SCHOLARLY REVIEW ONLINE

Scholarly Review Online - Winter 2024/2025

Digital Image Processing using the Fast Fourier Transform

By Hanson Hanchu Xiong

AUTHOR BIO

A senior from Keystone Academy Beijing. Interested in studying applied and computational math, completed most of the study of linear algebra, and attended multiple math camps over the last two years. At the same time, he has been participating in and leading the community service of visually impaired people by creating a club at school and building a website incorporating the accessibilities for them to share their life stories and learn from others as well.

ABSTRACT

Image processing is an important technique with diverse applications such as medical treatment, screen reading etc. This paper is aimed to gain insights into how the type of filters applied influence the quality of the image. Through the use of the Fast Fourier Transform algorithm and convolution, the effectiveness of denoising filters (e.g., Low-Pass, Gaussian, Smoothing, and HighBoost Sharpening) are examined.The result reveals the combined filter to have the best quality with significant improvement on denoising and sharpening.

INTRODUCTION

Image processing has become a critical tool across diverse fields, including medical imaging, remote sensing, multimedia applications, and security systems. A primary challenge in this domain is image denoising, which aims to remove unwanted distortions or noise from digital images [Gonzalez and Woods, 2018]. Noise, introduced by factors such as sensor imperfections, environmental conditions, or transmission errors, can significantly degrade image quality and information content [Buades et al., 2005].

Effective denoising is crucial for enhancing the accuracy and reliability of image analysis and decisionmaking processes in various applications. In medical imaging, it improves disease detection and diagnosis, while in remote sensing, it enhances the interpretation of satellite or aerial imagery for land-use mapping and environmental monitoring. For multimedia and security applications, denoising improves image clarity, facilitating better object recognition, surveillance, and authentication [Maini and Aggarwal, 2010].

Over the years, previous studies have developed numerous image denoising algorithms, each with unique strengths and limitations. These algorithms can be broadly categorized into filtering-based methods, transform-domain techniques, and learning-based approaches [Kang et al., 2015]. Filteringbased methods, such as Gaussian, median, and bilateral filters, aim to smooth the image while preserving important features [Tomasi and Manduchi, 1998]. Transform-domain techniques exploit multiscale or frequency-domain representations to identify and suppress noise components [Donoho and Johnstone, 1995]. In this paper, we will typically focus on the transform-domain method and evaluate the efficiency of different filters applied to the image.

METHODOLOGY

Each image consists of an infinite linear combination of waves with different frequencies, encoded as complex exponentials with different phases. The Fourier Transform decomposes an image into its constituent waves, where a pixel's position is denoted by its x- and y-frequencies, while its intensity is conveyed through amplitude. This way of encoding an image is said to present the frequency (or Fourier) domain.

This domain serves as the foundation for numerous image filters employed for tasks like noise reduction, image sharpening, pattern analysis, and feature extraction.

Fourier Transform

Fourier Transform is the type of generalization of the complex Fourier series, it is used in mathematical functions and could be used to represent the frequency domain.

To begin with, the general formula of Fourier Transform for function f(x) = F(ω is

$$F(\omega) = \int_{-\infty}^{\infty} f(t) e^{-j\omega t} dt$$

One of the main properties of the Fourier transform is the fact that it is an invertible operation. The formula for the Inverse Fourier Transform is given by:

$$f(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} F(\omega) e^{j\omega t} d\omega$$

FFT is a very important tool for effectively computing the Discrete Fourier Transform (DFT) of an image. The DFT is the Fourier Transform in discrete space that allows the convolution to be implemented onto a finite array. The DFT is particularly well-suited for image processing, since images are encoded as arrays of discrete pixels. [HeckBert, 1995] The DFT of a 2D image could be presented mathematically: Given an N ×M image represented as a matrix of pixel values X = ($x_{n,m}$), where n and m are the row and column indices respectively, the DFT is computed as follows:

$$X_{k,l} = \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} x_{n,m} e^{-2\pi i \left(\frac{nk}{N} + \frac{ml}{M}\right)}$$

Where

- $X_{k,l}$ is the DFT output at frequency indices k and l.
- N and M are the dimensions of the image (height and width).
- The exponential term represents the complex sinusoidal basis functions, where $e^{-2\pi i\theta}$ corresponds to a rotation in the complex plane.

