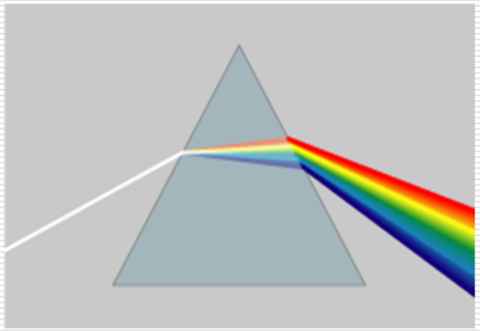
Frequency Domain Processing

- Image Transformations

ALPHA BREATHING

- Breathe in
- Breathe out
- Hold

Remember to breath, that is after all the life of all

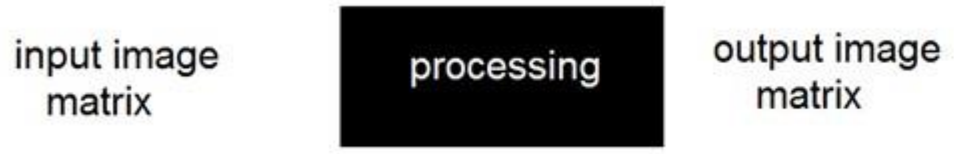


$$\begin{tikzcd} {} \arrow[r, "\text{Time}", "\text{domain}"'] & \boxed{\text{Transform}} \arrow[r, "\text{Frequency}", "\text{domain}"'] & {} \end{tikzcd}$$

Difference between spatial domain and frequency domain.

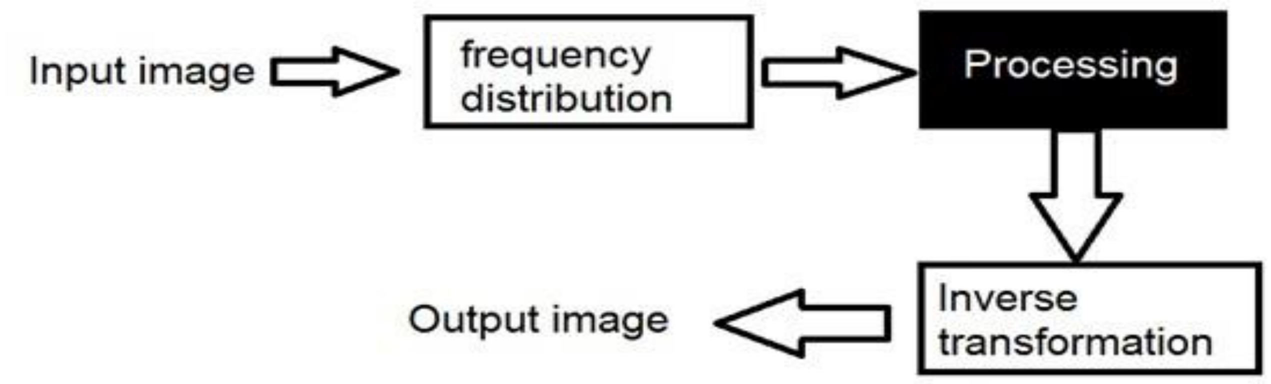
- In spatial domain , we deal with images as it is. The value of the pixels of the image change with respect to scene.
- Whereas in frequency domain , we deal with the rate at which the pixel values are changing in spatial domain.

Spatial domain



- In simple spatial domain , we directly deal with the image matrix.

Frequency domain



- We first transform the image to its frequency distribution.
- Then our black box system perform what ever processing it has to performed , and the output of the black box in this case is not an image , but a transformation.
- After performing inverse transformation , it is converted into an image which is then viewed in spatial domain.

Transformation

- A signal can be converted from time domain into frequency domain using mathematical operators called transforms.
- There are many kind of transformation that does this.
- The reason for migrating from one domain to another is to perform some tasks easily.
- Image transforms are useful for convolution and correlation operations.
- Transforms change the representation of a signal by projecting it onto a set of basis functions.

Transformation (Contd….)

- The transforms do not change the information content present in the signal.
- They give information about the frequency contents in an image.
- Transforms play a significant role in various image processing applications such as image analysis, enhancement, filtering and compression.

Need for transform

- A basic mathematical tool to represent a signal.
- <u>Mathematical convenience:</u> Every action in time domain will have an impact in the frequency domain.
- The complex convolution operation in the time domain is equal to simple multiplication operation in the frequency domain.
- <u>To extract more information:</u>

