SCSV3213

FUNDAMENTAL OF IMAGE PROCESSING

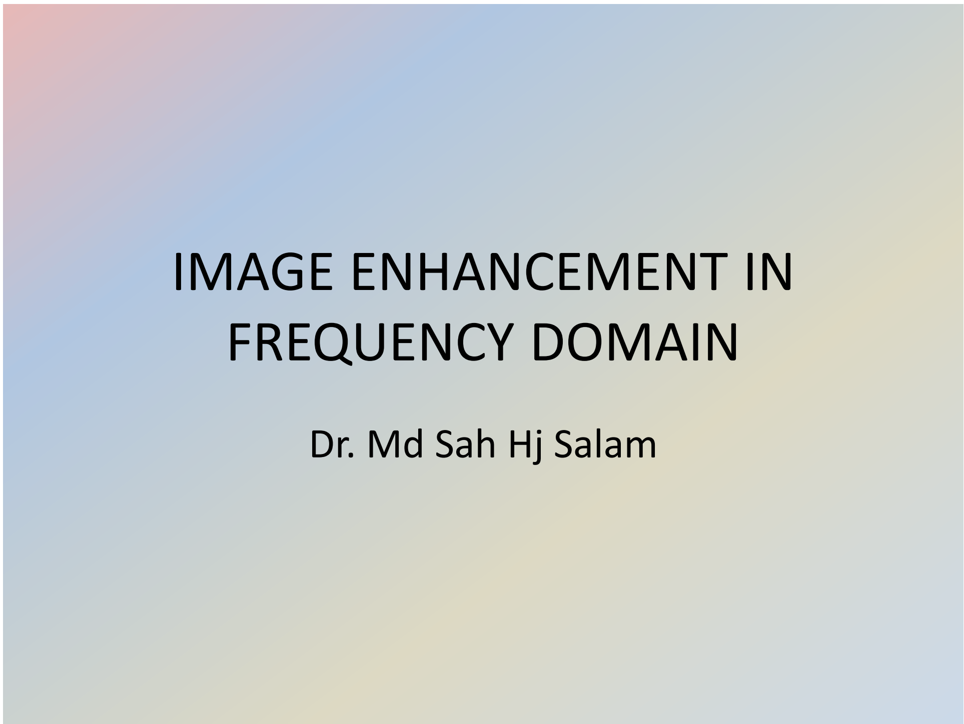


IMAGE ENHANCEMENT IN FREQUENCY DOMAIN

Dr. Md Sah Hj Salam

Acknowledgements

Most of the slide are taken and modified from other resources including books and slides from lectures from others universities especially from Oge Marques. It is rearranged to suit the syllabus of the course.

SYNOPSIS

In this lecture, frequency domain filtering are described for both the low pass filter and also high pass filter. Students are expected able to know the processed involved in frequency domain filtering and why some operation are better operate in frequency domain. The frequency domain transform exaplain in this lecture is mainly on Fourier Transform.

1. FOURIER TRANSFORM

Jean Baptiste Joseph Fourier (1768-1830)

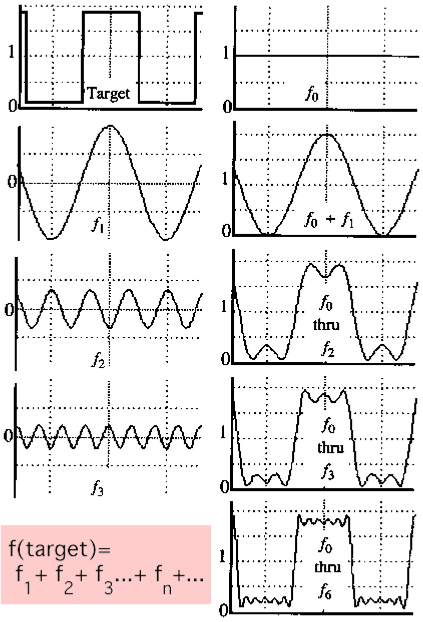
- Had crazy idea (1807):
- **Any** periodic function can be rewritten as a weighted sum of Sines and Cosines of different frequencies.
- Don’t believe it?
  - Neither did Lagrange, Laplace, Poisson and other big wigs
  - Not translated into English until 1878!
- But it’s true!
  - called Fourier Series
  - Possibly the greatest tool used in Engineering

A Sum of Sinusoids

- Our building block:

$$A\sin(\omega x+\phi)$$

- Add enough of them to get any signal *f(x)* you want!
- How many degrees of freedom?
- What does each control?
- Which one encodes the coarse vs. fine structure of the signal?



