Lab Report 5

Aim

To apply frequency domain filters for image enhancement in MATLAB.

Theory

In image processing, frequency domain filters are used to modify or enhance the frequency content of an image. An image can be represented in the frequency domain using the Fourier transform, which decomposes the image into its constituent frequencies. Frequency domain filters in image processing can be broadly classified into two categories: low-pass filters and high-pass filters. Low-pass filters attenuate high-frequency components in the image, while high-pass filters attenuate low-frequency components. Low-pass filters are commonly used in image smoothing, where they are used to remove high-frequency noise from the image while preserving the lower-frequency content. The most common type of low-pass filter used in image processing is the Gaussian filter, which attenuates high-frequency components according to a Gaussian distribution. High-pass filters, on the other hand, are used to enhance the high-frequency content in an image, making the edges and details of the image more visible. Commonly used high-pass filters in image processing include the Laplacian filter, the Sobel filter, and the Canny filter. Other types of frequency domain filters used in image processing include band-pass filters, which allow a certain range of frequencies to pass through while attenuating other frequencies, and band-stop filters, which attenuate a certain range of frequencies while passing other frequencies. Frequency domain filters in image processing are useful in a wide range of applications, including medical imaging, remote sensing, and industrial inspection, where image quality is critical and noise or interference can have a significant impact on the results.

LOW PASS AND HIGH PASS FILTER

Low Pass Filter:

In image processing, low-pass filters are used to remove high-frequency noise and to smooth images by attenuating high-frequency components while preserving low-frequency components. The most common type of low-pass filter used in image processing is the Gaussian filter. The Gaussian filter is a frequency domain filter that attenuates high-frequency components according to a Gaussian distribution. It is a popular choice for image smoothing because it has a simple mathematical form, is easy to implement, and produces visually pleasing results. The Gaussian filter works by convolving the image with a Gaussian kernel, which is a two-dimensional bell-shaped curve centred at the origin. The size of the kernel and the standard deviation of the Gaussian distribution determine the degree of smoothing and the amount of detail preserved in the image. Another commonly used low-pass filter in image processing is the mean filter, which replaces each pixel in the image with the average of its neighbouring pixels. The size of the neighbourhood or the window used for averaging determines the degree of smoothing, with larger neighbourhoods resulting in greater smoothing and more detail loss. Low-pass filters in image processing can be used to reduce noise, blur, or hide details in an image. They are commonly used in applications such as image denoising, feature extraction, and image segmentation. It is important to note that excessive smoothing can result in loss of important details and edges in the image. Therefore, the choice of the filter and the parameters used for filtering should be carefully tuned to achieve the desired level of smoothing while preserving important features in the image.

High Pass Filter:

In image processing, high-pass filters are used to enhance the high-frequency content of an image by attenuating the low-frequency components. High-pass filters are commonly used for edge detection and sharpening of images. The most common type of high-pass filter used in image processing is the Laplacian filter. The Laplacian filter enhances edges and details in an image by detecting areas where the brightness changes rapidly. It does this by convolving the image with the second derivative of the Gaussian function, which is a measure of the rate of change of the image intensity. Another commonly used high-pass filter is the Sobel filter, which is used for edge detection in images. The Sobel filter works by computing the gradient of the image intensity in the x and y directions, and then combining these gradients to obtain a measure of the edge strength. Other types of high-pass filters used in image processing include the Prewitt filter, the Roberts cross filter, and the Canny filter. Each of these filters has its own advantages and disadvantages, and the choice of filter depends on the specific application and the desired outcome. High-pass filters can be useful in a wide range of applications, including medical imaging, industrial inspection, and remote sensing, where image detail and clarity are critical. However, it is important to note that excessive filtering can result in the loss of important image information, such as subtle details and textures. Therefore, the choice of filter and the parameters used for filtering should be carefully tuned to achieve the desired level of enhancement while preserving important image features.

CODE

```
% LOW PASS FILTER

fprintf("PLEASE SELECT AN IMAGE\n")
i=uigetfile('*.*');
j=imread(i)
k=rgb2gray(j);
imshow(k)
[r,c]=size(k)
l=fft2(k);
m=fftshift(l);
n=abs(m);
o=log(1+n);
imshow(o,[]);
Z=zeros(r,c);
for R=1:r
    for C=1:c
        if (R-r/2)^2+(C-c/2)^2 <= 50^2
                    Z(R,C)=1;
```

```
            end
        end
end
imshow(Z)
new=m.*Z;
new1=fftshift(new);
new2=ifft2(new1);
imshow(abs(new2),[]);
edge=abs(new2);
sharp=double(k)+edge;
imshow(sharp,[]);
```

