Brief Summary of Filtering in the Frequency Domain

Chuan Sun
University of Kansas
Lawrence, KS
chuansun@ku.edu

***Abstract*—In this paper, the frequency domain filtering technique was investigated, with three types of commonly used filters discussed. The frequency domain filtering is applied to a pattern image to demonstrate the difference between different filters and parameters, and then applied to a space image to solve the periodic noise in background.**

Keywords—Frequency domain filtering, Fourier transform, lowpass filter, high pass filter, bandpass filter

I. Introduction

Frequency domain filtering has been an important concept in image processing, which is typically used for image smoothing and sharpening based on the frequency of images. Generally it uses the Fourier transformation to transform an image from spatial domain into frequency domain and applies different kinds of pass or reject filters (e.g. low pass, high pass, etc.) to achieve various filtering effects.

A. Fourier Transform

Fourier transform can be used to approximate a continuous variable using a weighted sum of multiple periodic functions such as sines and cosines. It's constructed using the Euler's formula, denoted as

$$FT[f(t)] = \int_{-\infty}^{\infty} f(t)e^{-j2\pi xt}dt \quad (1)$$

where f(t) is the continuous function of a continuous variable t, and j equals to $\sqrt{-1}$. Formula (1) enables the transformation from spatial domain to frequency domain, and the inverse operation, denoted by formula (2), conducts the inverse transformation from frequency domain back to spatial domain.

$$f(t) = \int_{-\infty}^{\infty} FT[f(t)]e^{j2\pi xt}dx \quad (2)$$

Fourier transform enables discrete sampling on continuous functions by multiplying a train of sampling impulses. The sampling mechanism can also be applied to 2-variables to form the 2D Fourier transform so that image representations in frequency domain could be retrieved. This representation is described in the form of Fourier spectrum and phase angle, as denoted in formula (3) and (4) respectively.

$$|F(u,v)| = \sqrt{R^2(u,v) + I^2(u,v)} \quad (3)$$

$$\varphi(u,v) = \arctan\left[\frac{I(u,v)}{R(u,v)}\right] \quad (4)$$

The magnitude of amplitude for a frequency is an indication of pixel intensities in an image. While the phase angle represents the displacement of an sinusoids, which denotes object locations in an image.

B. Frequency Filtering Process

To conduct filtering in the frequency domain, several prerequisite steps are needed before the Fourier transform. First zero padding is applied on the image to solve the wraparound errors caused by the convolution of two discrete functions in the scenario of 2D image. Enough zeros are padded to the image to fully cover the length and width of two function patterns, so that the convolution contains intact periodicity. Also, in order to center the spectrum to the origin point, $(-1)^{x+y}$ is applied to the padded image.

Then the discrete Fourier transform is applied on the preprocessed image so that the image is transformed into frequency domain. We further multiply it by a generated filter function which is in the same size of the image, to perform a filtering process in frequency domain.

To convert the filtering effect back to spatial domain, the inverse discrete Fourier transform is applied on the product. We extract the real part of it and invert it back from centering by multiplying $(-1)^{x+y}$, as well as from zero padding by extract the top left quadrant to finally get the filtered image in spatial domain.

It could be summarized that the frequency filtering process basically preprocesses and transforms an image into frequency domain in a centered way, and applies a filter in frequency domain with the support of zero padding for optimal convolution, and finally converts it back to spatial domain using a set of inverse operations.

C. Filters

The filter we use determines the effect we would like to apply to the image. Several commonly used filters are discussed below.

1. Lowpass filters

Technically lowpass filters only allow low frequency to pass through the image, so in frequency domain perspective plot, areas around the center (indicating low frequency) would have a bulge while all other areas are zero plain. To implement this effect, there're several options to approach this effect, such as the ideal lowpass which sets a hard threshold of cutoff frequency; the Butterworth lowpass that shrinks with the distance to the origin point by setting an order n and a cutoff frequency D_0; the Gaussian lowpass that constructs the filter shape using a Gaussian bell, parameterized by the distance to the center and a D_0 as well.

