MATLAB Codes

The filter design involved designing the algorithms based on each filter's function and running the algorithm on MATLAB. Below are the MATLAB codes for each filter designed.

1. Lowpass Filters

1) ILPF code

```
% This program code is used to design ILPF for filtering noise in the image
clear all; close all; clc;
f=im2double(rgb2gray(imread('Lenna.png'))); % input image
figure(1)
imshow(f);
% add noise example, speckle noise of variance 0.08
y=imnoise(f, 'speckle', 0.08);
% pre processing stage (fast Fourier transform)
F=fft2(f);
% compute 2D Discrete Fast Fourier Transform
F=fftshift(F);% Shift zero-frequency component to center of spectrum
figure(2)
imshow(log(1+abs(F)),[3,7]);% log scale for better visual

% finding the center of the image M*N
[M,N]=size(f);
[M,N]=meshgrid(M,N);
H=zeros(M,N); % zeros changes all regions to black
Do=35; % frequency 0% ~ 100%

for i=1:M
    for j=1:N
        if sqrt((i-M/2)^2+(j-N/2)^2)<=Do
            H(i,j)=ones;
        end
    end
end
figure(3)
imshow(H,[ ]);
% multiply f(u,v) by a filter function.
G=H.*F;
figure(4)
imshow(log(1+abs(G)),[3,7]);
% compute the inverse FFT shift of the result.
G=ifftshift(G);
%obtain the real part of the result.
g=real(ifft2(G));
% obtain the output filtered image
figure(5)
imshow(g,[ ]);
title('Ideal LPF')
% create the intensity mesh plot
```

```
figure(6)
surf([-1:2/(N-1):1],[-1:2/(M-1):1], H);
shading interp;
title('Ideal Low Pass Filter');
xlabel('M');
ylabel('N');
zlabel('intensity');
grid on
```

2) BLPF code

```
% Butterworth High pass and Low pass filter
close all; clear all; clc;
RGB=imread('Lenna.png'); % input image
I=rgb2gray(RGB); % convert the image to grey
% add noise example, speckle noise of variance 0.08
y=imnoise(f, 'speckle', 0.08);
A = fft2(double(y)); % compute FFT of the grey image
A1=fftshift(A); % Shift zero-frequency component to center of spectrum

% Butterworth Filter Response Calculation
[M,N]=size(A); % image size
R=30; % filter size parameter
n=2; % order
X=0:N-1;
Y=0:M-1;
[X,Y]=meshgrid(X,Y);
Cx=0.5*N;
Cy=0.5*M;
D=sqrt((X-Cx).^2+(Y-Cy).^2);
Lo=1./(1+(D./R).^(2*n)); %butterworth lowpass equation

% Filtered image=ifft(filter response*fft(original image))
J=A1.*Lo;
J1=ifftshift(J);
B1=real(ifft2(J1));

% visualizing the results
figure(1)
imshow(I);colormap gray
title('original image','fontsize',14)

figure(2)
imshow(abs(A1),[-12 300000]); colormap gray
title('fft of original image','fontsize',14)

figure(3)
imshow(Lo, []); colormap gray
title('Low pass spectrum','fontsize',14)

figure(4)
imshow(log(1+abs(J)),[3,7]); colormap gray
title('low pass spectrum filter function','fontsize',14)
```

```
figure(5)
imshow(B1,[ ]), colormap gray
title('low pass filtered image','fontsize',14)

figure(6)
   mesh(X,Y,Lo)
   axis([ 0 N 0 M 0 1])
   h=gca;
   get(h,'FontSize')
   set(h,'FontSize',14)
   title('Butterworth LPF H(f)','fontsize',14)
   xlabel('M')
   ylabel('N')
```

