Video Compression
視訊壓縮

Yu-Lun (Alex) Liu

劉育綸

with slides by Wen-Hsiao Peng,

Shao-Yi Chien,

Hsueh-Ming Hang,

and Aggelos K. Katsaggelos

Week	Date	Topic	Assignments
1	2025-09-01		
2	2025-09-08	Introduction to Image and Video Processing	
3	2025-09-16	Signals and Systems	#1 – Color Transform, due: 2025-09-29 1:19pm
4	2025-09-22	Fourier Transform and Sampling	
5	2025-09-29	教師節補假	
6	2025-10-06	中秋節	
7	2025-10-13	Fourier Transform and Sampling	#2 – 2D-DCT, due: 2025-10-27 1:59pm
8	2025-10-20	Motion Estimation	Final project assigned (group together in fours)
9	2025-10-27	Lossless Compression	#3 – MEMC, due: 2025-11-10 1:59pm
10	2025-11-03	Image Compression	
11	2025-11-10	Video Compression	#4 – Entropy coding, due: 2025-11-24 1:59pm
12	2025-11-17	Learning-based Image/Video Compression	
13	2025-11-24	Paper Presentation	
14	2025-12-01	Guest Lecturer – MEDIATEK Meta NETFLIX	
15	2025-12-08	Guest Lecturer – MEDIATEK Meta NETFLIX	
16	2025-12-15	Final Project Presentation	

Fourier Transform and Sampling

with slides by Wen-Hsiao Peng, Shao-Yi Chien, Hsueh-Ming Hang, and Aggelos K. Katsaggelos

Fourier Transform and Sampling

- 2D Fourier Transform
- Sampling
- Discrete Fourier Transform
- Filtering in the Frequency Domain
- Change of Sampling Rate
- The Discrete Cosine Transform
- The Two-Dimensional Discrete Cosine Transform

Fourier Transform and Sampling

- **2D Fourier Transform**
- Sampling
- Discrete Fourier Transform
- Filtering in the Frequency Domain
- Change of Sampling Rate
- The Discrete Cosine Transform
- The Two-Dimensional Discrete Cosine Transform

Eigenfunctions of LSI Systems

$$x(n_1,n_2) = e^{j(\omega_1' n_1+\omega_2' n_2)} \longrightarrow \boxed{h(n_1,n_2)} \longrightarrow y(n_1,n_2) = ?$$

$$y(n_1,n_2) = x(n_1,n_2) ** h(n_1,n_2)$$

$$= \sum_{k_1=-\infty}^{\infty} \sum_{k_2=-\infty}^{\infty} e^{j\omega_1'(n_1-k_1)} e^{j\omega_2'(n_2-k_2)} h(k_1,k_2)$$

$$= e^{j\omega_1' n_1} e^{j\omega_2' n_2} \underbrace{\sum_{k_1} \sum_{k_2} h(k_1,k_2) e^{-j\omega_1' k_1} e^{-j\omega_2' k_2}}_{H(\omega_1',\omega_2') \,\triangleq\, \text{frequency response}}$$

2D Fourier Transform

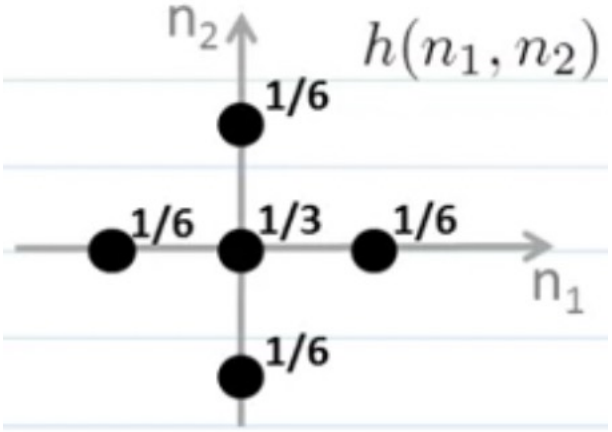
$$X(\omega_1,\omega_2) = \sum_{n_1=-\infty}^{\infty}\sum_{n_2=-\infty}^{\infty} x(n_1,n_2)e^{-j\omega_1 n_1}e^{-j\omega_2 n_2}$$

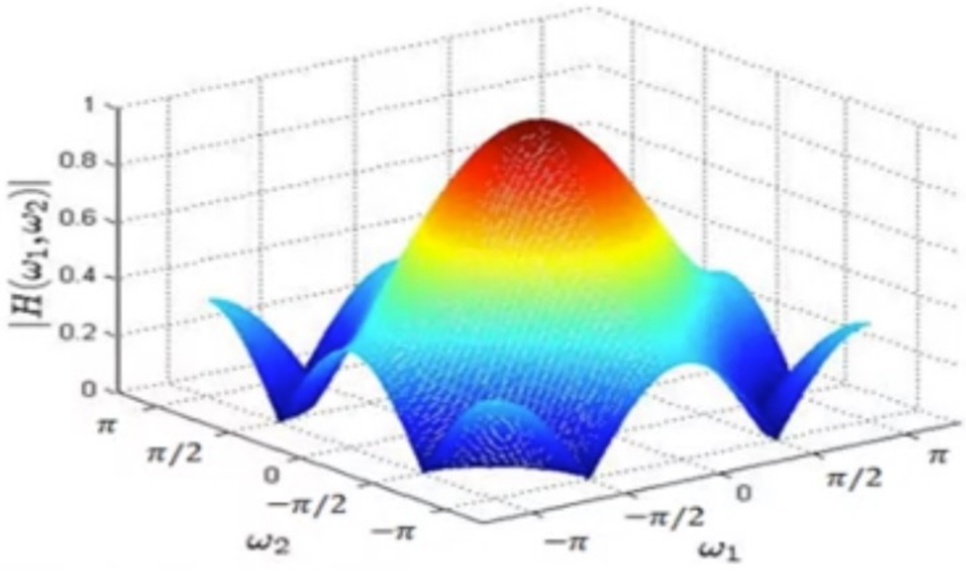
$$x(n_1,n_2) = \frac{1}{4\pi^2}\int_{-\pi}^{\pi}\int_{-\pi}^{\pi} X(\omega_1,\omega_2)\, e^{j\omega_1 n_1}e^{j\omega_2 n_2}d\omega_1 d\omega_2$$

Properties

- $X(\omega_1,\omega_2) = X(\omega_1+2\pi,\omega_2+2\pi)$
- $x(n_1-m_1,n_2-m_2) \leftrightarrow e^{-j\omega_1 m_1 - j\omega_2 m_2}X(\omega_1,\omega_2)$
- $x(n_1,n_2)e^{j\theta_1 n_1 + j\theta_2 n_2} \leftrightarrow X(\omega_1-\theta_1,\omega_2-\theta_2)$
- for $x(n_1,n_2)$ real: $|X(\omega_1,\omega_2)| = |X(-\omega_1,-\omega_2)|$ and $\arg X(\omega_1,\omega_2) = -\arg X(-\omega_1,-\omega_2)$
- $\sum_{n_1}\sum_{n_2}|x(n_1,n_2)|^2 = \frac{1}{4\pi^2}\int_{-\pi}^{\pi}\int_{-\pi}^{\pi}|X(\omega_1,\omega_2)|^2 d\omega_1 d\omega_2$

Example





$$H(\omega_1, \omega_2) = \sum_{n_1=-\infty}^{\infty} \sum_{n_2=-\infty}^{\infty} h(n_1, n_2) e^{-j\omega_1 n_1} e^{-j\omega_2 n_2}$$

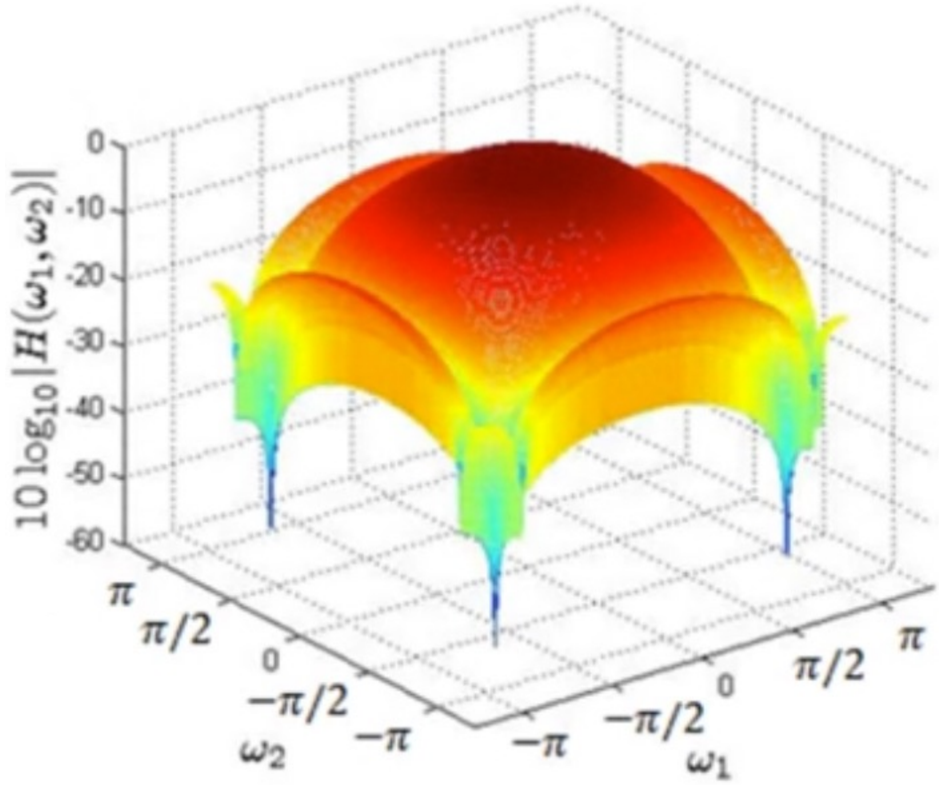
$$= h(0,0) + h(-1,0)e^{j\omega_1} + h(1,0)e^{-j\omega_1} + h(0,-1)e^{j\omega_2} + h(0,1)e^{-j\omega_2}$$

$$= \frac{1}{3} + \frac{1}{6}e^{j\omega_1} + \frac{1}{6}e^{-j\omega_1} + \frac{1}{6}e^{j\omega_2} + \frac{1}{6}e^{-j\omega_2}$$

$$= \frac{1}{3} + \frac{1}{6} \cdot 2\cos\omega_1 + \frac{1}{6} \cdot 2\cos\omega_2$$

$$= \frac{1}{3}(1 + \cos\omega_1 + \cos\omega_2)$$

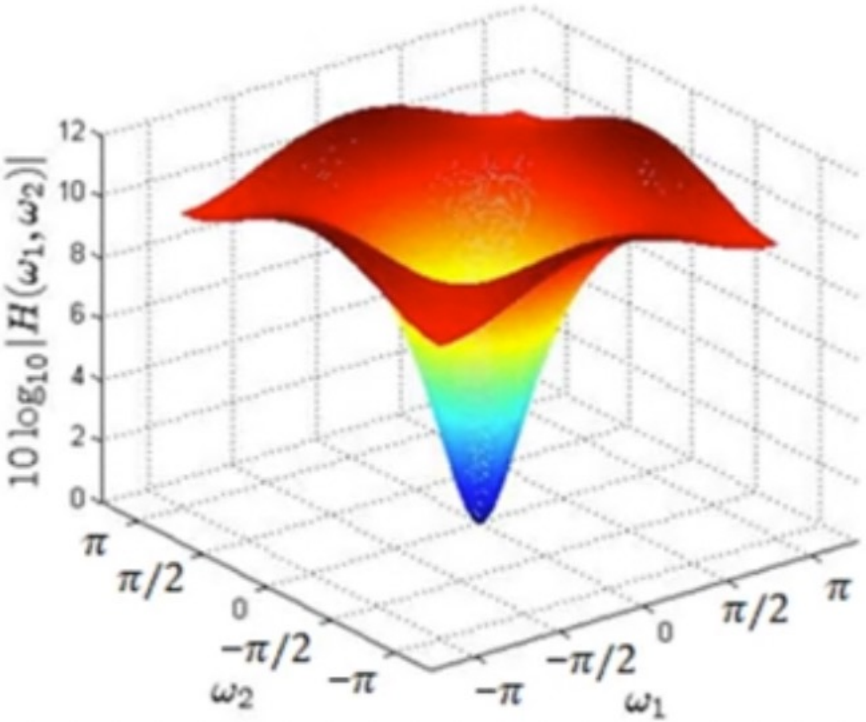
Example



$$h(n_1, n_2) = \begin{bmatrix} 0.075 & 0.124 & 0.075 \\ 0.124 & 0.204 & 0.124 \\ 0.075 & 0.124 & 0.075 \end{bmatrix}$$

$H(\omega_1, \omega_2) = 0.204 + 0.124 \cdot 2 \cdot \cos\omega_1 + 0.124 \cdot 2 \cdot \cos\omega_2 + 0.075 \cdot 2 \cdot \cos(\omega_1 + \omega_2) + 0.075 \cdot 2 \cdot \cos(\omega_1 - \omega_2)$

Example



$$h(n_1,n_2)=\begin{bmatrix}-1 & -1 & -1\\ -1 & 9 & -1\\ -1 & -1 & -1\end{bmatrix}$$

$$H(\omega_1,\omega_2)$$
$$=9-2\cdot\cos\omega_1-2\cdot\cos\omega_2-2\cdot\cos(\omega_1+\omega_2)\ \ -2\cdot\cos(\omega_1-\omega_2)$$

The Convolution Theorem

$$x(n_1,n_2) \longrightarrow \boxed{h(n_1,n_2)} \longrightarrow y(n_1,n_2) =$$

$$y(n_1,n_2) = T[x(n_1,n_2)]$$

$$= T\left[\frac{1}{4\pi^2}\iint_{-\pi}^{\pi} X(\omega_1,\omega_2)e^{j\omega_1 n_1}e^{j\omega_2 n_2}d\omega_1 d\omega_2\right]$$

$$= \frac{1}{4\pi^2}\iint_{-\pi}^{\pi} X(\omega_1,\omega_2)T[e^{j\omega_1 n_1}e^{j\omega_2 n_2}]\,d\omega_1 d\omega_2$$

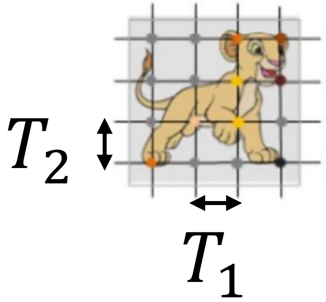
$$= \frac{1}{4\pi^2}\iint_{-\pi}^{\pi} X(\omega_1,\omega_2)H(\omega_1,\omega_2)\,e^{j\omega_1 n_1}e^{j\omega_2 n_2}d\omega_1 d\omega_2$$

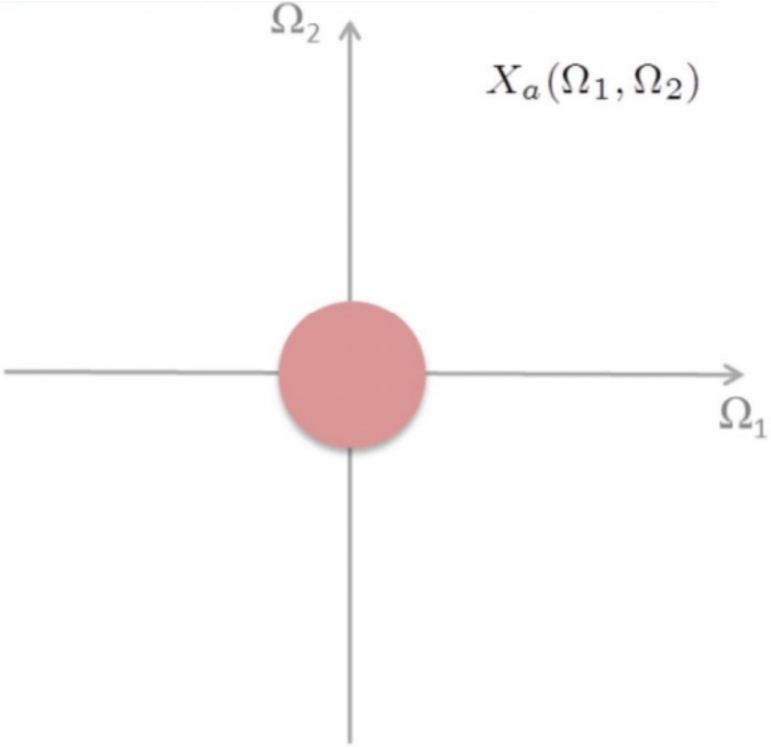
$$\Rightarrow Y(\omega_1,\omega_2) = X(\omega_1,\omega_2)\cdot H(\omega_1,\omega_2)$$

Fourier Transform and Sampling

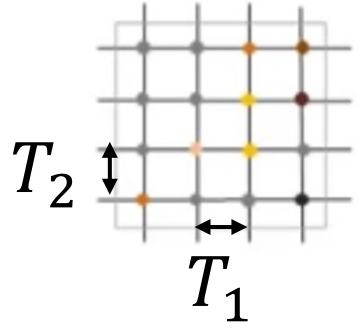
- 2D Fourier Transform
- **Sampling**
- Discrete Fourier Transform
- Filtering in the Frequency Domain
- Change of Sampling Rate
- The Discrete Cosine Transform
- The Two-Dimensional Discrete Cosine Transform

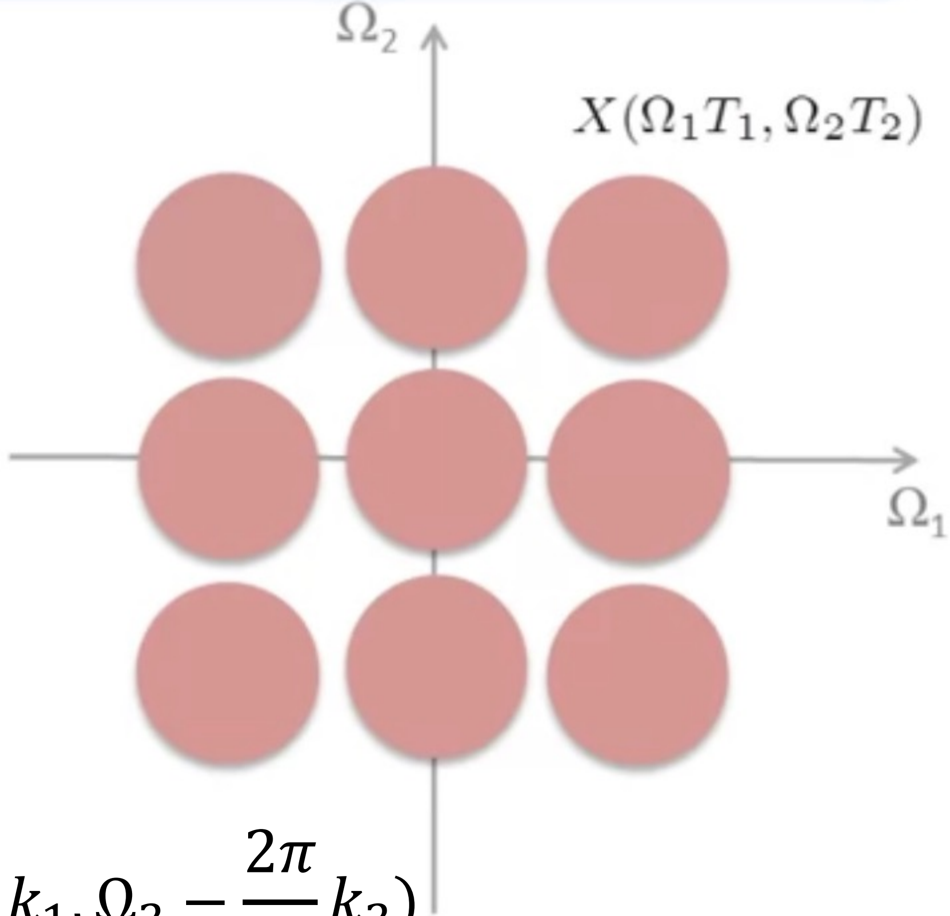
2D Sampling





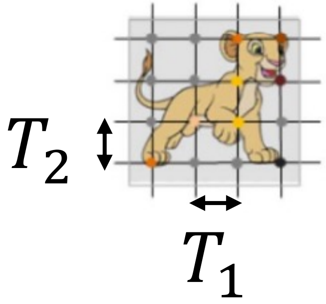
2D Sampling

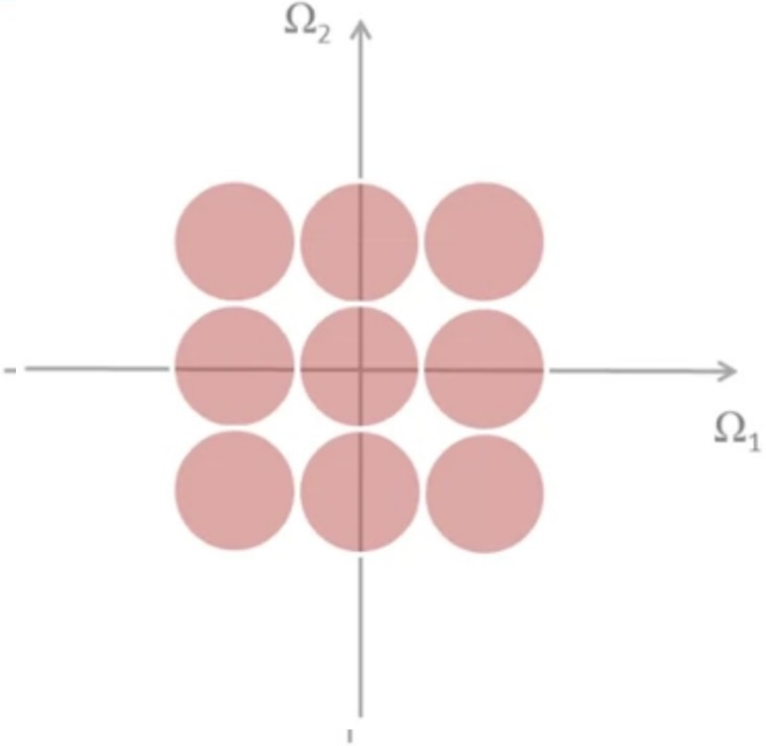




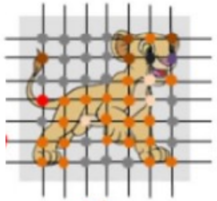
$$X(\Omega_1 T_1, \Omega_2 T_2) = \frac{1}{T_1 T_2} \sum_{k_1=-\infty}^{\infty} \sum_{k_2=-\infty}^{\infty} X_a\left(\Omega_1 - \frac{2\pi}{T_1} k_1, \Omega_2 - \frac{2\pi}{T_2} k_2\right)$$

