ENGDATA201 – Image Processing and Computer Vision

Project 2 - Discrete Fourier Transform in Image Processing

Authors:

Joanikij Chulev

March, 2024

Professor:

Dr. Ir. Frank
van der Stappen

Abstract

The Fourier Transform, a fundamental mathematical operation, converts signals from the time domain to the frequency domain. Discrete Fourier Transform (DFT) is essential for discrete signals like digital images. Magnitude and phase information of DFT coefficients provide insights into signal characteristics. DFT finds extensive application in signal processing and image analysis, with Fast Fourier Transform (FFT) offering efficient computation methods. We present an overview of DFT's implementation and its significance in image computation with our own examples.

Contents

Experimental platform

Hardware technology platform:

OS Name: Microsoft Windows 11 Pro - Windows version: 22H2

System Type: x64-based PC

Processor: Intel(R) Core (TM) i7-1065G7 CPU @ 1.30GHz, 1498 Mhz, 4 Core(s), 8 Logical Proc.

Installed Physical Memory (RAM): 4.00 GB

GPU: Integrated Intel GPU (Intel Graphics)

Software technology platform:

Spyder version: 5.4.3 (conda)

Python version: 3.10.17 64-bit

Qt version: 5.15.2

PyQt5 version: 5.15.7

Operating System: Windows 11

Library - OpenCV: cv2.

1 Introduction

Signals and systems form the backbone of modern communication, control, and signal processing technologies. In essence, signals are representations of physical quantities that vary with time, space, or any other independent variable. They can be analog, like continuous audio waves, or digital, such as discrete sequences used in computer systems. Mathematically, signals are often described using functions or sequences, where time or space is the independent variable. These functions can be continuous (like sinusoidal waves) or discrete (like sampled data points). Systems, on the other hand, are mathematical entities that transform input signals into output signals. They can represent anything from electronic circuits and mechanical devices to algorithms used in digital processing. These signals can be represented as periodic functions. The main idea is that any periodic function can be decomposed into a summation of sines and cosines.

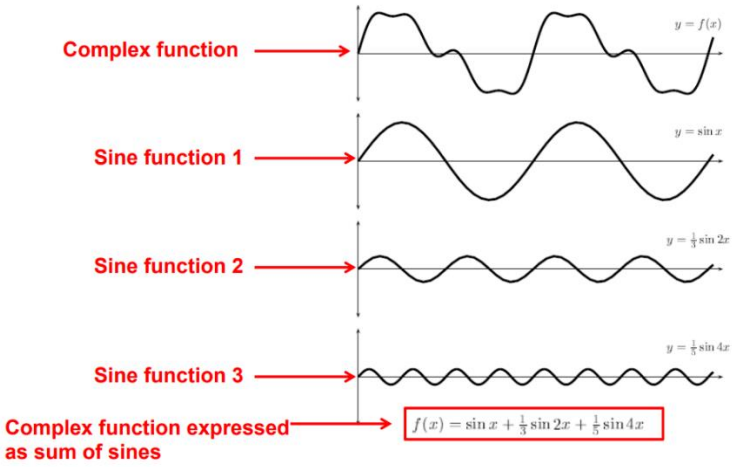


Figure 1: Example of a function expressed as a sum of sines.

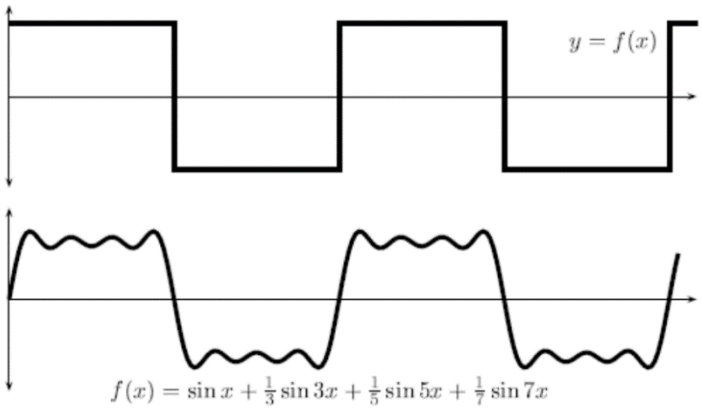


Figure 2: Example of a square signal function expressed as a sum of sines.

