Azmi Tewfiq. Hussain Al-Rawi,
Faculty of C. S.& I.T. University of Anbar

The Fourier Transform

The Fourier Transform is of fundamental importance to image processing. It allows us to perform tasks which would be impossible to perform any other way; its efficiency allows us to perform other tasks more quickly. The Fourier is a powerful alternative to linear spatial filtering; it is more efficient to use the Fourier transform than a spatial filter for a large filter.

Definition of the one dimensional discrete Fourier transform (DFT):

Suppose

$$\mathbf{f} = [f_0, f_1, f_2, \ldots, f_{N-1}]$$

is a sequence of length N. We define its *discrete Fourier transform* to be the sequence

$$\mathbf{F} = [F_0, F_1, F_2, \ldots, F_{N-1}]$$

where

$$F_u = \frac{1}{N} \sum_{x=0}^{N-1} \exp\left[-2\pi i \frac{xu}{N}\right] f_x.$$

The inverse DFT:

The formula for the inverse DFT is very similar to the forward transform

$$x_u = \sum_{x=0}^{N-1} \exp\left[2\pi i \frac{xu}{N}\right] F_u.$$

Properties of the one-dimensional DFT:

Linearity: Suppose f and g are two vectors of equal length, p and q are scalars, with h=pf+qg . If F, G and H are the DFT's of f,g and h respectively, we

H=Pf+qG

This follows from the definition

$$\mathtt{F} = \mathcal{F}\mathtt{f}, \quad \mathtt{G} = \mathcal{F}\mathtt{g}, \quad \mathtt{H} = \mathcal{F}\mathtt{h}$$

Shifting: we change the sign of every second element x by $(-1)^n$. Let the resulting vector be denoted $x^/$. The DFT $X^/$ of $x^/$ is equal to the DFT X of x DFT with the swapping of the left and right halves.

The Fast Fourier Transform: There are a number of extremely fast and efficient algorithms for computing a DFT; such an algorithm is called a fast Fourier transform, or FFT. The use of an FFT vastly reduces the time needed to compute a DFT. One FFT method works recursively by dividing the original vector into two halves, computing the FFT of each half.

For a vector of length 2^n , the direct method takes $(2^n)^2=2^{2n}$ multiplications; the FFT only $n2^n$. The saving in time is thus of an order of $2^n/n$. Clearly the advantage of using an FFT algorithm becomes greater as the size of the vector increases.

Because of the this computational advantage, any implementation of the DFT will use an FFT algorithm:

Table 1: Comparison of FFT and direct arithmetic.

2^n	Direct arithmetic	FFT	Increase in speed
4	16	8	2.0
8	84	24	2.67
16	256	64	4.0
32	1024	160	6.4
64	4096	384	10.67
128	16384	896	18.3
256	65536	2048	32.0
512	262144	4608	56.9
1024	1048576	10240	102.4

The two-dimensional DFT In two dimensions:

the DFT takes a matrix as input, and returns another matrix, of the same size, as output. If f(x,y) the original matrix values are where x and y are the indices, then the output matrix values are F(u,v)

The forward and inverse transforms for an MxN matrix, where for notational convenience we assume that the x indices are from 0 to M-1 and the y indices are from to 0 to N-1, are

$$F(u,v) = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)\exp\left[-2\pi i\left(\frac{xu}{M}+\frac{yv}{N}\right)\right].$$

$$f(x,y) = \frac{1}{MN}\sum_{u=0}^{M-1}\sum_{v=0}^{N-1} F(u,v)\exp\left[2\pi i\left(\frac{xu}{M}+\frac{yv}{N}\right)\right].$$

Some properties of the two dimensional Fourier transform :All the properties of the one-dimensional DFT transfer into two dimensions. But there are some further properties as :

Similarity: First notice that the forward and inverse transforms are very similar, with the exception of the scale factor 1/MN in the inverse transform, and the negative sign in the exponent of the forward transform.

The DFT as a spatial filter: Note that the values

$$\exp\left[\pm 2\pi i\left(\frac{xu}{M}+\frac{yv}{N}\right)\right]$$

are independent of the values f or F. as above. It also means that every value is obtained by multiplying every value of f(x,y) by a fixed value, and adding up all the results.

what a linear spatial filter does? it multiplies all elements under a mask with fixed value and adds them all up. Thus we can consider the DFT as a linear spatial filter which is as big as the image.