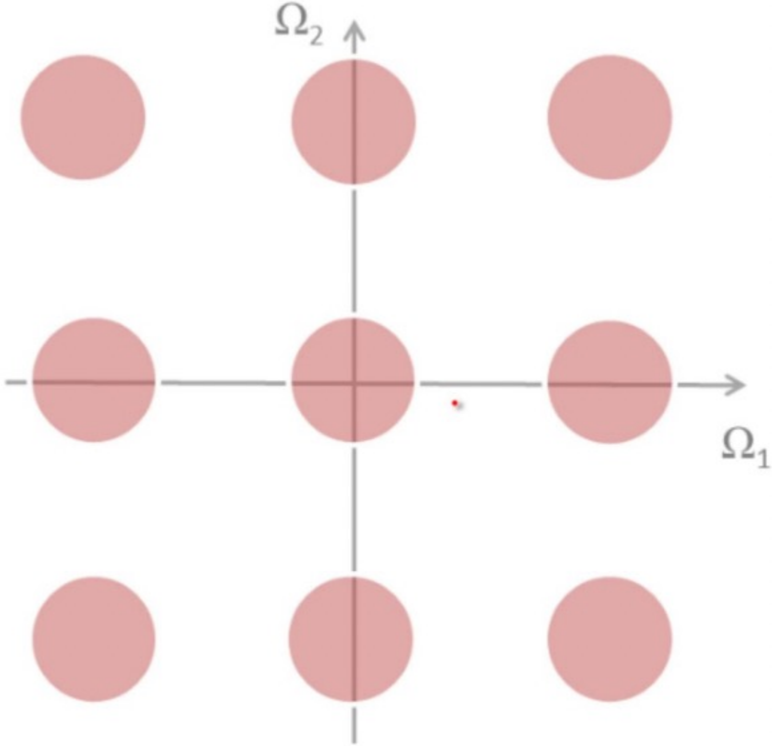
Critically Sampled



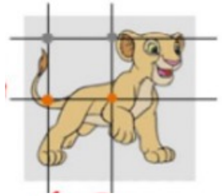


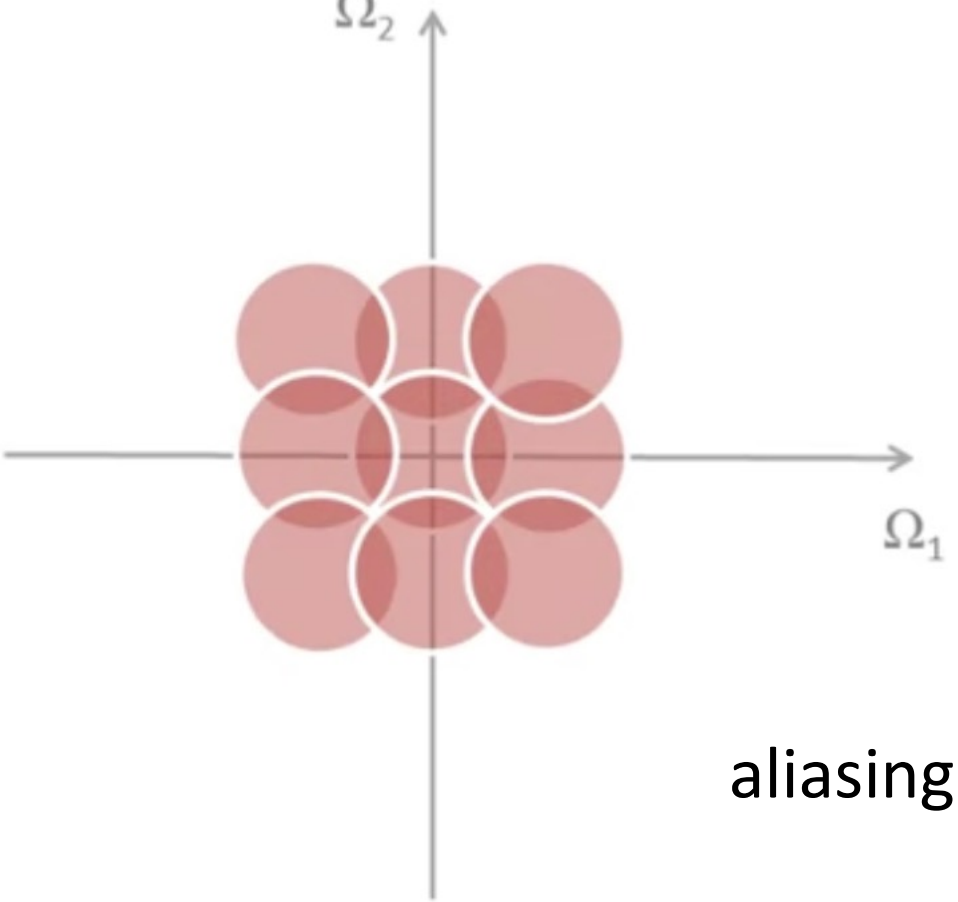
Over-Sampled





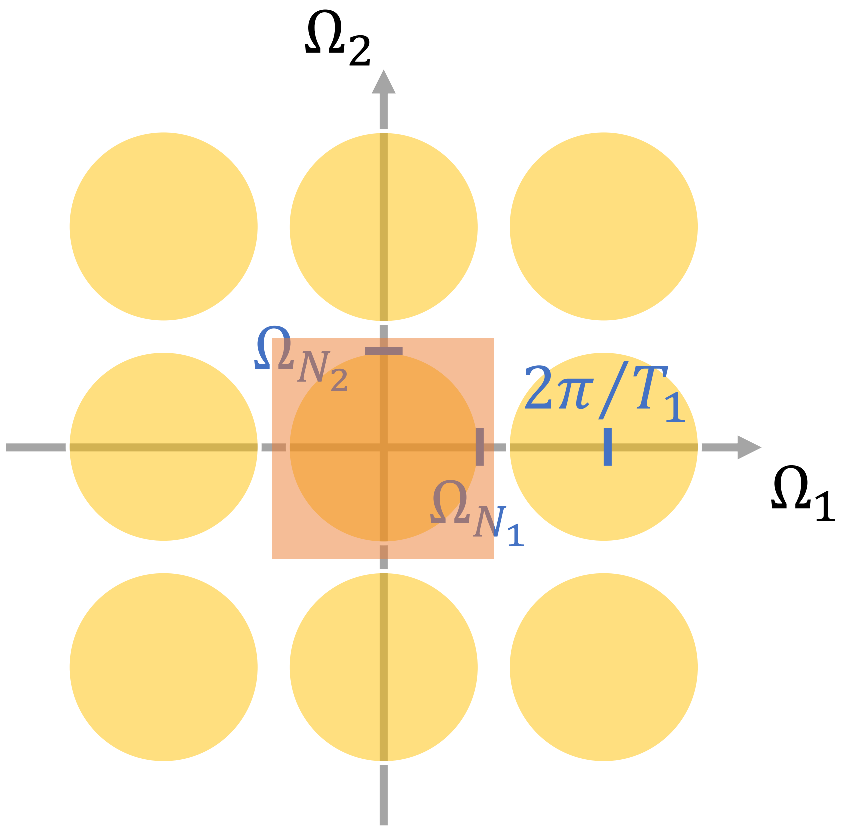
Under-Sampled





aliasing

2D Nyquist Theorem

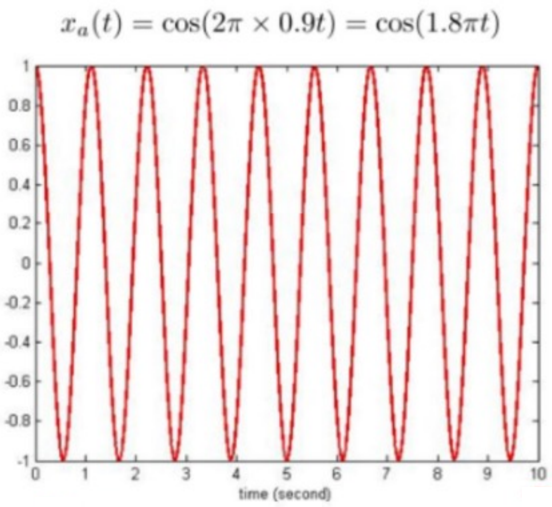


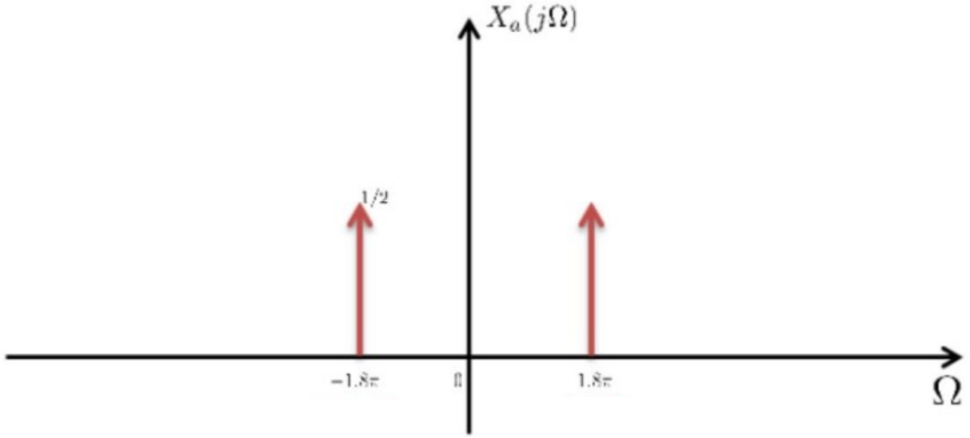
$$\frac{2\pi}{T_1} - \Omega_{N_1} \geq \Omega_{N_1} \Rightarrow \frac{2\pi}{T_1} \geq 2\Omega_{N_1}$$

$$\frac{2\pi}{T_2} - \Omega_{N_2} \geq \Omega_{N_2} \Rightarrow \frac{2\pi}{T_2} \geq 2\Omega_{N_2}$$

$$F(\Omega_1, \Omega_2) = \begin{cases} T_1 T_2, & \begin{array}{r} |\Omega_1| < \pi/T_1 \\ |\Omega_2| < \pi/T_2 \end{array} \\ 0, & \text{otherwise} \end{cases}$$

Analog Signal





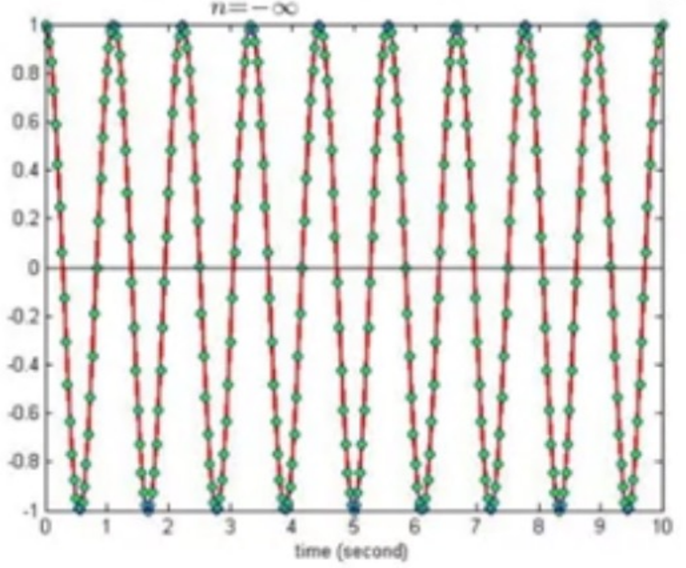
$$f = 0.9 \text{ Hz}$$
$$\Omega = 2\pi f = 1.8\pi \text{ rads/sec}$$

Nyquist freq: $\begin{cases} 1.8\text{Hz} \\ 2\pi \cdot 1.8 \dfrac{\text{rads}}{\text{sec}} \end{cases}$

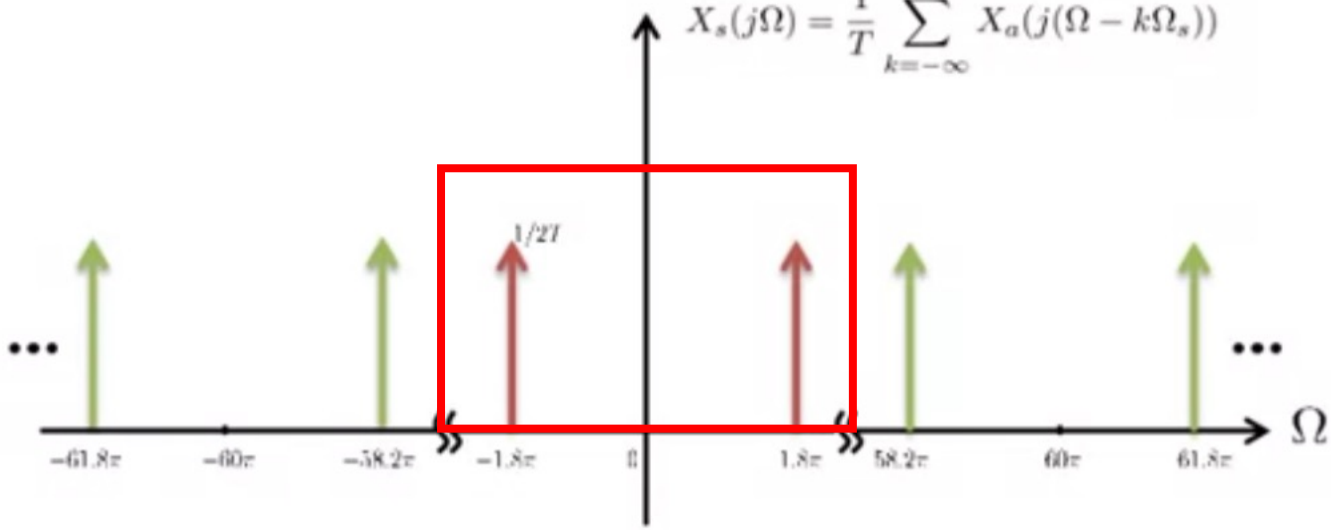
Over-Sampling

$$\Omega_N = 1.8 \times 2\pi = 3.6\pi \text{ rads/sec}$$

$$x_s = \sum_{n=-\infty}^{\infty} x_a(nT)\delta(t - nT) \quad T = \frac{1}{30}\text{s}, \Omega_s = \frac{2\pi}{T} = 60\pi \text{ rads/s}$$

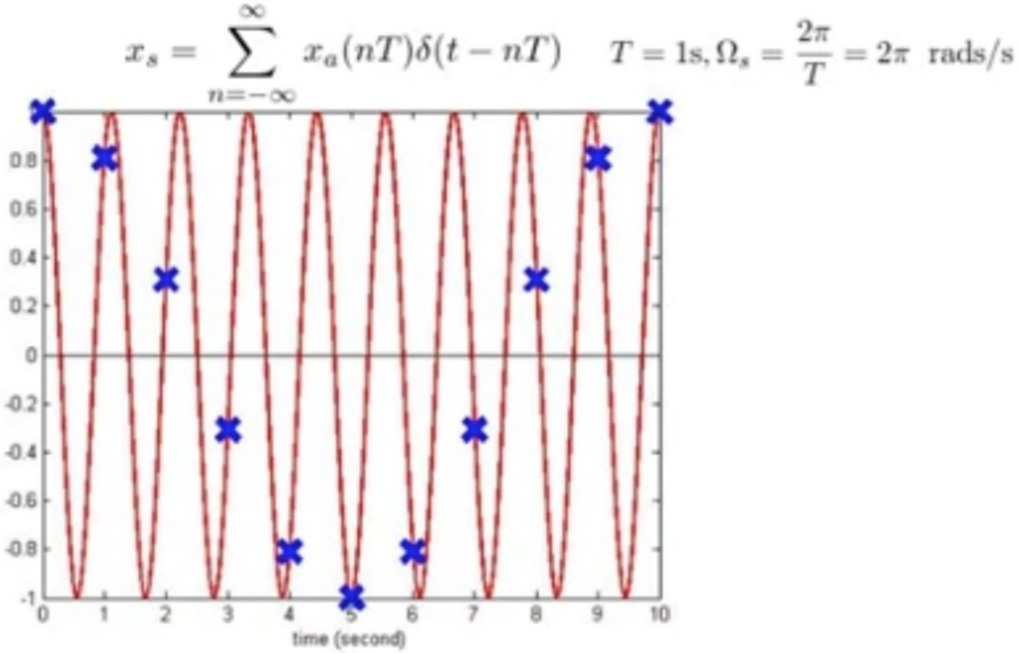


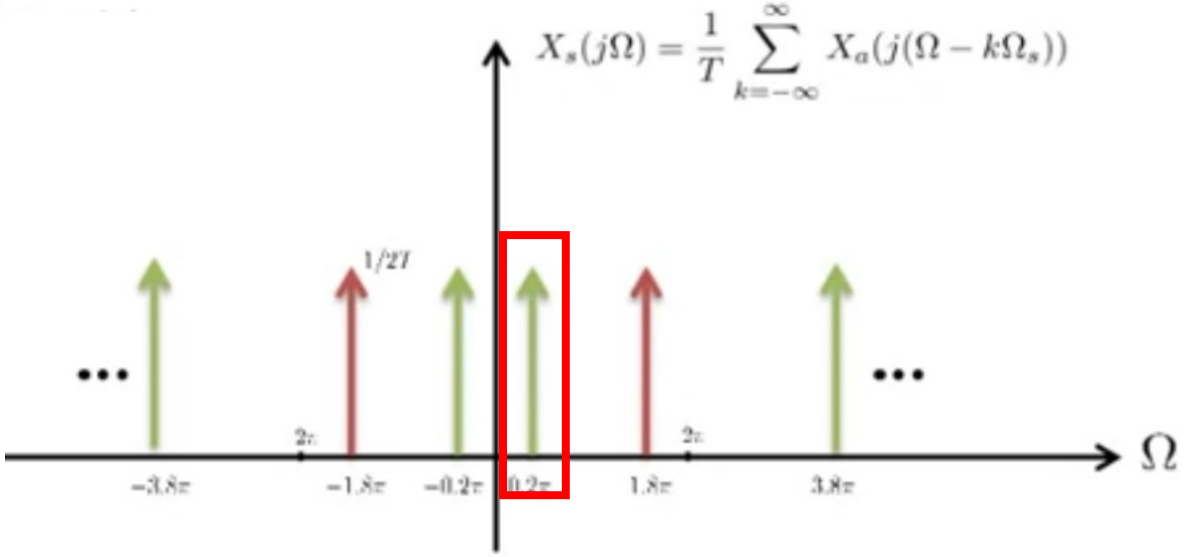
$$X_s(j\Omega) = \frac{1}{T}\sum_{k=-\infty}^{\infty} X_a(j(\Omega - k\Omega_s))$$