The DFT decomposes the image into its frequency components: Low-frequency components represent the smooth variations in the image (e.g., large areas of uniform color); and high-frequency components represent rapid changes (e.g., edges, noise).

However, if the DFT is calculated directly, its time complexity would be very high, meaning that the time taken for processing will be high and therefore reduce the efficiency. This is because the direct calculation involves iterating over all pixels for each frequency component: For an N × M image, the naive computation requires:

- Outer sum (over k): N iterations
- Inner sum (over l): M iterations
- Inner nested sum (over n and m): N × M iterations

Thus, the total complexity becomes:

$O(N \times M \times N \times M) = O(N^2M^2)$

The time complexity for DFT is $O(N^2M^2)$, which is relatively deficient for higher-resolution images. One of the main solutions to this problem is to use the **Fast Fourier Transform**, which is an optimization of the FFT algorithm with significantly reduced time complexity :

1. Divide: Split the DFT into smaller DFTs by separating even and odd indexed pixels.
2. Conquer: Recursively compute the DFTs of the smaller sub-problems.
3. Combine: Combine the results of the smaller DFTs to obtain the final DFT.

For a 2D image, this can be done in two stages: Firstly, the convolution is used to compute the 1D FFT along the rows. Second, it is used to compute the 1D FFT along the columns of the resulting matrix, which enables the complete image to be convolved.

This results in a computational complexity of:

$O(N \times M \log(N \times M)) = O(NM \log(NM))$

For larger images, this produces a distinct improvement in the efficiency of the algorithm, and without FFT algorithm. The FFT algorithm can be easily implemented using the FFT function

```
# Apply FFT to the noisy image
f_noisy = np.fft.fft2(noisy_image)
fshift_noisy = np.fft.fftshift(f_noisy)
```

Figure 1: The Numpy's FFT Implementation

Convolution

One of the main tools in digital image processing is the use of image filters, which change image pixels based on the values of the neighboring pixels. One mathematical function that enables such transformations based on local pixel values is the convolution operation. Convolution is a mathematical operation that combines two functions to produce a third function, representing how the shape of one function is modified by the other. In the context of signal processing, convolution is used to apply filters to signals or images, allowing for operations such as smoothing, sharpening, and edge detection. For two continuous functions f(t) and g(t), the convolution $(f * g)(t)$ is defined as:

$$(f * g)(t) = \int_{-\infty}^{\infty} f(\tau) g(t - \tau)\, d\tau$$

For discrete signals, the convolution operation is defined as:

$$(x * h)[n] = \sum_{m=-\infty}^{\infty} x[m]h[n-m]$$

Here, x[n] is the input signal, h[n] is the impulse response (or filter), and the result (x∗h)[n] is the output signal. The summation runs over all values of m, effectively combining the input signal with the filter. [Ian T. Young; Jan.J.Gerbrands, 2007]

One of the most significant aspects of convolution is its relationship with the Fourier Transform. The most important result from this area is known as the Convolution Theorem, which states that convolution in the time (or spatial) domain corresponds to multiplication in the frequency domain. Specifically, if F(ω) and G(ω) are the Fourier Transforms of f(t) and g(t), respectively, then the Fourier Transform of the convolution (f ∗ g)(t) is given by:

$$F\{f * g\} = F(\omega)G(\omega)$$

This relationship is particularly powerful because it allows us to perform convolution operations more efficiently using the Fast Fourier Transform (FFT).