This fact was discovered by the French mathematician and physicist Jean Baptiste Joseph Fourier (1768−1830) and the work was published in 1822. It has been regarded as one of the most revolutionary contributions of the nineteenth century.

The Fourier Transform is essential as it simplifies the analysis of complex signals by breaking them down into simpler functions. This mathematical approach allows for a clearer understanding of how transmission mediums and noise affect signals, making it easier to analyze and manipulate signal or function characteristics.

2 Fourier Transform

Essentially, Fourier Transform is a mathematical operation for converting a signal from time domain into its frequency domain.

The time domain refers to the representation of signals in terms of amplitude variations over time. It provides insights into signal behavior at different instants. On the other hand, the frequency domain represents signals in terms of their frequency components, revealing the constituent sinusoidal frequencies and their respective amplitudes. Understanding a signal's frequency content often provides insights into its behavior and characteristics that are not readily apparent in the time domain.

- The big picture!

	Continuous Time	Discrete Time
Fourier Series	$x(t) = \sum_{k=-\infty}^{\infty} a_k e^{jk\omega_0 t}$ continuous and periodic in time $a_k = \frac{1}{T}\int_T x(t)e^{-jk\omega_0 t}dt$ discrete and aperiodic in frequency	$x[n] = \sum_{k=<N>} a_k e^{jk\frac{2\pi}{N}n}$ discrete and periodic in time $a_k = \frac{1}{N}\sum_{n=<N>} x[n]e^{-jk\frac{2\pi}{N}n}$ discrete and periodic in frequency
Fourier Transform	$x(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} X(j\omega)e^{j\omega t}d\omega$ continuous and aperiodic in time $X(j\omega) = \int_{-\infty}^{\infty} x(t)e^{-j\omega t}dt$ continuous and aperiodic in frequency	$x[n] = \frac{1}{2\pi}\int_{2\pi} X(e^{j\omega})e^{j\omega n}d\omega$ discrete and aperiodic in time $X(e^{j\omega}) = \sum_{n=-\infty}^{\infty} x[n]e^{-j\omega n}$ continuous and periodic in frequency

Table 4.1: Comparison of Fourier series and Fourier transform.

Figure 3: Quick overview of Fourier formulas and their comparisons and attributes.

As computers process information discreetly, we will stick to the discrete Fourier Transform. Below we can see the general formula for crossing over a function/ signal to the Frequency domain from the time domain. Discrete-time signals are sequences of values defined at specific discrete points in time, such as digital data samples (such as images) taken at regular intervals. These signals are denoted by $x[n]$ where n represents the discrete time index. In contrast for continuous we use the notation of t for continuous time.

- Discrete-time Fourier transform:

$$x[n] = \frac{1}{2\pi} \int_{\langle 2\pi \rangle} X\left(e^{j\omega}\right) e^{j\omega n} d\omega \Longleftrightarrow X\left(e^{j\omega}\right) = \sum_{n=-\infty}^{+\infty} x[n] e^{-j\omega n},$$

Figure 4: DFT formula for 1D signals.

Note that the Image is a discrete 2D function. Thus, we need to extend the Fourier transform from 1D to 2D. Thus, we get the following expression for the 2D Fourier transform:

$$F(u,v) = \iint_{-\infty}^{\infty} f(x,y) e^{-i2\pi(ux+vy)} dxdy$$

Figure 5: CFT of a 2D function.

Where $f(x,y)$ is the image function and x and y are the spatial coordinates, and u, v are frequencies along x and y respectively.

This would hold true if digital images were continuous, which they are not. They are discrete. Thus, we need to map this function into its discrete form:

$$F[p,q] = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f[m,n] e^{-i2\pi pm/M} e^{-i2\pi qn/N}$$

Figure 6: DFT a 2D function.

