# Fourier Transform Model of Image Formation

## Lens as Fourier Transformer

The Fourier Transform (FT) is one among the many classes of linear transforms that recast information into another set of linear basis functions. In particular, FT converts a signal with physical dimension X into a signal with dimension 1/X. If the signal has dimensions of space, a Fourier Transform of the signal will be inverse space or spatial frequency. The FT of an image f(x,y) is given by

$$F(f_x, f_y) = \iint f(x, y) \exp(-i\, 2\pi(f_x x + f_y y))\, dx\, dy \qquad (1)$$

where $f_x$ and $f_y$ are the spatial frequencies along x and y, respectively. An efficient and fast implementation of FT is the Fast Fourier Transform algorithm or FFT by Cooley and Tukey. So powerful and useful is the algorithm that all signal and image processing software have FFT routines among their functions. In Scilab the function is **`fft()`** for 1D signals and **`fft2()`** for 2D signals.

The discrete FFT or FFT2 has the following properties.

1. The FFT2 is most efficient (fastest) if the image size is a power of 2 (e.g.64×64, 128×128, 256×256, etc.)
2. The output of FFT2 has quadrants along the diagonals interchanged. That is, if the image quadrants are labeled in the clockwise direction as $\begin{matrix} 4 & 1 \\ 3 & 2 \end{matrix}$, the FFT comes out as $\begin{matrix} 2 & 3 \\ 1 & 4 \end{matrix}$. The **`fftshift()`** function interchanges the quadrants back.
3. The output of the FFT is a **complex number array**. To view the intensity use **`abs()`** function which computes the modulus of a complex number.
4. The inverse FFT is just the same as the forward FFT with the image inverted.

© 2019 Maricor Soriano

# Activity 1. Familiarization with discrete FFT

1. Create a 128×128 image of a white circle, centered and with the background black.
2. Open the image in Scilab using **`imread()`** and if the image variable is a 128 x 128 x 3 matrix convert to grayscale using **`RGB2Gray().`**
3. Apply **`fft2()`** on the image and compute the intensity values using **`abs()`**. Display the output using **`imshow()`**.

```
I = imread('circle.bmp');
Igray = RGB2Gray(I); //do this step only if it I is
an MxNx3 matrix.
FIgray = fft2(double(Igray)); //remember, FIgray is
complex
imshow(uint8(abs(FIgray));   //abs   computes   the
modulo of the complex variable FIgray
```

Alternatively, you can create a circle within Scilab using the following code:

```
x = [-1:0.1:1];
[X,Y] = meshgrid(x);
r = sqrt(X.^2 + Y.^2);
circle = zeros(size(X,1), size(X,2));
circle(find (r <=0.5)) = 1.0;
imshow(circle);
```

4. Now apply **`fftshift()`** on the image and again observe the output. Is the FT you observe consistent with the analytical fourier transform of a circle?

```
imshow(uint8(fftshift(abs(FIgray))));
```

5. Apply **`fft2()`** again on the fft2 output FIgray and display the intensity image. Alternatively, you can apply fft2 on the image I twice. Explain what you observe.

```
imshow(uint8(abs(fft2(FIgray))));
```

or

```
imshow(unt8(abs(fft2(fft2(I)))));
```

6. Perform steps 2 to 5 for a 128×128 image of a letter “A” and explain what you observe.


© 2019 Maricor Soriano

7. To convince yourself that the FT of a function is complex, display separately, the real and imaginary parts of the circle and the “A” image.
8. Apply FT to the following patterns
    1. Sinusoid along x (corrugated roof)
    2. Simulated double slit
    3. Square function
    4. 2D Gaussian bell curve

## Convolution

The convolution between two 2-dimensional functions f and g is given by

$$h(x,y)=\iint f(x',y')\,g(x-x',y-y')\,dx'\,dy'. \tag{2}$$

The shorthand notation for convolution is “*”, i.e.

$$h=f*g. \tag{3}$$

The convolution is a linear operation which means that if f and g are recast by linear transformations, such as the Laplace or Fourier transform, they obey the convolution theorem:

$$H=FG \tag{4}$$

where H , F and G are the transforms of h, f and g, respectively. The convolution is a ***"smearing"*** of one function against another such that the resulting function h looks a little like both f and g. Convolution is used to model the linear regime of instruments or detection devices such as those used in imaging. For example, in imaging, f can be the object, g can be the transfer function of the imaging system. Their convolution, h, is then the image produced by the detection system which in general is not 100% identical to the original object.

