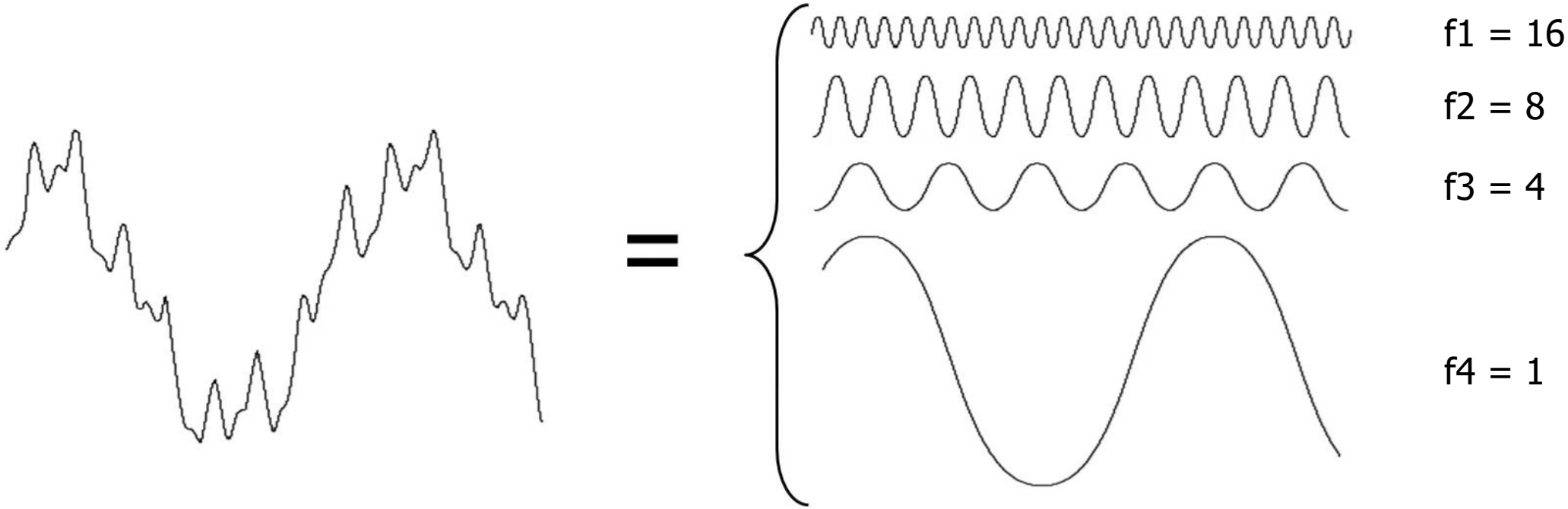
Xử lý ảnh & Thị giác máy tính

Chương 2. Các phép biến đổi

- Tích chập (Convolution)
- Biến đổi Fourier (Fourier Transform)
- Biến đổi DCT (Discrete)

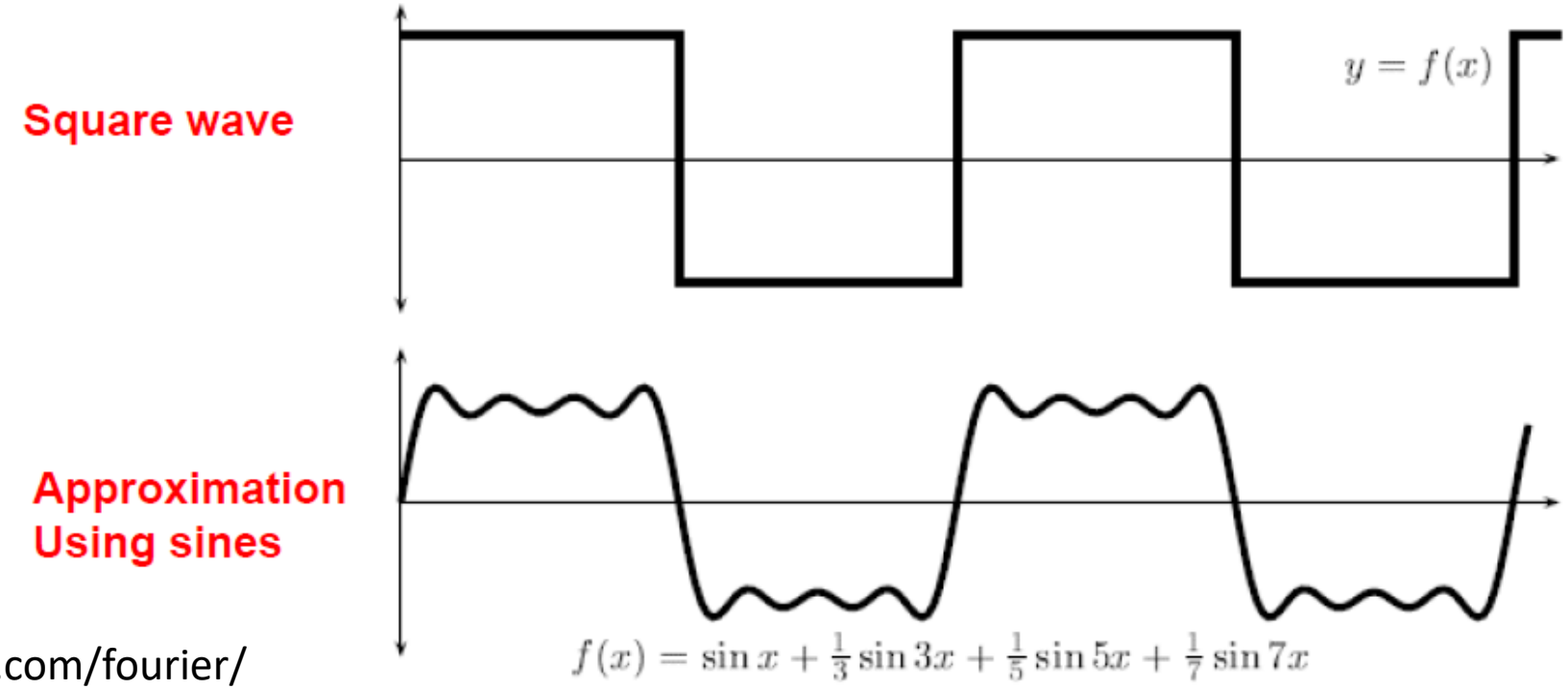
2.2 Ôn tập biến đổi Fourier

- Biến đổi một tín hiệu (thực) về miền tần số (số phức)
- Một tín hiệu có thể biến đổi thành tổng của nhiều hàm sin hoặc cos với các tần số khác nhau



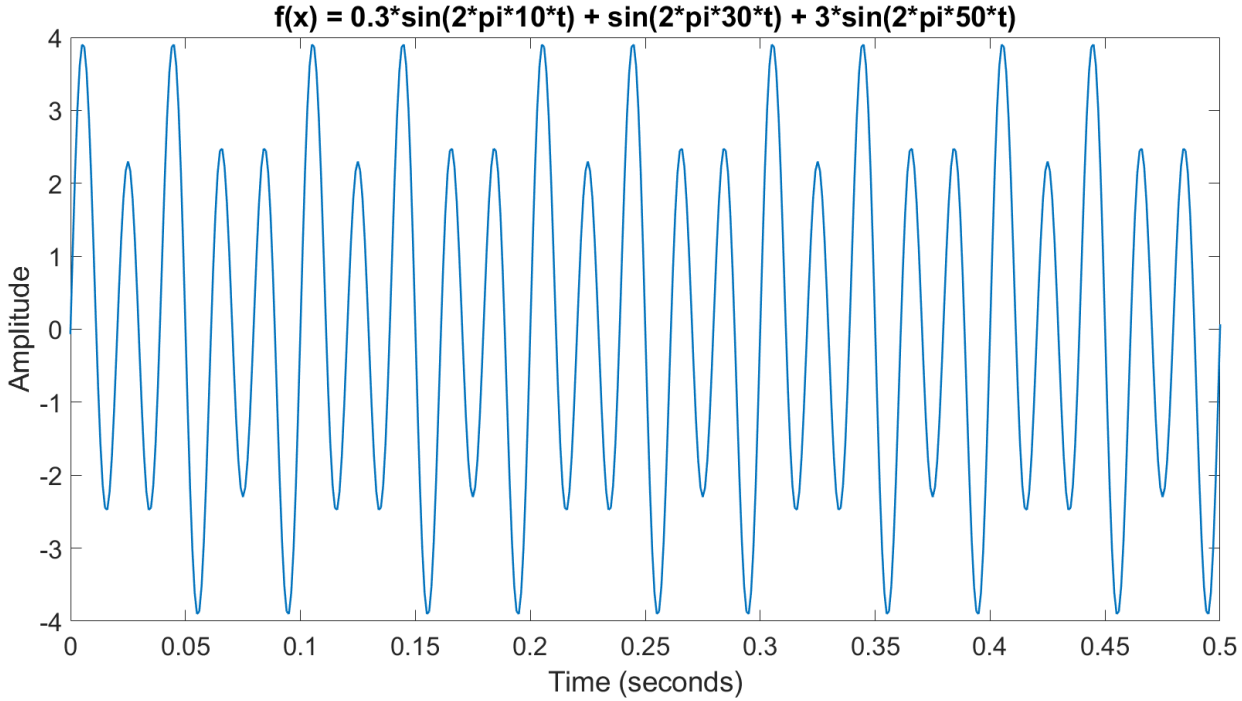
Ví dụ

- Tính xấp xỉ hàm square wave thành tổng 4 hàm sin
- Càng nhiều hàm sin thì xấp xỉ sẽ càng chính xác

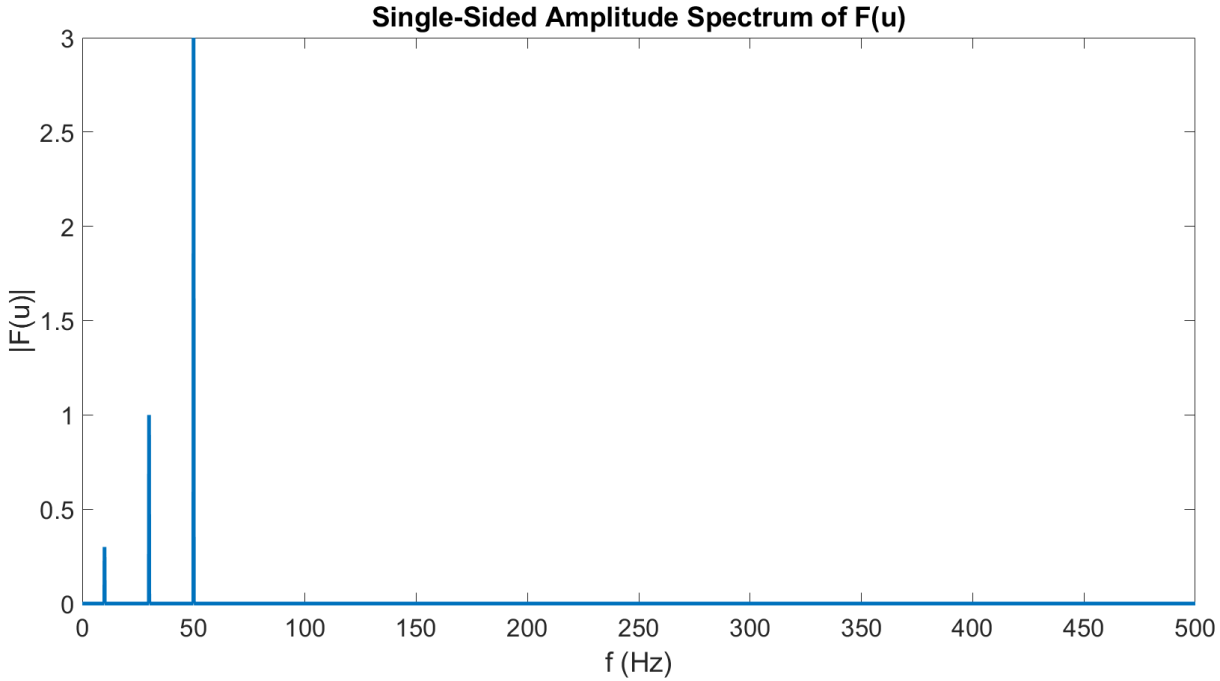


http://www.falstad.com/fourier/

Ví dụ



$f(x) = 0.3\sin(20\pi t) + \sin(60\pi t) + 3\sin(100\pi t)$

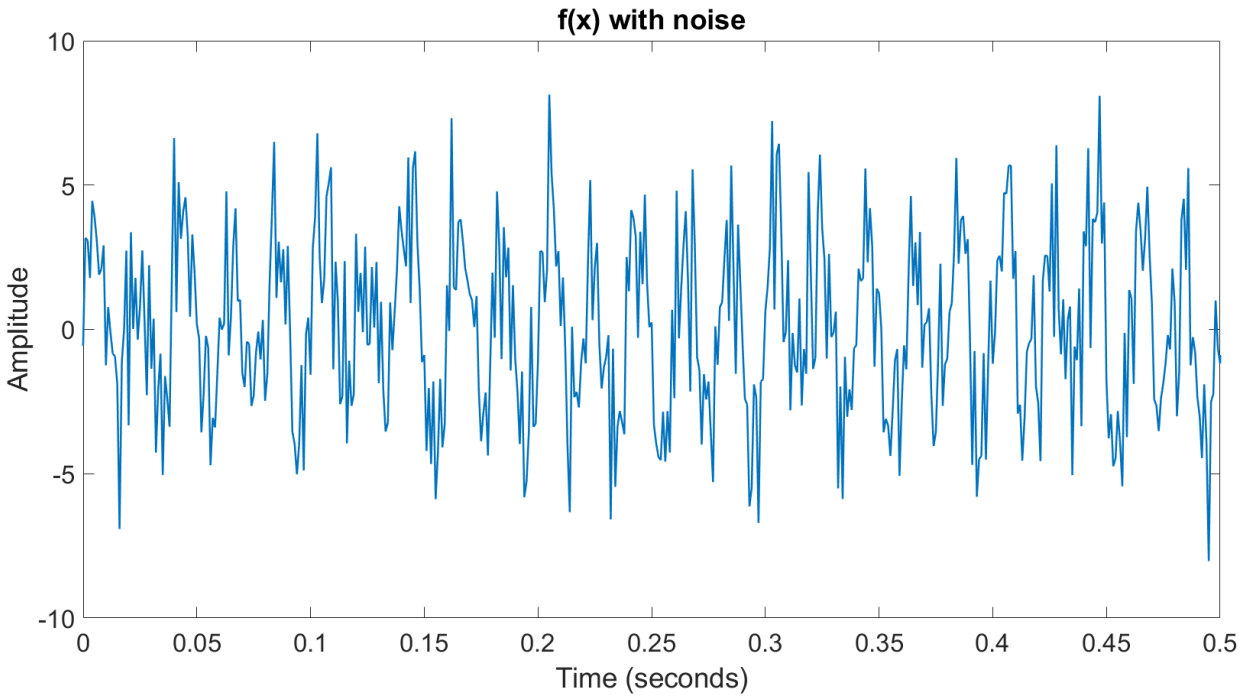


$F(u)$

Tín hiệu gốc f(x) tập trung tại các tần số f = 10Hz, 30Hz, 50Hz

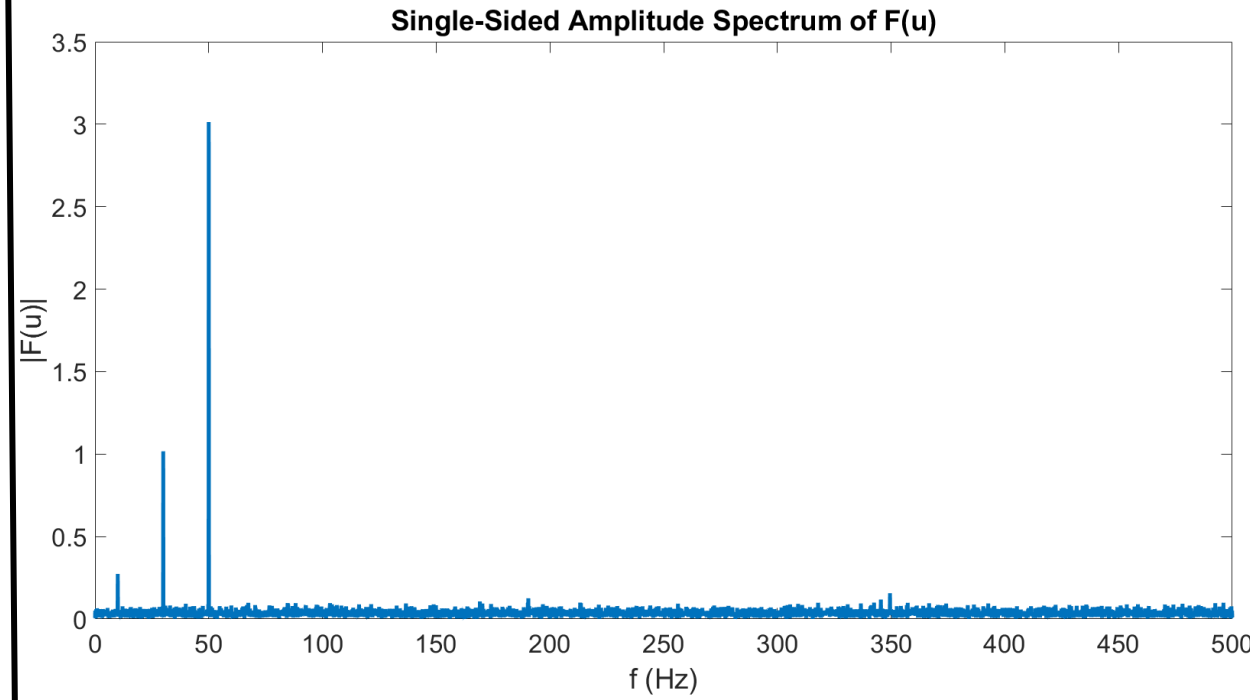
Ví dụ

f(x)



