subplot(312)
plot(t,Vc)
title('Carrier signal','color','red','fontsize',4)
xlabel('Time period','fontsize',2)
ylabel('Amplitude','fontsize',2)

//m=ka*Am;
//-----------ka=1/Ac

m = Am/Ac;//when Ka(amplitude sensitivity)is not specified

//case - m<1 for under mod

// plot for under, critical, over modulated signals with different m values

Vamp = (Ac*(1+m*sin(((2*%pi)*Fm)*t))) .*sin(((2*%pi)*Fc)*t);
subplot(313)
plot(t,Vamp)
title('Amplitude Modulated Signal','color','red','fontsize',4)
xlabel('Time period','fontsize',2)
ylabel('Amplitude','fontsize',2)
```

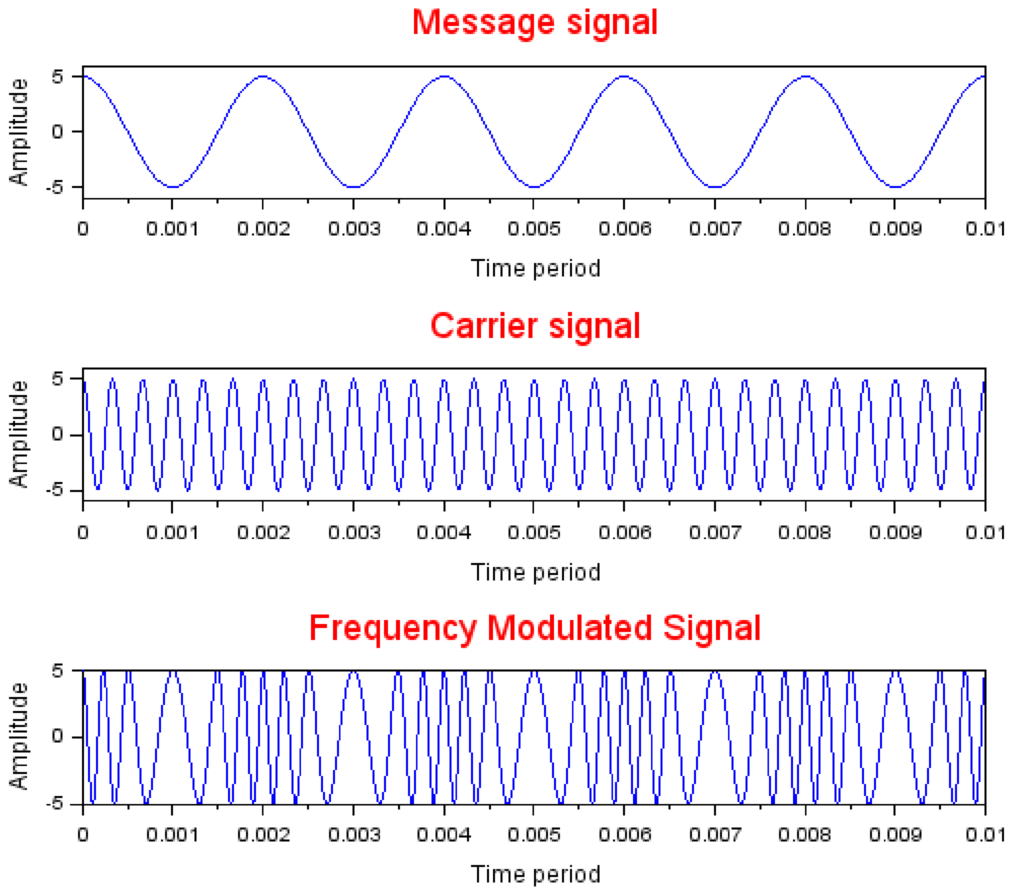
OUTPUT:


Message signal
Amplitude
5
0
-5
0
0.0005
0.001
0.0015
0.002
0.0025
0.003
0.0035
0.004
0.0045
0.005
0.0055
0.006
0.0065
0.007
0.0075
0.008
0.0085
0.009
Time period
Carrier signal
Amplitude
10
0
-10
Time period
Amplitude Modulated Signal
Amplitude
10
0
-10
Time period

2. FREQUENCY MODULATION

```
clc;
clear();
xdel(winsid());
Am=input('Enter Modulating signal amplitude value  : ')
Fm=input('Enter frequency value  : ')

Ac=input('Enter carrier signal amplitude value  : ')
Fc=input('Enter frequency value  : ')

//mod index((kf*Am)/Fm) <1 for Nrbnd FM and >1 for Wdbnd FM

Kf=input('enter Frequency sensitivity  : ')


n=input('enter no. of cycles  : ')
t=(0:(1/(1000*Fc)):n/Fm)

Vm = Am*cos(((2*%pi)*Fm)*t)
subplot(311)
plot(t,Vm)
title('Message signal','color','red','fontsize',4)
xlabel('Time period','fontsize',2)
ylabel('Amplitude','fontsize',2)

Vc = Ac*cos(((2*%pi)*Fc)*t)
subplot(312)
plot(t,Vc)
title('Carrier signal','color','red','fontsize',4)
xlabel('Time period','fontsize',2)
ylabel('Amplitude','fontsize',2)

m = ((Kf*Am)/Fm);


Vfm = Ac*cos((2*%pi*Fc*t)+(m*sin(2*%pi*Fm*t)));
subplot(313)
plot(t,Vfm)
title('Frequency Modulated Signal','color','red','fontsize',4)
xlabel('Time period','fontsize',2)
ylabel('Amplitude','fontsize',2)
```

OUTPUT:



Message signal

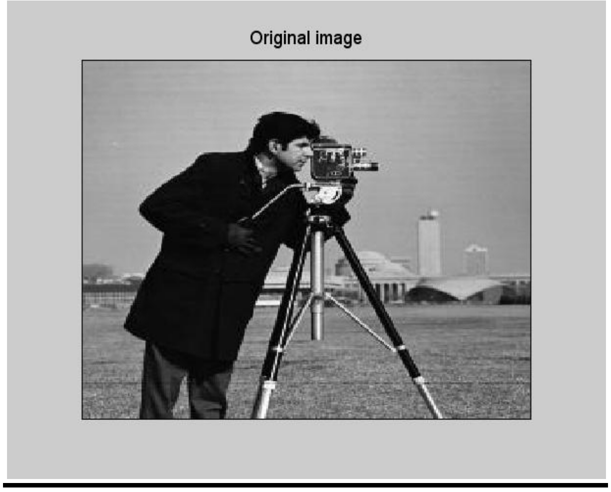
Carrier signal

Frequency Modulated Signal

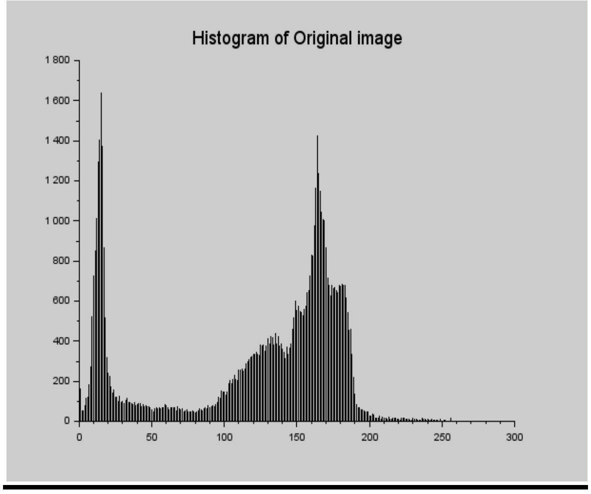
3. HISTOGRAM EQUALIZATION

