Digital Image Processing

Lecture #8

Ming-Sui (Amy) Lee

Announcement

- **The following schedule**

02/17	Lecture 1	04/14	Lecture 8
02/24	Lecture 2	04/21	Proposal
03/03	Lecture 3	04/28	Lecture 9
03/10	Lecture 4	05/05	Lecture 10
03/17	Lecture 5	05/12	Lecture 11
03/24	Lecture 6	05/19	Demo
03/31	Lecture 7	05/26	Demo
04/07	Midterm	06/02	Final Package Due

- **Midterm Exam**
  - **Apr. 07, 2022**

Announcement

- Midterm
  - Apr. 07, 2022 @ 102
  - Closed-book exam
  - Photo/Student ID
  - One-line calculator

- Proposal
  - Apr. 21, 2022
  - Please form a team with 2 students
  - Email TA the member list by Apr. 08

    Update the member list on NTU COOL by Apr. 09

Announcement

- Proposal
  - Apr. 21, 2022
  - Check the list announced on NTU COOL
  - Oral presentation (PPT) AND Written report (PDF)
    - Paper title
    - Motivation
    - Problem definition
    - Algorithm
    - Expected results
    - Reference

Image Sampling & Transforms

Image Sampling & Transforms

- **Image sampling**
  - **A/D conversion**
    - **Usually deals with arrays of numbers obtained by spatially sampling points of a physical image**
    - $F_I(x,y)$ **: continuous, infinite-extent, ideal image field**
    - $F_P(x,y)$ **: the sampled image**

$$F_I(x,y) \longrightarrow \bigotimes \longrightarrow F_P(x,y)$$

$$S(x,y) = \sum_{j=-\infty}^{\infty} \sum_{k=-\infty}^{\infty} \delta(x - j\Delta x, y - k\Delta y)$$

Dirac delta function

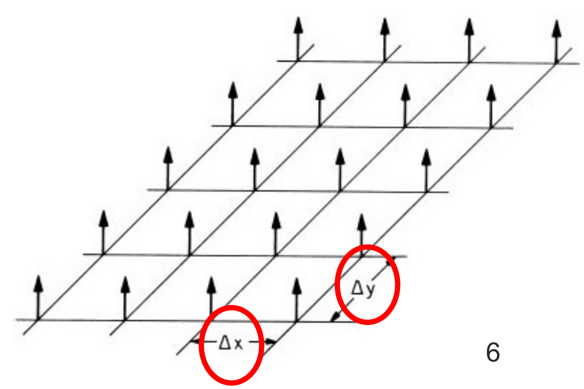


Image Sampling & Transforms

- **Image sampling**
  - **Sampling function** $S(x,y)=\sum_{j=-\infty}^{\infty}\sum_{k=-\infty}^{\infty}\delta(x-j\Delta x, y-k\Delta y)$
    - **An infinite array of the Dirac delta functions arranged in a grid of spacing,** $(\Delta x, \Delta y)$
  - **Space domain**

$$F_P(x,y)=F_I(x,y)S(x,y)=\sum_{j=-\infty}^{\infty}\sum_{k=-\infty}^{\infty}F_I(j\Delta x,k\Delta y)\delta(x-j\Delta x, y-k\Delta y)$$

  - **Transform domain (Fourier transform)**

(Convolution Theorem)

$$\mathcal{F}_P(\omega_x,\omega_y)=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}F_P(x,y)\exp\{-i(\omega_x x+\omega_y y)\}dxdy=\frac{1}{4\pi^2}\mathcal{F}_I(\omega_x,\omega_y)\otimes S(\omega_x,\omega_y)$$

Sampling frequency

where $S(\omega_x,\omega_y)=\frac{4\pi^2}{\Delta x\Delta y}\sum_{j=-\infty}^{\infty}\sum_{k=-\infty}^{\infty}\delta(\omega_x-j\omega_{xs},\omega_y-k\omega_{ys})$

and $\omega_{xs}=2\pi/\Delta x \quad and \quad \omega_{ys}=2\pi/\Delta y$

Image Sampling & Transforms

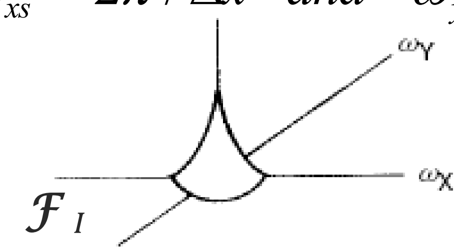
- **Image sampling**
  - **Transform domain (Fourier transform)**
    - **Assume the spectrum of the ideal image is band-limited**

$$\mathcal{F}_P(\omega_x,\omega_y)=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}F_P(x,y)\exp\{-i(\omega_x x+\omega_y y)\}dxdy=\frac{1}{4\pi^2}\mathcal{F}_I(\omega_x,\omega_y)\otimes S(\omega_x,\omega_y)$$

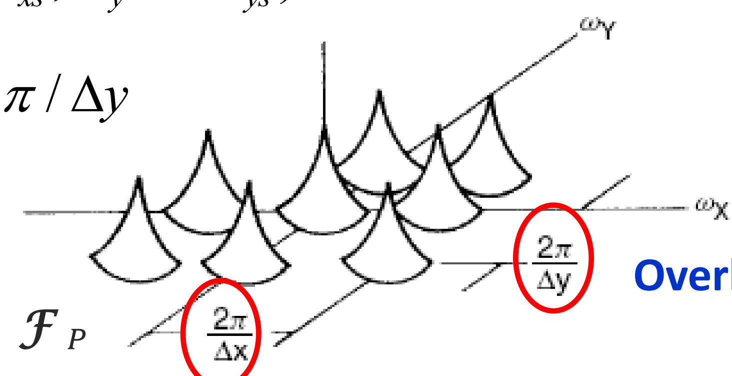
$$=\frac{1}{\Delta x\Delta y}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\mathcal{F}_I(\omega_x-\alpha,\omega_y-\beta)\times\sum_{j=-\infty}^{\infty}\sum_{k=-\infty}^{\infty}\delta(\alpha-j\omega_{xs},\beta-k\omega_{ys})d\alpha d\beta$$

$$=\frac{1}{\Delta x\Delta y}\sum_{j=-\infty}^{\infty}\sum_{k=-\infty}^{\infty}\mathcal{F}_I(\omega_x-j\omega_{xs},\omega_y-k\omega_{ys})$$

where $\omega_{xs}=2\pi/\Delta x \quad and \quad \omega_{ys}=2\pi/\Delta y$



(a) Original image

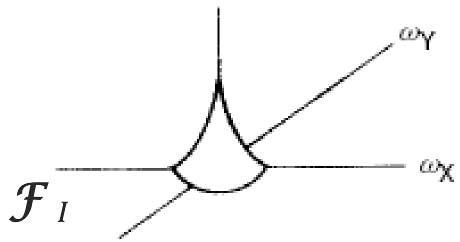


Overlapping?