3) GLPF code

```
% Gaussian Highpass & Lowpass filter
close all; clear all; clc;
RGB=imread('Lenna.png'); % input image
I=rgb2gray(RGB); % convert the image to gray
% add noise example, speckle noise of variance 0.08
y=imnoise(f, 'speckle', 0.08);
A = fft2(double(y)); % compute FFT of the grey image
A1=fftshift(A); % Shift zero-frequency component to center of spectrum

% Gaussian Filter Response Calculation
[M,N]=size(A); % image size
R=80; % filter size parameter
X=0:N-1;
Y=0:M-1;
[X,Y]=meshgrid(X,Y);
Cx=0.5*N;
Cy=0.5*M;
Lo=exp(-((X-Cx).^2+(Y-Cy).^2)./(2*(R.^2)));
Hi=1-Lo; % High pass filter=1-low pass filter

% Filtered image=ifft(filter response*fft(original image))
J=A1.*Lo;
J1=ifftshift(J);
B1=real(ifft2(J1));

K=A1.*Hi;
K1=ifftshift(K);
B2=real(ifft2(K1));

% visualizing the results
figure(1)
imshow(I);colormap gray
title('original image','fontsize',14)

figure(2)
imshow(abs(A1),[-12 300000]); colormap gray
title('fft of original image','fontsize',14)
```

```
figure(3)
imshow(Lo, []); colormap gray
title('Low pass spectrum','fontsize',14)

figure(4)
imshow(log(1+abs(J)),[3,7]); colormap gray
title('low pass spectrum filter function','fontsize',14)

figure(5)
imshow(B1,[ ]), colormap gray
title('lowpass filtered image','fontsize',14)


figure(6)
imshow(B2,[ ]), colormap gray
title('Highpass filtered image','fontsize',14)

figure(7)
   mesh(X,Y,Lo)
   axis([ 0 N 0 M 0 1])
   h=gca;
   get(h,'FontSize')
   set(h,'FontSize',14)
   title('Gaussian LPF H(f)','fontsize',14)


figure(8)
   mesh(X,Y,Hi)
   axis([ 0 N 0 M 0 1])
   h=gca;
   get(h,'FontSize')
   set(h,'FontSize',14)
   title('Gaussian HPF H(f)','fontsize',14)
```

4) ELPF code

```
% Exponential Low pass filter
close all; clear all; clc;
RGB=imread('Lenna.png'); % read input image
I=rgb2gray(RGB); % convert the image to grey
y=imnoise(I, 'speckle', 0.08); % add noise to the input image
A = fft2(double(y)); % compute FFT of the grey image
A1=fftshift(A); % Shift zero-frequency component to center of spectrum

% Exponential Filter Response Calculation
[M,N]=size(A); % image size
R=40; % filter size parameter
n=3;
X=0:N-1;
Y=0:M-1;
[X,Y]=meshgrid(X,Y);
Cx=0.5*N;
Cy=0.5*M;
D=sqrt((X-Cx).^2+(Y-Cy).^2);
```

```
Lo=exp(-(D./R).^n); %exponential filter equation

% Filtered image=ifft(filter response*fft(original image))
J=A1.*Lo;
J1=ifftshift(J);
B1=ifft2(J1);

% visualizing the results
figure(1)
imshow(I);colormap gray
title('original image','fontsize',14)

figure(2)
imshow(abs(A1),[-12 300000]); colormap gray
title('fft of original image','fontsize',14)

figure(3)
imshow(Lo, []); colormap gray
title('Low pass spectrum','fontsize',14)

figure(4)
imshow(log(1+abs(J)),[3,7]); colormap gray
title('low pass spectrum filter function','fontsize',14)

figure(5)
imshow(abs(B1),[12 290]), colormap gray
title('low pass filtered image','fontsize',14)

figure(6)
   mesh(X,Y,Lo)
   axis([ 0 N 0 M 0 1])
   xlabel('M');
   ylabel('N');
   h=gca;
   get(h,'FontSize')
   set(h,'FontSize',14)
   title('Exponential LPF H(f)','fontsize',14)
```