```
clc;
clear();
xdel(winsid());
a=imread('path to Cameraman Image');
[m n]=size(a);
// Histogram of Input Image
for i=1:256
   b(i)=length(find(a==(i-1)));
end
//Applying Histogram Equalization
pb=b/(m*n)
cmpb(1)=pb(1);
for i=2:256
    cmpb(i)=pb(i)+cmpb(i-1)
end
    ni=(cmpb*255);
    new=uint8(round(ni))
for i=1:m
    for j=1:n
        ind=double(a(i,j));
        hea(i,j)=new(ind+1);
        end
end
figure
imshow(a)
title('Original image','fontsize',4)
figure
plot2d3(b);
title('Histogram of Original image','fontsize',4)
figure
imshow(uint8(hea));
title('Equalized image','fontsize',4)

for i=1:256
   c(i)=length(find(hea==i-1))
end
figure
plot2d3(c)
title('Histogram representation of Equalized Image','fontsize',4)
```

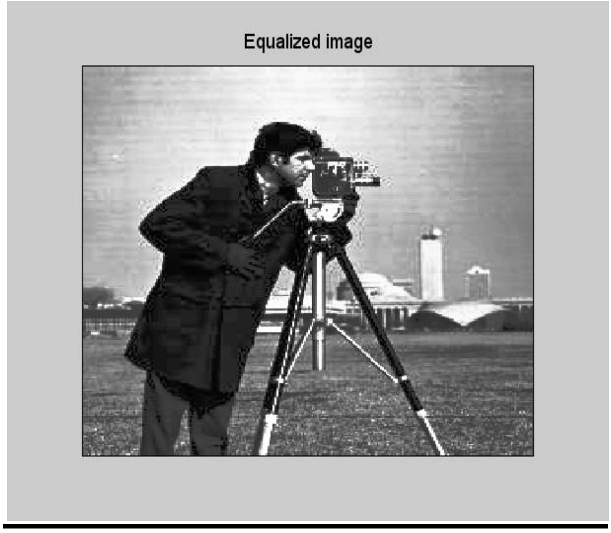
OUTPUT:



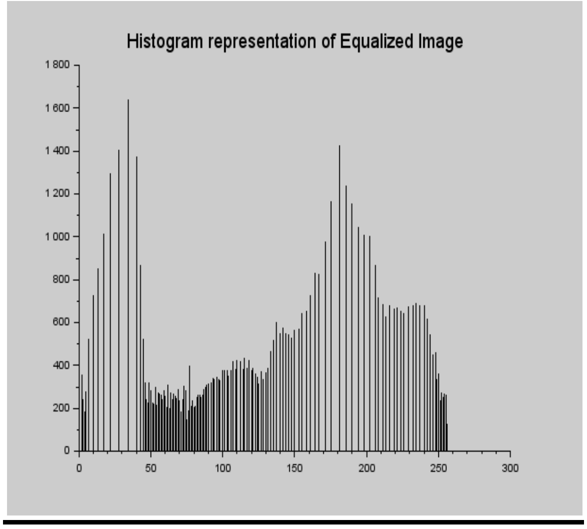
Original image



Histogram of Original image



Equalized image

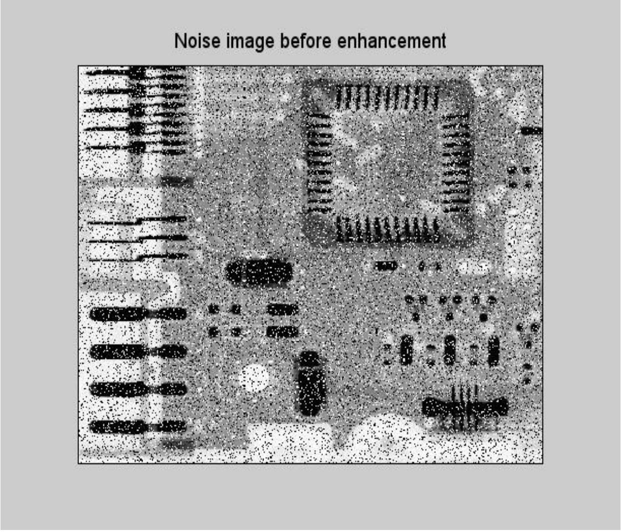


Histogram representation of Equalized Image

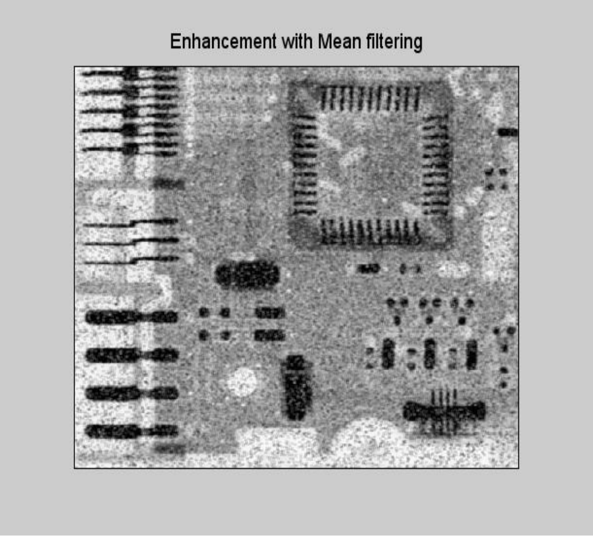
4. KERNEL PROCESSING

```
clc
clear()
xdel(winsid())
a=imread('path to cktnoise image'); //SIVP toolbox

[m n]=size(a);    //storing original image size m rows and n columns

a1=zeros(m+2,n+2)      //new image a1 with all zeros and size m+2 rows and n+2 columns

for i=2:m+1        //creating a1 as original image but border with all zeros remains unchanged
   for j=2:n+1
      a1(i,j)=a(i-1,j-1)
   end
end

for i=2:m+1       //creating mask 3x3      and   finding mean & median which stores in b and c
respectively
   for j=2:n+1
      mask=a1((i-1):(i+1),(j-1):(j+1));
      b(i-1,j-1)=mean(mask);
      c(i-1,j-1)=median(mask);
   end
end

figure
imshow(a)
title ('Noise image before enhancement','fontsize',4);

figure
imshow(uint8(b))
title ('Enhancement with Mean filtering','fontsize',4);

figure
imshow(uint8(c))
title('Enhancement with Median filtering','fontsize',4);
```

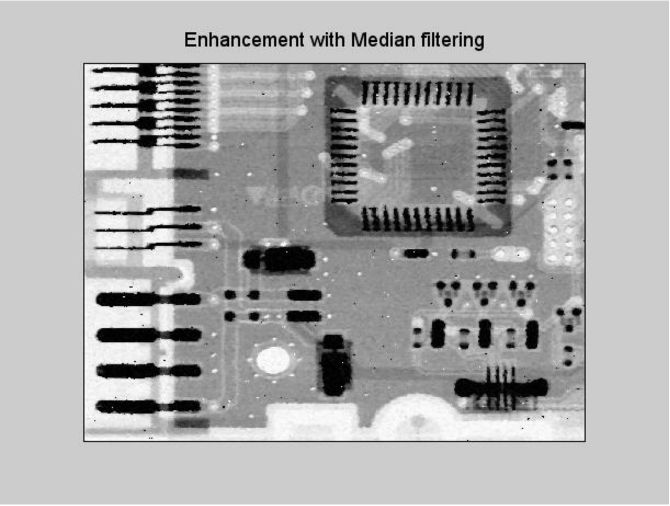
OUTPUT:



Noise image before enhancement



Enhancement with Mean filtering



Enhancement with Median filtering

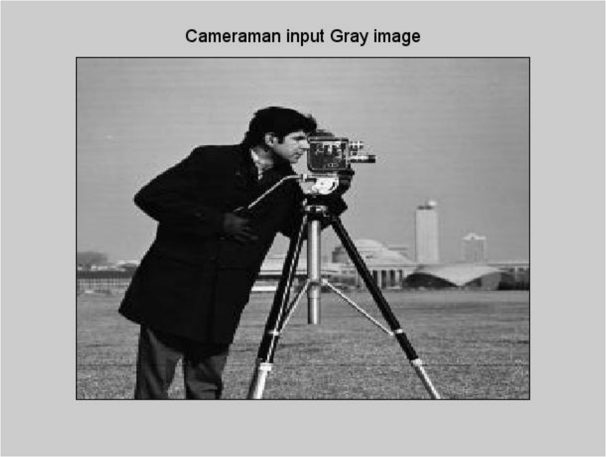
5. WATERMARKING

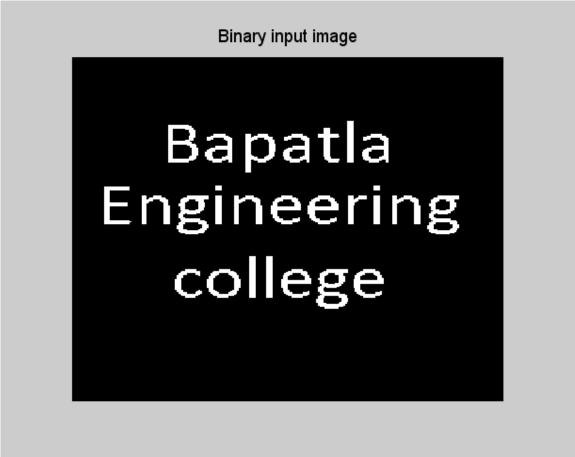
```
clc
clear()
xdel(winsid())
a=imread('path to cameraman image')
b=imread(' path to wat image')
[m n]=size(a);     //storing 1st image size m rows and n columns
[p q]=size(b);     //storing 2nd image size p rows and q columns
// Note : 1) here two input images are same size ie.., m=p & n=q
//        2) Here Second image is Binary image ie.., L=2 & k=1

figure
imshow(a)
title('Cameraman input Gray image','fontsize',4)
figure
imshow(b)
title('Binary input image','fontsize',4)
//if image b is grey scale image then convert it into binary image using thresholding