(b) Sampled image

Image Sampling & Transforms

- Image sampling
  - Transform domain (Fourier transform)
    - Assume the spectrum of the ideal image is limited and suppose that the cutoff frequency is $(\omega_{xc}, \omega_{yc})$
      i.e. $\mathcal{F}_I(\omega_x, \omega_y)$ is non-zero only in the region bounded by $|\omega_x| \leq \omega_{xc}$ $|\omega_y| \leq \omega_{yc}$



(a) Original image

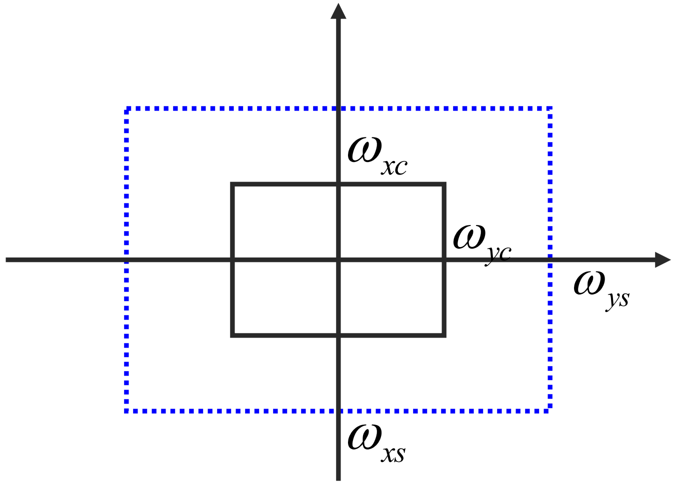


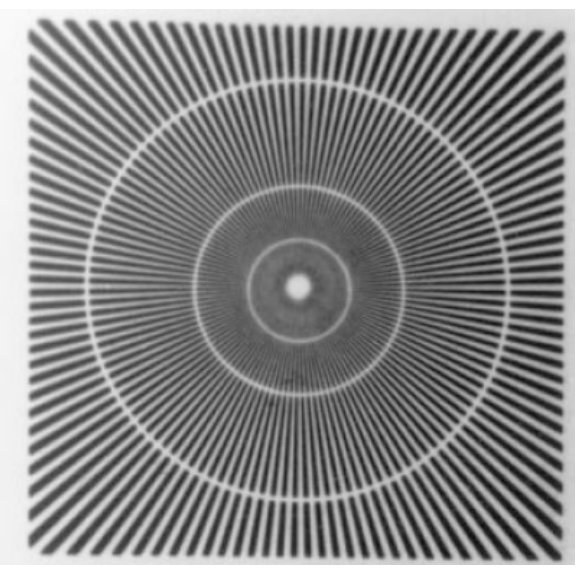
Image Sampling & Transforms

- Image sampling
  - Transform domain (Fourier transform)
    - If $\omega_{xc} \le \dfrac{\omega_{xs}}{2}, \quad \omega_{yc} \le \dfrac{\omega_{ys}}{2}$,

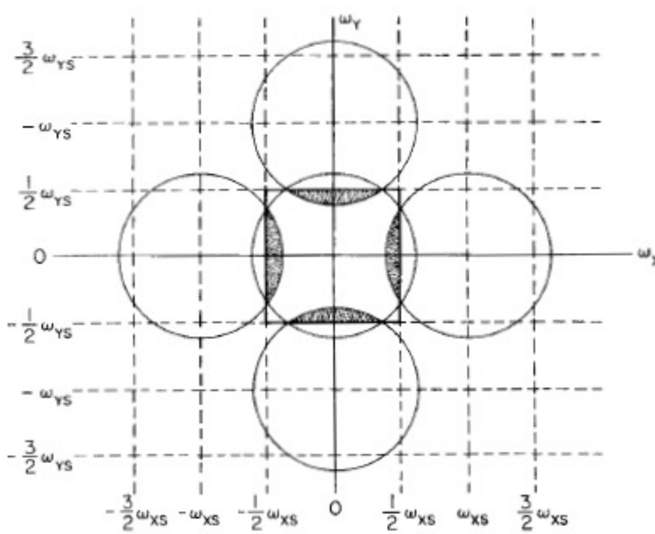
      there is no overlapping (no aliasing) between adjacent shifted waveforms of $\mathcal{F}_P(\omega_x, \omega_y)$
  - Sampling Theorem
    - To sample a band-limited signal, the sampling period must be no longer than one-half of the period of the finest details within the image to avoid aliasing
    - Generalization of the 1D Nyquist Theorem to the 2D case

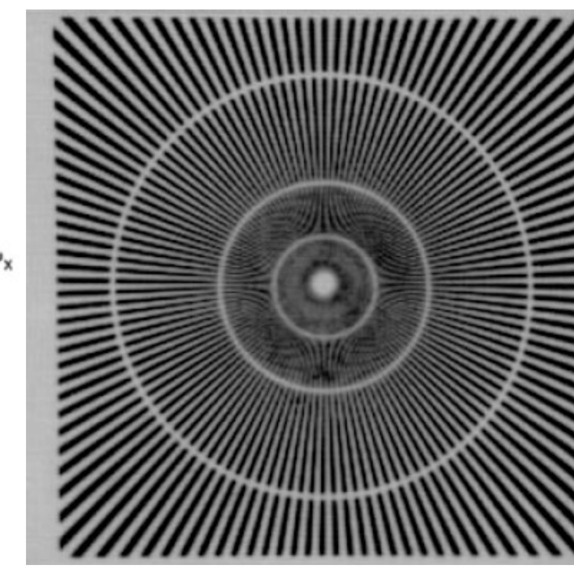
Image Sampling & Transforms

- **Sampling Theorem**
  - **To sample a band-limited signal, the sampling period must be no longer than one-half of the period of the finest details within the image to avoid aliasing**



Original image





Sampled image

Image Sampling & Transforms

- Image Transform
  - Why image transform?
    - Feature extraction
    - Energy compaction
  - We are able to concentrate energy distribution over a small number of transform coefficients
    - Can be exploited for compression purpose
      - DCT – Discrete Cosine Transform
      - Block-based transform – 8x8

Image Sampling & Transforms

- **General 2D transform**
  - **Forward**

$$\underset{0 \le j,k \le N-1}{F(j,k)} \longrightarrow \boxed{\begin{array}{c}\textbf{Transform}\\ \textbf{Kernel}\\ \textbf{A(j,k;u,v)}\end{array}} \longrightarrow \underset{0 \le u,v \le N-1}{\tilde{F}(u,v)}$$

  - **Backward**

$$\underset{0 \le u,v \le N-1}{\tilde{F}(u,v)} \longrightarrow \boxed{\begin{array}{c}\textbf{Inverse Transform}\\ \textbf{Kernel}\\ \textbf{B(u,v;j,k)}\end{array}} \longrightarrow \underset{0 \le j,k \le N-1}{F(j,k)}$$

Image Sampling & Transforms

- **2D separable transform**
  - General 2D transform
    - Forward

$$\tilde{F}(u,v) = \sum_{j=0}^{N-1}\sum_{k=0}^{N-1} F(j,k)A(j,k;u,v)$$

    - Backward

$$F(j,k) = \sum_{u=0}^{N-1}\sum_{v=0}^{N-1} \tilde{F}(u,v)B(u,v;j,k)$$

    - If A and B are separable, we have

$$A(j,k;u,v) = A_X(j,u)A_Y(k,v)$$

$$B(u,v;j,k) = \underbrace{B_X(u,j)}_{\text{x-direction}}\underbrace{B_Y(v,k)}_{\text{y-direction}}$$

Image Sampling & Transforms

- 2D separable transform
  - The real advantage is the low computational cost of the separable transform
    - General 2D transform

$$\tilde{F}(u,v) = \sum_{j=0}^{N-1}\sum_{k=0}^{N-1} F(j,k)A(j,k;u,v)$$

$O(N^2)$ per (u,v) & $O(N^4)$ for all (u,v)

    - Separable 2D transform

$$\tilde{F}(u,v) = \sum_{k=0}^{N-1}\left(\sum_{j=0}^{N-1} F(j,k)A_X(j,u)\right)A_Y(k,v)$$

$2N$ per (u,v) & $2N^3$ for all (u,v)

Image Sampling & Transforms

- **Unitary Transform**
  - **Consider 1D transform**

$$\widetilde{F}(u)=\sum_{j=0}^{N-1}F(j)A(j,u)$$

$$\underline{f}=\begin{bmatrix}F(0)\\F(1)\\\vdots\\F(N-1)\end{bmatrix}\quad \underline{\widetilde{f}}=\begin{bmatrix}\widetilde{F}(0)\\\widetilde{F}(1)\\\vdots\\\widetilde{F}(N-1)\end{bmatrix}\quad A=\begin{bmatrix}A(0,0)&\cdots&A(N-1,0)\\\vdots&\ddots&\vdots\\A(0,N-1)&\cdots&A(N-1,N-1)\end{bmatrix}$$

