3.4 The Fast Fourier Transform (FFT)

- implementation of DFT :

$$F(u) = \frac{1}{N}\sum_{x=0}^{N-1} f(x)\exp[-j2\pi ux/N]$$

  - no. of complex multiplications and additions : proportional to N^2

$\downarrow$

$N\log_2 N$ By proper decomposition (called FFT algorithm)

Table 3.1

N	N^2 (*Direct* FT)	$N\log_2 N$ (FFT)	*Computational Advantage* ($N/\log_2 N$)
2	4	2	2.00
4	16	8	2.00
8	64	24	2.67
16	256	64	4.00
32	1,024	160	6.40
64	4,096	384	10.67
128	16,384	896	18.29
256	65,536	2,048	32.00
512	262,144	4,608	56.89
1024	1,048,576	10,240	102.40
2048	4,194,304	22,528	186.18
4096	16,777,216	49,152	341.33
8192	67,108,864	106,496	630.15

- FFT approach :
  - significant saving in computational effort

3.4.1 FFT Algorithm

- **Successive doubling method : "decomposition"**

$F(u) = \frac{1}{N}\sum_{x=0}^{N-1} f(x)\, W_N^{ux}$ where $W_N^{ux} = \exp[-j2\pi ux / N]$

N is assumed to be $N=2^n$, n is positive integer

Let N=2M

$$F(u) = \frac{1}{2M}\sum_{x=0}^{2M-1} f(x)\, W_{2M}^{ux}$$

$$= \frac{1}{2}[\frac{1}{M}\sum_{x=0}^{M-1} f(2x)\, W_{2M}^{u2x} + \frac{1}{M}\sum_{x=0}^{M-1} f(2x+1)\, W_{2M}^{u(2x+1)}]$$

$$= \frac{1}{2}[\frac{1}{M}\sum_{x=0}^{M-1} f(2x)\, W_M^{ux} + \frac{1}{M}\sum_{x=0}^{M-1} f(2x+1)\, W_M^{ux}\, W_{2M}^{u}] \quad (\Theta \;\; W_{2M}^{2ux} = W_M^{ux})$$

let us define

$F_{even}(u) = \frac{1}{M}\sum_{x=0}^{M-1} f(2x)\, W_M^{ux}$, $F_{odd}(u) = \frac{1}{M}\sum_{x=0}^{M-1} f(2x+1)\, W_M^{ux}$ for u=0,…, M-1

then, $F(u) = \frac{1}{2}[F_{even}(u) + F_{odd}(u)\, W_{2M}^{u}]$

$$F(u+M) = \frac{1}{2}[F_{even}(u+M) + F_{odd}(u+M)\, W_{2M}^{u+M}]$$

$$= \frac{1}{2}[F_{even}(u) + F_{odd}(u)\, W_{2M}^{u}(-1)] \quad (\Theta \;\; W_M^{u+M} = W_M^{u} \text{ and } W_{2M}^{u+M} = -W_{2M}^{u})$$

- N point transform can be computed by dividing the original expression into two parts.
- Computation of the first half of *F(u)* requires evaluation of two (N/2) point transform, $F_{even}(u), F_{odd}(u)$.
- The other half follows directly from F(u+M) without additional transform evaluations

- Computation implication : (complexities)

Let us define:

m(n)=number of multiplication, one multiplication =M
a(n) = number of addition, one addition =A
number of sample = 2^n.

- At n=1,
  - $N=2^1=2$ points, $M=N/2=1$
  - Remind $F(u)=\frac{1}{N}\sum_{x=0}^{N-1} f(x)\,W_N^{ux}$.
  - Two point transform F(0), F(1)
  - By successive doubling algorithm, to obtain *F(0)*, compute $F_{even}(0), F_{odd}(0)$,first.

  → $F_{even}(u)\big|_{M=1} = \frac{1}{1}\sum_{x=0}^{1-1} f(2x)\,W_1^{ux}$ => one point transform : no addition and multiplication.

  → Same as $F_{odd}(u)\big|_{M=1}$

  → $F(0)=\frac{1}{2}[F_{even}(0)+F_{odd}(0)\,W_2^0]$ =>1A(one addition) and 1M(one multiplication).

  → $F(0+M)=F(1)=\frac{1}{2}[F_{even}(0)-F_{odd}(0)\,W_2^0]$ => 1A(one more addition).

  m(1)=1M, a(1)=2A

- At n=2, (4-pt transform) → 2, 2-pt transforms
  $N=2^2=4$ points, $M=N/2=2$

  $F_{even}(u)\big|_{M=2} = \frac{1}{2}\sum_{x=0}^{2-1} f(2x)\,W_2^{ux}$ => two point transform. # of calculations are m(1)+a(1).

  Same as $F_{odd}(u)\big|_{M=2}$ =>.# of calculations are m(1)+a(1).

