Why?

- To extract information from the image
- To ‘process’ the image ‘easily’

e.g. For continuous functions, orthogonal series expansions provide series coefficients which can be used for any further processing/analyses.

Unitary Transform: 1D

For a one-dimensional sequence $\{\, u(n), 0 \le n \le N-1 \,\}$, represented as a vector $\mathbf{u}$ of size N, a general transformation is written as

$$\mathbf{v} = \mathbf{A}\mathbf{u} \quad \Rightarrow v(k) = \sum_{n=0}^{N-1} a(k,n)u(n), \quad 0 \le k \le N-1$$

and

$$\mathbf{u} = \mathbf{A}^{-1}\mathbf{v}$$

In a special case, when $\mathbf{A}^{-1} = \mathbf{A}^{*\mathbf{T}}$, i.e. $\mathbf{A}$ is unitary matrix, we called the transform as unitary transform and can be written as

$$\mathbf{u} = \mathbf{A}^{*\mathbf{T}}\mathbf{v} \quad \Rightarrow u(n) = \sum_{k=0}^{N-1} v(k)a^{*}(k,n), \quad 0 \le n \le N-1$$

The columns of $\mathbf{A}^{*\mathbf{T}}$, that is the vectors $\{\mathbf{a}_{\mathbf{k}}^{*} = a^{*}(k,n), 0 \le n \le N-1\}^{T}$ are called basis vectors of $\mathbf{A}$.

2-D orthogonal and unitary transforms

$$v(k,l) = \sum_{m=0}^{N-1}\sum_{n=0}^{N-1} u(m,n)a_{k,l}(m,n) \qquad 0 \le k,l \le N-1$$

Complexity? $O(N^4)$!!

Separable unitary transforms

$$a_{k,l}(m,n) = a_k(m)b_l(n) = a\underbrace{(k,m)}_{A}\underbrace{b(l,n)}_{B}$$

Where $\{a_k(m), k = 0,\dots,N-1\}$ and $\{b_l(n), l = 0,\dots,N-1\}$ are one-dimensional complete orthonormal sets of basis vectors.

$$v(k,l) = \sum_{m=0}^{N-1}\sum_{n=0}^{N-1} a(k,m)u(m,n)b(l,n) \leftrightarrow \mathbf{V} = \mathbf{AUB}^{\mathbf{T}}$$

$$u(m,n) = \sum_{k=0}^{N-1}\sum_{l=0}^{N-1} a^*(k,m)v(k,l)b^*(l,n) \leftrightarrow \mathbf{U} = \mathbf{A}^{*\mathbf{T}}\mathbf{V}\mathbf{B}^{*\mathbf{T}}$$

Complexity?

$O(N^3)$!!

Basis Images

Let $\mathbf{a}_k^*$ denote the kth column of $\mathbf{A}^{*T}$. Then we can define matrices

$$\mathbf{A}_{k,l}^* = \mathbf{a}_k^* \mathbf{a}_l^{*T} \qquad \leftrightarrow a_{k,l}^*(m,n) = A^*_{k,l}(m,n)$$

- $<\mathbf{F},\mathbf{G}>= \sum_{m=0}^{N-1}\sum_{n=0}^{N-1} f(m,n)g^*(m,n)$ is the matrix inner product

$$\begin{cases} v(k,l) = \sum_{m=0}^{N-1}\sum_{n=0}^{N-1} u(m,n)a_{k,l}(m,n) \\ u(m,n) = \sum_{k=0}^{N-1}\sum_{l=0}^{N-1} v(k,l)a_{k,l}^*(m,n) \end{cases} \Rightarrow \begin{cases} \mathbf{U} = \sum_{k=0}^{N-1}\sum_{l=0}^{N-1} v(k,l)\mathbf{A}_{k,l}^* \\ v(k,l) = <\mathbf{U},\mathbf{A}_{k,l}^*> \end{cases}$$

Basis Images

$$\begin{cases} v(k,l) = \sum_{m=0}^{N-1}\sum_{n=0}^{N-1} u(m,n)\, a_{k,l}(m,n) \\ u(m,n) = \sum_{k=0}^{N-1}\sum_{l=0}^{N-1} v(k,l)\, a^*_{k,l}(m,n) \end{cases} \Rightarrow \begin{cases} \mathbf{U} = \sum_{k=0}^{N-1}\sum_{l=0}^{N-1} v(k,l)\, \boxed{\mathbf{A}^*_{k,l}} \\ v(k,l) = \langle \mathbf{U}, \mathbf{A}^*_{k,l} \rangle \end{cases}$$

A^*_{00} A^*_{10}
A^*_{01} A^*_{11}
$\vdots$
A^*_{0N-1} A^*_{1N-1}

- Image **U** can be described as a linear combination of N^2 matrix $\mathbf{A}^*_{k,l}$, $k, l = 0,\ldots,N\text{-}1$
- $\mathbf{A}^*_{k,l}$ are called the *basis images*
- The transform coefficient $v(k,l)$ is simply the inner product of the (k,l)th basis image with the given image.
- Any NxN image can be expanded in a series using set of N^2 basis images.

