Intro to Digital Signal Processing

CONNEXIONS
CNX.ORG

Intro to Digital Signal Processing

Collection edited by: Robert Nowak

Content authors: Robert Nowak, Don Johnson, Michael Haag, Behnaam Aazhang, Nick Kingsbury, Benjamin Fite, Melissa Selik, Richard Baraniuk, Dan Calderon, Ivan Selesnick, Bill Wilson, Hyeokho Choi, Douglas Jones, Swaroop Appadwedula, Matthew Berry, Mark Haun, Dima Moussa, Daniel Sachs, Roy Ha, Justin Romberg, and Phil Schniter

Online: <http://cnx.org/content/col10203/1.4>

Intro to Digital Signal Processing

Table of Contents

Chapter 1. DSP Systems I

1.1. Image Restoration Basics*

Image Restoration

In many applications (*e.g.*, satellite imaging, medical imaging, astronomical imaging, poor-quality family portraits) the imaging system introduces a slight distortion. Often images are slightly blurred and image restoration aims at **deblurring** the image.

The blurring can usually be modeled as an LSI system with a given PSF $h[m, n]$.

$$h[m,n] \xleftrightarrow{\text{FT}} H(u,v)$$

Figure 1.1.

Fourier Transform (FT) relationship between the two functions.

The observed image is

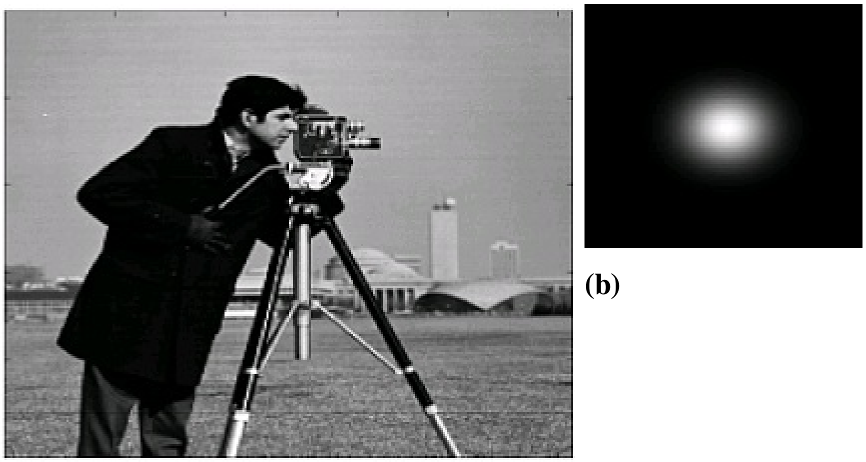
$$g[m, n]=h[m, n]*f[m, n] \quad ()$$

$$G(u, v)=H(u, v)F(u, v) \quad ()$$

$$F(u, v)=\frac{G(u, v)}{H(u, v)} \quad ()$$

Example 1.1. Image Blurring

Above we showed the equations for representing the common model for blurring an image. In Figure 1.2 we have an original image and a PSF function that we wish to apply to the image in order to model a basic blurred image.



(b)

(a)

Figure 1.2.

Once we apply the PSF to the original image, we receive our blurred image that is shown in Figure 1.3:

Figure 1.3.

Frequency Domain Analysis

Example 1.1 looks at the original images in its typical form; however, it is often useful to look at our images and PSF in the frequency domain. In Figure 1.4, we take another look at the image blurring example above and look at how the images and results would appear in the frequency domain if we applied the fourier transforms.

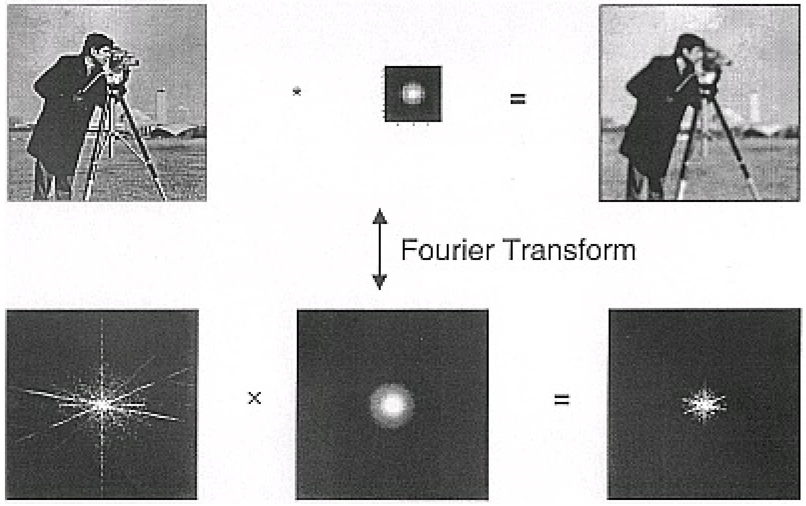


Figure 1.4.

