Negative Image

```
clc;
clear all;
close all;
a=imread("cameraman.tif");
for i=1:256
    for j=1:256
        m(i,j)=255-a(i,j);
    end
end
subplot(121);imshow(a);title("Original Image");
subplot(122);imshow(m);title("Negative image");
```

Black & White using threshold

```
clc;
clear all;
close all;
i1=imread("cameraman.tif");
b=im2bw(i1);
for i=1:256
    for j=1:256
        if i1(i,j)>128
            m(i,j)=1;
        else
            m(i,j)=0;
        end
    end
end
subplot(131);imshow(i1);title("Original Image");
subplot(132);imshow(b);title("B&W using function");
subplot(133);imshow(m);title("B&W without function");
```

RGB to B&W and Gray

```
clc;
clear all;
close all;
i1=imread("images/Tulips.jpg");
i2=im2bw(i1);
i3=rgb2gray(i1);
subplot(131);imshow(i1);title("Original image");
subplot(132);imshow(i2);title("Black & White image");
subplot(133);imshow(i3);title("Gray image");
```

Rotate Picture

```
clc;
clear all;
close all;
I = imread("images/cameraman.tif");
imshow(I);

I2=imrotate(I,45);
imshow(I2);
```

Separate Color Image

```
clc;
close all;
clear all;
i1=imread("images/Tulips.jpg");
r=i1(:,:,1);
g=i1(:,:,2);
b=i1(:,:,3);
m=cat(3,uint8(r),uint8(g),uint8(b));
subplot(221);imshow(m);title("Original Image");
subplot(222);imshow(r);title("Red Image");
subplot(223);imshow(g);title("Green Image");
subplot(224);imshow(b);title("Blue Image");
```

Different Pixel Add

```
clc;
clear all;
close all;
i1=imread("cameraman.tif");
add_10=imadd(i1,10);
add_50=imadd(i1,50);
add_100=imadd(i1,100);
add_200=imadd(i1,200);
add_300=imadd(i1,300);
subplot(231);imshow(i1);title("Original Image");
subplot(232);imshow(add_10);title("10 pixels");
subplot(233);imshow(add_50);title("50 pixels");
subplot(234);imshow(add_100);title("100 pixels");
subplot(235);imshow(add_200);title("200 pixels");
subplot(236);imshow(add_300);title("300 pixels");
```

Power Law Transformation

```
clc;
clear all;
close all;
i=imread("images/Tulips.jpg");
I=im2double(i);
r=size(I,1);
c=size(I,2);
for i=1:r
    for j=1:c
        I1(i,j)=I(i,j)^.5;
    end
end
for i=1:r
    for j=1:c
        I2(i,j)=I(i,j)^.1;
    end
end

for i=1:r
    for j=1:c
        I3(i,j)=I(i,j)^1.667;
    end
end
subplot(221);imshow(I);title("Original Image");
subplot(222);imshow(I1);title("Power law 0.5");
subplot(223);imshow(I2);title("Power law 0.1");
subplot(224);imshow(I3);title("Power law 1.667");
```

Log Transformation

```
clc;
clear all;
close all;
i1=imread("images/circuit.tif");
I=im2double(i1);
r=size(I,1);
c=size(I,2);
for i=1:r
    for j=1:c
        s(i,j) = log(1 + I(i,j));
    end
end
subplot(121);imshow(I);title("Original Image");impixelinfo;
subplot(122);imshow(s);title("Log Transformation Image");impixelinfo;
```

Binary Count

```
clc;
clear all;
close all;
a=imread("images/cameraman.tif");
I=im2bw(a);
b=imbinarize(a);
c=0;
for i=1:256
    for j=1:256
        if b(i,j)==1
            c=c+1;
        end
    end
end
subplot(131);imshow(a);title("Original Image");
subplot(132);imshow(I);title("Black and White Image");
subplot(133);imshow(b);title("Binary Image");
```

Bit plane slicing without loop

```
clc;
clear all;
close all;
a=imread("images/Tulips.jpg");
b=rgb2gray(a);
a=double(b);
b=bitget(a,1);
subplot(241);imshow(b);title("Bit plane 1");
b=bitget(a,2);
subplot(242);imshow(b);title("Bit plane 2");
b=bitget(a,3);
subplot(243);imshow(b);title("Bit plane 3");
b=bitget(a,4);
subplot(244);imshow(b);title("Bit plane 4");
b=bitget(a,5);
subplot(245);imshow(b);title("Bit plane 5");
b=bitget(a,6);
subplot(246);imshow(b);title("Bit plane 6");
b=bitget(a,7);
subplot(247);imshow(b);title("Bit plane 7");
b=bitget(a,8);
subplot(248);imshow(b);title("Bit plane 8");
```

