Low-pass filter in frequency domain of Digital Image Processing

Amir Mahmood Mousavi _ 3971242026

***Abstract*-- the frequency domain refers to the analysis of mathematical functions or signals with respect to frequency, rather than time . Given this fact, we want to apply a low pass filter on an image in frequency domain .**
This article aims to provide a comprehensive tutorial and survey about review of the Fourier transform and its usage in digital image processing in frequency domain .

***Index Terms*— frequency domain, digital image processing – Fourier transform, low-pass , filter, Gaussian .**

I. Introduction

Frequency filters process an image in the frequency domain. The image is Fourier transformed, multiplied with the filter function and then re-transformed into the spatial domain. Attenuating high frequencies results in a smoother image in the spatial domain, attenuating low frequencies enhances the edges.

All frequency filters can also be implemented in the spatial domain and, if there exists a simple kernel for the desired filter effect, it is computationally less expensive to perform the filtering in the spatial domain. Frequency filtering is more appropriate if no straightforward kernel can be found in the spatial domain, and may also be more efficient.

II. Fourier transformed

The concept behind the Fourier transform is that any waveform can be constructed using a sum of sine and cosine waves of different frequencies. The exponential in the above formula can be expanded into sines and cosines with the variables u and v determining these frequencies.

The inverse of the above discrete Fourier transform is given by the following equation:

$$f(x,y) = \frac{1}{MN}\sum_{u=0}^{M-1}\sum_{v=0}^{N-1} F(u,v)e^{i2\pi\left(\frac{ux}{M}+\frac{vy}{N}\right)}$$

Thus, if we have F(u,v), we can obtain the corresponding image (f(x,y)) using the inverse, discrete Fourier transform.

III. How it works

Frequency filtering is based on the Fourier Transform. (For the following discussion we assume some knowledge about the Fourier Transform, therefore it is advantageous if you have already read the corresponding worksheet.) The operator usually takes an image and a filter function in the Fourier domain. This image is then multiplied with the filter function in a pixel-by-pixel fashion:

$$G(k,l) = F(k,l)H(k,l)$$

where F(k,l) is the input image in the Fourier domain, H(k,l) the filter function and G(k,l) is the filtered image. To obtain the resulting image in the spatial domain, G(k,l) has to be re-transformed using the inverse Fourier Transform.
Since the multiplication in the Fourier space is identical to convolution in the spatial domain, all frequency filters can in theory be implemented as a spatial filter. However, in practice, the Fourier domain filter function can only be approximated by the filtering kernel in spatial domain.

The form of the filter function determines the effects of the operator. There are basically three different kinds of filters: low-pass, high-pass and band-pass filters. A low-pass filter attenuates high frequencies and retains low frequencies unchanged. The result in the spatial domain is equivalent to that of a smoothing filter; as the blocked high frequencies correspond to sharp intensity changes, i.e. to the fine-scale details and noise in the spatial domain image.
A high-pass filter, on the other hand, yields edge enhancement or edge detection in the spatial domain, because edges contain many high frequencies. Areas of rather constant gray level consist of mainly low frequencies and are therefore suppressed.
A band-pass attenuates very low and very high frequencies, but retains a middle range band of frequencies. Band-pass filtering can be used to enhance

edges (suppressing low frequencies) while reducing the noise at the same time (attenuating high frequencies).

The most simple lowpass filter is the ideal lowpass. It suppresses all frequencies higher than the cut-off frequency and leaves smaller frequencies unchanged:

$$H(k,l) = \begin{cases} 1 & if \sqrt{k^2 + l^2} < D_0 \\ 0 & if \sqrt{k^2 + l^2} > D_0 \end{cases}$$

. In most implementations, is given as a fraction of the highest frequency represented in the Fourier domain image.

IV. LOW-PASS FILTER

Low-pass filters:

- create a blurred (or smoothed) image
- attenuate the high frequencies and leave the low frequencies of the Fourier transform relatively unchanged

The corresponding formula and visual representations of the filter is shown in the Fig . 1 . In the formula, D0 is a specified nonnegative number. D(u,v) is the distance from point (u,v) to the center of the filter.

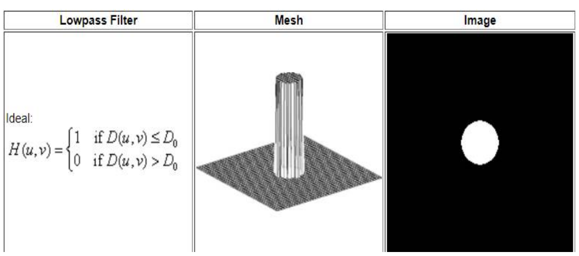

Fig . 1 .

.

V. IMPLEMENTATION OF SINGLE LAYER PERCEPTRON

Algorithm for filtering in the frequency Domain

- Step1: Given ain input image f(x,y) of size M x N, obtain the padding parameters P and Q. Typically, we select P = 2M and Q = 2N
- Step2: Form a padded image fp(x,y) of size P X Q by appending the necessary number of zeros to f(x,y).
- Step3: Multiply fp(x,y) by (-1)^(x+y)
- Step4: Compute the DFT, F(u,v) of the image from Step 3
- Step5: Generate a Real, Symmetric Filter Function H(u,v) of size P X Q with center at coordinates (P/2,Q/2),
- Step 6:Form the product G(u,v) = H(u,v)F(u,v) using array multiplication Obtain the processed image
- Step 7: gp(x,y) = {real{inverse DFT[G(u,v)]}(-1)^(x+y)
- Step 8: Obtain the final processed result g(x,y) by extracting the M X N region from the top, left quadrant of gp(x,y).