Example:

- Given $\mathbf{A} = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$, $\quad \mathbf{U} = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$ compute transform $\mathbf{V}$

$$v(k,l) = \sum_{m=0}^{N-1}\sum_{n=0}^{N-1} a(k,m)u(m,n)b(l,n) \leftrightarrow \mathbf{V} = \mathbf{AUB}^{\mathrm{T}}$$

$$\mathbf{V} = \frac{1}{2}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} = \begin{pmatrix} 5 & -1 \\ -2 & 0 \end{pmatrix}$$

Basis images ?? $\boxed{\mathbf{A}^*_{k,l} = \mathbf{a}^*_k \mathbf{a}^{*T}_l}$

$$\mathbf{A}^*_{0,0} = \frac{1}{2}\begin{pmatrix} 1 \\ 1 \end{pmatrix}\begin{pmatrix} 1 & 1 \end{pmatrix} = \frac{1}{2}\begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$$

$$\mathbf{A}^*_{0,1} = \frac{1}{2}\begin{pmatrix} 1 \\ 1 \end{pmatrix}\begin{pmatrix} 1 & -1 \end{pmatrix} = \frac{1}{2}\begin{pmatrix} 1 & -1 \\ 1 & -1 \end{pmatrix} = \mathbf{A}^{*T}_{1,0}$$

$$\mathbf{A}^*_{1,1} = \frac{1}{2}\begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}$$

Example:

- Given $\mathbf{A} = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$, $\mathbf{U} = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$ compute transform $\mathbf{V}$:

$$v(k,l) = \sum_{m=0}^{N-1}\sum_{n=0}^{N-1} a(k,m)u(m,n)b(l,n) \leftrightarrow \mathbf{V} = \mathbf{AUB}^{\mathrm{T}}$$

$$\mathbf{V}' = \frac{1}{2}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} = \begin{pmatrix} 5 & -1 \\ -2 & 0 \end{pmatrix}$$

$$\mathbf{A}^{*T}\mathbf{V}'\mathbf{A}^{*} = \frac{1}{2}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}\begin{pmatrix} 5 & -1 \\ -2 & 0 \end{pmatrix}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$$

Properties:

- In the unitary transform defined as $\mathbf{v} = \mathbf{A}\mathbf{u}$,

$$\|\mathbf{v}\|^2 = \|\mathbf{u}\|^2$$

Proof:

$$\|\mathbf{v}\|^2 = \sum_{k=0}^{N-1} |v(k)|^2 = \mathbf{v}^{*T}\mathbf{v} = \mathbf{u}^{*T}\mathbf{A}^{*T}\mathbf{A}\mathbf{u} = \mathbf{u}^{*T}\mathbf{u} = \|\mathbf{u}\|^2$$

- Length of the vector **u** does not change after transform and **0** will be map to **0**
- Unitary transform can be thought as a rotation in N-dimensional space

Properties:

mean $\boldsymbol{\mu}_{\mathbf{v}} = E[\mathbf{v}] = E[\mathbf{A}\mathbf{u}] = \mathbf{A}E[\mathbf{u}] = \mathbf{A}\boldsymbol{\mu}_{\mathbf{u}}$

Covariance $\mathbf{R}_{\mathbf{v}} = E[(\mathbf{v}-\boldsymbol{\mu}_{\mathbf{v}})(\mathbf{v}-\boldsymbol{\mu}_{\mathbf{v}})^{*T}]$

$= \mathbf{A}E[(\mathbf{u}-\boldsymbol{\mu}_{\mathbf{u}})(\mathbf{u}-\boldsymbol{\mu}_{\mathbf{u}})^{*T}]\mathbf{A}^{*T}$

$= \mathbf{A}\mathbf{R}_{\mathbf{u}}\mathbf{A}^{*T}$

One Dimensional Discrete Fourier Transform:

The DFT of a sequence $\{u(n),\ n = 0,\ldots N-1\}$ is defined as

$$v(k) = \sum_{n=0}^{N-1} u(n) W_N^{kn} \qquad k = 0,1,\ldots N-1$$

where

$$W_N = exp\{\frac{-j2\pi}{N}\}$$

The inverse transform is given by

$$u(n) = \frac{1}{N}\sum_{k=0}^{N-1} v(k) W_N^{-kn} \qquad n = 0,1,\ldots N-1$$

The pair of equations are not scaled properly to be unitary transformations.

One Dimensional Discrete Fourier Transform:

The unitary DFT

$$v(k) = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} u(n) W_N^{kn} \quad k = 0, 1, \ldots N-1$$

$$u(n) = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} v(k) W_N^{-kn} \quad n = 0, 1, \ldots N-1$$

The NxN unitary DFT matrix **F** is given by

$$\mathbf{F} = \{\frac{1}{\sqrt{N}} W_N^{kn}\} \quad 0 \leq k, n \leq N-1$$

Properties:

- Symmetry $\mathbf{F} = \mathbf{F}^T \rightarrow \mathbf{F}^{-1} = \mathbf{F}^*$ (unitary : $\mathbf{F}^{-1} = \mathbf{F}^{*T}$)
- FFT needs $O(N \log_2 N)$ operations (DFT needs $O(N^2)$)
- Real DFT is conjugate symmetrical about $N/2$

$$v^*(\frac{N}{2} - k) = v(\frac{N}{2} + k)$$

- $x_2(n)$ is the circular convolution between $h(n)$ and $x_1(n)$

$$DFT\{x_2(n)\}_N = DFT\{h(n)\}_N \, DFT\{x_1(n)\}_N$$

  - Extend the length of $h(n)(N')$ and $x_1(n)$ (N) with zeros to have the same length ($M \geq N' + N - 1$), the above equation can be used to compute linear convolution

Properties:

- Symmetry $\mathbf{F} = \mathbf{F}^T \rightarrow \mathbf{F}^{-1} = \mathbf{F}^*$ (unitary : $\mathbf{F}^{-1} = \mathbf{F}^{*T}$)
- FFT needs $O(N \log_2 N)$ operations (DFT needs $O(N^2)$)
- Real DFT is conjugate symmetrical about $N/2$

$$v^*(\frac{N}{2} - k) = v(\frac{N}{2} + k)$$

- $x_2(n)$ is the circular convolution between $h(n)$ and $x_1(n)$

$$DFT\{x_2(n)\}_N = DFT\{h(n)\}_N DFT\{x_1(n)\}_N$$

  - Extend the length of $h(n)(N')$ and $x_1(n)$ (N) with zeros to have the same length ($M \geq N' + N - 1$), the above equation can be used to compute linear convolution