The comparison of these three lowpass filters are displayed in Fig 1. The ideal lowpass filter preserves original pattern in low frequency areas, so when converting back to spatial domain, the Fourier series would result in ringing effects. While the Butterworth and the Gaussian lowpass smooth the periodicity patterns and alleviate ringing.

Lowpass filters are typically used to smooth images to diminish some undesired patterns, or improve low resolution patterns for further analysis.

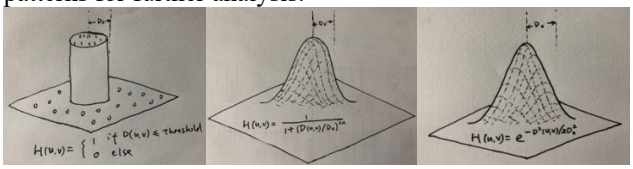

Fig 1: Three types of lowpass filters. Left: ideal lowpass filter; middle: Butterworth lowpass filter; right: Gaussian lowpass filter.

2. Highpass filters

Highpass filters work in an opposite way to apply sharpening to images. Similarly, the idea of ideal filter, Butterworth filter and Gaussian filter can be transitioned to highpass as well by 1 – lowpass filter. In perspective plot, highpass filters act as a funnel that rejects low frequency areas.

3. Band filters

Unlike lowpass or highpass filters, the band filters tends to allow only specific frequency bands to pass. It is extremely powerful at removing periodic noises that cannot be removed by spatial approaches. Other variants of band filters include notch filters that allows or rejects frequencies located at certain neighborhood area in a frequency spectrum.

II. Experiment

Frequency domain filtering is applied to two images to demonstrate their effects on image noises.

A. pattern.tif

First, some random Gaussian noises are added to the pattern image, shown in Fig 2.

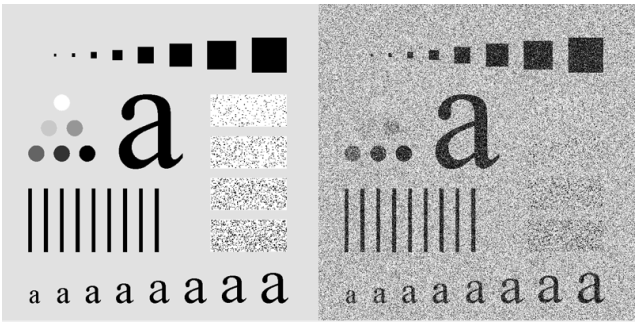

Fig 2: The original pattern image and after adding random Gaussian noise.

A Fourier transform is applied on the noised image, and the Fourier spectrum is displayed in Fig 3.

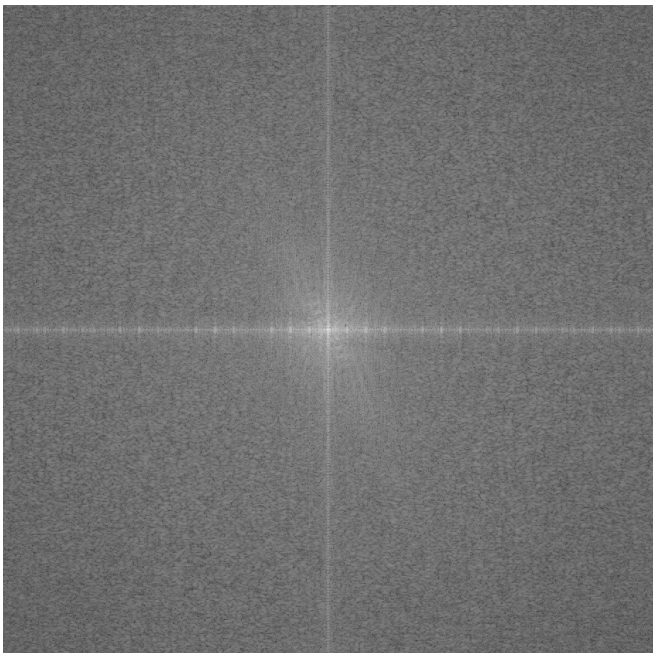

Fig 3: The Fourier spectrum of noised pattern image.