Bit plane slicing with loop

```
clc;
clear all;
close all;
a=imread("images/Tulips.jpg");
b=rgb2gray(a);
a=double(b);
[r,c]=size(a);
for i=1:r
    for j=1:c
        i1(i,j)=bitand(a(i,j),1);
        i2(i,j)=bitand(a(i,j),2);
        i3(i,j)=bitand(a(i,j),4);
        i4(i,j)=bitand(a(i,j),8);
        i5(i,j)=bitand(a(i,j),16);
        i6(i,j)=bitand(a(i,j),32);
        i7(i,j)=bitand(a(i,j),64);
        i8(i,j)=bitand(a(i,j),128);
    end
end
subplot(241);imshow(i1);title("Bitplane 1");
subplot(242);imshow(i2);title("Bitplane 2");
subplot(243);imshow(i3);title("Bitplane 3");
subplot(244);imshow(i4);title("Bitplane 4");
subplot(245);imshow(i5);title("Bitplane 5");
subplot(246);imshow(i6);title("Bitplane 6");
subplot(247);imshow(i7);title("Bitplane 7");
subplot(248);imshow(i8);title("Bitplane 8");
```

Max Filtering

```
clc;
close all;
clear all;

a = imread("cameraman.tif");
[Mi, Ni] = size(a);

w = ones(3,3);
[m, n] = size(w);

f = padarray( a, [m-1, n-1] );
[M, N] = size(f);

filter_img = zeros(M,N);

sM = (m+1)/2;
sN = (n+1)/2;

eM = sM - 1;
eN = sN - 1;

w = w';
w1 = w(:);
[r, c] = size(w1);

for i = sM:M-eM
    for j = sN:N-eN
        array = zeros(r,c);
        iw = 1;
        for ii = i-eM:i+eM
            for jj = j-eN:j+eN
                array(iw) = w1(iw) * f(ii, jj);
                iw = iw + 1;
            end
        end
        filter_img(i,j) = ceil(max(array));
    end
end
filter_img = filter_img( m:m+Mi-1 , n:n+Ni-1);
subplot(121);imshow(a);title("Original Image");
subplot(122);imshow(uint8(filter_img));title("Original Image");
```

Sharpening Image

```
clc;
clear all;
close all;

a=im2double(imread("images/moon.tif"));
lap=[-1 -1 -1; -1 8 -1;-1 -1 -1];
resp=imfilter(a,lap,'conv');
minR= min(resp(:));
maxR= max(resp(:));
resp= (resp-minR)/(maxR-minR);

sharp= a+resp;
minS= min(sharp(:));
maxS= max(sharp(:));
sharp= (sharp-minS)/(maxS-minS);
sharp= imadjust(sharp, [60/255 200/255], [0 1]);

subplot(131);imshow(a);title("Original Image");
subplot(132);imshow(resp);title("Laplacian Filter Image");
subplot(133);imshow(sharp);title("Sharpned Image");
```

a=imread("cameraman.tif");
b=imsharpen(a);

Laplacian Filter Using Function

```
clc;
clear all;
close all;

img = imread("images/moon.tif");

g = [0 1 0; 1 -4 1; 0 1 0];

lap = imfilter(img, g);

sharpened_image = img - lap;
subplot(131);imshow(img);title('Original Image');
subplot(132);imshow(lap);title('Laplacian Filtered Image');
subplot(133);imshow(sharpened_image);title('Sharpened Image');
```

Fourier Transform

```
clc;
clear all;
close all;
warning off;

a=imread("images/letter.png");

f1=fftshift(fft2(a));

f2=log(1+abs(f1));
fm=max(f2(:));
f2=im2uint8(f2/fm);

f3=mat2gray(log(1+abs(f1)));

subplot(221);imshow(a);title("Original Image");
subplot(222);imshow(f1);title("Fourier Transformed Image With shifting");
subplot(223);imshow(f2);title("Fourier Transformed Image Manual Scaling");
subplot(224);imshow(f3);title("Fourier Transformed Image Automatic Scaling");
```

Ideal LowPass Filter

```
clc;
clear all;
close all;

a=imread("images/cameraman.tif");
[m,n]=size(a);
fi=fftshift(fft2(a));
c=30;
[u,v]=meshgrid(-floor(m/2):floor(m-1)/2, -floor(n/2):floor(n-1)/2);
z=sqrt(u.^2+v.^2);
ideal_filter=z<c;
o1=fi.*ideal_filter;
o2=ifft2(o1);
o3=mat2gray(abs(o2));

subplot(221);imshow(a);title("Original Image");
subplot(222);imshow(ideal_filter);title("Radius Image");
subplot(223);surf(z);title("3D Image");
subplot(224);imshow(o3);title("Ideal Low Pass Filtering Image");
```