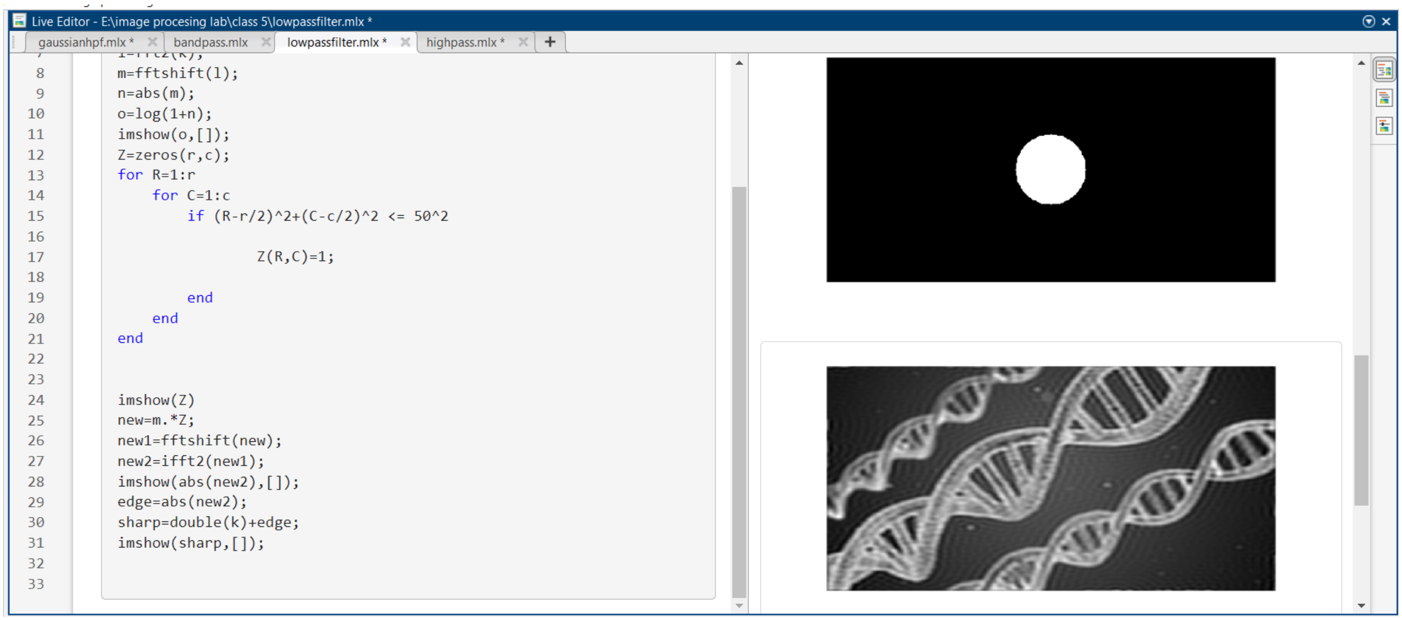

Figure 1: use of low pass filter to smoothen image in MATLAB

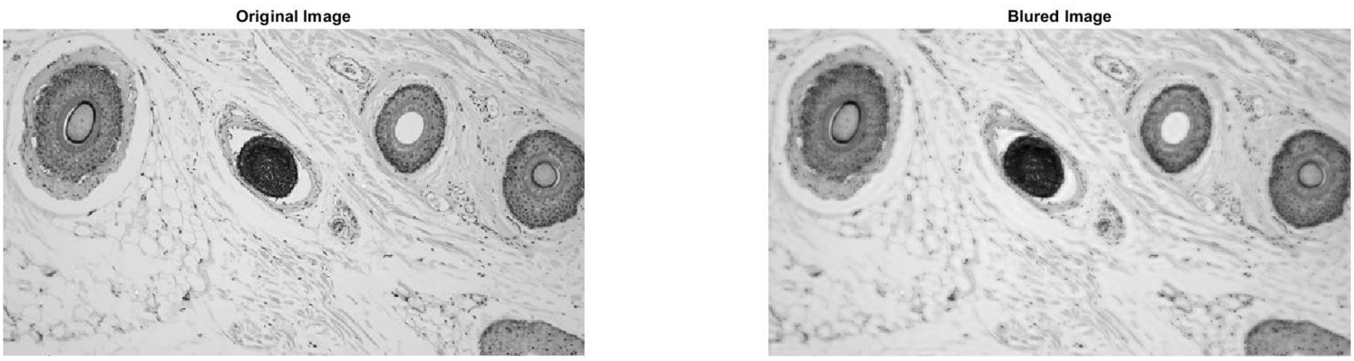

Figure 2: image smoothening using low pass filter in MATLAB

CODE

```
% HIGH PASS FILTER
fprintf("PLEASE SELECT AN IMAGE\n")
i=uigetfile('*.*');
j=imread(i)
k=rgb2gray(j);
imshow(k)
[r,c]=size(k)
l=fft2(k);
m=fftshift(l);
n=abs(m);
o=log(1+n);
imshow(o,[]);
Z=zeros(r,c);
for R=1:r

    for C=1:c

        if (R-r/2)^2+(C-c/2)^2 >= 50^2

        Z(R,C)=1;

        end

    end

end
imshow(Z)
new=m.*Z;
new1=fftshift(new);
new2=ifft2(new1);
imshow(abs(new2),[]);
edge=abs(new2);
sharp=double(k)+edge;
imshow(sharp,[]);
```