A Gaussian lowpass filter and a Butterworth highpass filter are applied on the image in frequency domain. The cutoff frequency is tuned for each filter. Fig 4 shows the results obtained by Gaussian lowpass filter at different cutoff frequencies.

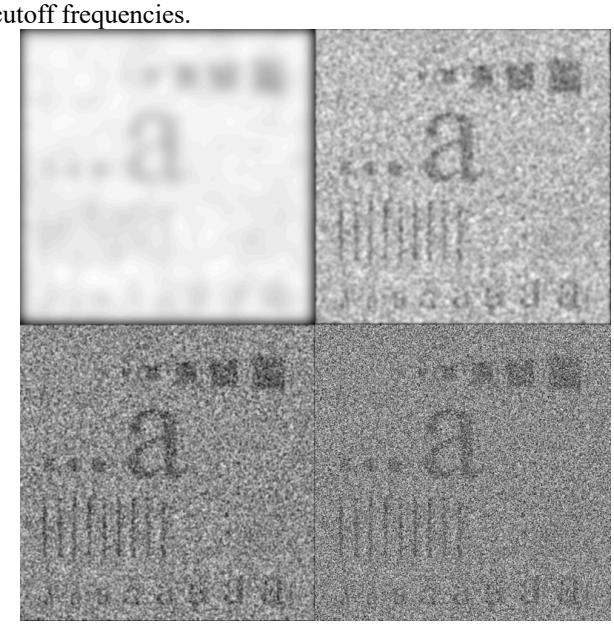

Fig 4: Spatial domain results after applying Gaussian lowpass filter with different cutoff frequencies. Top left: 0.01; top right: 0.05; bottom left: 0.1; bottom right: 0.3.

It can be observed that smaller cutoff values generate more blurred outputs, since more high frequency areas are prohibited to pass in frequency domain. These outputs do not work well on removing the noises.

Fig 5 demonstrates the effect of Butterworth highpass filter with different cutoff values.

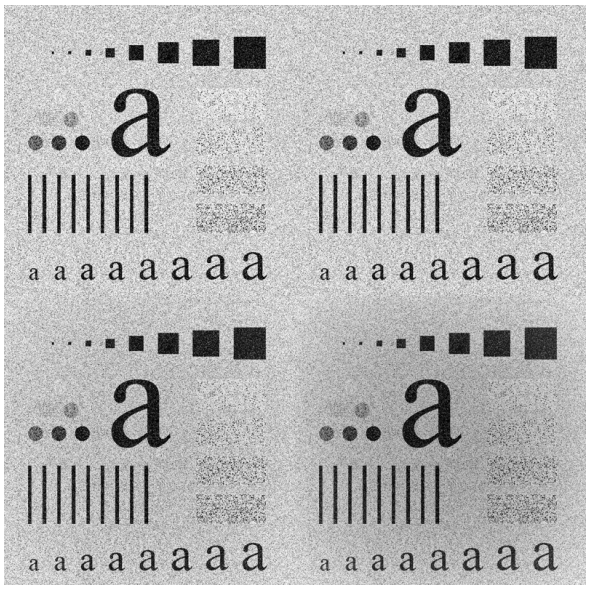

Fig 5: Spatial domain results after applying Butterworth highpass filter with different cutoff frequencies. Top left: 0.05; top right: 0.2; bottom left: 0.5; bottom right: 0.9.

Butterworth highpass filter increases the sharpness of the image. With the cutoff frequency rising up, it tends to preserve the prominent features and suppress other backgrounds.

B. space.tif

The Fourier spectrum is computed for the space image, displayed in Fig 6.