//for i=1:p
//    for j=1:q
//        if b(i,j)<128
//            then b(i,j)=0
//        else
//            then b(i,j)=1
//            end
//end     //this set of lines only applicable to jpeg or jpg or png where intensities are in integer form,
//..........................................but here image b is in bmp form so we need to convert intensities which are in Boolean into integer

for i=1:m      //watermarking the image a with image b by placing b intensities into LSB(1) and MSB(8) of a
    for j=1:n
        c(i,j)=bitset(a(i,j),1,uint8(b(i,j)))     //setting bit at specified position
        d(i,j)=bitset(a(i,j),8,uint8(b(i,j)))
        end
end
figure
imshow(uint8(c))
title('Bitset image with position 1','fontsize',4)
figure
imshow(uint8(d))
title('Bitset image with position 8','fontsize',4)
for i=1:m      // obtaining back the second input image with reference to bits of image b stored in LSB and MSB of c & d
    for j=1:n
        e(i,j)=bitget(c(i,j),1)     //getting bit at specified position
        f(i,j)=bitget(d(i,j),8)
        end
```

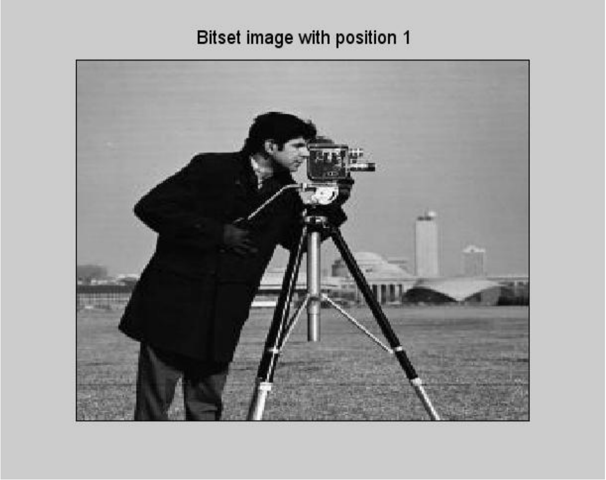
```
end

errl=double((a-c)).^2
errm=double((a-d)).^2    //to know how much intensities changed or effected
sqerrl=sum(sum(errl));
sqerrm=sum(sum(errm));
MSEl=(sqerrl/(m*n));
MSEm=(sqerrm/(m*n));
zl=log10((255*255)/MSEl);
zm=log10((255*255)/MSEm);
PSNRl=(10*zl);
PSNRm=(10*zm);
g=corr2(b,e);  //to see image b and extracted images are same or not
h=corr2(b,f);
disp('PSNR output : ',PSNRl);
disp('PSNR output : ',PSNRm);
disp('correlation output : ',g);
disp('conrrelation output : ',h)
figure
imshow(double(e))
title('Bitget image with position 1','fontsize',4)
figure
imshow(double(f))
title('Bitget image with position 8','fontsize',4)
```

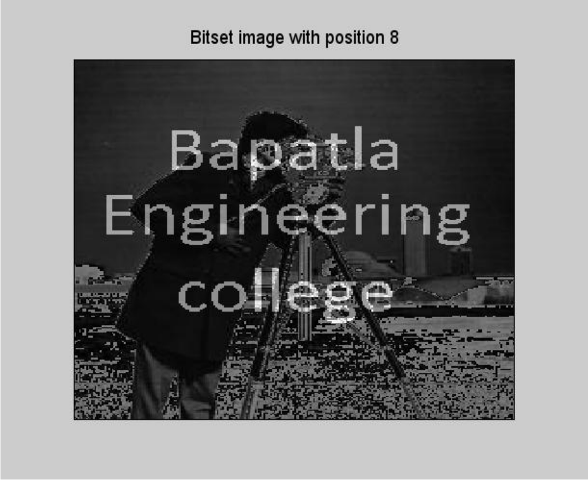
OUTPUT:

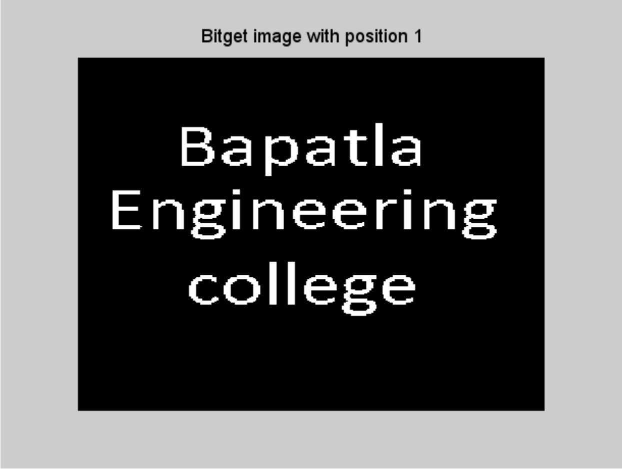
```
PSNR output: 14.624737

PSNR output: 8.2046058

Correlation output: 1.

Correlation output: 1.
```

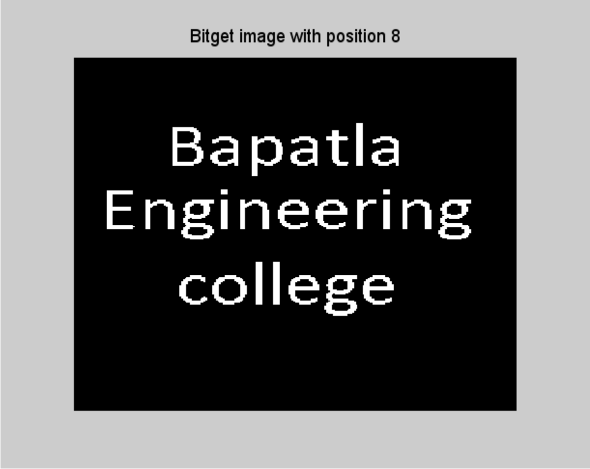

Cameraman input Gray image


Binary input image
Bapatla
Engineering
college


Bitset image with position 1


Bitset image with position 8
Bapatla
Engineering
college


Bitget image with position 1
Bapatla
Engineering
college

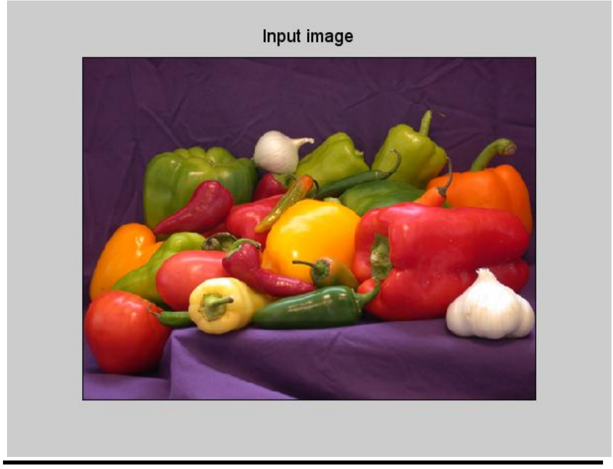

Bitget image with position 8
Bapatla
Engineering
college

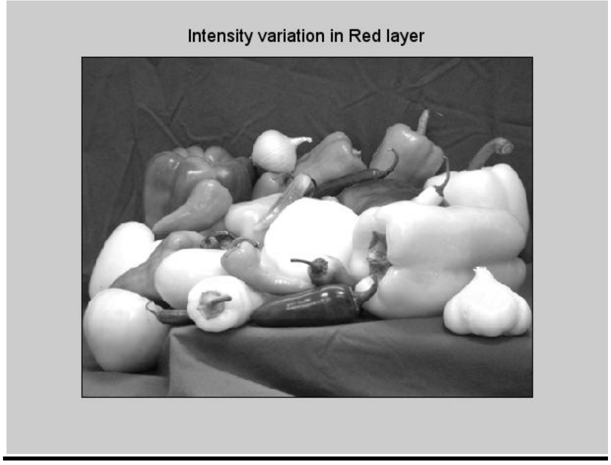
6. COLOR IMAGE MANIPULATIONS

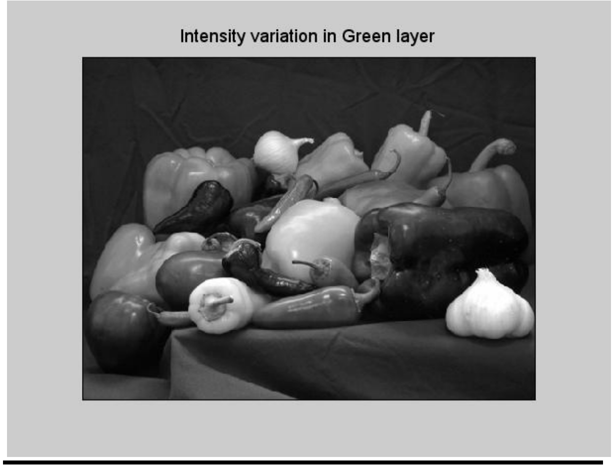
```
//Color images manipulations, reading and writing of color images
clc
clear()
xdel(winsid())
a=imread('path to peppers image');
figure
imshow(a)
title('Input image','fontsize',4)