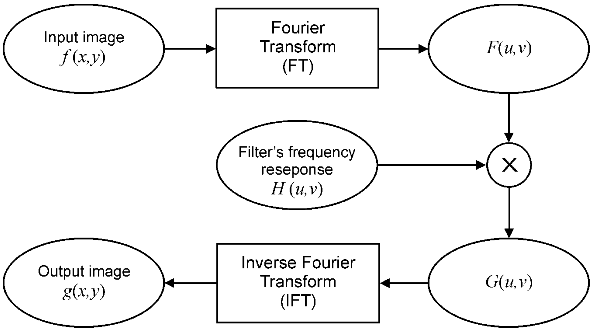
f(target)= $f_1 + f_2 + f_3 ... + f_n + ...$

Fourier Transform

- It is a mathematical tool just like calculus used in engineering to solve engineering problems like
  - Speech processing
  - Image proccessing
  - Computer Graphics
  - etc
- We will not go depth in the mathematical theory of FT. Refer notes from others on FT.

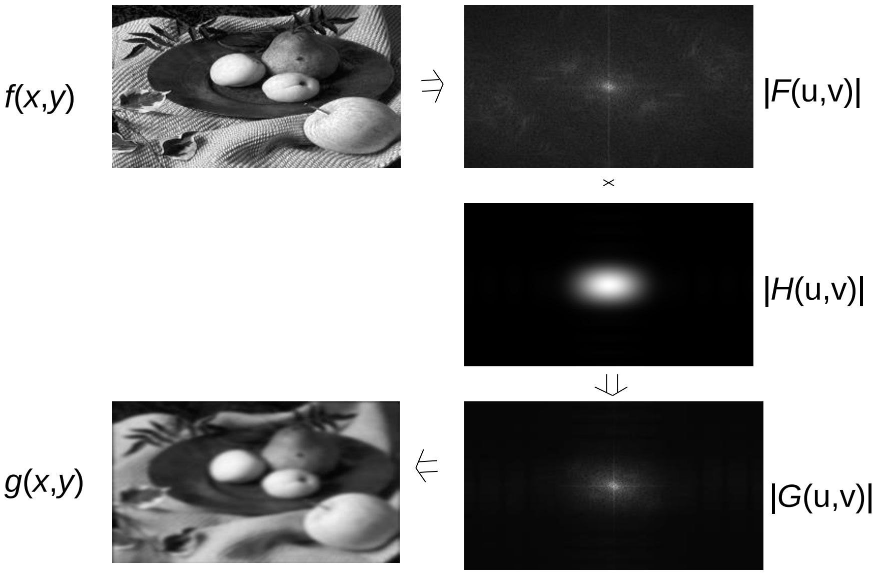
2. FREQUENCY DOMAIN OPERATION

Frequency-domain operations



By Oge Marques

Another way to look at it



Mathematical foundation

- Convolution theorem

$$\begin{tikzcd} g(x,y) \arrow[d, "FT"] & = & f(x,y) \arrow[d] & * & h(x,y) \arrow[d] \\ G(u,v) & = & F(u,v) & & H(u,v) \end{tikzcd}$$

Spatial domain

Freq. domain

$$g(x,y) = \underset{\uparrow}{\mathfrak{F}^{-1}} \left[F(u,v)H(u,v)\right]$$

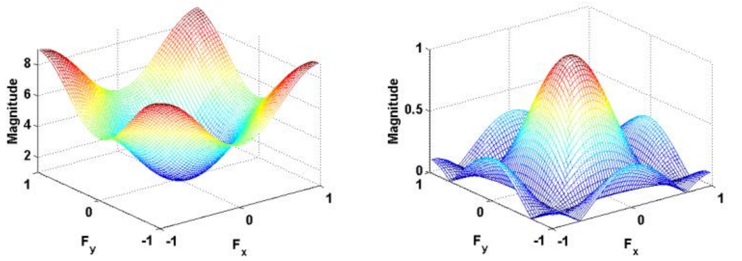
Inverse 2D Fourier Transform (FT)

Designing frequency-domain filters with MATLAB and the IPT

- 2 options:
  1. Obtain the frequency domain filter response function from spatial filter convolution mask. The IPT has a function that does exactly that: **`freqz2`**.
  2. Generate filters directly in the frequency domain. In this case a meshgrid array (of the same size as the image) is created using the MATLAB function **`meshgrid`**.