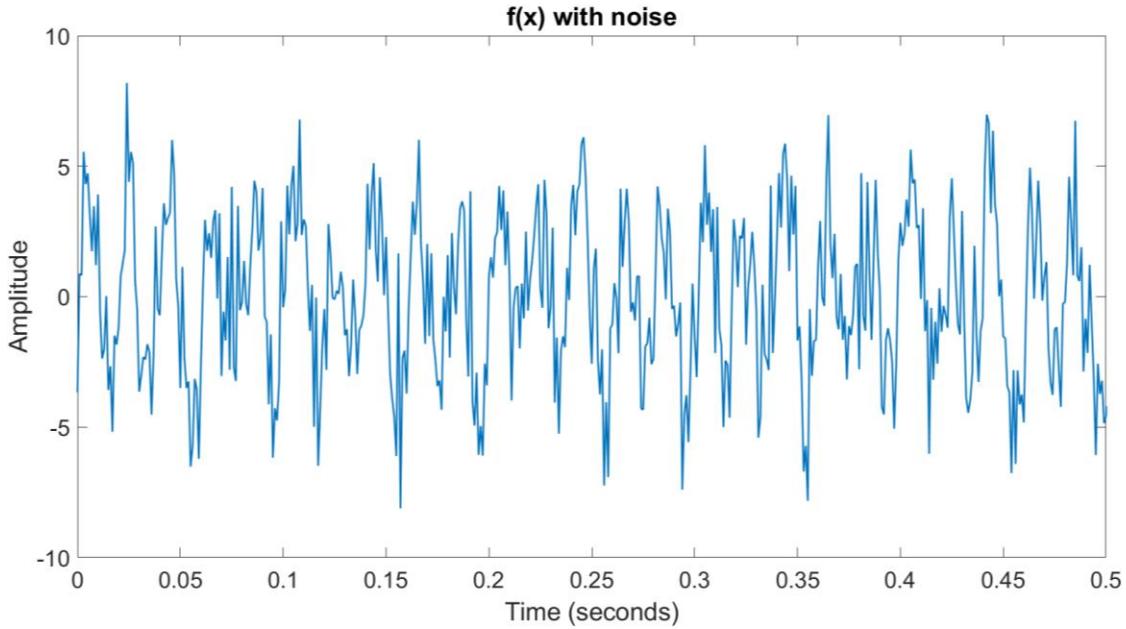
$f(x) = 0.3\sin(20\pi t) + \sin(60\pi t) + 3\sin(100\pi t)$ + **Noise**

F(u)

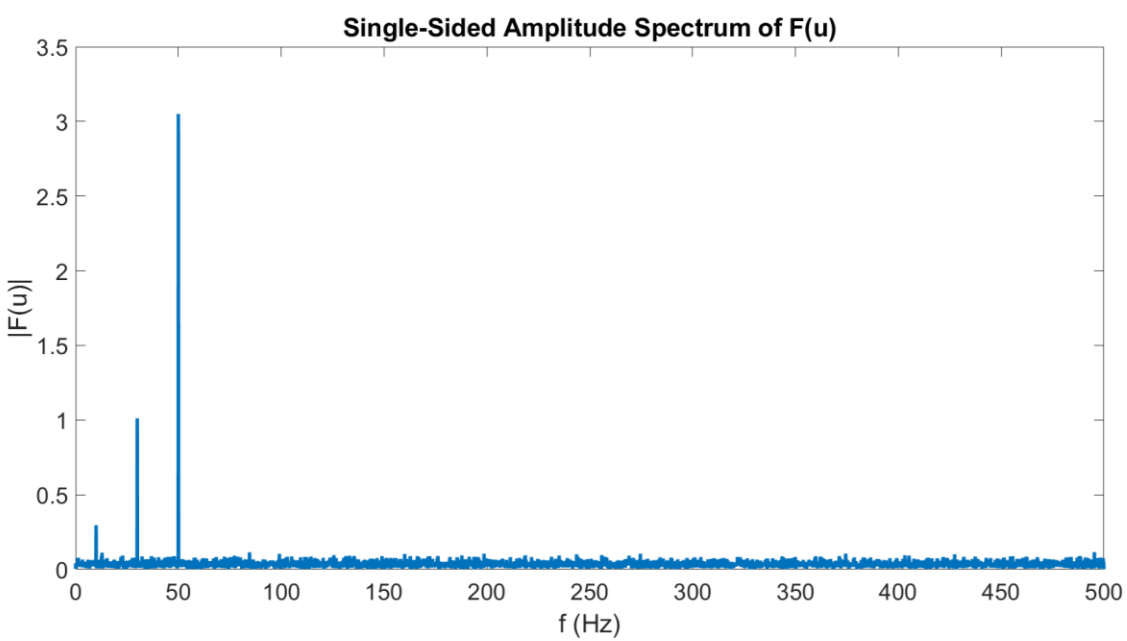


Tín hiệu gốc f(x) tập trung tại các tần số

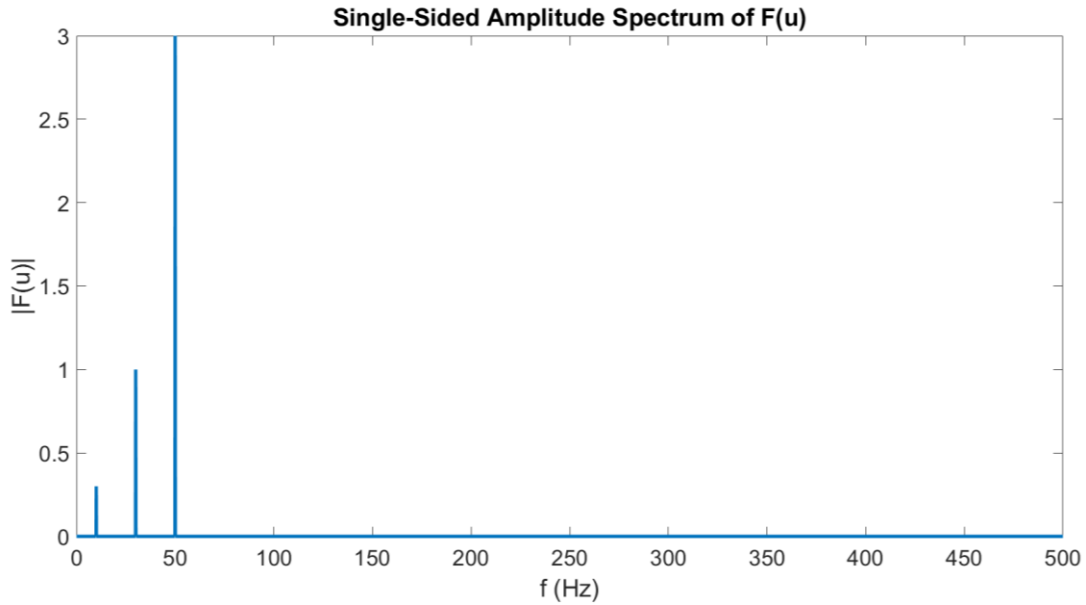
f = 10Hz, 30Hz, 50Hz



DFT

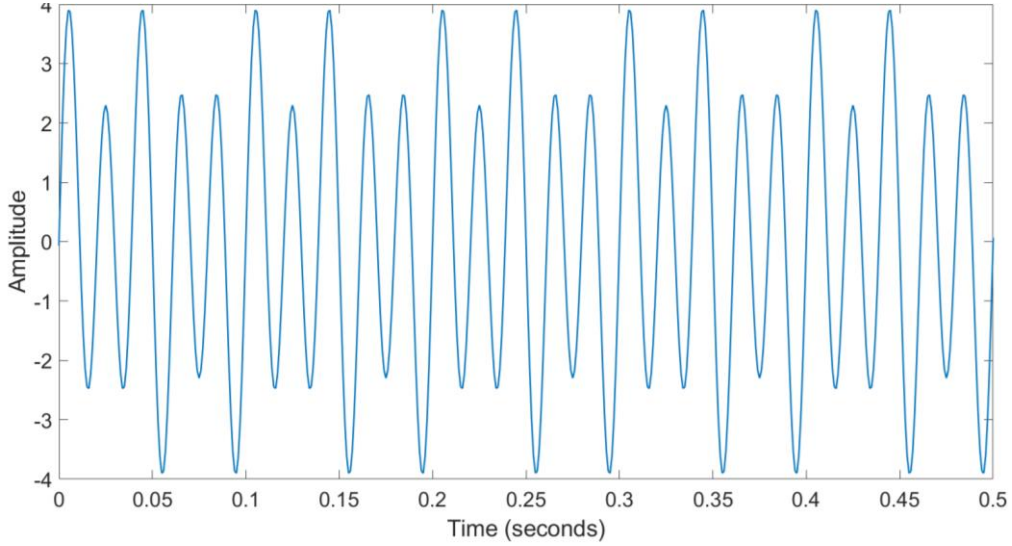


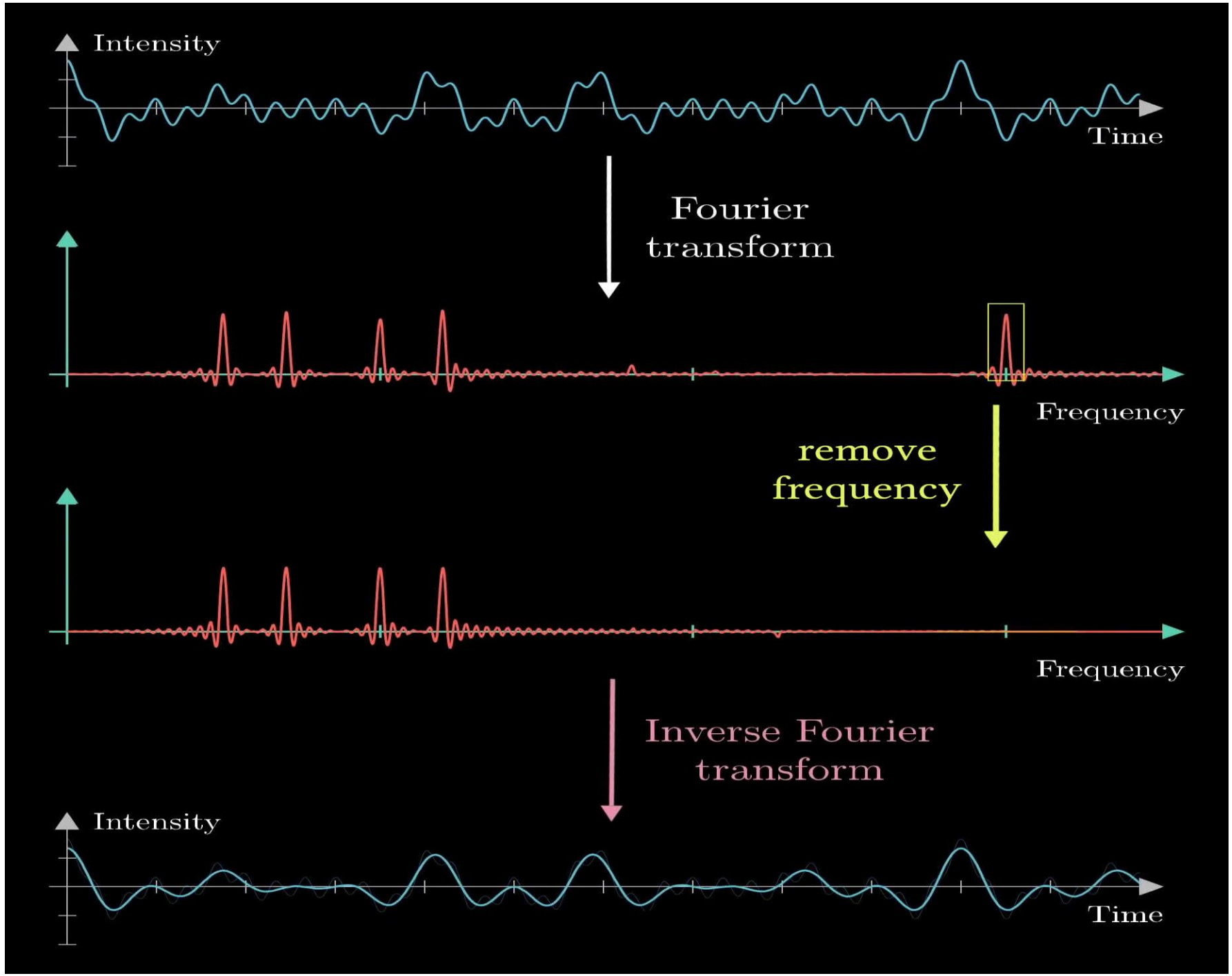
Filter



DFT ngược

Tín hiệu được cải thiện



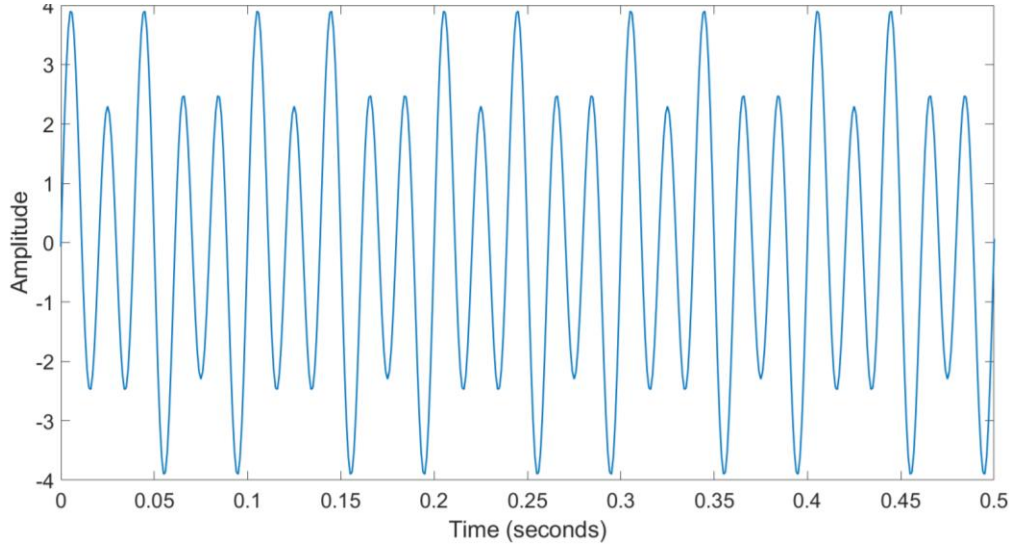

Intensity
Time
Fourier
transform
Frequency
remove
frequency
Frequency
Inverse Fourier
transform
Intensity
Time