Under-Sampling

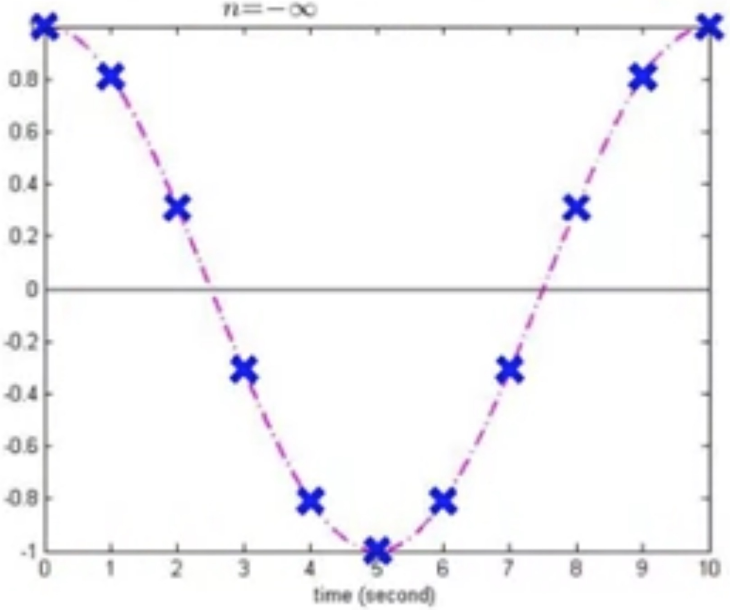
$$\Omega_N = 3.6\pi \text{ rads/sec}$$

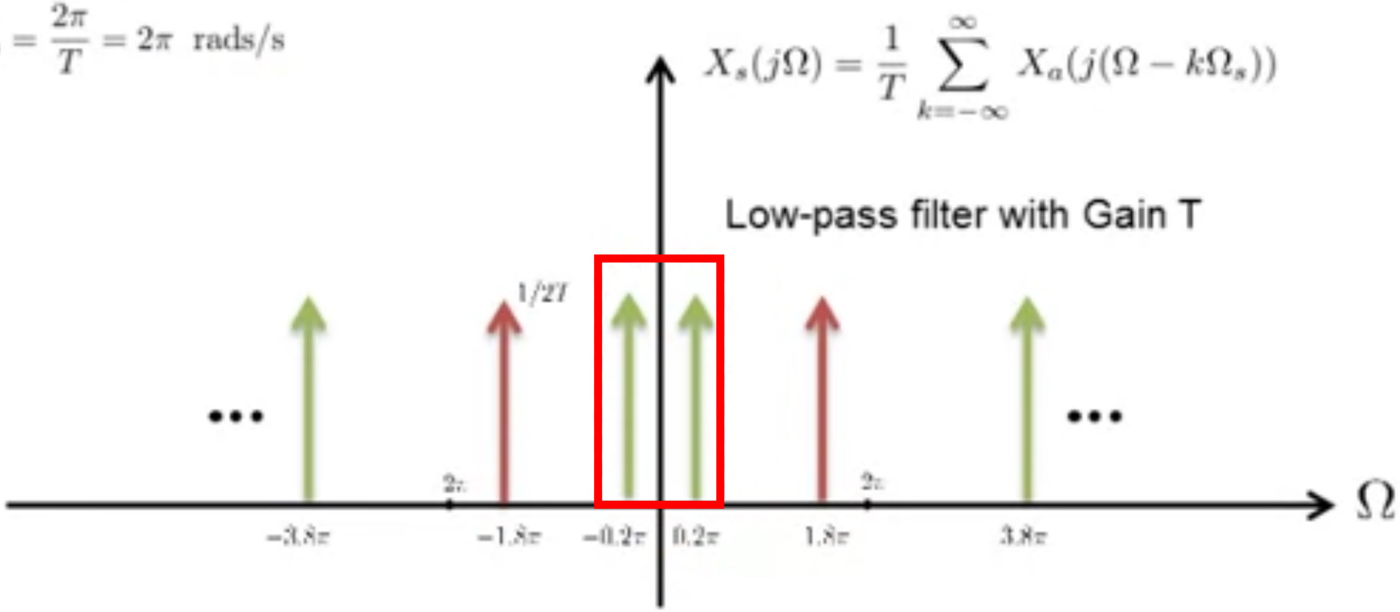




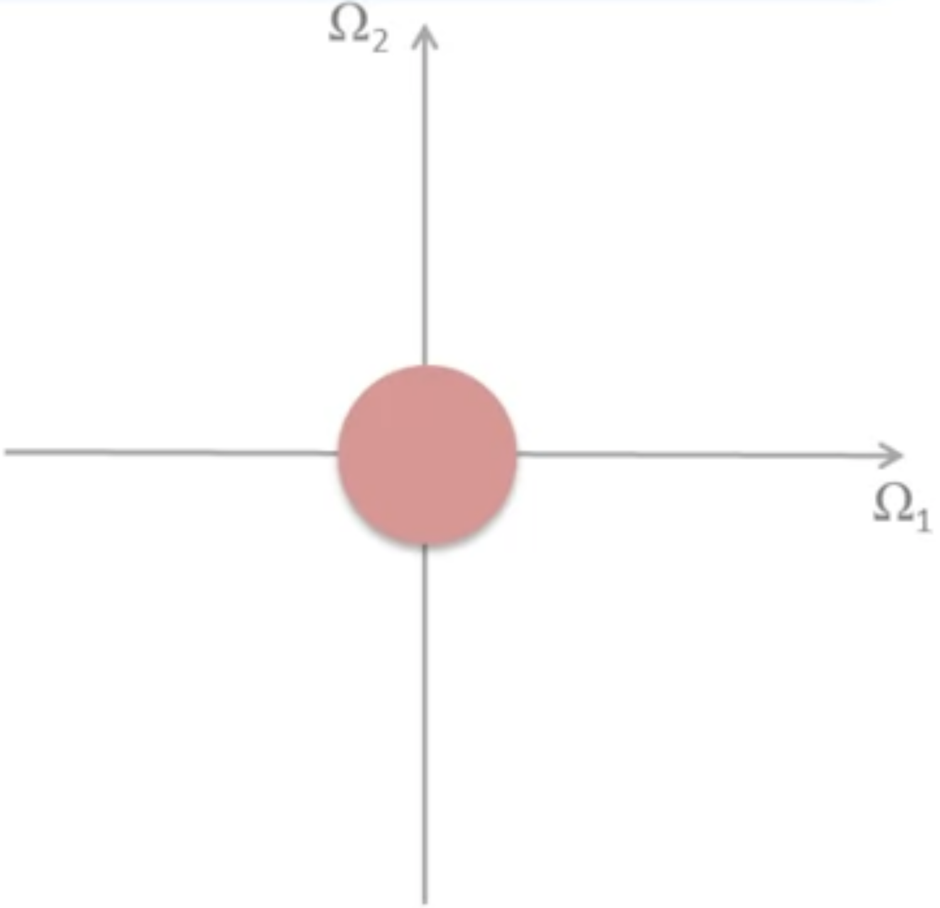
Reconstructed Aliased Signal

$$x_s = \sum_{n=-\infty}^{\infty} x_a(nT)\delta(t - nT) \qquad T = 1\text{s}, \Omega_s = \frac{2\pi}{T} = 2\pi \ \text{rads/s}$$

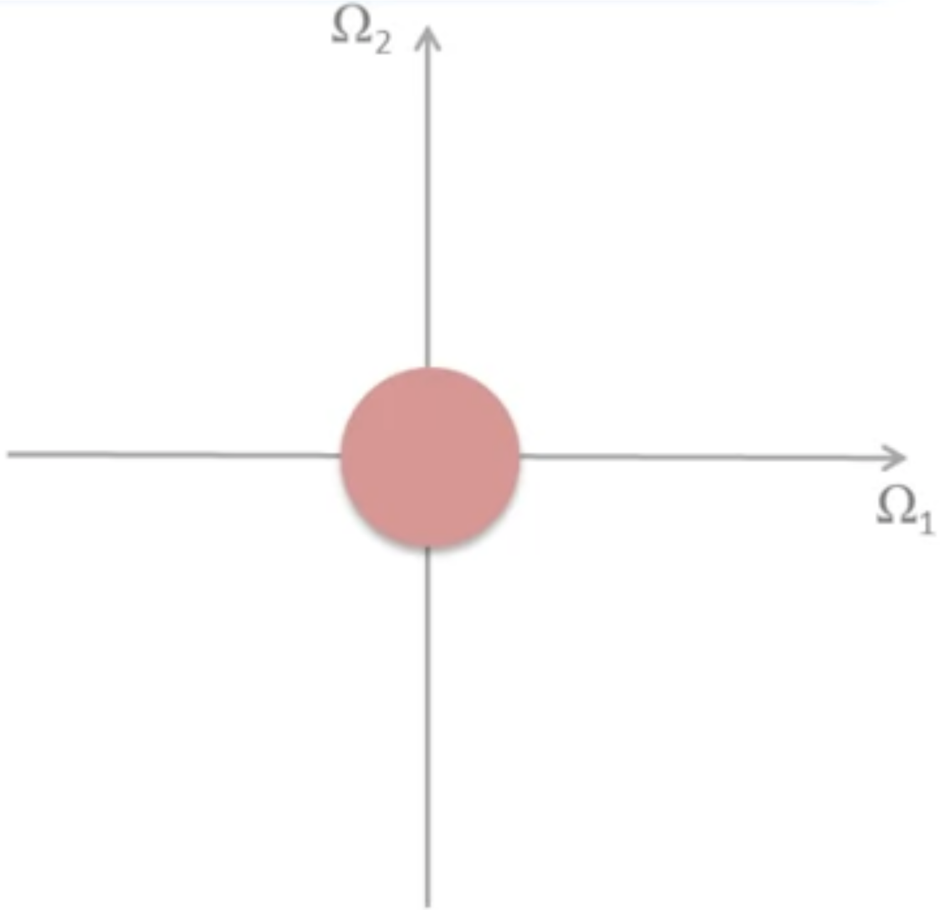




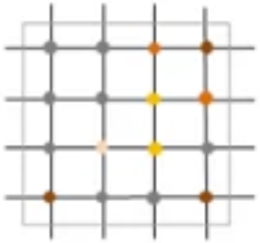
Sampling in the Frequency Domain

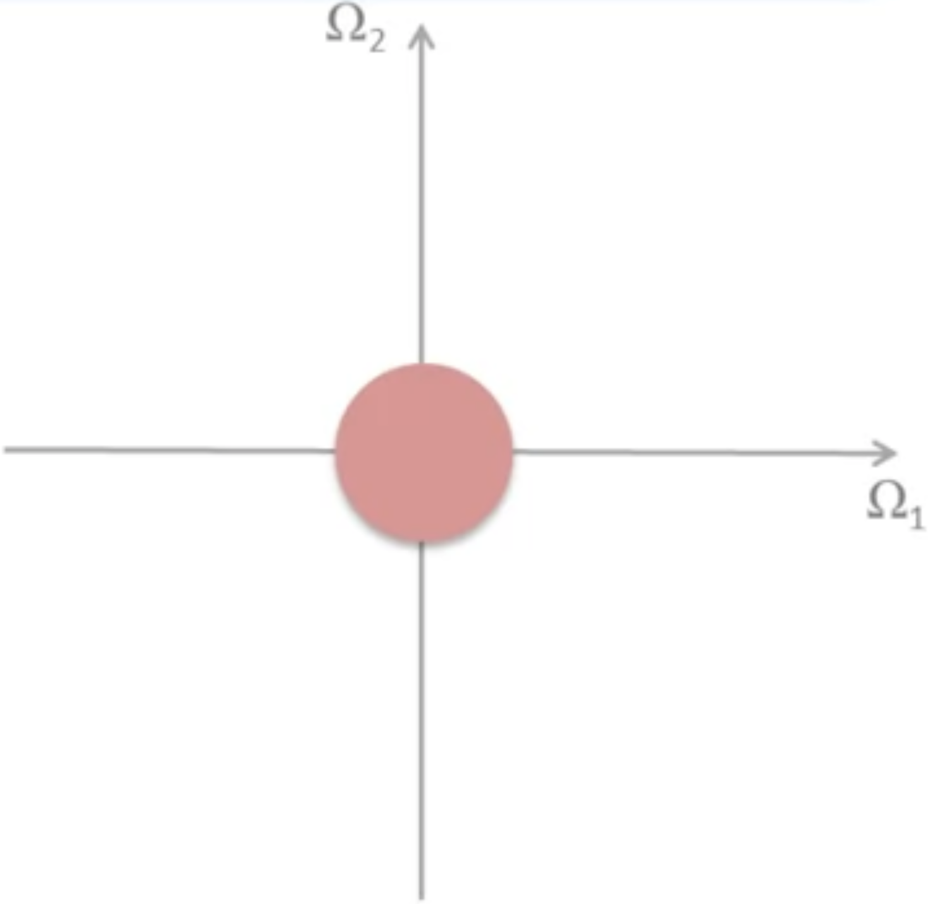


Sampling in the Frequency Domain

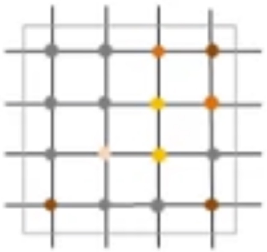


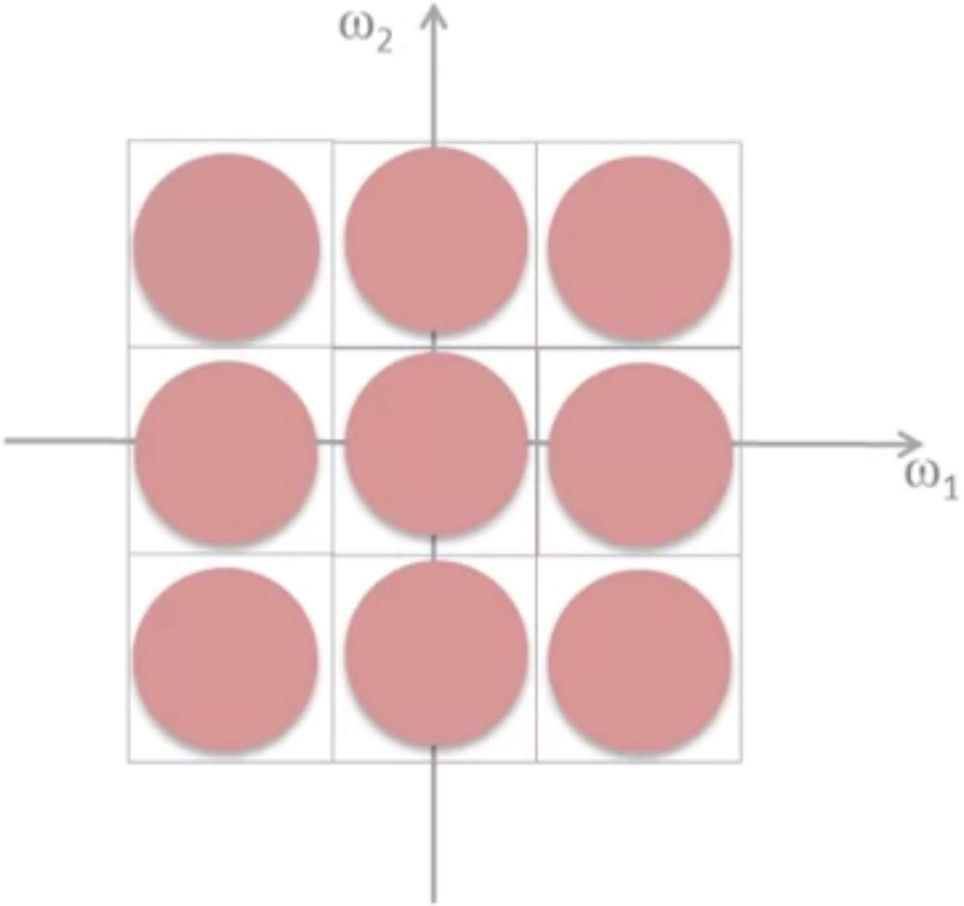
Sampling in the Frequency Domain



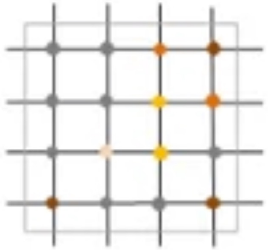


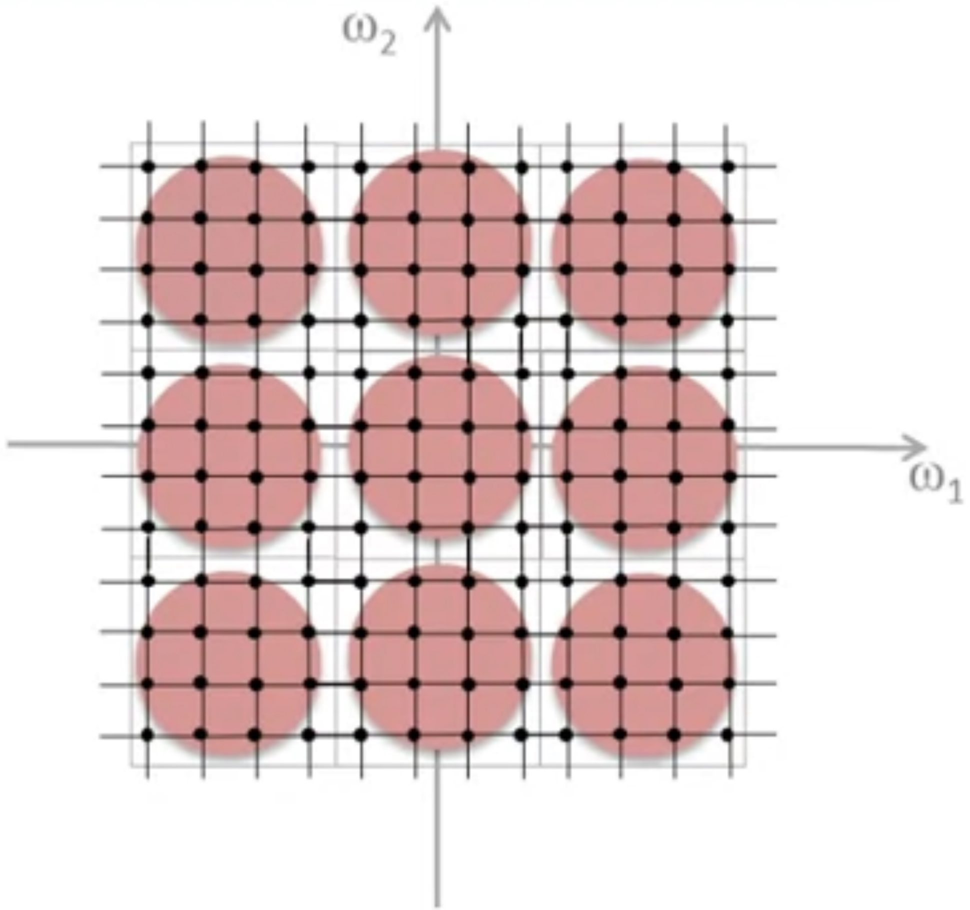
Sampling in the Frequency Domain



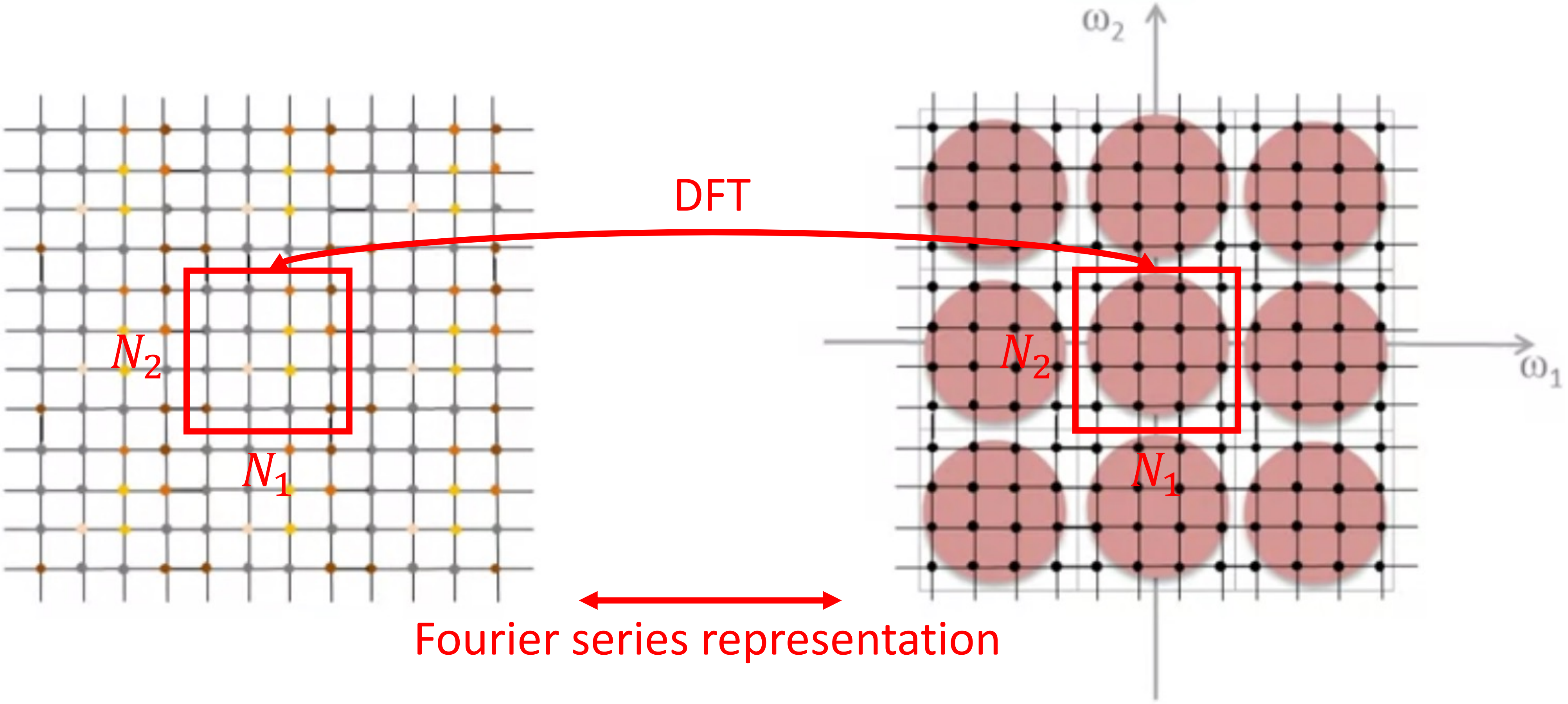


Sampling in the Frequency Domain





Sampling in the Frequency Domain



Fourier Transform and Sampling

- 2D Fourier Transform
- Sampling
- **Discrete Fourier Transform**
- Filtering in the Frequency Domain
- Change of Sampling Rate
- The Discrete Cosine Transform
- The Two-Dimensional Discrete Cosine Transform

Discrete Fourier Transform (DFT)

$$X(\omega_1, \omega_2) = \sum_{n_1=0}^{N_1-1} \sum_{n_2=0}^{N_2-1} x(n_1, n_2) e^{-j\omega_1 n_1} e^{-j\omega_2 n_2}$$

$$X(k_1, k_2) = X(\omega_1, \omega_2) \Big|_{\omega_1 = \frac{2\pi}{N_1}k_1, \omega_2 = \frac{2\pi}{N_2}k_2} \quad \begin{cases} k_1 = 0, \dots, N_1 - 1 \\ k_2 = 0, \dots, N_2 - 1 \end{cases}$$

$$X(k_1, k_2) = \sum_{n_1=0}^{N_1-1} \sum_{n_2=0}^{N_2-1} x(n_1, n_2) e^{-j\frac{2\pi}{N_1}n_1 k_1} e^{-j\frac{2\pi}{N_2}n_2 k_2} \quad \begin{cases} k_1 = 0, \dots, N_1 - 1 \\ k_2 = 0, \dots, N_2 - 1 \end{cases}$$

$$x(n_1, n_2) = \frac{1}{N_1 N_2} \sum_{n_1=0}^{N_1-1} \sum_{n_2=0}^{N_2-1} X(k_1, k_2)\, e^{j\frac{2\pi}{N_1}n_1 k_1} e^{j\frac{2\pi}{N_2}n_2 k_2} \quad \begin{cases} n_1 = 0, \dots, N_1 - 1 \\ n_2 = 0, \dots, N_2 - 1 \end{cases}$$

