Volume 4, Issue 8, August 2014 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Comparative Study of Different Filters on Images in Frequency Domain

Samayita Bhattacharya
Department of Computer Science & Engineering,
University of Kalyani, Kalyani, West Bengal, India

Kalyani Mali
Department of Computer Science & Engineering,
University of Kalyani, Kalyani, West Bengal, India

Abstract— In electronics, control systems engineering, and statistics, frequency domain is a term used to describe the domain for analysis of mathematical functions or signals with respect to frequency, rather than time. The reason for doing the filtering in the frequency domain is generally because it is computationally faster to perform two 2D Fourier transforms.

Keywords— Fingerprint, Frequency Domain, Fourier Transform, High Pass Filter, Low Pass Filter.

I. INTRODUCTION

Much signal processing is done in a mathematical space known as the frequency domain. In order to represent data in the frequency domain, some transform is necessary. Perhaps the most studied one is the Fourier transform. Any signal is composed of different frequencies. This applies to 1-dimensional signals such as an audio signal going to a speaker or a 2-dimensional signal such as an image.

The spatial frequency of an image refers to the rate at which the pixel intensities change. Figure 1 shows an image consisting of different frequencies. The high frequencies are concentrated around the axes dividing the image into quadrants. High frequencies are noted by concentrations of large amplitude swings in the small checkerboard pattern. The corners have lower frequencies. Low spatial frequencies are noted by large areas of nearly constant values.

The easiest way to determine the frequency composition of signals is to inspect that signal in the frequency domain. The frequency domain shows the magnitude of different frequency components. A simple example of a Fourier transform is a cosine wave.

Many different transforms are used in image, due to its wide range of applications in image processing, the Fourier transform is one of the most popular. It operates on a continuous function of infinite length. It is also possible to transform image data from the frequency domain back to the spatial domain. This is done with an inverse Fourier transform.

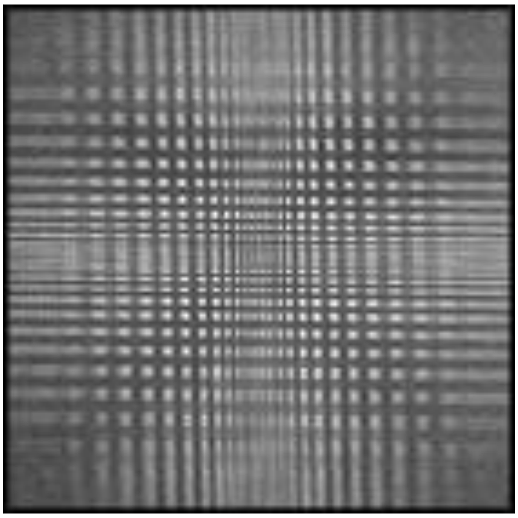


Figure 1: Image of varying frequencies.

II. WHY FREQUENCY DOMAIN

Frequency domain allows for techniques which could be used to determine the stability of the system. Also, these techniques can be use in conjunction with the S-domain (Laplace transform) which gives more insight to the stability of the system, transient response, and steady state response.

For a filter which has a large spatial extent, the frequency domain techniques will be generally faster. For an input with N elements and a filter with M elements where M is less than or equal to N, the spatial domain techniques generally perform on the order of N*M calculations and the frequency domain techniques perform N*log(N) calculations.

III. TYPES OF FREQUENCY DOMAIN

Frequency domain analysis can be one of the following types: (These are the most common transforms and the fields in which they are used)

- Fourier series – repetitive signals, oscillating systems
- Fourier transform – nonrepetitive signals, transients
- Laplace transform – electronic circuits and control systems
- Wavelet transform – digital image processing, signal compression
- Z transform – discrete signals, digital signal processing
- Gaussian Filter
- Gabor filter.

IV. FOURIER TRANSFORM

The Fourier Transform is an important image processing tool which is used to decompose an image into its sine and cosine components. The output of the transformation represents the image in the *Fourier* or frequency domain, while the input image is the spatial domain equivalent. In the Fourier domain image, each point represents a particular frequency contained in the spatial domain image. The Fourier Transform is used in a wide range of applications, such as image analysis, image filtering, image reconstruction and image compression.

For digital images the *Discrete Fourier Transform* (DFT) is used.