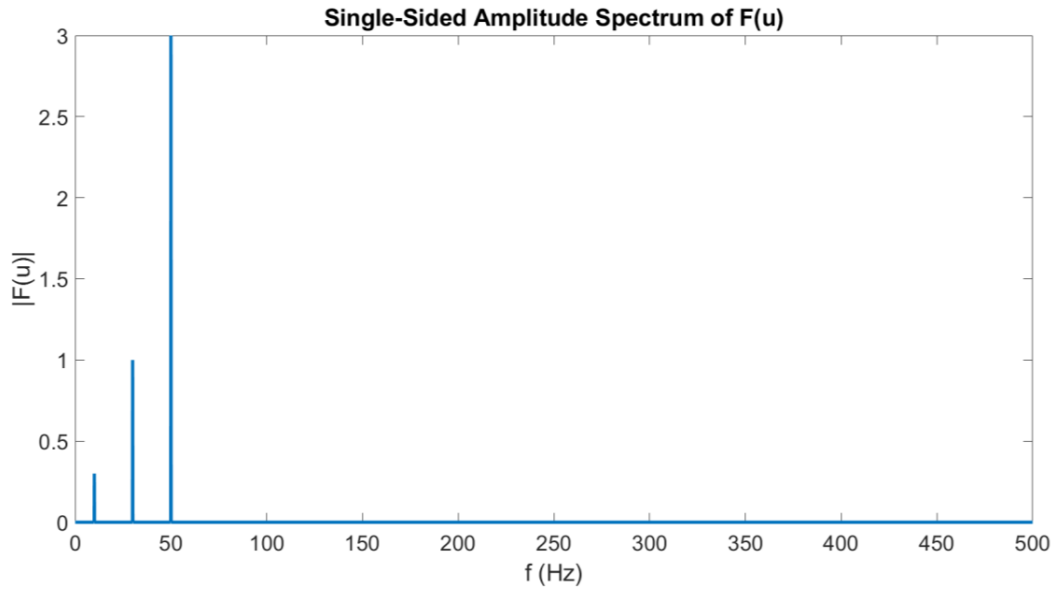
Tóm lại

- Biến đổi Fourier: miền thời gian/không gian f(x) sang miền tần số F(u)

$f(x) \overset{DFT}{\Longrightarrow} F(u)$ với u là tần số (đơn vị Hz)

- Xem quan hệ giữa u và biên độ $|F(u)|$ để phân tích thành phần tần số nào sẽ là thành phần chính của tín hiệu





Biến đổi Fourier của một ảnh

- Biến đổi Fourier rời rạc (DFT): một ảnh trong miền không gian sẽ được chuyển sang miền tần số
- DFT của ảnh là 2 chiều, chứa các số phức
- Công thức biến đổi Fourier rời rạc 2 chiều

$$G(u,v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} g(x,y)\, e^{-j\frac{2\pi(ux)}{M}} e^{-j\frac{2\pi(vy)}{N}}$$

Công thức

- Nếu G(u,v) là ảnh Fourier của một ảnh g(x,y) có M dòng, N cột:

$$G = U^T \times g \times V \qquad (2)$$

Với $U(x,u) = e^{-\frac{j2\pi xu}{M}}; x,u = 0:M-1$

$\mathrm{V}(y,v) = e^{-\frac{j2\pi yv}{N}}; y,v = 0:N-1$

Nếu U đối xứng thì (2) sẽ được viết lại thành

$$G = U \times g \times V$$

*Có thể tính trực tiếp từ công thức biến đổi Fourier rời rạc 2 chiều

Ví dụ

- Biến đổi Fourier của ảnh sau

$$g(x,y) = \begin{bmatrix} 1 & 0 \\ 1 & 0 \end{bmatrix}$$

U =

V =

$U^T * g * V =$

Với $U(x,u) = e^{-\frac{j2\pi xu}{M}}; x,u = 0:M-1$

$V(y,v) = e^{-\frac{j2\pi yv}{N}}; y,v = 0:N-1$

* Chú ý kích thước M và N của ma trận gốc g(x,y)
* Dùng hàm fft2 trong Python để kiểm tra

- Tính ma trận biên độ và ma trận phổ dựa vào ma trận số phức tìm được

Ví dụ

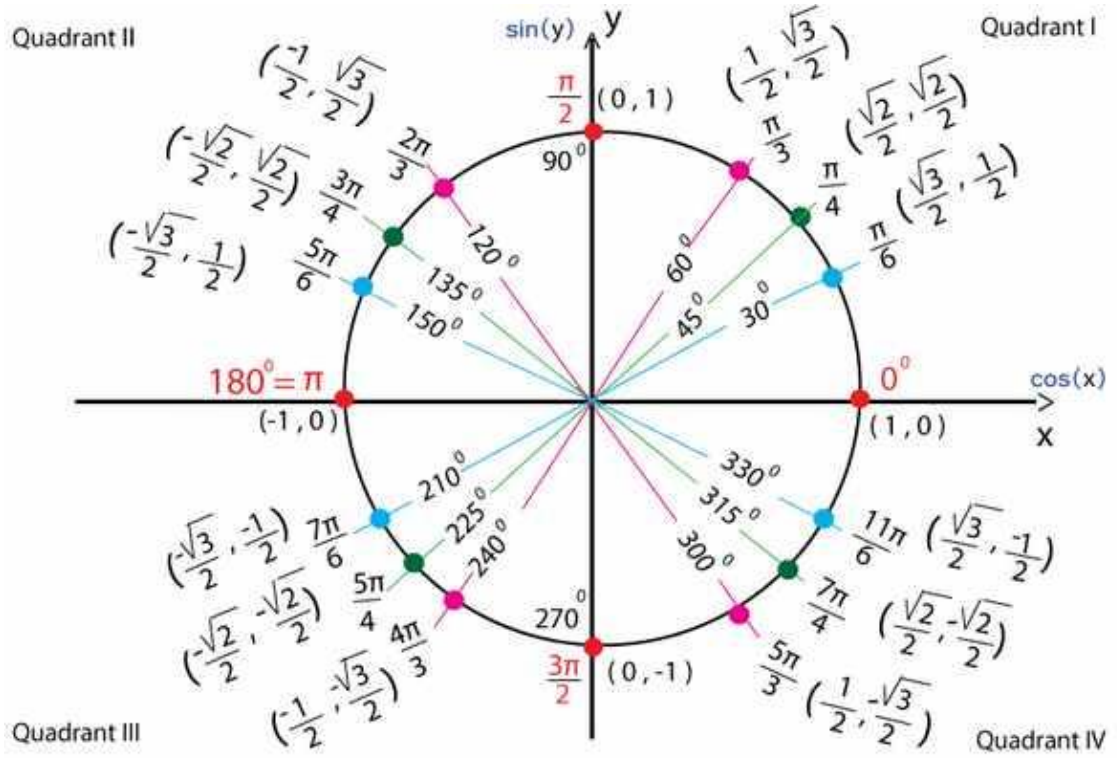
❑ Biến đổi Fourier của ảnh sau

$$g(x,y) = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \end{bmatrix}$$

U =

V =

$U^T * g * V =$



Với $U(x,u) = e^{-\frac{j2\pi xu}{M}}; x,u = 0:M-1$

$V(y,v) = e^{-\frac{j2\pi yv}{N}}; y,v = 0:N-1$

* Dùng hàm fft2 trong python để kiểm tra

Bài tập

- Biến đổi Fourier của ảnh sau

$$g(x,y) = \begin{array}{|c|c|}\hline 1 & 1 \\ \hline 0 & 0 \\ \hline 1 & 0 \\ \hline \end{array}$$

Biến đổi Fourier trong ảnh

Bước 1. Biến đổi DFT ảnh 2 chiều số thực g(x,y) → G(u,v): dùng hàm fft2() trong python.

G_uv = np.fft.fft2(I)

Bước 2. Tính ma trận biên độ của số phức của ma trận

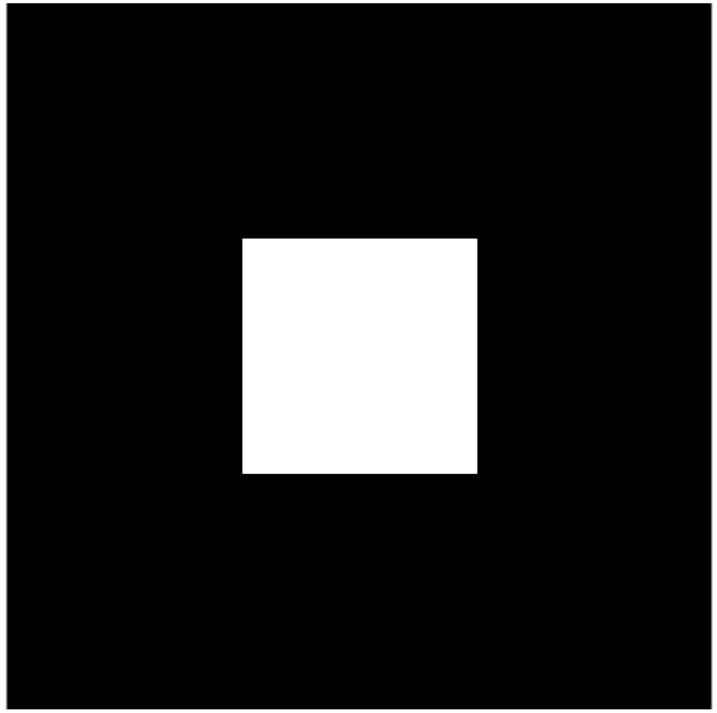
$$\mathrm{Mag}(u,v) = \sqrt{R(u,v)^2 + I(u,v)^2}$$

dùng hàm abs(F) trong python để tính biên độ

magnitude = abs(G_uv)

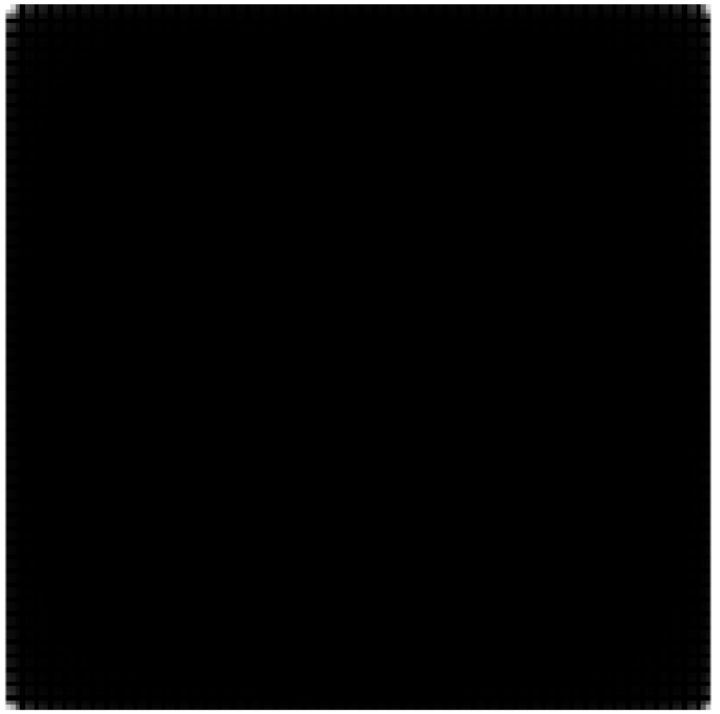
Ảnh gốc và Biên độ của ảnh Fourier

Image



Ảnh trắng đen, hình chữ nhật trắng nền đen.

Amplitude of Fourier Image (not center)



Góc trái trên cùng là F(0,0),
thành phần DC của ảnh Fourier

Biến đổi Fourier trong ảnh

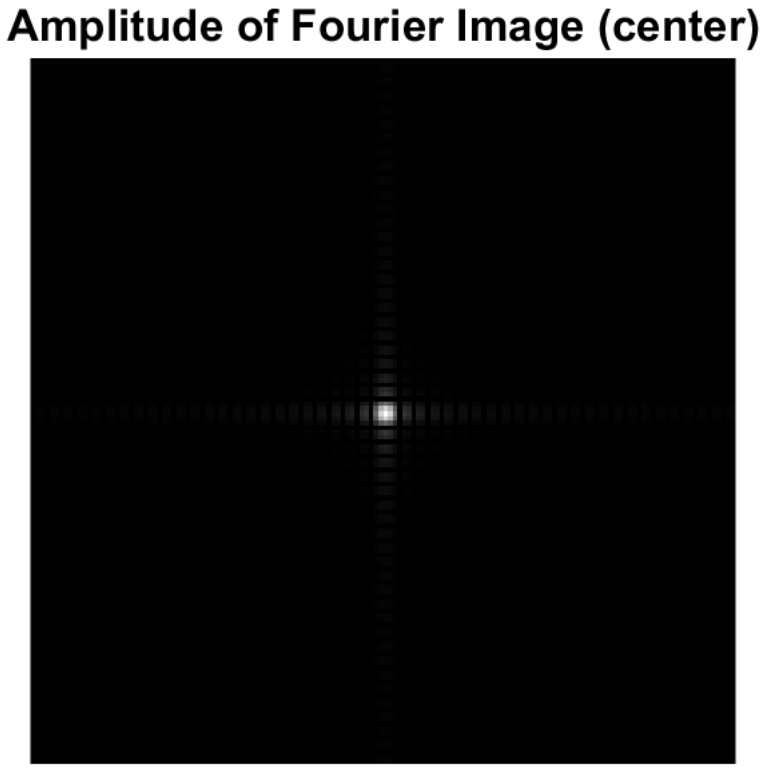
Bước 3. Dời trục tọa độ của biên độ, đem điểm F(0,0) vào chính giữa bức ảnh. Dùng hàm fftshift() để dời trục tọa độ

magnitude_shifted = fftshift(magnitude)

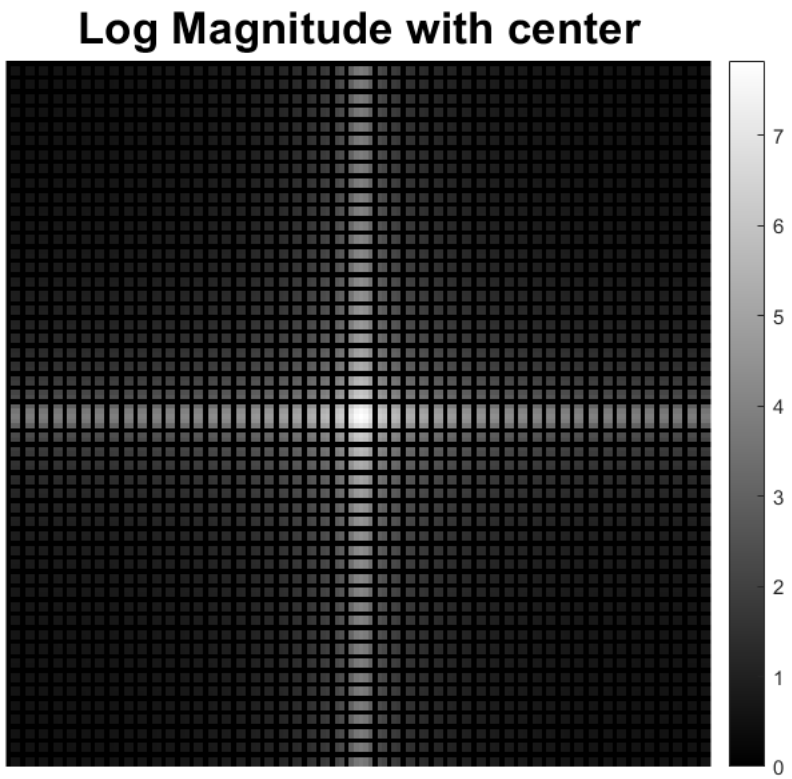
Bước 4. Dùng hàm log1p() để biểu diễn biên độ

log_magnitude = log1p(magnitude_shifted)