Separability: Notice that the Fourier transform "filter elements" can be expressed as products:

$$\exp\left[2\pi i\left(\frac{xu}{M}+\frac{yv}{N}\right)\right] = \exp\left[2\pi i\frac{xu}{M}\right]\exp\left[2\pi i\frac{yv}{N}\right].$$

The first product value

$$\exp\left[2\pi i\frac{xu}{M}\right]$$

depends only on x and u, and is independent of y and v. Conversely, the second product valu

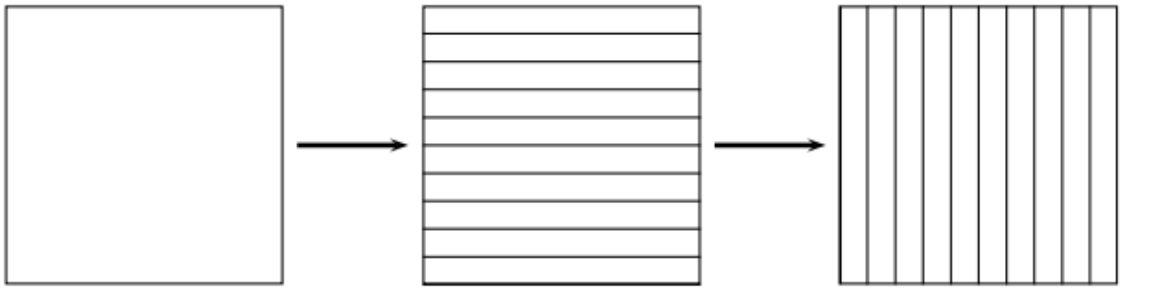
$$\exp\left[2\pi i\frac{yv}{N}\right]$$

depends only on y and v, and is independent of x and u. This means that we can break dow formulas above to simpler formulas that work on single rows or columns:

$$F(u) = \sum_{x=0}^{M-1} f(x)\exp\left[-2\pi i\frac{xu}{M}\right],$$

$$f(x) = \frac{1}{M}\sum_{u=0}^{M-1} F(u)\exp\left[2\pi i\frac{xu}{M}\right].$$

The 2-D DFT can be calculated by using this property of "separability"; to obtain the 2-D DFT of a matrix, we first calculate the DFT of all the rows, and then calculate the DFT of all the columns of the result, as shown in figure 1



(a) Original image (b) DFT of each row of (a) (c) DFT of each column of (b)

Figure 1 : Calculating DFT

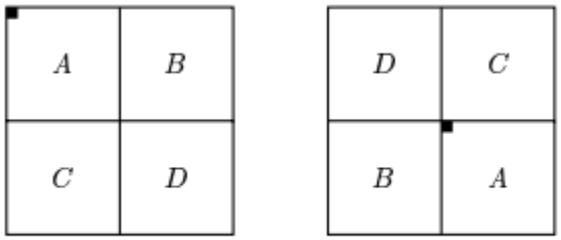
Linearity: An important property of the DFT is its linearity;

$$\mathcal{F}(f+g) = \mathcal{F}(f)+\mathcal{F}(g)$$
$$\mathcal{F}(kf) = k\mathcal{F}(f)$$

The DC coefficient. The value F(0,0) of the DFT is called the DC coefficient;

$$F(0,0) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) \exp(0) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y).$$

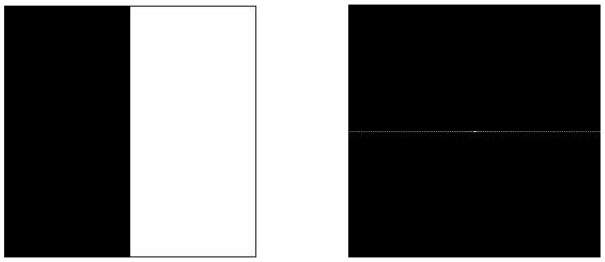
Shifting: For purposes of display, it is convenient to have the DC coefficient in the center of the matrix. This will happen if all elements in the matrix are multiplying by $(-1)^{x+y}$ before transform.



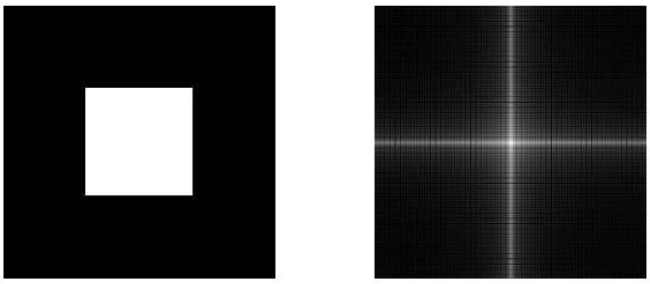
An FFT After shifting

Fourier transforms of images:

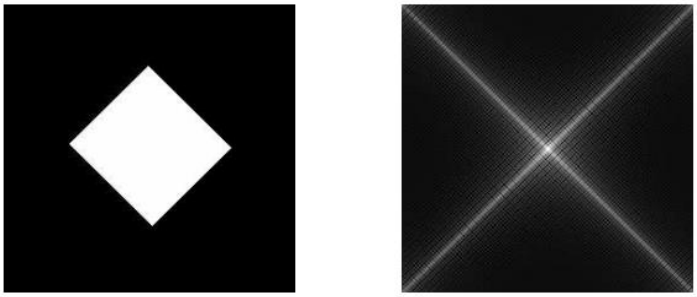
Example 1: produce a simple image consisting of a single edge and its DFT.