  - **Forward** $\underline{\widetilde{f}}=A\underline{f}$
  - **Backward** $\underline{f}=B\underline{\widetilde{f}}$

$\Rightarrow B=A^{-1}$

Gaussian elimination $\Rightarrow O(N^3)$

Image Sampling & Transforms

- **Unitary Transform**
  - **The inverse is easy to compute**

$$A^H A = AA^H = I \quad \Rightarrow A^{-1} = A^H$$

$$A^H = (A^*)^T = (A^T)^*$$

- **Karhunen Loeve Transform (KLT)**
  - **Hotelling transform / Eigenvector transform**
  - **Linear transform**
  - **Ideal for energy compaction**
  - **Principal component analysis (PCA)/Singular Value Decomposition (SVD)**
  - **Basis functions are image dependent**
  - **High computation cost for obtaining basis images**

Image Sampling & Transforms

- **Karhunen Loeve Transform (KLT)**
  - **Mean and covariance matrix**

$$\underline{m}_x = E\{\underline{x}\} = [m_1 \quad m_2 \quad \ldots \quad m_n]^T = [E\{x_1\} \quad E\{x_2\} \quad \ldots \quad E\{x_n\}]^T$$

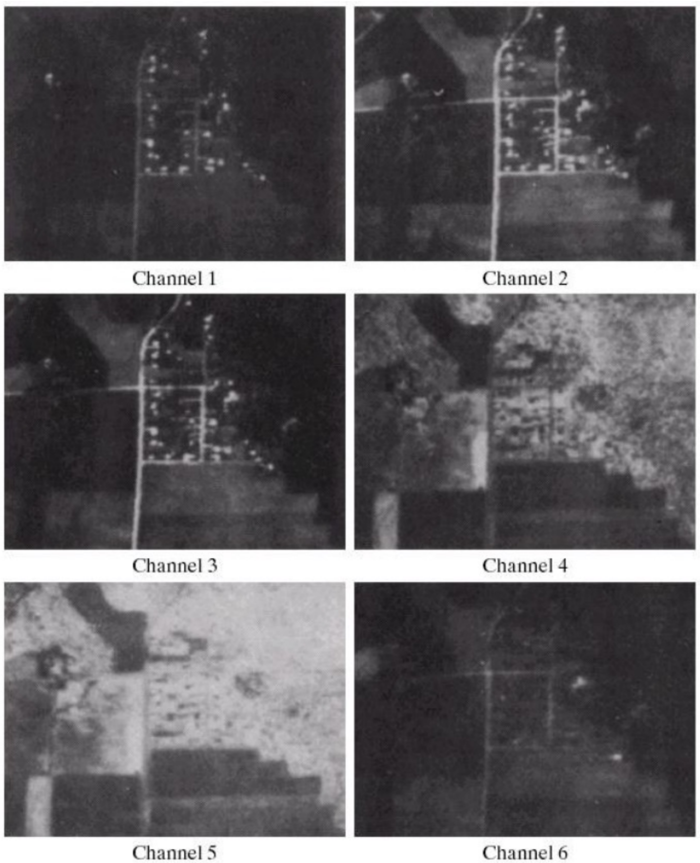
$$C = E\left\{(\underline{x} - \underline{m}_x)(\underline{x} - \underline{m}_x)^T\right\}$$

  - **Let A be a matrix whose rows are formed from the eigenvectors of the covariance matrix C**
  - **A is ordered so that the first row is the eigenvector corresponding to the largest eigenvalue, and the last row is the eigenvector corresponding to the smallest eigenvalue**
  - **The following transform is called the Karhunen-Loeve transform**

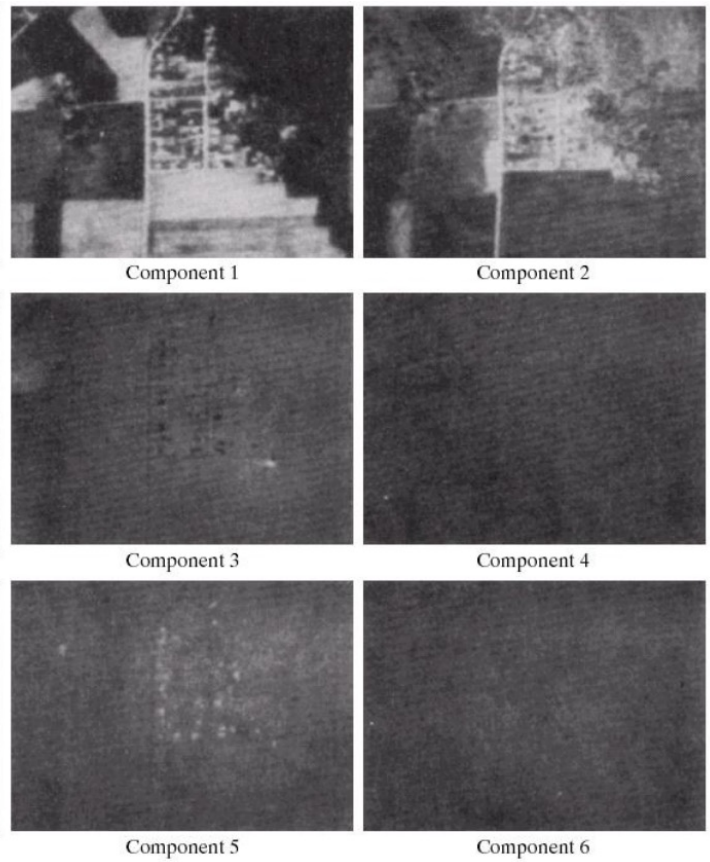
$$\underline{y} = A(\underline{x} - \underline{m}_x)$$

Image Sampling & Transforms

- **Example**



Original images (channels)



Six principal components after KL transform

Image Sampling & Transforms

- Discrete Fourier Transform (DFT)
  - Idea
    - Any function that periodically repeats itself can be expressed as a sum of sine and cosine of different frequencies.

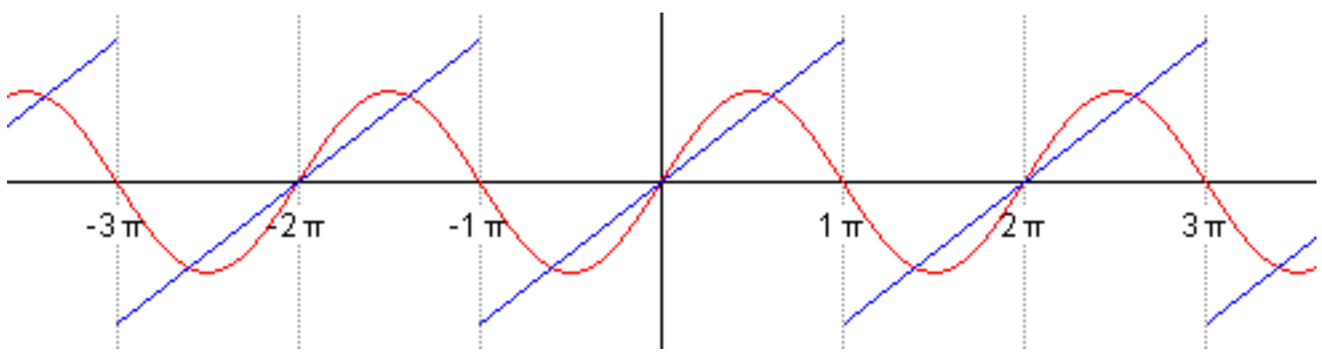


Image Sampling & Transforms

- **Discrete Fourier Transform (DFT)**
  - **Transform**

$$F(u,v) = \frac{1}{n}\sum_{j=0}^{n-1}\sum_{k=0}^{n-1} f(j,k)\exp\{-\frac{2\pi i(uj+vk)}{n}\}$$