Biến đổi Fourier trong ảnh

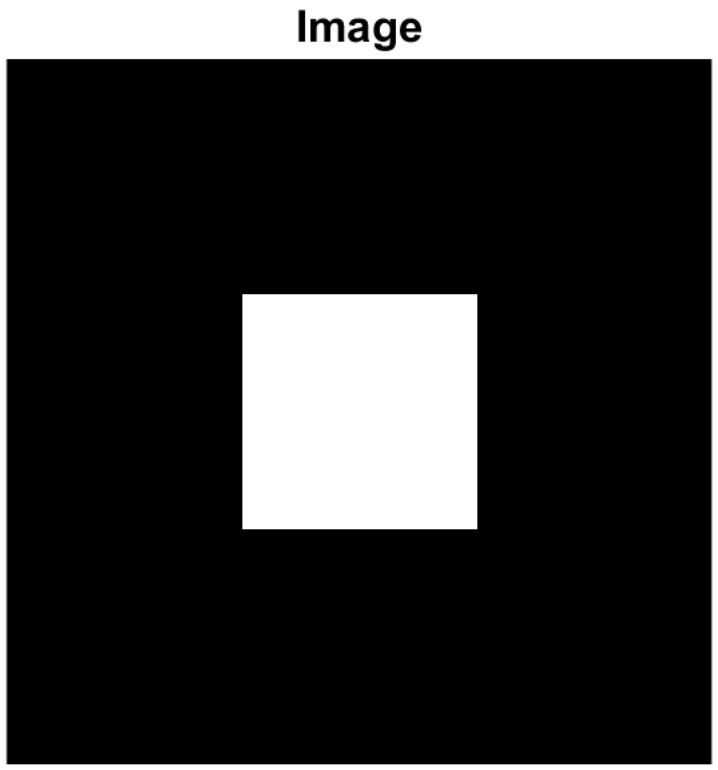


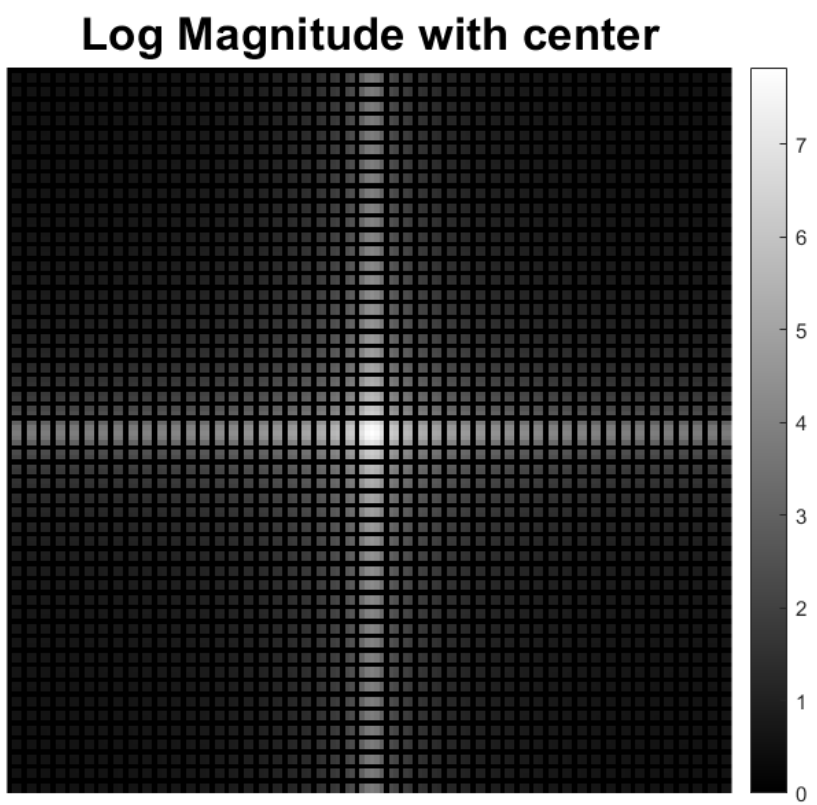
F(0,0) – thành phần DC đã được dời vào giữa. Xung quanh F(0,0) là các thành phần tần số thấp

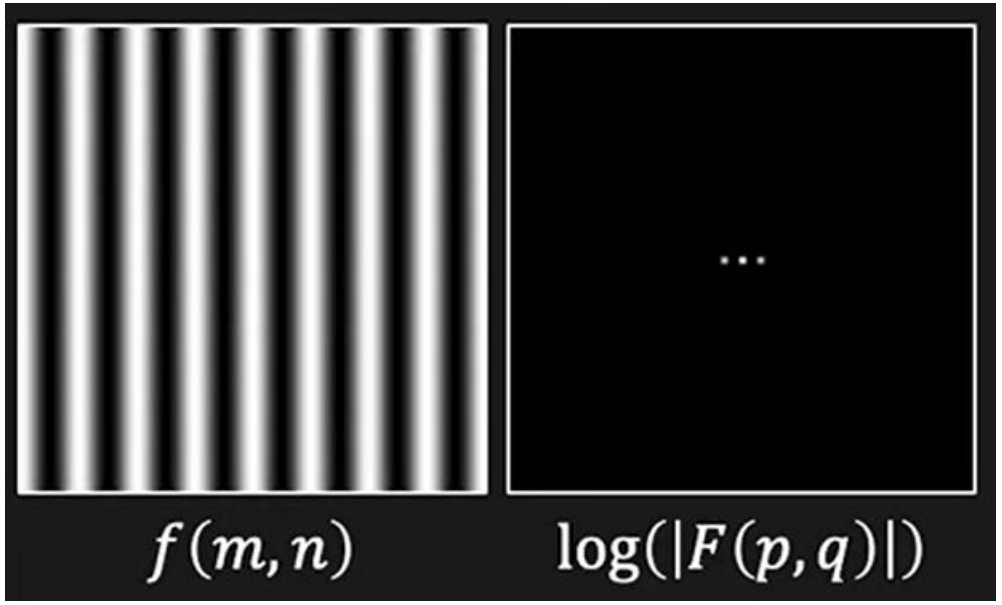


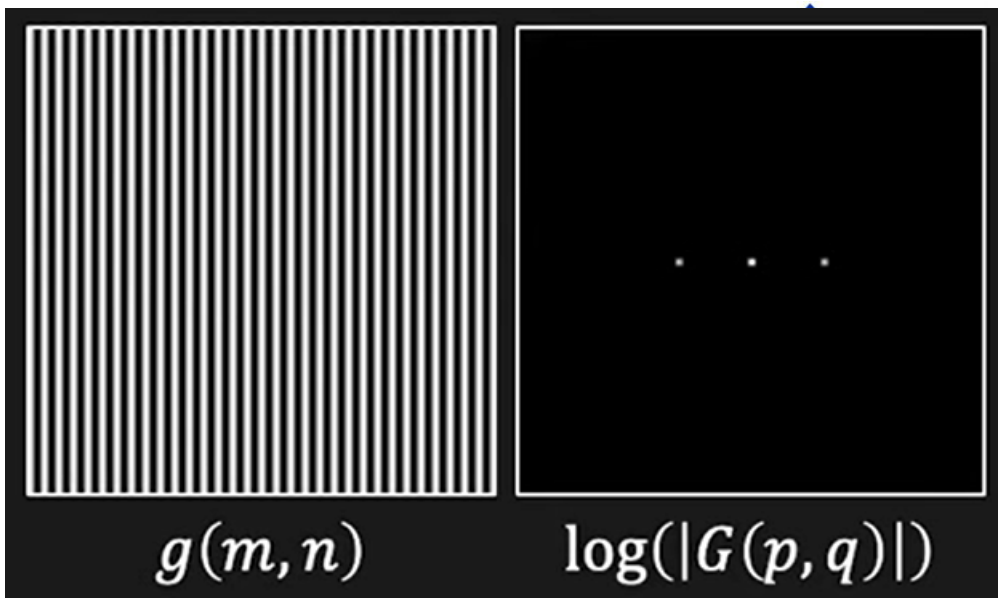
Biên độ sau khi được áp hàm log F(0,0) nằm giữa, các thành phần phía xa F(0,0) là tần số cao

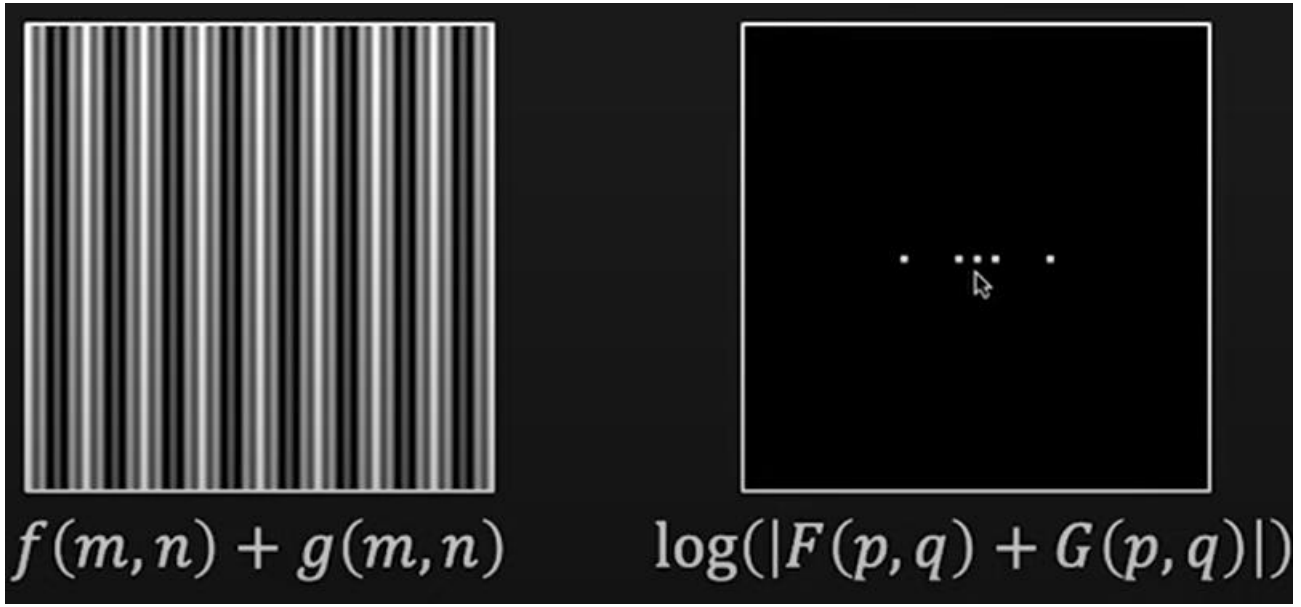
Kết quả



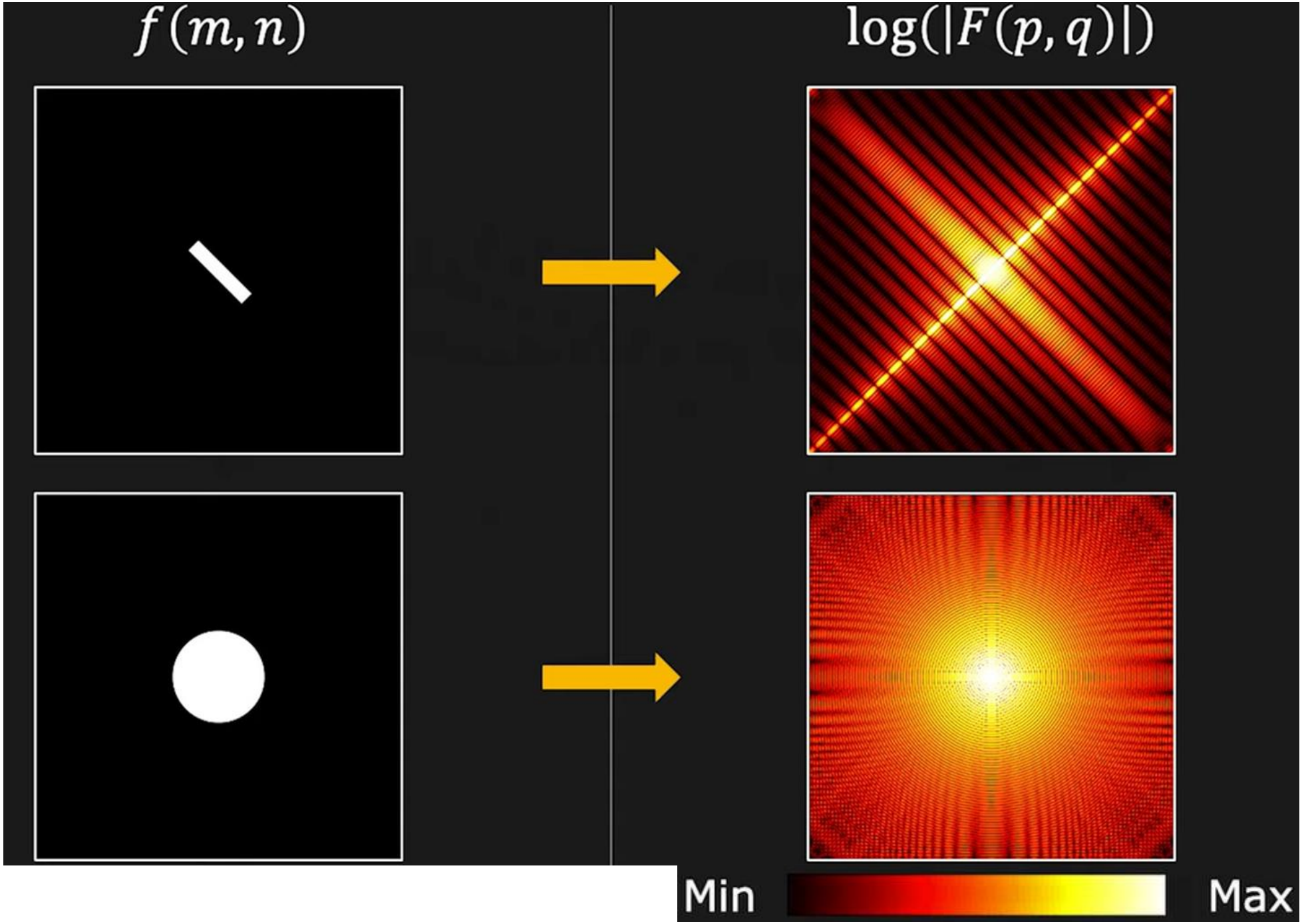




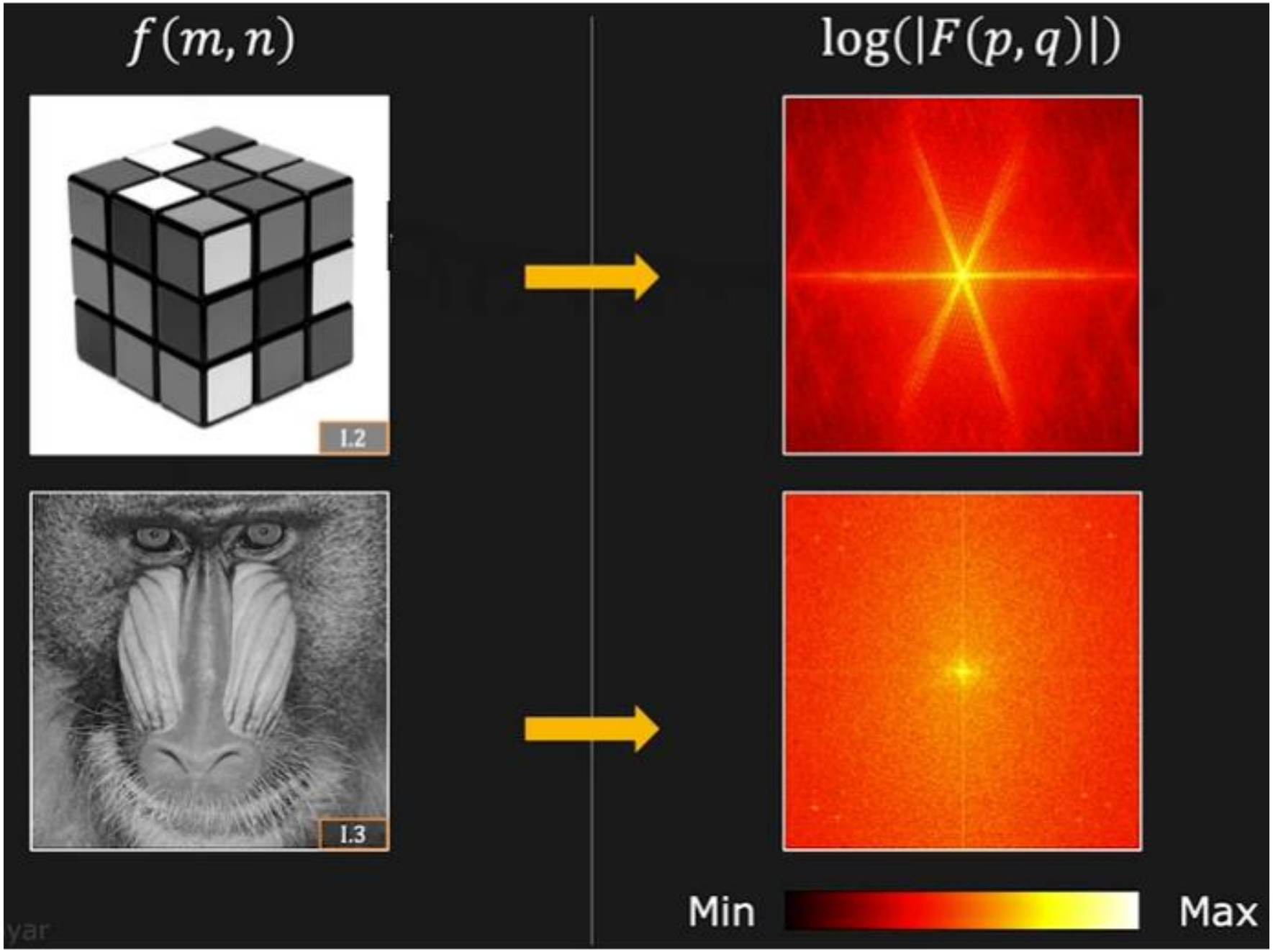




Shree K. Nayar, 2020



Shree K. Nayar, 2020



Shree K. Nayar, 2020

Ý nghĩa

- **Xác định chi tiết và kết cấu**: Nếu phổ có nhiều thành phần tần số cao, có nghĩa là ảnh có nhiều chi tiết và đường nét sắc bén. Ngược lại, nếu phần tần số cao yếu, ảnh có thể mờ và ít chi tiết
- **Phát hiện nhiễu**: Nếu ảnh chứa nhiều nhiễu, thành phần tần số cao sẽ rất nổi bật và mạnh mẽ. Việc quan sát phổ giúp xác định và có thể lọc bỏ nhiễu từ ảnh
- **Lọc ảnh**: Trong lọc ảnh, có thể giữ lại hoặc loại bỏ các tần số nhất định để làm sắc nét, làm mờ, hoặc loại bỏ nhiễu
- **Nén ảnh**: Phân tích phổ giúp trong các ứng dụng nén ảnh bằng cách loại bỏ các tần số không quan trọng