Reading Assignment:
https://www.youtube.com/watch?v=CVV0TvNK6pk&list=PL32DC1B4A05136109
Prof. PK Biswas's NPTEL lectures (Lectures 13-14), Introduction to Digital Image Processing

Two Dimensional Discrete Fourier Transform:

The two-dimensional DFT of an NxN image {u(m,n)} is defined as

$$v(k,l)=\sum_{m=0}^{N-1}\sum_{n=0}^{N-1}u(m,n)W_N^{km}W_N^{ln}$$

$$u(m,n)=\frac{1}{N^2}\sum_{k=0}^{N-1}\sum_{l=0}^{N-1}v(k,l)W_N^{-km}W_N^{-ln}$$

The two-dimensional unitary DFT of an NxN image {u(m,n)} is defined as

$$v(k,l)=\frac{1}{N}\sum_{m=0}^{N-1}\sum_{n=0}^{N-1}u(m,n)W_N^{km}W_N^{ln}$$

$$u(m,n)=\frac{1}{N}\sum_{k=0}^{N-1}\sum_{l=0}^{N-1}v(k,l)W_N^{-km}W_N^{-ln}$$

$$\mathbf{V}=\mathbf{FUF}$$ (**F** is a symmetric matrix)

Two Dimensional Discrete Fourier Transform:

The two-dimensional DFT of an NxN image {u(m,n)} is defined as

$$v(k,l) = \sum_{m=0}^{N-1}\sum_{n=0}^{N-1} u(m,n) W_N^{km} W_N^{ln}$$

$$u(m,n) = \frac{1}{N^2}\sum_{k=0}^{N-1}\sum_{l=0}^{N-1} v(k,l) W_N^{-km} W_N^{-ln}$$

The two-dimensional unitary DFT of an NxN image {u(m,n)} is defined as

$$v(k,l) = \frac{1}{N}\sum_{m=0}^{N-1}\sum_{n=0}^{N-1} u(m,n) W_N^{km} W_N^{ln} = \frac{1}{N}\sum_{m=0}^{N-1} W_N^{km} . \sqrt{N} . \frac{1}{\sqrt{N}}\sum_{n=0}^{N-1} u(m,n) W_N^{ln}$$

Two Dimensional Discrete Fourier Transform:

The two-dimensional DFT of an NxN image $\{u(m,n)\}$ is defined as

$$v(k,l) = \sum_{m=0}^{N-1}\sum_{n=0}^{N-1} u(m,n) W_N^{km} W_N^{ln}$$

$$u(m,n) = \frac{1}{N^2}\sum_{k=0}^{N-1}\sum_{l=0}^{N-1} v(k,l) W_N^{-km} W_N^{-ln}$$

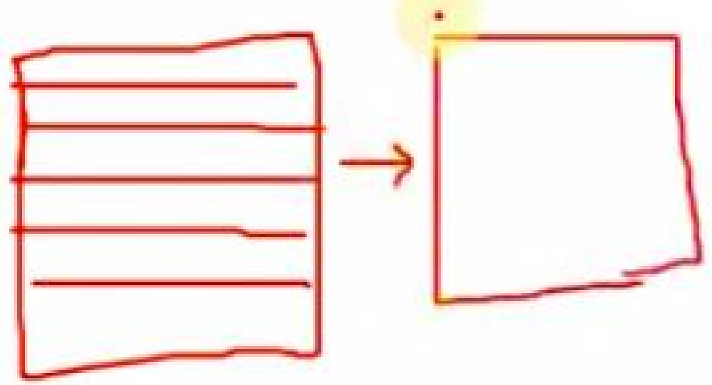
The two-dimensional unitary DFT of an NxN image $\{u(m,n)\}$ is defined as

$$v(k,l) = \frac{1}{N}\sum_{m=0}^{N-1}\sum_{n=0}^{N-1} u(m,n) W_N^{km} W_N^{ln} = \frac{1}{N}\sum_{m=0}^{N-1} W_N^{km} \cdot \sqrt{N} \cdot \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} u(m,n) W_N^{ln}$$

Two Dimensional Discrete Fourier Transform:

The two-dimensional DFT of an NxN image {u(m,n)} is defined as

U



$$v(k,l) = \sum_{m=0}^{N-1}\sum_{n=0}^{N-1} u(m,n) W_N^{km} W_N^{ln}$$

$$u(m,n) = \frac{1}{N^2}\sum_{k=0}^{N-1}\sum_{l=0}^{N-1} v(k,l) W_N^{-km} W_N^{-ln}$$