  - **Inverse DFT**

$$f(j,k) = \frac{1}{n}\sum_{u=0}^{n-1}\sum_{v=0}^{n-1} F(u,v)\exp\{\frac{2\pi i(uj+vk)}{n}\}$$

$i = \sqrt{-1}$ **and** $f(j,k)$ **is the input sequence**

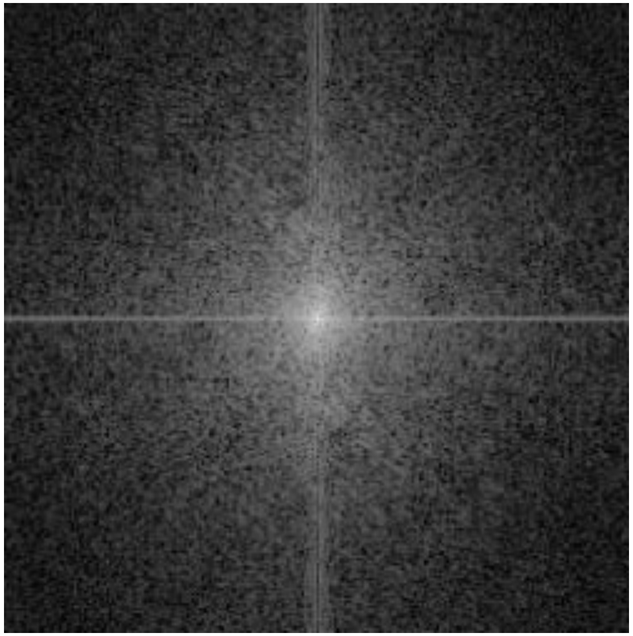
Image Sampling & Transforms

- Discrete Fourier Transform (DFT)
  - Fast Fourier Transform (FFT)
    - $O(n \log n)$
  - Not so popular in image compression
    - performance is not good enough
    - computational cost for complex number is expensive

Image Sampling & Transforms

- Discrete Fourier Transform (DFT)
  - Example

original



log magnitude

Image Sampling & Transforms

- Discrete Fourier Transform (DFT)
  - Application – Noise reduction

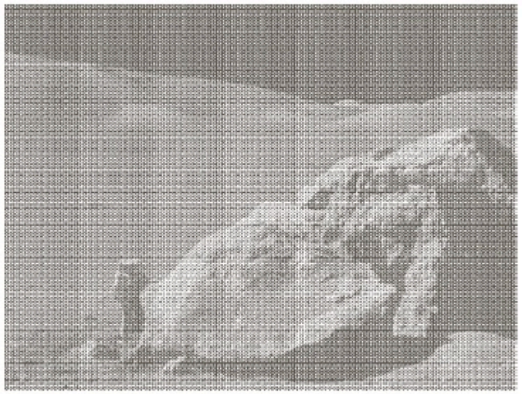


Image Sampling & Transforms

- Discrete Fourier Transform (DFT)
  - Application– Low Pass Filter

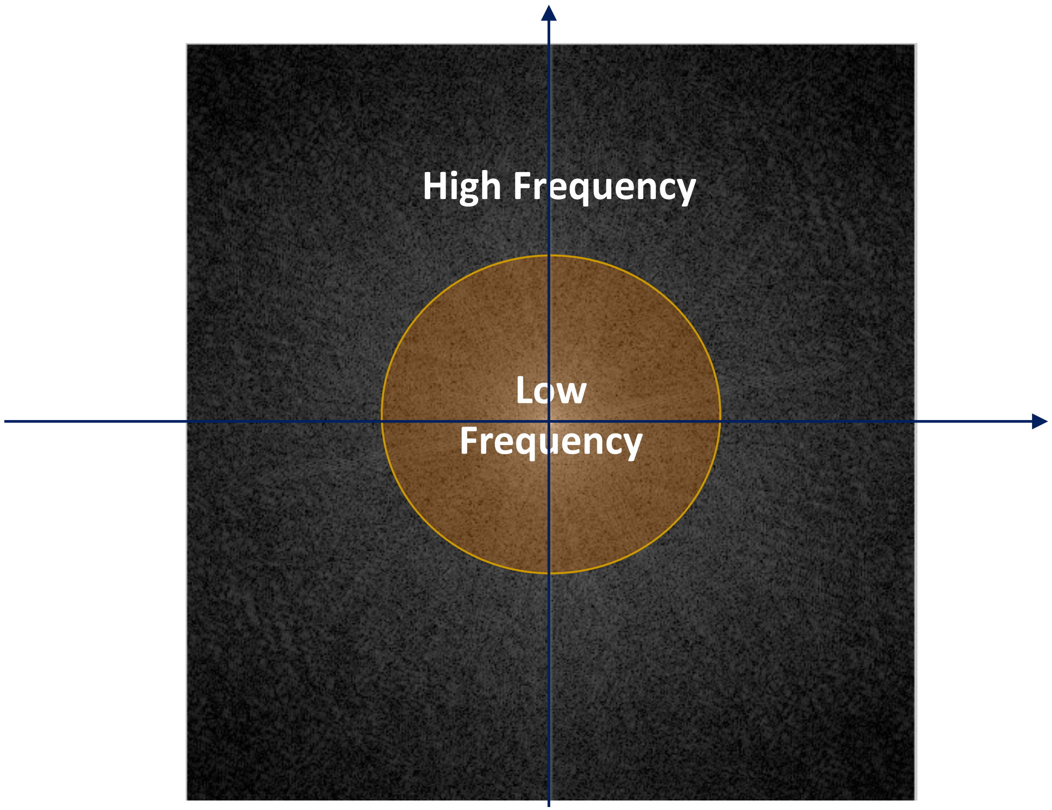


Image Sampling & Transforms

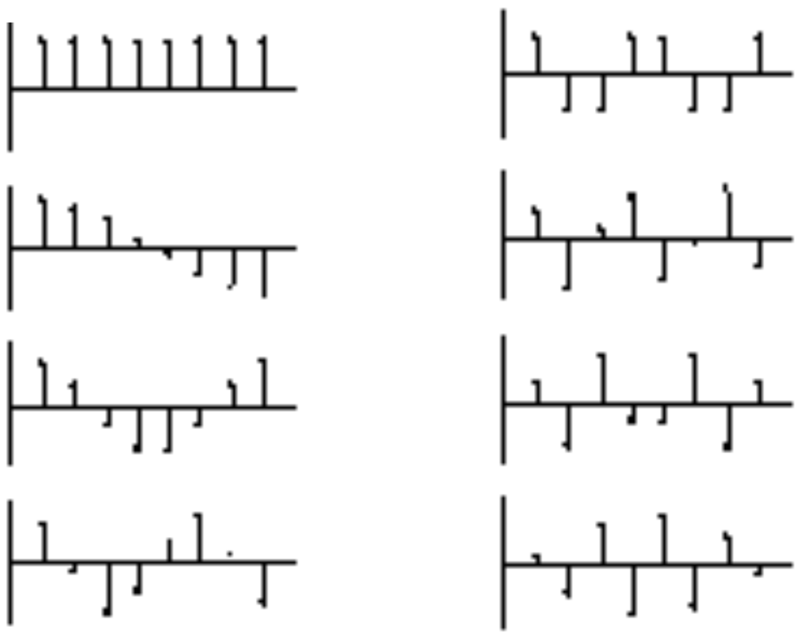
- Discrete Fourier Transform (DFT)
  - Application– Low Pass Filter

Original image

After low pass filter

Image Sampling & Transforms

- Discrete Cosine Transform (DCT)
  - 1D DCT basis functions



n=8

Image Sampling & Transforms

- **Discrete Cosine Transform (DCT)**
  - **Transform**

$$F(u,v) = \frac{C(u)C(v)}{4} \sum_{j=0}^{n-1} \sum_{k=0}^{n-1} f(j,k) cos\left[\frac{(2j+1)u\pi}{2n}\right] cos\left[\frac{(2k+1)v\pi}{2n}\right]$$