Examples of response functions for frequency-domain filters

```
h2 = ones(3,3) / 9
freqz2(h2) %(a)
h3 = [0 -1 0; -1 5 -1; 0 -1 0]
freqz2(h3) %(b)
```



(a)

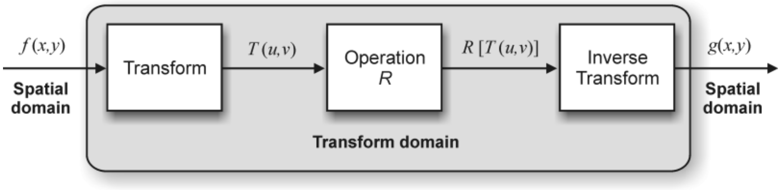
(b)

(a) is low pass filter (b) is the high pass laplacian sharpening filter

Mathematical foundation

- *Transform*: a mathematical tool that allows the conversion of a set of values to another set of values, creating, therefore, a new way of representing the same information.
- In image processing:
  - original domain = spatial domain
  - results of frequency-domain operations lie in the transform domain.
- Motivation for using mathematical transforms in image processing: some tasks (e.g., frequency-domain filtering, image compression) are best performed by applying selected algorithms in the transform domain. transformation to the result

Mathematical foundation



$$T(u,v) = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)\cdot r(x,y,u,v)$$

$$f(x,y) = \sum_{u=0}^{M-1}\sum_{v=0}^{N-1} T(u,v)\cdot s(x,y,u,v)$$

Mathematical foundation

- 2D Fourier transform (in MATLAB: **`fft2`** and **`ifft2`**)

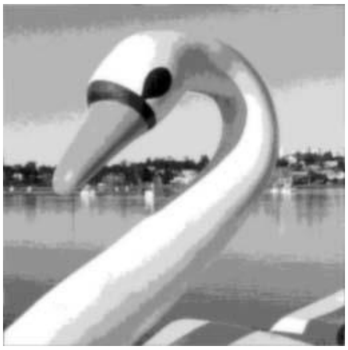
$$F(u,v) = T(u,v) = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)\cdot\exp[-j2\pi(ux/M+vy/N)]$$

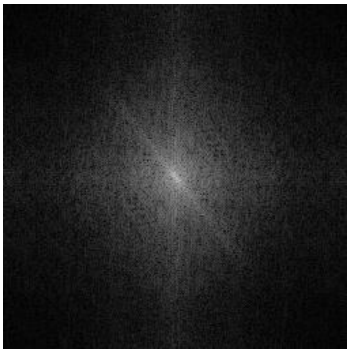
$$f(x,y) = \mathfrak{F}^{-1}[F(u,v)] = \frac{1}{MN}\sum_{u=0}^{M-1}\sum_{v=0}^{N-1} T(u,v)\cdot\exp[j2\pi(ux/M+vy/N)]$$

$$|F(u,v)| = \sqrt{[R(u,v)]^2+[I(u,v)]^2}$$

$$\phi(u,v) = \arctan\left[\frac{I(u,v)}{R(u,v)}\right]$$

Example of an image and its FT





```
I = imread('Figure11_04_a.png');
Id = im2double(I);
ft = fft2(Id);
ft_shift = fftshift(ft);
imshow(log(1 + abs(ft_shift)), [])
```

Fourier Transform: properties

- Linearity

$$\mathfrak{F}[a \cdot f1(x,y) + b \cdot f2(x,y)] = a \cdot F1(u,v) + b \cdot F2(u,v)$$

and

$$a \cdot f1(x,y) + b \cdot f2(x,y) = \mathfrak{F}^{-1}[a \cdot F1(u,v) + b \cdot F2(u,v)]$$