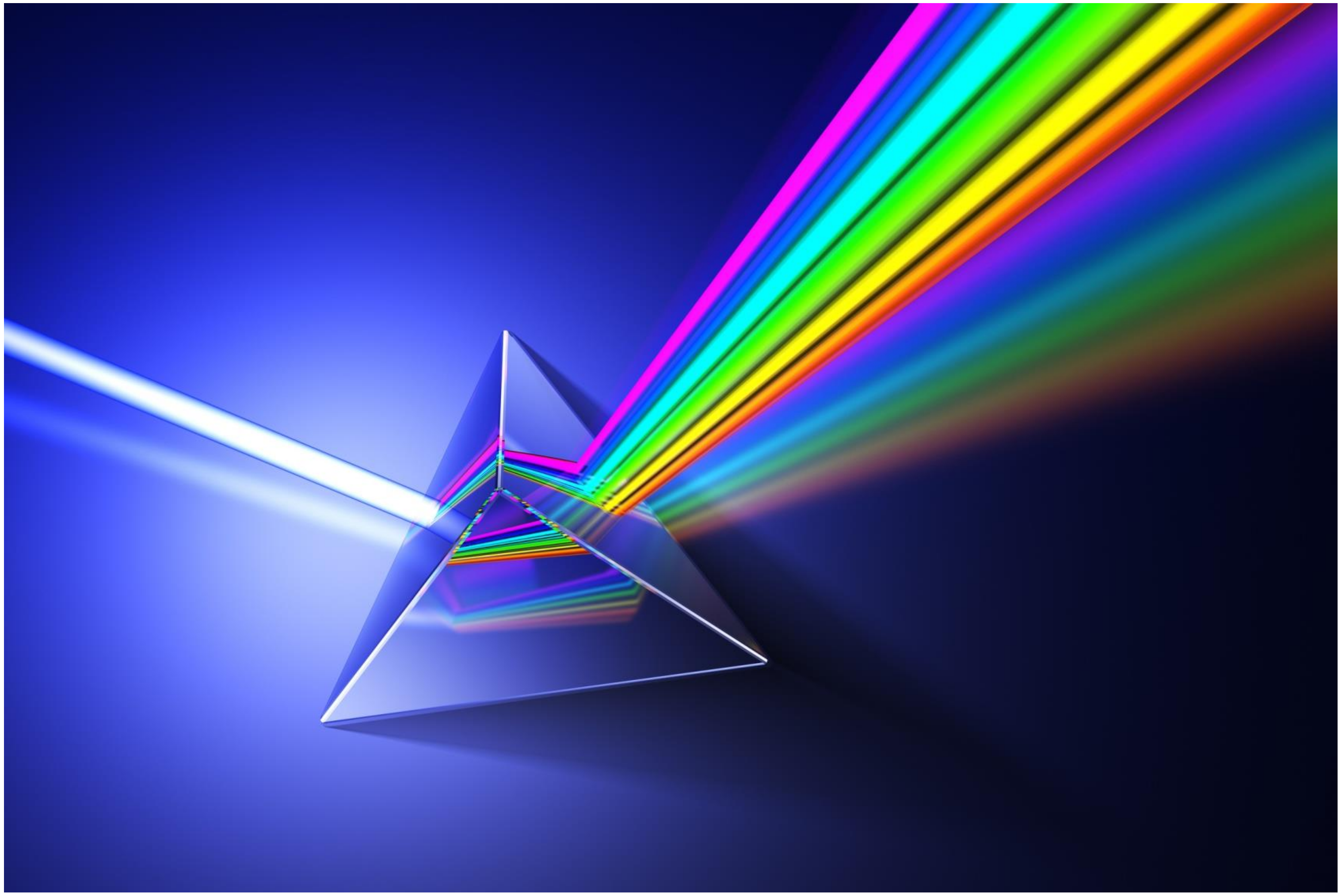


Image Transforms

- Reasons for transformation
  - It will isolate critical components of the image pattern so that they are directly accessible for analysis.
  - Transformation makes the image data in a more compact form so that they can be stored and transmitted efficiently.

Different types of image transforms

- Fourier transform
- Walsh transform
- Hadamard transform
- Slant transform
- Discrete cosine transform
- KL transform
- Radon transform
- Wavelet transform (Haar transform)

Classification of image transforms

- Transforms with orthogonal basis functions.
  - Fourier, discrete cosine and sine transforms.
    - DCT widely used in image compression.
- Transforms with non-sinusoidal orthogonal basis functions.
  - Haar, walsh, hadamard and slant transforms.
    - Haar used in represnting images in different resolutions.
- Transforms basis functions depend on the statistics of the input data.
  - KL and SVD transforms.
    - Energy compaction.
- Directional Transformation.
  - Hough, radon, ridgelet and contourlet transforms.
    - Representing the directional information of a signal.

Walsh Transform

- Fourier analysis is basically the representation of a signal by a set of orthogonal sinusoidal waveforms.
- Walsh in 1923 introduced a complete set of orthonormal square-wave functions to represent these functions.
- Walsh function – computationally simple.
- They are real.
- Take only two values +1 or -1.

Walsh Transform

In 1-D case we have : $g(x,u)=\frac{1}{N}\prod_{i=0}^{n-1}(-1)^{b_i(x)b_{n-1-i}(u)}$ 1-D kernel

$n= \log_2 N$

Find the 1D Walsh Basis for the fourth-order system. (N=4).

Solution:

Here N=4.
$n=\log_2 N = 2$

x and u will vary from 0 to N-1. So, here x and u have values 0,1,2,3.

Walsh Transform

$$g(x,u)=\frac{1}{N}\prod_{i=0}^{n-1}(-1)^{b_i(x)b_{n-1-i}(u)}$$ 1-D kernel

$n = \log_2 N$

I varies from 0 to n-1. So I has values of 0 and 1.

The construction of Walsh basis for N=4 is given in table below:

Walsh Transform

- Shortcut method for finding the Walsh transform basis:
- All values in Walsh basis will be having the value 1/N.
- Only sign will be + or -.
- How to determine the sign?
- Steps:
- 1.Write the binary representation of n.
- 2.Write the binary representation of k in reverse order
- 3. Check for the number of overlaps of 1 between n and k.
- 4. If the number of 1s is
  - i)zero, then sign is positive
  - ii) even , then positive
  - Iii) odd, then negative.

Walsh Transform

This figure shows the basis functions (kernels) as a function of u and v (excluding the 1/N constant term) for computing the Walsh transform when N=4. Each block corresponds to varying x and y form 0 to 3 (that is, 0 to N-1), while keeping u and v fixed at the values corresponding to that block. Thus each block consists of an array of 4×4 binary elements (White means "+1" and Black means "-1"). To use these basis functions to compute the Walsh transform of an image of size 4×4 simply requires obtaining W(0,0) by multiplying the image array point-by-point with the 4×4 basis block corresponding to u=0 and v=0, adding the results, and dividing by 4, and continue for other values of u and v.

