|  |
| --- |
|  |
| Fourier TransformDefinition of Fourier TransformThe Fourier transform is a representation of an image as a sum of complex exponentials of varying magnitudes, frequencies, and phases. The Fourier transform plays a critical role in a broad range of image processing applications, including enhancement, analysis, restoration, and compression.If *f*(*m*,*n*) is a function of two discrete spatial variables *m* and *n*, then the two-dimensional Fourier transform of *f*(*m*,*n*) is defined by the relationshiphttp://www.mathworks.in/help/toolbox/images/eqn1248888637.gifThe variables *ω*1 and *ω*2 are frequency variables; their units are radians per sample. *F*(*ω*1,*ω*2) is often called the frequency-domain representation of *f*(*m*,*n*). *F*(*ω*1,*ω*2) is a complex-valued function that is periodic both in *ω*1 and *ω*2, with period http://www.mathworks.in/help/toolbox/images/eqn1248889504.gif. Because of the periodicity, usually only the range http://www.mathworks.in/help/toolbox/images/eqn1248889588.gifis displayed. Note that *F*(0,0) is the sum of all the values of *f*(*m*,*n*). For this reason, *F*(0,0) is often called the constant component or DC component of the Fourier transform. (DC stands for direct current; it is an electrical engineering term that refers to a constant-voltage power source, as opposed to a power source whose voltage varies sinusoidally.)The inverse of a transform is an operation that when performed on a transformed image produces the original image. The inverse two-dimensional Fourier transform is given byhttp://www.mathworks.in/help/toolbox/images/eqn1248889757.gifRoughly speaking, this equation means that *f*(*m*,*n*) can be represented as a sum of an infinite number of complex exponentials (sinusoids) with different frequencies. The magnitude and phase of the contribution at the frequencies (*ω*1,*ω*2) are given by *F*(*ω*1,*ω*2).Visualizing the Fourier TransformTo illustrate, consider a function *f*(*m*,*n*) that equals 1 within a rectangular region and 0 everywhere else. To simplify the diagram, *f*(*m*,*n*) is shown as a continuous function, even though the variables m and n are discrete.Rectangular Functionhttp://www.mathworks.in/help/toolbox/images/transform26.gifThe following figure shows, as a mesh plot, the magnitude of the Fourier transform, http://www.mathworks.in/help/toolbox/images/eqn1248895793.gifof the rectangular function shown in the preceding figure. The mesh plot of the magnitude is a common way to visualize the Fourier transform.Magnitude Image of a Rectangular Functionhttp://www.mathworks.in/help/toolbox/images/transform27.gifThe peak at the center of the plot is *F*(0,0), which is the sum of all the values in *f*(*m*,*n*). The plot also shows that *F*(*ω*1,*ω*2) has more energy at high horizontal frequencies than at high vertical frequencies. This reflects the fact that horizontal cross sections of *f*(*m*,*n*) are narrow pulses, while vertical cross sections are broad pulses. Narrow pulses have more high-frequency content than broad pulses.Another common way to visualize the Fourier transform is to display http://www.mathworks.in/help/toolbox/images/eqn1248895879.gifas an image, as shown.Log of the Fourier Transform of a Rectangular Functionhttp://www.mathworks.in/help/toolbox/images/ft_image1.gifUsing the logarithm helps to bring out details of the Fourier transform in regions where *F*(*ω*1,*ω*2) is very close to 0.Examples of the Fourier transform for other simple shapes are shown below.Fourier Transforms of Some Simple Shapeshttp://www.mathworks.in/help/toolbox/images/transform38.gif[Back to Top of Page](http://www.mathworks.in/help/toolbox/images/f21-17064.html%22%20%5Cl%20%22top_of_page)[Back to Top](http://www.mathworks.in/help/toolbox/images/f21-17064.html#top_of_page)Discrete Fourier TransformWorking with the Fourier transform on a computer usually involves a form of the transform known as the discrete Fourier transform (DFT). A discrete transform is a transform whose input and output values are discrete samples, making it convenient for computer manipulation. There are two principal reasons for using this form of the transform:* The input and output of the DFT are both discrete, which makes it convenient for computer manipulations.
* There is a fast algorithm for computing the DFT known as the fast Fourier transform (FFT).