The two-dimensional unitary DFT of an NxN image {u(m,n)} is defined as

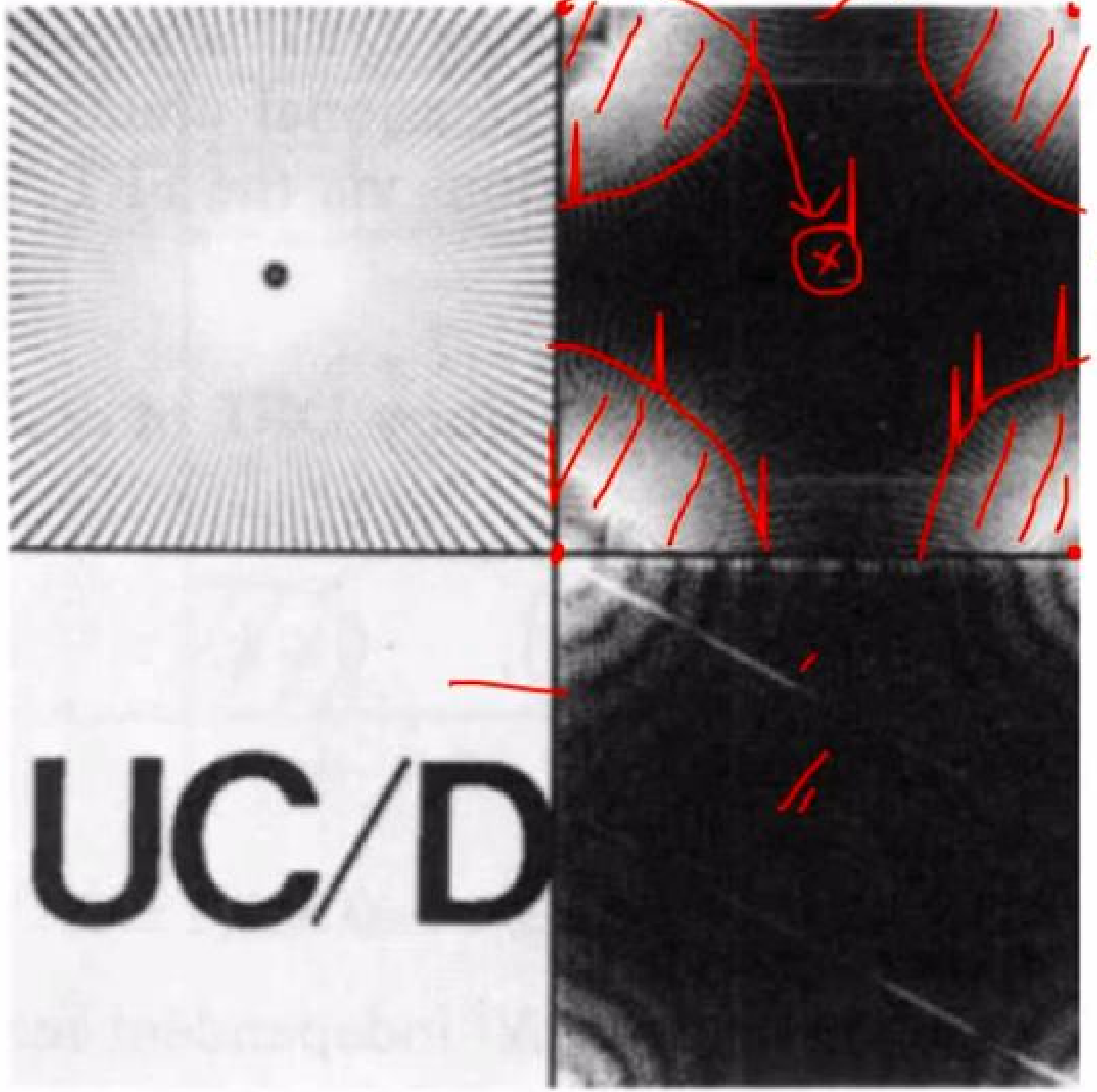
$$v(k,l) = \frac{1}{N}\sum_{m=0}^{N-1}\sum_{n=0}^{N-1} u(m,n) W_N^{km} W_N^{ln} = \frac{1}{N}\sum_{m=0}^{N-1} W_N^{km}.\sqrt{N}.\frac{1}{\sqrt{N}}\sum_{n=0}^{N-1} u(m,n) W_N^{ln}$$

Fixed

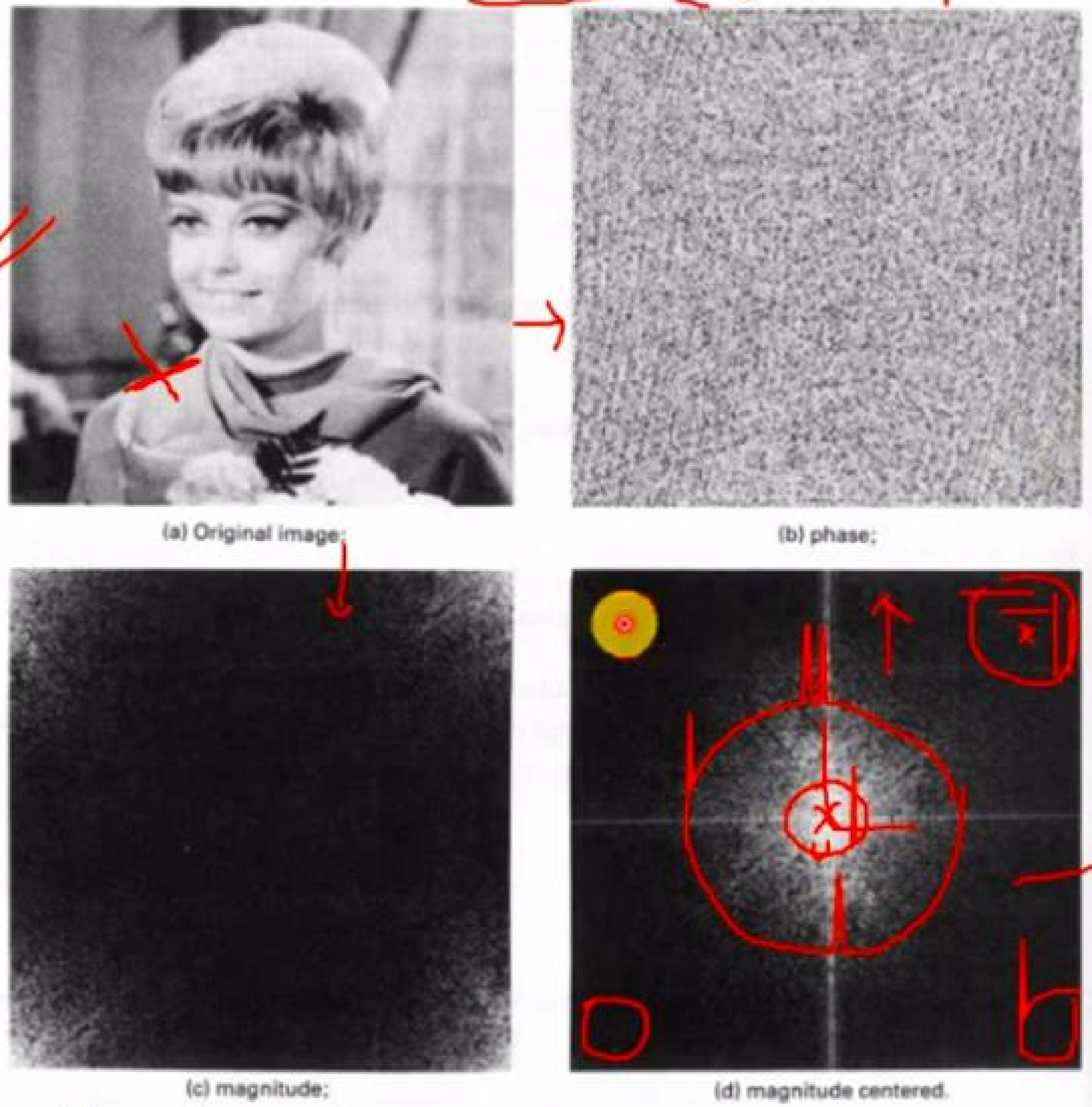
$$= \frac{1}{N}\sum_{m=0}^{N-1} W_N^{km}.\sqrt{N}.v(m,l)$$

