Abstract:

In this project, image filtering in frequency domain is studied. The main two parts of the project are mainly concerned with finding and displaying the Fourier spectrum of an image. For better visualization of the frequency, the Fourier spectrum is shifted to the center when be displayed. The average value of the image is found by conventional averaging methods, and it is compared to the result obtained directly from the spectrum.

In the third part an image is corrupted by a sinusoidal noise and a Butterworth filter is designed to clean the noised image. The filter parameters are determined from spectrum analysis.
Technical Discussion

PART 1:

For the 2D continuous function, its Fourier Transform can be given as follow:

\[ F(u, v) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x, y) e^{-j2\pi(ux + vy)} \, dx \, dy \]

Where \( F(u,v) \) is the Fourier Transform result with the frequency components \( u \) and \( v \) corresponding to \( x \) and \( y \), respectively. And \( f(x,y) \) is the original continuous function.

For the 2D discrete function, the Fourier Transform can be given as follow:

\[ F(u, v) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) e^{-j2\pi\left(\frac{ux}{M} + \frac{vy}{N}\right)} \]

Where \( u=0,1,2,\ldots,M-1 \) \( v=0,1,2,\ldots,N-1 \)

From the 2D discrete Fourier Transform equation, one can show that the translation property of discrete Fourier Transform is given as:

\[ F(u-u_o, v-v_o) = f(x, y) e^{j2\pi\left(\frac{u_o x}{M} + \frac{v_o y}{N}\right)} \]

When \( u_o=M/2, \ v_o=N/2 \) we get:

\[ F(u-M/2, v-N/2) = f(x, y) e^{j2\pi\left(\frac{x}{2} + \frac{y}{2}\right)} = f(x, y)(-1)^{(x+y)} \]

This equation shows that the Fourier spectrum can be shifted to the central point for better displaying by multiplying the original image by \((-1)^{(x+y)}\).

The 2D inverse Fourier Transform for discrete function can be computed as follow:

\[ f(x, y) = \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u, v) e^{j2\pi\left(\frac{ux}{M} + \frac{vy}{N}\right)} \]

Where \( x=0,1,2,\ldots,M-1 \) \( y=0,1,2,\ldots,N-1 \)

PART 2:
The average value of the image is computed by summing all the pixel values in the image and dividing it by the image size (M*N).

From the Fourier Transform formula when \( u = v = 0 \) we get the following:

\[
F(0,0) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y)
\]

Thus the zero frequency component in the Fourier Transform spectrum can be used to find the average value of the image. When using the centered frequency spectrum the average value is the middle component in the spectrum.

PART 3:

A sinusoidal noise of the form

\[
f(x, y) = A \sin(u_o x + v_o y)
\]

Where, \( A = 50 \), \( u_o = M/2 \) and \( v_o = 0 \) are implemented.

This noise is added to image and the spectrum of the noisy image is displayed. The effect of the noise on the spectrum can be observed. Using this information, \( D_o \) and \( W \) values can be determined. A band-reject Butterworth filter is:

\[
H(u, v) = \frac{1}{1 + \left[ \frac{D(u, v) \times W}{D^2(u, v) - D^2_o} \right]^{2\pi}}
\]

Where, \( D_o \) is the distance from the center of the spectrum to the middle part of the band which is desired to be rejected, \( W \) is the width of the band and \( D \) is the distance of the point \((u,v)\) to the origin of the Fourier Transform.
A) The Fourier Transform (circles image)

- Fig(1-1) shows the original image
- Fig(1-2) shows the Fourier Spectrum without Centered Translation
- Fig(1-3) shows the Fourier Spectrum with Centered Translation

A) The Fourier Transform (crosses image)

- Fig(1-4) shows the original image
- Fig(1-5) shows the Fourier Spectrum without Centered Translation
- Fig(1-6) shows the Fourier Spectrum with Centered Translation

Fig(1-4)

Fig(1-5)       Fig(1-6)