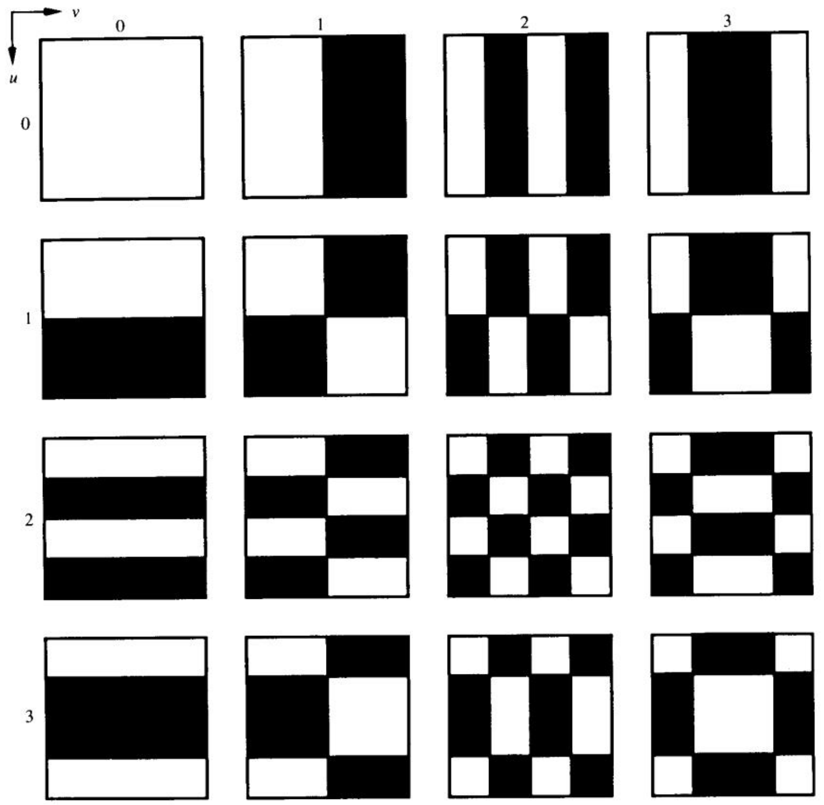


Figure 3.25 *Walsh basis functions for N = 4. Each block consists of 4 × 4 elements, corresponding to x and y varying from 0 to 3. The origin of each block is at its top left. White and black denote +1 and −1, respectively.*

Hadamard Transform

When $N=2^n$, the 2-D forward and inverse Hadamard kernels are given by the relations

$$g(x,y,u,v)=\frac{1}{N}\prod_{i=0}^{n-1}(-1)^{[b_i(x)b_i(u)+b_i(y)b_i(v)]} \quad \text{and} \quad h(x,y,u,v)=\frac{1}{N}\prod_{i=0}^{n-1}(-1)^{[b_i(x)b_i(u)+b_i(y)b_i(v)]}$$

Where $b_k(z)$ is the *k*th bit in the binary representation of z.

$$H(u,v)=\frac{1}{N}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f(x,y)(-1)^{\sum_{i=0}^{n-1}[b_i(x)b_i(u)+b_i(y)b_i(v)]}$$

So the forward and inverse Hadamard transforms are equal in form; that is:

$$f(x,y)=\frac{1}{N}\sum_{u=0}^{N-1}\sum_{v=0}^{N-1} H(u,v)(-1)^{\sum_{i=0}^{n-1}[b_i(x)b_i(u)+b_i(y)b_i(v)]}$$

Hadamard Transform

This figure shows the basis functions (kernels) as a function of u and v (excluding the 1/N constant term) for computing the Hadamard transform when N=4. Each block corresponds to varying x and y form 0 to 3 (that is, 0 to N-1), while keeping u and v fixed at the values corresponding to that block. Thus each block consists of an array of 4×4 binary elements (White means "+1" and Black means "-1") like Walsh transform. If we compare these two transforms we can see that they only differ in the sense that the functions in Hadamard transform are ordered in increasing sequency and thus are more "natural" to interpret.

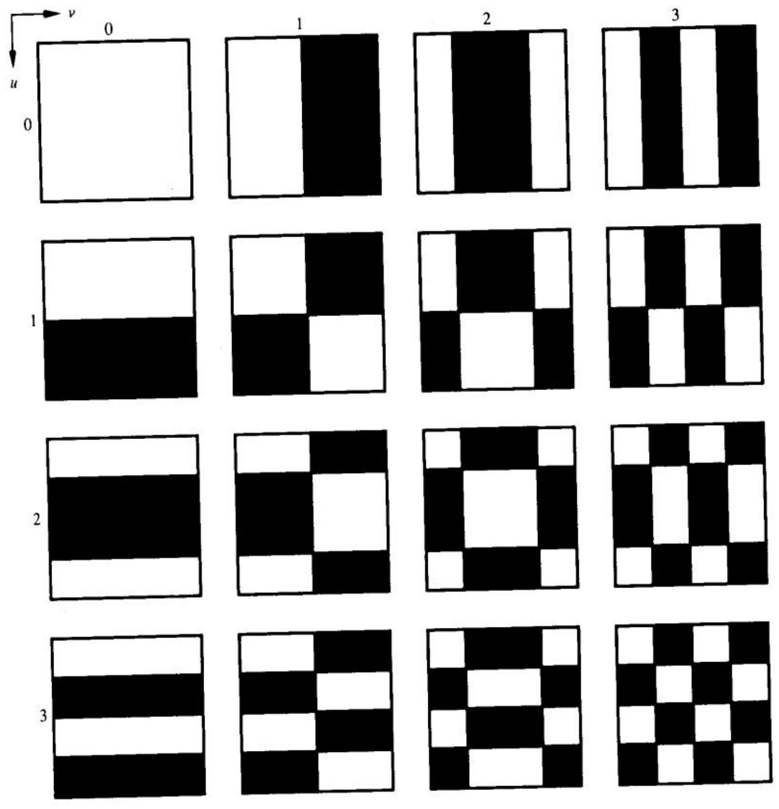


Figure 3.26 Ordered Hadamard basis functions for N = 4. Each block consists of 4 × 4 elements, corresponding to x and y varying from 0 to 3. The origin of each block is at its top left. White and black denote +1 and −1, respectively.