The DFT is usually defined for a discrete function *f*(*m*,*n*) that is nonzero only over the finite region http://www.mathworks.in/help/toolbox/images/eqn1248890637.gifand http://www.mathworks.in/help/toolbox/images/eqn1248890671.gif. The two-dimensional M-by-N DFT and inverse M-by-N DFT relationships are given byhttp://www.mathworks.in/help/toolbox/images/eqn1248894858.gifandhttp://www.mathworks.in/help/toolbox/images/eqn1248890703.gifThe values *F*(*p*,*q*) are the DFT coefficients of *f*(*m*,*n*). The zero-frequency coefficient, *F*(0,0), is often called the "DC component." DC is an electrical engineering term that stands for direct current. (Note that matrix indices in MATLAB always start at 1 rather than 0; therefore, the matrix elements *f*(1,1) and *F*(1,1) correspond to the mathematical quantities *f*(0,0) and *F*(0,0), respectively.)The MATLAB functions [fft](http://www.mathworks.in/help/techdoc/ref/fft.html), [fft2](http://www.mathworks.in/help/techdoc/ref/fft2.html), and [fftn](http://www.mathworks.in/help/techdoc/ref/fftn.html) implement the fast Fourier transform algorithm for computing the one-dimensional DFT, two-dimensional DFT, and N-dimensional DFT, respectively. The functions [ifft](http://www.mathworks.in/help/techdoc/ref/ifft.html), [ifft2](http://www.mathworks.in/help/techdoc/ref/ifft2.html), and [ifftn](http://www.mathworks.in/help/techdoc/ref/ifftn.html) compute the inverse DFT.Relationship to the Fourier TransformThe DFT coefficients *F*(*p*,*q*) are samples of the Fourier transform *F*(*ω*1,*ω*2).http://www.mathworks.in/help/toolbox/images/eqn1210352983.gifVisualizing the Discrete Fourier Transform1. Construct a matrix f that is similar to the function *f*(*m*,*n*) in the example in [Definition of Fourier Transform](http://www.mathworks.in/help/toolbox/images/f21-17064.html#f21-17097). Remember that *f*(*m*,*n*) is equal to 1 within the rectangular region and 0 elsewhere. Use a binary image to represent *f*(*m*,*n*).
2. f = zeros(30,30);
3. f(5:24,13:17) = 1;

imshow(f,'InitialMagnification','fit')http://www.mathworks.in/help/toolbox/images/fourier_bw.gif1. Compute and visualize the 30-by-30 DFT of f with these commands.
2. F = fft2(f);
3. F2 = log(abs(F));
4. imshow(F2,[-1 5],'InitialMagnification','fit');

colormap(jet); colorbarDiscrete Fourier Transform Computed Without Paddinghttp://www.mathworks.in/help/toolbox/images/ft_image2.gifThis plot differs from the Fourier transform displayed in [Visualizing the Fourier Transform](http://www.mathworks.in/help/toolbox/images/f21-17064.html#f21-16292). First, the sampling of the Fourier transform is much coarser. Second, the zero-frequency coefficient is displayed in the upper left corner instead of the traditional location in the center.1. To obtain a finer sampling of the Fourier transform, add zero padding to f when computing its DFT. The zero padding and DFT computation can be performed in a single step with this command.

F = fft2(f,256,256);This command zero-pads f to be 256-by-256 before computing the DFT. imshow(log(abs(F)),[-1 5]); colormap(jet); colorbarDiscrete Fourier Transform Computed with Paddinghttp://www.mathworks.in/help/toolbox/images/ft_image3.gif1. The zero-frequency coefficient, however, is still displayed in the upper left corner rather than the center. You can fix this problem by using the function fftshift, which swaps the quadrants of F so that the zero-frequency coefficient is in the center.
2. F = fft2(f,256,256);F2 = fftshift(F);