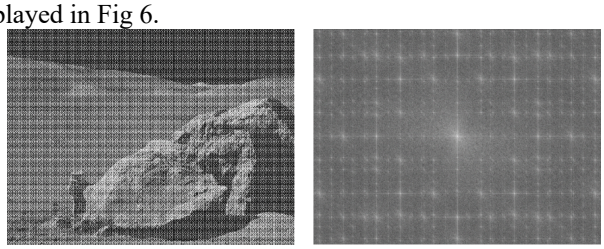

Fig 6: The original space image and its Fourier spectrum.

There're quite a few noise peaks scattered around the center point in the frequency spectrum, in a fixed interval. So we designed a notch array that programmatically covers a dot matrix that corresponds to the majority of prominent noise peaks in the Fourier spectrum. The result images in both spatial domain and frequency domain after applying the notch filters are displayed in Fig 7.

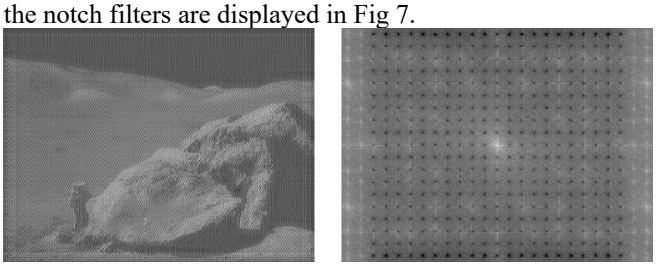

Fig 7: The filtered space image and its Fourier spectrum.

After applying notch filters on prominent noise peaks, most of the periodic noises are removed. While there are still some subtle noises caused by the tiny peaks existing in Fourier spectrum.

Lowpass filters are also attempted for this scenario. A Butterworth low pass filter with different cutoff frequencies are applied on the image, and Fig 8 demonstrates their difference.

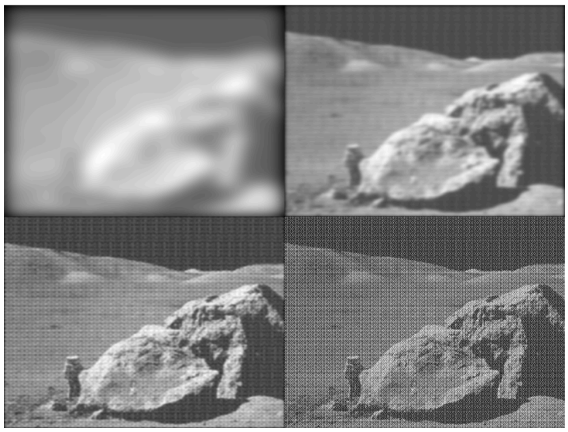

Fig 8: Result images after applying Butterworth lowpass filter at different cutoff frequencies. Top left: 0.01; top right: 0.05; bottom left: 0.1 bottom right: 0.5.

When lowpass filters are applied on the image, the noises are smoothed out at the cost of blurring the image as well, which is a tradeoff from non-selective filters.

III. Conclusion

In this paper, we first present a brief overview of the process of frequency domain filtering, as well as some commonly used filters including lowpass filter, high pass filter and bandpass filter.

Then different type of filters with varying parameters were applied to the pattern image to demonstrate their corresponding filtering effect. And finally we tuned a notch filter and a lowpass filter to alleviate the noises in the space image.

References

[1] M. Souden, J. Benesty and S. Affes, "On Optimal Frequency-Domain Multichannel Linear Filtering for Noise Reduction," in IEEE Transactions on Audio, Speech, and Language Processing, vol. 18, no. 2, pp. 260-276, Feb. 2010.

[2] Soo-Chang Pei and Chien-Cheng Tseng, "Two dimensional IIR digital notch filter design," in IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing, vol. 41, no. 3, pp. 227-231, March 1994.