//Intensity variation in Red layer
figure
imshow(a(:,:,1))
title('Intensity variation in Red layer','fontsize',4)
//Intensity variation in Green layer
figure
imshow(a(:,:,2))
title('Intensity variation in Green layer','fontsize',4)
//Intensity variation in Blue layer
figure
imshow(a(:,:,3))
title('Intensity variation in Blue layer','fontsize',4)

//Representing image in Red color
b=a
b(:,:,2:3)=0
figure
imshow(b)
title('Representation of image in Red layer','fontsize',4)
//Representing image in Green color
c=a
c(:,:,1)=0
c(:,:,3)=0
figure
imshow(c)
title('Representation of image in Green layer','fontsize',4)
//Representing image in Blue color
d=a
d(:,:,1:2)=0
figure
imshow(d)
title('Representation of image in Blue layer','fontsize',4)

//Representation of image using combination of Green and Blue
e=a
e(:,:,1)=0
figure
imshow(e)
title('Representation of image  using Green and Blue','fontsize',4)
```

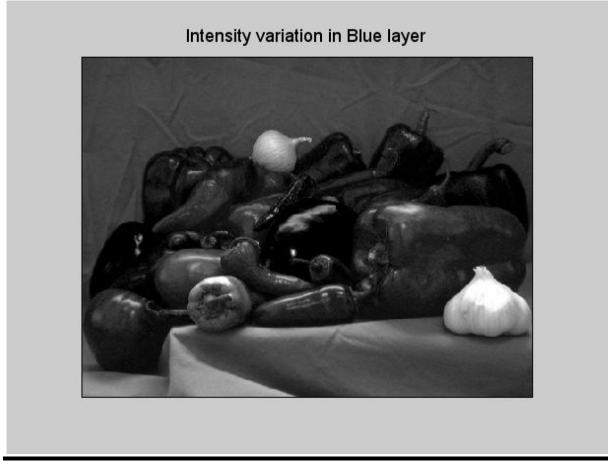
```
//Representation of image using combination of Red and Blue
f=a
f(:,:,2)=0
figure
imshow(f)
title('Representation of image  using Red and Blue','fontsize',4)
//Representation of image using combination of Red and Green
g=a
g(:,:,3)=0

figure
imshow(g)
title('Representation of image  using Red and Green','fontsize',4)