  - **Inverse DCT**

$$f(j,k) = \sum_{v=0}^{n-1} \sum_{u=0}^{n-1} C(u)C(v)F(u,v) cos\left[\frac{(2j+1)u\pi}{2n}\right] cos\left[\frac{(2k+1)v\pi}{2n}\right]$$

$$C(w) = \begin{cases} 1/\sqrt{2} & \text{if } w = 0 \\ 1 & \text{if } w = 1,2,\ldots,n\text{-}1 \end{cases}$$

Image Sampling & Transforms

- **Discrete Cosine Transform (DCT)**
  - **Transform**

$$F(u,v) = \frac{C(u)C(v)}{4} \sum_{j=0}^{n-1} \sum_{k=0}^{n-1} f(j,k) cos\left[\frac{(2j+1)u\pi}{2n}\right] cos\left[\frac{(2k+1)v\pi}{2n}\right]$$

$$F_{10} = \frac{c_1 c_0}{4}\left[\cos\frac{\pi}{16}\left(\sum_{i=0}^{7} f(0,j) - \sum_{i=0}^{7} f(7,j)\right) + \cos\frac{3\pi}{16}\left(\sum_{i=0}^{7} f(1,j) - \sum_{i=0}^{7} f(6,j)\right) + \right.$$

$$\left.\cos\frac{5\pi}{16}\left(\sum_{i=0}^{7} f(2,j) - \sum_{i=0}^{7} f(5,j)\right) + \cos\frac{7\pi}{16}\left(\sum_{i=0}^{7} f(3,j) - \sum_{i=0}^{7} f(4,j)\right)\right]$$

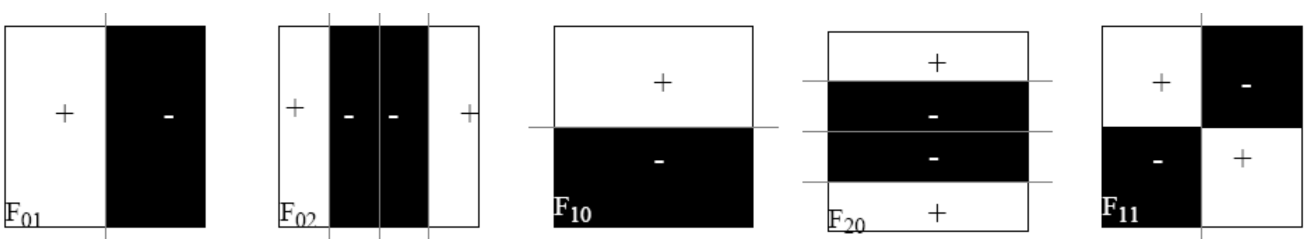


Image Sampling & Transforms

- **2D DCT basis functions (8x8)**

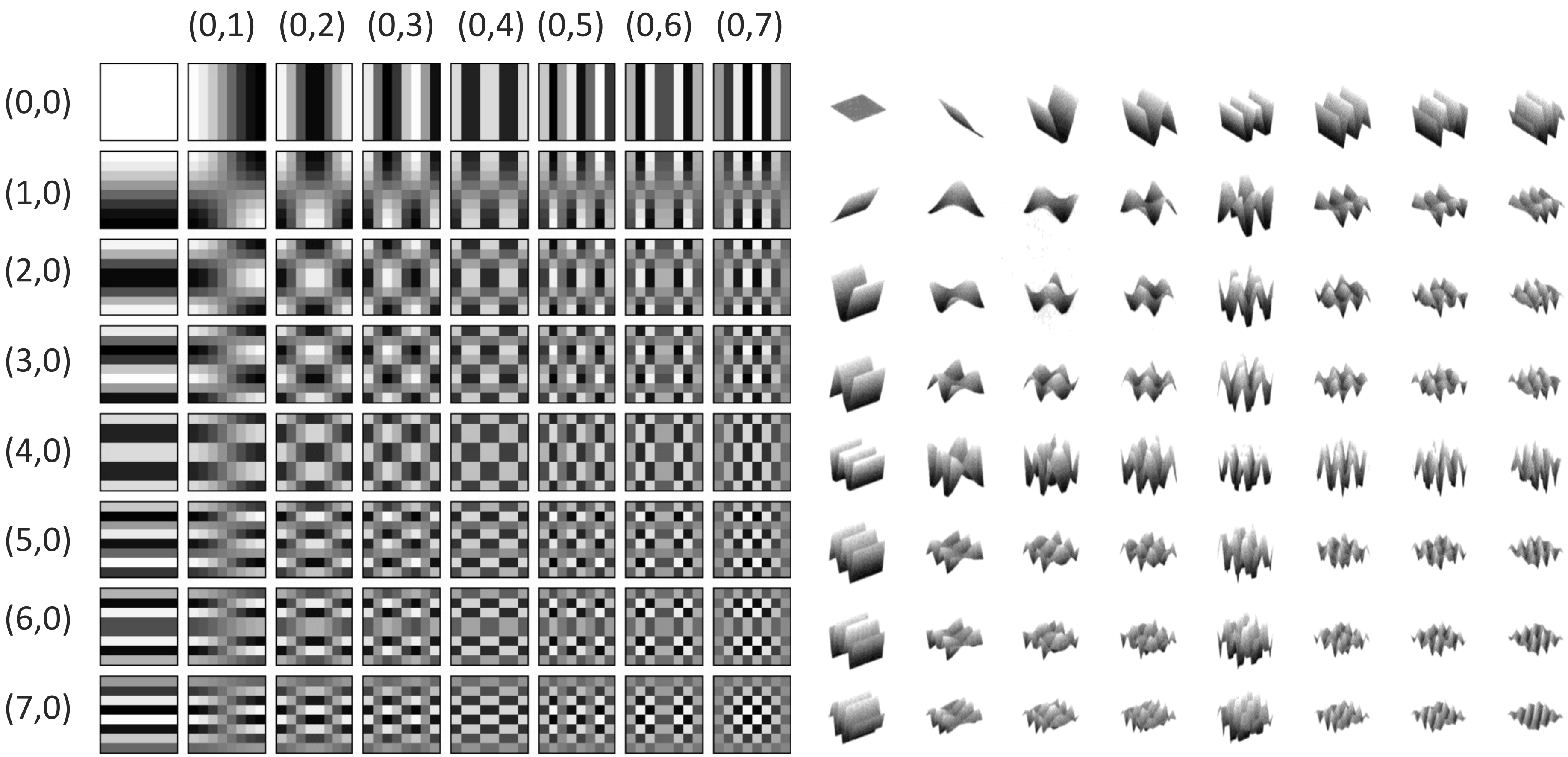


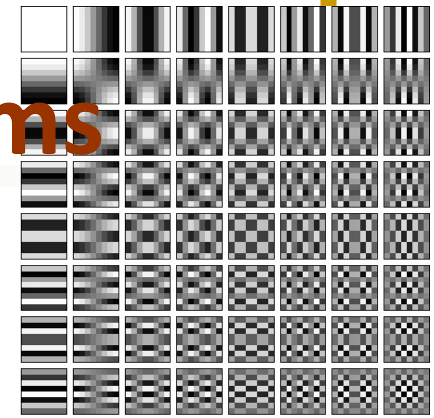
Image Sampling & Transforms

- The Meaning of DCT Coefficients
  - Represent the spatial frequency content within an 8x8 image block
  - The (0,0) coefficient is called DC coefficient
    - The average of the 64 image pixel values in the block
  - The rest of 63 coefficients are called AC coefficients
  - Move to the right of the block