Butterworth Low Pass Filter

```
clc;
clear all;
close all;

img = imread('cameraman.tif');

[M, N] = size(img);
f_img = fftshift(fft2(img));

C0 = 60; % Cut Off Frequency
n = 2;   % Order of the filter

[u, v] = meshgrid(-floor(M/2):floor(M-1)/2, -floor(N/2):floor(N-1)/2);
D = sqrt(u.^2 + v.^2);

butterworth_filter = 1 ./ (1 + (D./C0).^(2*n));    %low pass
```

```
output_img = f_img .* butterworth_filter;
output_img1 = ifft2(output_img);
output_img2 = mat2gray(abs(output_img1));

subplot(221);imshow(img);title("Original Image");
subplot(222);imshow(butterworth_filter);title("Radius Image");
subplot(223);imshow(output_img2);title("BLF Image");
subplot(224);surf(butterworth_filter);title("3D Image");
```

Gaussian Low Pass Filter

```
clc;
clear all;
close all;

img = imread('cameraman.tif');

[M, N] = size(img);
f_img = fftshift(fft2(img));

C0 = 30; % Cut Off Frequency

[u, v] = meshgrid(-floor(M/2):floor((M-1)/2), -floor(N/2):floor((N-1)/2));
D = sqrt(u.^2 + v.^2);

g_filter = exp(-D.^2/(2*C0.^2));    % Gaussian Low Pass Filter
output_img = f_img .* g_filter;
output_img1 = ifft2(ifftshift(output_img));
output_img2 = mat2gray(abs(output_img1));

subplot(221);imshow(img);title("Original Image");
subplot(222);imshow(log(1 + abs(fftshift(g_filter))),[]);title("Filter Spectrum");
subplot(223);imshow(output_img2);title("Filtered Image");
subplot(224);surf(u, v, g_filter);title("3D Filter");
```

Arithmetic Mean

```
clc;
clear all;
close all;

img = imread('cameraman.tif');
b=imnoise(img,'gaussian');    %add gaussian noise
b1=imnoise(img,'salt & pepper',0.001);   %add salt & pepper noise

w=ones(3,3);

[Mi,Ni] = size(b);
[m,n] = size(w);

f = padarray(b,[m-1 n-1]);
[M,N] = size(f);
filtered_img = zeros(M,N);

sM = (m+1)/2;
sN = (n+1)/2;

eM = sM-1;
eN = sN-1;

w = w';
w1 = w(:);

[r,c] = size(w1);

for i = sM:M-eM
    for j = sN:N-eN
        array = zeros(r,c);
        iw=1;
        for ii = i-eM:i+eM
            for jj = j-eN:j+eN
                array(iw) = f(ii,jj);
                iw = iw+1;
            end
        end
        filtered_img(i, j) = mean(array);
    end
end
```

```
filtered_img = filtered_img(m:m+Mi-1, n:n+Ni-1);
subplot(221);imshow(img);title('Original Image');
subplot(222);imshow(b);title('Noisy Image using gaussian noise');
subplot(223);imshow(b1);title('Noisy Image using salt & pepper noise');
subplot(224);imshow(uint8(filtered_img));title('Arithmetic Mean Filter');
```

Contra harmonic Mean

```
for i = sM:M-eM
    for j = sN:N-eN
        array = zeros(r,c);
        iw=1;
        for ii = i-eM:i+eM
            for jj = j-eN:j+eN
                array(iw) = f(ii,jj);
                iw = iw+1;
            end
        end
        filtered_img(i, j) = mean(array);
    end
end
```

Geometric Mean

```
for i = sM:M-eM
    for j = sN:N-eN
        array = zeros(r,c);
        iw=1;
        sum=1;
        for ii = i-eM:i+eM
            for jj = j-eN:j+eN
                array(iw) = w1(iw)*f(ii,jj);
                sum=sum*array(iw);
                iw = iw+1;
            end
        end
        filtered_img(i, j)=ceil(sum^(1/9));
    end
end
```

Harmonic Mean

```
for i = sM:M-eM
    for j = sN:N-eN
        array = zeros(r,c);
        iw=1;
        sum=0;
        for ii = i-eM:i+eM
            for jj = j-eN:j+eN
                array(iw) = w1(iw)*f(ii,jj);
                h=1/array(iw);
                sum=sum+h;
                iw = iw+1;
            end
        end
        h1=ceil(9/sum);
        filtered_img(i,j)=h1;
    end
end
```

Histogram Basic

```
clc;
clear all;
close all;
Img = imread("images/cameraman.tif");
%b = rgb2gray(Img);

subplot(2,2,1);
imshow(Img);
subplot(2,2,2);

 imhist(Img);
b = imhist(Img);
arr = zeros(1,256);
[r,c] = size(Img);
%plot(b);
for i=1:r
    for j=1:c
        arr(Img(i,j))=arr(Img(i,j))+1;
    end
end
disp(arr);
subplot(2,2,3);
bar(b);
%subplot(2,2,2)
```