Code

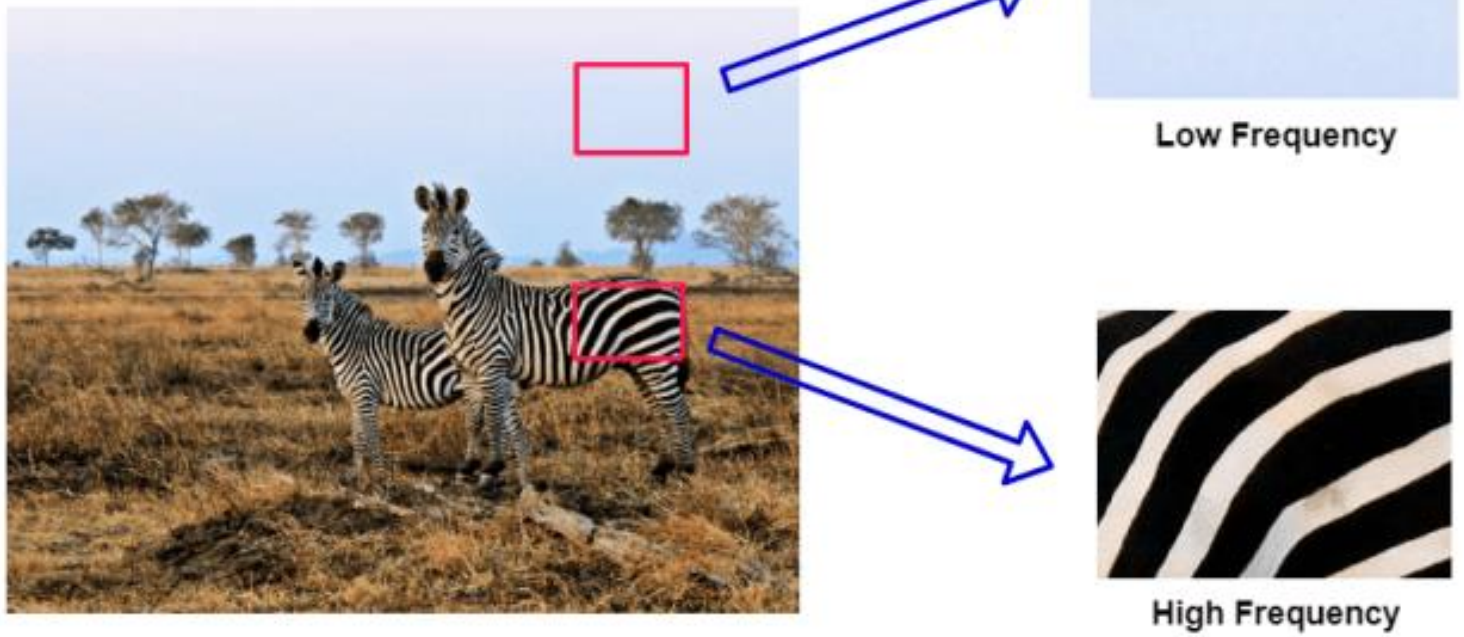
```python
import numpy as np
import matplotlib.pyplot as plt
from numpy.fft import fft2, fftshift
from PIL import Image

# Đọc ảnh
image_path = 'square0-1.png'
image = Image.open(image_path).convert('L')
image_array = np.array(image)

# Step 1: Biến đổi Fourier rời rạc bằng fft2
G_uv = fft2(image_array)

# Step 2: Tính ma trận độ lớn của số phức
magnitude = np.abs(G_uv)

# Step 3: Chuyển trục tọa độ 0,0 vào giữa
magnitude_shifted = fftshift(magnitude)

# Step 4: Dùng hàm logarit để biểu diễn biên độ
log_magnitude = np.log1p(magnitude_shifted)

# Hiển thị hình ảnh gốc và phổ theo thang logarit
plt.figure(figsize=(12, 6))

# Ảnh gốc
plt.subplot(1, 2, 1)
plt.title('Image')
plt.imshow(image_array, cmap='gray')
plt.axis('off')

# Phổ ảnh theo thang logarit
plt.subplot(1, 2, 2)
plt.title('Log Magnitude Spectrum with center')
plt.imshow(log_magnitude, cmap='hot')
plt.axis('off')

plt.show()
```

Thực hành Python

- Phân tích phổ Fourier của các ảnh sau

Ý nghĩa thành phần tần số của ảnh

- **Tần số**: tỉ lệ thay đổi cường độ (intensity)
- Thành phần tần số thấp: những vùng ít có sự biến động về cường độ như background
- Thành phần tần số cao: những vùng có sự biến động mạnh về cường độ như biên



Ví dụ

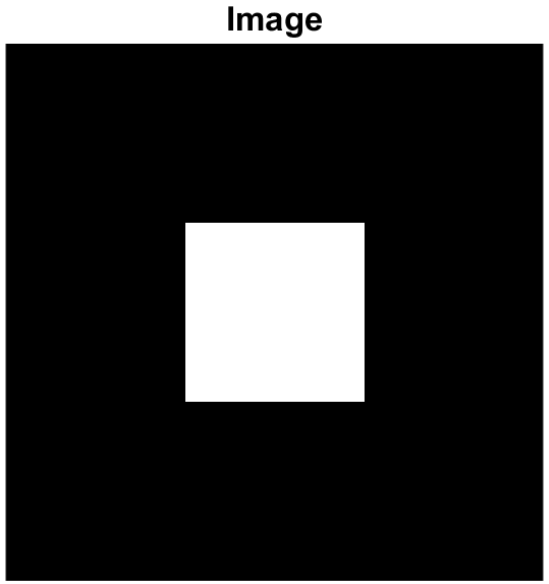
- Low_fre vùng nào?
- High_fre vùng nào?

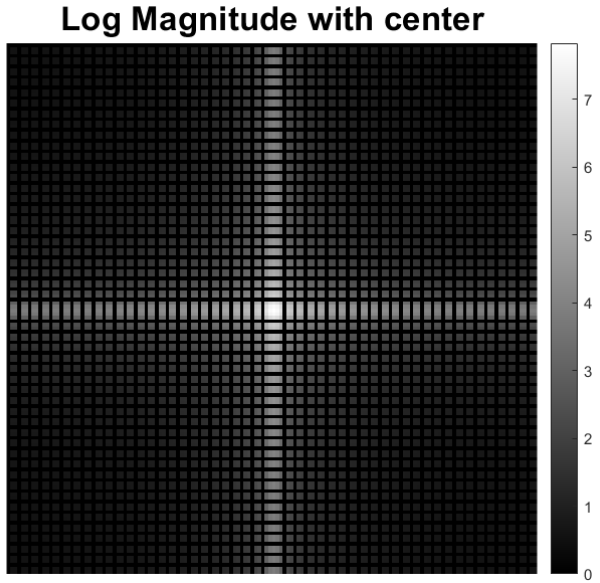
Tóm lại

- Biến đổi Fourier: đưa ảnh từ ma trận pixel (số thực) sang ma trận miền tần số (số phức)

$f(x,y) \overset{DFT}{\Longrightarrow} F(u,v)$ với u,v là tần số

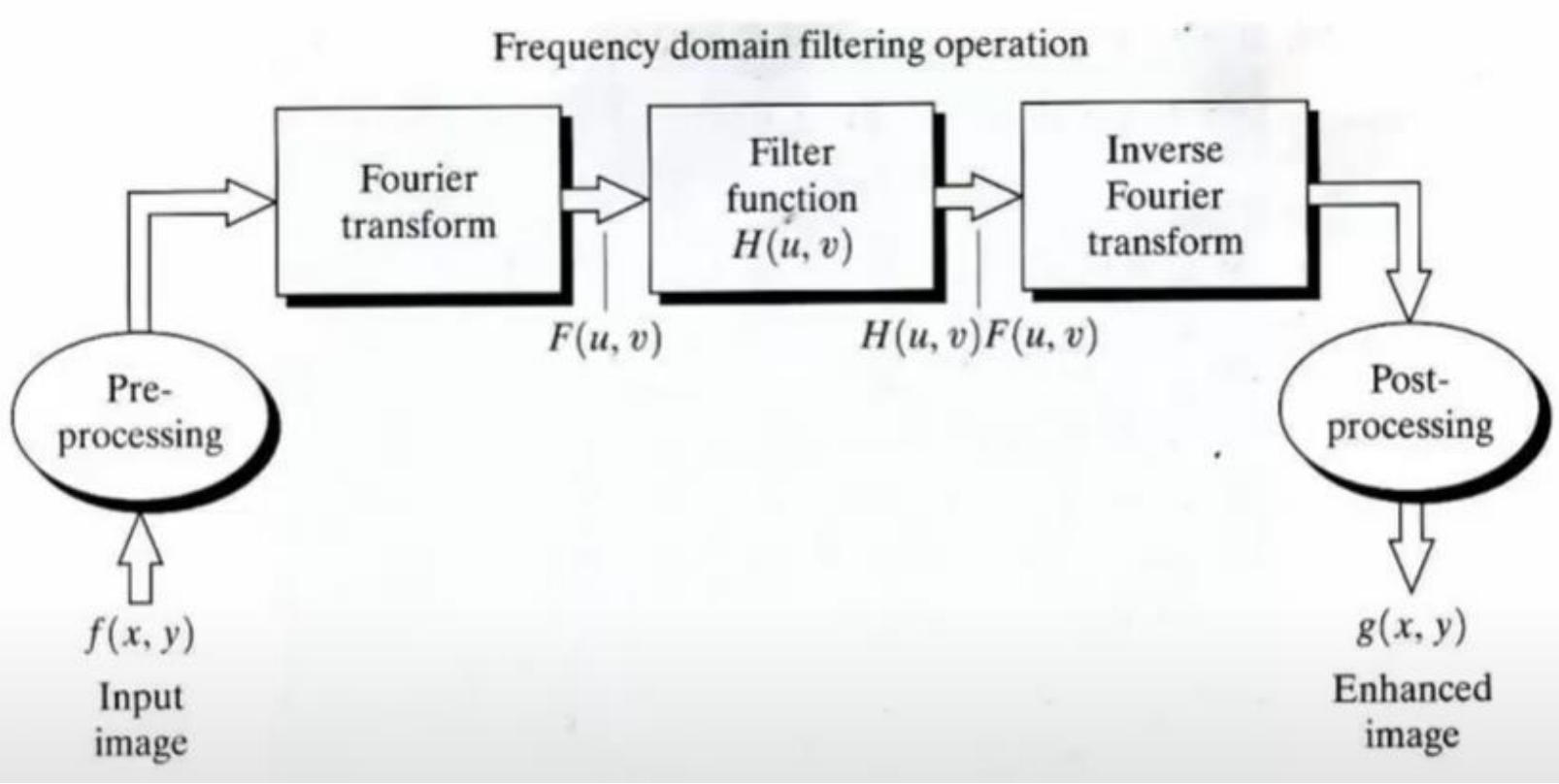
- Xem quan hệ giữa cặp tần số (u,v) và biên độ $|F(u,v)|$ để hiểu các thành phần tần số cao và thấp, từ đó dùng bộ lọc phù hợp





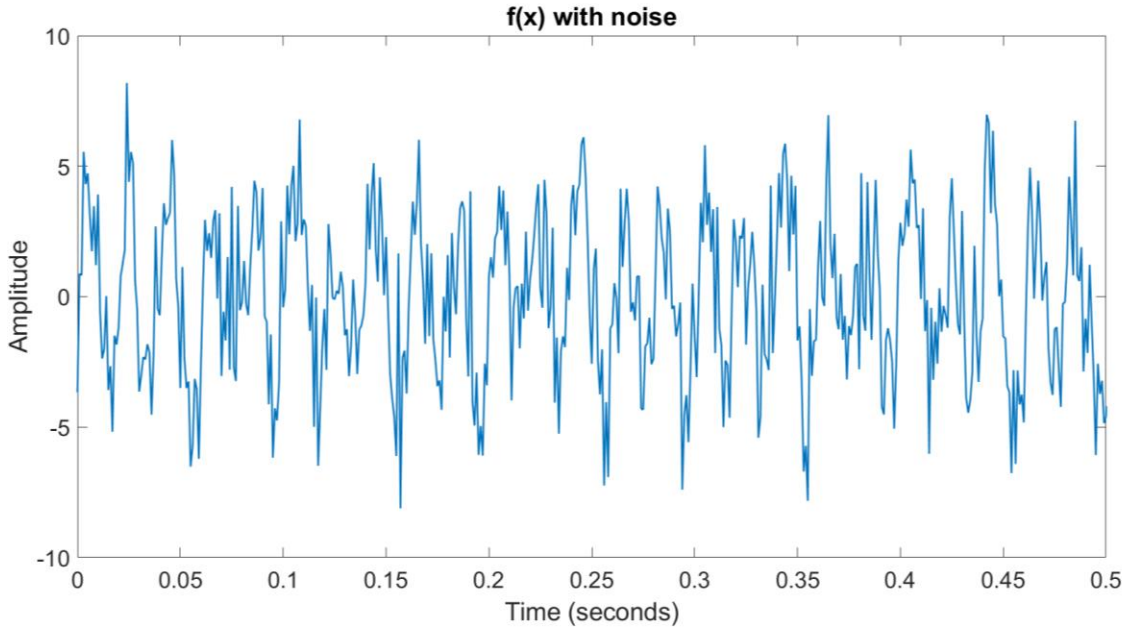
Ứng dụng DFT trong xử lý ảnh

- DFT trong xử lý ảnh là DFT 2 chiều

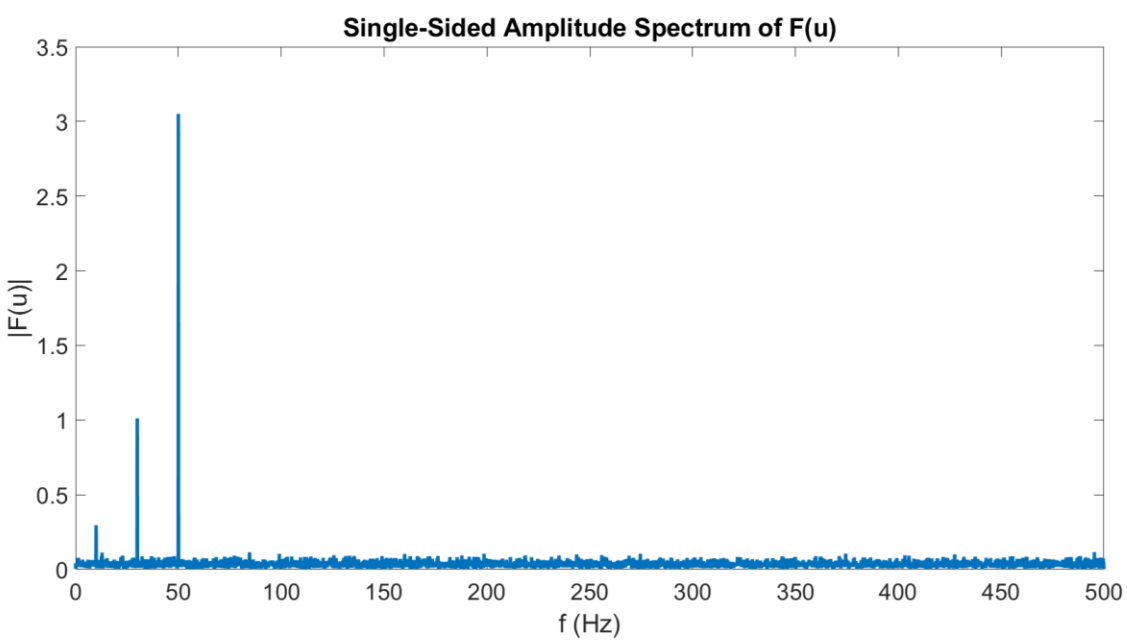


Ứng dụng:

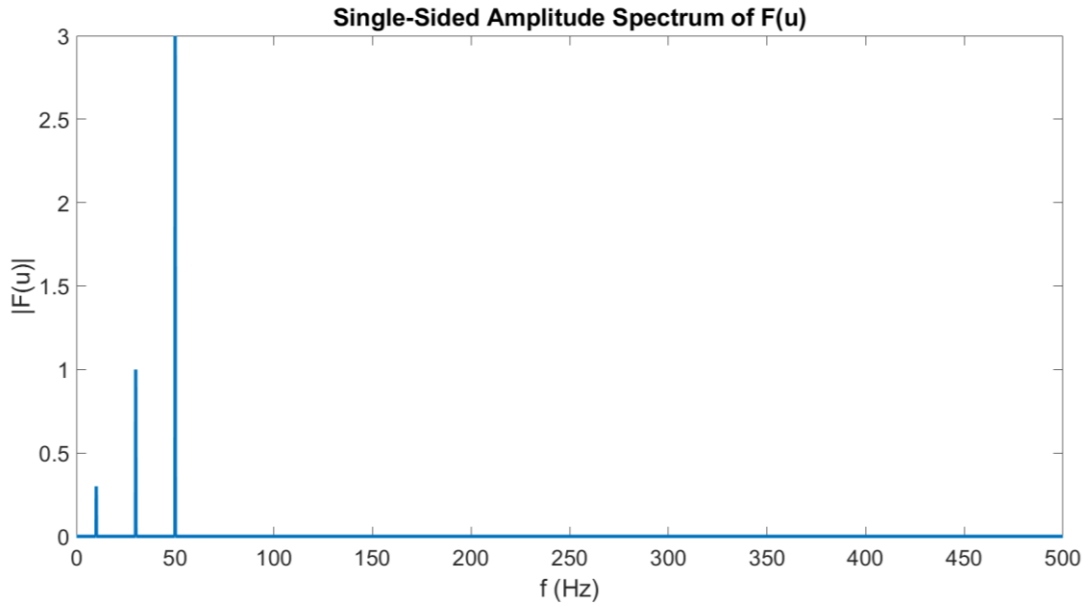
- Rút trích đặt trưng trong miền tần số
- Lọc nhiễu
- Xác định tần số của một bức ảnh



DFT

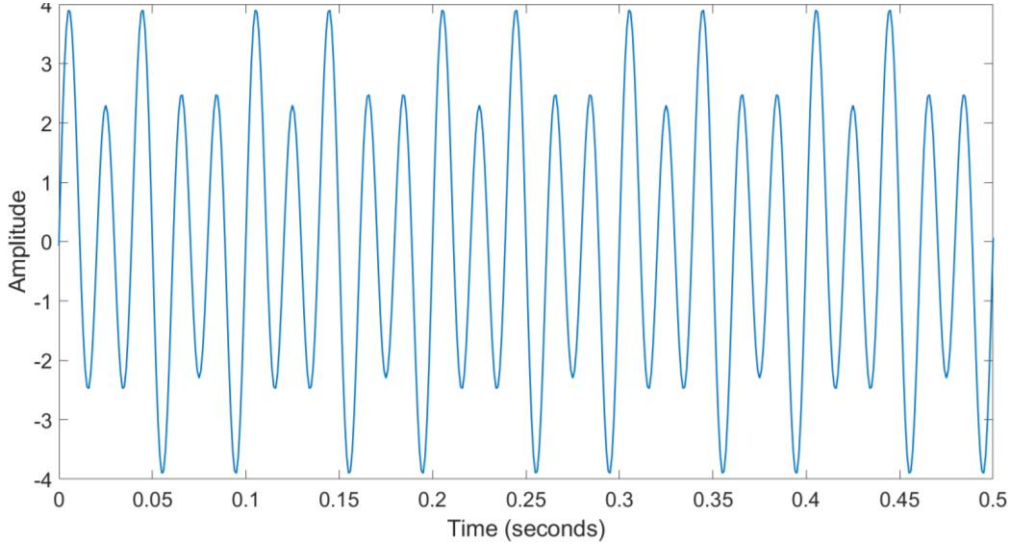


Lọc



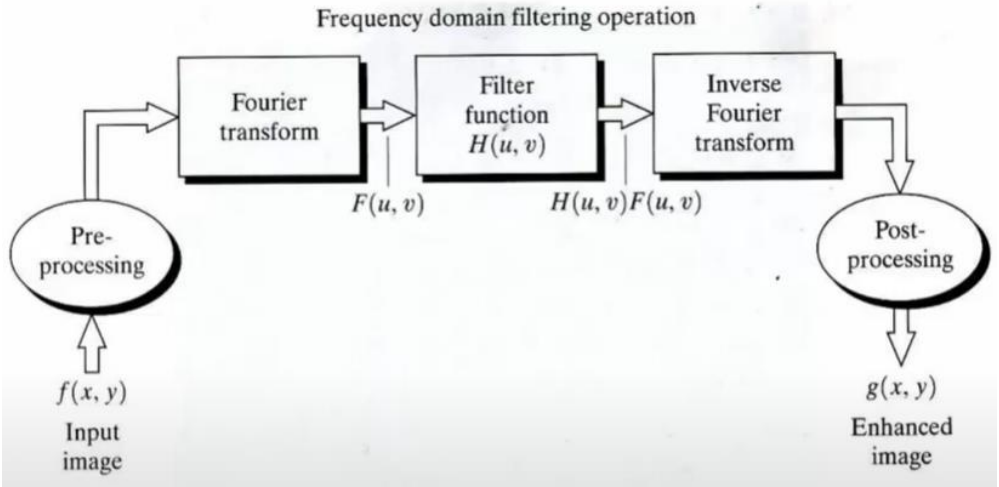
DFT ngược

Tín hiệu được cải thiện

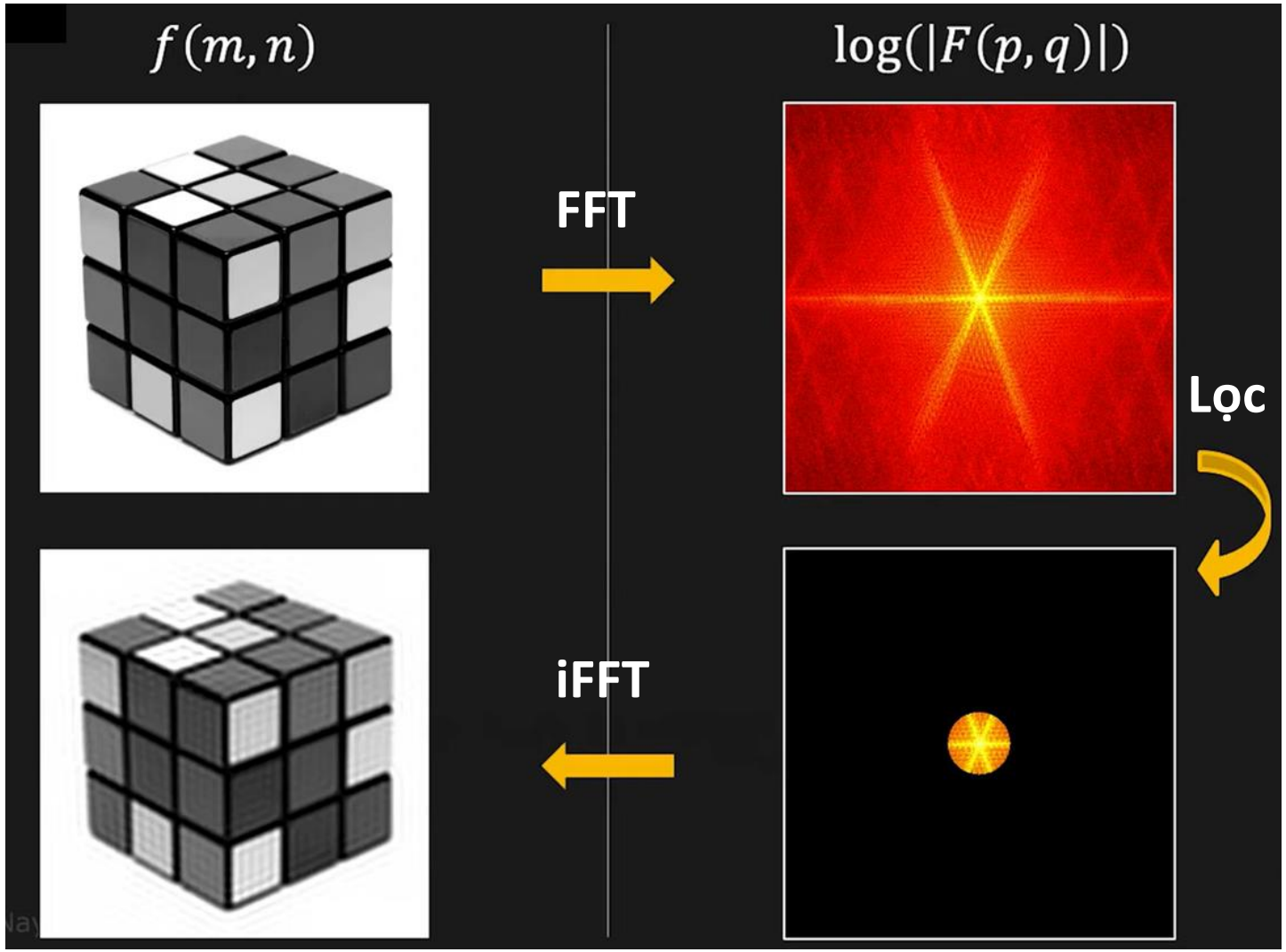


Ứng dụng các bộ lọc (1)

- Lọc thông thấp (Low-pass filter)
  - Ảnh mờ hơn

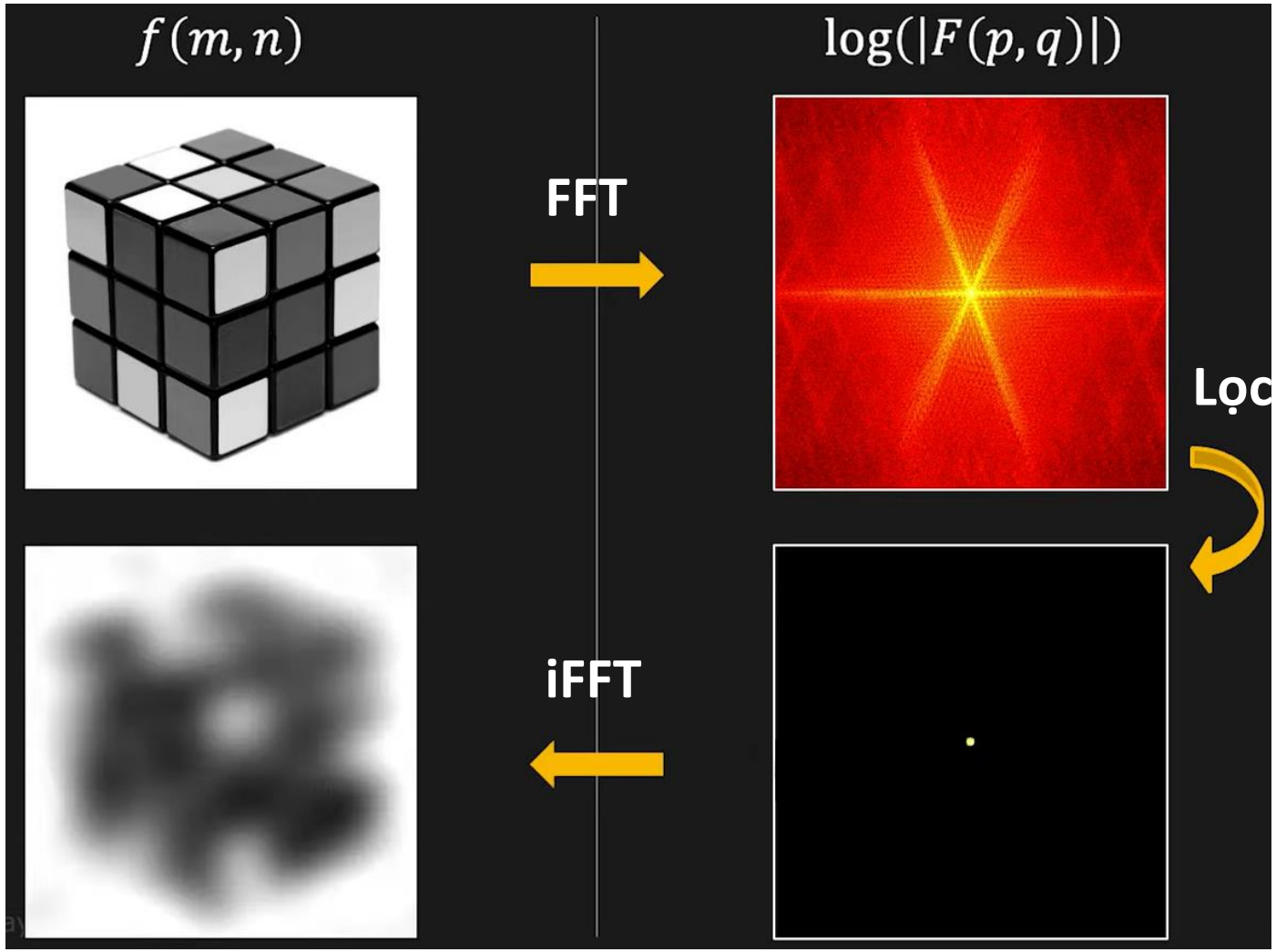


Shree K. Nayar, 2020



Ứng dụng các bộ lọc (1)

