Fast Fourier Transform of 2-Dimensional Signals

Rene L. Principe Jr.,
National Institute of Physics, University of the Philippines Diliman, Philippines

1 Introduction

Any signal can be decomposed into its most basic harmonic waves, the sines and cosines. Images can also be considered as a two-dimensional (2D) signal which is just a superposition of sines and cosines with varying frequencies [1]. For a gray image, each pixel has a value corresponding to its brightness, an 8-bit integer ranging from 0 (black) to 255 (white) [2]. Patterns may be formed by the periodic repetition of these bright and dark pixels, as if a trough (black) and crest (white) parallelism to a one-dimensional sine wave.

1.1 Mathematics

For an image, the sum of all the sines along x and y is si compressed into a function $f(x, y)$. To decompose $f(x, y)$, a Fourier Transform (FT) is applied given by Equation 1, which allows access to the frequency space. The original image can be reconstructed by performing the Inverse Fourier Transform as per Eq. 2.

$$F(f_x, f_y) = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} f(x, y) e^{-i2\pi(f_x x + f_y y)} dxdy, \tag{1}$$

$$f(x, y) = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} F(f_x, f_y) e^{i2\pi(f_x x + f_y y)} dxdy, \tag{2}$$

where f_x and f_y are the corresponding spatial frequencies [3].

Intuitively, the novelty of dealing with signals in the Fourier domain is that the dealing the signal in the Fourier domain allows us to capture more information, namely, the magnitude (pixel value), the spatial frequencies f_x and f_y, as well as the phase [1]. The latter two were a consequence of the image decomposition into its sinusoidal components.

2 Algorithm

This paper aims to provide a hand-in-hand intuitive and mathematical approach into understanding the Fourier Domain by applying Fourier Transformation to images containing various basic shapes, and eventually, exploit valuable information encoded in the frequency space.

Using a Python 3.0 environment in the Jupyter Notebook, functions were manually defined to create basic shapes and figures. Parameters were initially set to be an L×L zero array where L = 256 for a reason that will be revealed later on. Variables contained on the defined functions were strategically chosen for a controlled analysis. Sample codes for the initial steps into generating various shapes and figures are shown in the succeeding lines.

```
def Circ(r):           #generates a circle with a variable radius 'r'
def Ref(w,h):          #generates a rectangle (square) with variables width 'w'
                        and height 'h'
def Slit(d,h,w):       #generates a double slit (double dots)
                        with variables 'w','h', and slit separation 'd'
def Gaussian(sigma):   #generates a 2D Gaussian bell with a variable sigma 'sigma'
```

where all of the defined shape generating functions **return** *aperture* (shape projected onto the L×L array). Transforming the image into the frequency domain is pretty straight forward. The *numpy.fft* package provides a convenient and a non-tedious algorithm to implement the Fourier Transformation, as well as the inverse process.

```
FFT_Aperture = fftshift(np.abs(fft2(aperture)))
InvFFT_Aperture = ifft2(FFT_Aperture)
```

Unlike the usual Discrete Fourier Transform (DFT), Fast Fourier Transform (FFT) is faster in terms of implementation since it reduces N-point discrete calculations to (N/2)$\log_2$(N) [4]. The chosen parameter for axis length L = 256, such that the total number of Fourier transformation calculations needed is $256 \times 56 = 65526 = 2^{16}$. Using FFT, the Fourier Transform can be carried out with just (256/2)$\log_2$(256) = 1024 calculations, faster by a factor-of-64 times. The form of FFT is best subscribed at L = 2^n, which explains the parameter choice (L = 256). FFT is an intuitive choice because of its convenient mathematical consequences and all mathematical derivations explaining its novelty is beyond the scope of this study.

3 Results

The computer program generated the plots almost instantaneously, and the Fourier Transformation of various images is shown in Figure 1.