FT linear shifts <-> DFT circular shifts

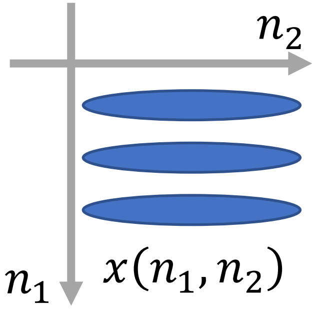
Fast Fourier Transforms (FFTs)

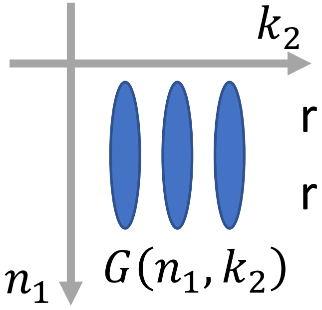
$$X(k_1, k_2) = \sum_{n_1=0}^{N_1-1} \sum_{n_2=0}^{N_2-1} x(n_1, n_2) e^{-j\frac{2\pi}{N_1}n_1 k_1} e^{-j\frac{2\pi}{N_2}n_2 k_2} \quad \begin{cases} k_1 = 0, \dots, N_1 - 1 \\ k_2 = 0, \dots, N_2 - 1 \end{cases}$$

A. Direct Computations
   For each (k_1, k_2): $N_1 N_2$ mults; Total $N_1^2 N_2^2$ mults ($N_1 = N_2 = N$: N^4)

B. Row-Column Decomposition

$$X(k_1, k_2) = \sum_{n_1=0}^{N_1-1} \left[\sum_{n_2=0}^{N_2-1} x(n_1, n_2) e^{-j\frac{2\pi}{N_2}n_2 k_2} \right] e^{-j\frac{2\pi}{N_1}n_1 k_1}$$





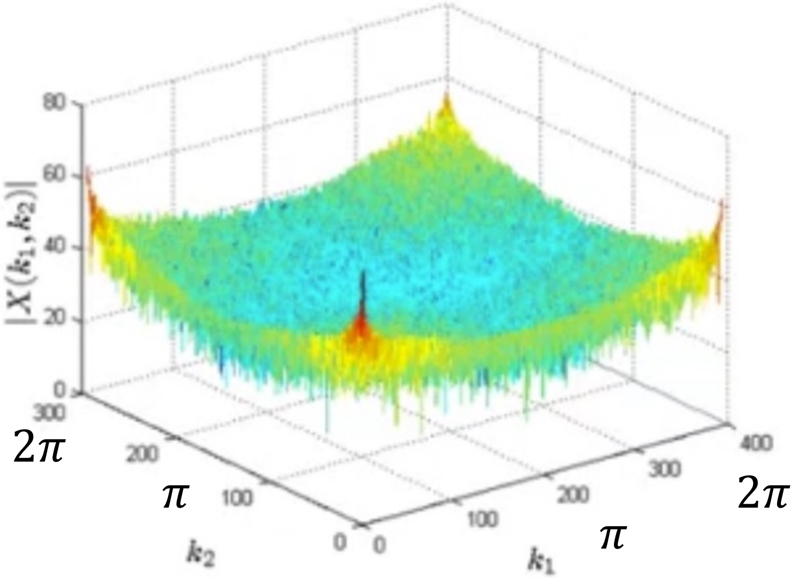
r|c directly: $N_1 \cdot N_2^2 + N_2 \cdot N_1^2 = N_1 N_2 (N_1 + N_2)$ ($N_1 = N_2 = N$: $2N^3$)

r|c FFT: $N_1 \cdot \frac{N_2}{2} \log_2 N_2 + N_2 \cdot \frac{N_1}{2} \log_2 N_1 = \frac{N_1 N_2}{2} \log(N_1 N_2)$ ($N_1 = N_2 = N$: $N^2 \log_2 N$)

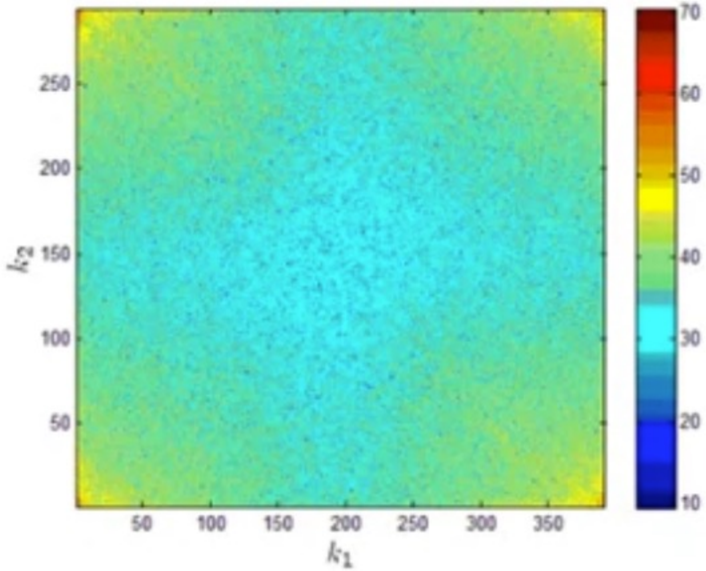
DFT

N_2

N_1

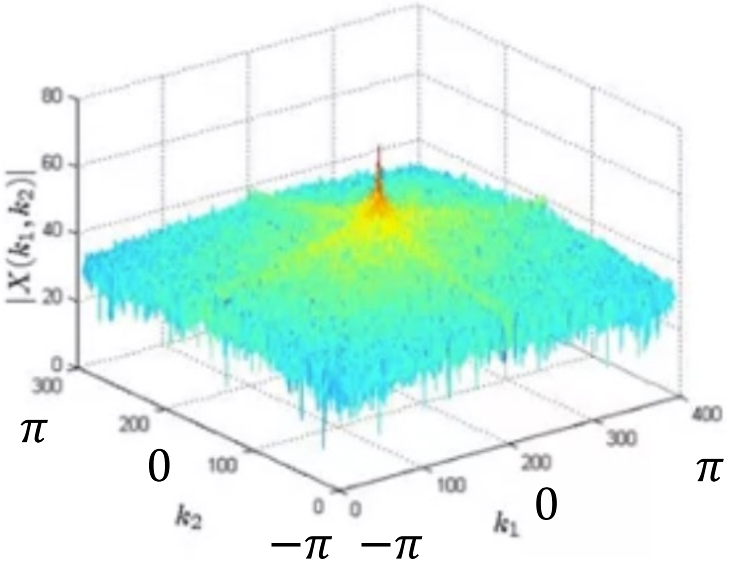


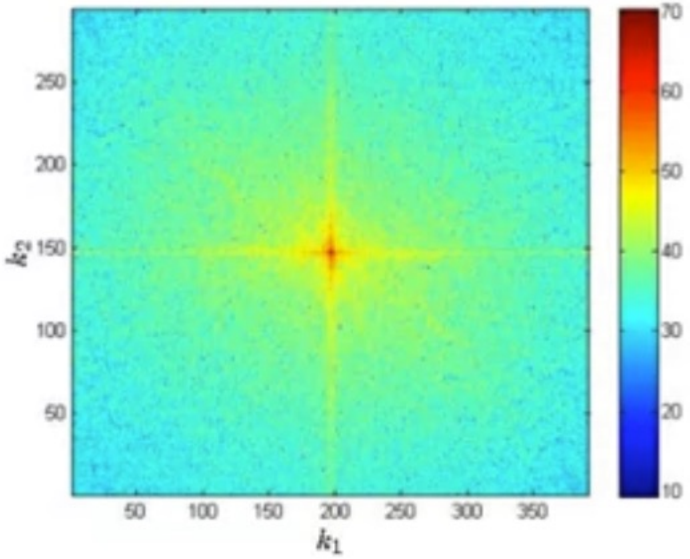
$|X(k_1,k_2)|$
80
60
40
20
0
300
200
100
0
2π
π
k_2
0
100
200
300
400
π
2π
k_1



k_2
250
200
150
100
50
50
100
150
200
250
300
350
k_1
70
60
50
40
30
20
10

DFT (centered)



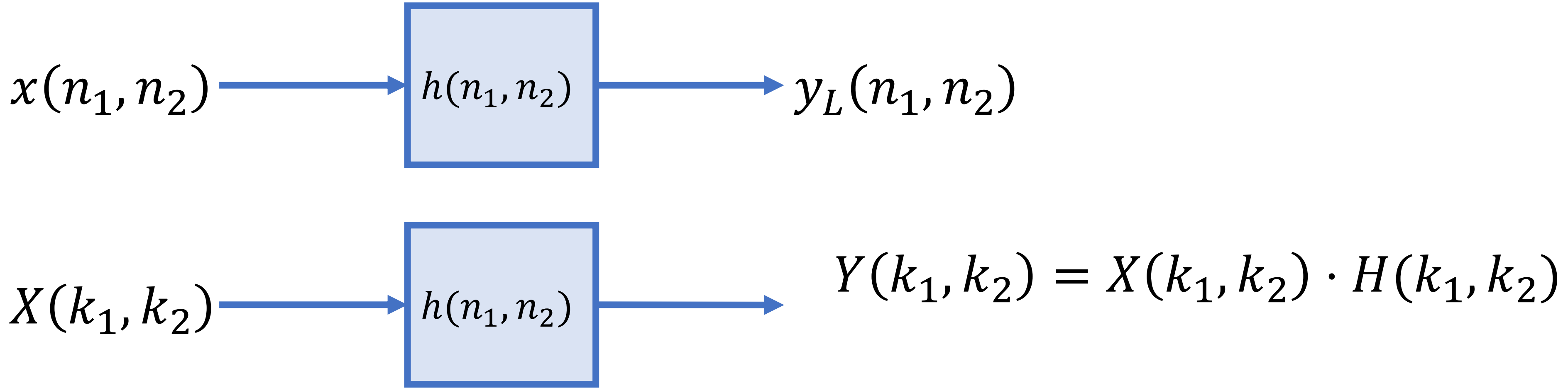


$$(-1)^{n_1+n_2}$$
$$= e^{-j\pi(n_1+n_2)}$$

Fourier Transform and Sampling

- 2D Fourier Transform
- Sampling
- Discrete Fourier Transform
- **Filtering in the Frequency Domain**
- Change of Sampling Rate
- The Discrete Cosine Transform
- The Two-Dimensional Discrete Cosine Transform

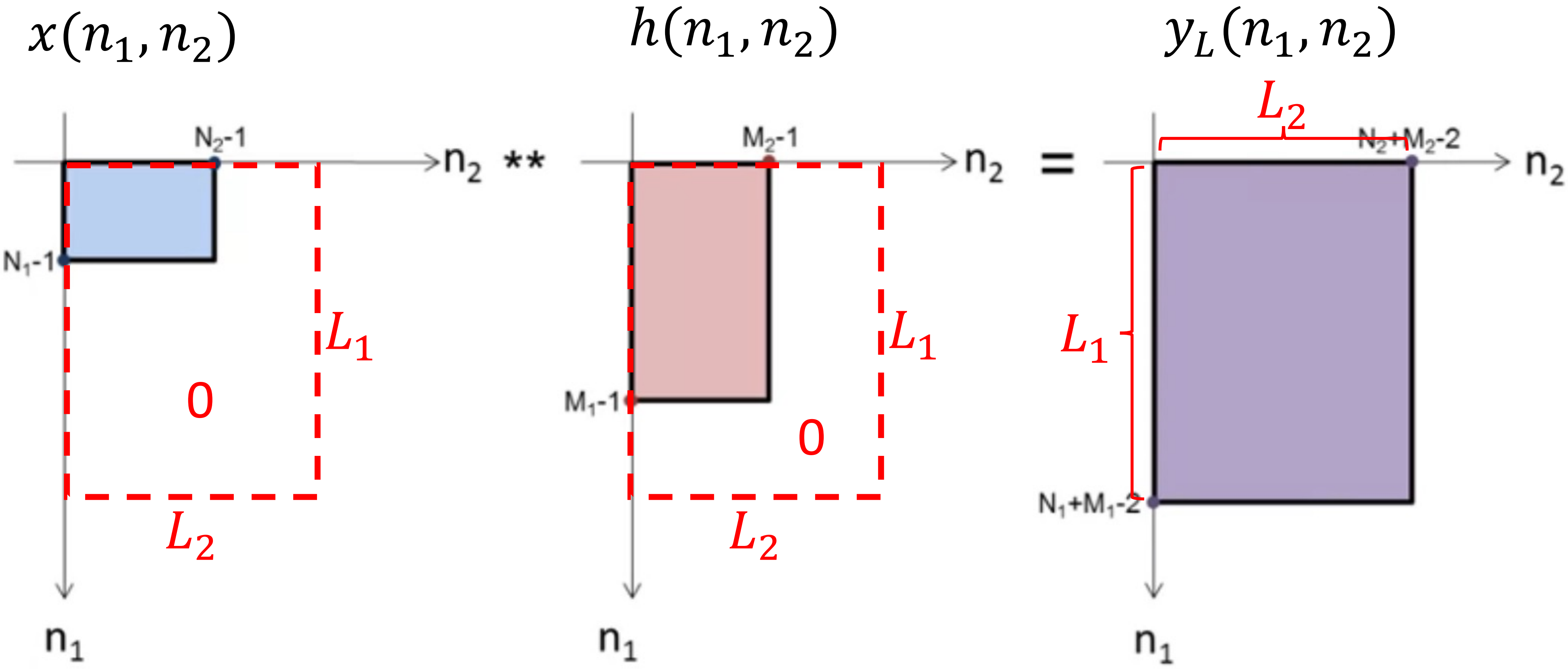
2D Circular Convolution



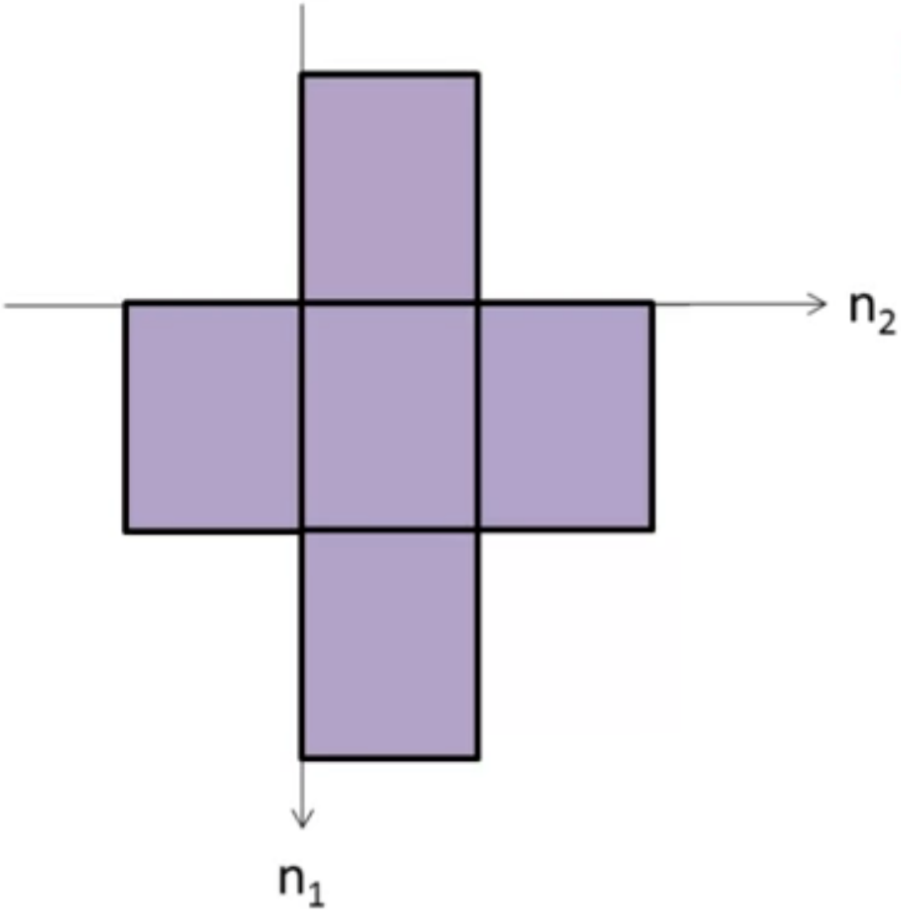
Circular Convolution

$$y(n_1, n_2) = \sum_{r_1} \sum_{r_2} y_L(n_1 - r_1 N_1, n_2 - r_2 N_2) \begin{cases} n_1 = 0, \dots, N_1 - 1 \\ n_2 = 0, \dots, N_2 - 1 \end{cases}$$