The DFT is the sampled Fourier Transform and therefore does not contain all frequencies forming an image, but only a set of samples which is large enough to fully describe the spatial domain image. The number of frequencies corresponds to the number of pixels in the spatial domain image. For a square image of size N×N, the two-dimensional DFT is given by:

$$F(k,l) = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} f(i,j)\, e^{-\iota 2\pi\left(\frac{ki}{N}+\frac{lj}{N}\right)} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \; (1)$$

where $f(a,b)$ is the image in the spatial domain and the exponential term is the basis function corresponding to each point $F(k,l)$ in the Fourier space. The equation can be interpreted as: the value of each point $F(k,l)$ is obtained by multiplying the spatial image with the corresponding base function and summing the result.

The basis functions are sine and cosine waves with increasing frequencies, *i.e.* $F(0,0)$ represents the DC-component of the image which corresponds to the average brightness and $F(N-1,N-1)$ represents the highest frequency.

V. WHAT DO FREQUENCIES MEAN IN AN IMAGE

If an image has large values at *high* frequency components then the data is changing rapidly on a short distance scale. *e.g.* a page of text.

If the image has large *low* frequency components then the large scale features of the picture are more important. *e.g.* a single fairly simple object which occupies most of the image.

For colour images, the measure (now a 2D matrix) of the frequency content is with regard to colour/chrominance: this shows if values are changing rapidly or slowly. Where the fraction or value in the frequency matrix is low, the colour is changing gradually. Now the human eye is insensitive to gradual changes in colour and sensitive to intensity. So we can ignore gradual changes in colour and throw away data without the human eye noticing.

VI. HOW TRANSFORMING INTO THE FREQUENCY DOMAIN HELPS

Any function can be decomposed into purely sinusoidal components (sine waves of different size/shape) which when added together make up the original signal. Thus transforming a signal into the frequency domain allows us to see what sine waves make up the signal. More complex signals will give more complex graphs but the idea is exactly the same. The graph of the frequency domain is called the frequency spectrum.

VII. FOURIER TRANSFORM AND IMAGE PROCESSING

The Fourier Transform produces a complex number valued output image which can be displayed with two images, either with the *real* and *imaginary* part or with *magnitude* and *phase*. In image processing, often only the magnitude of the Fourier Transform is displayed, as it contains most of the information of the geometric structure of the spatial domain image. However, if we want to re-transform the Fourier image into the correct spatial domain after some processing in the frequency domain, we must make sure to preserve both magnitude and phase of the Fourier image.

The Fourier domain image has a much greater range than the image in the spatial domain. Hence, to be sufficiently accurate, its values are usually calculated and stored in float values. The Fourier Transform is used if we want to access the geometric characteristics of a spatial domain image. Because the image in the Fourier domain is decomposed into its sinusoidal components, it is easy to examine or process certain frequencies of the image, thus influencing the geometric structure in the spatial domain.

The result shows that the image contains components of all frequencies, but that their magnitude gets smaller for higher frequencies. Hence, low frequencies contain more image information than the higher ones. The transform image also tells us that there are two dominating directions in the Fourier image, one passing vertically and one horizontally through the center. These originate from the regular patterns in the background of the original image.

VIII. HIGH PASS AND LOW PASS FILTERS

Applying a low pass filter in the frequency domain means zeroing all frequency components above a cutoff frequency. Applying a high pass filter frequency domain is the opposite to the low pass filter, that is, all the frequencies below some cutoff radius are removed. Apparently higher noise levels are false, and the graphs are auto scaled and thus the field only appears larger because of the removal of the low frequency components.
We did implement and check the output of different low pass and high pass filtering. Those filters are:

- Ideal low pass and high pass filter,
- Gaussian low pass and high pass filter,
- Butterworth low pass and high pass filters.

A. Ideal low pass filter

A **low-pass filter** is an electronic filter that passes low-frequency signals but attenuates signals with frequencies higher than the cutoff frequency. The actual amount of attenuation for each frequency varies from filter to filter. There are many different types of filter circuits, with different responses to changing frequency. The frequency response of a filter is generally represented using a Bode plot, and the filter is characterized by its cutoff frequency and rate of frequency rolloff. In all cases, at the *cutoff frequency,* the filter attenuates the input power by half or 3 dB. So the **order** of the filter determines the amount of additional attenuation for frequencies higher than the cutoff frequency.
A **first-order filter**, for example, will reduce the signal amplitude by half (so power reduces by a factor of 4), or 6 dB, every time the frequency doubles (goes up one octave); more precisely, the power rolloff approaches 20 dB per decade in the limit of high frequency. The magnitude Bode plot for a first-order filter looks like a horizontal line below the cutoff frequency, and a diagonal line above the cutoff frequency.
A **second-order filter** attenuates higher frequencies more steeply.
Third-and higher-order filters are defined similarly.