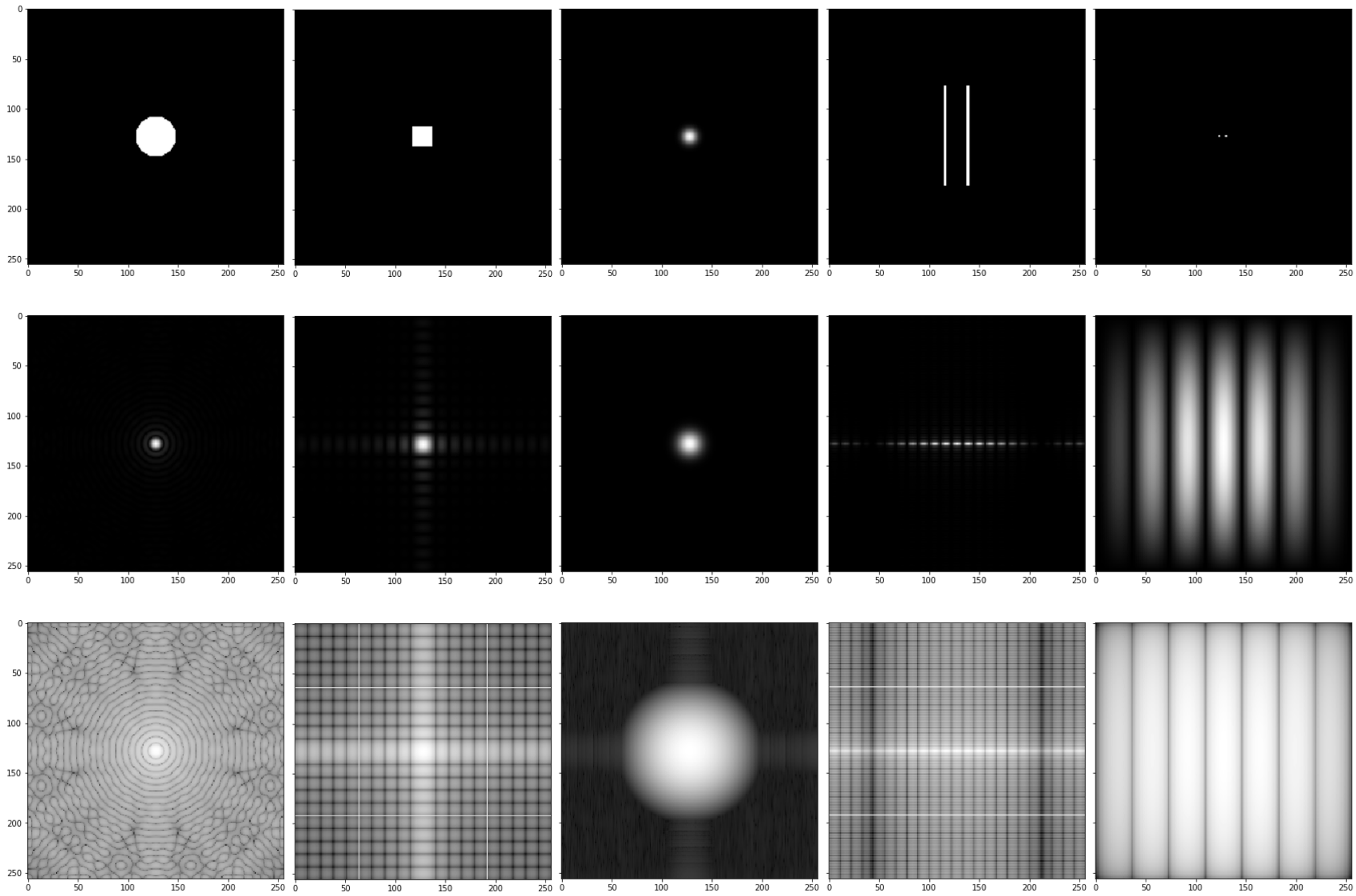

Figure 1: From left to right: [circle (r = 20), square (w = 20, h = 20), Gaussian bell (sigma = 5), double slit (d = 20), and double dot (d = 5)] (*variables in unit pixels*). The images on the first row are the generated aperture shapes. On the second row are their corresponding Fourier Transforms carried out using the *numpy.fft* package. The images on the third row are the Fourier Transforms in the logarithmic scale, intended to accentuate the features in the frequency domain.