ROS of Linear Convolution

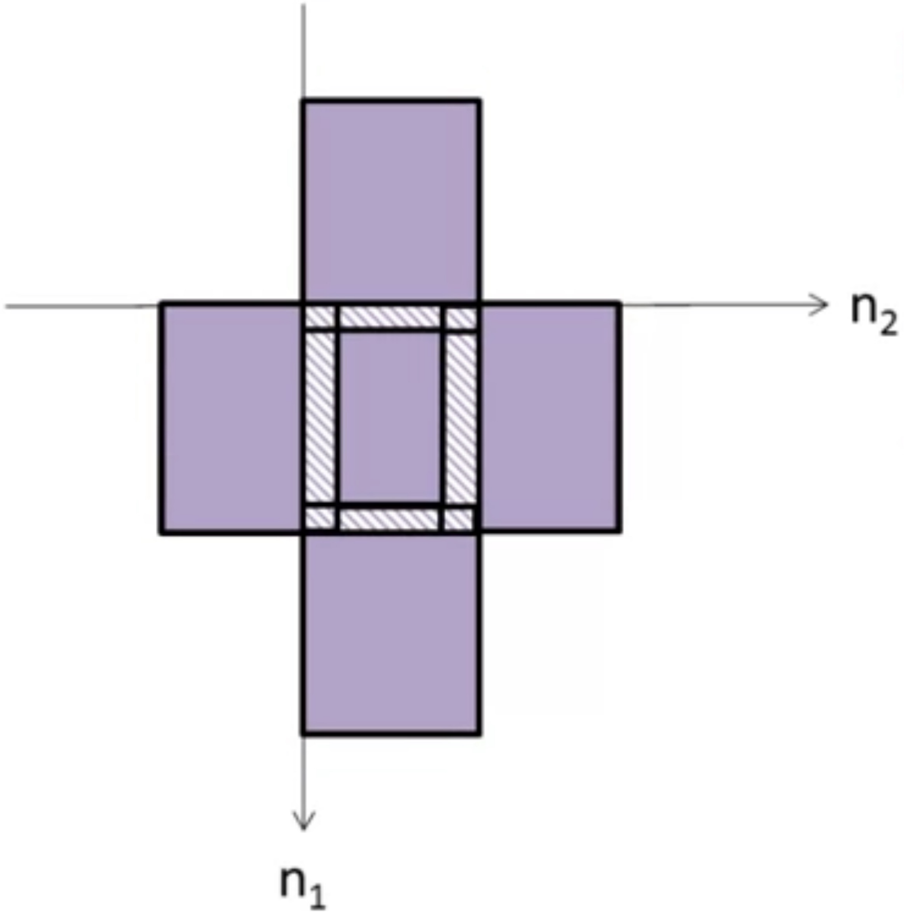


Spatial Aliasing



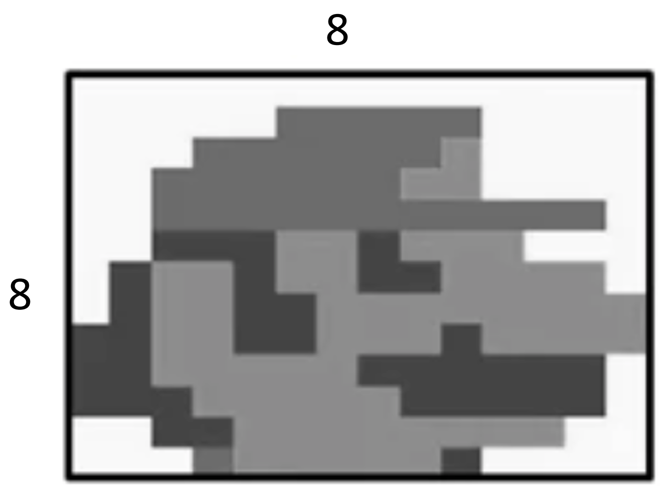
L_1, L_2
$L_1' < L_1$
$L_2' < L_2$

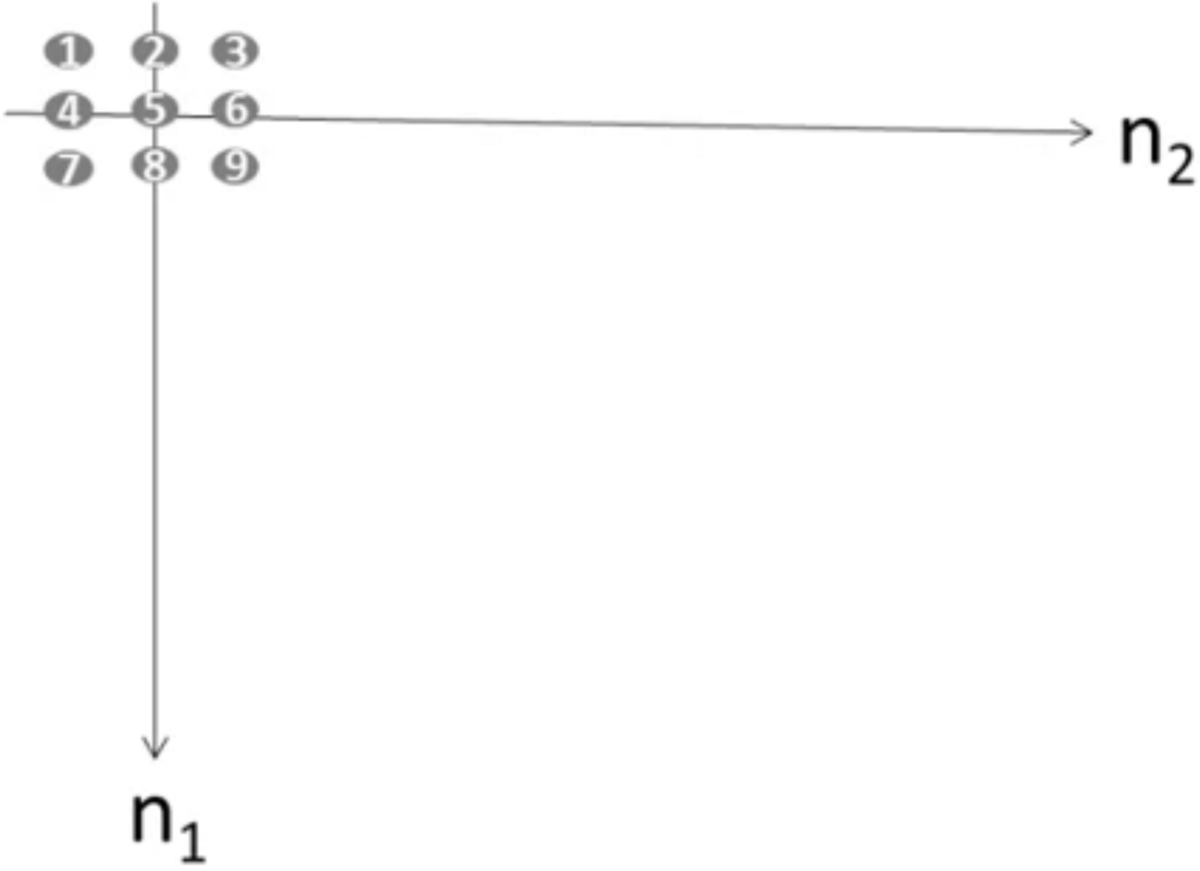
Spatial Aliasing



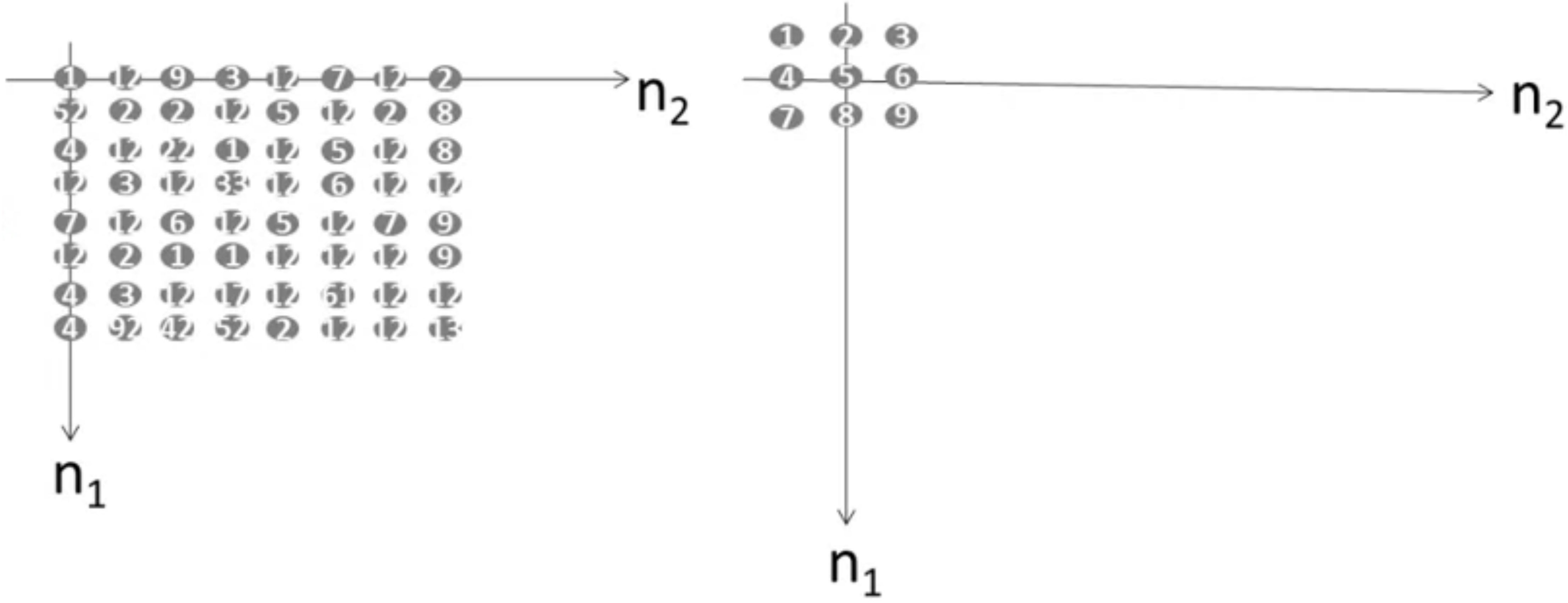
L_1, L_2
$L_1' < L_1$
$L_2' < L_2$

Use of DFT for Filtering

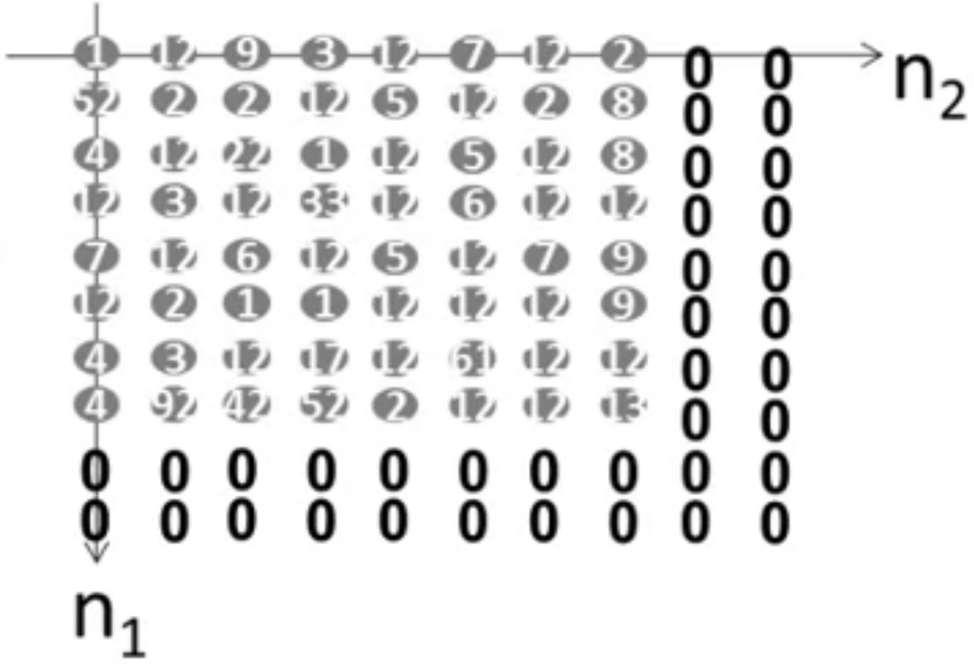


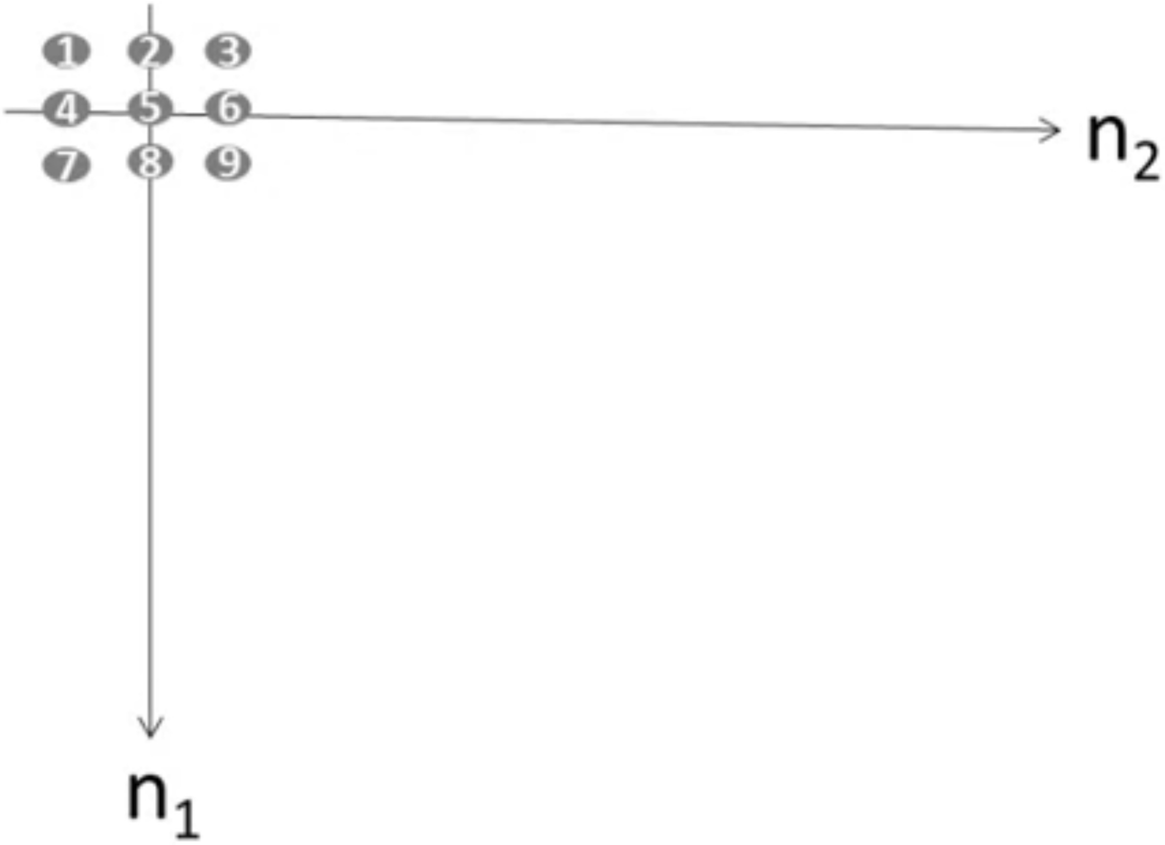


Use of DFT for Filtering

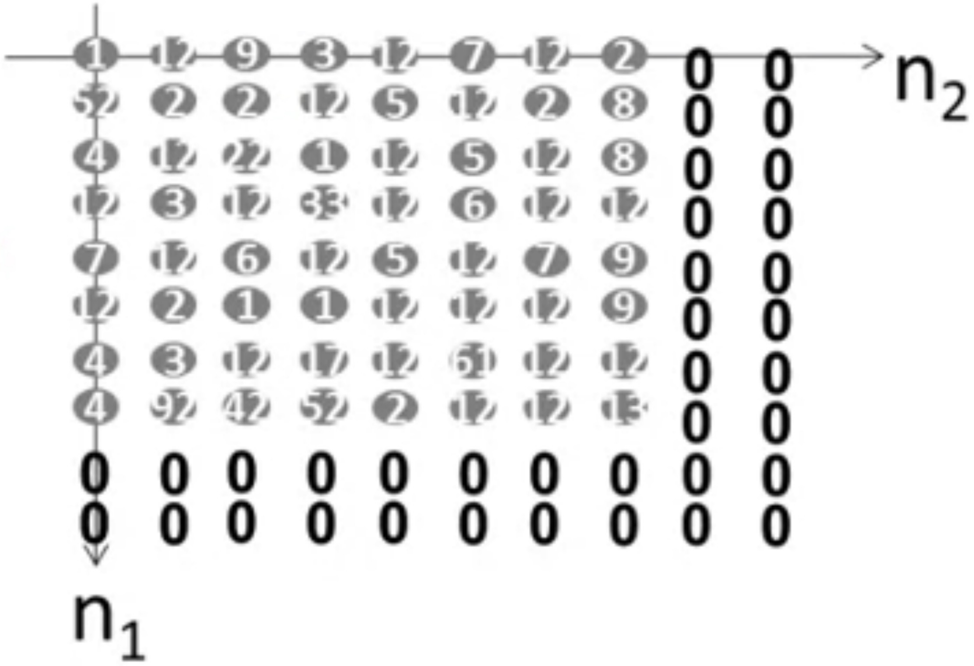


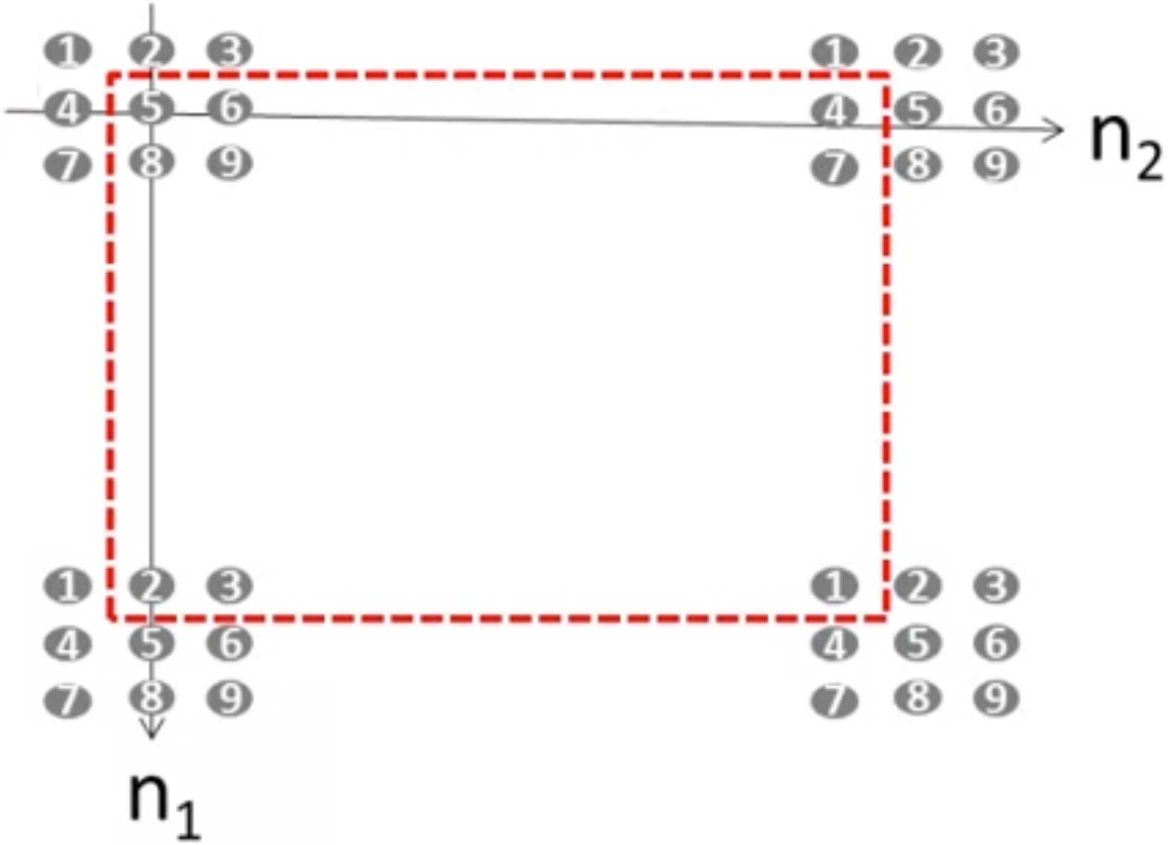
Use of DFT for Filtering



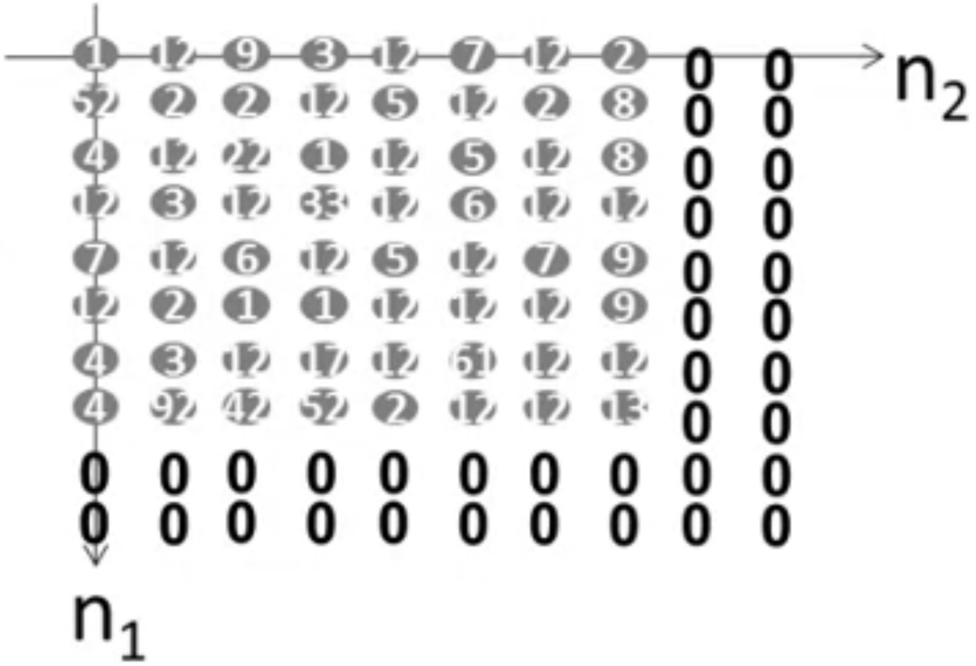


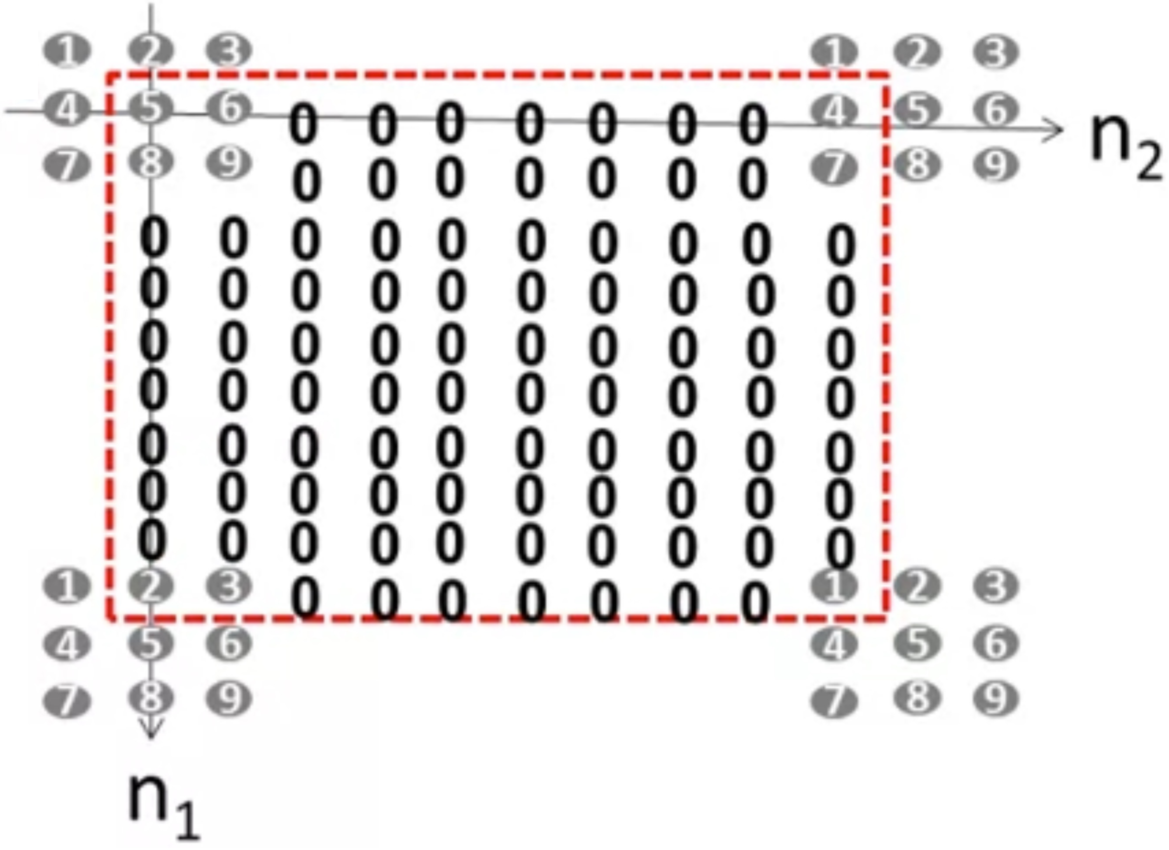
Use of DFT for Filtering





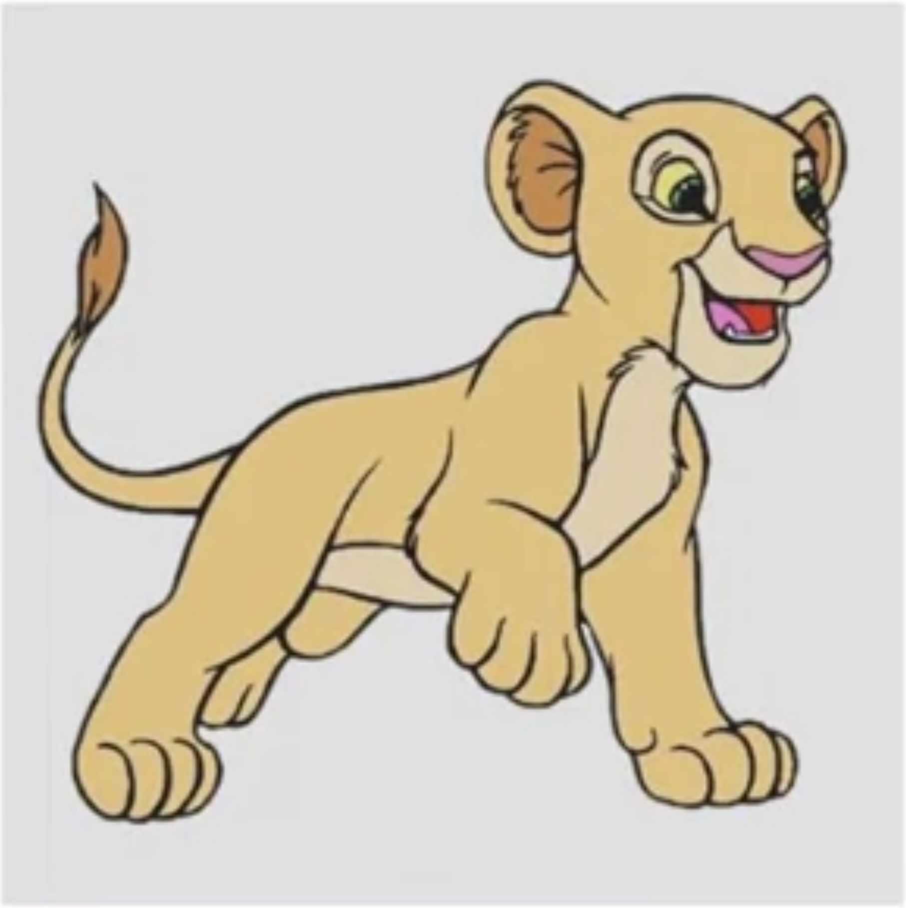
Use of DFT for Filtering





Frequency Filtering

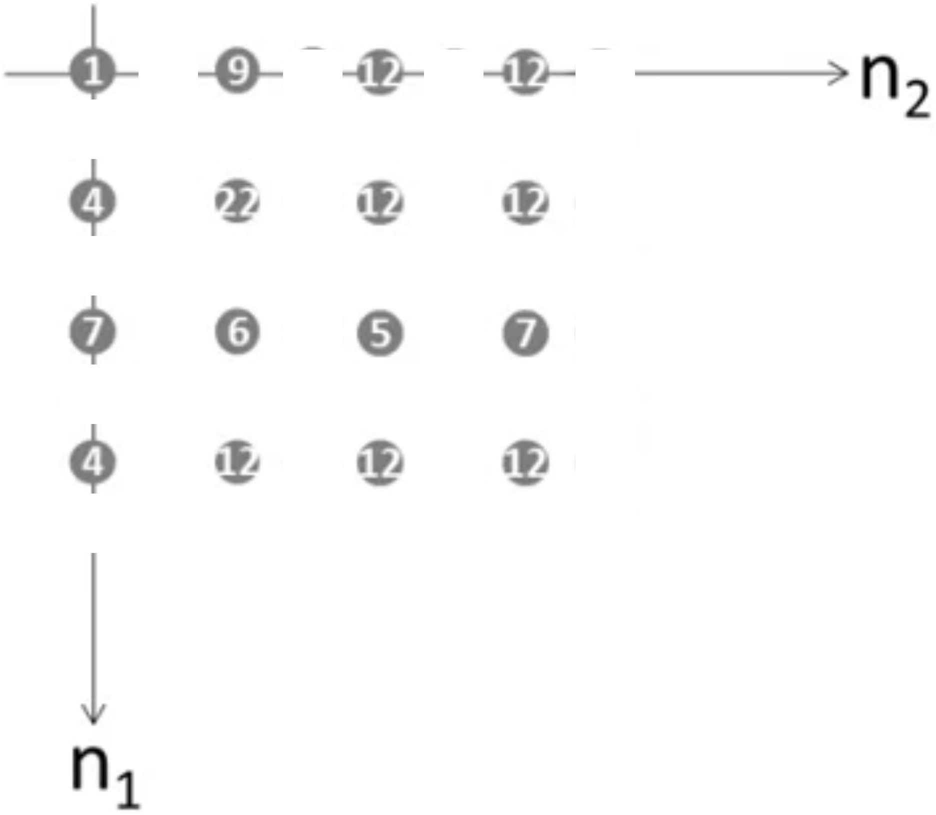
