Programming Assignment 3

CS 474

https://github.com/alexander-novo/CS474-PA3

Alexander Novotny
50% Work

Matthew Page
50% Work

Due: November 16, 2020
Submitted: November 16, 2020

Contents

1 Experiment 1

1.1 Theory

The Discrete Fourier Transform (DFT) is defined as a transformation that converts a finite sequence of complex values from the time or spatial domain into the frequency domain. Mathematically, the DFT is defined as

$$\mathcal{F}\{f(x)\}(u) = \sum_{x=0}^{N-1} f(x) \exp\left(\frac{-i2\pi ux}{N}\right)$$

The DFT is a generalization of the continuous Fourier Transform, which transforms a continuous function from the time domain to the frequency domain. The DFT can be interpreted as transforming a sequence of real values into a linear combination of complex exponential waves, or equivalently cosine and sine waves. In order to return the values into the time domain, the inverse DFT can be used, which is defined as

$$\mathcal{F}^{-1}\{F(u)\}(x) = \frac{1}{N} \sum_{u=0}^{N-1} F(u) \exp\left(\frac{i2\pi ux}{N}\right)$$

The main benefit of the DFT stems from the ability to view the frequency information of a sequence of discrete values. This is particularly useful in fields such as spectral analysis and signal processing, as it allows for the removal of undesired or noisy frequencies from the data, which produces a smoother signal. The DFT can be generalized into two dimensions for use with any 2D function, including 2D images when viewed as a function $f(x, y)$.

In order to perform the DFT on a continuous function, the function must first be sampled into a finite sequence of discrete, equally spaced values. This is equivalent to multiplying the continuous function by a series of equally spaced impulse functions, which can be defined as

$$f(x)s(x) = \sum_{k=-\infty}^{\infty} f(x)\delta(x - k\Delta x)$$

where Δx is the interval between each sample. When sampling from a distribution such as the Gaussian distribution it is important to consider the limits of the sampling values in order to ensure the full curve is represented during the DFT operation.

The DFT has the property of being periodic with period N, where N is the number of samples. In order to view the full period, it is necessary to shift the DFT by $N/2$. This can be achieved using the property

$$f(x)(-1)^x \Leftrightarrow F(u - N/2)$$

This step is necessary for the one-dimensional DFT as well as the two-dimensional DFT when visualization of the frequency domain is required. In the case of images, other transformations such as logarithmic scaling of the pixel values may be necessary as well.

1.2 Implementation

In order to implement the DFT, the Fast Fourier Transform (FFT) is used due to its more efficient time complexity of $\mathcal{O}(N \log N)$ compared to the naive implementation which runs at $\mathcal{O}(N^2)$. The main data structure used to store the sample points is an array of complex values. The data points that are to be plotted, including the magnitude and phase information, are stored in output files. These files are then read by Gnuplot in order to generate the necessary plots and graphs.

In order to sample the cosine function, the interval between each sample was calculated by subtracting the endpoints of one period and dividing that by the number of samples. The k-th cosine function sample was then evaluated as $y_k = \cos(2\pi uk\Delta x)$ for $0 \leq k < 128$, $u = 8$, and $\Delta x = \frac{1}{128}$.

1.3 Results and Discussion

Figure 1 showcases the results of performing the DFT on the function f sample data. According to the results, the real part of the first sample remains positive and the last three samples become negative. The DFT also introduces imaginary values to two of the samples. The magnitudes all remain positive as expected, and according to the phase plot the phase value flips sign for the second value of 4 in f.