APPENDIX

Code to calculate the Fourier spectrum of the pattern.tif:

```
pattern=imread('pattern.tif');
imshow(pattern)

PQ = paddedsize(size(pattern));
F=fft2(double(pattern),PQ(1),PQ(2));
Fc=fftshift(F);
S1=log(1+abs(Fc));
figure, imshow(S1,[])
```

Code to apply Gaussian lowpass filter to pattern.tif:

```
origin = imread('pattern.tif');
pattern = imnoise(origin, 'gaussian', 0, 0.1);

%Determine good padding for Fourier transform
PQ = paddedsize(size(pattern));
%Create a Gaussian Lowpass filter 5% the width of the Fourier transform
D0 = 0.05*PQ(1);

d_options = [0.01 0.03 0.05 2];
results = {};
for idx = 1:numel(d_options)
    ratio = d_options(idx);
    D0 = ratio * PQ(1);
    H = lpfilter('gaussian', PQ(1), PQ(2), D0);
    % Calculate the discrete Fourier transform of the image
    F = fft2(double(pattern), size(H,1),size(H,2));
    % Apply the highpass filter to the Fourier spectrum of the image
    lp = H.*F;
    % convert the result to the spacial domain.
    spacial=real(ifft2(lp));
    % Crop the image to undo padding
    spacial=spacial(1:size(pattern,1), 1:size(pattern,2));
    spacial=uint8(255 * mat2gray(spacial));
    results = [results, spacial];
end
figure
montage(results);
```

Code to apply Butterworth highpass filter to pattern.tif:

```
pattern = imread('pattern.tif');
pattern = imnoise(pattern, 'gaussian', 0, 0.1);

PQ = paddedsize(size(pattern));
D0 = 0.05*PQ(1);

d_options = [0.05 0.2 0.5 0.9];
results = {};
for idx = 1:numel(d_options)
    D0 = d_options(idx);
    H = hpfilter('btw', PQ(1), PQ(2), D0);
    F = fft2(double(pattern), size(H,1),size(H,2));
    lp = H.*F;
    spacial=real(ifft2(lp));
    spacial=spacial(1:size(pattern,1), 1:size(pattern,2));
    spacial=uint8(255 * mat2gray(spacial));
    results = [results, spacial];
end
```

```
figure
montage(results);
```

Code to denoise space.tif using notch filter:

```
space=imread('space.tif');
imshow(space);

PQ = paddedsize(size(space));
F=fft2(double(space),PQ(1),PQ(2));

%Create Notch matrix, ranging from -500 to 500, excluding the center point.
xs = linspace(-500,500,21);
ys = linspace(-500,500,21);
result = F;
for idx = 1:numel(xs)
    for idy = 1:numel(ys)
        x = xs(idx);
        y = ys(idy);
        if x == 0 && y == 0
            continue
        end
        H = notch('btw', PQ(1), PQ(2), 15, x, y);
        result = result.*H;
    end
end

% convert the result to the spacial domain.
spatial=real(ifft2(result));

spatial=spatial(1:size(space,1), 1:size(space,2));
figure, imshow(spatial,[])

% Display the Fourier Spectrum
Fc = fftshift(F);
Fcf = fftshift(result);

S1=log(1+abs(Fc));
S2=log(1+abs(Fcf));
figure, imshow(S1,[])
figure, imshow(S2,[])
```

Code to denoise space.tif using lowpass filter:

```
space=imread('space.tif');
PQ = paddedsize(size(space));
D0 = 0.05*PQ(1);

d_options = [0.01 0.05 0.1 0.5];
results = {};
for idx = 1:numel(d_options)
    ratio = d_options(idx);
    D0 = ratio * PQ(1);
    H = lpfilter('gaussian', PQ(1), PQ(2), D0);
    F = fft2(double(space), size(H,1),size(H,2));
    lp = H.*F;
    special = real(ifft2(lp));
    special = spacial(1:size(space,1), 1:size(space,2));
    special = uint8(255 * mat2gray(spacial));
    results = [results, spacial];
end
```

```
figure
montage(results);
```