    → the energy in higher horizontal frequencies
  - Move down the block

    → the energy in higher vertical frequencies

Image Sampling & Transforms



- **Examples of 8x8 DCT Coefficients**

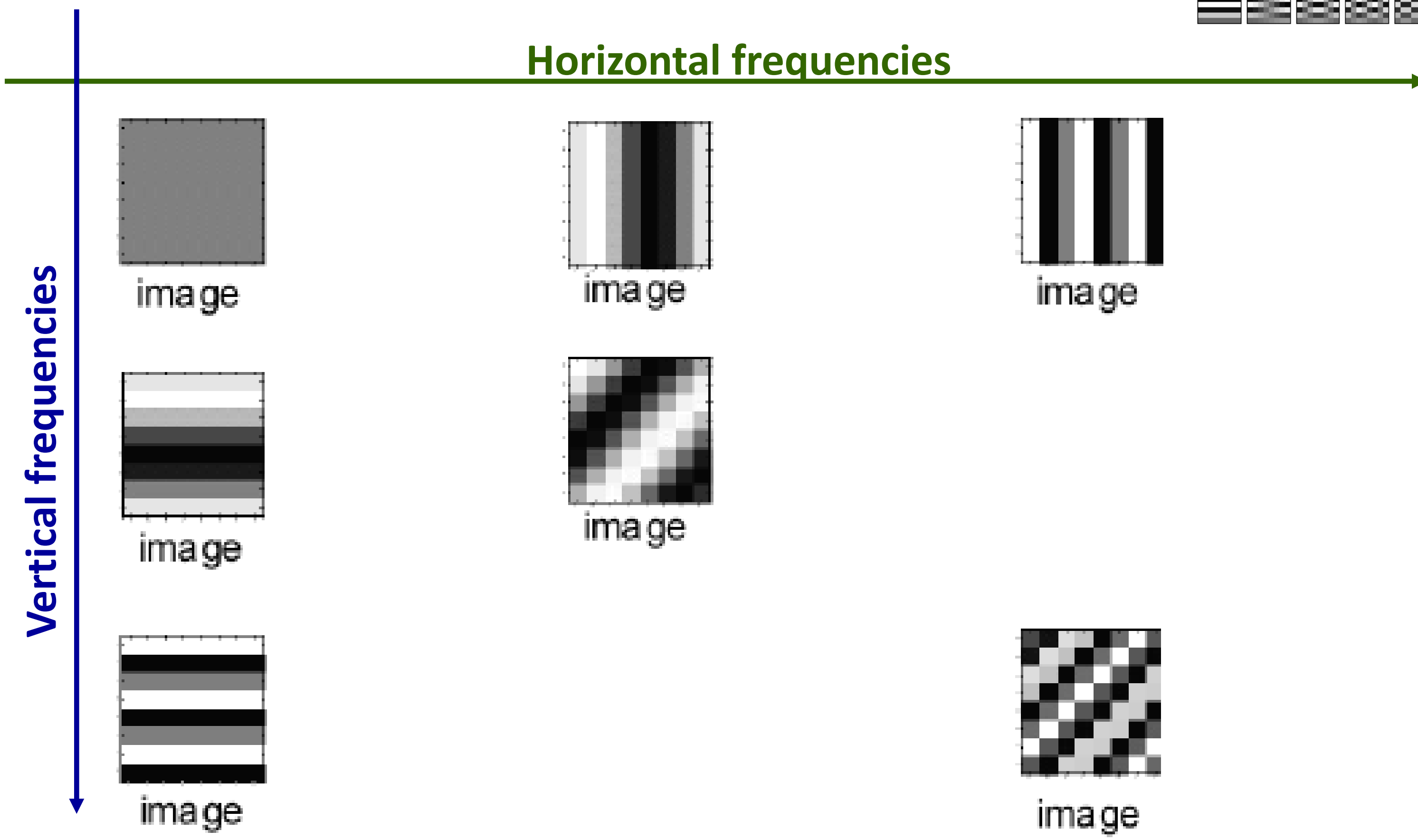
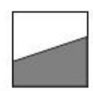
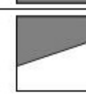
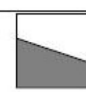
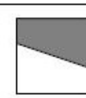
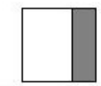
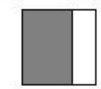
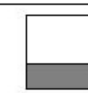
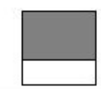
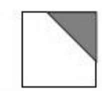
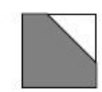
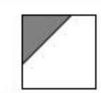
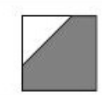


Image Sampling & Transforms

- **Discrete Cosine Transform (DCT)**

Vertical-dominant	section 1	$\lvert F01\rvert > \lvert F10\rvert$	$F01 \le 0$	$F10 > 0$	
				$F10 < 0$	
	section 2		$F01 \ge 0$	$F10 > 0$	
				$F10 < 0$	
Horizontal-dominant	section 3	$\lvert F01\rvert < \lvert F10\rvert$	$F10 \le 0$	$F01 > 0$	
				$F01 < 0$	
	section 4		$F10 \ge 0$	$F01 > 0$	
				$F01 < 0$	

Vertical	$F10 = 0$	$F01 > 0$	
		$F01 < 0$	
Horizontal	$F01 = 0$	$F10 > 0$	
		$F10 < 0$	
Diagonal	$\lvert F01\rvert = \lvert F10\rvert$	$F01 < 0$, $F10 > 0$	
		$F01 > 0$, $F10 < 0$	
		$F01 > 0$, $F10 > 0$	
		$F01 < 0$, $F10 < 0$	

Bo Shen and Ishwar K. Sethi, "Direct feature extraction from compressed images,"1996

Image Sampling & Transforms

- Discrete Cosine Transform (DCT)
  - Use cosine function as its basis function
  - Performance similar to KLT
  - Fast algorithm available
  - Most popular in image compression
  - JPEG (Joint Photographic Experts Group)
  - The periodicity implied by DCT implies that less blocking artifact will be introduced than DFT

Image Sampling & Transforms

- **Hadamard transform**
  - Based on Hadamard matrix
    - A square array with elements of plus and minus 1's
    - Rows and columns are orthogonal
    - A normalized Hadamard matrix satisfies $HH^T = I$
    - The general form:

$$H_{2N} = \frac{1}{\sqrt{2}}\begin{bmatrix} H_N & H_N \\ H_N & -H_N \end{bmatrix}$$

$$\mathbf{H}_2 = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

$$\mathbf{H}_4 = \frac{1}{2}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} \quad \begin{array}{c} \text{Sign Changes} \\ 0 \\ 3 \\ 1 \\ 2 \end{array}$$

$$\mathbf{H}_8 = \frac{1}{2\sqrt{2}}\left[\begin{array}{cccc|cccc} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ \hline 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{array}\right] \quad \begin{array}{c} \text{Sign Changes} \\ 0 \\ 7 \\ 3 \\ 4 \\ 1 \\ 6 \\ 2 \\ 5 \end{array}$$