$$= \frac{1}{\sqrt{N}}\sum_{m=0}^{N-1} W_N^{km} v(m,l)$$

Two Dimensional Discrete Fourier Transform:



Binary image



(a) Original image; (b) phase; (c) magnitude; (d) magnitude centered.

Monochrome image

One Dimensional Discrete Cosine Transform:

The DFT of a sequence $\{u(n),\ n = 0, \ldots N-1\}$ is defined as

$$v(k) = \alpha(k) \sum_{n=0}^{N-1} u(n) cos[\frac{\pi(2n+1)k}{2N}] \quad 0 \le k \le N-1$$

$$\alpha(0) = \sqrt{\frac{1}{N}} \quad \alpha(k) = \sqrt{\frac{2}{N}} \quad 1 \le k \le N-1$$

$$u(n) = \sum_{k=0}^{N-1} \alpha(k) v(k) cos[\frac{\pi(2n+1)k}{2N}] \quad 0 \le k \le N-1$$

(Walsh-) Hadamard transform

- Hadamard matrix

Kronecker operator

$$\mathbf{H}_1 = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \qquad \mathbf{H}_n = \mathbf{H}_{n-1} \otimes \mathbf{H}_1 = \frac{1}{\sqrt{2}}\begin{pmatrix} \mathbf{H}_{n-1} & \mathbf{H}_{n-1} \\ \mathbf{H}_{n-1} & -\mathbf{H}_{n-1} \end{pmatrix}$$

- Components of HT vector contain only 1 and -1
  - The number of transitions from 1 to -1 is called sequency (like ω in the continuous case)

$$\mathbf{H}_3 = ??$$

(Walsh-) Hadamard transform

- The number of transitions from 1 to -1 is called sequency (like ω in the continuous case)

$$\mathbf{H}_3 = \frac{1}{\sqrt{8}}\left[\begin{array}{cccc|cccc} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ \hline 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{array}\right]$$

(Walsh-) Hadamard transform

- Real, symmetric, and orthogonal → $\mathbf{H} = \mathbf{H}^* = \mathbf{H}^T = \mathbf{H}^{-1}$
- Fast computation (only addition is needed)
- For highly correlated images, Hadamard transform also has good energy compaction

(Walsh-) Hadamard transform

- Hadamard transform pair :

$$\begin{cases} \mathbf{v} = \mathbf{H}\mathbf{u} \\ \mathbf{u} = \mathbf{H}\mathbf{v} \end{cases} \qquad v(k) = \frac{1}{\sqrt{N}} \sum_{m=0}^{N-1} u(m)(-1)^{b(k,m)}, \qquad 0 \le k \le N-1$$