$$y(n_1, n_2) = x(n_1, n_2) + \cos(0.1\pi n_2)$$

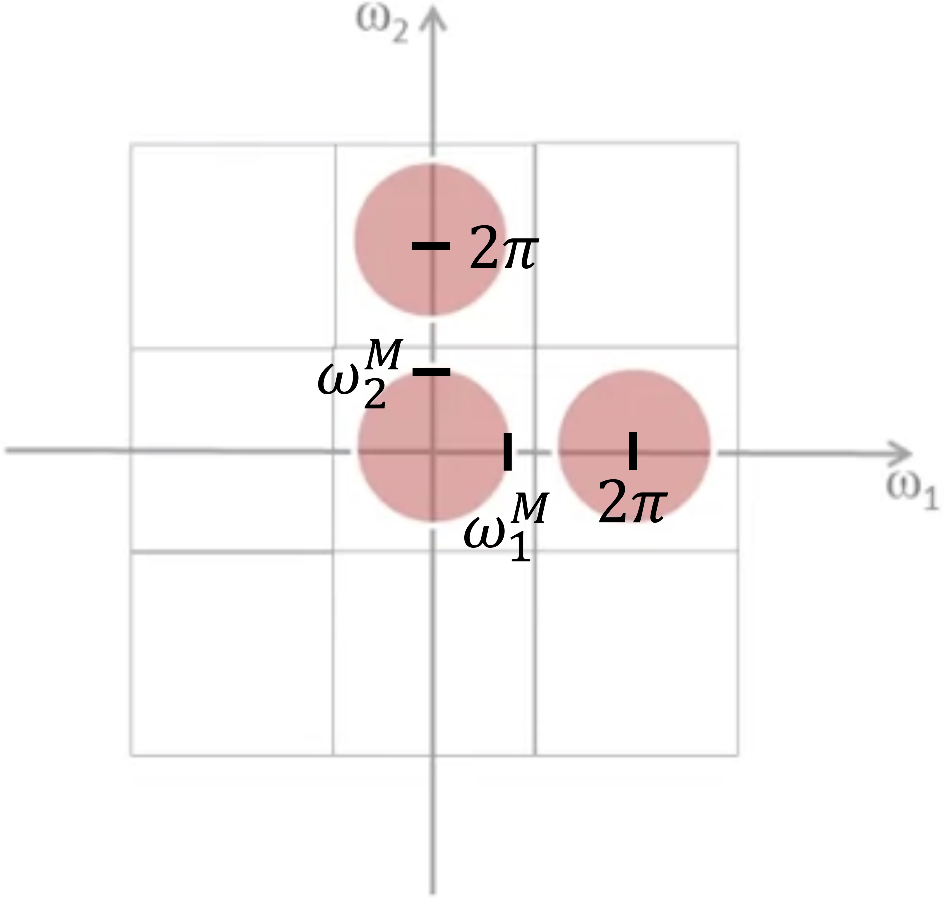


Fourier Transform and Sampling

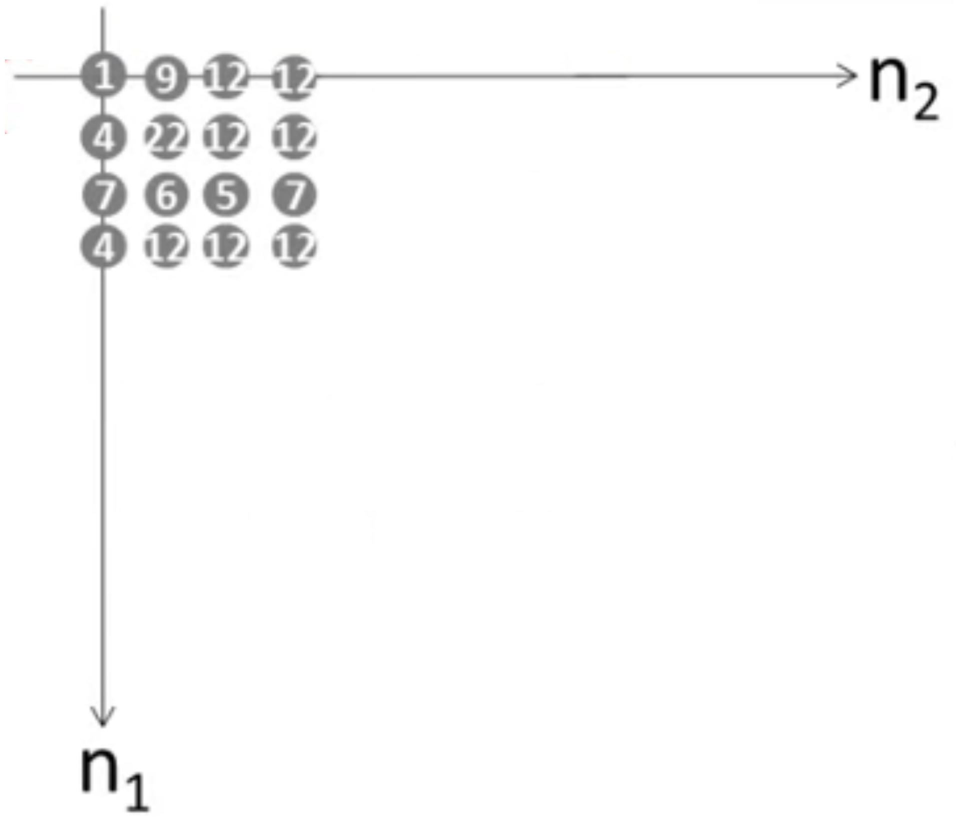
- 2D Fourier Transform
- Sampling
- Discrete Fourier Transform
- Filtering in the Frequency Domain
- **Change of Sampling Rate**
- The Discrete Cosine Transform
- The Two-Dimensional Discrete Cosine Transform

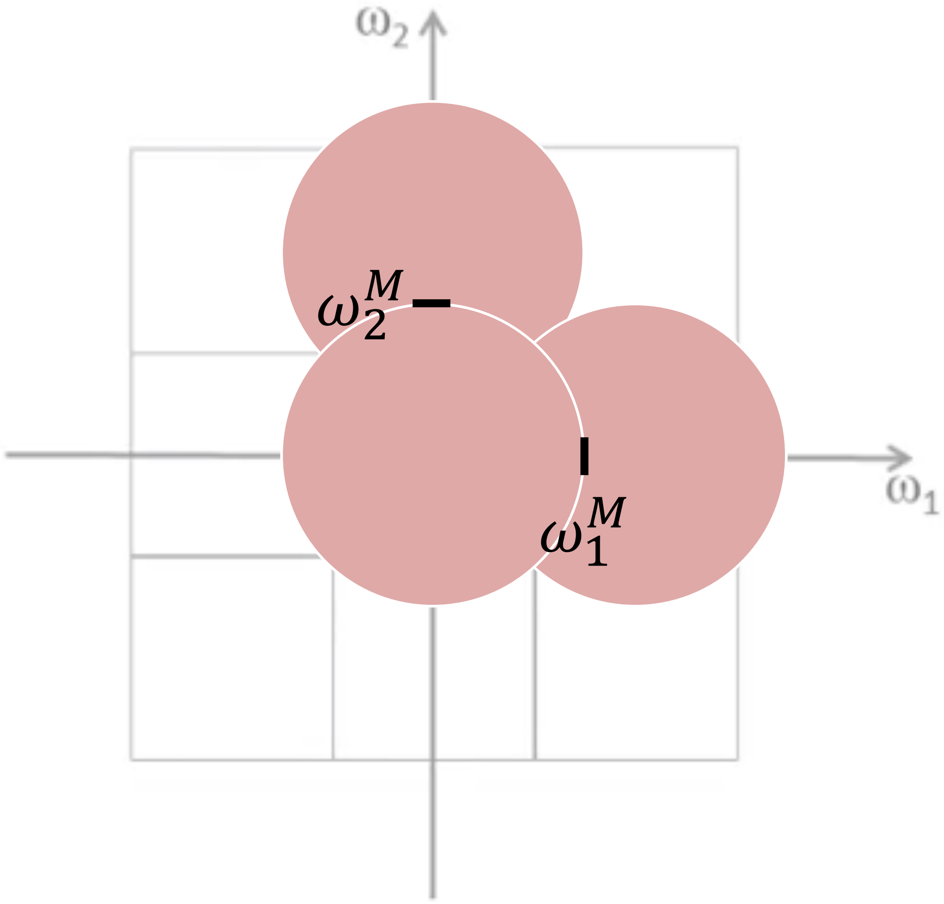
Down-Sampling





Down-Sampling

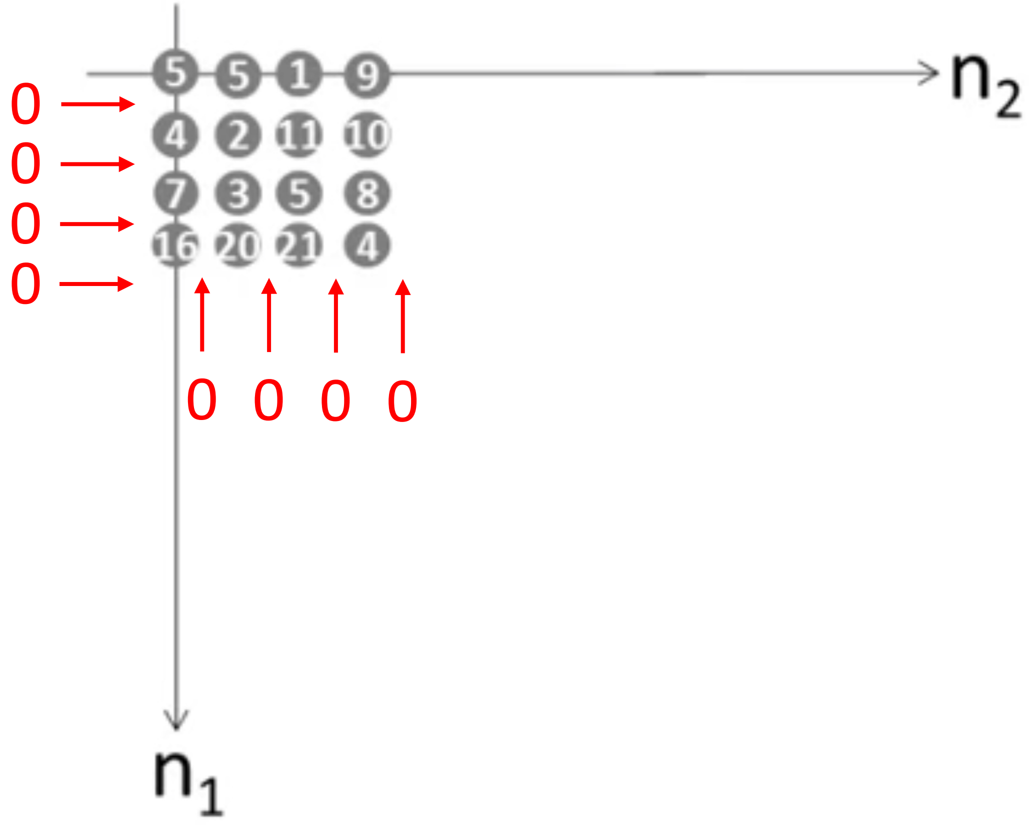




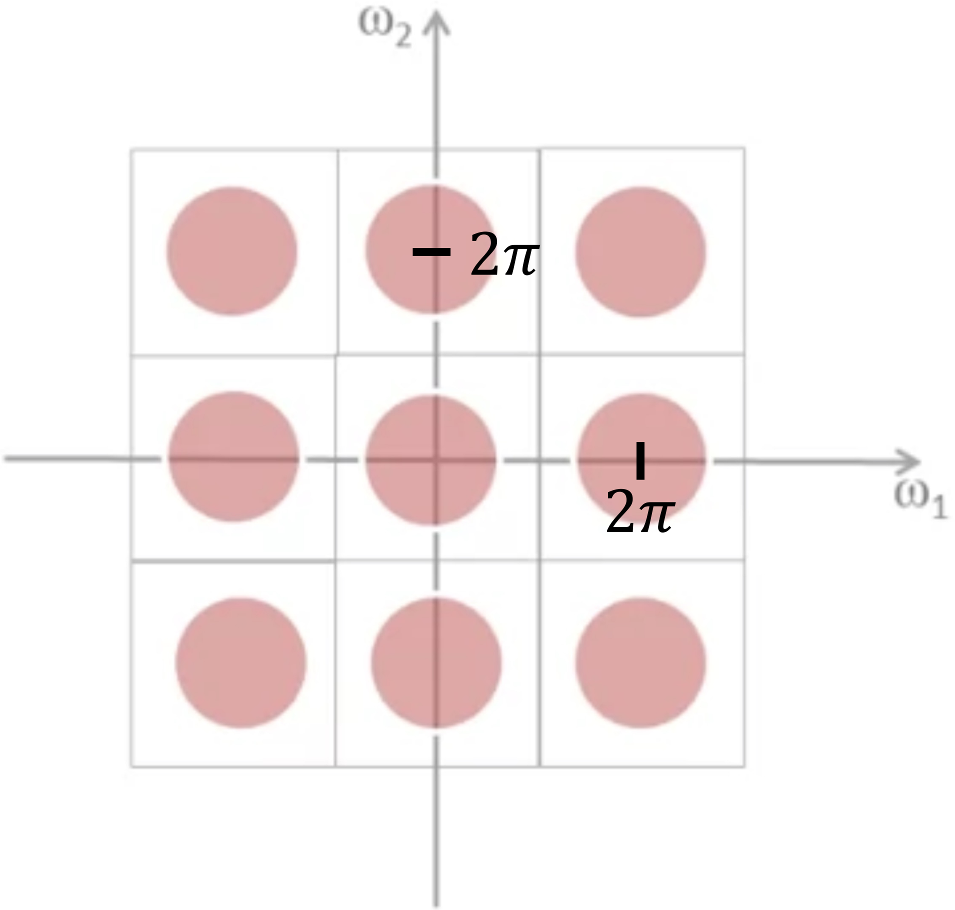
Down-Sampling Example

Resolution of original image: 4200 x 3000 pixels (not shown due to space limitations)
Left image: Down-sampled by a factor of 10, direct pixel removal.
Right image: LPF by a Gaussian 11x11 filter, then down-sampled by a factor of 10