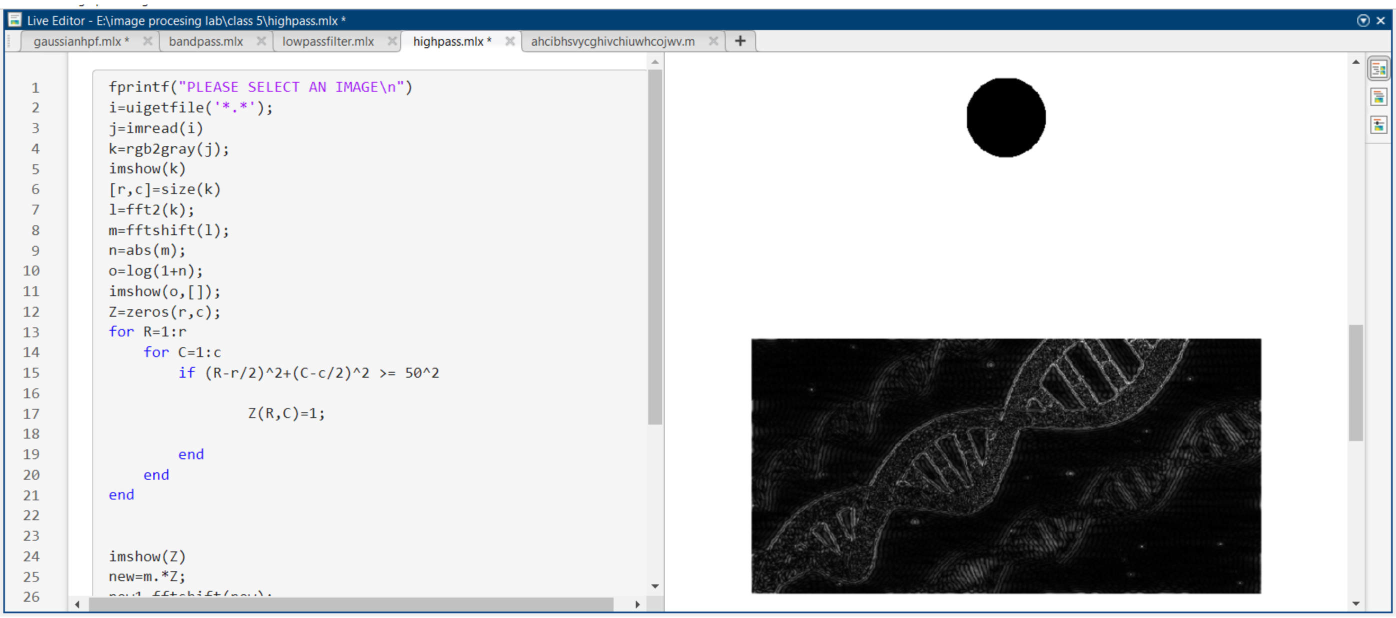

Figure 3: use of high pass filter to detect the edges in MATLAB

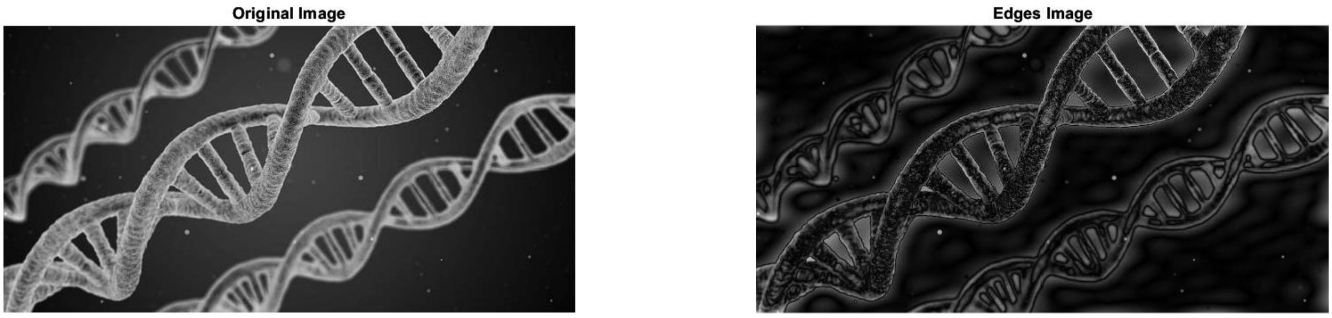

Figure 4: image edge detection using high pass filter in MATLAB

GAUSSIAN FILTERS

Gaussian Filter:

A Gaussian filter is a popular image processing technique used for smoothing or blurring an image by reducing the high-frequency components in the image. The filter is based on the Gaussian function, which is a bell-shaped curve that describes the probability distribution of a random variable.

In image processing, the Gaussian filter works by convolving each pixel of the image with a 2D Gaussian kernel, which is a matrix of values that represents the Gaussian function. The kernel size and standard deviation of the Gaussian function can be adjusted to control the amount of smoothing applied to the image.

The Gaussian filter is commonly used in applications such as noise reduction, edge detection, and feature extraction. It is particularly useful for removing high-frequency noise from an image while preserving the edges and other important details.

Overall, the Gaussian filter is a simple and effective way to enhance the visual quality of images in a variety of applications.

CODE

```
% GAUSSIAN HIGH PASS FILTER

fprintf("PLEASE SELECT AN IMAGE\n")
i=uigetfile('*.*');
j=imread(i)
k=rgb2gray(j);
imshow(k)
[r,c]=size(k)
l=fft2(k);
m=fftshift(l);
n=abs(m);
o=log(1+n);
imshow(o,[]);
Z=zeros(r,c);
d0=5;
for R=1:r
    for C=1:c
        d=sqrt((R-(r/2))^2+(C-(c/2))^2);
        Z(R,C)=1-exp(-(d^2)/(2*(d0)^2));
    end
end



imshow(Z)
new=m.*Z;
new1=fftshift(new);
new2=ifft2(new1);
imshow(abs(new2),[]);
edge=abs(new2);
```

```
sharp=double(k)+edge;
subplot(1,2,1); imshow(sharp,[]); title('Sharpened Image');

subplot(1,2,2); imshow(k,[]); title('Original Image');
```