We get that m and n are the spatial coordinates, while p (also labeled k) and q (also labeled l) are the frequency coordinates for m and n, respectively.

This is the image mapping of sinusoidal functions where k and l are noted as the frequency coordinates p and q, respectively. Now we get a sense of how these frequencies would look like images.

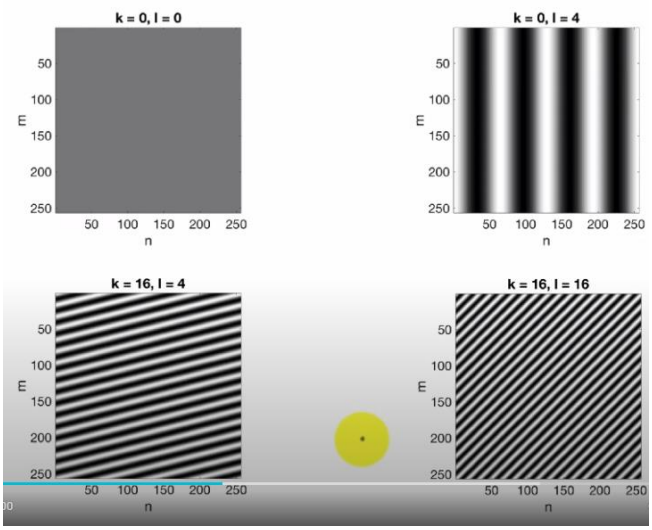


Figure 7: Various sinusoidal waves as images.

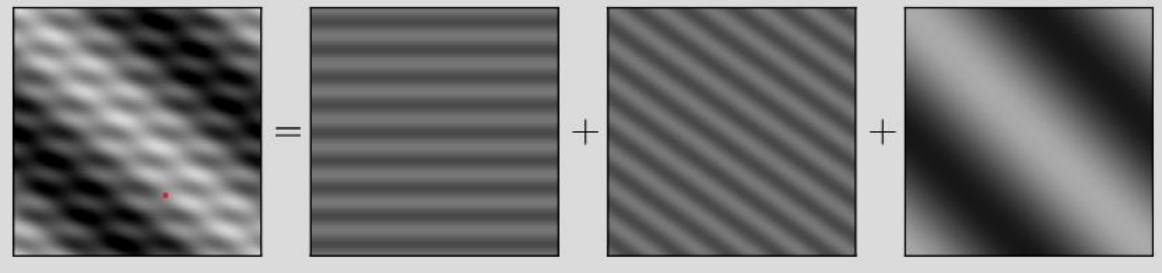


Figure 8: The decomposition of a synthetic image into oscillations. In this toy-example, the image is simple enough to be decomposed by using only three oscillations.

3 Magnitude and Phase of FT Images

The definition indicates that the Fourier transformation of an image can be complex. Every complex number can also be represented as:

$$Z = x + iy = re^{j\theta}$$

Where r is the magnitude (real number), $r = abs(Z)$ and θ is the phase.

In a typical visualization or representation of the Discrete Fourier Transform (DFT) output images, both magnitude and phase information are usually not directly included in a single image or as separate images. In this context, both magnitude and phase information are utilized. The magnitude of the DFT coefficients provides insights into the amplitude or strength of the frequency components present in a signal. The phase of the DFT coefficients reflects the timing or alignment of the sinusoidal components at different frequencies. We can see that as images below:

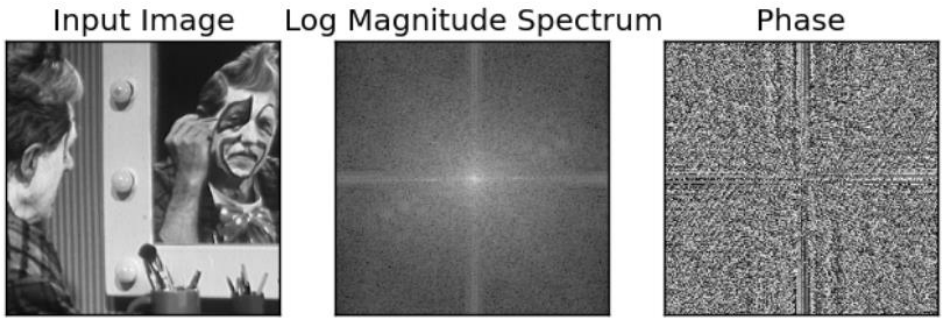


Figure 9: Source image, Magnitude of the DFT image, Phase of the DFT image.