The circular aperture has a Fourier Transform resembles a ripple like shape which is actually a 2D version of a Sinc function (where $sinc(x) = sin(x)/x$) [3]. Taking into account its rotational symmetry, $f(x, y) = 1$, at any set radius r, and 0 everywhere else. Inserting $f(x, y)_{circle}$ unto Eq. 1 yields a Bessel Function, and this actually manifested evidently on the Fourier Transform of the circle in the logarithmic scale [3] where the non-radial patterns are 2D higher-order kinds of Bessel function. Applying the same analysis, $f(x, y)_{square}$ on the Fourier domain turns out to be $F(f_x, f_y)$ = (x)(y)sinc(πxf_x)sinc(πyf_y) [3]. Both the circular and square apertures have Sinc function FTs. Due to the difference in the symmetry of the original 2-D signals, the former is radially separable while the latter is separable along f_x and f_y.

Interestingly, a Gaussian function has in fact, a Fourier Transform that's also a Gaussian (in the inverse scale relation) [3]. This makes sense because frequency domain is literally the reciprocal space. Nonetheless, even with varying σ (as shown in Fig. 2), the invariant plot is due to the exponential form of the Gaussian which preserves it form under various transformations.

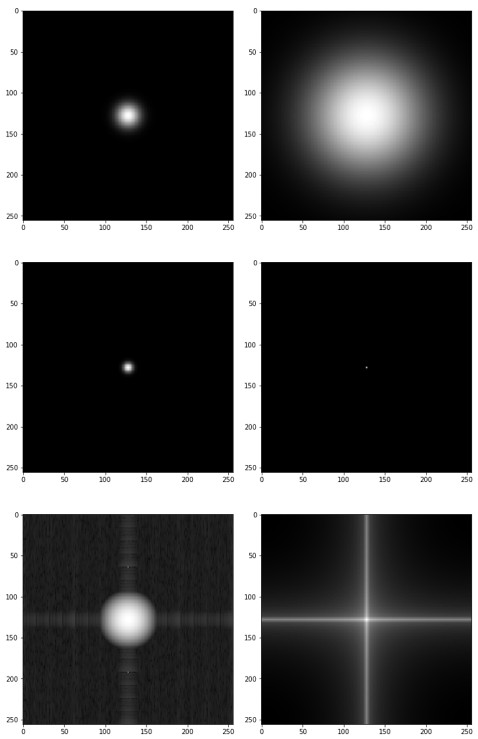

Figure 2: Fourier Transform of a 2D Gaussian bell is also a 2D Gaussian bell, with an inverse-scale relation [3]. This was carried out for various $\sigma = 10$ (left) and $\sigma = 50$ (right). On higher values of σ, the spatial frequency f_x and f_y is distinct.