4.7.3 Summary of Steps for Filtering in the Frequency Domain

The material in the previous two sections can be summarized as follows:

1. Given an input image $f(x, y)$ of size $M \times N$, obtain the padding parameters P and Q from Eqs. (4.6-31) and (4.6-32). Typically, we select $P = 2M$ and $Q = 2N$.
2. Form a padded image, $f_p(x, y)$, of size $P \times Q$ by appending the necessary number of zeros to $f(x, y)$.
3. Multiply $f_p(x, y)$ by $(-1)^{x+y}$ to center its transform.
4. Compute the DFT, $F(u, v)$, of the image from step 3.
5. Generate a real, symmetric filter function, $H(u, v)$, of size $P \times Q$ with center at coordinates $(P/2, Q/2)$.[†] Form the product $G(u, v) = H(u, v)F(u, v)$ using array multiplication; that is, $G(i, k) = H(i, k)F(i, k)$.
6. Obtain the processed image:
$$g_p(x, y) = \left\{\text{real}\left[\Im^{-1}[G(u, v)]\right]\right\}(-1)^{x+y}$$
where the real part is selected in order to ignore parasitic complex components resulting from computational inaccuracies, and the subscript p indicates that we are dealing with padded arrays.
7. Obtain the final processed result, $g(x, y)$, by extracting the $M \times N$ region from the top, left quadrant of $g_p(x, y)$.

Fig . 2 .

The code has been written in MATLAB , here we have the results , shown in Fig.3.:

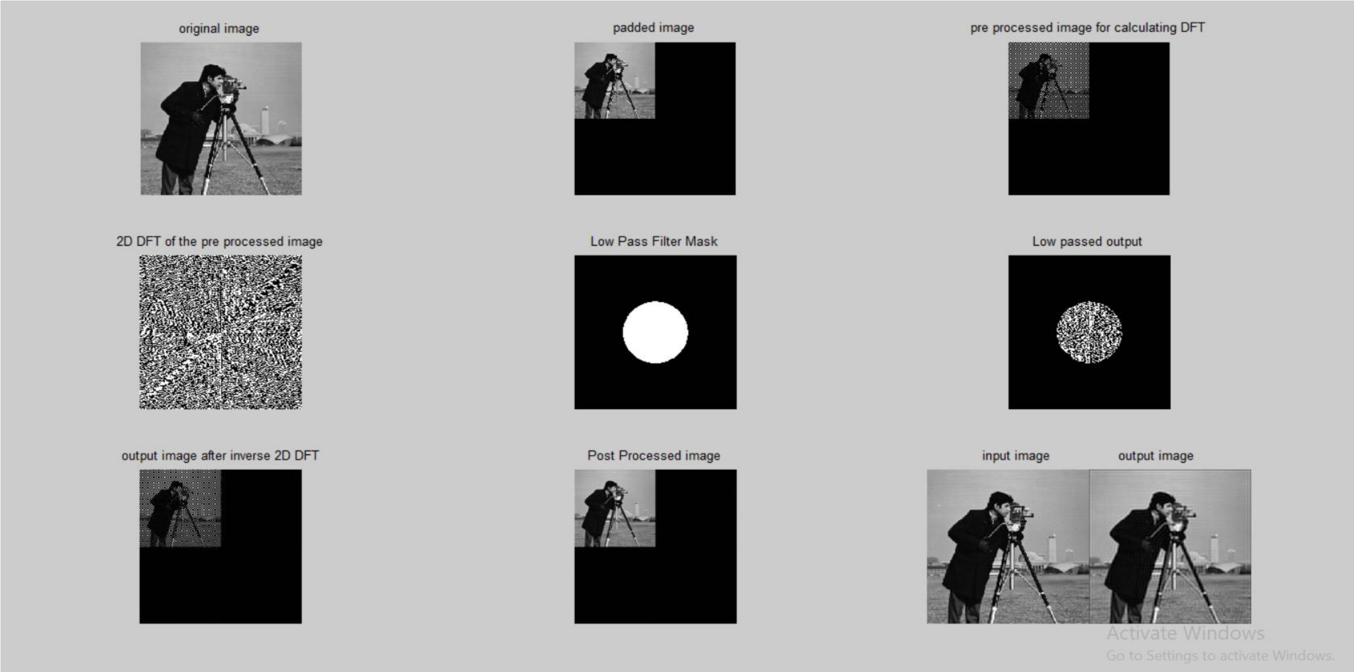

Fig . 3

VI. Conclusion

Image enhancement is the prime aspect of digital image processing. Image filtering is an very important step in image enhancement. In our paper we implemented low pass . We analyzed the various results as shown, after successful simulation of the filter. its performance depends on selection of cutoff frequency. Other than it, Ideal filter is the desire filter for both the image smoothing and sharpening. This can be achieved using higher order Butterworth filter. Especially for large images frequency domain filtering is much faster. High pass filter preserves the edges by image sharpening and shows only sharp transition of pixel intensity. While low pass filter denoises the image by smoothing the image and preserves image detail. Selection of cutoff frequency and filter order for Butterworth filter gives variable performance. Gaussian filter gives normal generalized performance exponentially

VII. References

[1] https://pdfs.semanticscholar.org/3371/66e45402ddc0266737a9242182c7f163c716.pdf

[2] https://stackoverflow.com

[3] https://homepages.inf.ed.ac.uk/rbf/HIPR2/freqfilt.htm

[4] https://www.youtube.com/watch?v=ZSvJHHZ-HDI

[5] https://www.youtube.com/watch?v=0lTSERUFuQQ