PART 2:
A) The Fourier Transform and the Average Value (bridge image)

- Fig(2-1) shows the original image
- Fig(2-2) shows the Fourier Spectrum with Centered Translation

Fig(2-1)       Fig(2-2)

- The Average Value computed from the image is 113.8547,
- The Average Value computed from the Fourier Transform Spectrum is 113.8547.
B) The Fourier Transform and the Average Value (camera image)

- Fig(2-3) shows the original image
- Fig(2-4) shows the Fourier Spectrum with Centered Translation

![Original Image](Fig(2-3)) ![Fourier Spectrum](Fig(2-4))

- The Average Value computed from the image is 118.7245,
- The Average Value computed from the Fourier Transform Spectrum is 118.7245.

PART 3:

Image Filtering in Frequency Domain (lena image)
- Fig(3-1) shows the original image
- Fig(3-2) shows the Spectrum of original image with Centered Translation
- Fig(3-3) shows the noisy image
- Fig(3-4) shows the Spectrum of noisy image with Centered Translation

Fig(3-1)       Fig(3-2)

Fig(3-3)       Fig(3-4)

- Fig(3-5) shows the Spectrum of Butterworth Band rejected Filter with 
  \[ D_o=100, \quad W=20, \quad n=4. \]
- Fig(3-6) shows the filtered image with $D_o = 10$
- Fig(3-7) shows the filtered image with $D_o = 95$
- Fig(3-8) shows the filtered image with $D_o = 120$

**Results Discussion**

**PART I:**
- As shown from the results the centered Fourier Transform Spectrum (e.g. Fig.(1.2)) gives better visualization than the original one (e.g. Fig.(1.3))
- Without any disturbance, most of the spectrum located in the low frequency part, that is, the centered part of the spectrum.

PART 2:

- from the property of the Fourier Transform, the average value of the image can be easily obtained from its Fourier transform, that is, the value of the centered pixel in the centered spectrum
- The average value of the image which is obtained from the centered pixel means, that the majority of the signal power located in the centered pixel then the image can be partially reconstructed only from the low frequency components, this fact is used in image compression to reduce the memory size required by the image.

PART 3:

- From the spectrum of the noisy image (e.g. Fig.(3.4)) and from noise information one can detect that the noise is located in specified frequency so to remove it we need a band rejected filter at this frequency.
- Butterworth band rejected filter is used to remove the noise. The ideal band rejected filter did not use to avoid the sharp edges which lead to ringing effect.
- The parameters of the filter are chosen by trails and error. D_o is the most important parameter, when it is small the low frequency component of the image is filtered out giving a bad results (e.g. Fig.(3.6)), when it becomes bigger the high frequency component which contains the noise is filtered out and gives better output image (e.g. Fig.(3.7)). But if it becomes even bigger until exceeds the edges of the spectrum image all frequencies will pass and the output image looks like the noisy one (e.g. Fig.(3.8)).

Appendix {Program Listing}

*Fourier Transform & its Inverse are Implemented by C++*
int FFT2(Cmplex** cFFtIm, double** Img, double** FftIm, int Hight, int Width)
{
    cout<<"input fft2"<<endl;
    //int i;
    int x,y,u,v;

    ///Shift Image
    for(x=0;x<Hight;x++)
    for(y=0;y<Width;y++)
        Img[x][y]=pow(-1,x+y)*Img[x][y];

    ///FFT2
    for(u=0;u<Hight;u++)
    {
        for(v=0;v<Width;v++)
        {
            cout<<".
            cFFtIm[u][v].Re=0.0;
            cFFtIm[u][v].Im=0.0;
            for(x=0;x<Hight;x++)
            {
                for(y=0;y<Width;y++)
                    cFFtIm[u][v].Re=cFFtIm[u][v].Re+Img[x][y]*cos(2*pi*(1.0*u*x/Hight+1.0*v*y/Width));
                    cFFtIm[u][v].Im=cFFtIm[u][v].Im-
                    Img[x][y]*sin(2*pi*(1.0*u*x/Hight+1.0*v*y/Width));
            }
        }
    }