We need both values to successfully contrast a DFT image and apply the inverse which we will do later.

4 Application of DFT in Images

Let us implement the Fourier transform algorithm function in action. Note that the resulting output will also be in 2D, thus we will get another image.

- Consider DFT of image with single edge
- For display, DC component shifted to center
- Log of magnitudes of Fourier Transform displayed

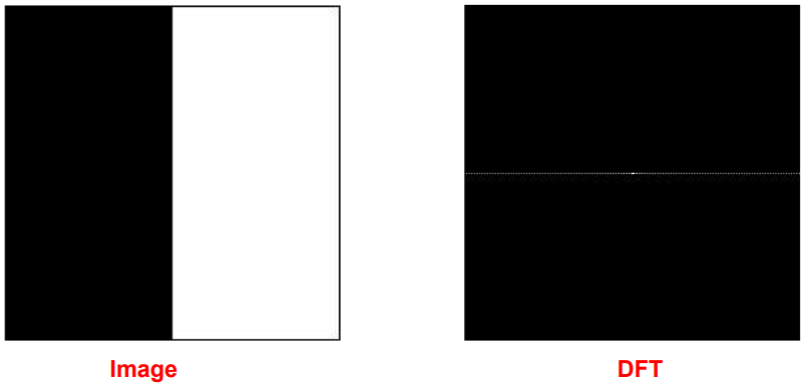


Figure 10: Simple example of applied DFT to an image.

N

ow let us explain in detail with a square example. See figure below:

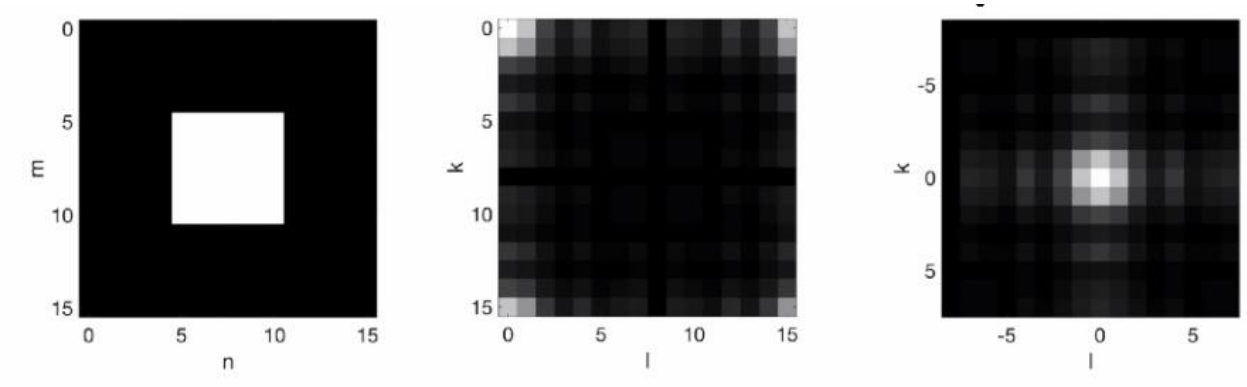


Figure 10: Source image, DFT image, Adjusted DFT image for better visuals.

We can see the Fourier transform of the small example image of the square. The amplitude is very large in the square region, in discrete numeral terms 255, and outside the square the amplitude is very low in discrete numeral terms 0. Image 2 shows the Fourier transformation of the simple example image. The value 0,0 in the second image represents the average value of the image when k=0 and l=0.