With a digital camera, the transfer function is mostly due to the finite size of the camera lens. A finite lens radius means the lens can only gather a limited number of rays reflected off an object therefore reconstruction of the object is never perfect.

© 2019 Maricor Soriano 

# Activity 2. Simulation of an imaging device.

1. Create a 128×128 image of the letters "**VIP**" using Paint (Arial font recommended). Let the letters fill 50% of the space. This represents your object. Note that the edges of a sans serif font are sharp.
2. Create another 128×128 image of a white circle (centered) against a black background. This image represents the "aperture" of a circular lens.
3. Open both images in Scilab as grayscale images, get the 2D FFT of the VIP image and fftshift the aperture image.

```
r=imread('C:\MyDocuments\circle.bmp');
a=imread('C:\MyDocuments\VIP.bmp');
rgray = rgb2gray(r);
agray = rgb2gray(a);
Fr = fftshift(rgray);
//aperture is already in the Fourier Plane and need
not be FFT'ed
Fa = fft2(agray);
```

4. Get the product of their FFT as in Equation (4) and get the inverse to get the convolved image. Display the output.

```
FRA = Fr.*(Fa);
IRA = fft2(FRA); //inverse FFT
FImage = abs(IRA);
imshow(uint8(FImage));
```

5. Compare the original "**VIP**" and the "imaged" "**VIP**" for different radii of the white circle. Discuss the effect of the aperture of the lens on the quality of the image.

# Correlation

The correlation between two 2-dimensional functions f and g is given by

$$p(x,y)=\iint f(x',y')\,g(x+x',y+y')\,dx'\,dy' \qquad (5)$$

or in shorthand notation

$$p=f\odot g. \qquad (6)$$

It is related to the convolution integral by

$$p=f\odot g=f(-x,-y)*g(x,y) \qquad (7)$$


© 2019 Maricor Soriano

where the superscript asterisk at f means complex conjugation.

Similar to the convolution, there is a correlation theorem which holds for linear transforms of f and g:

$$P=F*G \tag{8}$$

where P, F and G are the Fourier transforms of p, f and g and the asterisk above F stands for complex conjugation.

The correlation p measures the degree of similarity between two functions f and g. The more identical they are at a certain position x,y, the higher their correlation value. Therefore, the correlation function is used mostly in template matching or pattern recognition as was done in correlation.sce.

An important consequence of Equation (7) is that if f or g is an even function, the correlation is equal to the convolution.

## Activity 3. Template Matching using correlation

Template Matching is a pattern recognition technique suitable for finding exactly identical patterns in a scene such as in the case of finding a certain word in a document.

1. Type the following text in a 128×128 white background using Paint "THE RAIN IN SPAIN STAYS MAINLY IN THE PLAIN."
2. Create a 128×128 image of the letter "A" using the same font and font size as the image in step 1.
3. Open both images in Scilab as grayscale and get both their Fourier Transforms.
4. Element-per element multiply the FT of A and the conjugate of the FT of the text image. Use `conj()` to get complex conjugate.
5. Compute the inverse FFT and display the output. Explain the meaning of the output you obtain.

## Activity 4. Edge detection using the convolution integral

Edge detection can be seen as template matching of an edge pattern with an

© 2019 Maricor Soriano 

image.

1. Create a $3\times3$ matrix pattern of an edge such that the total sum is zero. E.g. $\begin{bmatrix} -1 & -1 & -1 \\ 2 & 2 & 2 \\ -1 & -1 & -1 \end{bmatrix}$ or variants in other direction. In Scilab, this can be called by

   ```
   pattern = [-1 -1 -1; 2  2  2; -1 -1 -1];
   ```

2. Convolve this image with the **VIP** image. Try other direction patterns such as vertical, $\begin{bmatrix} -1 & 2 & -1 \\ -1 & 2 & -1 \\ -1 & 2 & -1 \end{bmatrix}$, diagonal or try a spot pattern $\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$ and comment on the convolved image for various patterns.


© 2019 Maricor Soriano