B. Butterworth low pass filter

The Butterworth filter is a type of signal processing filter designed to have as flat a frequency response as possible in the passband so that it is also termed a maximally flat magnitude filter. Butterworth showed that a low pass filter could be designed. The gain $G(\omega)$ of an n-order Butterworth low pass filter is given in terms of the transfer function $H(s)$ as

$$G^2(\omega) = |H(j\omega)|^2 = \frac{{G_0}^2}{1 + \left(\frac{\omega}{\omega_c}\right)^{2n}} \quad \text{……….. (2)}$$

Where n = order of filter, ω_c = cutoff frequency (approximately the -3dB frequency), G_0 is the DC gain (gain at zero frequency).

C. Gaussian filter

In electronics and signal processing, a Gaussian filter is a filter whose impulse response is a Gaussian function. Gaussian filters are designed to give no overshoot to a step function input while minimizing the rise and fall time. This behavior is closely connected to the fact that the Gaussian filter has the minimum possible group delay. Mathematically, a Gaussian filter modifies the input signal by convolution with a Gaussian function.
The one-dimensional Gaussian filter has an impulse response given by

$$g(x) = \sqrt{\frac{a}{\pi}} \cdot e^{-a \cdot x^2} \quad \text{……….. (3)}$$

or with the standard deviation as parameter

$$g(x) = \frac{1}{\sqrt{2 \cdot \pi} \cdot \sigma} \cdot e^{-\frac{x^2}{2\sigma^2}} \quad \text{……….. (4)}$$

In two dimensions, it is the product of two such Gaussians, one per direction:

$$g(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \quad \text{……….. (5)}$$

Where, x is the distance from the origin in the horizontal axis, y is the distance from the origin in the vertical axis, and σ is the standard deviation of the Gaussian distribution.

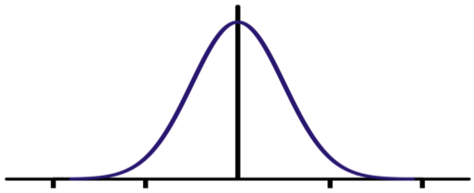


Figure 2: Shape of a typical Gaussian filter

D. High pass filter

A **high-pass filter** (HPF) is a device that passes high frequencies and attenuates (i.e., reduces the amplitude of) frequencies lower than its cutoff frequency. A high-pass filter is usually modeled as a linear time-invariant system.
All the frequency filters used for low pass filter can be used for high pass filtering as well.

IX. STEPS OF FREQUENCY DOMAIN AND FILTERING IN FREQUENCY DOMAIN (FOURIER TRANSFORM):

With the frequency domain techniques, care needs to be taken in choosing the size of the input region: certain sizes (those that have small prime factors) are handled much more efficiently. By default, this is done automatically by adding additional elements so that the input can be handled efficiently and then stripping these additional elements off before writing the result. To change this behavior, modification the edge handling and output size inputs.
Frequency domain techniques treat the right edge (after padding) as contiguous with the left edge; the top and bottom are handled similarly. This can be problematic when the values along the edges are very different. Preprocessing the data to remove background trends which cause the edges to be different or padding the data with additional elements so that the number of elements added is greater than the approximate spatial extent of the filter are two possible approaches to mitigate this problem.
Filtering in the frequency domain is a common image and signal processing technique. It can smooth, sharpen, de-blur, and restore some images.
There are three basic steps to frequency domain filtering:

- The image must be transformed from the spatial domain into the frequency domain using the Fast Fourier transform.
- The resulting complex image must be multiplied by a filter (that usually has only real values).
- The filtered image must be transformed back to the spatial domain.

X. EXPERIMENTS AND RESULTS

A given function or signal can be converted between the time and frequency domains with a pair of mathematical operators called a transform. An example is the Fourier transform, which decomposes a function into the sum of a (potentially infinite) number of sine wave frequency components. The 'spectrum' of frequency components is the frequency domain representation of the signal. The inverse Fourier transform converts the frequency domain function back to a time function.
A **spectrum analyzer** is the tool commonly used to visualize real-world signals in the frequency domain.
We have used different pictures in frequency domain and check out the outputs after high pass and low pass filtering in frequency domain.
MSE: In statistics, the mean squared error (MSE) of an estimator is one of many ways to quantify the difference between values implied by a kernel density estimator and the true values of the quantity being estimated. MSE is a risk function, corresponding to the expected value of the squared error loss or quadratic loss. MSE measures the average of the squares of the "errors.
For images mean squared error (MSE) for two $m \times n$ monochrome images I and K where one of the images is considered a noisy approximation of the other is defined as:

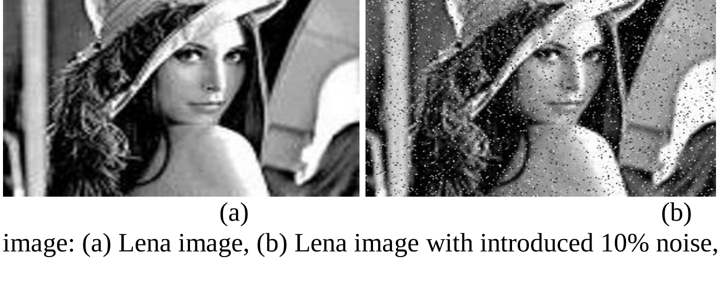
$$MSE = \frac{1}{m\,n}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}[I(i,j) - K(i,j)]^2$$ …………… (6)

PSNR: The phrase peak signal-to-noise ratio, often abbreviated PSNR, is an engineering term for the ratio between the maximum possible power of a signal and the power of corrupting noise that affects the fidelity of its representation. Because many signals have a very wide dynamic range, PSNR is usually expressed in terms of the logarithmic decibel scale. The PSNR is defined as:

$$PSNR = 10 \cdot \log_{10}\left(\frac{MAX_I^2}{MSE}\right) = 20 \cdot \log_{10}\left(\frac{MAX_I}{\sqrt{MSE}}\right)$$ ……………………….. (7)

Typical values for the PSNR in lossy image and video compression are between 30 and 50 dB, where higher is better.

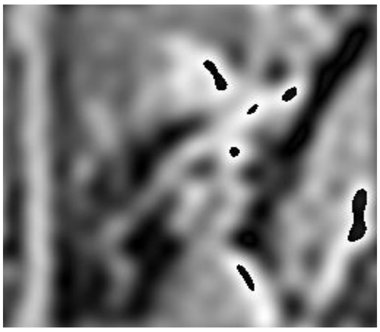
A. Low pass filters



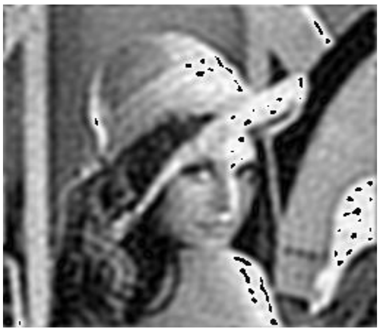
(a) (b)

Figure 3: Sample image: (a) Lena image, (b) Lena image with introduced 10% noise, used for filtering

Output after filtering: Ideal Low Pass Filter



Ideal10



Ideal30

Ideal50

Ideal100

Output after filtering: Butterworth Low Pass Filter

Butterworth 10

Butterworth 30

Butterworth 50

Butterworth 100

Output after filtering: Gaussian Low Pass Filter

Gaussian 10

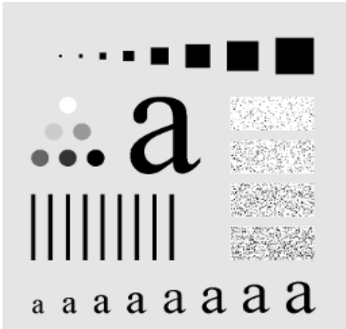
Gaussian 30


Gaussian 50

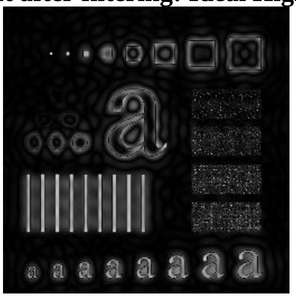

Gaussian 100

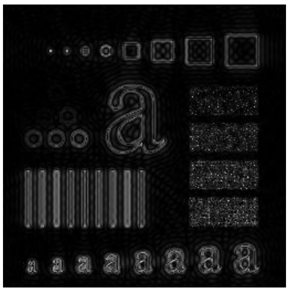
B.