For visualization of the DSF images we would always use something like the third image for symmetricity in the origin. We can arrange the values from image 2 to resemble image 3, since complex sinusoids can be represented differently. For example, complex sinusoids with frequencies 15/16 and -1/16 are the same in this context. Thus, we can put 0 in the center on both axes. Moreover, for more complex images that we will explore later, the range of the DFT coefficients is very large and we will need to use a logarithmic scale representation of those DFT coefficients.

We now include more examples of DFT images with strong edges. For example, this high-res square image. It has very strong horizontal and vertical edges, thus when represented in the frequency domain these would require the opposite: vertical and horizontal frequencies, respectively.

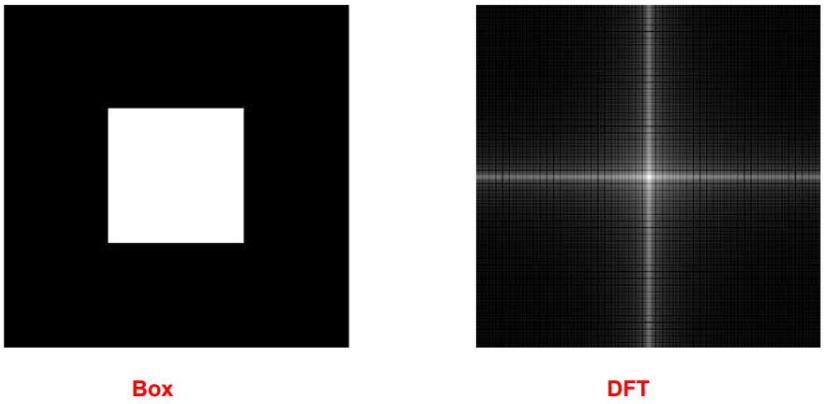


Figure 11: Source image, DFT image.

The same logic would go for an image of a rotated square or a diamond. The frequencies would need to be rotated diagonally due to the diagonal edges but in the reverse direction of the edges.

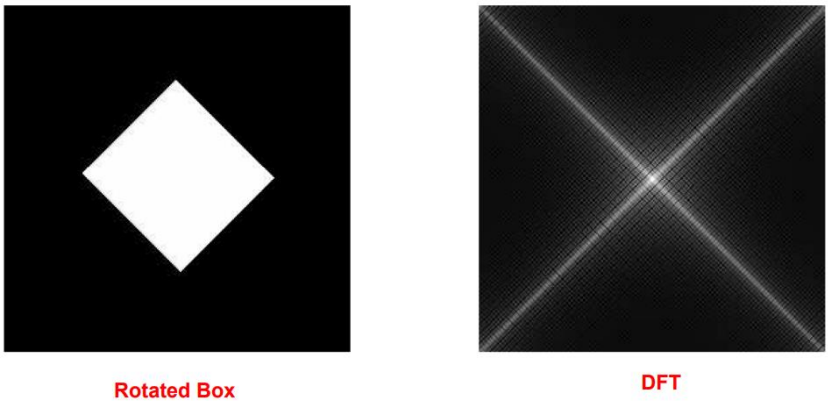


Figure 12: Source image, DFT image.

Finally, we include the figure below to demonstrate how a centered spectrum would look like in a higher resolution practical image, to contrast with our simple example in figure 10.