1.2. Digital Image Processing Basics*

Digital Image Processing

A sampled image gives us our usual 2D array of pixels $f[m, n]$ (Figure 1.5):

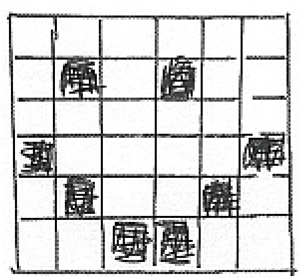


Figure 1.5.

We illustrate a "pixelized" smiley face.

We can filter $f[m, n]$ by applying a 2D discrete-space **convolution** as shown below (where $h[m, n]$ is our PSF):

$$\begin{aligned} g[m, n] &= h[m, n] * f[m, n] \\ &= \sum_{k=-\infty}^{\infty} \left(\sum_{l=-\infty}^{\infty} (h[m-k, n-l] f[k, l]) \right) \end{aligned} \quad ()$$

Example 1.2. Sampled Image

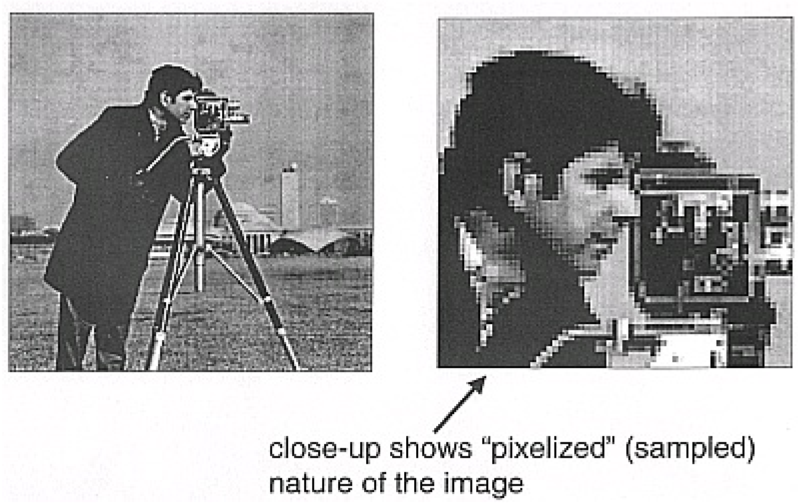


Figure 1.6.

Illustrate the "pixelized" nature of all digital images.

We also have discrete-space FTS:

$$F[u, v] = \sum_{m=-\infty}^{\infty} \left(\sum_{n=-\infty}^{\infty} \left(f[m, n] e^{-(ium)} e^{-(ivn)} \right) \right) \qquad ()$$

where $F[u, v]$ is analogous to **DTFT** in 1D.

"Convolution in Time" is the **same** as "Multiplication in Frequency"

$$g[m, n] = h[m, n] * f[m, n] \qquad ()$$

which, as stated above, is the same as:

$$G[u, v] = H[u, v] F[u, v] \qquad ()$$

Example 1.3. Magnitude of FT of Cameraman Image

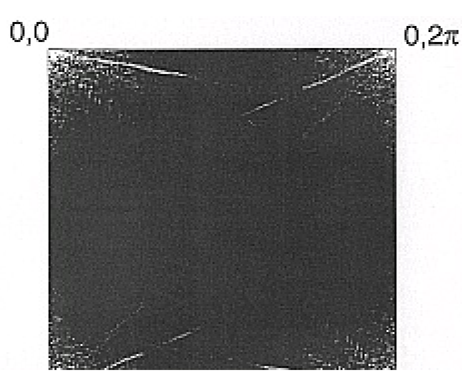


Figure 1.7.

To get a better image, we can use the `fftshift` command in Matlab to center the Fourier Transform. The resulting image is shown in Figure 1.8:

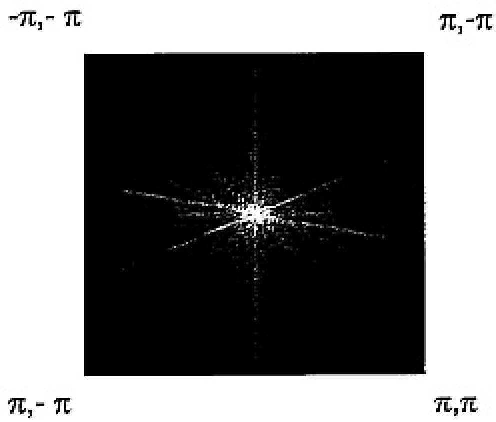


Figure 1.8.