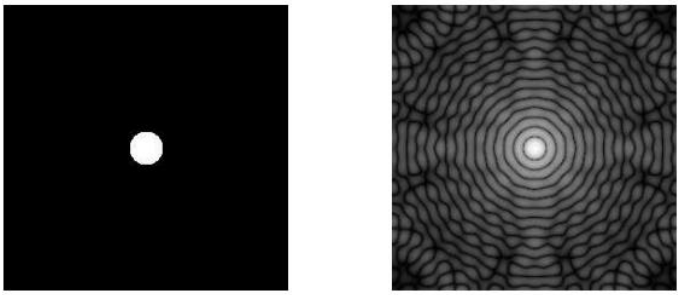
Example 2: create a box, and then its Fourier transform



Example 3: A box rotate $45^0$and its DFT



Example 4: Create a circle and its DFT



-An image can be expanded in terms of a discrete set of basis arrays called basis images. Unitary matrices can generate these basis images. An image transform provides a set of coordinates or basis vectors for vector space.

The Applications Of Transform:

To reduce band width

To reduce redundancy

To extract feature.

-The Conditions For Perfect Transform

Transpose of matrix = Inverse of a matrix. Orthoganality.

Discrete Cosine Transform:

its computational efficiency the DFT is very popular represent the signal in the frequency domain. This is very important for image compression. A much better transform

The DCT Equation

The DCT equation (Eq. 1) computes the i,jth entry of the DCT of an image.

$$D(i,j) = \frac{1}{\sqrt{2N}} C(i)C(j) \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} p(x,y) \cos\left[\frac{(2x+1)i\pi}{2N} \right] \cos\left[\frac{(2y+1)j\pi}{2N} \right] \quad 1$$

$$C(u) = \left\{ \begin{array}{ll} \frac{1}{\sqrt{2}} & \text{if } u = 0 \\ 1 & \text{if } u > 0 \end{array} \right\} \quad 2$$

p(x,y) is the x,yth element of the image represented by the matrix p and N is the block size that is DCT done.

The Properties Of Cosine Transform

-Real & orthogonal.

-Fast transform.

-Has excellent energy compaction for highly correlated data

The Properties Of Sine Transform?

Real, symmetric and orthogonal.

Not the imaginary part of the unitary DFT.

Fast transform. For 8x8 block , there we D(i,j) would be in eq.3

$$D(i,j) = \frac{1}{4} C(i)C(j) \sum_{x=0}^{7} \sum_{y=0}^{7} p(x,y) \cos\left[\frac{(2x+1)i\pi}{16} \right] \cos\left[\frac{(2y+1)j\pi}{16} \right] \quad 3$$

Image compression

Image compression is the process of encoding or converting an image file in such a way that it consumes less space than the original file.

compression technique that reduces the size of an image file without affecting or degrading its quality to a greater extent.

Image compression is typically performed through an image/data compression algorithm or codec. Typically such codecs/algorithms apply different techniques to reduce the image size.

Some of the common image compression techniques are:

Fractal

Wavelets

Chroma sub sampling

Transform coding

Run-length encoding.

Types of image compression:

1-**Lossless compression** is preferred for archival purposes and often for medical imaging, technical drawings, clip art, or comics.

2-**Lossy compression** methods, especially when used at low bit rates, introduce compression artifacts. Lossy methods are especially suitable for natural images such as photographs in applications.

Methods for lossless image compression are:

Run-length encoding – used in default method in PCX and as one of possible in BMP, TGA, TIFF

Area image compression

DPCM and Predictive Coding

Entropy encoding

Adaptive dictionary algorithms such as LZW – used in GIF and TIFF

DEFLATE – used in PNG, MNG, and TIFF

Chain codes

Methods for lossy compression:

Reducing the color space to the most common colors in the image. The selected colors are specified in the colour palette in the header of the compressed image. Each pixel just references the index of a color in the color palette, this method can be combined with dithering to avoid posterization.

Transform coding is the most commonly used method. In particular, a Fourier-related transform such as the Discrete Cosine Transform (DCT) is, "Discrete Cosine Transform," The more recently developed wavelet transform is also used extensively, followed by quantization and entropy coding