The formula of convolution could be generalized by replacing the single integral with multiple integrals: $(f * g)(x_1, \ldots, x_n) = \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} f(\tau_1, \ldots, \tau_n) g(x_1 - \tau_1, \ldots, x_n - \tau_n)\, d\tau_1 \cdots d\tau_n$.[Ian T. Young; Jan.J.Gerbrands, 2007]

Convolution has several important properties that make it a powerful tool in signal processing:

1. Commutative: f ∗ g = g ∗ f
2. Associative: f ∗ (g ∗ h) = (f ∗ g) ∗ h
3. Distributive: f ∗ (g + h) = f ∗ g + f ∗ h

These properties allow for more flexible operations in manipulating signals and filters.

Image Processing using FFT

With the preliminary background information provided that convolution and the computation of DFT are contributing factors of applying the filters onto the FFT algorithm. In this section, it will dive deeper into different types of filter being applied. All of these filters have the same structure: first, convert an image into the Fourier domain, then multiply by some function (called a mask), and then take the inverse Fourier transform.[Ian T. Young; Jan.J.Gerbrands, 2007]

Low-Pass Filter

The low-pass filter is a type of filter that only allows the points with low frequency to pass through the mask, and reduces all the high-frequency components from passing through. Normally, the high-frequency component corresponds to the small-scale variations in the image. In particular, noise, which often appears as small deviations of pixel brightness from their original values is an example of such variation, so one would expect a low-pass filter to effectively remove noise, i.e. act as an image denoising filter. Thus, by removing the high-frequency it achieves the purpose of denoising. In simplest words, this low-pass filter could be achieved by setting up a cutoff boundary that allows the low-frequency component signal to pass and block the ones with high frequency. [Fisher, 2024]

```
rows, cols = noisy_image.shape
crow, ccol = rows // 2, cols // 2
mask = np.zeros((rows, cols), np.uint8)
r = 30
for i in range(rows):
    for j in range(cols):
        dist = np.sqrt((i - crow)**2 + (j - ccol)**2)
        if dist < r:
            mask[i, j] = 1
```

Figure 2: The implementation of the low-pass filter

Gaussian Smoothing Filter

A Gaussian smoothing filter, it is a 2 dimensional convolution operator that smooths out the image and removes the noise with gradient. The Gaussian filter convolve the original image with a Gaussian function The 2D Gaussian function can be expressed mathematically as:

$$G(x, y) = A \cdot e^{-\left(\frac{(x-\mu_x)^2}{2\sigma_x^2} + \frac{(y-\mu_y)^2}{2\sigma_y^2}\right)}$$

Gaussian functions are special because the Fourier transform of a Gaussian is a Gaussian, so a Gaussian filter corresponds to multiplication by a Gaussian function on the Fourier side. In particular, this acts similar to a low-pass filter because it attenuates the importance of higher frequencies in the image. Theoretically, different from the low-pass filter, the Gaussian smoothing filter is non-zero in all the components, suggesting an infinitely large convolution kernel. However, most of the time the components will be set to 0 if it is equal to or more than 3 standard deviations from the mean. [Fisher et al., 2003]

```
rows, cols = image.shape
crow, ccol = rows // 2, cols // 2
x = np.arange(cols) - ccol
y = np.arange(rows) - crow
X, Y = np.meshgrid(x, y)
sigma = 50
gaussian_mask = np.exp(-(X**2 + Y**2) / (2*sigma**2))
gaussian_mask = gaussian_mask / np.sum(gaussian_mask)
```

Figure 3: The implementation of Gaussian function

This explains the code in figure 3in which the Gaussian mask created in the code follows the exact format of the Gaussian function. By multiplying the mask with the FFT it will add the mask for filtering. By the convolution theorem, involving an image with Gaussian is equivalent to in a frequency spectrum, it multiplies another Gaussian function which smooths out the image, thereby removing its noise.

High Boost/Sharpening Filter

The high boost filter is a type of sharpening filter that emphasizes the higher frequency components while maintaining the lower frequency so that the edges and corners will be more significant while the details in the image will not be lost. [GeeksforGeeks, 2022] In figure 4 it defines the HBF kernel that

```
A = 1.5  # Boosting factor
high_boost_kernel = np.array([[-1, -1, -1],
                              [-1, A+8, -1],
                              [-1, -1, -1]])
```

Figure 4: The implementation of the High boost filter

has A value = 1.5 and the central = A+8. Furthermore, the surroundings will all be -1 to keep the low-frequency components by creating a Laplacian-like operator that enhances the high frequency as well.