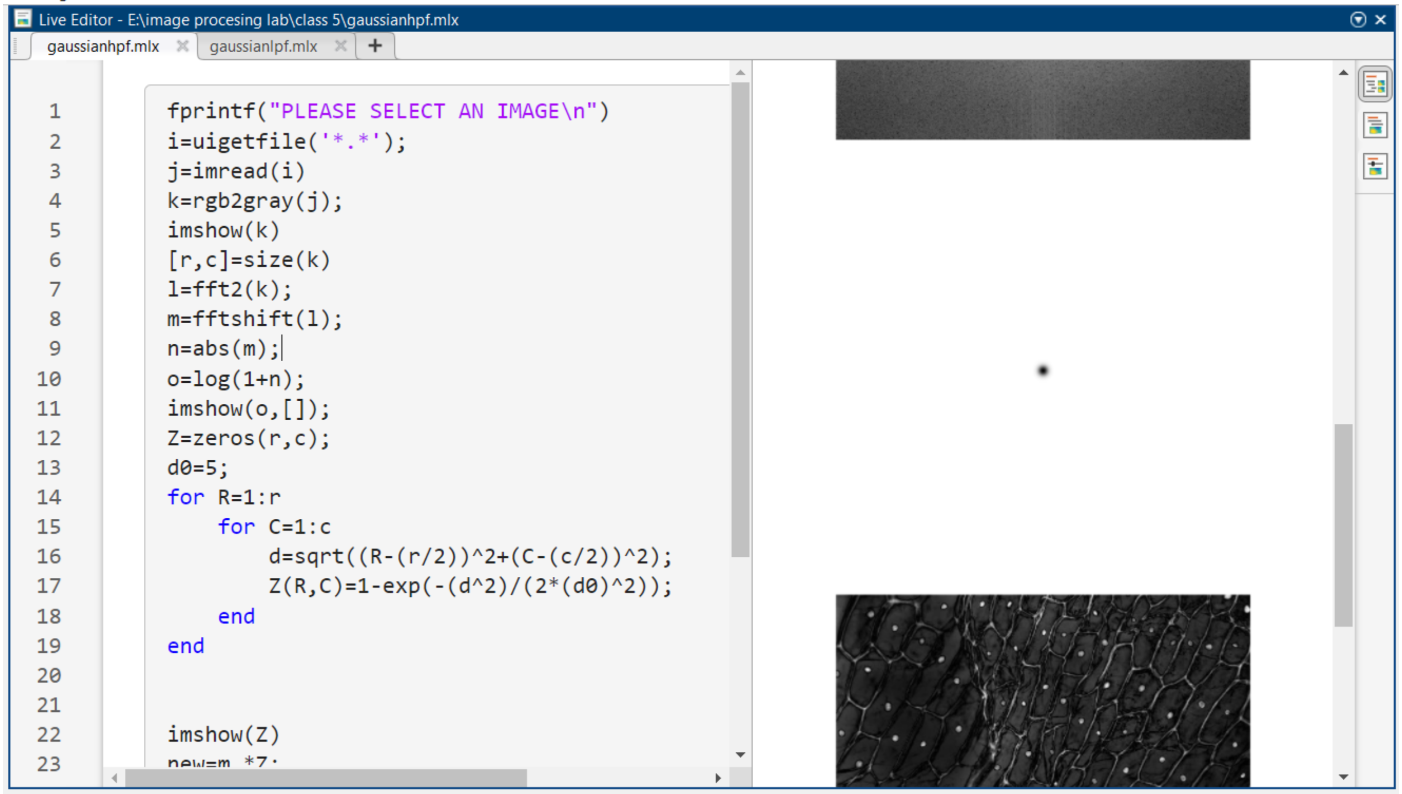

Figure 5: use of gaussian high pass filter in MATLAB

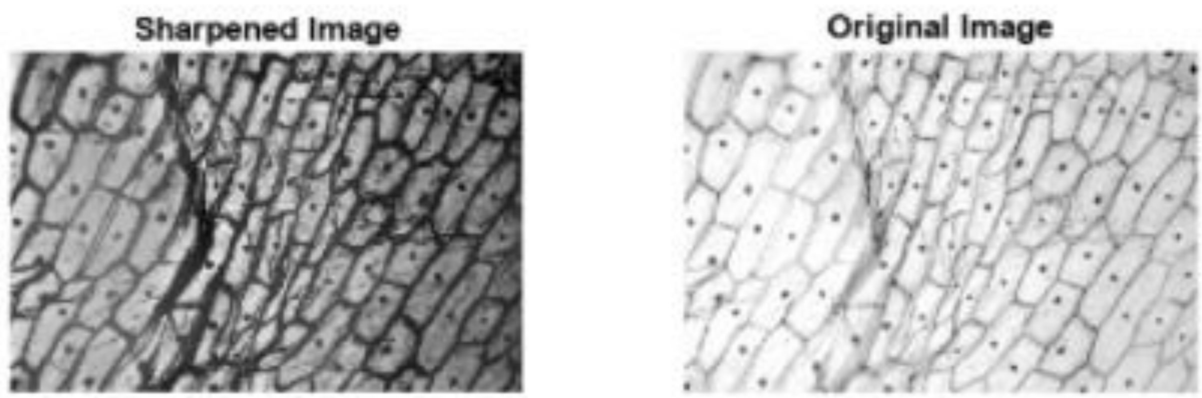

Figure 6: image processing using high pass filter in MATLAB

CODE

```
% GAUSSIAN LOW PASS FILTER
```

```
fprintf("PLEASE SELECT AN IMAGE\n")
i=uigetfile('*.*');
j=imread(i)
k=rgb2gray(j);
imshow(k)
[r,c]=size(k)
l=fft2(k);
m=fftshift(l);
n=abs(m);
o=log(1+n);
imshow(o,[]);
Z=zeros(r,c);
d0=50;
for R=1:r
    for C=1:c
        d=sqrt((R-(r/2))^2+(C-(c/2))^2);
        Z(R,C)=exp(-(d^2)/(2*(d0)^2));
    end
end


imshow(Z)
new=m.*Z;
new1=fftshift(new);
new2=ifft2(new1);
imshow(abs(new2),[]);
edge=abs(new2);
sharp=double(k)+edge;
subplot(1,2,1); imshow(sharp,[]); title('Sharpened Image');

subplot(1,2,2); imshow(k,[]); title('Original Image');
```