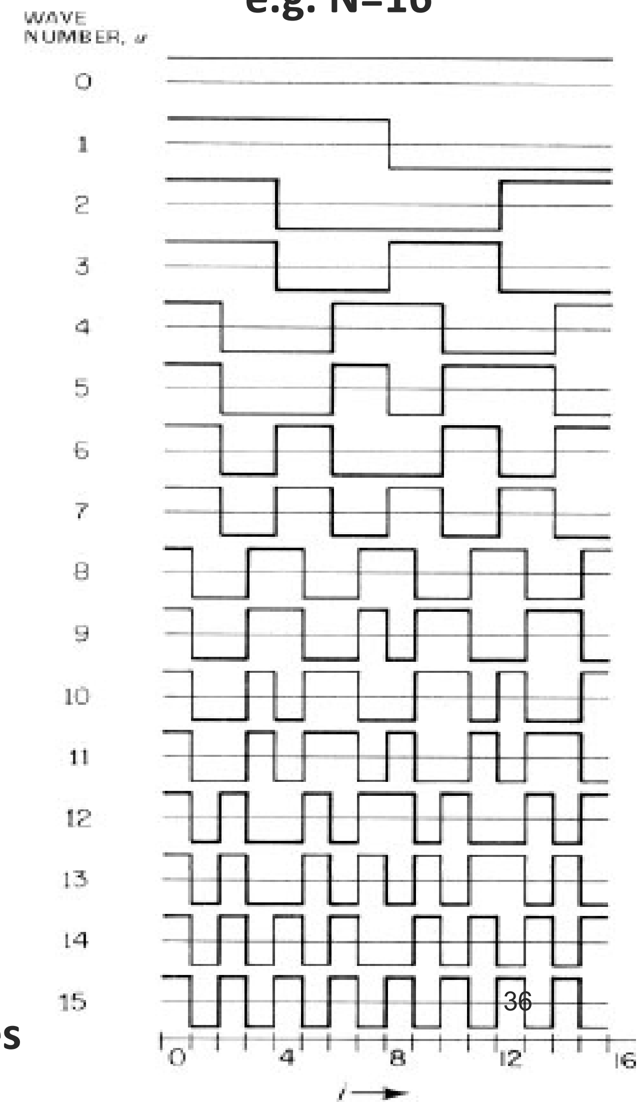
Image Sampling & Transforms

- **Hadamard transform**

e.g. N=4

$$\mathbf{H}_8 = \frac{1}{2\sqrt{2}} \left[\begin{array}{rrrr|rrrr} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ \hline 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{array}\right] \quad \begin{array}{c} \text{Sign Changes} \\ 0 \\ 7 \\ 3 \\ 4 \\ 1 \\ 6 \\ 2 \\ 5 \end{array}$$

e.g. N=16



- **Applications**
  - data encryption and signal processing
  - data compression algorithms
    - HD Photo and MPEG-4 AVC
    - video compression
      - the sum of absolute transformed differences

Image Sampling & Transforms

- **Haar transform**
  - **Derived from the Haar matrix**

$$R_N = \begin{bmatrix} V_N \\ W_N \end{bmatrix}_{NxN}$$

e.g.

$$\mathbf{H}_4 = \frac{1}{2}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ \sqrt{2} & -\sqrt{2} & 0 & 0 \\ 0 & 0 & \sqrt{2} & -\sqrt{2} \end{bmatrix}$$

Scaling matrix:

$$\mathbf{V}_N = \frac{1}{\sqrt{2}}\begin{bmatrix} 1\ 1\ 0\ 0\ 0\ 0 & \cdots & 0\ 0\ 0\ 0 \\ 0\ 0\ 1\ 1\ 0\ 0 & \cdots & 0\ 0\ 0\ 0 \\ & & \\ 0\ 0\ 0\ 0\ 0\ 0 & \cdots & 1\ 1\ 0\ 0 \\ 0\ 0\ 0\ 0\ 0\ 0 & \cdots & 0\ 0\ 1\ 1 \end{bmatrix}$$

Wavelet matrix:

$$\mathbf{W}_N = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 & \ldots & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & \ldots & 0 & 0 & 0 & 0 \\ & & \vdots & & & & & & \vdots & & \\ 0 & 0 & 0 & 0 & 0 & 0 & \ldots & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \ldots & 0 & 0 & 1 & -1 \end{bmatrix}$$

$$\mathbf{H}_8 = \frac{1}{\sqrt{8}}\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ \sqrt{2} & \sqrt{2} & -\sqrt{2} & -\sqrt{2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \sqrt{2} & \sqrt{2} & -\sqrt{2} & -\sqrt{2} \\ 2 & -2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & -2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & -2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & -2 \end{bmatrix}$$

Image Sampling & Transforms

- **Haar transform**
  - **1st-level Haar transform of a Nx1 vector f**

$$f_1 = R_N f = [a_1 \mid d_1]^T \quad where \quad \underbrace{a_1 = V_N f}_{\text{trend}}; \quad \underbrace{d_1 = W_N f}_{\text{fluctuation}}$$

  - **2nd-level Haar transform of a Nx1 vector f**

$$f_2 = [a_2 \mid d_2 \mid d_1]^T \quad where \quad a_2 = V_{N/2} a_1; \quad d_2 = W_{N/2} a_1$$

  - **The process continues till the full transformation**

$$\Rightarrow f_n = [a_n \mid d_n \mid d_{n-1} \mid \cdots \mid d_1]^T$$

Image Sampling & Transforms

- Haar transform
  - E.g. basis functions with N=16
  - Sample an input data sequence with finer and finer resolution increasing at the power of 2
  - Provide a transform domain in which type of differential energy is concentrated in localized regions

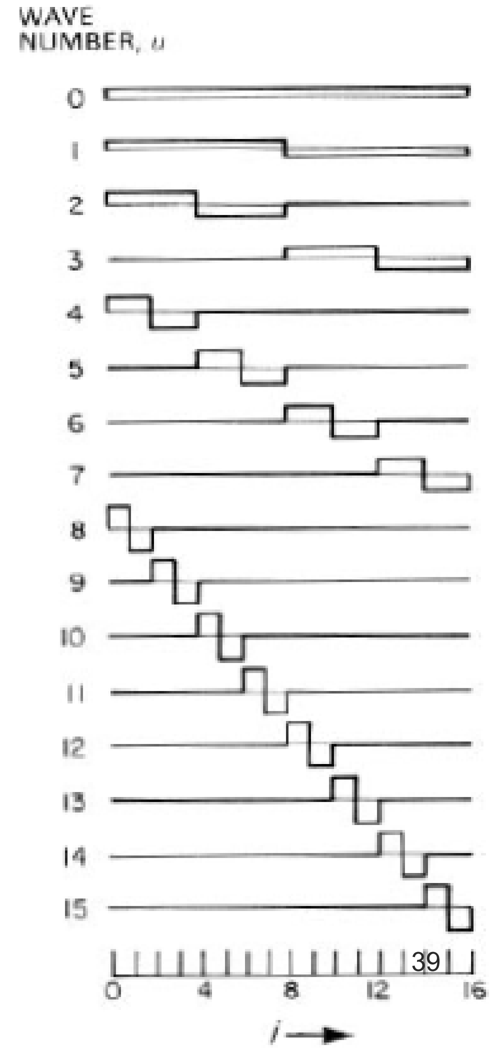
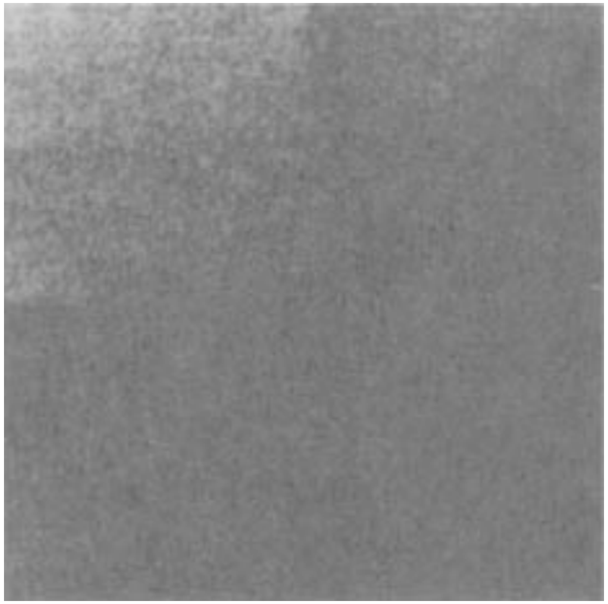
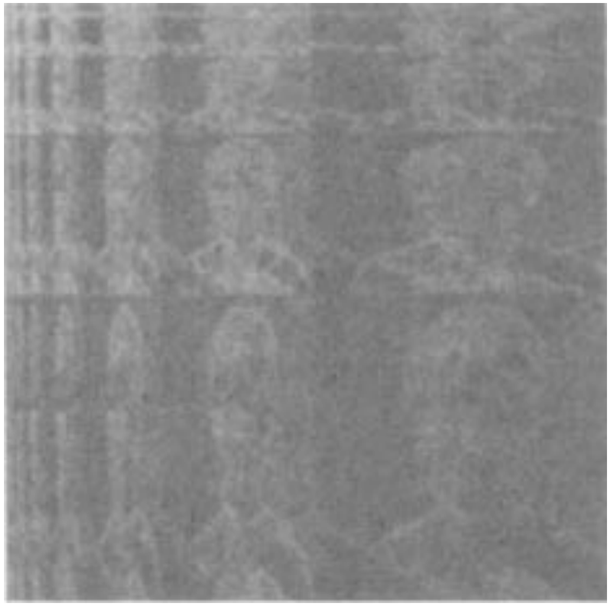


Image Sampling & Transforms

- **Hadamard transform and Haar transform**
  - Example



Hadamard transform



Haar transform

Image Sampling & Transforms

- **Hough transform**
  - An edge is not a line...
  - How to detect lines in an image?