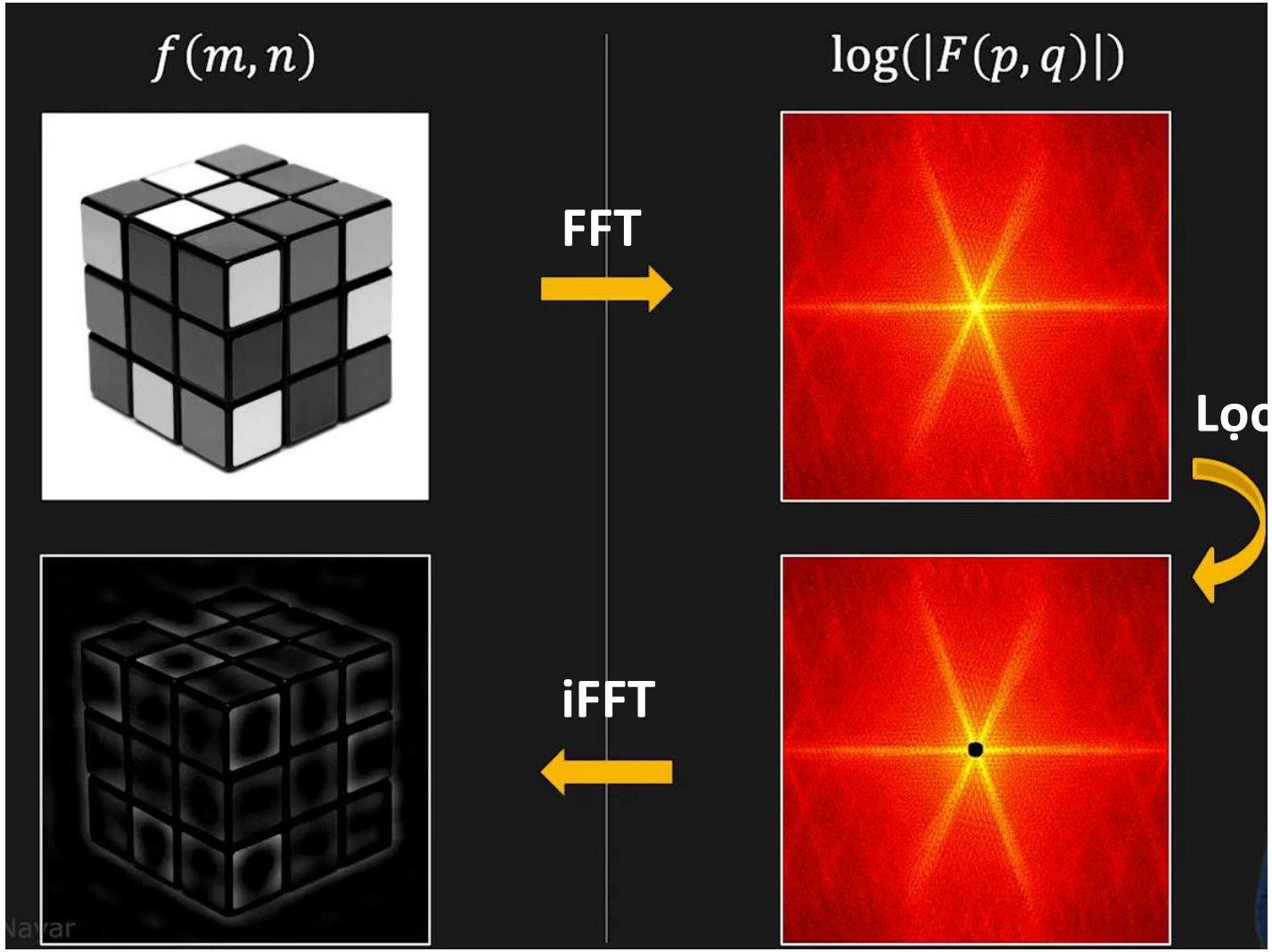
- Lọc thông thấp
  - Tần số cắt càng nhỏ làm ảnh càng mờ



Shree K. Nayar, 2020

Ứng dụng các bộ lọc (1)

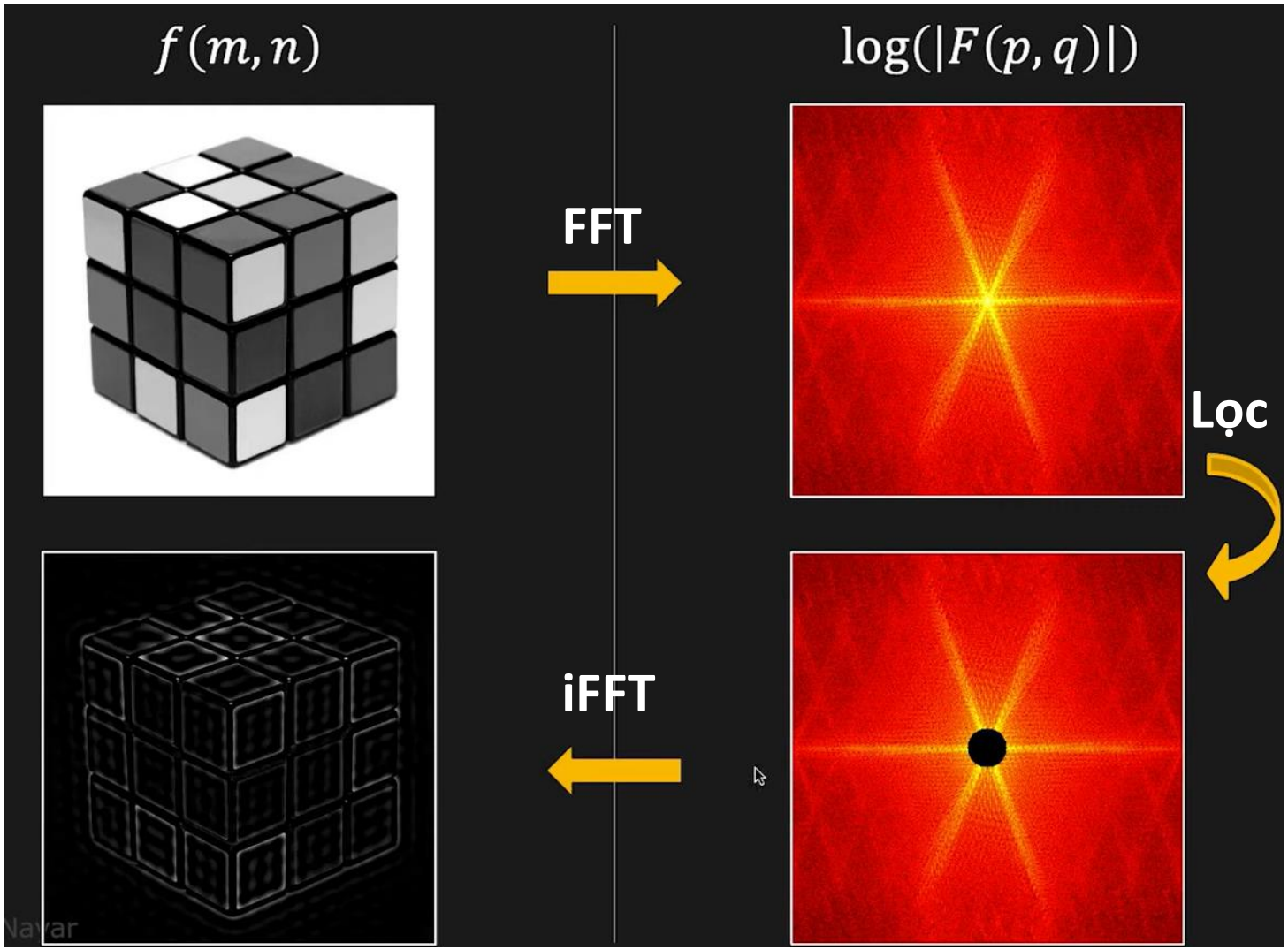
- Lọc thông cao (High-pass filter):
  - Loại bỏ các vùng có độ sáng giống nhau (các mặt khối rubik) – tức vùng có tần số thấp
  - Giữ lại các vùng có sự biến đổi nhanh (các rìa)



Shree K. Nayar, 2020

Ứng dụng các bộ lọc (1)

- Lọc thông cao:
  - Khi tần số cắt càng lớn độ sắc nét tại các vùng rìa càng rõ

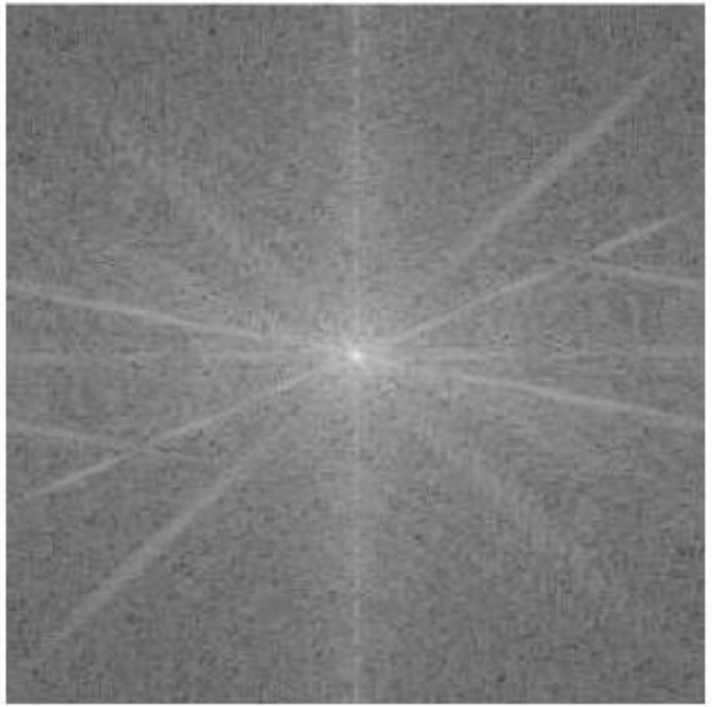


Shree K. Nayar, 2020

Ứng dụng trong các bộ lọc (2)

- Giả sử ảnh cameraman trong miền tần số sau khi DFT

Image



DFT

Lọc thông thấp (Low-pass filter)

Applying
low pass filter
to DFT
Cutoff D = 15

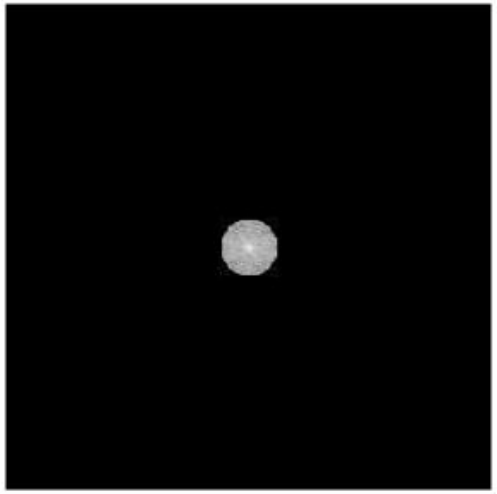
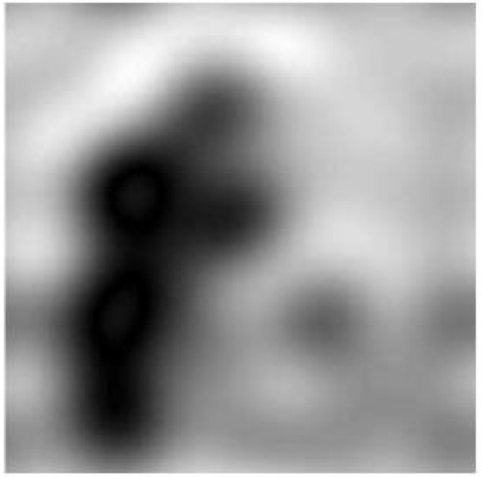


Image after
inversion

low pass filter
Cutoff D = 5

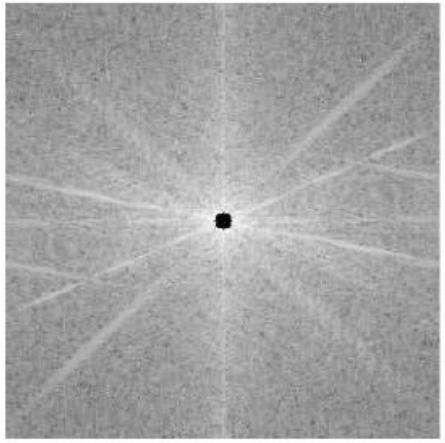


low pass filter
Cutoff D = 30

Note: Sharp filter
Cutoff causes
ringing

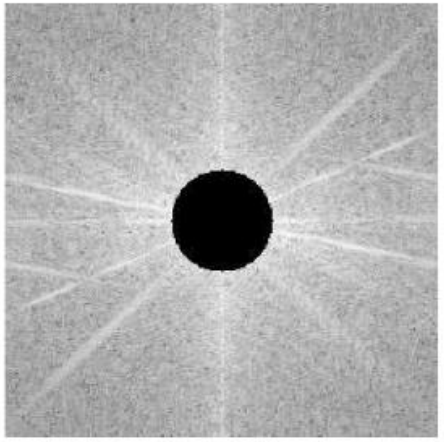
Lọc thông cao (High-pass filter)

High pass filtering of DFT with filter Cutoff D = 5



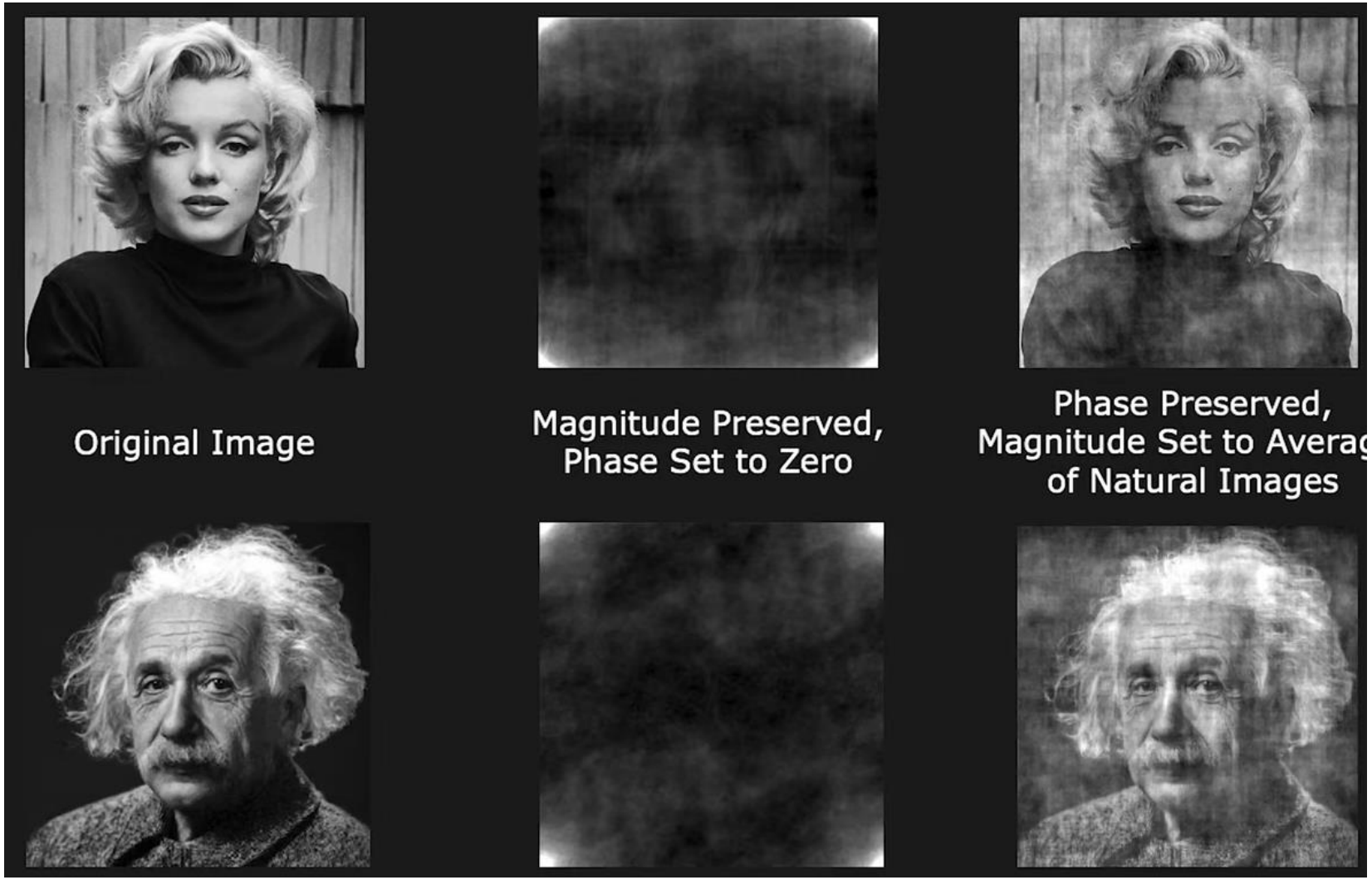
Low cutoff frequency removes Only few lowest frequencies