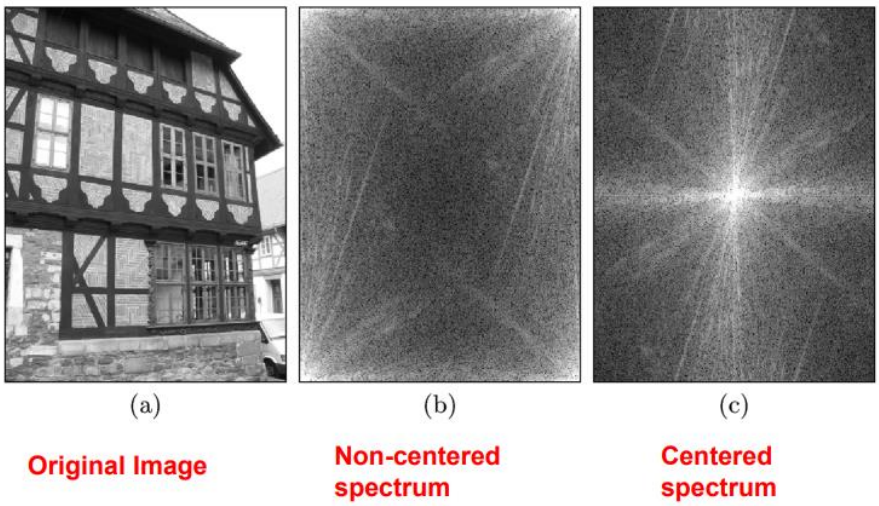


Figure 12: Practical example of a better DFT visualization, using log scale as well.

5 Inverse Fourier Transform in Images

The opposite of the Fourier transform can also be done, it takes the output we'd obtain from it and recovers the initial functions from the Fourier transform. If we assume that a Fourier statement is

integrable and f is also integrable and continuous then we can state the inverse as an integral like such for a 2D function:

$$f(x,y) = \iint_{-\infty}^{\infty} F(u,v)e^{i2\pi(xu+yv)}dudv$$

Figure 13: Inverse CFT.

Note that now the output would be spatial coordinates x and y. In general, the use cases for this can often simplify complicated problems and make them easy to solve in the frequency domain, thus you can inverse FT that transformed solution to get the initial solution we want. Flowsheet of this:

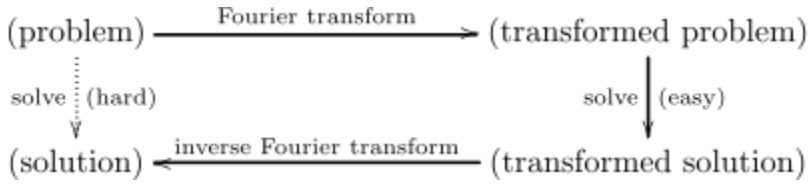


Figure 13: Flowsheet for using Inverse FT.

The use cases go past signal reconstruction. It is also applied to images to find the original source image which is what we will demonstrate. Of course, once again, Image Functions are 2D and discrete functions. Thus, we need to use the Discrete Inverse FT. We have:

$$f[m,n] = \frac{1}{MN}\sum_{p=0}^{M-1}\sum_{q=0}^{N-1} F[p,q]e^{i2\pi pm/M}e^{i2\pi qn/N}$$

Figure 14: Inverse DFT.

Using this formula, we can traverse between the DFT image and the normal source image with spatial image coordinates.

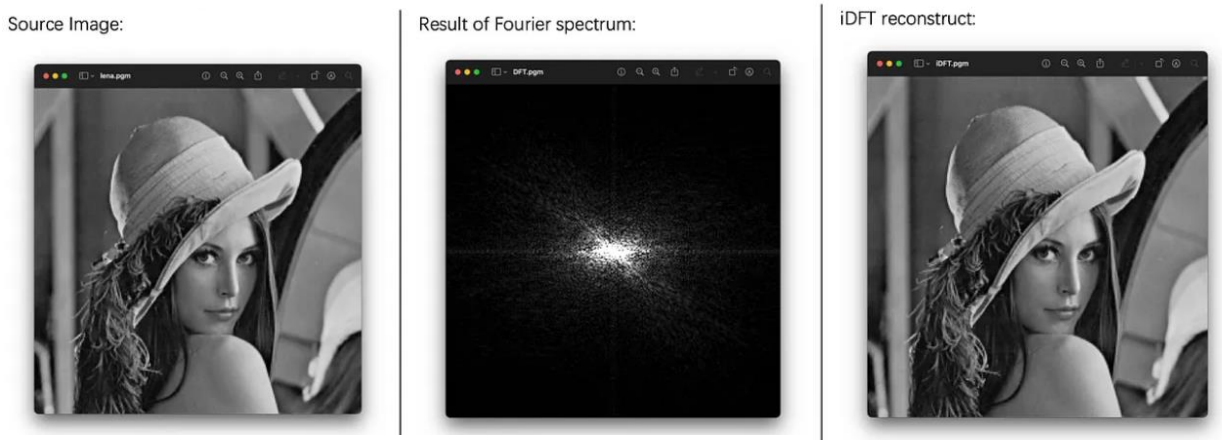


Figure 15: Practical use of Inverse DFT in images.