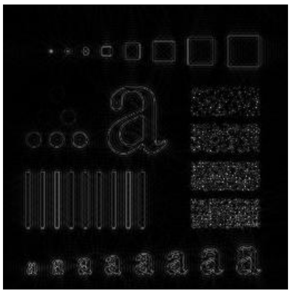
C. High Pass filters

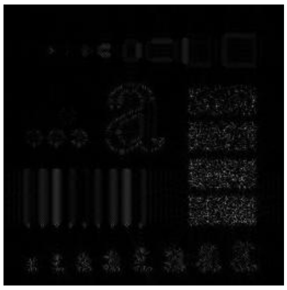

Input Image

Output after filtering: Ideal High Pass Filter

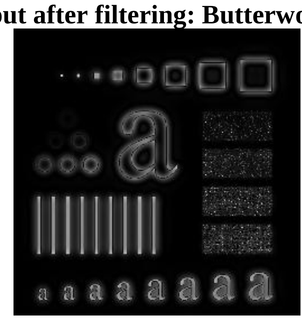

Ideal high pass diameter 15

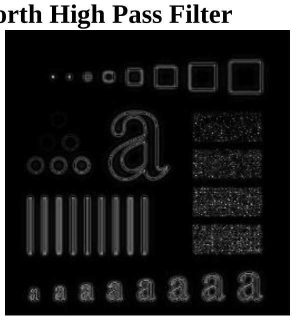

Ideal high pass diameter 30

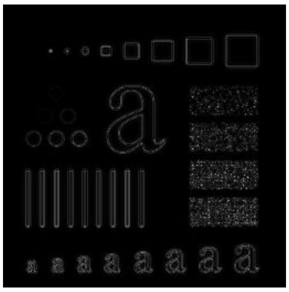

Ideal high pass diameter 50

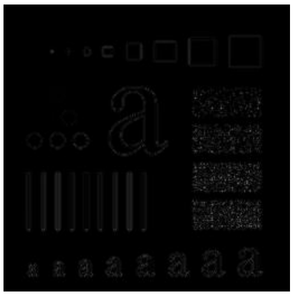

Ideal high pass diameter 100

Output after filtering: Butterworth High Pass Filter

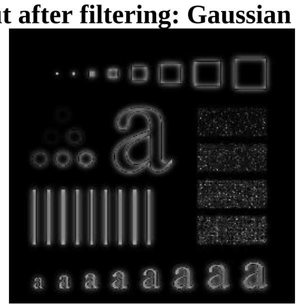

Butterworth 15

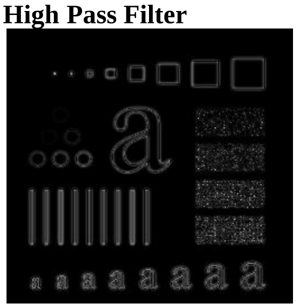

Butterworth 30

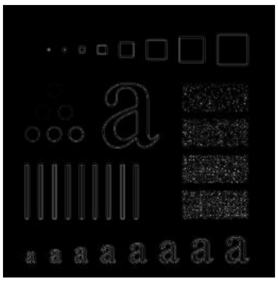

Butterworth 50

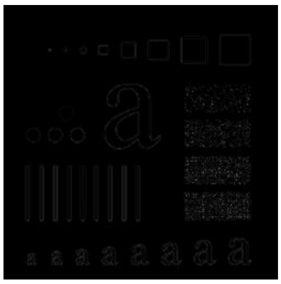

Butterworth 100

Output after filtering: Gaussian High Pass Filter


Gaussian 15


Gaussian 30


Gaussian 50


Gaussian 100

D. Comparative Analysis

By analysing the different filters we can see Gaussian filter gives the best result amongst the filters used for the experiment.

Table1: Low pass filters comparative results

Ideal10.pgm	The MSE = 2656.824463	The PSNR(dB) = 13.887175
Ideal30.pgm	The MSE = 1698.208862	The PSNR(dB) = 15.830893
Ideal50.pgm	The MSE = 1700.746460	The PSNR(dB) = 15.824408
Ideal100.pgm	The MSE = 2029.941040	The PSNR(dB) = 15.055969
butterworth10.pgm order(2)	The MSE = 1885.225708	The PSNR(dB) = 15.377170
butterworth30.pgm order(2)	The MSE = 1144.279053	The PSNR(dB) = 17.545485
butterworth50.pgm order(2)	The MSE = 975.545410	The PSNR(dB) = 18.238329
butterworth100.pgm order(2)	The MSE = 1022.252502	The PSNR(dB) = 18.035221
gaussian10.pgm	The MSE = 1754.614258	The PSNR(dB) = 15.688987
gaussian30.pgm	The MSE = 1045.872314	The PSNR(dB) = 17.936016
gaussian50.pgm	The MSE = 722.998108	The PSNR(dB) = 19.539433
gaussian100.pgm	The MSE = 445.886261	The PSNR(dB) = 21.638563

REFERENCES

[1] H. C. Lee and R. E. Gaensslen, Eds., Advances in Fingerprint Technology. New York: Elsevier, 1991.
[2] Rafeal C. Gonzalez, Richard E. Woods, “Digital Image Processing,” Pearson Education Asia, 2002.
[3] R. C. Gonzalez and P. Wintz. Digital Image Processing. Addison Wesly, Menlo Park, Ca. second edition, 1987.