//conversions
c1=rgb2hsv(a)
figure
imshow(c1)
title('rgb2hsv converted image','fontsize',4)
c2=rgb2gray(a)
figure
imshow(c2)
title('rgb2gray converted image','fontsize',4)
c3=rgb2ntsc(a)
figure
imshow(c3);
title('rgb2ntsc converted image','fontsize',4)
c4=rgb2ycbcr(a)
figure
imshow(c4);
title('rgb2ycbcr converted image','fontsize',4)
c5=rgb2ind(a)
figure
imshow(c5);
title('rgb2ind converted image','fontsize',4)
```

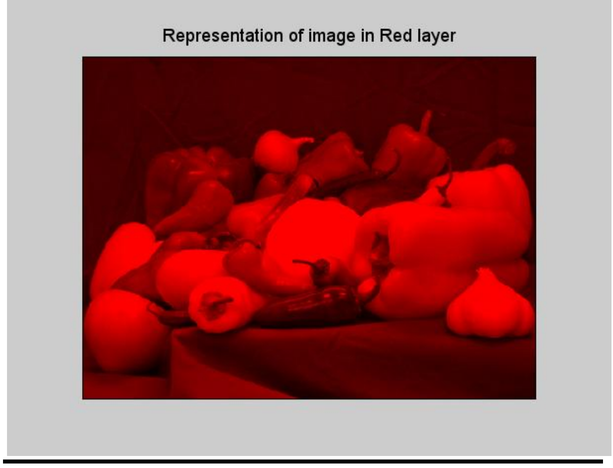
OUTPUT:

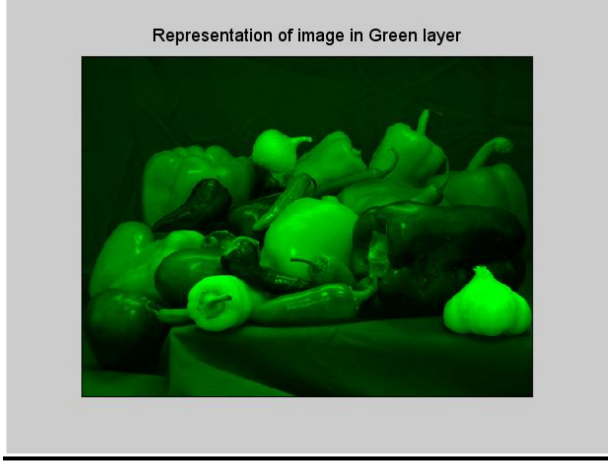

Input image

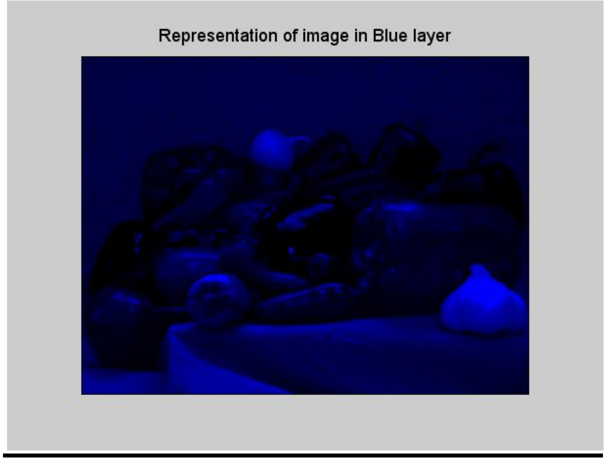

Intensity variation in Red layer

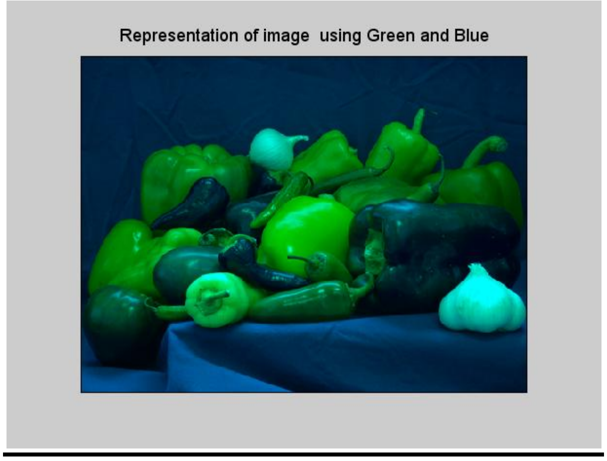

Intensity variation in Green layer

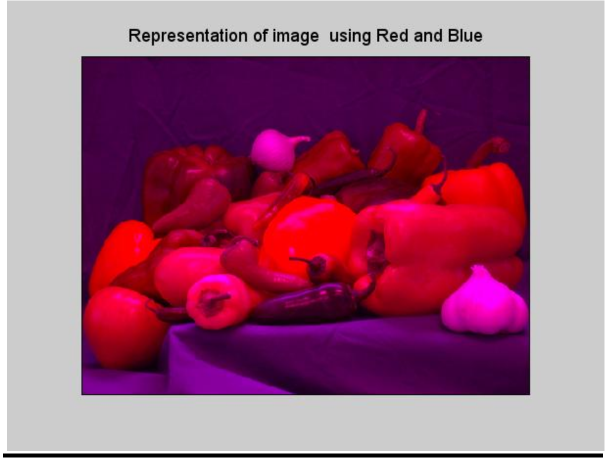

Intensity variation in Blue layer

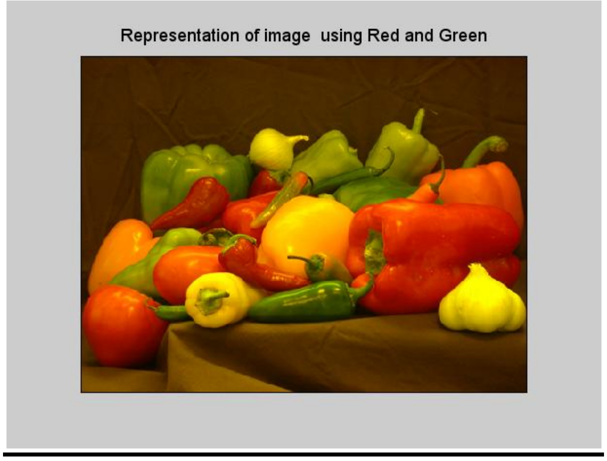

Representation of image in Red layer

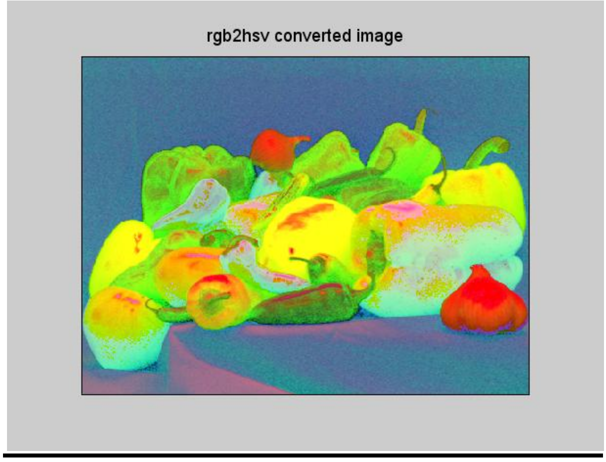

Representation of image in Green layer

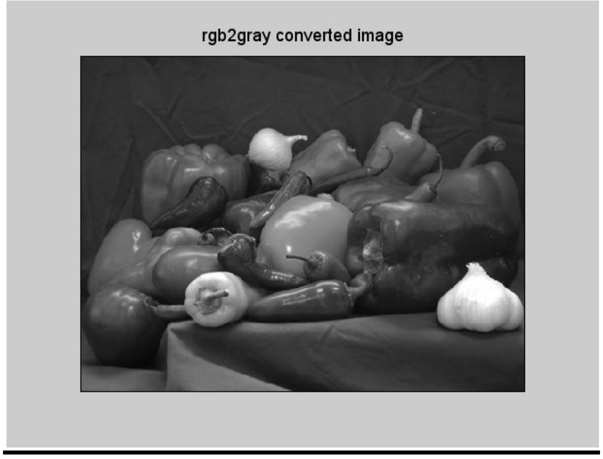

Representation of image in Blue layer

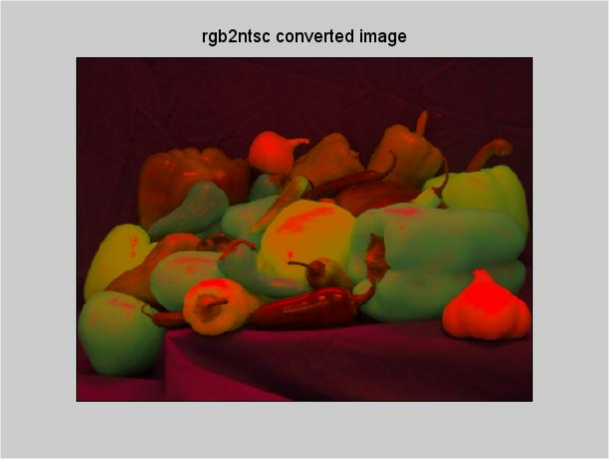

Representation of image using Green and Blue

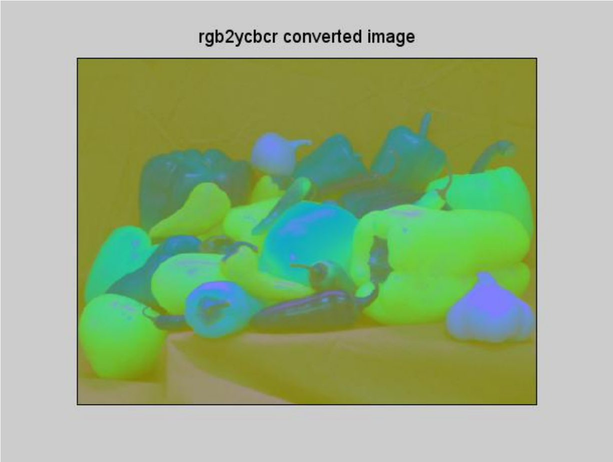

Representation of image using Red and Blue


Representation of image using Red and Green


rgb2hsv converted image


rgb2gray converted image


rgb2ntsc converted image


rgb2ycbcr converted image

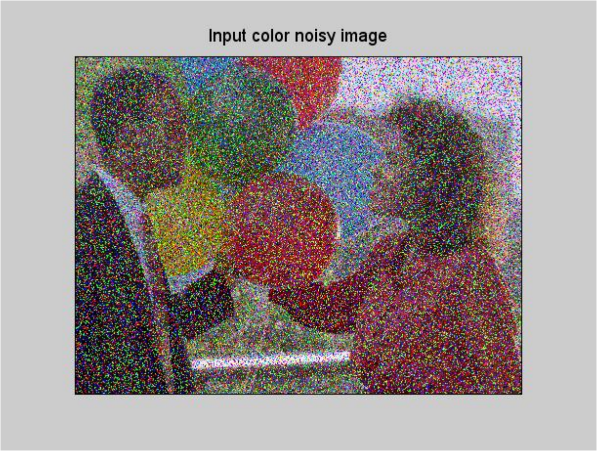
7. COLOR IMAGE ENHANCEMENT

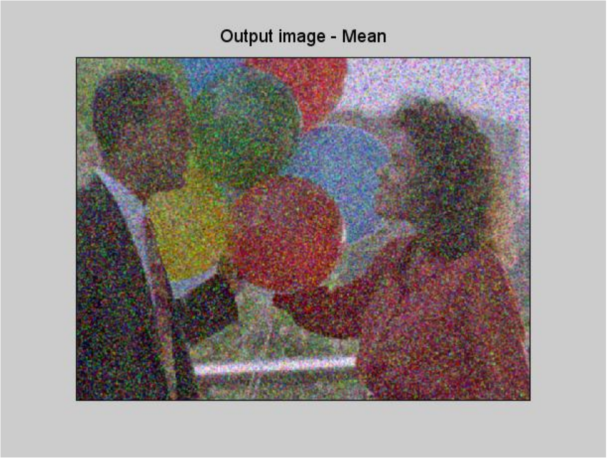
```
//Color image enhancement

//Save the function required for this program in same location

clc
clear()
xdel(winsid())
a=imread('path to balloonsnoisy image');
figure
imshow(a)
title('Input color noisy image','fontsize',4)

exec('path to the corresponding function name as "imageenh.sce" ');  // execution access to
the function in this program

//Applying Image Enhancement for individual layer
for i=1:3
   [p,q]=imageenh(a(:,:,i));
   b(:,:,i)=uint8(p);
   c(:,:,i)=uint8(q);

end
figure
imshow(b)
title('Output image - Mean','fontsize',4)

figure
imshow(c)
title('Output image image - Median','fontsize',4)
```

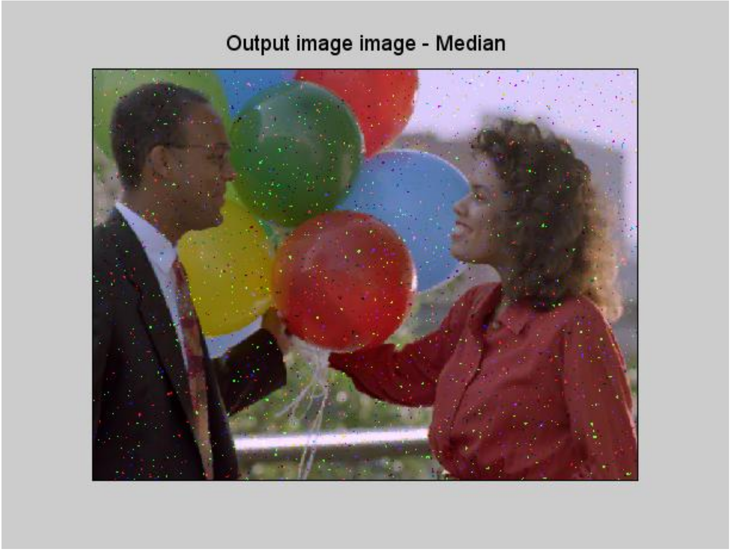
7(a). FUNCTION FOR COLOR IMAGE ENHANCEMENT