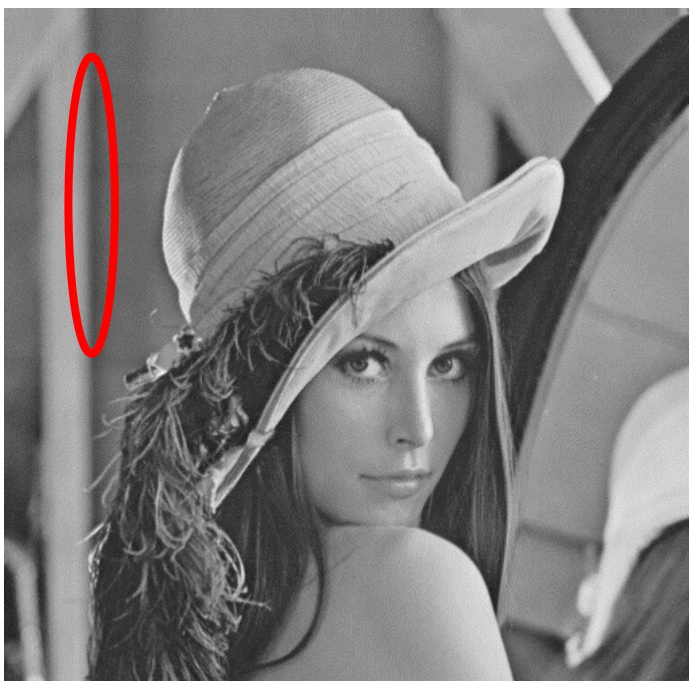


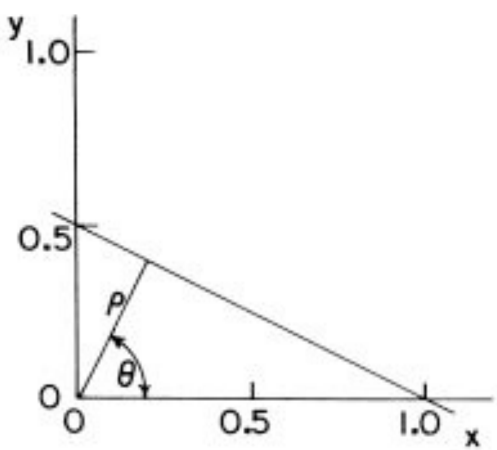

Image Sampling & Transforms

- Hough transform
  - How to detect lines in an image?
    - Search for the lines at every position with possible orientations
    - Voting scheme: Hough transform
  - Performed after edge detection
  - Can locate straight lines, circles, parabolas, ellipses, etc
    - As the curve can be specified in a parametric form
  - Advantages
    - Tolerate gaps between edges
    - Relatively unaffected by noise
    - Can deal with occlusion

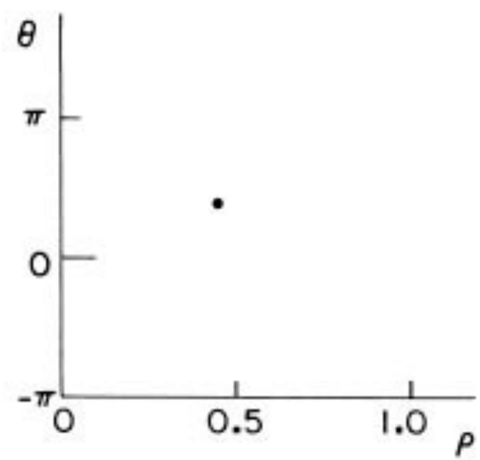
Image Sampling & Transforms

- **Hough transform**
  - A straight line can be represented as $y = mx + b$
    - Vertical lines?
  - Another representation is

$$\rho = x cos\theta + y sin\theta, \qquad -\frac{\pi}{2} \le \theta \le \frac{\pi}{2}$$



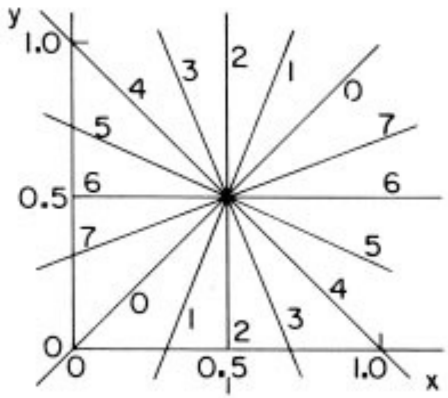
(a) Parametric line



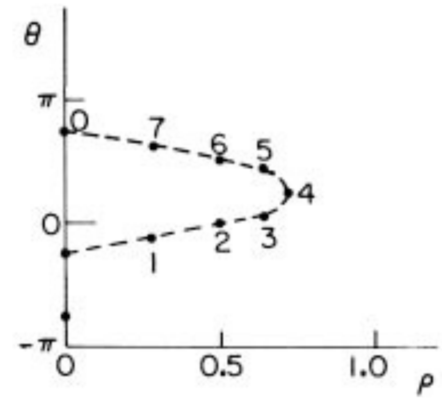
(b)Hough transform

Image Sampling & Transforms

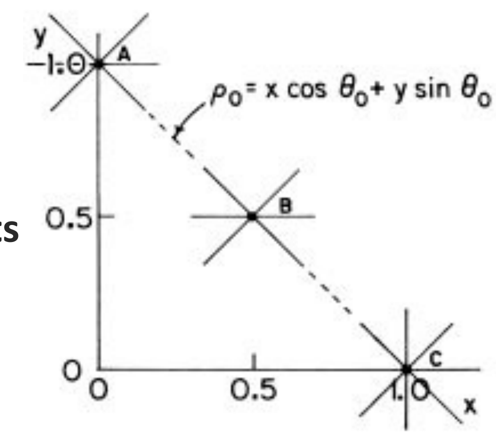
- **Hough transform**



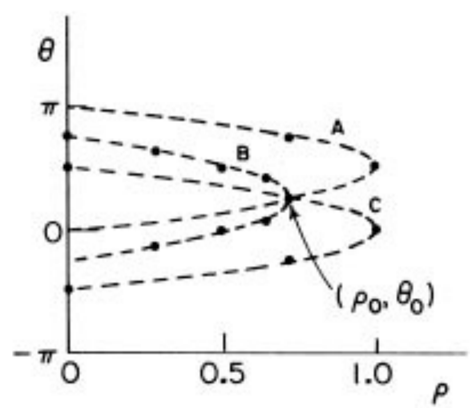
(c) Family of lines, common point



(d) Hough transform of (c)



(e) Colinear points



(f) Hough transform of (e)

Image Sampling & Transforms

- **Hough transform for line detection**
  - **Quantize the Hough Transform space**
    - Identify the maximum and minimum values of ρ and θ
  - **Generate an accumulator array A(ρ,θ)**
  - **For all edge points (xi, yi) in the image**
    - Use gradient direction as θ
    - Compute ρ from the equation
    - Increment A(ρ,θ) by one
  - **For all cells in A(ρ,θ)**
    - Search for the maximum value
    - Calculate the equation of the line

Image Sampling & Transforms

- **Hough transform for line detection**