High pass filtering of DFT with filter Cutoff D = 30



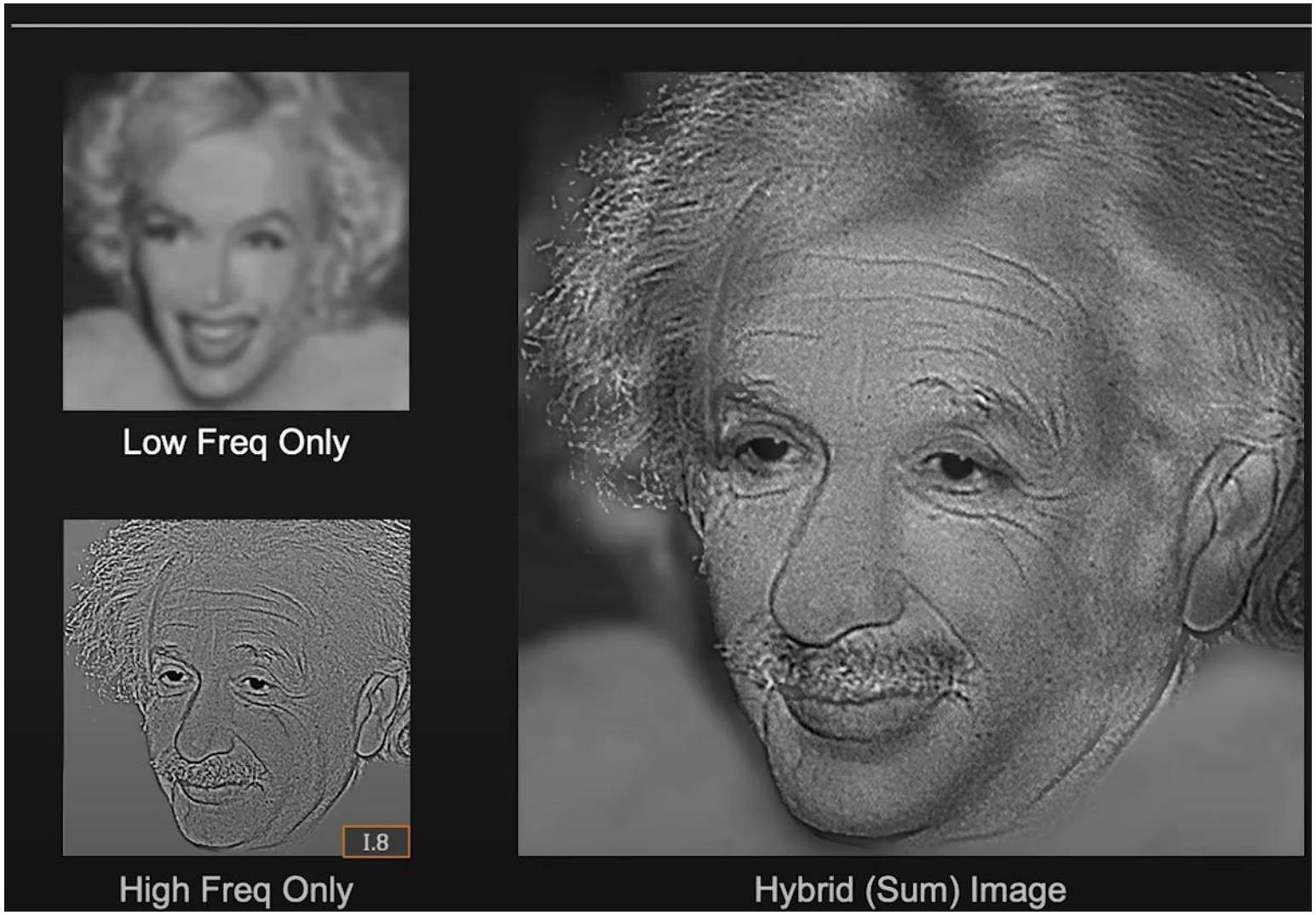
High cutoff frequency removes many frequencies, leaving only edges

Biên độ và pha



Shree K. Nayar, 2020

Ảnh hydrid



Oliva, 2006

2.3 Biến đổi DCT

- Discrete Cosine Transform
- Công thức biến đổi 2D-DCT, biến đổi DCT cơ bản
- Matlab code cho hàm 2D-DCT cơ bản
- Cách tính biến đổi DCT sử dụng FFT và sử dụng ma trận chuyển
- Nén ảnh sử dụng DCT

Biến đổi DCT (Discrete Cosine Transform)

- Tương tự như biến đổi Fourier, DCT biến đổi hình ảnh từ miền không gian về miền tần số. Kết quả là số thực (Fourier là số phức)

- Được sử dụng trong nén ảnh, hầu hết thông tin quan trọng được tập trung ở một số lượng nhỏ hệ số của DCT → đặc tính "gói năng lượng" (Energy Compaction)

- DCT được sử dụng trong nén JPEG

Công thức biến đổi DCT

Biến đổi DCT 1 chiều: $f(x) \overset{DCT}{\Longrightarrow} F(u)$

Biến đổi DCT 2 chiều: $f(x,y) \overset{DCT}{\Longrightarrow} F(u,v)$

Công thức

$$\boldsymbol{F(p,q) = \alpha_p \alpha_q \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) cos\left[\frac{\pi(2x+1)p}{2M}\right] cos\left[\frac{\pi(2x+1)q}{2M}\right]}$$

Trong đó:

F(p,q) : Hệ số DCT tại vị trí (*p,q*) trong miền tần số.

$f(x,y)$: Giá trị độ (tín hiệu/điểm ảnh) tại (x,y) trong miền không gian.

M,N : Kích thước của tín hiệu/ảnh.

$\alpha(p)$ và $\alpha(q)$: là các hệ số điều chỉnh

Công thức DCT 2 chiều

$$\boldsymbol{F(u,v) = \alpha_p \alpha_q \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) cos \left[\frac{\pi(2x+1)u}{2M} \right] cos \left[\frac{\pi(2x+1)v}{2M} \right]}$$

$C(u,v)$

Với

$$\alpha_p = \begin{cases} \frac{1}{\sqrt{M}}, & p = 0 \\ \sqrt{\frac{2}{M}}, & 1 \le p \le M-1 \end{cases}$$

$$\alpha_q = \begin{cases} \frac{1}{\sqrt{N}}, & q = 0 \\ \sqrt{\frac{2}{N}}, & 1 \le q \le N-1 \end{cases}$$

Ma trận biến đổi cosine NxN

- Thông thường biến đổi DCT sẽ được chia ra thành các block như 4x4, 8x8, 16x16

$$C(u,v) = \begin{cases} \sqrt{\frac{1}{N}} & u=0, \\ \sqrt{\frac{2}{N}} \cos\left[\frac{\pi(2v+1)u}{2N}\right] & 1 \le u \le N-1 \\ & 0 \le v \le N-1 \end{cases}$$

$$C_4 = \begin{bmatrix} 0.5 & 0.5 & 0.5 & 0.5 \\ 0.653 & 0.2705 & -0.2705 & -0.653 \\ 0.5 & -0.5 & -0.5 & 0.5 \\ 0.2705 & -0.653 & 0.653 & -0.2705 \end{bmatrix}$$

2 –D DCT of an image can be generated using equation $F = CfC^T$

Ví dụ

- Tìm ma trận sau khi biến đổi DCT của ảnh sau

2	4	4	2
4	6	8	3
2	8	10	4
3	8	6	2

2	4	4	2
4	6	8	3
2	8	10	4
3	8	6	2

$$F = CfC^T$$

$$F=\begin{bmatrix} 0.5 & 0.5 & 0.5 & 0.5 \\ 0.653 & 0.2705 & -0.2705 & -0.653 \\ 0.5 & -0.5 & -0.5 & 0.5 \\ 0.2705 & -0.653 & 0.653 & -0.2705 \end{bmatrix} \begin{bmatrix} 2 & 4 & 4 & 2 \\ 4 & 6 & 8 & 3 \\ 2 & 8 & 10 & 4 \\ 3 & 8 & 6 & 2 \end{bmatrix} \begin{bmatrix} 0.5 & 0.653 & 0.5 & 0.2705 \\ 0.5 & 0.2705 & -0.5 & -0.653 \\ 0.5 & -0.2705 & -0.5 & -0.653 \\ 0.5 & -0.653 & 0.5 & -0.2705 \end{bmatrix}$$

```
19.0000   -0.2706   -8.0000    0.6533
-2.6924   -0.2500    2.3097    0.8964
-3.5000    1.4651    1.5000   -1.6892
 0.0328   -1.6036   -0.9567   -0.2500
```

- Trong scipy dùng hàm DCT = dct()

Các thành phần tần số của DCT

- Thành phần DC ở góc trái trên cùng
- Thành phần có tần số thấp nằm xung quanh thành phần DC
- Thành phần có tần số cao nằm ở góc dưới bên phải

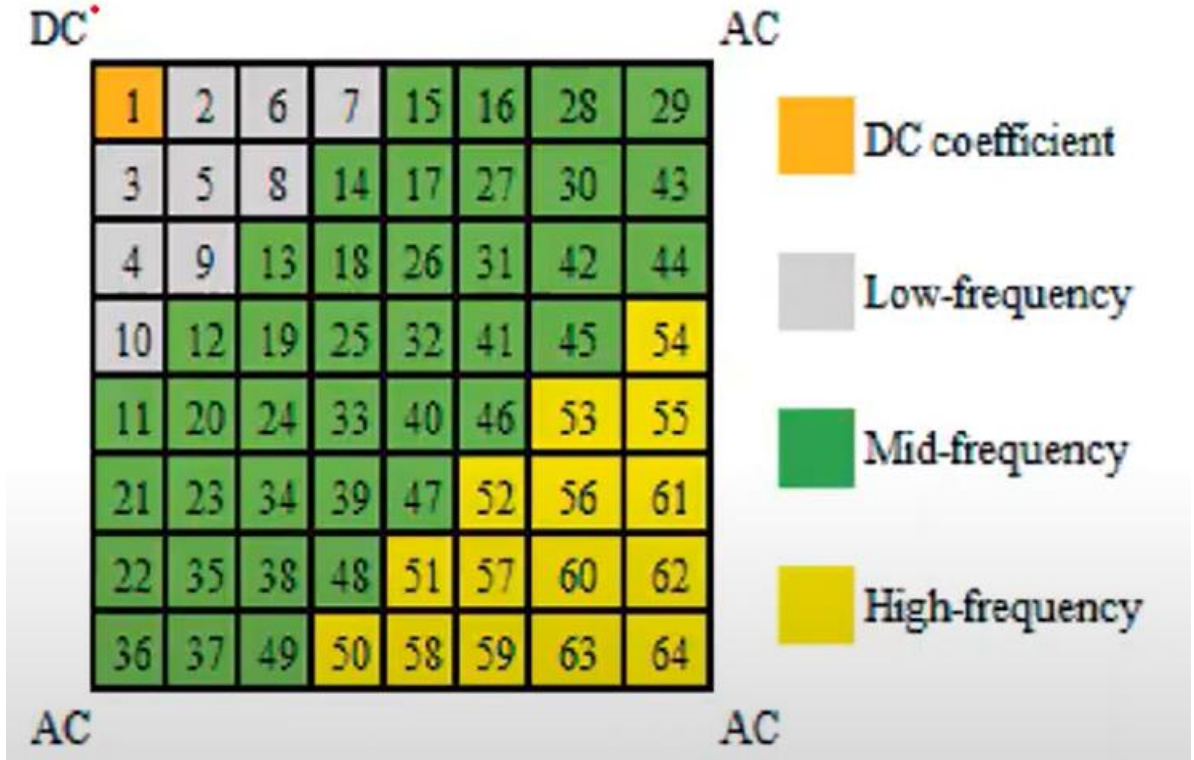
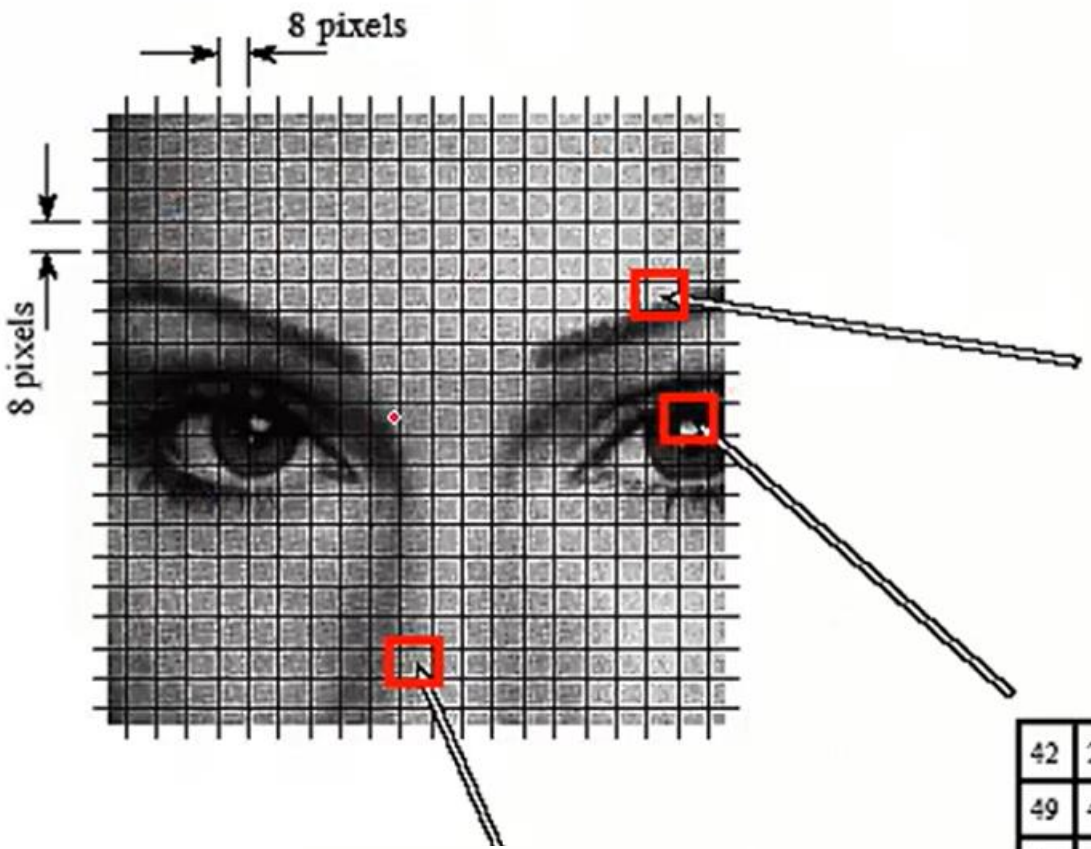


Figure 1. DCT coefficients.

Ứng dụng trong nén ảnh

- Đối với hầu hết các hình ảnh, phần lớn năng lượng tín hiệu nằm ở tần số thấp; chúng xuất hiện ở góc trên bên trái của DCT
- Nén được thực hiện do các giá trị bên phải thấp hơn biểu thị tần số cao hơn và thường nhỏ - đủ nhỏ để bị bỏ qua với ít biến dạng có thể nhìn thấy



231	224	224	217	217	203	189	196
210	217	203	189	203	224	217	224
196	217	210	224	203	203	196	189
210	203	196	203	182	203	182	189
203	224	203	217	196	175	154	140
182	189	168	161	154	126	119	112
175	154	126	105	140	105	119	84
154	98	105	98	105	63	112	84

d. Eyebrow spectrum

174	19	0	3	1	0	-3	1
52	-13	-3	-4	-4	-4	5	-8
-18	-4	8	3	3	2	0	9
5	12	-4	0	0	-5	-1	0
1	2	-2	-1	4	4	2	0
-1	2	1	3	0	0	1	1
-2	5	-5	-5	3	2	-1	-1
3	5	-7	0	0	0	-4	0

42	28	35	28	42	49	35	42
49	49	35	28	35	35	35	42
42	21	21	28	42	35	42	28
21	35	35	42	42	28	28	14
56	70	77	84	91	28	28	21
70	126	133	147	161	91	35	14
126	203	189	182	175	175	35	21
49	189	245	210	182	84	21	35

e. Eye spectrum

70	24	-28	-4	-2	-10	-1	0
-53	-35	43	13	7	13	1	3
23	9	-10	-8	-7	-6	5	-3
6	2	-2	8	2	-1	0	-1
-10	-2	-1	-12	2	1	-1	4
3	0	0	11	-4	-1	5	6
-3	-5	-5	-4	3	2	-3	5
3	0	4	5	1	2	1	0

f. Nose spectrum

174	-11	-2	-3	-3	6	-3	4
-2	-3	1	2	0	3	1	2
3	0	-4	0	0	0	-1	9
-4	-6	-2	1	-1	4	-10	-3
1	2	-2	0	0	-2	0	-5
3	-1	3	-2	2	1	1	0
3	5	2	-2	3	0	4	3
4	-3	-13	3	-4	3	-5	3

Ứng dụng trong nén ảnh JPEG

- Đầu vào DCT là một mảng 8-8 số nguyên
- Sau đó đầu ra của DCT sẽ được lượng tử hóa, mã hóa entrophy để tạo ảnh nén