```
//Save this file with the function name

//Don't Execute this file  -  This is a function not a program

function [b, c]=imageenh(a)
   [m n]=size(a);    //storing original image size m rows and n columns

a1=zeros(m+2,n+2)      //new image a1 with all zeros with size m+2 rows and n+2 columns

for i=2:m+1        //creating a1 as original image but border with all zeros
   for j=2:n+1
      a1(i,j)=a(i-1,j-1)
   end
end

for i=2:m+1        //creating mask 3x3      and   finding mean & median which stores in b and c
respectively
   for j=2:n+1
      mask=a1((i-1):(i+1),(j-1):(j+1));
      b(i-1,j-1)=mean(mask);
      c(i-1,j-1)=median(mask);
   end
end
endfunction
```

OUTPUT:


Input color noisy image


Output image - Mean


Output image image - Median

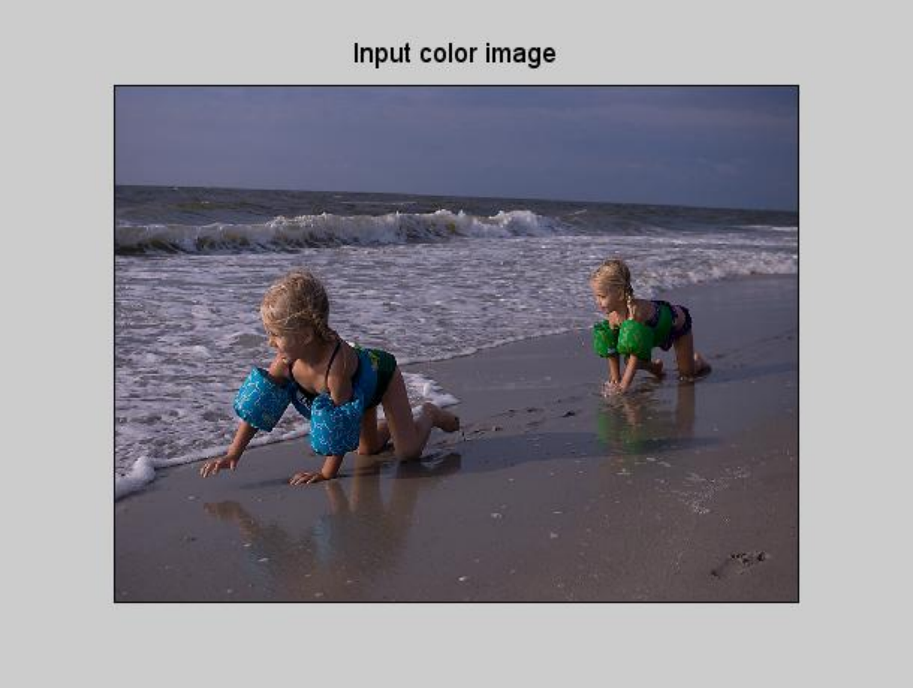
8. COLOR IMAGE HISTOGRAM

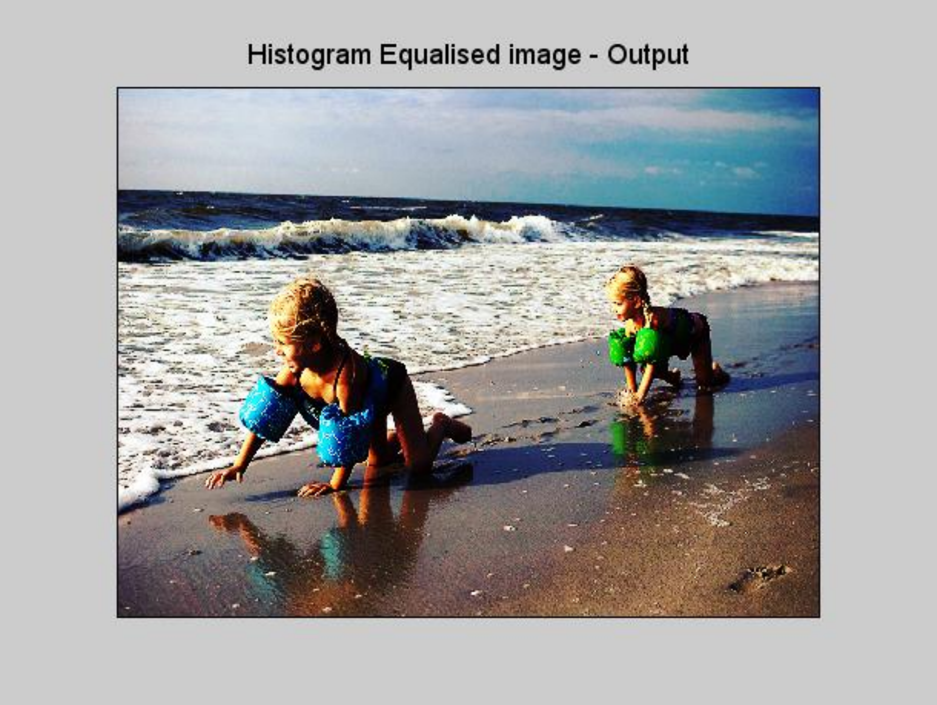
```
//color image histogram manipulations

//Save the function required for this program in same location

clc
clear()
xdel(winsid())
a=imread('Cpath to kids image');
figure
imshow(a)
title('Input color image','fontsize',4);

exec('path to the corresponding function name as “hist_eq.sce” '); // execution access to the
function in this program

//Applying Histogram Equalization for individual layer
for i=1:3
   [p]=hist_eq(a(:,:,i));
   b(:,:,i)=uint8(p);

end
figure
imshow(b)
title('Histogram Equalised image - Output','fontsize',4);
```

8(a). FUNCTION FOR COLOR IMAGE HISTOGRAM

```
//Save this file with the function name

//Don't Execute this file - This is a function not a program

function [hea]=hist_eq(a)
[m n]=size(a);
for i=1:256
   b(i)=length(find(a==(i-1)));
end
pb=b/(m*n)
cmpb(1)=pb(1);
for i=2:256
    cmpb(i)=pb(i)+cmpb(i-1)
end
    ni=(cmpb*255);
    new=uint8(round(ni))
for i=1:m
    for j=1:n
        ind=double(a(i,j));
        hea(i,j)=new(ind+1);
        end
end