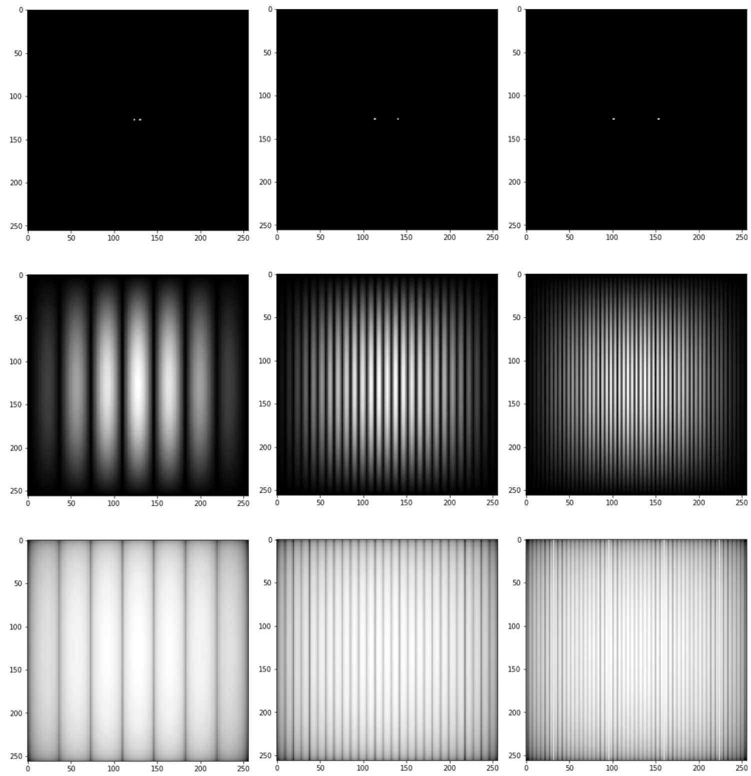

Figure 3: Two dots with increasing separation distance d (Top) and their corresponding Fourier Transforms (Middle) and FT in logarithmic scale (Bottom). From left to right: d = 5, 25, 50. Increasing d along the x-axis results to an increase in its corresponding spatial frequency along x f_x.

Another way to interpret the Fourier space is by considering the shapes generated as light apertures. Incoming light rays from these apertures obey the laws of geometric optics whenever they pass through a lens (analogously the Fourier Transformer). Eventually, the final image projected onto the viewing screen is the Fourier Transform of the light aperture [1]. Applying this intuitive approach to the shapes which were previously analyzed mathematically, circle and square images can be considered as collimated circular and rectangular light sources and thus, the final images (Fourier Transforms) emerged to have a similar shape and the ringings around are due to the diffraction around the edges. FTs in logarithmic scale (in Fig. 1) shows the constructive and destructive interference points which are the bright and dark regions on the plot. Suppose a double slit light aperture, the Fourier Transform distinctly showcases the interference phenomenon that we observe on our double-slit experiments.

As for the last aperture, a double dot was carried out using the code for double slit by minimizing the dimensions until slits were point-like. Distances were also varied between those two points, and the image of the aperture, its Fourier Transform and the FT in logarithmic scale can be seen in Fig. 3. The function $f(x, y)_{dots}$ describing the two dots separated about the x-axis only, when Fourier Transformed, yields a cosine wave with a slit separation d dependence $[F(f_x, f_y) = cos(2\pi d f_x)]$ [3]. These dots actively locates the corresponding frequency of the cosine wave [5]. It is important to note that in the Fourier space, the origin is where the zero frequency is, and it is symmetric on both x and y axes. Thus, placing the dots further from the origin implies a higher frequency of the Fourier transform. Results obtained from the simulations were shown in Fig. 3 and indeed, an increase in distance between the dots consequently resulted in an increase in the cosine wave frequency. The alternating bright and dark patterns in the frequency domain is what's meant in the Section 1 as the top-view of a sinusoidal wave.

To sum up, Fourier Transformation is implemented unto signals in order to access information exclusive in the frequency domain. This study has provided approaches in understanding the relationship of signal information in the real and reciprocal space through mathematical solutions. An intuitive approach which describes Fourier transforms as an optical system was also provided to facilitate the analysis. The simulation was carried out using a Fast Fourier Transformation (FFT) Method, which is significantly faster by a factor-of-64 times than the conventional Discrete Fourier Transform (DFT).

References

[1] S. Lehar, An Intuitive Explanaton of Fourier Theory, http://cns-alumni.bu.edu/~slehar/fourier/fourier.html.

[2] R. Fisher, S. Perkins, A. Walker, and E. Wolfart, Pixel Values (2000), https://homepages.inf.ed.ac.uk/rbf/HIPR2/value.html.

[3] A. Zisserman, 2D Fourier Transform (2014), http://www.robots.ox.ac.uk/~az/lectures/ia/lect2.pdf.

[4] DATAQ Instruments, Inc., FFT (Fast Fourier Transform) Waveform Analysis, https://www.dataq.com/data-acquisition/general-education-tutorials/fft-fast-fourier-transform-waveform-analysis.html.

[5] M. Bartolome, Properties and Applications of the 2D Fourier Transform (2015), https://barteezy.wordpress.com/2015/09/30/activity-6-properties-and-applications-of-the-2d-fourier-transform/.

Appendix

Source Code:

https://colab.research.google.com/drive/12osDY9T7DyipD1Fs0O7jmxixrdLQQnVX