Up-Sampling

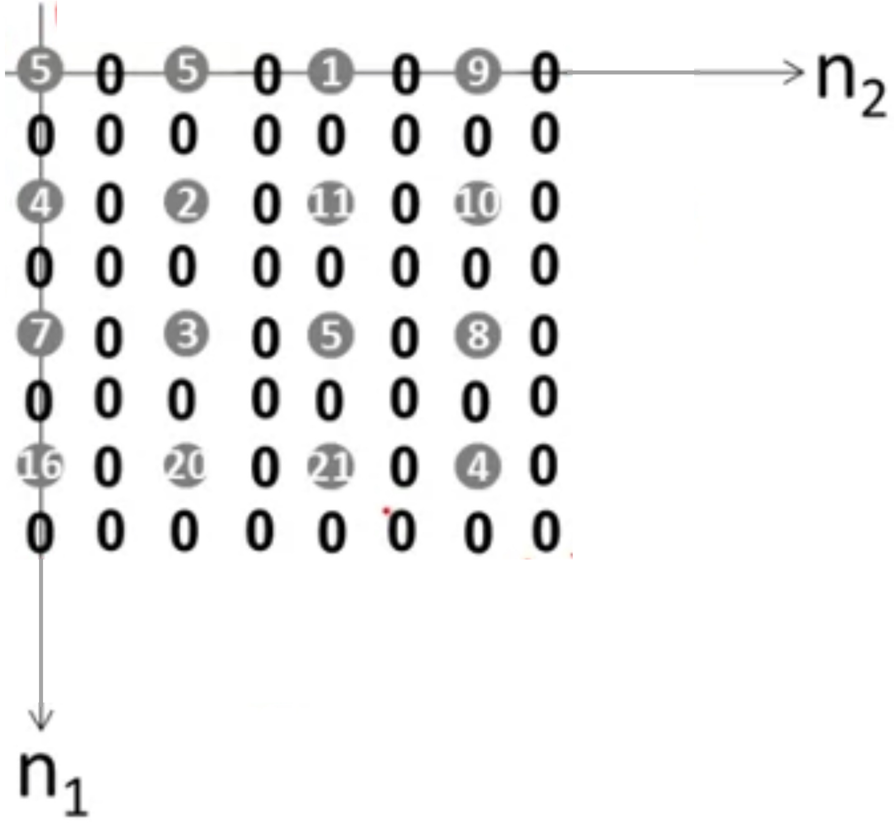


5 5 1 9
4 2 11 10
7 3 5 8
16 20 21 4
0
0
0
0
0 0 0 0
n_2
n_1

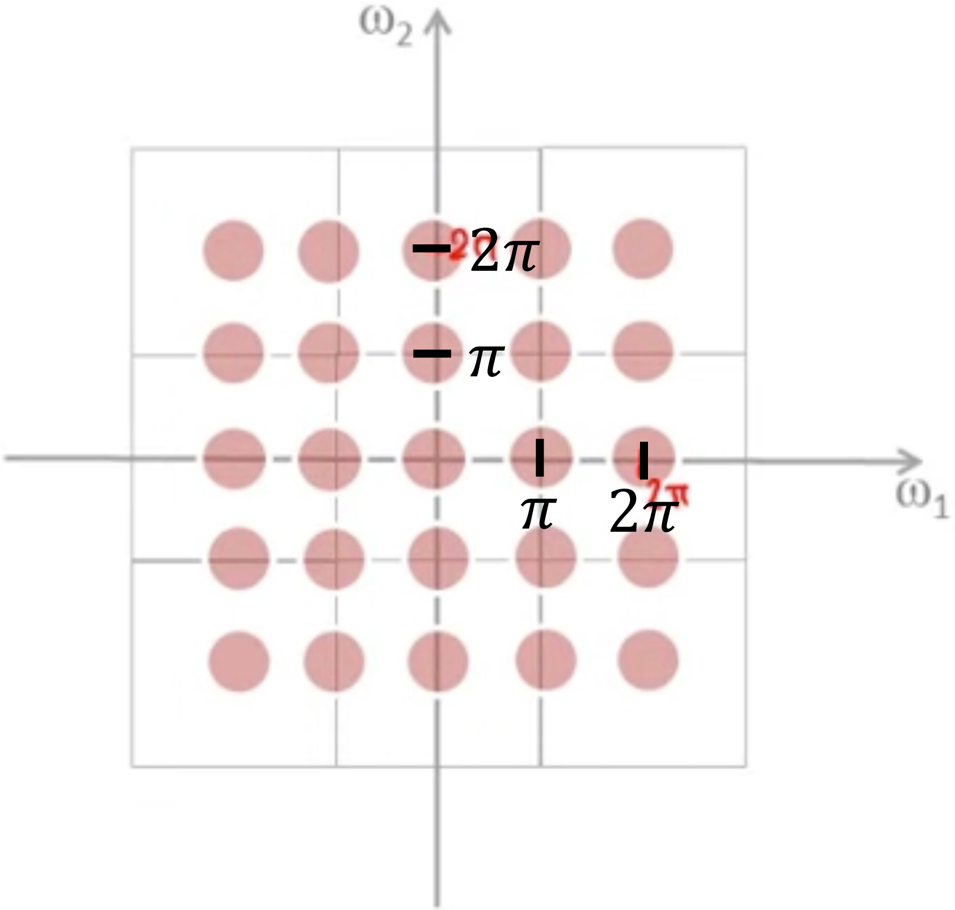


ω_2
ω_1
2π
2π

Up-Sampling

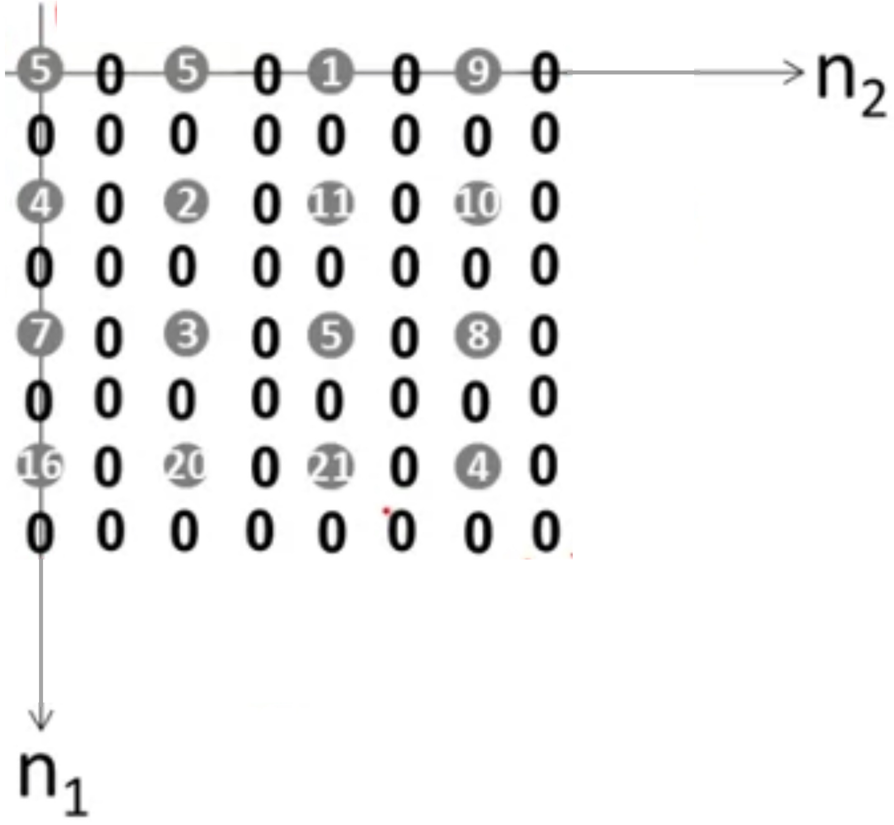


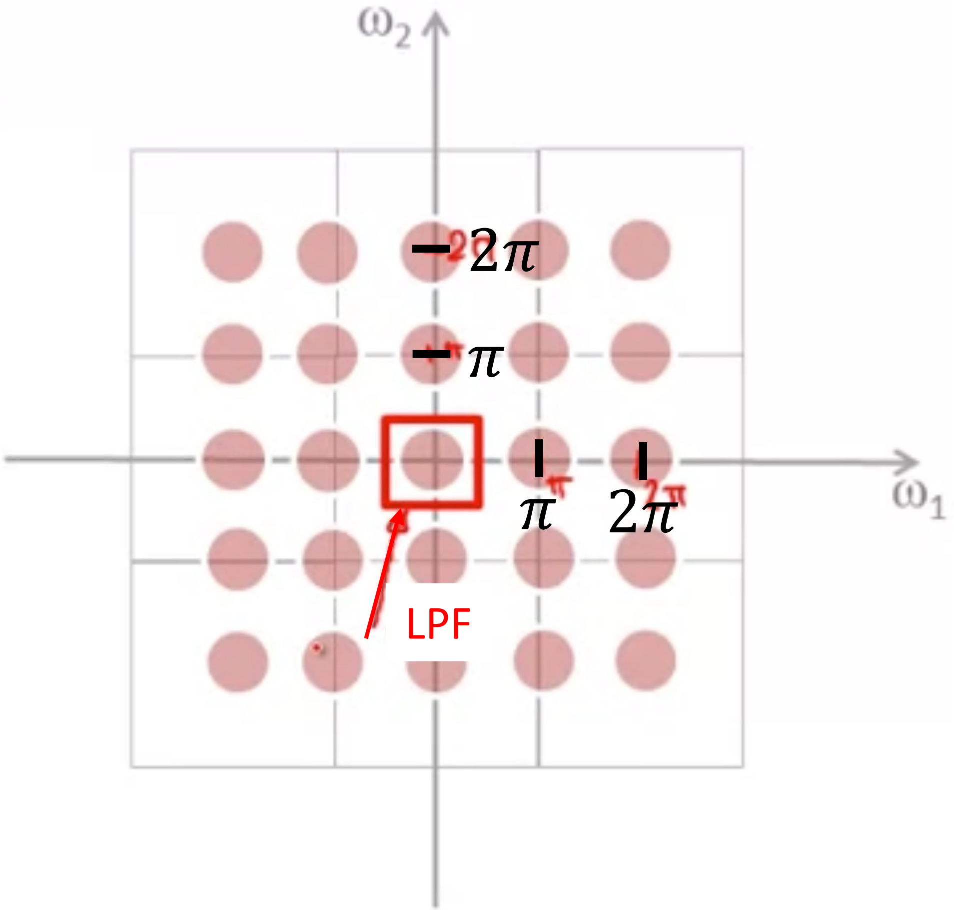
5 0 5 0 1 0 9 0
0 0 0 0 0 0 0 0
4 0 2 0 11 0 10 0
0 0 0 0 0 0 0 0
7 0 3 0 5 0 8 0
0 0 0 0 0 0 0 0
16 0 20 0 21 0 4 0
0 0 0 0 0 0 0 0
n_2
n_1



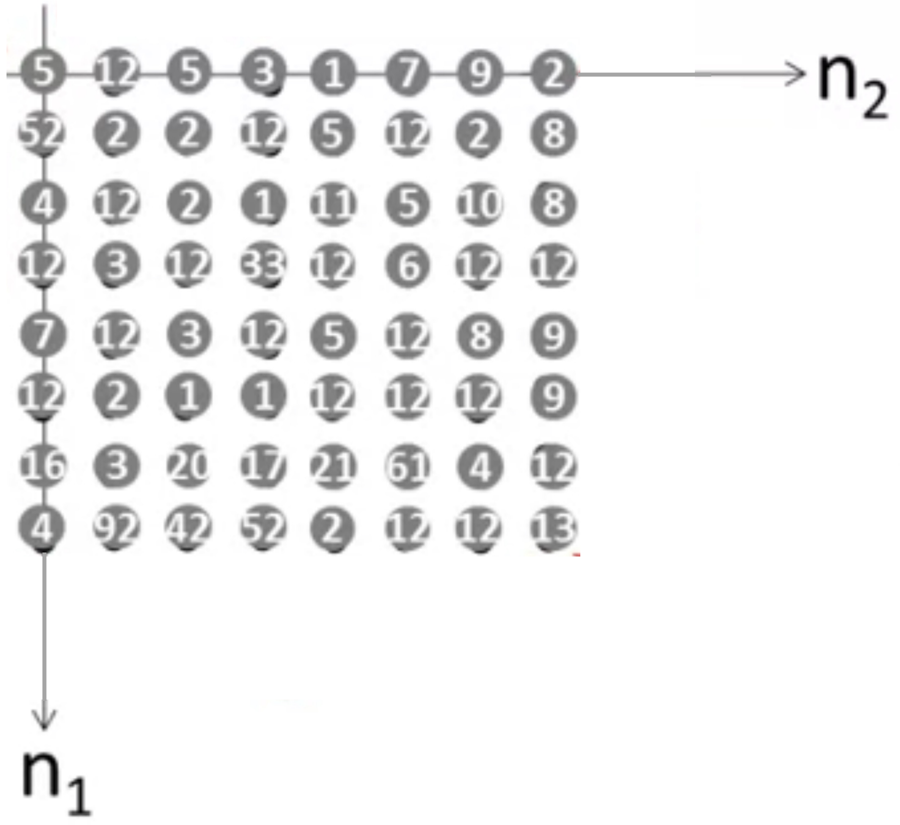
ω_2
2π
π
π
2π
ω_1

Up-Sampling

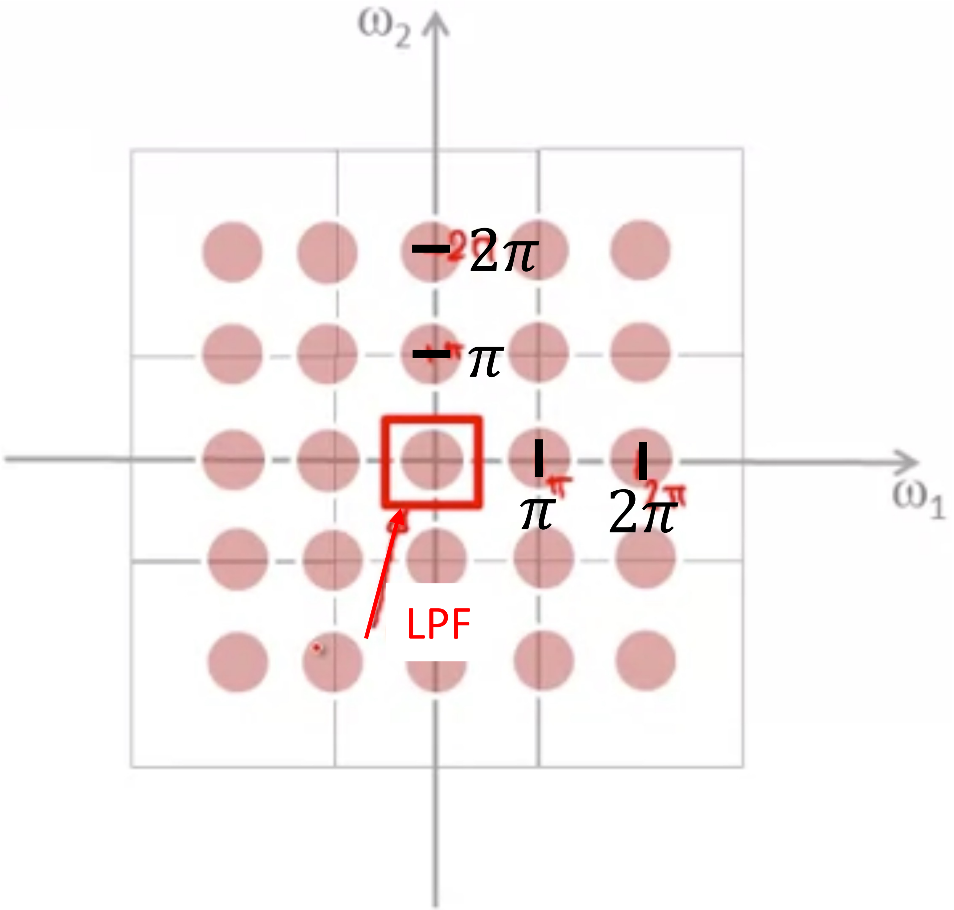

5 0 5 0 1 0 9 0
0 0 0 0 0 0 0 0
4 0 2 0 11 0 10 0
0 0 0 0 0 0 0 0
7 0 3 0 5 0 8 0
0 0 0 0 0 0 0 0
16 0 20 0 21 0 4 0
0 0 0 0 0 0 0 0
n_2
n_1


ω_2
2π
π
π
2π
ω_1
LPF

Up-Sampling



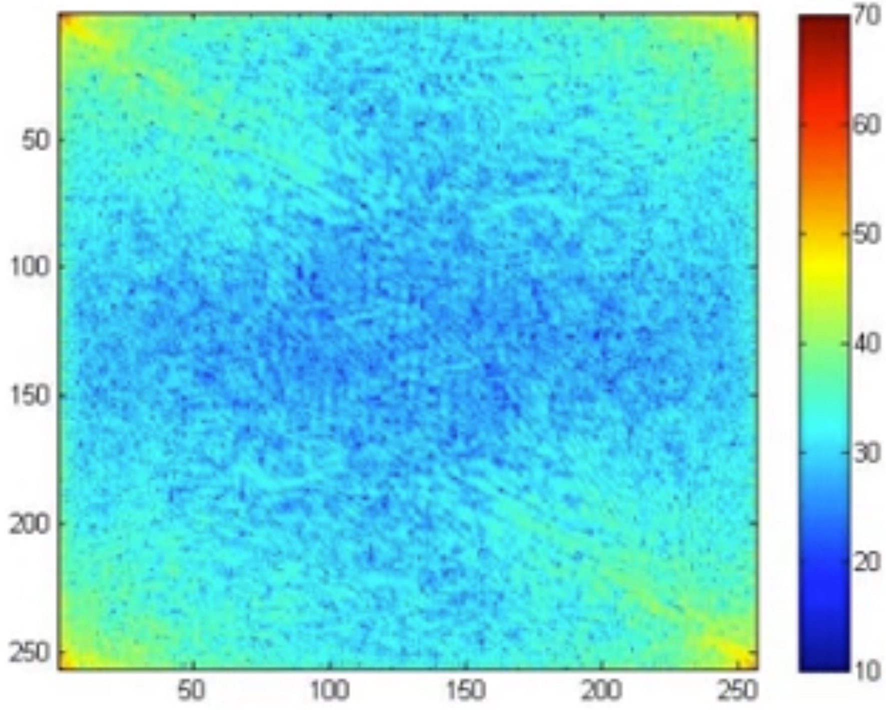
5 12 5 3 1 7 9 2
52 2 2 12 5 12 2 8
4 12 2 1 11 5 10 8
12 3 12 33 12 6 12 12
7 12 3 12 5 12 8 9
12 2 1 1 12 12 12 9
16 3 20 17 21 61 4 12
4 92 42 52 2 12 12 13
n_2
n_1



ω_2
2π
π
π
2π
ω_1
LPF

Up-Sampling Example

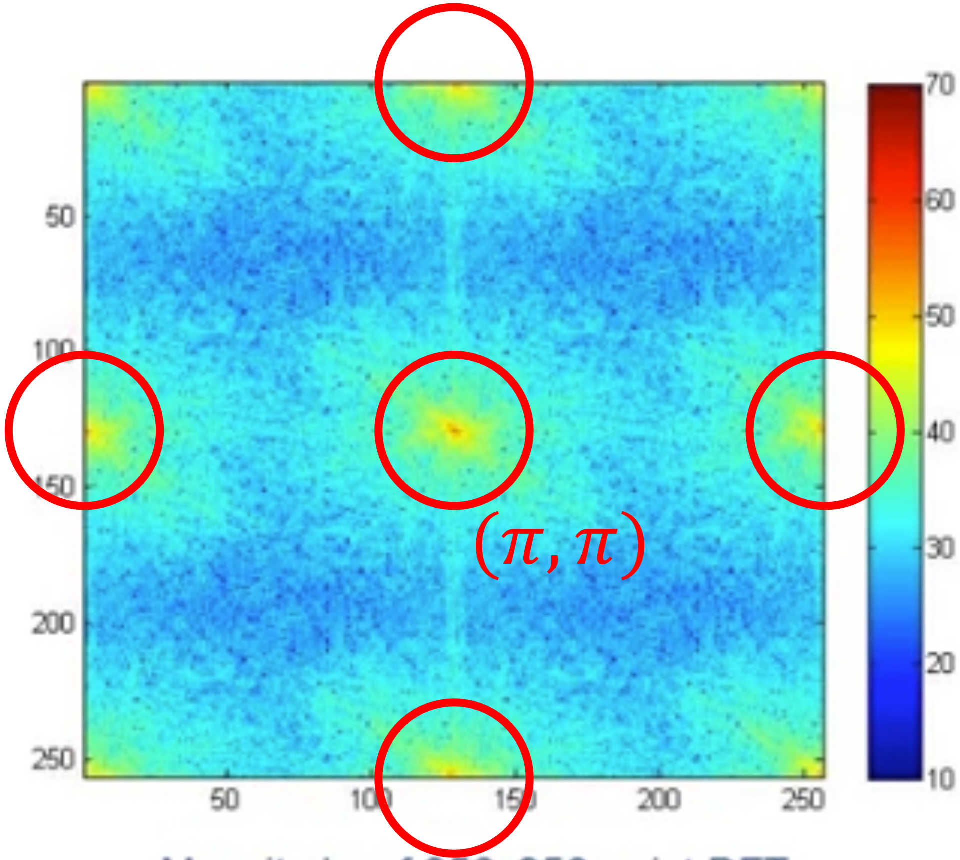
128x128 input image



Magnitude of the 256x 256-point DFT, obtained by zero padding the input image to size 256x256 and then taking the DFT

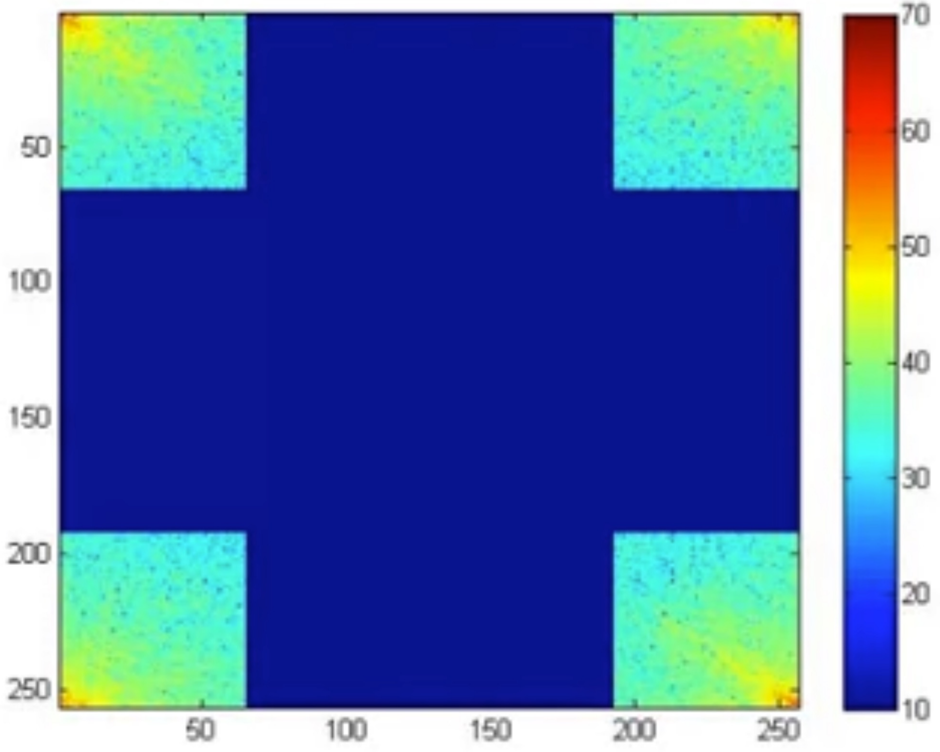
Up-Sampling Example

256x256 image obtained by inserting zero-columns and zero-rows to the original 128x128 image



Magnitude of 256x256-point DFT

Up-Sampling Example



LPF spectrum (blue area corresponds to zero values)

Inverse DFT

Up-Sampling Example

$\frac{a+b}{2}$

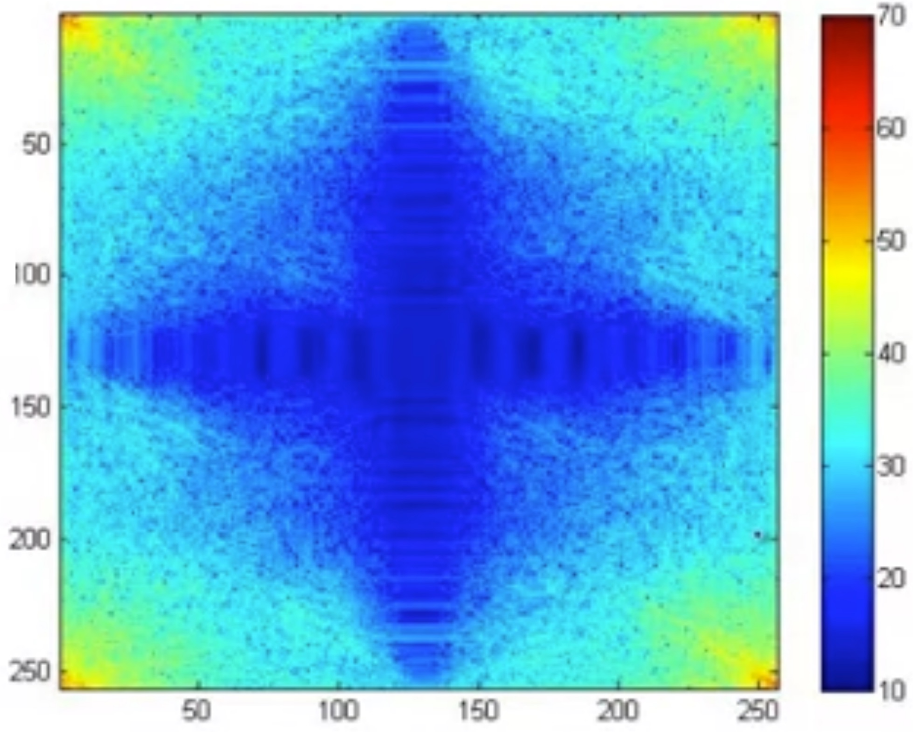
a b

c d

$\frac{a+b+c+d}{4}$

Up-sampled 256x256 image, with filter

0.25	0.5	0.25
0.5	1	0.5
0.25	0.5	0.25



Spectrum

$$H(\omega_1, \omega_2) = 1 + \cos\omega_1 + \cos\omega_2 + \frac{1}{2}\cos(\omega_1 + \omega_2) + \frac{1}{2}\cos(\omega_1 - \omega_2)$$

Fourier Transform and Sampling

- 2D Fourier Transform
- Sampling
- Discrete Fourier Transform
- Filtering in the Frequency Domain
- Change of Sampling Rate
- **The Discrete Cosine Transform**
- The Two-Dimensional Discrete Cosine Transform

In general --

$x[n] = \sum_{k=0}^{N-1} b_k c_k[n]\,, \quad n = 0,1,2,\ldots,N$

b_k: coefficients $\quad c_k[n]$: "basis function"

DFT – complex sinusoids $c_k[n] = e^{j2\pi\left(\frac{k}{N}\right)n}$

Why?