endfunction
```

OUTPUT:



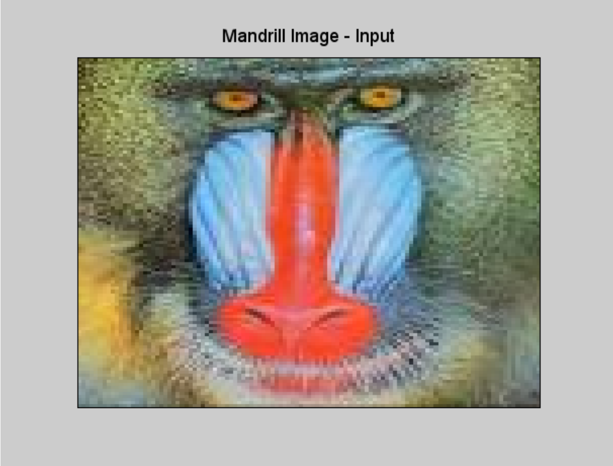


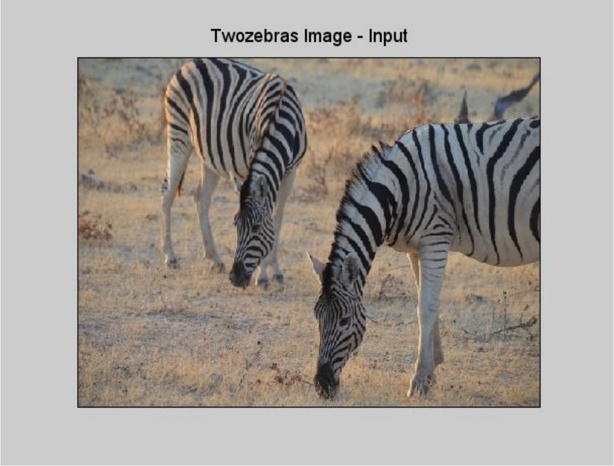
9. SPECIAL EFFECTS ON GRAY & COLOR IMAGES

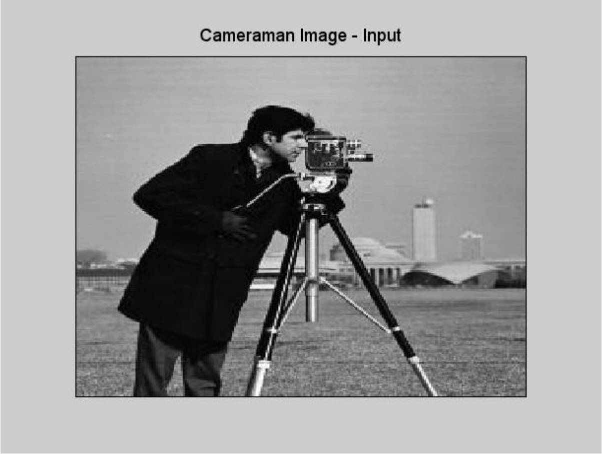
```
//special effects implementation of gray and color images
clc
clear()
xdel(winsid())
a=imread('path to mandrill image');
figure
imshow(a)
title('Mandrill Image - Input','fontsize',4);
b=imread(path to twozebras image');
figure
imshow(b)
title('Twozebras Image - Input','fontsize',4);
c=imread(path to cameraman image');
figure
imshow(c)
title('Cameraman Image - Input','fontsize',4);

//Applying Sobel filter
fi=fspecial('sobel')
a1=imfilter(a,fi)
b1=imfilter(b,fi)
c11=imfilter(c,fi)
figure
imshow(a1)
title('Sobel filtering on mandrill image','fontsize',4);
figure
imshow(b1)
title('Sobel filtering on Two zebras image','fontsize',4);
figure
imshow(c11)
title('Sobel filtering on Cameraman image','fontsize',4);

//image negative

// for gray image
[m,n]=size(c);
for i=1:m
    for j=1:n
        c1(i,j)=255-c(i,j);
    end