Histogram Equalization

```
clc;
clear all;
close all;
img=imread("cameraman.tif");
total=0;

for i=1:256
   M(2,i)=0;
end

for l=1:256
    M(1,l)=l-1;
for i=1:256
  for j=1:256
      if img(i,j)==l-1
          M(2,l)=M(2,l)+1;
      end
  end
end
total=total+M(2,l);
end

PDF=M(2,:)/total;
c=0;
for k=1:256
   cdf(k)=c+PDF(1,k);
   c=cdf(k);
end
for k=1:256
```

```
   h(k)=255*cdf(k);
   y(k)=round(h(k));

end

for l=1:256
for i=1:256
  for j=1:256
      if img(i,j)==l-1
          img2(i,j)=y(l);
      end
  end
end
end

img2=uint8(img2);

subplot(2,2,1);imshow(img);title("Before Histogram");
subplot(2,2,2);imhist(img);title("Before Histogram");
subplot(2,2,3);imshow(img2);title("After Histogram Equalization(Without function)");
subplot(2,2,4);imhist(img2);title("After Histogram Equalization(Without function)");
```

Histogram Equalization

```
clc;
clear all;
close all;

a=imread("eight.tif");
r=size(a,1);               %row
c=size(a,2);               %column
ah=uint8(zeros(r,c));      %final value
n=r*c;                     %total number of pixels
f=zeros(256,1);            %frequency(variable....intensity levels 256)
pdf=zeros(256,1);
cdf=zeros(256,1);
cum=zeros(256,1);
out=zeros(256,1);
```

```
for i=1:r
    for j=1:c
        value=a(i,j);
        f(value+1)=f(value+1)+1;
        pdf(value+1)=f(value+1)/n;
    end
end

sum=0;L=255;
for i=1:size(pdf)
    sum=sum+f(i);
    cum(i)=sum;
    cdf(i)=cum(i)/n;
    out(i)=round(cdf(i)*L);
end

for i=1:r
    for j=1:c
        ah(i,j)=out(a(i,j)+1);
    end
end
subplot(121);imshow(a);title("Original Image");

he=histeq(a);
subplot(122);imshow(he);title("Histogram Equalization Image");
```

Project

```
clc;
close all;
clear all;

% Read a Color Image
Img = imread('images/Tulips.jpg');

% Separate each channel
redChannel = Img(:, :, 1);
greenChannel = Img(:, :, 2);
blueChannel = Img(:, :, 3);

% Combine channels
Img = cat(3, redChannel, greenChannel, blueChannel);

% Apply Gaussian Low Pass Filter to each channel
sigma = 1.15;  % Standard Deviation
sz = 3;        % Window size
[x, y] = meshgrid(-sz:sz, -sz:sz);
M = size(x, 1) - 1;
N = size(y, 1) - 1;

% Create Gaussian Kernel
Exp_comp = -(x.^2 + y.^2) / (2 * sigma * sigma);
Kernel = exp(Exp_comp) / (2 * pi * sigma * sigma);

% Initialize filtered channels
filteredRed = zeros(size(redChannel));
filteredGreen = zeros(size(greenChannel));
filteredBlue = zeros(size(blueChannel));

% Pad the channels with zeros
r = padarray(double(redChannel), [sz, sz]);
g = padarray(double(greenChannel), [sz, sz]);
b = padarray(double(blueChannel), [sz, sz]);

% Convolution for each channel
for i = 1:size(r, 1) - M
    for j = 1:size(r, 2) - N
        % Apply Gaussian filter
        TempRed = r(i:i+M, j:j+M) .* Kernel;
        TempGreen = g(i:i+M, j:j+M) .* Kernel;
        TempBlue = b(i:i+M, j:j+M) .* Kernel;
        % Sum the values
        filteredRed(i, j) = sum(TempRed(:));
        filteredGreen(i, j) = sum(TempGreen(:));
        filteredBlue(i, j) = sum(TempBlue(:));
    end
end

% Combine filtered channels
filteredImg = cat(3, uint8(filteredRed), uint8(filteredGreen), uint8(filteredBlue));

% Display the results
subplot(331); imshow(Img);title('Original Image');

subplot(332); imshow(redChannel);title('Red Channel Image before filtering');
subplot(335); imshow(greenChannel);title('Green Channel Image before filtering');
subplot(338); imshow(blueChannel);title('Blue Channel Image before filtering');
subplot(334); imshow(Img);title('Combine image before filtering');

subplot(333); imshow(uint8(filteredRed));title('Red channel image after filtering');
subplot(336); imshow(uint8(filteredGreen));title('Green channel image after filtering');
subplot(339); imshow(uint8(filteredBlue));title('Blue channel image after filtering');

subplot(337); imshow(filteredImg);title('Combine image after filtering');
```