    ///Get The Magnitude Value
    double maxfft=0.0;
    for(u=0;u<Hight;u++)
    {
        for(v=0;v<Width;v++)
        {
            FftIm[u][v]=sqrt(cFFtIm[u][v].Re*cFFtIm[u][v].Re+cFFtIm[u][v].Im*cFFtIm[u][v].Im);
            if (FftIm[u][v] > maxfft) maxfft=FftIm[u][v];
        }
    }

    ///Normalize FFT2
    for(u=0;u<Hight;u++)
    {
        for(v=0;v<Width;v++)
        {
            FftIm[u][v]=255*FftIm[u][v]/maxfft;
        }
    }

    cout<<"output fft2"<<endl;
    return 0;
}

int inFFT2(Cmplex** cFFtIm, double** Img, int Hight, int Width)
{
    int x,y,u,v;
    for(x=0;x<Hight;x++)

MATLAB programs is used also to compare the results

- Shifted and Un-shifted Spectrum and Average Value

clear all;
clc;
im=readpgm('E:\Classes\MATLAB_PROGRAM\ECE618\pgm_images\camera.pgm');
L_max=max(max(im));
L_min=min(min(im));
figure,imshow(im,[L_min L_max]);
[N M]=size(im);
AverageValue=1.0*sum(sum(im))/(N*M)
im=double(im);
im=fft2(im);
imfM=abs(im);
AverageValue=imfM(M/2.0+1,N/2.0+1)/(N*M)
L_max=max(max(imfM));
imfM=500*255*imfM/L_max;
L_max=max(max(imfM));
L_min=min(min(imfM));
figure,imshow(imfM,[0 255]);
im=fftshift(im);
imfM=abs(im);
AverageValue=imfM(M/2.0+1,N/2.0+1)/(N*M)
L_max=max(max(imfM));
imfM=500*255*imfM/L_max;
L_max=max(max(imfM));
L_min=min(min(imfM));
figure,imshow(imfM,[0 255]);

- Image Filtering

clear all;
clc;
close all;
im1=readpgm('E:\Classes\MATLAB_PROGRAM\ECE618\pgm_images\lena.pgm');
[N M]=size(im1);
L_max=max(max(im1));
L_min=min(min(im1));
figure,imshow(im1,[L_min L_max]);

im=double(im1);
im=fft2(im);
im=fftshift(im);
imfM1=abs(im);
L_max=max(max(imfM1));
imfM1=350*255*imfM1/L_max;
L_max=max(max(imfM1));
L_min=min(min(imfM1));
figure,imshow(imfM1,[0 255]);

x=[0:M-1];
A=50;
f=zeros(M,N);
for i=1:N
    f(:,i)=A*sin(x*M/2.0);
end
Nim=f+im1;
L_max=max(max(Nim))
L_min=min(min(Nim))
figure,imshow(Nim,[L_min L_max]);

im=double(Nim);
im=fft2(im);
im=fftshift(im);
imfN=abs(im);
L_max=max(max(imfN));
imfN=350*255*imfN/L_max;
L_max=max(max(imfN));
L_min=min(min(imfN));
figure,imshow(imfN,[0 255]);

H=zeros(M,N);
Do=120;
n=4;
W=20;
for u=1:M
    for v=1:N
        us=u-1-M/2.0;
        vs=v-1-N/2.0;
        H(u,v)=1.0/(1+(W*sqrt(us*us+vs*vs))/(0.0000001+us*us+vs*vs-Do*Do))^(2*n);
    end
end
L_max=max(max(H));
L_min=min(min(H));
figure,imshow(H,[L_min L_max]);

fltIm=H.*im;
im1=fftshift(fltIm);
inimfM=ifft2(im1);
RinimfM=real(inimfM);
L_max=max(max(RinimfM))
L_min=min(min(RinimfM))
figure,imshow(RinimfM,[L_min L_max]);