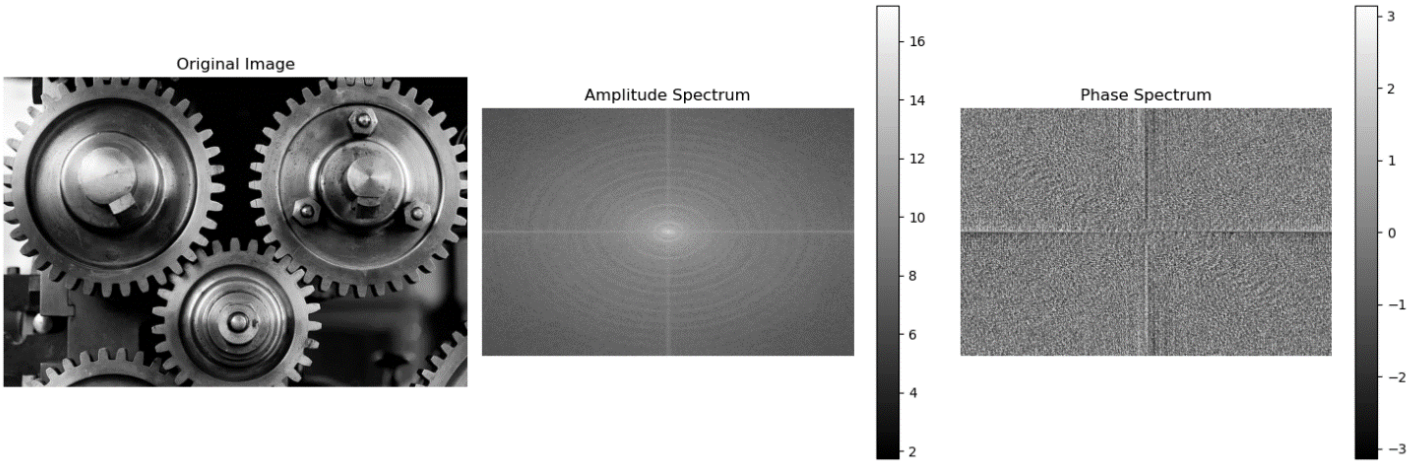
6 Our Implementation of DFT and Inverse DFT

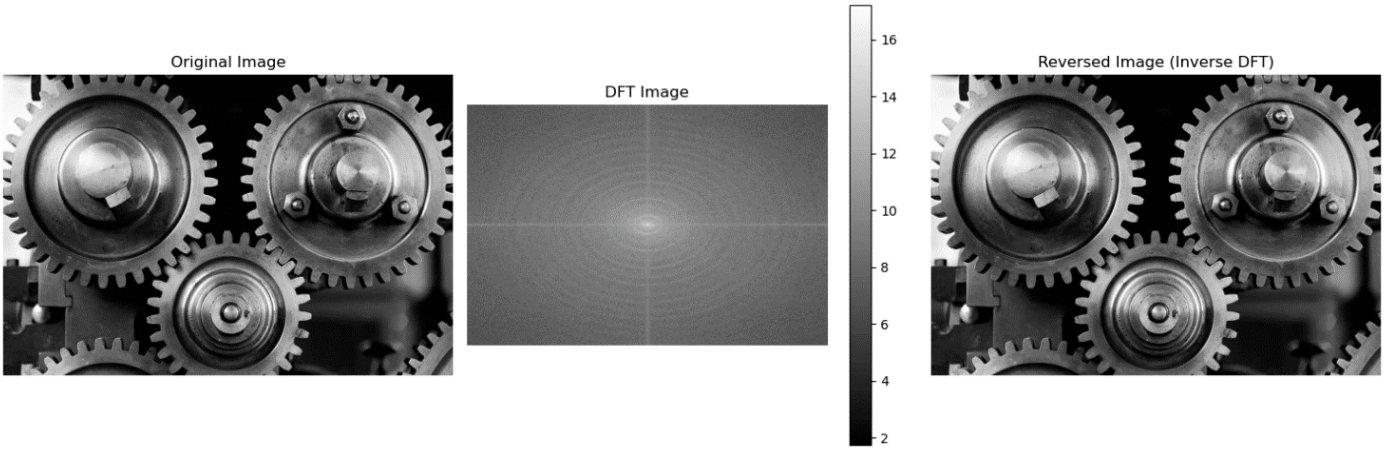
In computing a DFT, the Fast Fourier Transform (FFT) algorithm is a widely used method for efficiently computing the Discrete Fourier Transform (DFT). It offers significant computational advantages over straightforward DFT calculations, especially for large input sizes.