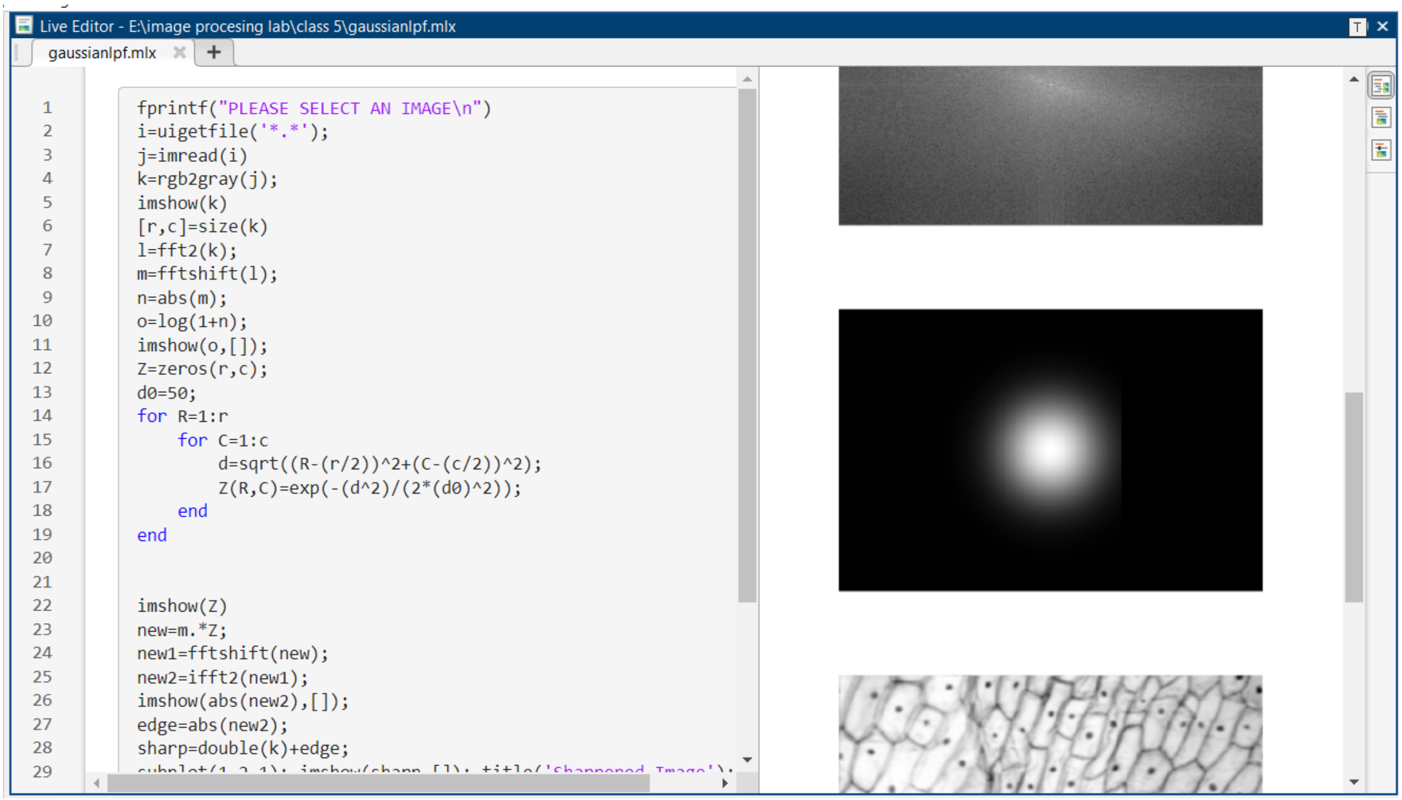

Figure 7: use of gaussian high low filter in MATLAB

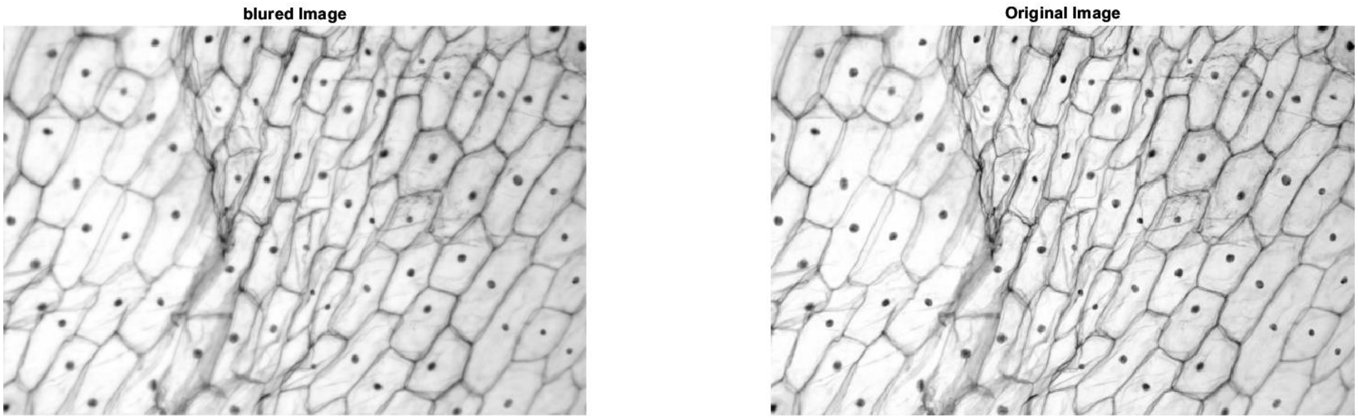

Figure 8: image procesing using gaussian low pass filter in MATLAB

BAND FILTER

Band Pass and Band Reject Filter:

In image processing, a band-pass filter is a type of filter that allows a specific range of frequencies to pass through while attenuating (reducing) frequencies outside that range. Similarly, a band-reject filter, also known as a notch filter, attenuates a specific range of frequencies while allowing frequencies outside that range to pass through.

A band-pass filter is often used to extract certain features or details from an image that exist in a particular frequency range. For example, a band-pass filter may be used to enhance the edges or texture of an image, or to remove certain types of noise that exist in a specific frequency range.

A band-reject filter, on the other hand, is often used to remove unwanted features or artifacts from an image. For example, a band-reject filter may be used to remove periodic noise or interference that exists in a certain frequency range.

Both band-pass and band-reject filters can be implemented using a variety of techniques, such as Fourier transforms or digital signal processing algorithms. The specific parameters of the filter, such as the cutoff frequencies or filter order, can be adjusted to achieve the desired filtering effect.

CODE

```
% BAND PASS FILTER

fprintf("PLEASE SELECT AN IMAGE\n")
i=uigetfile('*.*');
j=imread(i)
k=rgb2gray(j);
imshow(k)
[r,c]=size(k)
l=fft2(k);
m=fftshift(l);
n=abs(m);
o=log(1+n);
imshow(o,[]);
Z=zeros(r,c);
for R=1:r
    for C=1:c
        if (R-r/2)^2+(C-c/2)^2 >= 20^2
            if (R-r/2)^2+(C-c/2)^2 <= 40^2
                Z(R,C)=1;
            end
        end
    end
end
```

```
imshow(Z)
new=m.*Z;
new1=fftshift(new);
new2=ifft2(new1);
imshow(abs(new2),[]);
edge=abs(new2);
sharp=double(k)+edge;
imshow(sharp,[]);
```