5) Elliptical BLPF code

```
% Elliptical BLPF
close all; clear all; clc;
RGB=imread('Lenna.png'); % read input image
I=rgb2gray(RGB); % convert the image to grey
y=imnoise(I, 'speckle', 0.08); % add noise
A = fft2(double(y)); % compute FFT of the grey image
A1=fftshift(A); % Shift zero-frequency component to center of spectrum

% Butterworth Filter Response Calculation
[M,N]=size(A); % image size
DoM=90;
Dom=70;% filter size parameter
n=2; % order
X=0:N-1;
```

```
Y=0:M-1;
[X,Y]=meshgrid(X,Y);
Cx=0.5*N;
Cy=0.5*M;
x2=X-Cx;
y2=Y-Cy;
Lo=1./(1+((x2/(DoM)).^2 + (y2/(Dom)).^2).^n);% ellipse lowpass equation
J=A1.*Lo;
J1=ifftshift(J);
B1=real(ifft2(J1));

% visualizing the results
figure(1)
imshow(I);colormap gray
title('original image','fontsize',14)

figure(2)
imshow(abs(A1),[-12 300000]); colormap gray
title('fft of original image','fontsize',14)

figure(3)
imshow(Lo, []); colormap gray
title('Low pass spectrum','fontsize',14)

figure(4)
imshow(log(1+abs(J)),[3,7]); colormap gray
title('low pass spectrum filter function','fontsize',14)

figure(5)
imshow(B1,[ ]), colormap gray
title('low pass filtered image','fontsize',14)

figure(6)
   mesh(X,Y,Lo)
   axis([ 0 N 0 M 0 1])
   h=gca;
   get(h,'FontSize')
   set(h,'FontSize',14)
   title('Butterworth LPF H(f)','fontsize',14)
   xlabel('M')
   ylabel('N')
```

6) Elliptical GLPF code

```
% Elliptical GLPF
close all; clear all; clc;
RGB=imread('Lenna.png'); % read input image
I=rgb2gray(RGB); % convert the image to grey
y=imnoise(I, 'speckle', 0.08); % add noise
A = fft2(double(y)); % compute FFT of the grey image
A1=fftshift(A); % Shift zero-frequency component to center of spectrum

% Filter Response Calculation
[M,N]=size(A); % image size
```

```
D=20; % filter size parameter
d=70;
X=0:N-1;
Y=0:M-1;
[X,Y]=meshgrid(X,Y);
Cx=0.5*N;
Cy=0.5*M;
x2=X-Cx;
y2=Y-Cy;
Lo=exp(-0.5*((x2/D).^2 + (y2/d).^2));

% Filtered image=ifft(filter response*fft(original image))
J=A1.*Lo;
J1=ifftshift(J);
B1=real(ifft2(J1));

% Visualizing the results
figure(1)
imshow(I);colormap gray
title('original image','fontsize',14)

figure(2)
imshow(abs(A1),[-12 300000]); colormap gray
title('fft of original image','fontsize',14)

figure(3)
imshow(Lo, []); colormap gray
title('Low pass spectrum','fontsize',14)

figure(4)
imshow(log(1+abs(J)),[3,7]); colormap gray
title('low pass spectrum filter function','fontsize',14)

figure(5)
imshow(B1,[ ]), colormap gray
title('low pass filtered image','fontsize',14)

figure(6)
   mesh(X,Y,Lo)
   axis([ 0 N 0 M 0 1])
   h=gca;
   get(h,'FontSize')
   set(h,'FontSize',14)
   title('Gaussian LPF H(f)','fontsize',14)
```

7) Elliptical ELPF code

```
%Exponential Low pass filter
close all; clear all; clc;
RGB=imread('Lenna.png'); % read input image
I=rgb2gray(RGB); % convert the image to grey
y=imnoise(I, 'speckle', 0.08); % add noise
A = fft2(double(f)); % compute FFT of the grey image
A1=fftshift(A); % Shift zero-frequency component to center of spectrum
```

```
A2=log(1+abs(A1));

% Exponential Filter Response Calculation
[M,N]=size(A); % image size
D=40; % filter size parameter
d=25;
n=1;
X=0:N-1;
Y=0:M-1;
[X,Y]=meshgrid(X,Y);
Cx=0.5*N;
Cy=0.5*M;
x2=X-Cx;
y2=Y-Cy;
P=sqrt((x2/D).^2+(y2/d).^2);
Lo=exp(-(P.^n)); %exponential filter equation

J=A1.*Lo;
J1=ifftshift(J);
B1=real(ifft2(J1));

% Visualizing the results
figure(1)
imshow(I);colormap gray
title('original image','fontsize',14)

figure(2)
imshow(abs(A1),[-12 300000]); colormap gray
title('fft of original image','fontsize',14)

figure(3)
imshow(A2,[]); colormap gray
title('log transformation','fontsize',14)

figure(4)
imshow(Lo, []); colormap gray
title('Low pass spectrum','fontsize',14)

figure(5)
imshow(log(1+abs(J)),[3,7]); colormap gray
title('low pass spectrum filter function','fontsize',14)

figure(6)
imshow(B1,[ ]), colormap gray
title('low pass filtered image','fontsize',14)

figure(7)
   mesh(X,Y,Lo)
   axis([ 0 N 0 M 0 1])
   xlabel('M');
   ylabel('N');5
   h=gca;
   get(h,'FontSize')
   set(h,'FontSize',14)
```

```
   title('Exponential LPF H(f)','fontsize',14)
```

