CS474/674 Image Processing and Interpretation

Fall 2024 – Dr. George Bebis

Programming Assignment 3

Due Date: 11/13/2024

This assignment will help you to better understand the Discrete Fourier Transform (DFT) and its properties. An efficient 1-D Fast Fourier Transform (FFT) implementation in C can be downloaded from the course's webpage. The code is from the book *"Numerical Recipes in C"*. The function prototype is shown below:

void fft(float data[], unsigned long nn, int isign)

nn is the length of the input (i.e., nn=N) which should be a power of 2. Warning: the program does not check this condition).

The elements in the array data[] (i.e., f(x) for the forward FFT or F(u) for the inverse FFT) must be stored from data[1] to data[2*nn]; data[0] is not used.

The real part of the input is stored in the odd locations of the array (data[1], data[3], data[5]. etc) and the imaginary part in the even locations (data[2], data[4], data[6], etc.). Please note that the imaginary part is zero when you compute the forward FFT of f(x). This not the case when you compute the inverse FFT (i.e., the imaginary part of F(u) is typically not zero). In general, both f(x) and F(u) can be thought as being complex functions.

isign: -1 Forward FFT, isign: 1 Inverse FFT (based on our definition)

Warning: the FFT routine provided does not multiply by the normalization factor 1/N that appears in the forward DFT equation; you should do this yourself.

$$F(u) = \frac{1}{N}\sum_{x=0}^{N-1} f(x) e^{\frac{-j2\pi ux}{N}}, \quad u = 0,1,2,...,N-1$$

$$f(x) = \sum_{u=0}^{N-1} F(u) e^{\frac{j2\pi ux}{N}}, \quad x = 0,1,2,...,N-1$$

Experiment 1

(1.a) Compute the DFT of the signal f = [2, 3, 4, 4] using the fft routine above and plot the real, imaginary, and magnitude parts of the DFT. Verify that the inverse DFT works correctly; if you

do not obtain the original values back, then you must be doing something wrong. Note: we did this example in the lecture; therefore, you should be able to verify your results.

(1.b) Generate and display a one-dimensional cosine wave with 128 samples that makes 8 cycles over a period:

f(x) = cos(2 π u x) where u = 8 and N = 128

Provide a graph of f(x) to verify that you have obtained the samples correctly. Compute the DFT f(x) and plot the real, imaginary, magnitude, and phase parts of the result. Make sure that you shift the magnitude to the center of the frequency domain using the property **$f(x)(-1)^x \leftrightarrow F(u - N/2)$** (i.e., see page 287). Report and justify your findings in your report.

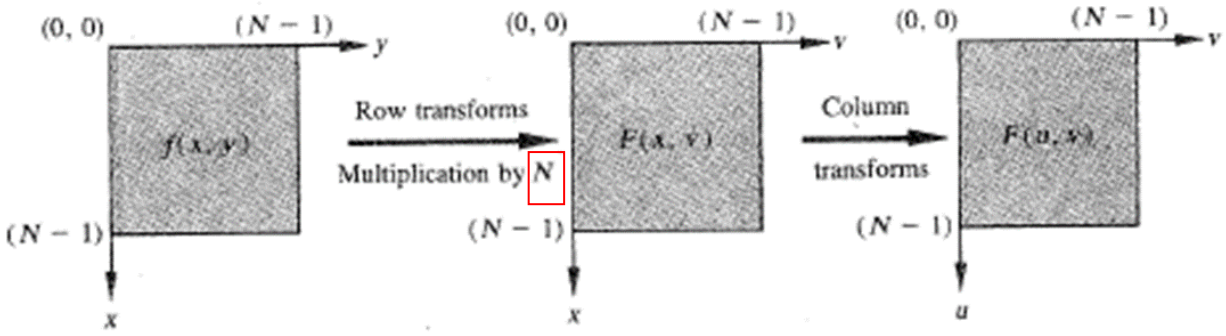
(1.c) Repeat the same experiment but this time for the Rectangular function (file: "Rect_128.dat").

For the rest of the experiments, you would need to implement the 2-D DFT as well as its inverse. Your implementation should take advantage of the **separability** property that we discussed in the lecture (i.e., compute the 2-D DFT by computing the 1-D DFT on the rows, followed by the 1-D DFT on the columns of the result). Use the function prototype below – you need to implement one function only, the value of **isign** and the inputs will determine whether you will be computing the forward or inverse transformation!

fft2D(N, M, real_Fuv, imag_Fuv, isign)

$$F(u,v) = \frac{1}{N}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f(x,y)e^{-j2\pi(\frac{ux+vy}{N})},$$

$$f(x,y) = \frac{1}{N}\sum_{u=0}^{N-1}\sum_{v=0}^{N-1} F(u,v)e^{j2\pi(\frac{ux+vy}{N})},$$



Before performing the experiments below, make sure that your 2-D DFT works correctly. For example, take the 2-D DFT of an image, followed by the inverse 2-D DFT; if you don't get the original image back, then you must be doing something wrong.

Experiment 2

(2.a) Generate a 512 x 512 image which contains a 32 x 32 square placed at the center of the image. Set the background to black (i.e., 0) and the interior of the square (and its boundary) to white (i.e., 255). The image should look like the one shown in Fig. 4.23 on page 296 in your textbook. Take the DFT of the image and show its magnitude without shifting it to the center of the frequency domain. Then, shift the magnitude to the center of the frequency domain using the property $\mathbf{f(x,y)(-1)^{x+y} \leftrightarrow F(u - N/2, v - N/2)}$ from (see Table 4.4 on page 305 in your textbook) and show the centered magnitude. Your results should look like those shown in Fig. 4.23(d) on page 296. Note: to properly show the magnitude of the DFT, you should use the **c x log(1+|F(u,v)|)** transformation as discussed in the lecture. Report and justify your findings in your report.

(2.b) Generate a 512 x 512 image which contains a 64 x 64 square placed at the center of the image. Repeat the steps given in (2.a). How do your results compare with (2.a)?

(2.c) Generate a 512 x 512 image which contains a 128 x 128 square placed at the center of the image. Repeat the steps given in (2.a). How do your results compare with (2.a) and (2.b)?

Experiment 3 (Extra Credit 20%)

In this experiment, you are going to examine the importance of magnitude and phase (check my slides as well as pages 298-299 in your textbook). First, compute the DFT of the *Lenna* image. Then, do the following:

(3.a) Set the phase equal to zero and take the inverse DFT (**hint:** set the real part to the **magnitude** of the image and the imaginary part to **zero**). The resulting image should look nothing like the original. Explain your results.

(3.b) Let the phase be the original one and set the magnitude equal to one and take the inverse DFT (**hint:** to set the magnitude equal to one, set the real part to **cos(theta)** and the imaginary part to **sin(theta)** where **theta=tan^{-1}(imag / real)** – make sure that you understand why this works). Since the magnitude is set to such a small value in the frequency domain, all the values in the spatial domain will be very small when you take the inverse DFT. To alleviate this problem, rescale the pixel values after the inverse DFT has been taken (i.e., values should be in [0, 255]). Explain your results.

Note: to compute **tan^{-1}**, use the function **atan2()**.