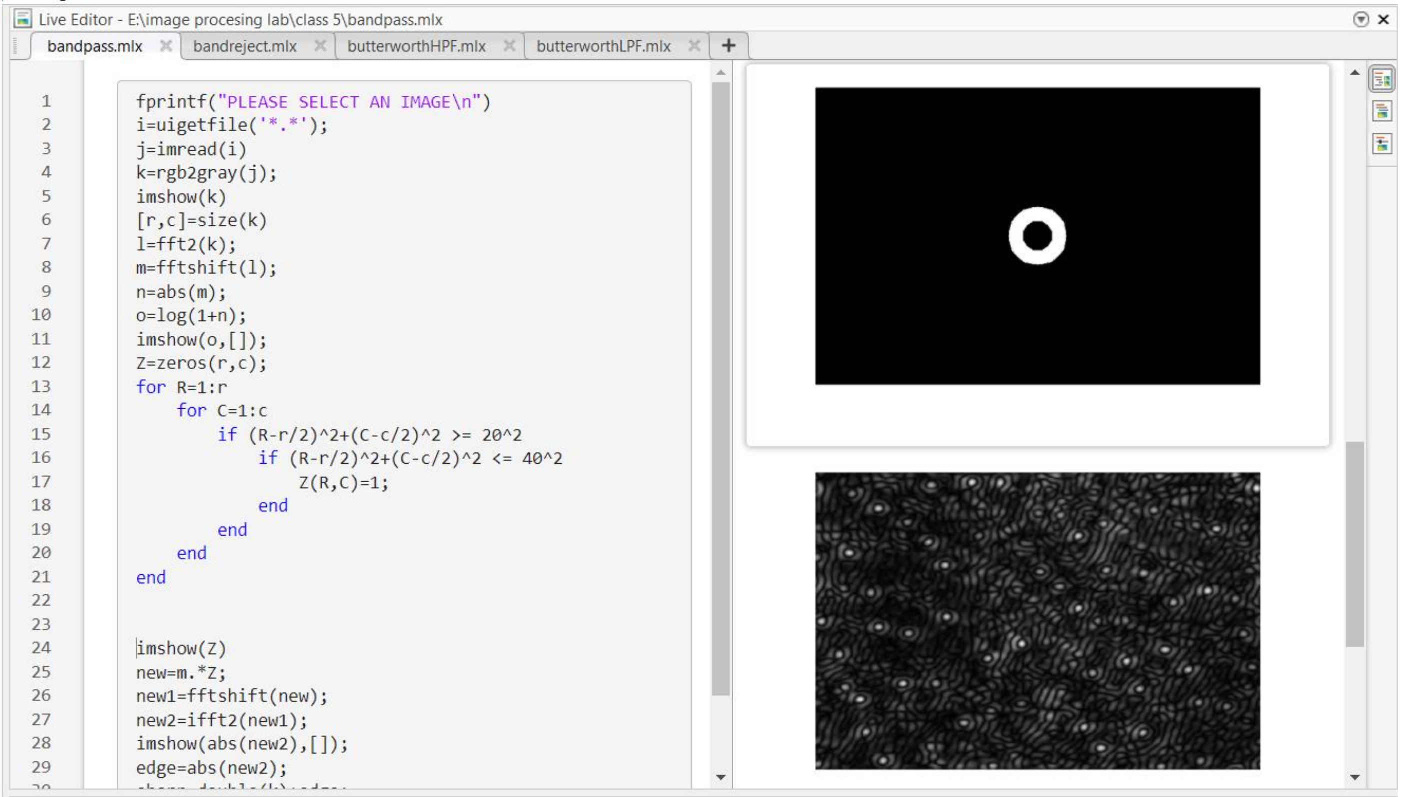

Figure 9: use of image band pass filter in MATLAB

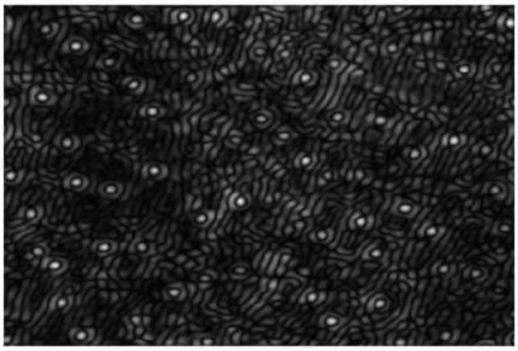

Figure 10: image processing using band pass filter in MATLAB

CODE

```
% BAND REJECT FILTER

fprintf("PLEASE SELECT AN IMAGE\n")
i=uigetfile('*.*');
j=imread(i)
k=rgb2gray(j);
imshow(k)
[r,c]=size(k)
l=fft2(k);
m=fftshift(l);
n=abs(m);
o=log(1+n);
imshow(o,[]);
Z=ones(r,c);
for R=1:r
    for C=1:c
        if (R-r/2)^2+(C-c/2)^2 >= 40^2
            if (R-r/2)^2+(C-c/2)^2 <= 80^2
                Z(R,C)=0;
            end
        end
    end
end


imshow(Z)
new=m.*Z;
new1=fftshift(new);
new2=ifft2(new1);
imshow(abs(new2),[]);
edge=abs(new2);
sharp=double(k)+edge;
imshow(sharp,[]);
```

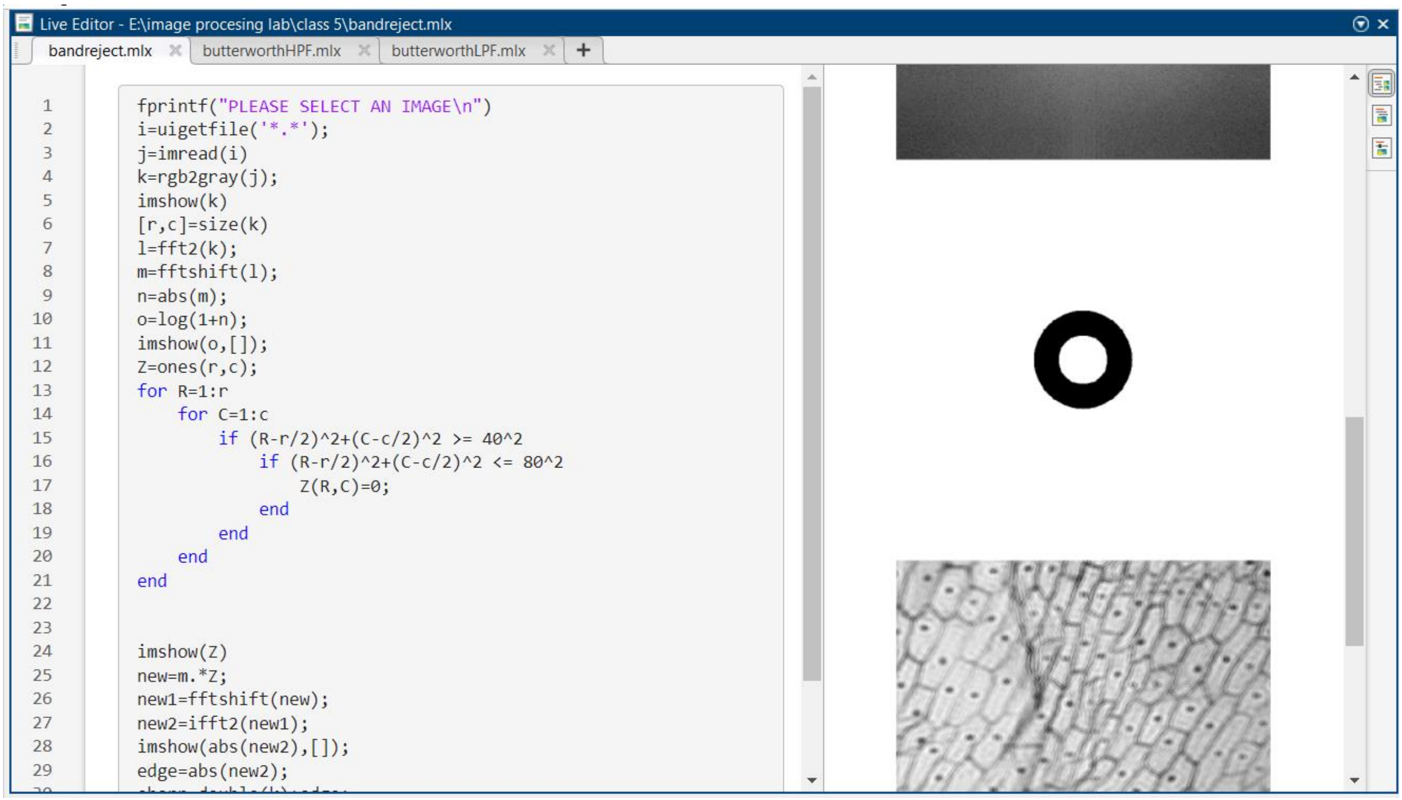

Figure 11: use of image band reject filter in MATLAB

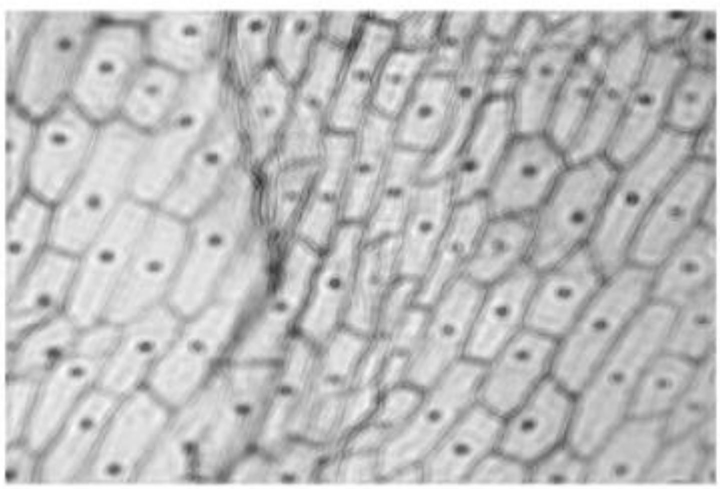

Figure 12: image processing using band reject filter in MATLAB

BUTTER WORTH FILTERS

Butter Worth High and Low pass filter:

A Butterworth filter is a type of digital filter used in image processing to enhance or suppress certain frequency components in an image. It is a low-pass filter that is designed to allow low-frequency components to pass through while attenuating high-frequency components. The filter is named after the British engineer and physicist Stephen Butterworth, who developed it in the 1930s. It is widely used in image processing applications such as image smoothing, noise reduction, and edge detection.