2. Bandpass Filters

1) IBPF code

```
% ideal bandpass filter
clear all; close all; clc;
f=imread('lena.jpg'); % input image
f=im2double(f);
y=imnoise(f, 'speckle', 0.08); % add noise

% pre-processing stage (fast fourier transform)
F=fft2(y);
% compute 2D Discrete Fast Fourier Transform
F1=fftshift(F);% Shift zero-frequency component to center of spectrum
figure(1)
imshow(log(1+abs(F1)),[]);% log scale for better visual

%finding the center of the image M*N
[M,N]=size(f);
 %zeros changes all regions to black
Do=30; %frequency 0% ~ 100%
Di=100;
for i=1:M
    for j=1:N
        D=sqrt((i-M/2)^2+(j-N/2)^2);
        if ( D<=Do )
            H(i,j)=zeros;
        else
            if (D>=Di)
             H(i,j)=zeros;
            else
                H(i,j)=ones;
            end
        end
    end
end
figure(2)
imshow(H,[]);

% multiply f(u,v) by a filter function.
G=F1+H.*F1;
figure(3)
imshow(log(1+abs(G)),[]);

%compute the inverse FFT shift of the result.
G1=ifftshift(G);
%obtain the real part of the result.
G2=real(ifft2(G1));

figure(4)
subplot(1,3,1)
imshow(f); title('a')
```

```
subplot(1,3,2)
imshow(y, []); title ('b')
subplot(1,3,3)
imshow(G2,[ ]), colormap gray
title('c')

figure(5)
surf([-1:2/(N-1):1],[-1:2/(M-1):1], H);
shading interp;
title('Do=80 & Di=100');
xlabel('M');
ylabel('N');
zlabel('intensity');
grid on
```

2) BBPF code

```
% Butterworth Bandpass Filter
close all; clear all; clc;
f=imread('cameraman.gif'); % read input image
f=im2double(f);
x=imnoise(f, 'gaussian', 0, 0.03); % add noise
A = fft2(x); % compute FFT of the grey image
A1=fftshift(A); % frequency scaling

% Butterworth Filter Response Calculation
[M,N]=size(f); % image size
Do=100; % filter size parameter
Di=90;
n=2; % order
X=0:N-1;
Y=0:M-1;
[X,Y]=meshgrid(X,Y);
Cx=0.5*N;
Cy=0.5*M;
D=sqrt((X-Cx).^2+(Y-Cy).^2);
Lo=1./(1+(D./Do).^(2*n)); %butterworth lowpass equation
H=1./(1+(D./Di).^(2*n));
Hi=1-H;
Bp=Lo.*Hi;
% Filtered image=ifft(filter response*fft(original image))

J=A1+Bp.*A1;
J1=ifftshift(J);
B1=real(ifft2(J1));

% visualizing the results
figure(1)
imshow(log(1+abs(A1)),[]); colormap gray
title('fft of original image','fontsize',14)

figure(2)
imshow(Bp, []); colormap gray
title('Low pass spectrum','fontsize',14)
```

```
figure(3)
imshow(log(1+abs(J)),[3,7]); colormap gray
title('low pass spectrum filter function','fontsize',14)

figure(4)
subplot(1,3,1)
imshow(f,[]);colormap gray
title('a')
subplot(1,3,2)
imshow(x,[]);colormap gray
title('b')
subplot(1,3,3)
imshow(B1, [ ]), colormap gray
title('c')

figure(5)
   mesh(X,Y,Bp)
   axis([ 0 N 0 M 0 1])
   h=gca;
   get(h,'FontSize')
   set(h,'FontSize',14)
   title('Do=100 & Di=30','fontsize',14)
   xlabel('M')
   ylabel('N')
```