1.3. 2D DFT*

2D DFT

To perform image restoration (and many other useful image processing algorithms) in a computer, we need a Fourier Transform (FT) that is discrete and two-dimensional.

$$F[k,l]=F(u,v)|_{u=\frac{2\pi k}{N},\, v=\frac{2\pi l}{N}} \quad ()$$

for $k=\{0, \ldots, N-1\}$ and $l=\{0, \ldots, N-1\}$.

$$F(u,v)=\sum_{m}\left(\sum_{n}\left(f[m,n]e^{-(ium)}e^{-(ivm)}\right)\right) \quad ()$$

$$F[k,l]=\sum_{m=0}^{N-1}\left(\sum_{n=0}^{N-1}\left(f[m,n]e^{(-i)\frac{2\pi km}{N}}e^{(-i)\frac{2\pi ln}{N}}\right)\right) \quad ()$$

where the above equation (Equation) has finite support for an NxN image.

Inverse 2D DFT

As with our regular fourier transforms, the 2D DFT also has an inverse transform that allows us to reconstruct an image as a weighted combination of complex sinusoidal basis functions.

$$f[m,n]=\frac{1}{N^2}\sum_{k=0}^{N-1}\left(\sum_{l=0}^{N-1}\left(F[k,l]e^{\frac{i2\pi km}{N}}e^{\frac{i2\pi ln}{N}}\right)\right) \qquad ()$$

Example 1.4. Periodic Extensions

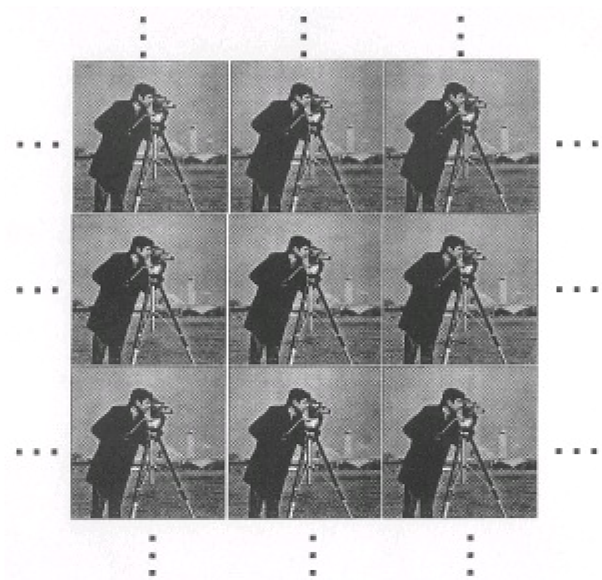


Figure 1.9.

Illustrate the periodic extension of images.

2D DFT and Convolution

The regular 2D convolution equation is

$$g[m,n]=\sum_{k=0}^{N-1}\left(\sum_{l=0}^{N-1}\left(f[k,l]h[m-k,n-l]\right)\right) \qquad ()$$

Example 1.5.

Below we go through the steps of convolving two two-dimensional arrays. You can think of f as representing an image and h represents a PSF, where $h[m,n]=0$ for m and $n>1$ and m and $n<0$. $h=\begin{pmatrix} h[0,0] & h[0,1] \\ h[1,0] & h[1,1] \end{pmatrix}$ $f=\begin{pmatrix} f[0,0] & \dots & f[0,N-1] \\ \vdots & \ddots & \vdots \\ f[N-1,0] & \dots & f[N-1,N-1] \end{pmatrix}$ Step 1 (Flip h):

$$h[-m,-n]=\begin{pmatrix} h[1,1] & h[1,0] & 0 \\ h[0,1] & h[0,0] & 0 \\ 0 & 0 & 0 \end{pmatrix} \quad ()$$

Step 2 (Convolve):

$$g[0,0]=h[0,0]f[0,0] \quad ()$$

We use the standard 2D convolution equation (Equation) to find the first element of our convolved image. In order to better understand what is happening, we can think of this visually. The basic idea is to take $h[-m,-n]$ and place it "on top" of $f[k,l]$, so that just the bottom-right element, $h[0,0]$ of $h[-m,-n]$ overlaps with the top-left element, $f[0,0]$, of $f[k,l]$. Then, to get the next element of our convolved image, we slide the flipped matrix, $h[-m,-n]$, over one element to the right and get the following result: $g[0,1]=h[0,0]f[0,1]+h[0,1]f[0,0]$ We continue in this fashion to find all of the elements of our convolved image, $g[m,n]$. Using the above method we define the general formula to find a particular element of $g[m,n]$ as:

$$g[m,n]=h[0,0]f[m,n]+h[0,1]f[m,n-1]+h[1,0]f[m-1,n]+h[1,1]f[m-1,n-1] \quad ()$$

Circular Convolution

Exercise 1.