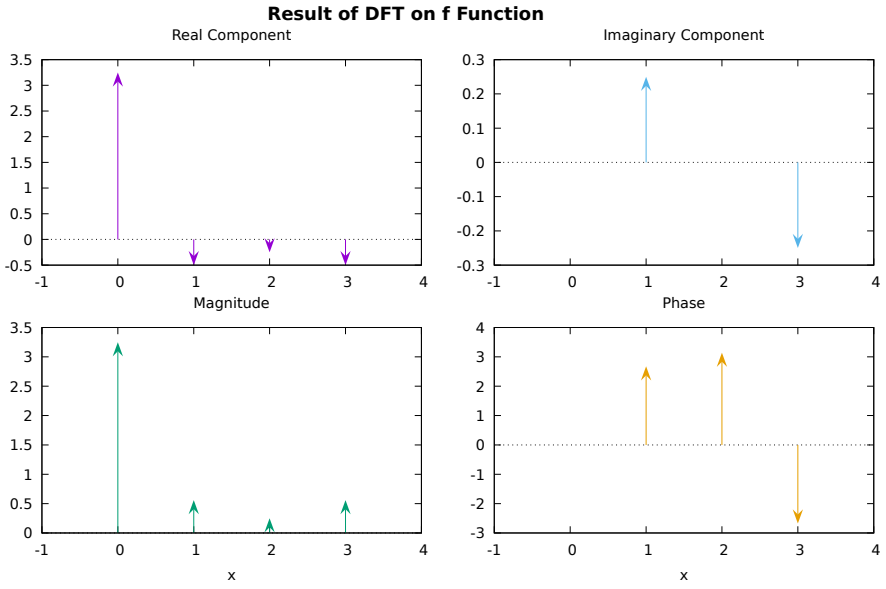


Figure 1: The result of performing the DFT on the sample data of the function f.

The cosine function along with its samples are shown in fig. 2. After performing the DFT on the cosine sample data, the results were plotted in fig. 3. According to the real component, the two non-zero values offset by 8 samples from the shifted center of the period represent the frequency of the cosine function, which is also 8. The imaginary values are all essentially zero as expected, and hence the magnitude corresponds to the absolute value of the real values. The phase also properly corresponds to the cyclical nature of the cosine function.

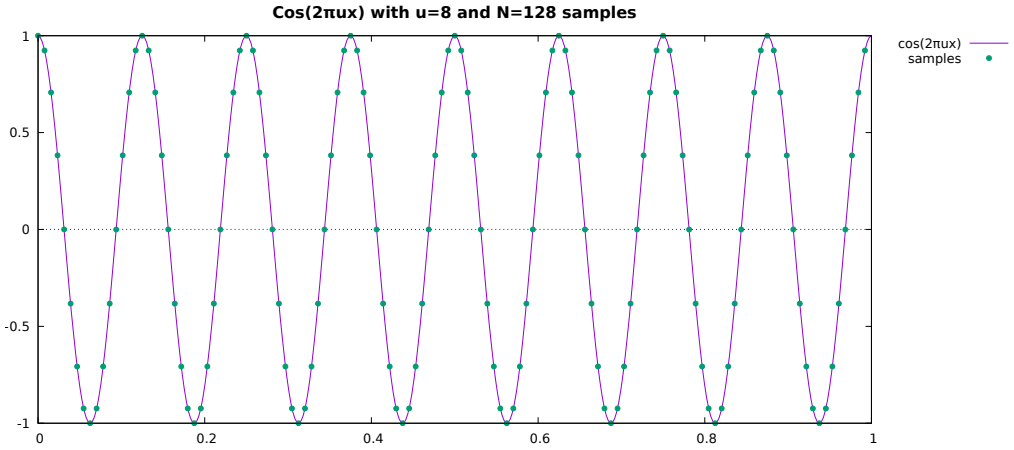


Figure 2: The given cosine was sampled along 128 points within its first $u = 8$ periods.

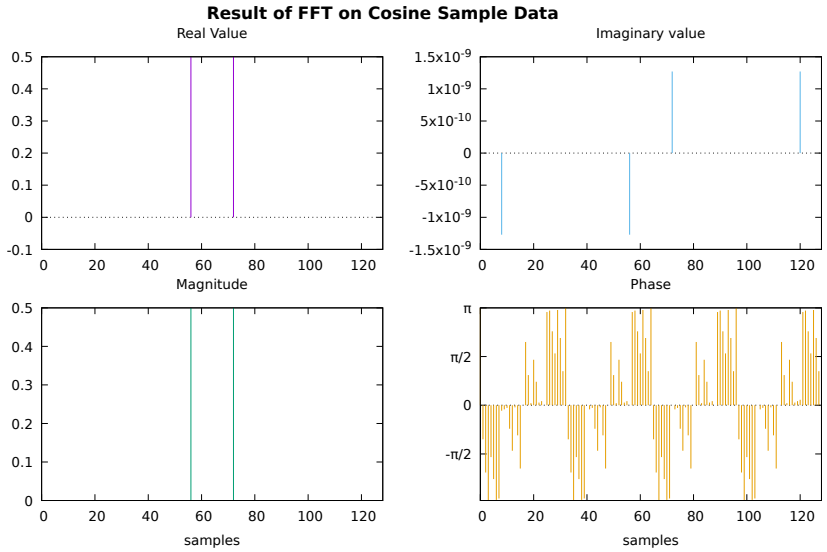


Figure 3: The result of performing the DFT on the cosine sample data.