imshow(log(abs(F2)),[-1 5]); colormap(jet); colorbarThe resulting plot is identical to the one shown in [Visualizing the Fourier Transform](http://www.mathworks.in/help/toolbox/images/f21-17064.html#f21-16292).[Back to Top of Page](http://www.mathworks.in/help/toolbox/images/f21-17064.html%22%20%5Cl%20%22top_of_page)[Back to Top](http://www.mathworks.in/help/toolbox/images/f21-17064.html#top_of_page)Applications of the Fourier TransformThis section presents a few of the many image processing-related applications of the Fourier transform.Frequency Response of Linear FiltersThe Fourier transform of the impulse response of a linear filter gives the frequency response of the filter. The function freqz2 computes and displays a filter's frequency response. The frequency response of the Gaussian convolution kernel shows that this filter passes low frequencies and attenuates high frequencies.h = fspecial('gaussian');freqz2(h)Frequency Response of a Gaussian Filterhttp://www.mathworks.in/help/toolbox/images/gaussian.gifSee [Designing and Implementing 2-D Linear Filters for Image Data](http://www.mathworks.in/help/toolbox/images/f16-15268.html) for more information about linear filtering, filter design, and frequency responses.Fast ConvolutionA key property of the Fourier transform is that the multiplication of two Fourier transforms corresponds to the convolution of the associated spatial functions. This property, together with the fast Fourier transform, forms the basis for a fast convolution algorithm.

|  |
| --- |
| **Note**   The FFT-based convolution method is most often used for large inputs. For small inputs it is generally faster to use imfilter. |

To illustrate, this example performs the convolution of A and B, where A is an M-by-N matrix and B is a P-by-Q matrix:1. Create two matrices.
2. A = magic(3);

B = ones(3);1. Zero-pad A and B so that they are at least (M+P-1)-by-(N+Q-1). (Often A and B are zero-padded to a size that is a power of 2 because fft2 is fastest for these sizes.) The example pads the matrices to be 8-by-8.
2. A(8,8) = 0;

B(8,8) = 0;1. Compute the two-dimensional DFT of A and B using fft2, multiply the two DFTs together, and compute the inverse two-dimensional DFT of the result using ifft2

C = ifft2(fft2(A).\*fft2(B));1. Extract the nonzero portion of the result and remove the imaginary part caused by roundoff error.
2. C = C(1:5,1:5);

C = real(C)This example produces the following result.C = 8.0000 9.0000 15.0000 7.0000 6.0000 11.0000 17.0000 30.0000 19.0000 13.0000 15.0000 30.0000 45.0000 30.0000 15.0000 7.0000 21.0000 30.0000 23.0000 9.0000 4.0000 13.0000 15.0000 11.0000 2.0000Locating Image FeaturesThe Fourier transform can also be used to perform correlation, which is closely related to convolution. Correlation can be used to locate features within an image; in this context correlation is often called template matching.This example illustrates how to use correlation to locate occurrences of the letter "a" in an image containing text:1. Read in the sample image.

bw = imread('text.png');1. Create a template for matching by extracting the letter "a" from the image.

a = bw(32:45,88:98);You can also create the template image by using the interactive version of imcrop. The following figure shows both the original image and the template.imshow(bw);figure, imshow(a);Image (left) and the Template to Correlate (right)http://www.mathworks.in/help/toolbox/images/fft_letter_a.gif1. Compute the correlation of the template image with the original image by rotating the template image by 180o and then using the FFT-based convolution technique described in [Fast Convolution](http://www.mathworks.in/help/toolbox/images/f21-17064.html#f21-16374).

(Convolution is equivalent to correlation if you rotate the convolution kernel by 180o.) To match the template to the image, use the fft2 and ifft2 functions.C = real(ifft2(fft2(bw) .\* fft2(rot90(a,2),256,256)));The following image shows the result of the correlation. Bright peaks in the image correspond to occurrences of the letter.figure, imshow(C,[]) % Scale image to appropriate display range.Correlated Imagehttp://www.mathworks.in/help/toolbox/images/text_fft.gif1. To view the locations of the template in the image, find the maximum pixel value and then define a threshold value that is less than this maximum. The locations of these peaks are indicated by the white spots in the thresholded correlation image. (To make the locations easier to see in this figure, the thresholded image has been dilated to enlarge the size of the points.)
2. max(C(:))
3. ans =
4. 68.0000
5. thresh = 60; % Use a threshold that's a little less than max.

figure, imshow(C > thresh) % Display pixels over threshold.Correlated, Thresholded Image Showing Template Locationshttp://www.mathworks.in/help/toolbox/images/text_fft_thresh.gif

|  |
| --- |
|   |

 |