What does $H[k,l]F[k,l]$ produce?

2D Circular Convolution

$$\begin{aligned} \tilde{g}[m,n] &= \mathrm{IDFT}(H[k,l]F[k,l]) \\ &= \text{circular convolution in 2D} \end{aligned} \quad ()$$

Due to periodic extension by DFT (Figure 1.10):

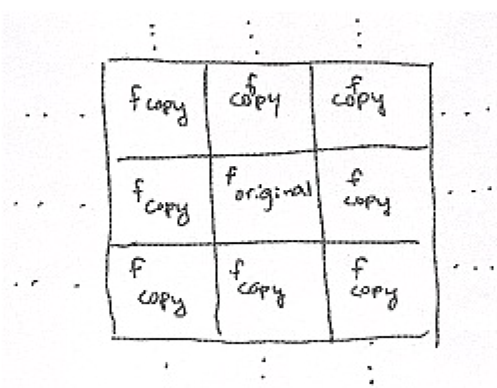


Figure 1.10.

Based on the above solution, we will let

$$\tilde{g}[m, n] = \mathrm{IDFT}(H[k, l]F[k, l]) \quad ()$$

Using this equation, we can calculate the value for each position on our final image, $\tilde{g}[m, n]$. For example, due to the periodic extension of the images, when circular convolution is applied we will observe a **wrap-around** effect.

$$\tilde{g}[0, 0] = h[0, 0]f[0, 0] + h[1, 0]f[N-1, 0] + h[0, 1]f[0, N-1] + h[1, 1]f[N-1, N-1] \quad ()$$

Where the last three terms in Equation are a result of the wrap-around effect caused by the presence of the images copies located all around it.

Zero Padding

If the support of h is MxM and f is NxN, then we zero pad f and h to $M+N-1$ x $M+N-1$ (see Figure 1.11).

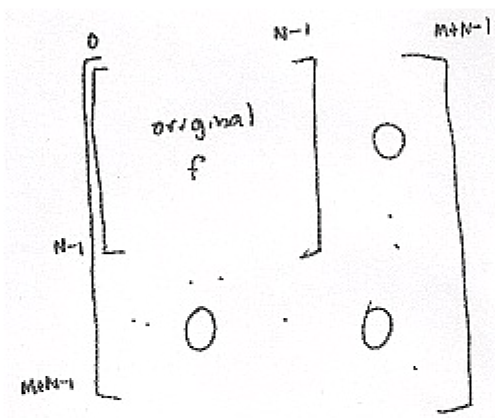


Figure 1.11.

Circular Convolution = Regular Convolution

Computing the 2D DFT

$$F[k, l] = \sum_{m=0}^{N-1} \left(\sum_{n=0}^{N-1} \left(f[m, n] e^{(-i)\frac{2\pi km}{N}} e^{(-i)\frac{2\pi ln}{N}} \right) \right) \quad ()$$

where in the above equation, $\sum_{n=0}^{N-1} \left(f[m, n] e^{(-i)\frac{2\pi ln}{N}} \right)$ is simply a 1D DFT over n. This means that we will take the 1D FFT of each row; if we have N rows, then it will require ***NlogN*** operations per row. We can rewrite this as

$$F[k, l] = \sum_{m=0}^{N-1} \left(f'[m, l] e^{(-i)\frac{2\pi k m}{N}} \right)$$ ()

where now we take the 1D FFT of each column, which means that if we have N columns, then it requires $\boldsymbol{N \log N}$ operations per column.

Therefore the overall complexity of a 2D FFT is $O(N^2 \boldsymbol{\log N})$ where N^2 equals the number of pixels in the image.

1.4. Images: 2D signals*

Image Processing

Image not finished
Figure 1.12.

Images are 2D functions $f(x, y)$

Linear Shift Invariant Systems

Image not finished
Figure 1.13.

H is LSI if:

1. $H(\alpha_1 f_1(x, y) + \alpha_2 f_2(x, y)) = H(f_1(x, y)) + H(f_2(x, y))$ for all images f_1 , f_2 and scalar.
2. $H(f(x - x_0, y - y_0)) = g(x - x_0, y - y_0)$

LSI systems are expressed mathematically as 2D convolutions: $g(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} h(x - \alpha, y - \beta) f(\alpha, \beta) \, d\alpha \, d\beta$ where $h(x, y)$ is the 2D impulse response (also called the **point spread function**).

2D Fourier Analysis