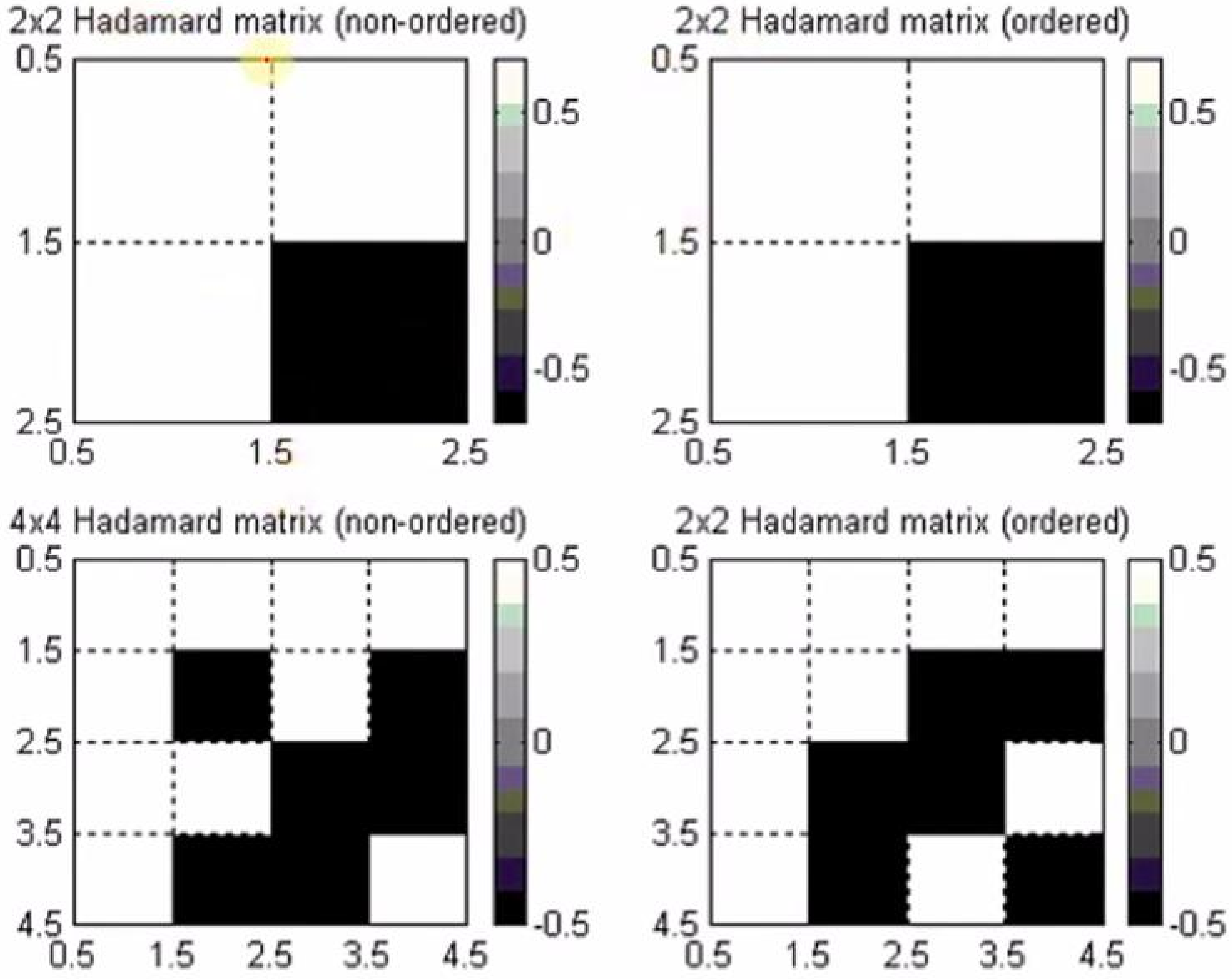
$$u(m) = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} v(k)(-1)^{b(k,m)}, \qquad 0 \le m \le N-1$$

$$b(k,m) = \sum_{i=0}^{n-1} k_i m_i \quad k_i, m_i = 0,1$$

(Walsh-) Hadamard transform

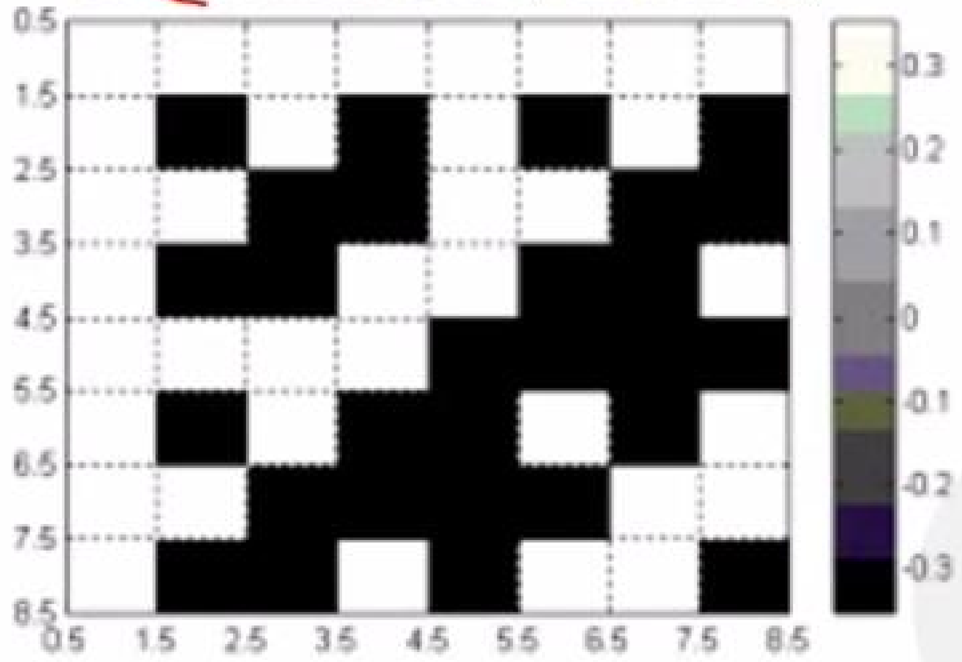
- Modified versions of the Walsh and Hadamard transforms can be formed by rearranging the rows of the transformation matrix so that the sequency increases as the index of the transform increases.
- These are called ordered transforms.
- The ordered Walsh/Hadamard transforms do exhibit the property of energy compaction whereas the original versions of the transforms do not.
- Ordered Hadamard is popular due to recursive matrix property and also energy compaction.