Lastly, fig. 4 shows similar DFT data for the Rect function. Similarly to the cosine function, the imaginary values of the DFT are close to zero (but large enough to show a difference in the phase). However, the real values seem to follow the general shape of a discrete sinc function as expected. Since the imaginary parts of the frequencies are zero, the magnitude of the transformed Rect function corresponds to taking the absolute value of the real values.

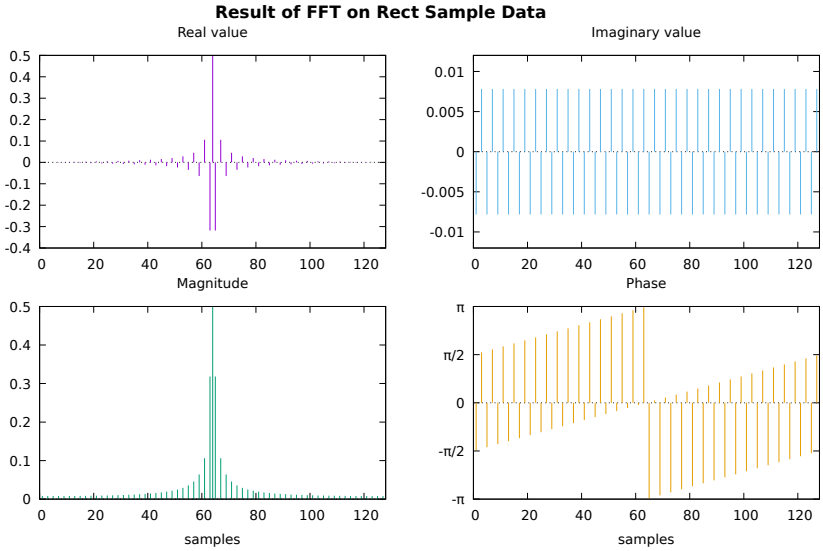


Figure 4: The result of performing the DFT on the Rect sample data.

2 Experiment 2

2.1 Theory

The 2-Dimensional Fourier Transform has a property called "separability", which shows that it can be computed as nested 1-Dimensional Fourier Transforms. A proof of this fact is given below:

$$\begin{aligned}
\mathcal{F}\{f(x,y)\}(u,v) &= \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)\exp\Big(-i2\pi\Big(\frac{ux}{M}+\frac{vy}{N}\Big)\Big) \\
&= \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)\exp\Big(-i2\pi\frac{ux}{M}\Big)\exp\Big(-i2\pi\frac{vy}{N}\Big) \\
&= \sum_{x=0}^{M-1}\left[\exp\Big(-i2\pi\frac{ux}{M}\Big)\sum_{y=0}^{N-1} f(x,y)\exp\Big(-i2\pi\frac{vy}{N}\Big)\right] \\
&= \sum_{x=0}^{M-1}\exp\Big(-i2\pi\frac{ux}{M}\Big)\mathcal{F}_y\{f(x,y)\}(x,v) \\
&= \mathcal{F}_x\{\mathcal{F}_y\{f(x,y)\}(x,v)\}(u,v),
\end{aligned} \tag{1}$$

where $\mathcal{F}_x$ is the 1-Dimensional Fourier Transform of the x variable.

Another property of the 2-D Fourier Transform is the translation property:

$$\mathcal{F}\Big\{f(x,y)\exp\Big(i2\pi\Big(\frac{u_0x}{M}+\frac{v_0y}{N}\Big)\Big)\Big\}(u,v) = \mathcal{F}\{f(x,y)\}(u-u_0,v-v_0), \tag{2}$$

from which we can see that the Fourier Transform translated half a period in each dimension is

$$\begin{aligned}\mathcal{F}\{f(x,y)\}(u-M/2,v-N/2) &= \mathcal{F}\left\{f(x,y)\exp\left(i2\pi\left(\frac{(M/2)x}{M}+\frac{(N/2)y}{N}\right)\right)\right\}(u,v)\\ &= \mathcal{F}\{f(x,y)\exp(i\pi(x+y))\}(u,v)\\ &= \mathcal{F}\left\{f(x,y)\exp(i\pi)^{(x+y)}\right\}(u,v)\\ &= \mathcal{F}\left\{f(x,y)(-1)^{(x+y)}\right\}(u,v).\end{aligned} \tag{3}$$