Fourier Transform: properties

- Translation

$$\mathfrak{F}[f(x-x_0, y-y_0)] = F(u,v) \cdot \exp[-j2\pi(ux_0/M + vy_0/N)]$$

and

$$f(x-x_0, y-y_0) = \mathfrak{F}^{-1}[F(u,v) \cdot \exp[j2\pi(ux_0/M + vy_0/N)]$$

Fourier Transform: properties

- Conjugate symmetry

$$F(u,v) = F^*(-u,-v)$$

$F^*(u,v)$ is the conjugate of $F(u,v)$

where:

$$F(u,v) = R(u,v) + jI(u,v)$$

i.e., if:

$$F^*(u,v) = R(u,v) - jI(u,v)$$

then:

$$|F(u,v)| = |F(-u,-v)|$$

By Oge Marques

Fourier Transform: properties

- Periodicity

$$F(u,v) = F(u+M, v+N)$$

and

$$f(x,y) = f(x+M, y+N)$$

By Oge Marques

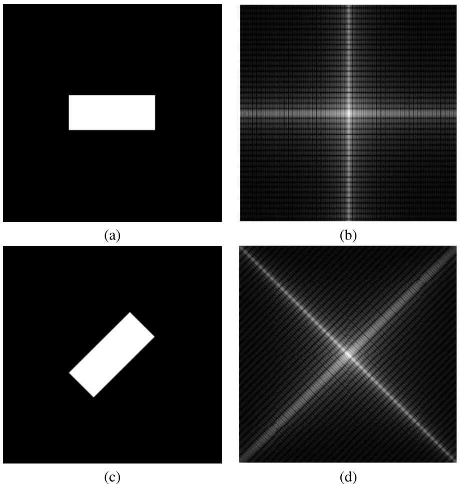
Fourier Transform: properties

- Separability
  - The Fourier Transform is separable, i.e., the FT of a 2D image can be computed by two passes of the 1D FT algorithm, once along the rows (columns), followed by another pass along the columns (rows) of the result.

By Oge Marques

Fourier Transform: properties

- Rotation
  - If an image is rotated by a certain angle ϑ, its 2D FT will be rotated by the same angle.



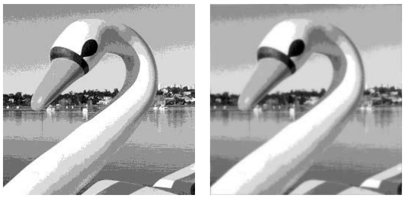
By Oge Marques

Low-pass filtering (LPF)

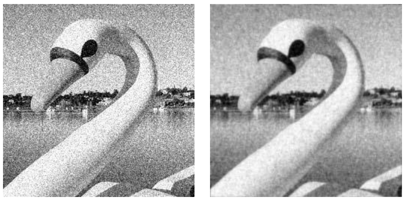
- Low-pass filters attenuate the high frequency components of the Fourier Transform of an image, while leaving the low frequency components unchanged.
- The typical overall effect of applying a low-pass filter (LPF) to an image is a controlled degree of blurring.

Low-pass filtering (LPF)