Combination of High-Boost and Gaussian smoothing

Lastly, it is hypothesized that the low-pass filter and Gaussian smoothing filter will reduce the noise better than the sharpening filter but sacrifice the sharpness and clarity of the image at the same time. Therefore, the filter that combines smoothing and sharpening is expected to work out the best in terms of balancing the noise removal and clarity of the image.

RESULTS

After applying the Fast Fourier Transform algorithm and different types of filters to it, we yielded the processed outcome of an input image. Some of the filters will be specifically focusing on blurring or sharpening the image. While some filters effectively remove the noise of the image, some of them do not. In this section, the effectiveness of the quality of the processed image output will be assessed and evaluated holistically according to the previous hypothesis regarding every filter. For every trial, some extra Gaussian noise is added for a clearer and more obvious demonstration.

Low-Pass Filter

As seen in Figure 5, the filter is successfully created as how it was intended to be: the low-pass filter mask has been constructed circularly located in the center of the image. The white section in this mask suggests the section where this region is selected to have value ”1” mentioned in the methodology section (being retained using the mask), whereas the rest of the region is completely black, which corresponds to the mechanism of the low-pass filter that all parts except the circular region have the high-frequency components being attenuated and suppressed.

The frequency component distribution could be seen from the spectrum of the noisy image. There are many darker dots on the spectrum, representing the high-frequency component and the lighter part suggests the lower frequency component. After applying the filter it could be seen that only a small portion with low-frequency was retained, all the high-frequency are removed, suggesting successful processing with the low-pass filter.

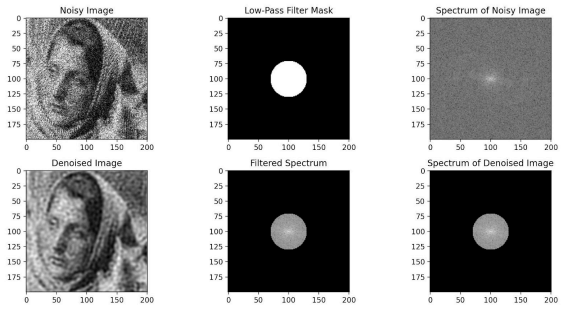

Figure 5: Output image using low-pass filter.

Gaussian blur filter

As mentioned in the methodology section, the Gaussian filter blurs the edge by applying the Gaussian function onto the frequency domain.

The Gaussian mask follows the ”bell curve” that outputs the balanced and smoothed pixels. From the Gaussian mask shown in Figure 6, the same thing could be observed: there is a clear gradient change from the center of the mask, which corresponds to the mathematical principle behind the Gaussian. The center of the mask contains the mean pixel and it gradually blurs depending on how many sigma from the center. The further a pixel is from the center mean pixel, the low-frequency it will be reduced to.

Thus, after applying the mask, the spectrum of the filtered image contains very little amount of high-frequency component, suggesting that this filter has significantly reduced the noise of the image. This is reflected in the quality of the filtered image in which a lot of the black spots on the original image are removed, especially the spots on the face. (see Figure 6)

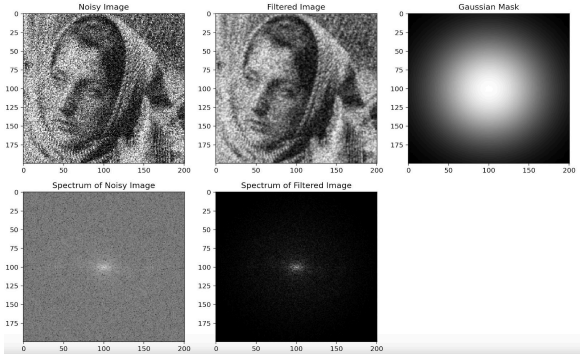

Figure 6: Image output using Gaussian blur filter

High-boost filter

The high boost filter enhances the high-frequency components indicated as the white pixels. With the scaling factor it could amplify these pixels to make them sharpened. The spectrum of the enhanced image is revealed, and the low-frequency contains little significance to the image. The enhanced image looks darker comparing to the noisy image because there is greater contrast among the high and low frequency pixel (see Figure 7).