Discrete Cosine Transform

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

for $u, v = 0, 1, 2, \ldots, N - 1$, and

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

for $x, y = 0, 1, 2, \ldots, N - 1$, where α is

$$\alpha(u) = \begin{cases} \sqrt{\dfrac{1}{N}} & \text{for } u = 0 \\ \sqrt{\dfrac{2}{N}} & \text{for } u = 1, 2, \ldots, N - 1. \end{cases}$$

Discrete Cosine Transform (DCT)

Each block consists of 4×4 elements, corresponding to x and y varying from 0 to 3. The highest value is shown in white. Other values are shown in grays, with darker meaning smaller.

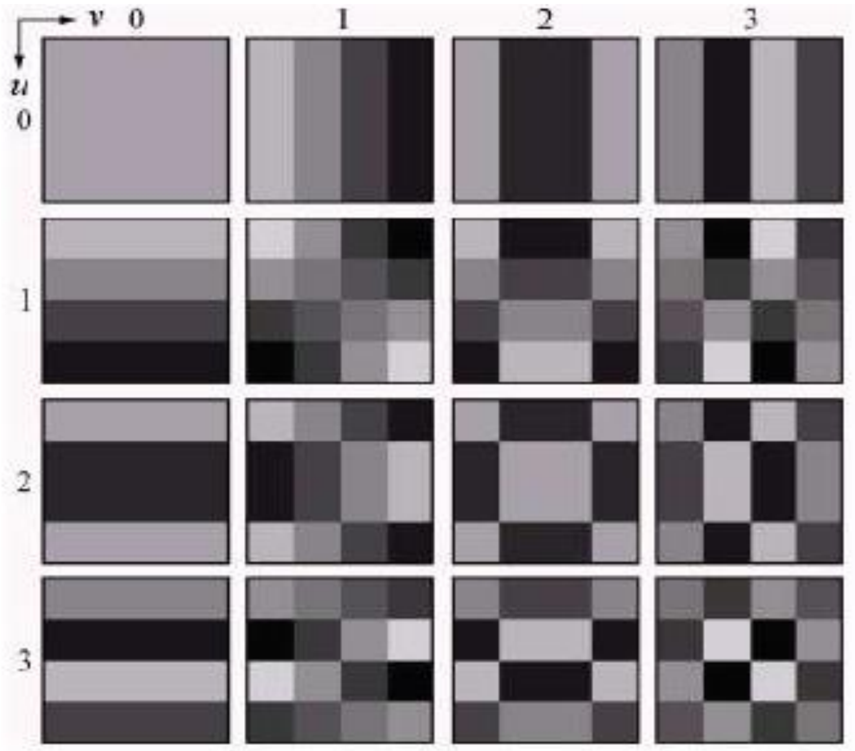
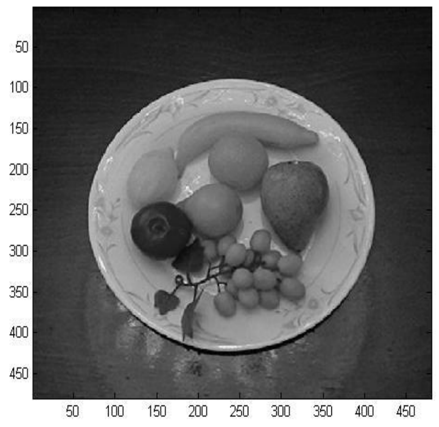


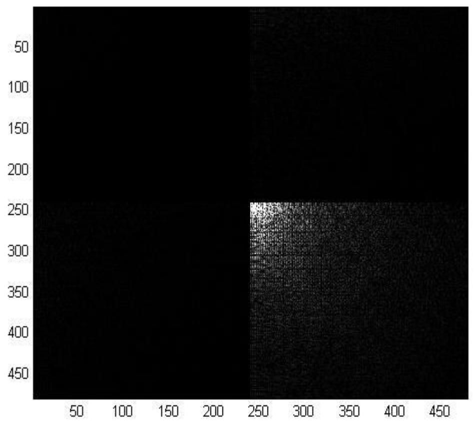
FIGURE 8.30 Discrete-cosine basis functions for $N = 4$. The origin of each block is at its top left.

Discrete Cosine Transform

Example

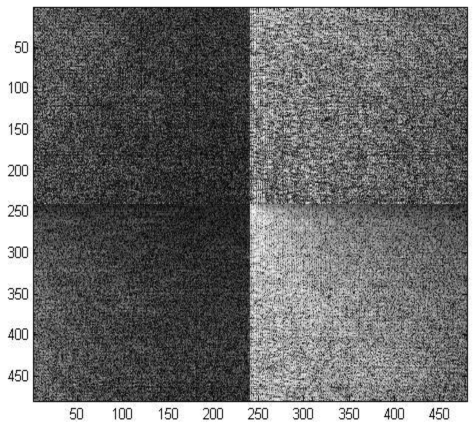


Main Image (Gray Level)



DCT of Main image

(Cosine spectrum)