- Example of LPF for smoothing of false contours

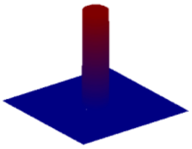


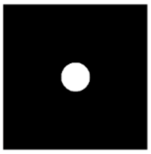
- Example of LPF for noise reduction



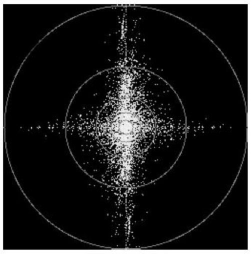
Low-pass filtering

- Ideal LPF



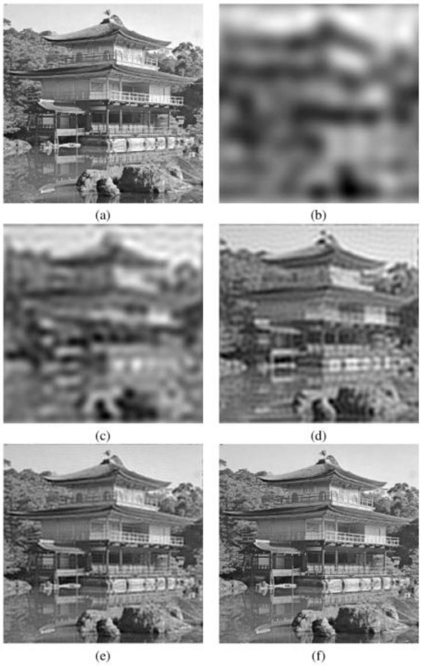


$$H_I(u,v) = \begin{cases} 1 & \text{if } D(u,v) \le D_0 \\ 0 & \text{if } D(u,v) > D_0 \end{cases}$$



Low-pass filtering

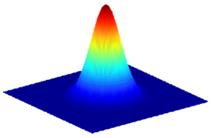
- Ideal LPF – example

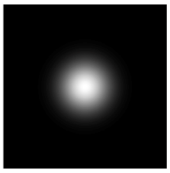


(a) Original 256x256 pixels
(b) - (f) result with cut off radii,, $D_\circ$ 6, 16,32,64 and 128

Low-pass filtering

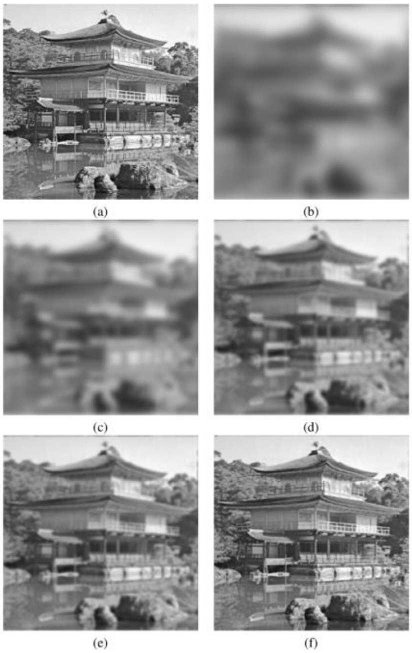
- Gaussian LPF





$$H_G(u,v) = e^{-\frac{D(u,v)^2}{2\sigma^2}}$$

Low-pass filtering

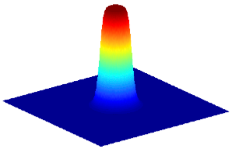


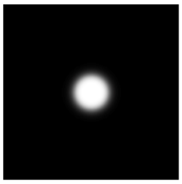
- Gaussian LPF – example

(a) Original 256x256 pixels
(b) - (f) result with diff σ, 5 , 10,20,30 and 75

Low-pass filtering

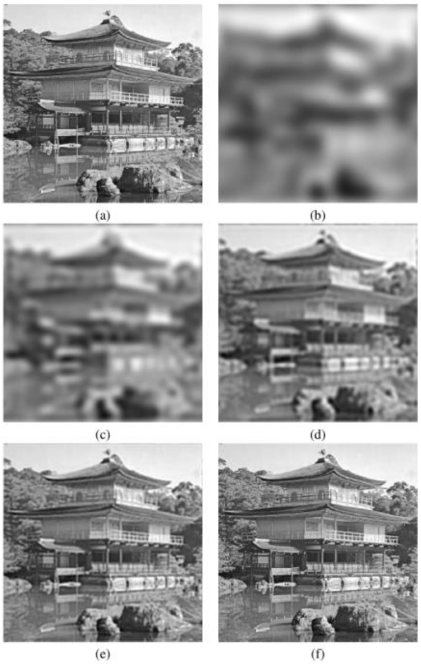
- Butterworth LPF





$$H_B(u,v) = \frac{1}{1 + [D(u,v)/D_0]^{2n}}$$

Low-pass filtering



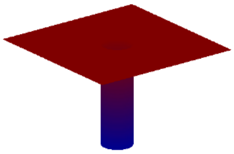
- Butterworth LPF – example

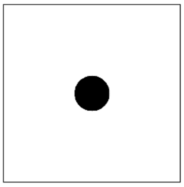
(a) Original 512x512 pixels
(b) - (f) result with cut off radii, D_o 6, 16,32,64 and 128