2.2 Implementation

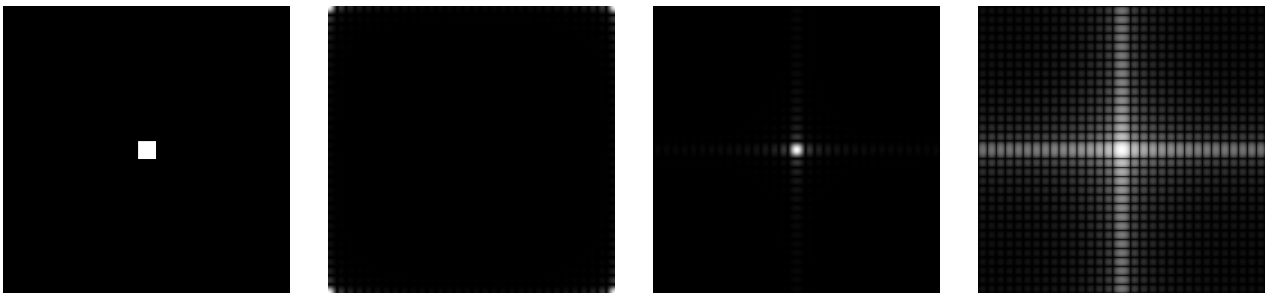
Equation (1) (the separability property) shows that we can compute the 2-Dimensional Fourier Transform of an image by simply reusing our 1-Dimensional Fast Fourier Transform algorithm and applying it first to the columns of an image, and then to the rows. A nice benefit to this is that the Fourier transform of each row (and each column) is independent from each other, so it is trivially parallelizable in shared memory (simply divide the rows evenly among threads and have them calculate each row separately, then do the same for columns - you must have a barrier between calculating columns and rows and it is recommended to use shared memory, as every row calculation depends on every column calculation).

Since the dc term $\mathcal{F}(f(x,y))(0,0)$ and surrounding values dominate the spectrum, they are the most visually interesting values. Unfortunately, in an image, $(0,0)$ is in the corner, so it is difficult to see, and surrounding values are spread across every corner due to the Fourier Transform being periodic in the dimensions of the image (see fig. 5b). It makes sense to shift these more important values to the center of the image, so we use eq. (3) to translate the Fourier Transform by one half period in each direction (half the image) by simply multiplying half the pixels by -1 in a checkerboard pattern before taking the Fourier Transform (see fig. 5c).

As well, the dc term and surrounding values dominates the spectrum by several orders of magnitude, typically. This means that when remapping to $[0,255]$ for an image, most of the spectrum will be mapped to 0 (or nearby), making it invisible. We can get rid of these orders of magnitude difference by simply applying a logarithmic intensity transformation before converting to an image (see fig. 5d).

2.3 Results and Discussion

Figure 5 shows the steps detailed above



(a) A 512×512 pixel image with a centered 32×32 pixel rectangle.

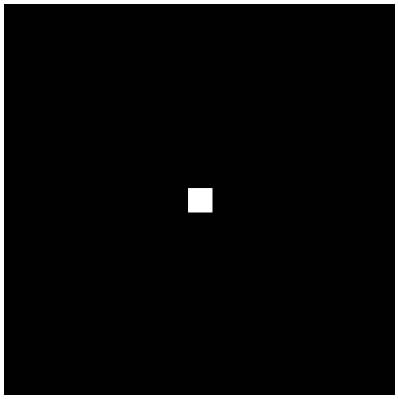
(b) The spectrum of the image.

(c) The spectrum shifted so that the dc term is centered on the image.

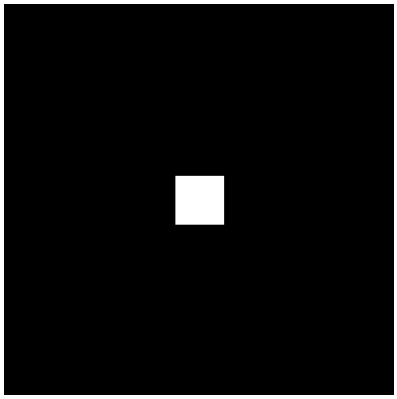
(d) A logarithmic intensity transformation is applied for better visualization.

Figure 5: A rectangle and its spectrum.

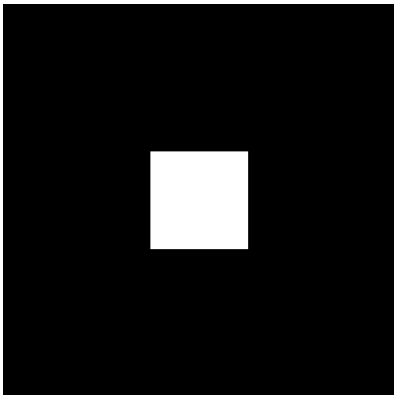
Figure 6 compares the spectrums of rectangles of different sizes after applying the above transformations. We can see that each one is the product of sinc functions (the expected outcome, as we went over in class) and as the rectangles get larger, the the sinc functions become higher frequency. This, combined with the symmetry property of the Fourier Transform, indicates that as the frequency of a sinc function gets higher, its Fourier Transform takes on values further from 0, which makes sense.