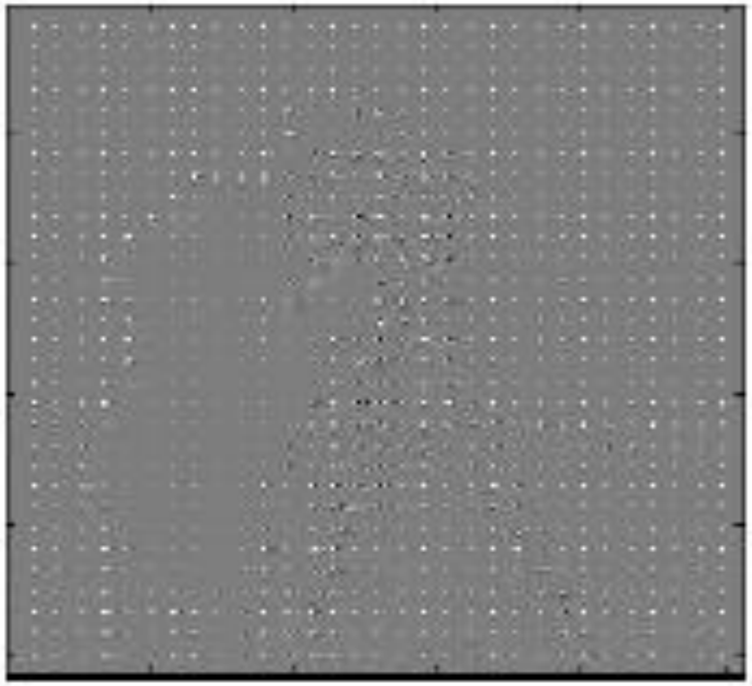


Logarithmic scaled

of the Cosine spectrum

Blockwise DCT Example

Original *Cameraman* 256x256


Blockwise 8x8 DCT of *Cameraman*

Haar Transform

The Haar transform is based on the Haar functions, $h_k(z)$, which are defined over the continuous, closed interval [0,1] for z, and for k=0,1,2,…,N-1, where $N=2^n$. The first step in generating the Haar transform is to note that the integer *k* can be decomposed uniquely as $k=2^p+q-1$

where $0\le p\le n-1$, q=0 or 1 for p=0, and $1\le q\le 2^p$ for $p\neq 0$.

With this background, the Haar functions are defined as

$$h_0(z) \overset{\Delta}{=} h_{00}(z) = \frac{1}{\sqrt{N}} \qquad \text{for } z \in [0,1]$$

and

$$h_k(z) \overset{\Delta}{=} h_{00}(z) = \frac{1}{\sqrt{N}} \begin{cases} 2^{p/2} & \frac{q-1}{2^p} \le z \prec \frac{q-1/2}{2^p} \\ -2^{p/2} & \frac{q-1/2}{2^p} \le z \prec \frac{q}{2^p} \\ 0 & \text{otherwise for } z \in [0,1] \end{cases}$$

Haar Transform

- Algorithm to generate Haar basis is given below:
- Step 1: Determine the order N of the Haar basis.
- Step 2: Determine n where $n = \log_2 N$.
- Step 3: Determine p and q.
  - (i) $0 \leq p < n-1$
  - (ii) If $p=0$ then $q = 0$ or $q = 1$
  - (iii) If $p \neq 0$, $1 \leq q \leq 2^p$.

Haar Transform

- Step 4: Determine k.

  $k=2^P + q - 1.$

- Step 5: Determine Z.

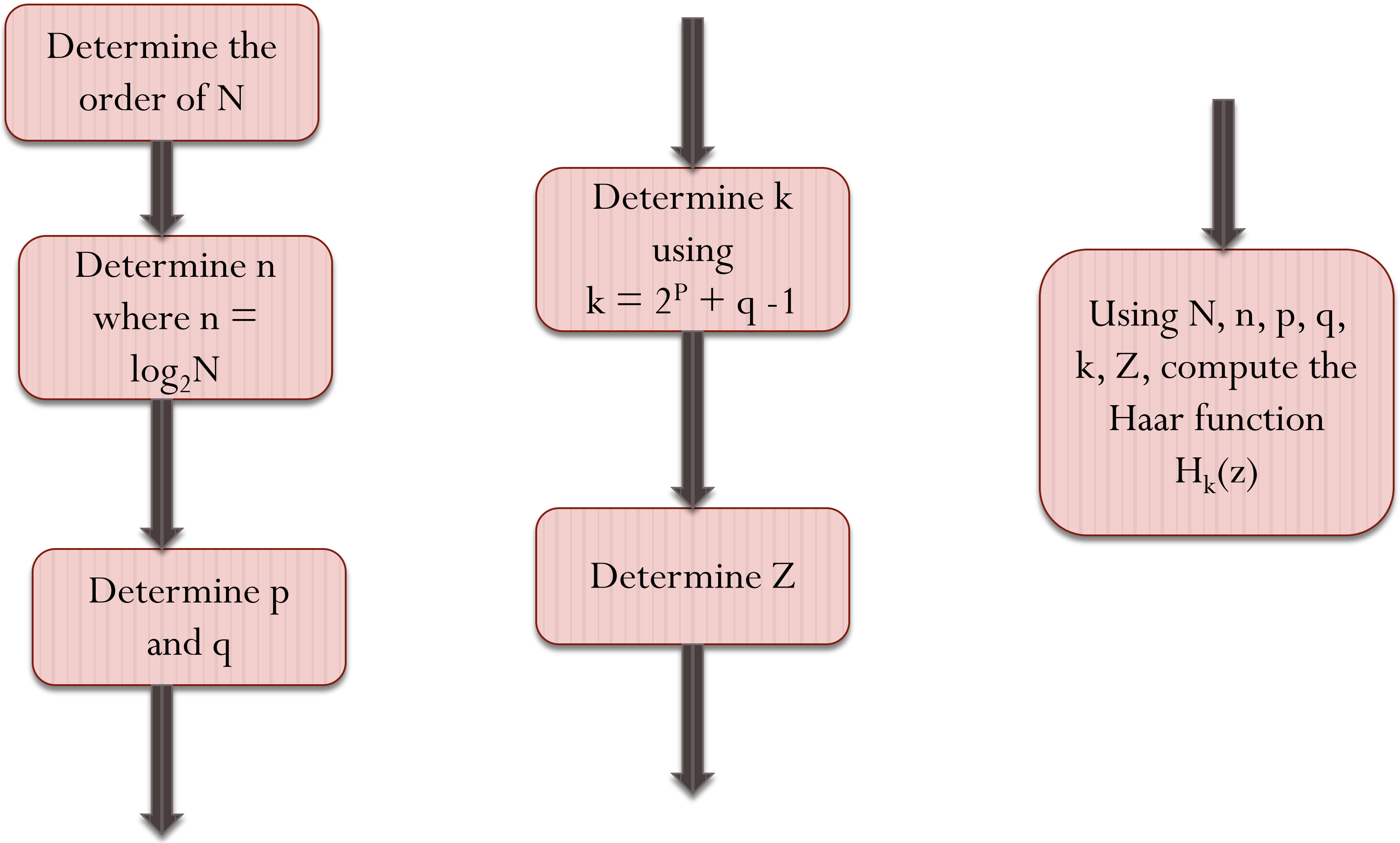
  $Z \rightarrow [0,1) \rightarrow \{0/N, 1/N, \ldots., (N-1)/N\}$