High-pass filtering

- Ideal HPF

$$H_I(u,v) = \begin{cases} 0 & \text{if } D(u,v) \leq D_0 \\ 1 & \text{if } D(u,v) > D_0 \end{cases}$$

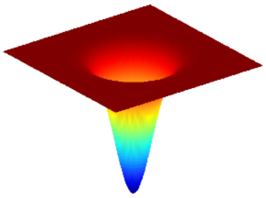


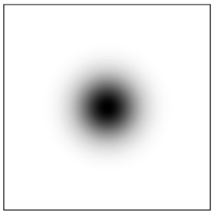


High-pass filtering

- Gaussian HPF

$$H_G(u,v) = 1 - e^{-\frac{D(u,v)^2}{2\sigma^2}}$$

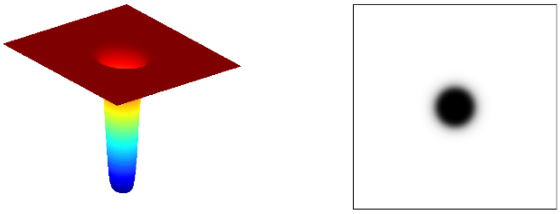




High-pass filtering

- Butterworth HPF

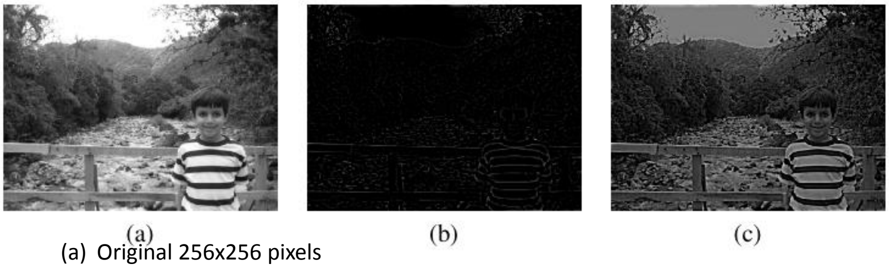
$$H_B(u,v) = \frac{1}{1 + [D_0/D(u,v)]^{2n}}$$



By Oge Marques

High-frequency emphasis

$$H_{hfe}(u,v) = a + bH(u,v)$$



(a) Original 256x256 pixels
(b) Result after applying second order Butterworth HPF(D_o = 30)
(c) Result after applying high freq emphasis (a = 0.5 and b = 1)

By Oge Marques

TUTORIAL FREQ DOMAIN FILTER

TUTORIAL1 :
How to Display a Fourier Spectrum using MATLAB

```
f=zeros(30,30);      %Create a black 30x30 image
f(5:24,13:17)=1;    %With a white rectangle in it.
imshow(f,'InitialMagnification', 'fit');  % show the image
```

```
 F=fft2(f);   %Calculate the DFT.
%There are real and imaginary parts to F. Use the abs function to compute the
% magnitude of the combined components.
 F2=abs(F);
figure, imshow(F2,[], 'InitialMagnification','fit')  % show the image
```

```
 %To create a finer sampling of the Fourier transform, you can add zero %
%padding to f when computing its DFT. Also note that we use a power of 2,
%2^256 . This is because the FFT -Fast Fourier Transform - is fastest when the
% image size has many factors.
F=fft2(f, 256, 256);  F2=abs(F);
figure, imshow(F2, [])
```

TUTORIAL1 : continue
How to Display a Fourier Spectrum using MATLAB

```
%The zero-frequency coefficient is displayed in the upper left hand corner. To
% display it in the center, you can use the function fftshift.

 F2=fftshift(F); F2=abs(F2); figure,imshow(F2,[])
```

```
%In Fourier transforms, high peaks are so high they hide details.
%Reduce contrast with the log function.
F2=log(1+F2); figure,imshow(F2,[])
```

TUTORIAL 2: Low Pass Filter in Freq Domain

- Download materials in elearning

TUTORIAL 3: Narrow High Filter in Freq Domain

- Download materials in elearning

End :
Frequency Domain Enhancemnet
SCSV 3213