(Walsh-) Hadamard transform

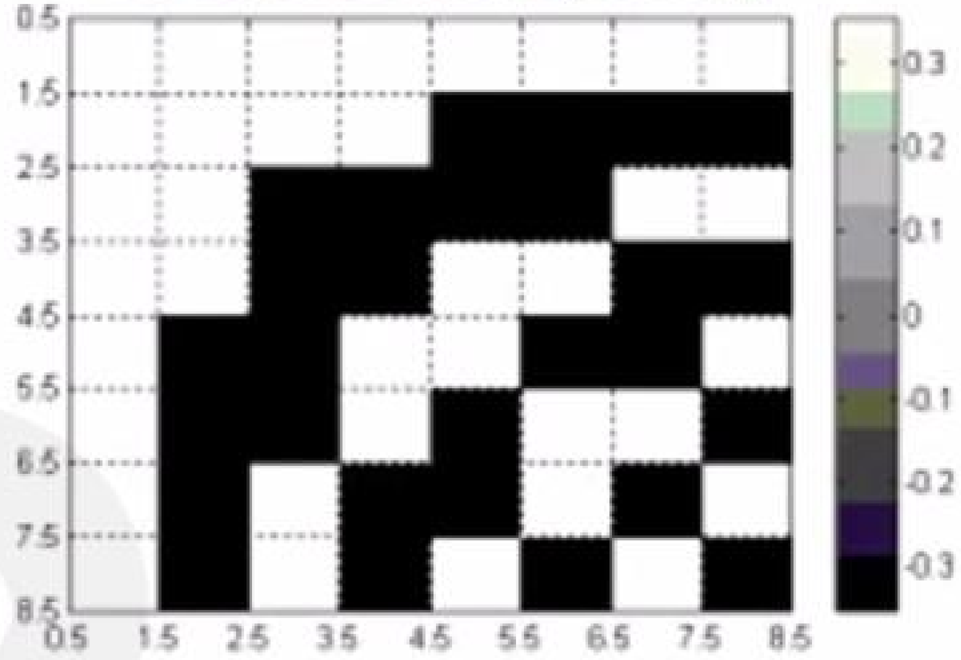


(Walsh-) Hadamard transform

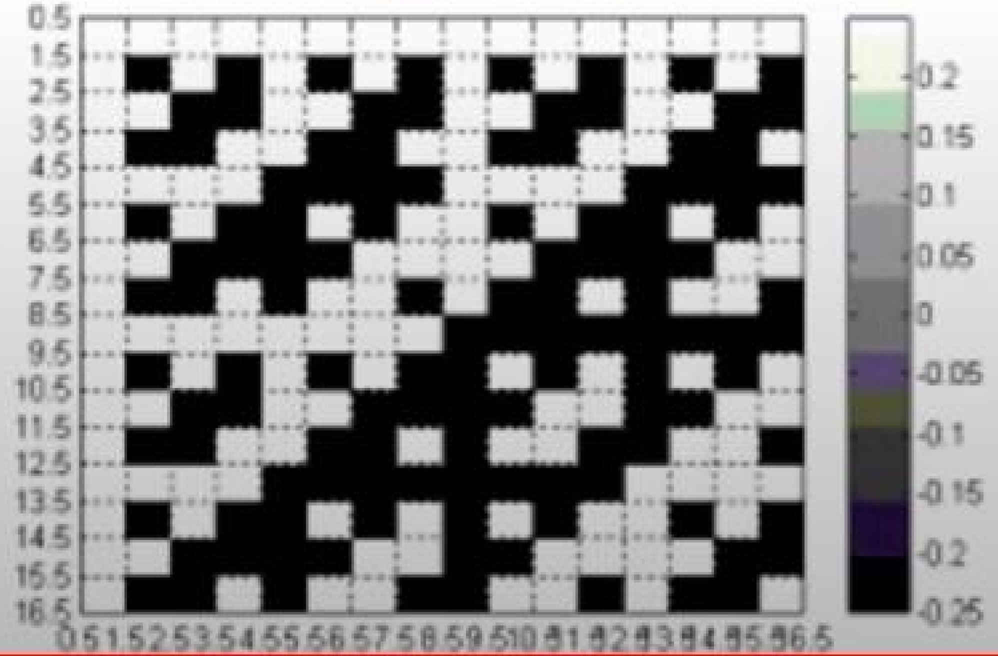
8x8 Hadamard matrix (non-ordered)



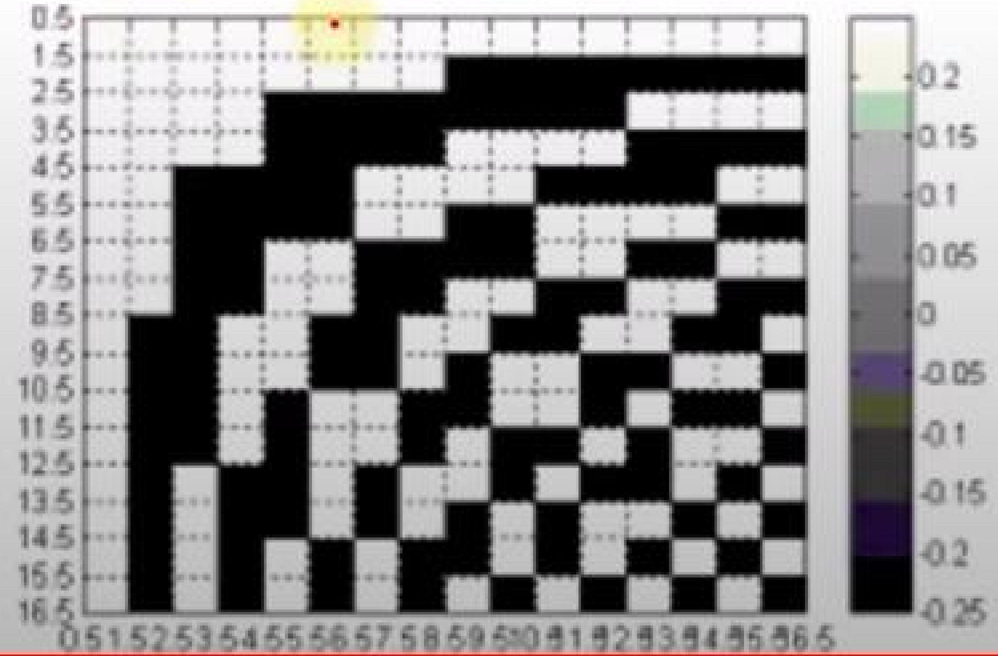
8x8 Hadamard matrix (ordered)