- Step 6:
- If $k=0$ then $H(Z) = 1/\sqrt{N}$

Otherwise,

- $H_k(Z) = H_{pq}(Z) = 1/\sqrt{N} * A$

Haar Transform Flowchart



Determine the order of N
Determine n where n = $\log_2 N$
Determine p and q
Determine k using $k = 2^p + q - 1$
Determine Z
Using N, n, p, q, k, Z, compute the Haar function $H_k(z)$

Haar Transform

- Haar transform matrix for sizes *N=2,4,8*

$$Hr_2 = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

$$Hr_4 = \frac{1}{\sqrt{4}}\begin{bmatrix} 1 & 1 & \sqrt{2} & 0 \\ 1 & 1 & -\sqrt{2} & 0 \\ 1 & -1 & 0 & \sqrt{2} \\ 1 & -1 & 0 & -\sqrt{2} \end{bmatrix}$$

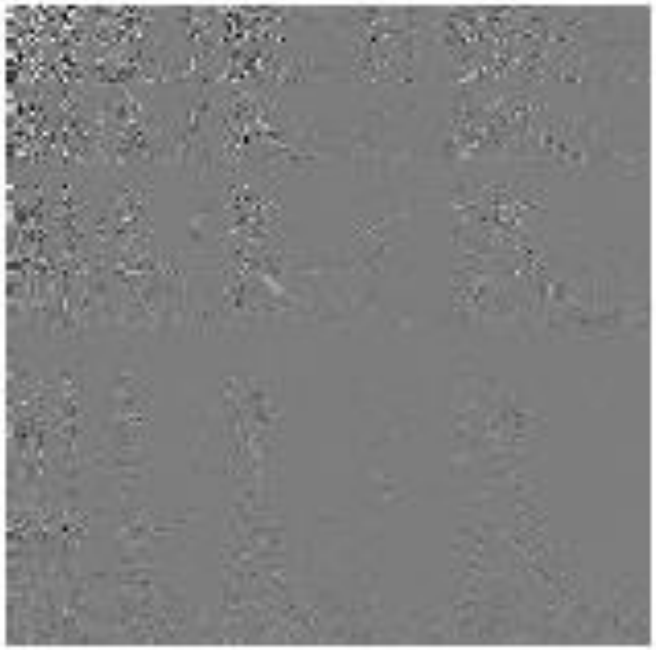
$$Hr_8 = \frac{1}{\sqrt{8}}\begin{bmatrix} 1 & 1 & \sqrt{2} & 0 & 2 & 0 & 0 & 0 \\ 1 & 1 & \sqrt{2} & 0 & -2 & 0 & 0 & 0 \\ 1 & 1 & -\sqrt{2} & 0 & 0 & 2 & 0 & 0 \\ 1 & 1 & -\sqrt{2} & 0 & 0 & -2 & 0 & 0 \\ 1 & -1 & 0 & \sqrt{2} & 0 & 0 & 2 & 0 \\ 1 & -1 & 0 & \sqrt{2} & 0 & 0 & -2 & 0 \\ 1 & -1 & 0 & -\sqrt{2} & 0 & 0 & 0 & 2 \\ 1 & -1 & 0 & -\sqrt{2} & 0 & 0 & 0 & -2 \end{bmatrix}$$

Can be computed by taking sums and differences.

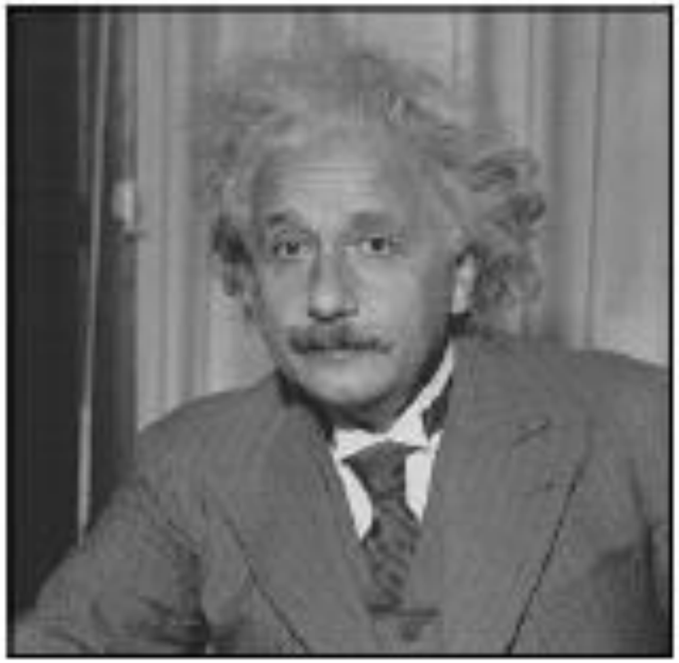
Fast algorithms by recursively applying **Hr$_2$.**