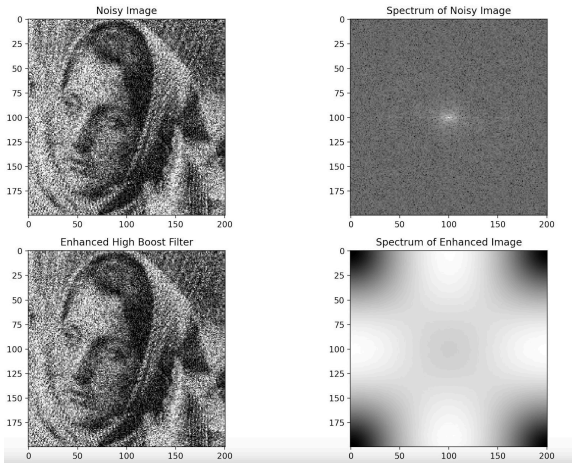

Figure 7: Image output with high-boost filter

Combination of Gaussian filter and Sharpening filter

This is an output that combines the sharpening filter and the blurring filter. The high-boost kernel is slightly modified compared to the one in the previous section by changing its

scaling factor. It could be seen from the image's spectrum that they are overlapped to form the new spectrum of the combined filter, it could therefore be seen in Figure 8 that the combined filter indeed performed both sharpening and smoothing process.

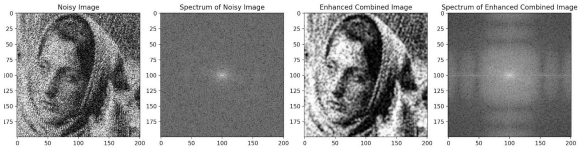

Figure 8: Combined image output

DISCUSSION

Overall, all four filters are working as they are intended to be in terms of the prediction based on the code implementation.

Firstly, starting with filters that are best for removing the high-frequency(noise) on the image: the low-pass filter passes only the low frequency through the filter; the Gaussian filter acts similarly to the low-pass filter in which it removes the noise using the bell curve, the more distant from the center mean, the less it will be smoothed out. Compared to the low-pass filter, the Gaussian filter will perform a smoother removal of the noise instead of a sharp cutoff. From the quality of the images, both filters have successfully removed some noises in which the image looks smoother, especially the Gaussian filter.

Secondly, the high boost filter did not work out well, even though some minor enhancement could be seen, there is negligible difference between the enhanced image and noisy image, suggesting that this filter will not have a significant effect on its own. This is probably due to the color of the original image which is black and white, making it very hard to observe the enhanced component since the darkened color is still black.

Despite that, the enhanced combined image that consists of the use of a Gaussian smooth filter and high boost filter did make significant progress. From the human eye, it greatly improved the clarity compared to the rest of the filter. Thus, evaluating the quality of the image, this combined filter presents the best outcome. However, there are still many limitations of the filter. It could be observed that there are still many spots on the image that make the quality lower, this is probably caused by the order of applying the filter. Since the high-boost filter is applied first, it makes the frequency component higher, thus when the smoothing filter is applied later it cannot remove a lot of noise because these components have been enhanced. This issue might be addressed by changing the order of applying the filters so that there will be less noise remaining on the image before sharpening.

Lastly, this investigation only focused on the gray-scale of a relatively complex image (which is a woman's face), and the noise applied to the demonstration is only Gaussian standard noise, limiting

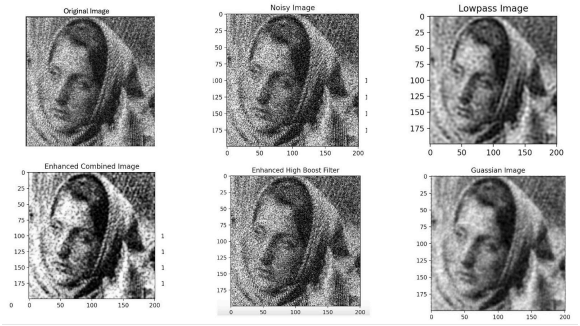

Figure 9: Result of all the images

the scope of the application of these filters. Therefore, it might be possible that their performances might vary according to the type of noise on the image, the color pixels, the complexity of the image, etc. These variables could be the focus in the future study as to expand the scope of the result interpreted from the current investigation.