end
figure
imshow(c1)
title('Image Negative of Gray image','fontsize',4);
//For color image
[p q r]=size(b)
for i=1:r
```

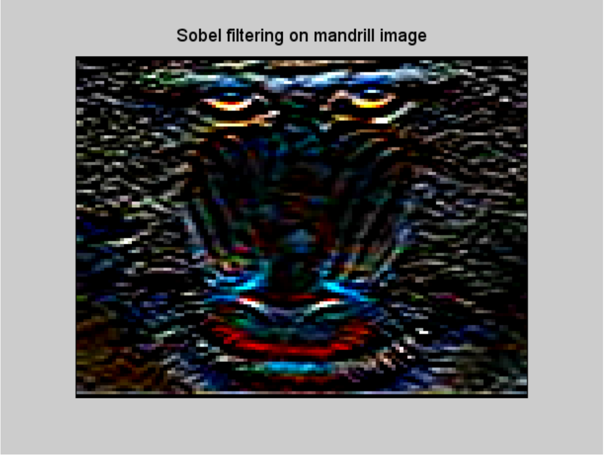
```
    for j=1:p
        for k=1:q
            b1(j,k,i)=255-b(j,k,i);
        end
    end
end
figure
imshow(b1)
title('Image Negative of color image','fontsize',4);
//image thresholding
for i=1:m
    for j=1:n
        if c(i,j)<150
            c2(i,j)=0;
        else
            c2(i,j)=255
        end
    end
end
figure
imshow(c2)
title('Threshold image','fontsize',4);
//image rotation
//mirror image
for i=1:m
    for j=1:n
        c3(i,j)=c(i,n-j+1)
    end
end
figure
imshow(c3)
title('Mirror Image','fontsize',4);
//clockwise rotation
c4=c';
c5=c4(:,n:-1:1);
figure
imshow(c5)
title('Clockwise rotated image','fontsize',4);
//anticlock wise rotation
c6=c4(m:-1:1,:)
figure
imshow(c6)
title('Anticlockwise rotated image','fontsize',4);
```

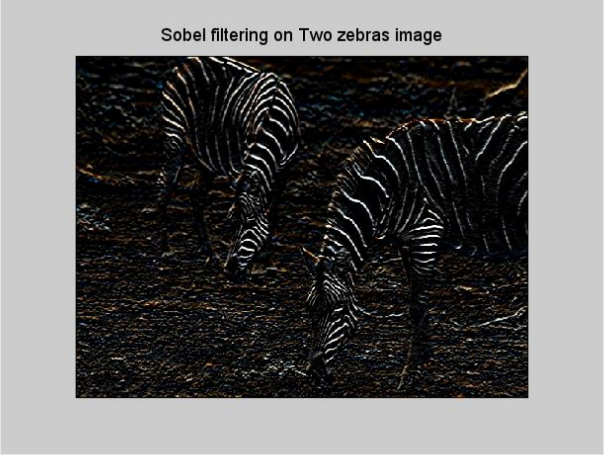
OUTPUT:

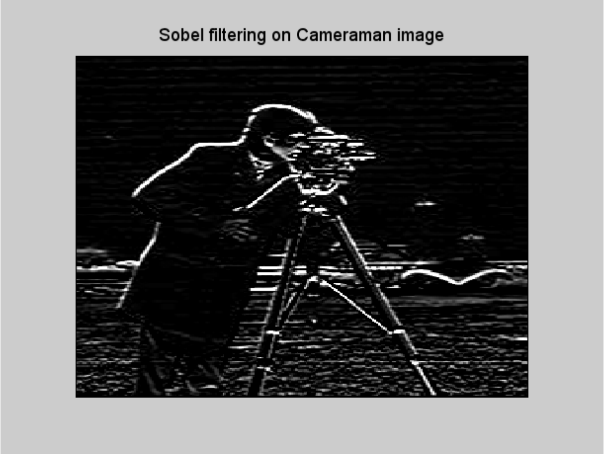

Mandrill Image - Input


Twozebras Image - Input


Cameraman Image - Input


Sobel filtering on mandrill image


Sobel filtering on Two zebras image


Sobel filtering on Cameraman image

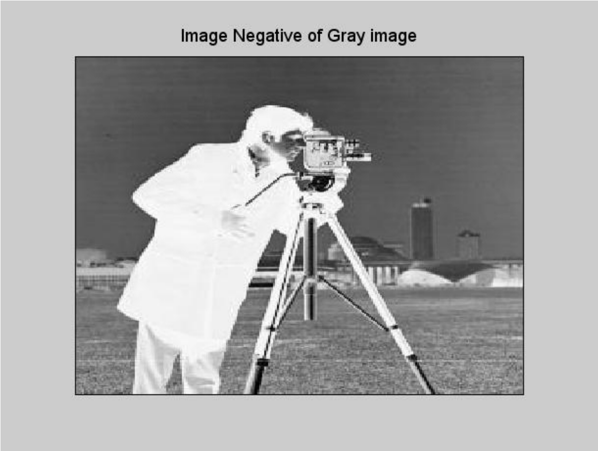


Image Negative of Gray image

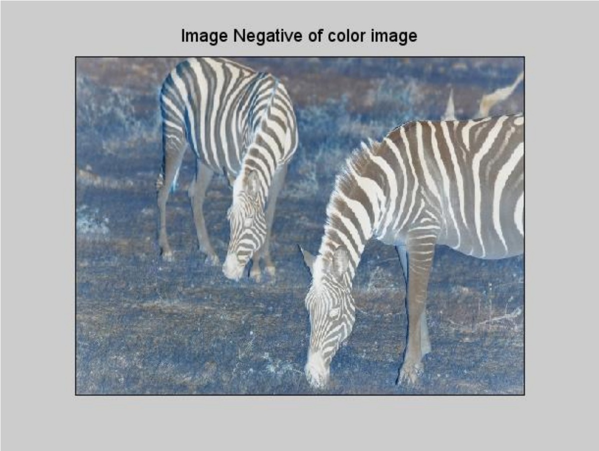
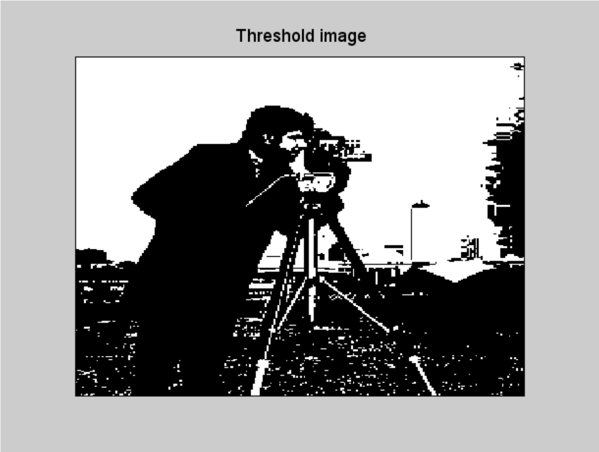
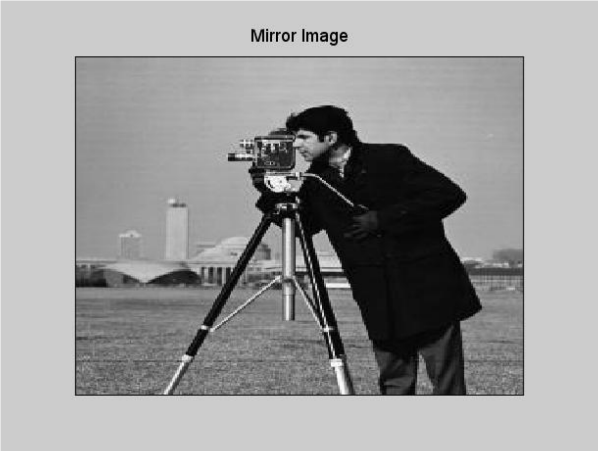
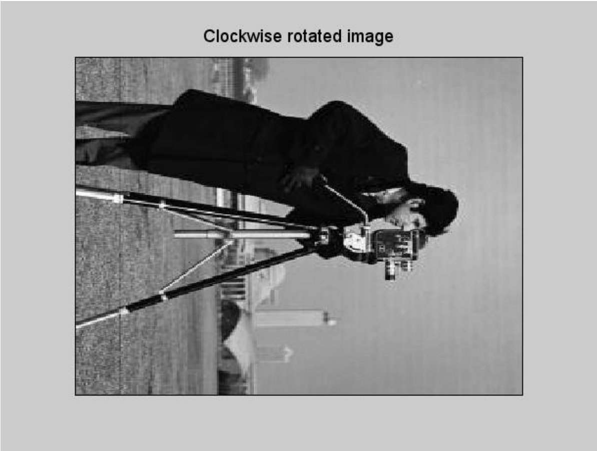
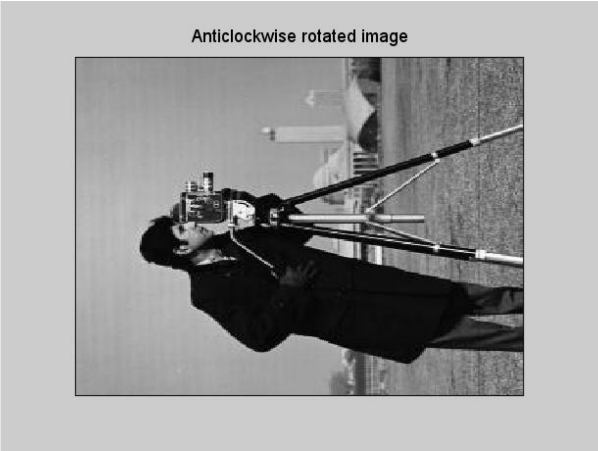


Image Negative of color image


Threshold image


Mirror Image

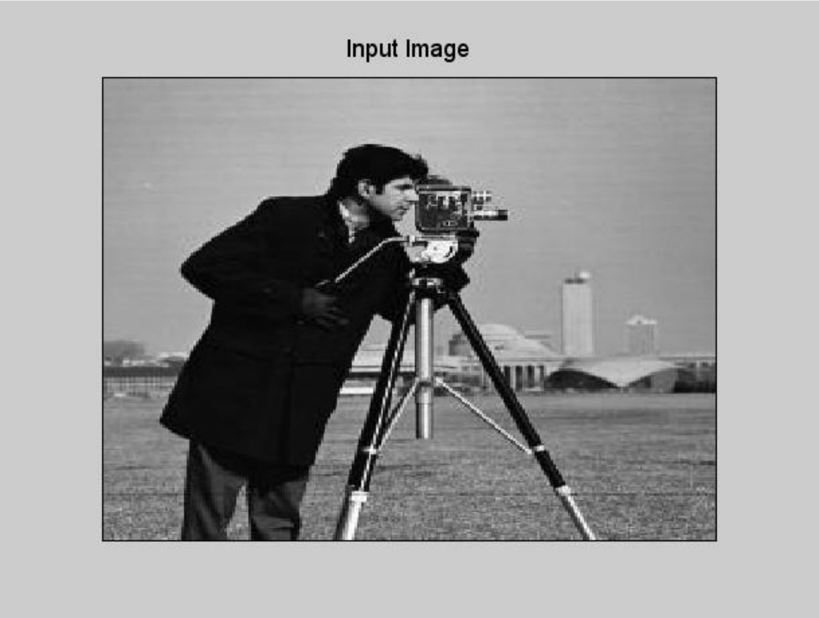

Clockwise rotated image

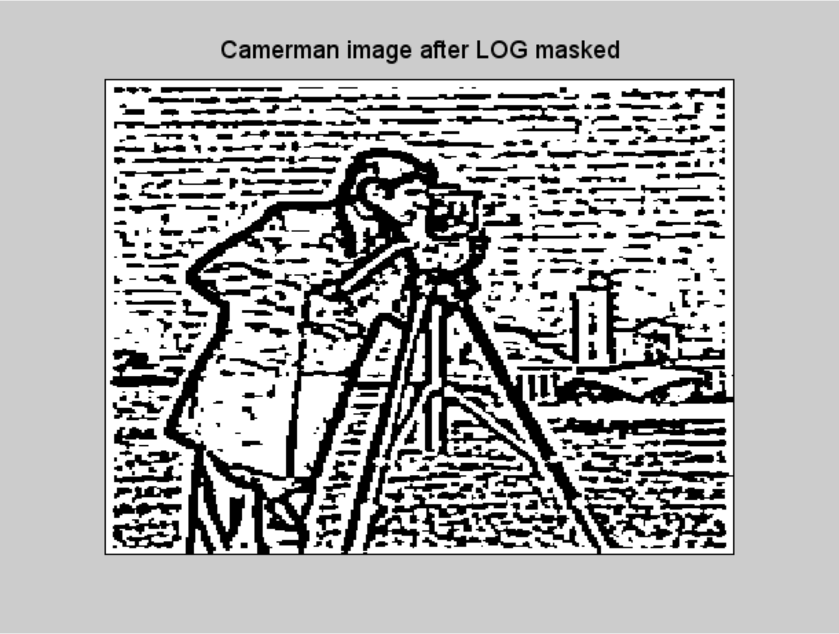

Anticlockwise rotated image

10. LOG MASK ON GRAY IMAGES

```
//LOG MASK Implementation

clc
clear()
xdel(winsid())
a= imread(path to cameraman.jpeg');
figure
imshow(a)
title('Input Image','fontsize',4);
a=double(a);
[m n]= size(a);

//Defining LOG mask coefficients with size 9x9
logmask=[0 1 1 2 2 2 1 1 0;1 2 4 5 5 5 4 2 1;1 4 5 3 0 3 5 4 1;2 5 3 -12 -24 -12 3 5 2;2 5 0 -24 -40 -24 0 5 2;2 5 3 -12 -24 -12 3 5 2;1 4 5 3 0 3 5 4 1;1 2 4 5 5 5 4 2 1;0 1 1 2 2 2 1 1 0];
//Adding rows and columns with zeros
[m1 n1 ]= size(logmask);
b= zeros(m+m1-1,n+n1-1) ;
m2=floor(m1/2);
n2=floor(n1/2);
b(m2+1:m+m2,n2+1:n+n2)=a;

//Applying LOG mask
for i=m2+1:m+m2
  for j=n2+1:n+n2
   c=b(i-m2:i+m2,j-n2:j+n2);
d= sum(sum(c.*logmask ));

//Applying Threshold to the mask
if d>150
e(i-m2,j-n2)=0;
else
e(i-m2,j-n2)=1;
end
end
end

figure
title('Camerman image after LOG masked','fontsize',4)
imshow(e)
```

OUTPUT:


Input Image


Camerman image after LOG masked