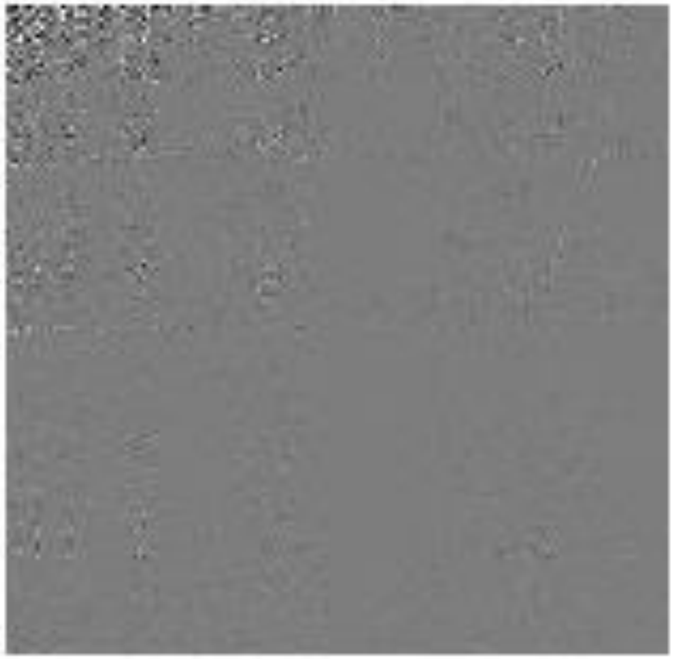
Haar Transform Example

Original *Cameraman* 256x256


256x256 Haar transform of *Cameraman*

Haar Transform Example


Original *Einstein* 256x256


256x256 Haar transform of *Einstein*

Slant Transform

- The Slant Transform matrix of order N*N is the recursive expression

$$\mathbf{S}_N = \frac{1}{\sqrt{2}} \left[\begin{array}{cc|c|cc|c} 1 & 0 & & 1 & 0 & \\ & & \mathbf{0} & & & \mathbf{0} \\ a_N & b_N & & -a_N & b_N & \\ \hline \multicolumn{2}{c|}{\mathbf{0}} & \mathbf{I}_{(N/2)-2} & \multicolumn{2}{c|}{\mathbf{0}} & \mathbf{I}_{(N/2)-2} \\ \hline 0 & 1 & & 0 & -1 & \\ & & \mathbf{0} & & & \mathbf{0} \\ -b_N & a_N & & b_N & a_N & \\ \hline \multicolumn{2}{c|}{\mathbf{0}} & \mathbf{I}_{(N/2)-2} & \multicolumn{2}{c|}{\mathbf{0}} & -\mathbf{I}_{(N/2)-2} \end{array} \right] \left[\begin{array}{c|c} \mathbf{S}_{N/2} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{S}_{N/2} \end{array} \right]$$

Slant Transform

Where I is the identity matrix, and

$$S_2 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

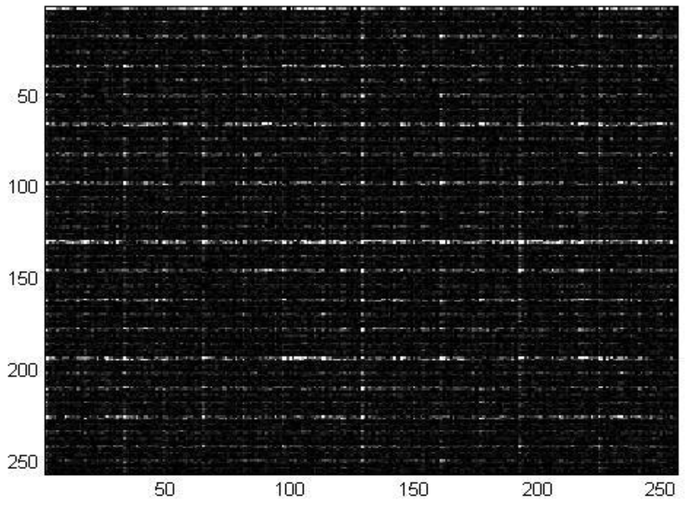
$$a_N = \left[\frac{3N^2}{4(N^2 - 1)} \right]^{1/2}$$

$$b_N = \left[\frac{N^2 - 4}{4(N^2 - 1)} \right]^{1/2}$$

Slant Transform

Example

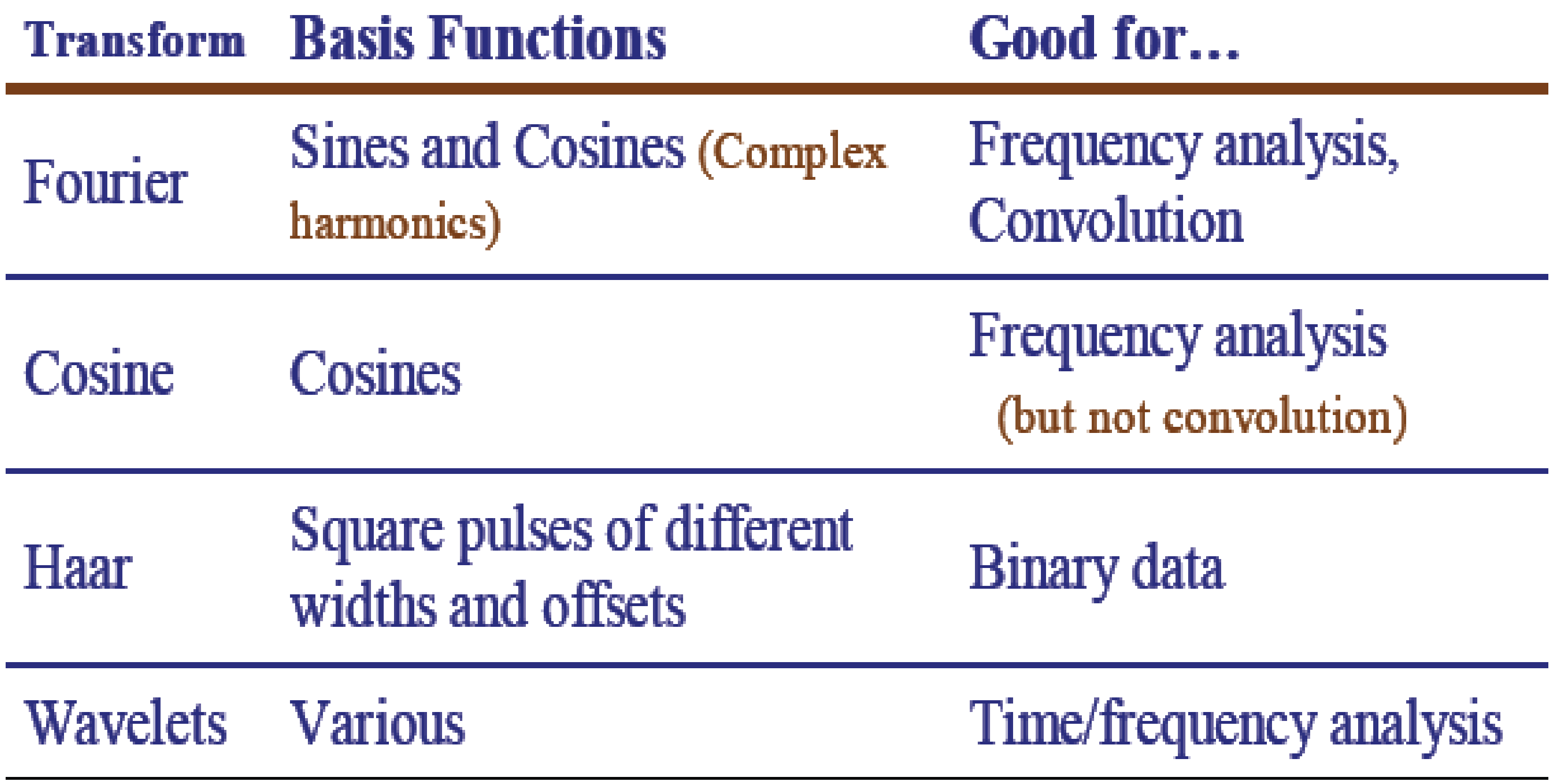
Main Image (Gray Level)