3) GBPF code

```
% Gaussian bandpass filter
close all; clear all; clc;
f=imread('lena.jpg'); % read input image
f=im2double(f);
y=imnoise(f, 'speckle', 0.08);% add noise
A = fft2(y); % compute FFT of the grey image
A1=fftshift(A% Shift zero-frequency component to center of spectrum

% Gaussian Filter Response Calculation
[M,N]=size(f); % image size
Do=100; % filter size parameter
Di=80;
X=0:N-1;
Y=0:M-1;
[X,Y]=meshgrid(X,Y);
Cx=0.5*N;
Cy=0.5*M;
Lo=exp(-((X-Cx).^2+(Y-Cy).^2)./(2*(Do.^2)));
% Highpass filter=1-lowpass filter
Hi=1-exp(-((X-Cx).^2+(Y-Cy).^2)./(2*(Di.^2)));
Bp=Lo.*Hi;

% Filtered image=ifft(filter response*fft(original image))
K=A1+Bp.*A1;
K1=ifftshift(K);
B2=real(ifft2(K1));
```

```
% visualizing the results
figure(1)
imshow(log(1+abs(A1)),[]); colormap gray
title('fft of original image','fontsize',14)

figure(2)
imshow(Bp, []); colormap gray
title('Low pass spectrum','fontsize',14)

figure(3)
subplot(1,2,1)
imshow(y);colormap gray
title('a')
subplot(1,2,2)
imshow(B2,[]), colormap gray
title('b')

figure(4)
   mesh(X,Y,Bp)
   axis([ 0 N 0 M 0 1])
   h=gca;
   get(h,'FontSize')
   set(h,'FontSize',14)
   title('GBPF H(f)','fontsize',14)
```

4) EBPF code

```
%Exponential Bandpass filter
close all; clear all; clc;
f=imread('cameraman.gif'); % read input image
f=im2double(f);
y=imnoise(f, 'speckle', 0.08); % add noise
A = fft2(y); % compute FFT of the grey image
A1=fftshift(A); % Shift zero-frequency component to center of spectrum

% Exponential Filter Response Calculation
[M,N]=size(f); % image size
Do=100; % filter size parameter
Di=30;
n=2;
X=0:N-1;
Y=0:M-1;
[X,Y]=meshgrid(X,Y);
Cx=0.5*N;
Cy=0.5*M;
D=sqrt((X-Cx).^2+(Y-Cy).^2);
Lo=exp(-(D./Do).^n); %exponential filter equation
Hi=1-exp(-(D./Di).^n);
Bp=Lo.*Hi;

% Filtered image=ifft(filter response*fft(original image))
J=A1+Bp.*A1;
J1=ifftshift(J);
B1=real(ifft2(J1));
```

```
% Visualizing the results
figure(1)
imshow(log(1+abs(A1)), [2 7]); colormap gray
title('fft of original image','fontsize',14)

figure(2)
imshow(Bp, [ ]); colormap gray
title('bandpass spectrum','fontsize',14)

figure(3)
imshow(log(1+abs(J)),[3,7]); colormap gray
title('bandpass spectrum filter function','fontsize',14)

figure(4)
subplot(1,2,1)
imshow(f), colormap gray
title('a')
subplot(1,2,2)
imshow(B1,[ ]), colormap gray
title('b')

figure(5)
   mesh(X,Y,Bp)
   axis([ 0 N 0 M 0 1])
   xlabel('M');
   ylabel('N');
   h=gca;
   get(h,'FontSize')
   set(h,'FontSize',14)
   title('Exponential BPF H(f)','fontsize',14)
```