The Butterworth filter is designed based on a specific order and cut-off frequency. The order of the filter determines how sharply the filter cuts off the high-frequency components. The cut-off frequency is the frequency at which the filter begins to attenuate the high-frequency components. In image processing, the Butterworth filter can be applied to the Fourier transform of an image, which represents the image's frequency content. The filter's cut-off frequency and order can be chosen based on the specific application requirements. The Butterworth filter has several advantages over other types of filters, including its smooth transition between passband and stopband, and its ability to attenuate high-frequency noise without affecting the image's edges. However, it can also introduce ringing artifacts and may not be suitable for all types of image processing applications. Overall, the Butterworth filter is a useful tool in image processing for selectively enhancing or suppressing frequency components in an image, and its specific design parameters can be adjusted to achieve the desired results.

CODE

```
% BUTTER WORTH LOW PASS FILTER


fprintf("PLEASE SELECT AN IMAGE\n")
i=uigetfile('*.*');
j=imread(i)
k=rgb2gray(j);
imshow(k)
[r,c]=size(k)
l=fft2(k);
m=fftshift(l);
n=abs(m);
o=log(1+n);
imshow(o,[]);
Z=ones(r,c);
d0=100;
n=10;
for R=1:r
    for C=1:c
        d=sqrt((R-(r/2))^2+(C-(c/2))^2);
        Z(R,C)=1/(1+(d/d0)^(2*n));
    end
end



imshow(Z)
```

```
new=m.*Z;
new1=fftshift(new);
new2=ifft2(new1);
imshow(abs(new2),[]);
edge=abs(new2);
sharp=double(k)+edge;
imshow(sharp,[]);
```