The FFT algorithm employs a divide-and-conquer approach to compute the DFT quickly. It begins by dividing the original vector or sequence into smaller segments, often by splitting it into two halves. The FFT is then recursively applied to each of these halves, calculating their respective FFTs. This recursive process continues until the base case is reached, typically when the segments are small enough to compute directly using simple DFT calculations.

We will utilize this one in our code. It is supplied within the OpenCV library. See code for details.

Our results are shown below with self-explanatory figures.





References

Fourier transform — Basics of Image Processing. (2023). Retrieved March 22, 2024, from Github.io website: https://vincmazet.github.io/bip/filtering/fourier.html

Oppenheim, Alan V, et al. Signals and Systems. Taipei, Pearson Education Taiwan Ltd, 2016.

Nguyen, A. (2023, February 21). Image Processing with Fourier Transform - Towards AI. Retrieved March 22, 2024, from Medium website: https://pub.towardsai.net/image-processing-with-fourier-transform-4ebc66651f2d

Image Transforms - Fourier Transform. (2024). Retrieved March 22, 2024, from Ed.ac.uk website: https://homepages.inf.ed.ac.uk/rbf/HIPR2/fourier.htm

LIN, Z. (2023, November 17). Digital Image Processing in C (Chapter 5): Fourier Transform and Reconstruction. Retrieved March 22, 2024, from Medium website: https://medium.com/@wilson.linzhe/digital-image-processing-in-c-chapter-5-fourier-transform-and-reconstruction-5a53f01846cd

Introduction to the Fourier Transform. (2024). Retrieved March 22, 2024, from Unm.edu website: https://www.cs.unm.edu/~brayer/vision/fourier.html

Ziemer, R. (2002). Modulation. Elsevier EBooks, 97–112. https://doi.org/10.1016/b0-12-227410-5/00456-7

Agu, E. (n.d.). Digital Image Processing (CS/ECE 545) Lecture 10: Discrete Fourier Transform (DFT). Retrieved from https://web.cs.wpi.edu/~emmanuel/courses/cs545/S14/slides/lecture10.pdf

Barry Van Veen. (2020). The Two-Dimensional Discrete Fourier Transform [YouTube Video]. Retrieved from https://www.youtube.com/watch?v=Iz6C1ny-F2Q

First Principles of Computer Vision. (2021). Image Filtering in Frequency Domain | Image Processing II [YouTube Video]. Retrieved from https://www.youtube.com/watch?v=OOu5KP3Gvx0

Friendly, C. (2021). Image Transforms and DFT (Discrete Fourier Transform) With Examples [YouTube Video]. Retrieved from https://www.youtube.com/watch?v=2oEt5lbsyhM

CSC589 Introduc/on to Computer Vision Lecture 8 Applica/ons of Fourier Transform, Sampling Bei Xiao. (n.d.). Retrieved from http://fs2.american.edu/bxiao/www/CSC589/Lecture8.pdf