Slant Transform of Main image

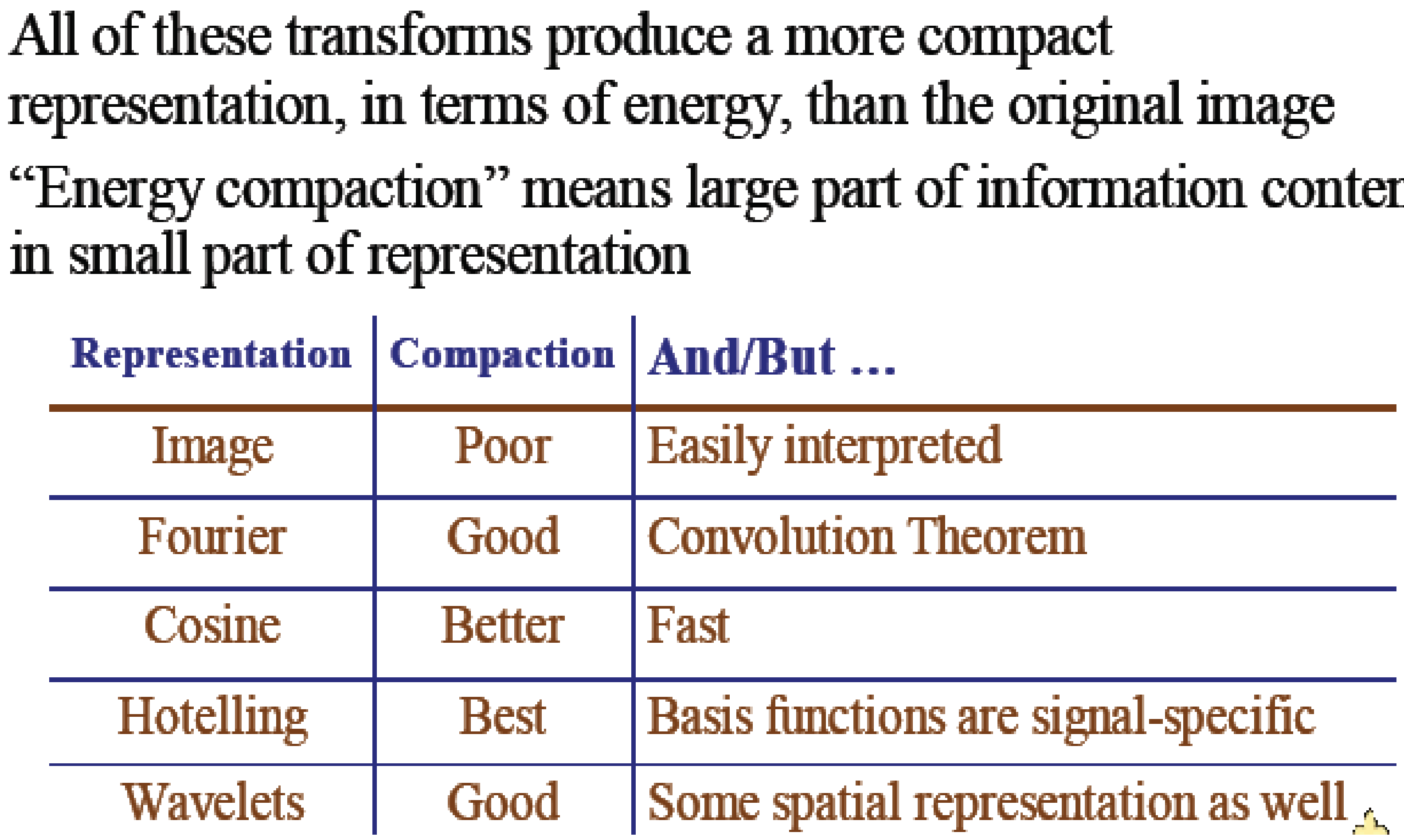
(Slant spectrum)

Comparison Of Various Transforms



Transform	Basis Functions	Good for…
Fourier	Sines and Cosines (Complex harmonics)	Frequency analysis, Convolution
Cosine	Cosines	Frequency analysis (but not convolution)
Haar	Square pulses of different widths and offsets	Binary data
Wavelets	Various	Time/frequency analysis

Comparison Of Various Transforms



All of these transforms produce a more compact representation, in terms of energy, than the original image

"Energy compaction" means large part of information content in small part of representation

Representation	Compaction	And/But ...
Image	Poor	Easily interpreted
Fourier	Good	Convolution Theorem
Cosine	Better	Fast
Hotelling	Best	Basis functions are signal-specific
Wavelets	Good	Some spatial representation as well