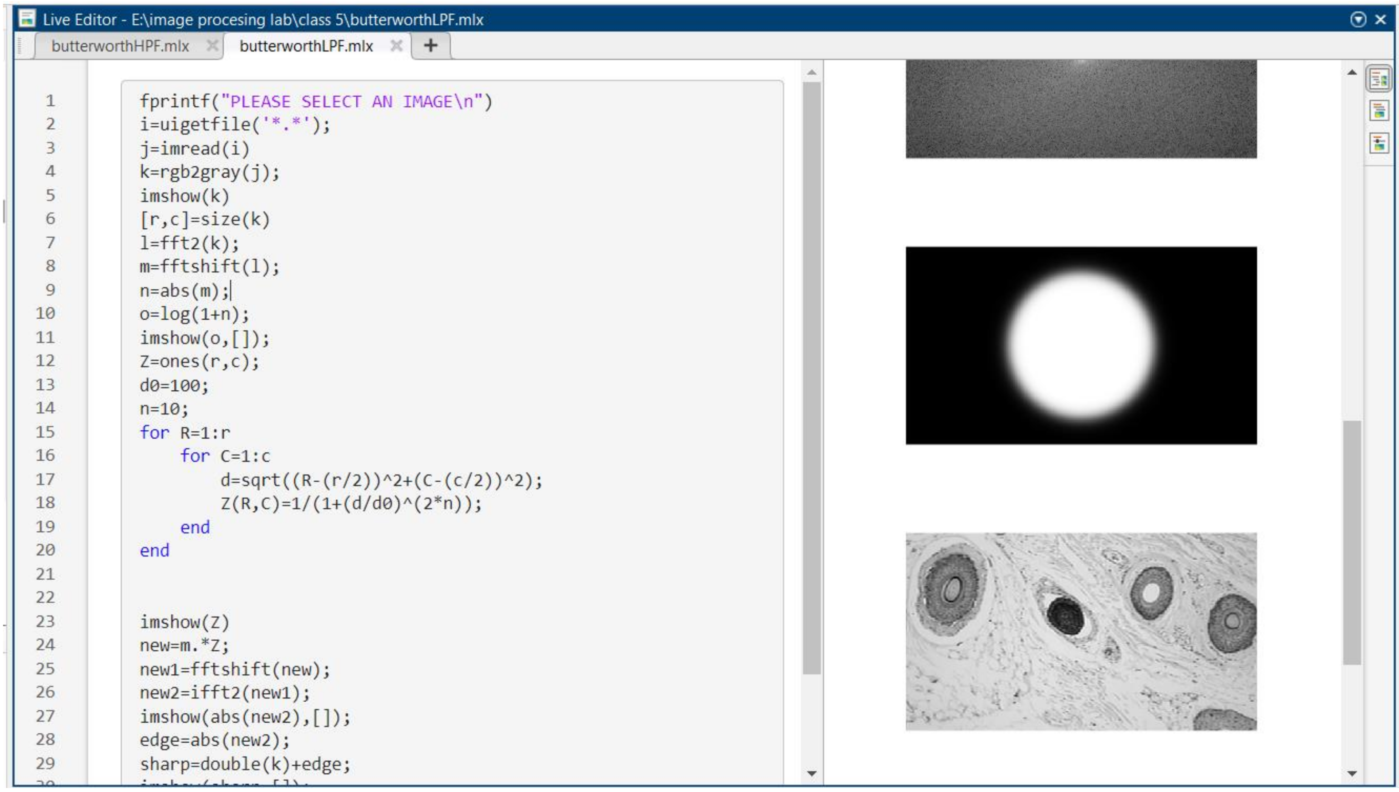

Figure 13: use of butter worth low pass filter in MATLAB

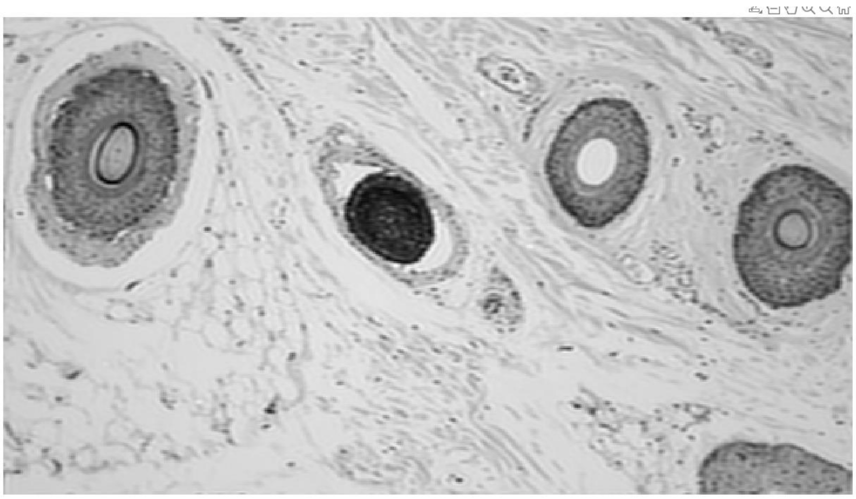

Figure 14: image processing butter worth low pass filter MATLAB

CODE

```
% BUTTER WORTH HIGH PASS FILTER

fprintf("PLEASE SELECT AN IMAGE\n")
i=uigetfile('*.*');
j=imread(i)
k=rgb2gray(j);
imshow(k)
[r,c]=size(k)
l=fft2(k);
m=fftshift(l);
n=abs(m);
o=log(1+n);
imshow(o,[]);
Z=zeros(r,c);
d0=20;
n=2;
for R=1:r
    for C=1:c
        d=sqrt((R-(r/2))^2+(C-(c/2))^2);
        Z(R,C)=1/(1+(d0/d)^(2*n));
    end
end



imshow(Z)
new=m.*Z;
new1=fftshift(new);
new2=ifft2(new1);
imshow(abs(new2),[]);
edge=abs(new2);
sharp=double(k)+edge;
subplot(1,2,1); imshow(sharp,[]); title('Sharpened Image');

subplot(1,2,2); imshow(k,[]); title('Original Image');
```

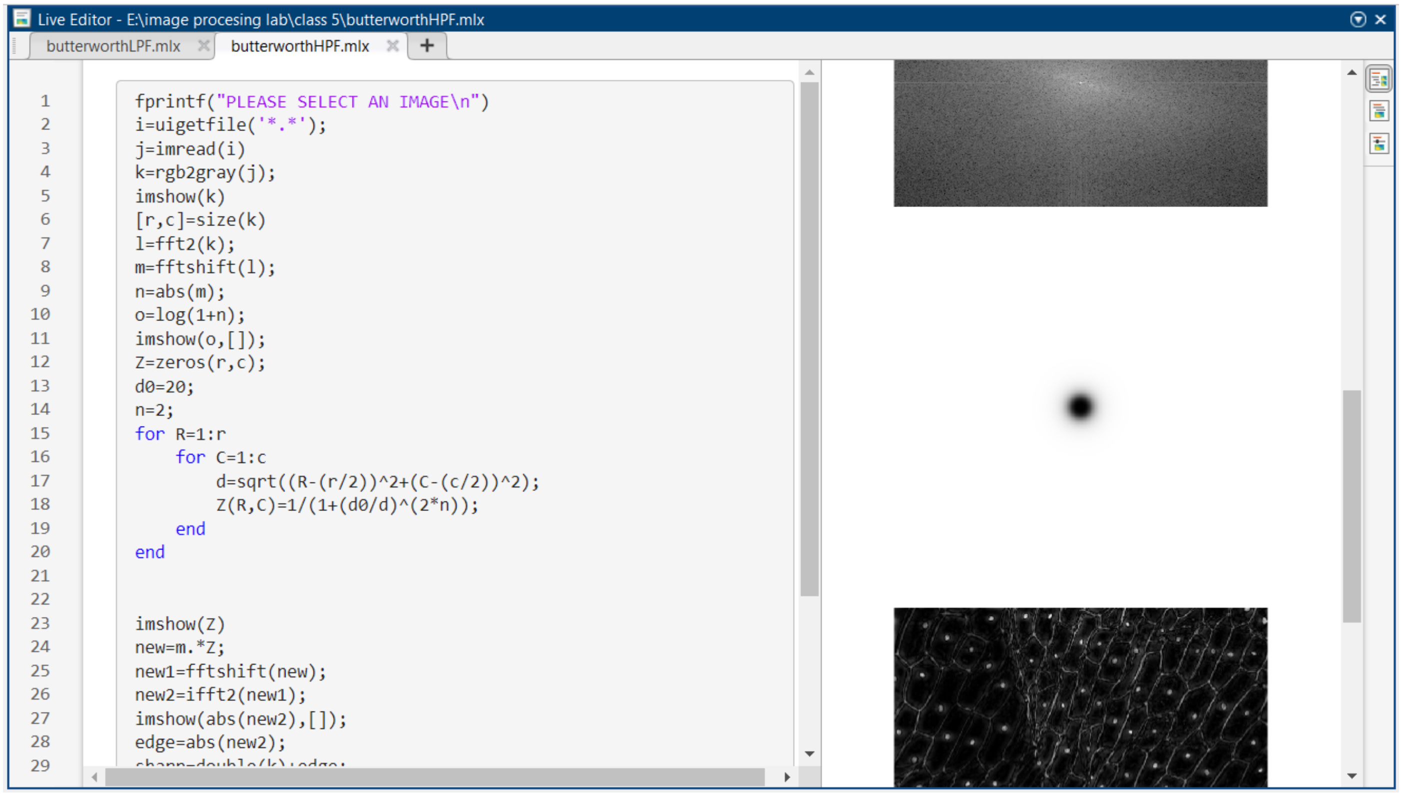

Figure 15: use of butter worth high pass filter in MATLAB

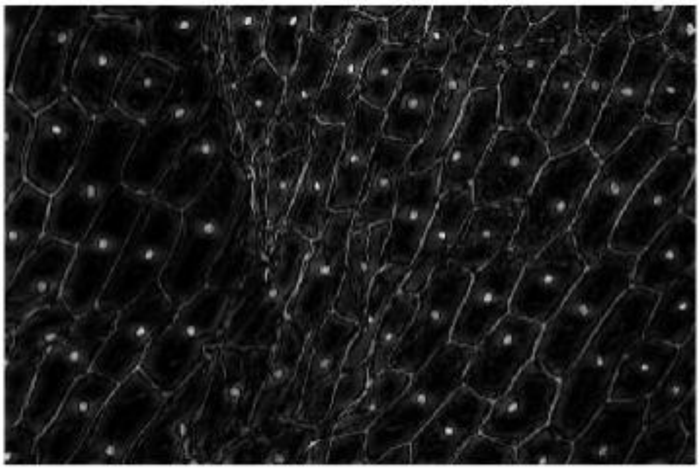

Figure 16: image processing butter worth high pass filter MATLAB

Shreenandan Sahu |120BM0806