CONCLUSION

In conclusion, by applying different filtering methods, such as low-pass filters and Gaussian smoothing filters, the clarity and detail of an image can be significantly improved.

Low-pass filters, which effectively attenuate high-frequency noise while maintaining the integrity of low-frequency information, are an important tool in

applications such as medical imaging and remote sensing. Meanwhile, Gaussian smoothing not only reduces noise but also maintains the natural appearance of the image through gradual transition characteristics, which could be viewed as a more advanced low-pass filter.

On the other hand, high-boost filtering enhances edge details by emphasizing high-frequency components while preserving low frequencies, although it could enhance the image to make it clear, this filter did not remove the noise in the image, therefore causing the quality of the image to still be poor.

To improve that, the combination of Gaussian smoothing and high-boost filtering techniques provides an innovative solution to achieve the purpose of removing noise while maintaining image sharpness, and the same as the hypothesis discussed above, it returns the image with the best quality.

In conclusion, this study emphasizes the importance of selecting the appropriate filtering technique based on the specific imaging requirements. The integration of FFT with various filtering methods provides a powerful framework for improving image processing capabilities, which ultimately improves the accuracy of analysis and decision-making in different domains.

ACKNOWLEDGEMENT

I would like to acknowledge my academic mentor and my teaching assistant for guiding me through the paper writing. Additionally, I have used Grammarly to correct grammar mistakes, and AI Assistant to help me convert mathematical equations into Latex form in the methodology section and improved the overall language quality.

REFERENCES

[Buades et al., 2005] Buades, A., Coll, B., and Morel, J.-M. (2005). A review of image denoising algorithms, with a new one. Multiscale Modeling & Simulation, 4(2):490–530.

[Donoho and Johnstone, 1995] Donoho, D. L. and Johnstone, I. M. (1995). Adapting to unknown smoothness via wavelet shrinkage. Journal of the American Statistical Association, 90(432):1200– 1224.

[Fisher et al., 2003] Fisher, R., Perkins, S., Walker, A., and Wolfart, E. (2003). Gaussian smoothing. [Online; accessed 2024.9.1].

[Fisher, 2024] Fisher, R. B. (2024). The simplest lowpass filter.

[GeeksforGeeks, 2022] GeeksforGeeks (2022). Image sharpening using laplacian filter and high boost filtering in matlab. [Online; accessed 2024.8.24].

[Gonzalez and Woods, 2018] Gonzalez, R. C. and Woods, R. E. (2018). Digital Image Processing. Pearson, New York, 4th edition.

[HeckBert, 1995] HeckBert, P. (1995). Fourier Transform and Fast fourier transform (fft) algorith. Computer Graphics.

[Ian T. Young; Jan.J.Gerbrands, 2007] Ian T. Young; Jan.J.Gerbrands, L. J. v. V. (2007). Fundamentals of image processing. Delft University of Technology.

[Kang et al., 2015] Kang, L.-W., Lin, C.-Y., and Fu, Y.-H. (2015). Automatic single-image-based rain streaks removal via image decomposition. IEEE Transactions on Image Processing, 24(4):1742–1755.

[Maini and Aggarwal, 2010] Maini, R. and Aggarwal, H. (2010). A comprehensive review of image enhancement techniques. Journal of Computing, 2(3):8–13.

[Tomasi and Manduchi, 1998] Tomasi, C. and Manduchi, R. (1998). Bilateral filtering for gray and color images. In Proceedings of the Sixth International Conference on Computer Vision, pages 839–846. IEEE.