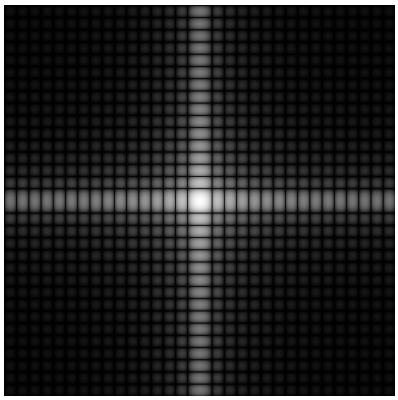
(a) A 512×512 pixel image with a centered 32×32 pixel rectangle.



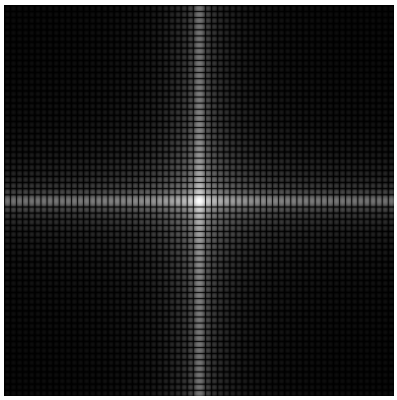
(b) A 512×512 pixel image with a centered 64×64 pixel rectangle.



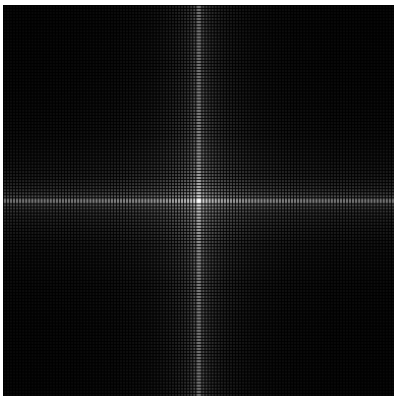
(c) A 512×512 pixel image with a centered 128×128 pixel rectangle.



(d) The spectrum of above.



(e) The spectrum of above.



(f) The spectrum of above.

Figure 6: A comparison of spectrums of rectangles of different sizes.

3 Experiment 3

3.1 Theory

Fundamentally, the DFT decomposes a signal into its underlying sinusoidal components that contribute to the original signal. In the case of a 2D image, the DFT decomposes the image into its two dimensional sinusoidal components. The DFT is a transformation on complex values, therefore each frequency component can be described by the magnitude and phase of that frequency. The magnitude of a frequency determines the strength of that frequency in the corresponding location in the spatial domain. The frequencies produced by the DFT also consists of a phase angle, which describes the relative shift of the different sinusoidal components that form the image.

The contributions of the magnitude and phase components of each frequency can be isolated to reform the original image using just one of the two components. This is done by either setting the magnitude of each frequency to one or by setting the phase of each frequency to zero and taking the Inverse DFT of the image. This allows the individual magnitude or phase components that make up the image to

be viewed individually.

3.2 Implementation

The implementation of this experiment consists of a single program that first reads in the input and output image paths, along with various options for generating the magnitude and phase images. The main algorithm consists of performing the 2D FFT on the input image. An array of complex¡float¿ type is used to store the results of the DFT. Next the magnitude or phase is removed using the above equations, and the inverse 2D FFT is performed to generate the resulting image. If specified, a log transformation is applied to the output image to improve visualization.

3.3 Results and Discussion

After performing the DFT on the lenna.pgm image, the magnitude and phase components of the image were analyzed. After setting the phase component to zero, the resulting image consists of a series of magnitudes that do not give any useful information on the general structure of the image. Figure 7 demonstrates this effect of removing phase information from the image. Conversely, figure 7 shows the original image with the magnitude information removed. Unlike the zero-phase image, there is much more information on the general structure of the image, including edge information. However, there is not much useful information on the strength of each pixel.

Figure 7: The result of removing the phase (left) and magnitude (right) component from the original `lenna.pgm` image. The center image is the result of applying a logarithmic intensity transformation to the reconstructed image without phase information to improve contrast.

Overall it seems that the phase information provides more useful information for reproducing the original image. Unlike the magnitude only image, features such as the woman's face and hat can be identified. However, small details remain lost along with the overall gray level strengths of each pixel. The magnitude only image provides no such information on the image features and is therefore not as useful for reconstructing the general structures in the image.