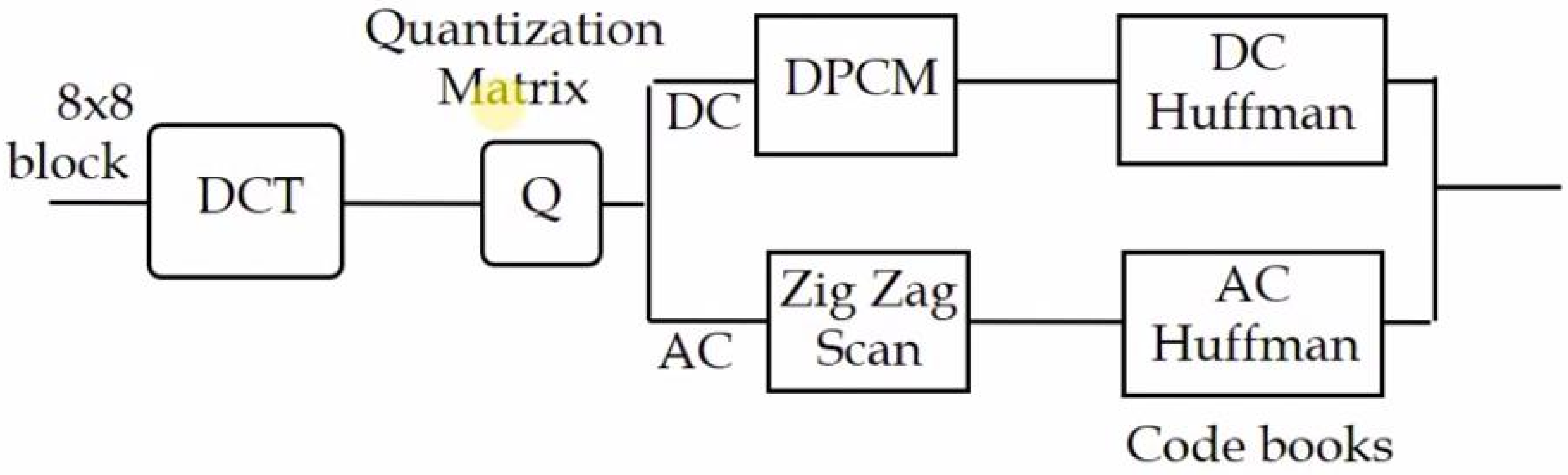
16x16 Hadamard matrix (non-ordered)



16x16 Hadamard matrix (ordered)

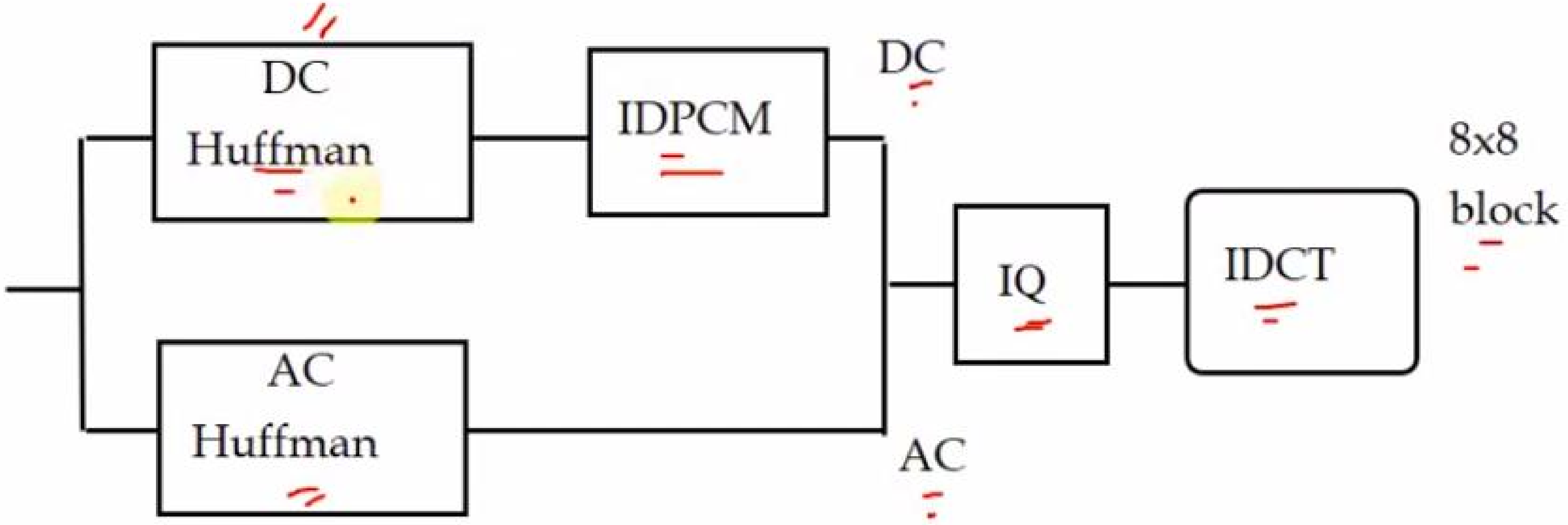


JPEG Compression



JPEG Encoding Process

JPEG Compression



JPEG Decoding Process

JPEG Compression

- DPCM

1216	1232	1224
1248	1248	1208

- DC coding: All DC coefficients of each 8 by 8 blocks of the entire image are combined to make a sequence of DC coefficients.
- Next, DPCM is applied:

  $$DiffDC(block_i) = DC(block_i) - DC(block_{i-1})$$

- Then DiffDCs will be encoded using Hoffman encoding

Example:

- Original:

  $1216 \rightarrow 1232 \rightarrow 1224 \rightarrow 1248 \rightarrow 1248 \rightarrow 1208$

- After DPCM:

  $1216 \rightarrow +16 \rightarrow -8 \rightarrow +24 \rightarrow 0 \rightarrow -40$