LSI systems

$$e^{j\omega n} \longrightarrow \boxed{h(n)} \longrightarrow H(\omega)e^{j\omega n}$$

$$x[n] = \sum_{k=0}^{N-1} c_k e^{j\omega n} \longrightarrow \boxed{\text{LSI}} \longrightarrow y[n] = \sum_{k=0}^{N-1} H(\omega) c_k e^{j\omega n}$$

Output coefficients : product of input coefficients and frequency response

Filtering!

Compression

$$x[n] = \sum_{k=0} b_k c_k[n], \qquad n = 0, 1, \dots, N-1$$

If most $b_k \approx 0$, store / transmit large ones

$x[n]$: N values. If L, b_k are significant, save factor $\frac{N}{L}$

JPEG, MP3, MPEG, etc.

Discrete Cosine Transform

$$x[n] = \frac{1}{N} X^c[0] + \frac{1}{N} 2X^c[k] \cos(2\pi \left(\frac{k + 1/2}{2N}\right) n) , \qquad n = 0,1, \dots , N - 1$$

$$b_0 = \frac{1}{N} X^c[0], c_0[n] = 1 \quad b_k = \frac{1}{N} 2X^c[k], c_k[n] = \cos(2\pi \left(\frac{k + 1/2}{2N}\right) n)$$

$$X^c[k] = \sum_{n=0}^{N-1} x[n] \cos(2\pi (\frac{k}{2N})(n + 1/2))$$

$X^c[k]$ real!

$$\cos(2\pi \hat{f} n) \longrightarrow \boxed{\text{LSI}} \longrightarrow \left|H(\hat{f})\right| \cos(2\pi \hat{f} n + \phi(\hat{f})) \neq A(\hat{f}) \cos(2\pi \hat{f} n)$$

Fourier Transform and Sampling

- 2D Fourier Transform
- Sampling
- Discrete Fourier Transform
- Filtering in the Frequency Domain
- Change of Sampling Rate
- The Discrete Cosine Transform
- **The Two-Dimensional Discrete Cosine Transform**

Objectives

- Interpret 2D-DCT: represent image as a sum of basis images
- Compression: set some DCT coefficients to zeros
- JPEG

Two-dimensional DCT

$$x[m,n] = \frac{1}{N}\sum_{k=0}^{N-1}\sum_{l=0}^{N-1} a_k a_l X^c[k,l] \underbrace{\cos\left(2\pi(\frac{l}{2N})(n+\frac{1}{2})\right)\cos\left(2\pi(\frac{k}{2N})(m+\frac{1}{2})\right)}_{c_{k,l}[m,n]}$$

$$a_i = \begin{cases} 1 & i = 0 \\ \sqrt{2} & \text{otherwise} \end{cases}$$

$$x[m,n] = \sum_{k=0}^{N-1}\sum_{l=0}^{N-1} b_{k,l} c_{k,l}[m,n]$$

Weighted sum of “basis” images

$c_{k,l}[m,n]$: Product of a vertical (frequency $\frac{k}{2N}$) and horizontal (frequency $\frac{l}{2N}$) cosine

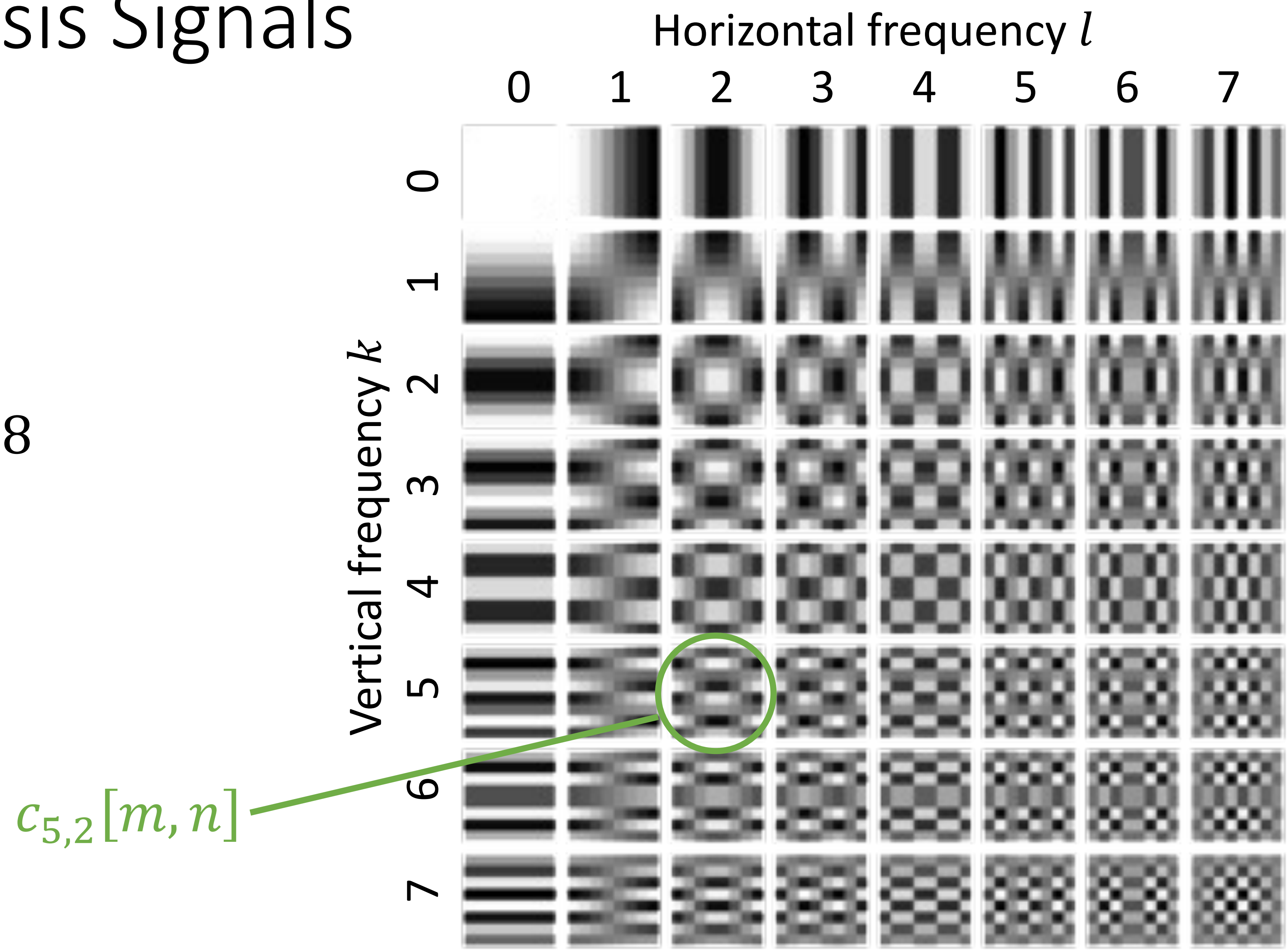
2D DCT Basis Signals

$c_{k,l}[m,n]$

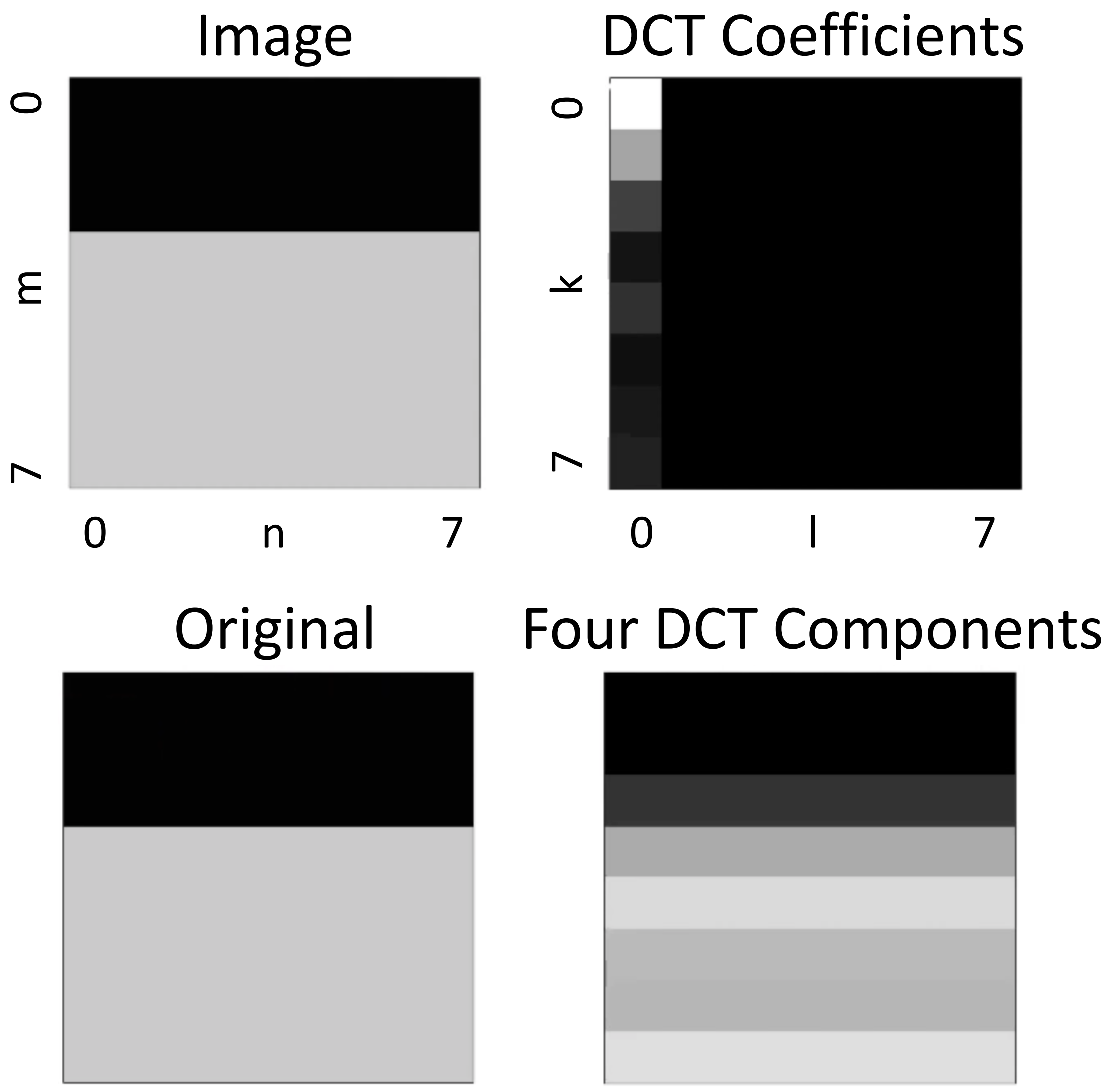
$N = 8$

(8×8 image)

Each entry is 8×8



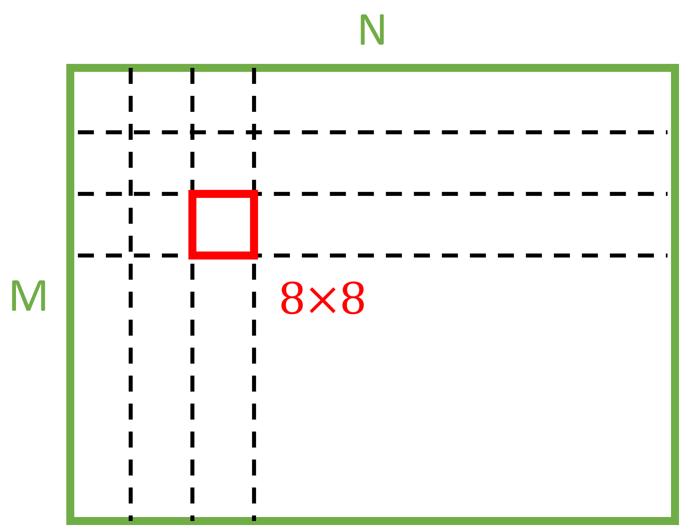
Example: Horizontal Edge



Compression
by factor
64/4=16

JPEG Image Compression

1. Break up $M \times N$ image into 8×8 blocks



2. Take 2D-DCT of each block to find DCT coefficients
3. Save important/significant DCT coefficients (eye is less sensitive to high spatial frequencies)
   - Allocate bits to each coefficients based on its significance/importance
   - More bits -> more accurate coefficient

JPEG Decoding

JPEG file has bits for each $8{\times}8$ block

1. Read bits for block
2. Convert bits to DCT coefficients
3. Take inverse DCT to find $8{\times}8$ image
4. Put blocks together

Two-dimensional DCT & IDCT

$$F(u,v) = \frac{2}{N} C(u) C(v) \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x,y) \cos\frac{(2x+1)u\pi}{2N} \cos\frac{(2y+1)v\pi}{2N}$$

$$f(x,y) = \frac{2}{N} \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} C(u) C(v) F(u,v) \cos\frac{(2x+1)u\pi}{2N} \cos\frac{(2y+1)v\pi}{2N}$$

$$\text{Where } C(u), C(v) = \begin{cases} \frac{1}{\sqrt{2}} & \text{for } u, v = 0 \\ 1 & \text{otherwise} \end{cases}$$

$$0 \le x, y, u, v \le N-1, N^2\text{: frame size}$$

One-dimensional DCT & IDCT

$$F(u,v) = \sqrt{\frac{2}{N}} C(u) \sum_{x=0}^{N-1} f(x) \cos \frac{(2x+1)u\pi}{2N}$$

$$f(x,y) = \sqrt{\frac{2}{N}} \sum_{u=0}^{N-1} C(u)F(u) \cos \frac{(2x+1)u\pi}{2N}$$

$$\text{Where } C(u) = \begin{cases} \frac{1}{\sqrt{2}} \text{ for } u = 0 \\ 1 \quad \text{otherwise} \end{cases}$$

$$0 \le x, u \le N-1, N^2\text{: frame size}$$

Fourier Transform and Sampling

- 2D Fourier Transform
- Sampling
- Discrete Fourier Transform
- Filtering in the Frequency Domain
- Change of Sampling Rate
- The Discrete Cosine Transform
- The Two-Dimensional Discrete Cosine Transform

Homework #2 – 2D-DCT

- 2D-DCT
  - Implement 2D-DCT to transform “lena.png” to DCT coefficients (visualize in log domain).
    - Convert the input image to gray-scale first.
    - Visualize the coefficients in log domain. Feel free to scale and clip the coefficients for visualization.
  - Implement 2D-IDCT to reconstruct the image.
  - Evaluate the PSNR.
- Two 1D-DCT
  - Implement a fast algorithm by two 1D-DCT to transform “lena.png ” to DCT coefficients.
- Compare the runtime between 2D-DCT and two 1D-DCT.
- Do **not** use any functions for DCT and IDCT, e.g., cv2.dct
  - Although, you can still use these functions to validate your output.
- Deadline: 2023/10/23 11:59 PM
- Upload to E3 with required files :
  - **VC_HW2_[student_id].pdf**: Report PDF
  - **VC_HW2_[student_id].zip**: Zipped source code (C/C++/Python/MATLAB)

Two-dimensional DCT & IDCT

$$F(u,v) = \frac{2}{N} C(u)C(v) \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x,y) \cos\frac{(2x+1)u\pi}{2N} \cos\frac{(2y+1)v\pi}{2N}$$

$$f(x,y) = \frac{2}{N} \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} C(u)C(v)F(u,v) \cos\frac{(2x+1)u\pi}{2N} \cos\frac{(2y+1)v\pi}{2N}$$

$$\text{Where } C(u), C(v) = \begin{cases} \frac{1}{\sqrt{2}} & \text{for } u, v = 0 \\ 1 & \text{otherwise} \end{cases}$$

$$0 \le x, y, u, v \le N-1, N^2\text{: frame size}$$

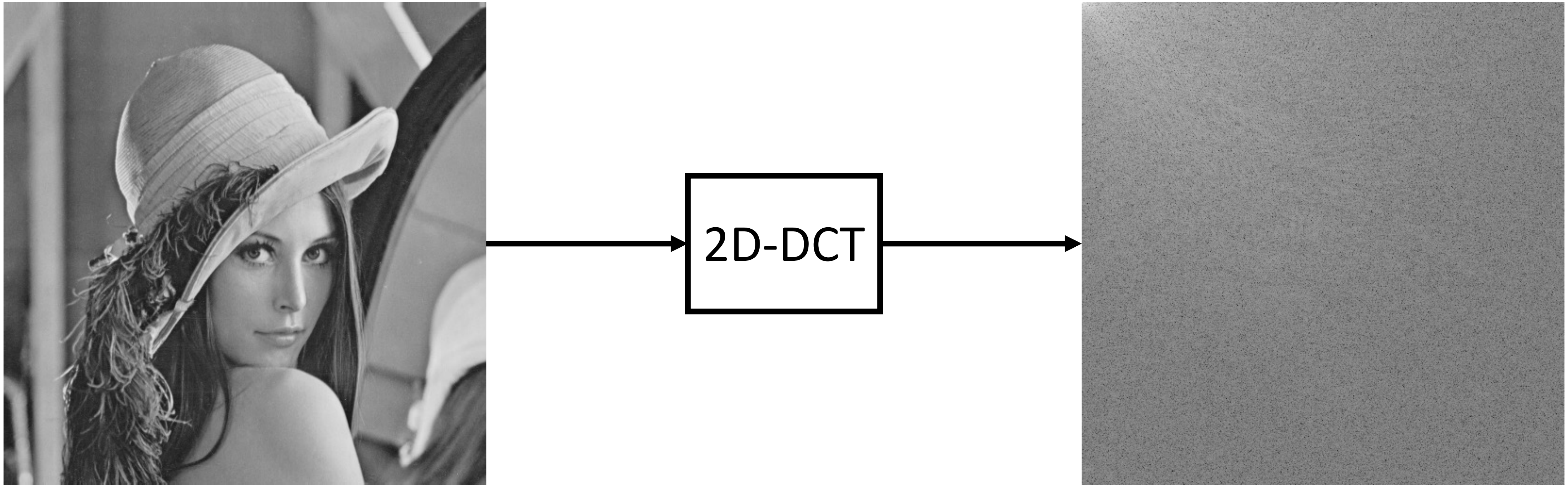
One-dimensional DCT & IDCT

$$F(u,v) = \sqrt{\frac{2}{N}}C(u)\sum_{x=0}^{N-1} f(x)\cos\frac{(2x+1)u\pi}{2N}$$

$$f(x,y) = \sqrt{\frac{2}{N}}\sum_{u=0}^{N-1} C(u)F(u)\cos\frac{(2x+1)u\pi}{2N}$$

$$\text{Where } C(u) = \begin{cases} \frac{1}{\sqrt{2}} & \text{for } u = 0 \\ 1 & \text{otherwise} \end{cases}$$

$$0 \leq x, u \leq N-1, N^2\text{: frame size}$$


2D-DCT