  Further,

  → $F(0)=\frac{1}{2}[F_{even}(0)+F_{odd}(0)\,W_{2\cdot 2}^0]$ => 1A and 1M

→ $F(1) = \frac{1}{2}[F_{even}(1) + F_{odd}(0)\, W_{2\cdot 2}^{1}]$ =>1A and 1M

→ $F(0+M) = F(2) = \frac{1}{2}[F_{even}(0) - F_{odd}(0)\, W_{2\cdot 2}^{0}]$ => 1A

→ $F(1+M) = F(3) = \frac{1}{2}[F_{even}(1) - F_{odd}(1)\, W_{2\cdot 2}^{1}]$ => 1A

$\therefore$ m(2)=2m(1)+2, a(2)=2a(1)+4

- At n=3, (8-pt transform) → 2, 4-pt transforms
  $N=2^3=8$ points, M=N/2=4

  $F_{even}(u)\big|_{M=3}$ =>4points transform
  Therefore, For $F_{even}(u)\big|_{M=3}$, $F_{odd}(u)\big|_{M=3}$, # of calculations are 2m(2)+2a(2).
  For F(u),F(u+M)|u=0,..3, # of calculations are 4M and 8A.
  $\therefore$ m(3)=2m(2)+4, a(3)=2a(2)+8

Continuing this, For any positive integer value n.

m(n) = $2m(n-1) + 2^{n-1}$

a(n) = $2a(n-1) + 2^{n}$ where, m(0) = a(0)=0.

3.4.2 Number of operations

- m(n) = $\frac{1}{2}2^n \log_2 2^n = \frac{1}{2}N\log_2 N = \frac{1}{2}Nn$ -----(1)

  a(n) = $2^n \log_2 2^n = N\log_2 N = Nn$ ------(2)

- proof by induction :

  - at n=1, m(1)=1/2(2)(1)=1, a(1)=(2)(1)=2
    Next, the assumption is made that expression hold for n.
  - Prove the truth for n+1.

    m(n+1) = $2m(n)+ 2^n$
    substituting (1),
    m(n+1) = $2(1/2)Nn + 2^n$

$=2(1/2)\ 2^n\ n+ 2^n$

$=2^n\ (n+1) = (1/2)(\ 2^{n+1})(n+1)$

therefore (1) is valid for all positive integer n.

$a(n+1) = 2a(n)+ 2^{n+1}$

substituting (2),

$a(n+1) = 2Nn + 2^{n+1}$

$=2\ 2^n\ n+ 2^{n+1}$

$=2^{n+1}\ (n+1)$

therefore (2) is valid for all positive integer n.

3.4.4 Implementation

- For eight points {f(0),f(1),f(2),f(3),f(4),f(5),f(6),f(7)} :

  - In the successive doubling algorithm.
  - Divide two parts; even and odd.
    - Even: {f(0),f(2),f(4),f(6)} Odd:{f(1),f(3),f(5),f(7)}
    - Each even and odd functions are considered as 4 point transform.
    - So divide further even-even, even-odd, odd-even and odd-odd.
  - Then {f(0),f(4)} {f(2),f(6)} {f(1),f(5)} {f(3),f(7)}
    → 'bit reversal' rule

3.5 Other Separable Image Transforms

- general relation :
  - $T(u) = \sum_{x=0}^{N-1} f(x)g(x,u)$; signal decomposition

  where T(u) : transform of f(x)
  g(x,u) : forward transformation kernel
  - the inverse transform ;

$$f(x) = \sum_{x=0}^{N-1} T(u)h(x,u)$$

  where h(x,u) : inverse transformation kernel

- 2-D square array :
  - $T(u,v) = \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x,y)g(x,y,u,v)$

$$f(x,y) = \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} T(u,v)h(x,y,u,v)$$

  - kernel : g(x,y,u,v), h(x,y,u,v)
    - ✓ depend on the indexes x, y, u, v not on the value of f(x, y) or T(u, v)
    - ✓ viewed as the basis function of a series expansion

- properties of kernel :
  - forward kernel ;
    - ✓ separable

$$g(x,y,u,v) = g_1(x,u)g_2(y,v)$$

    - ✓ symmetric

$$g(x,y,u,v) = g_1(x,u)g_1(y,v)$$

  ex) $g(x,y,u,v) = \frac{1}{N}\exp[-2\pi(ux+vy)/N]$

$$= \frac{1}{\sqrt{N}} \exp[-j2\pi ux / N] \frac{1}{\sqrt{N}} \exp[-j2\pi vy / N]$$

$$= g_1(x,u) g_1(y,v)$$

- transform with a separable kernel :

  - computed in two steps, each 1-D transform
  - first, 1-D transform along each row of f(x,y)

    $$T(x,v) = \sum_{x=0}^{N-1} f(x,y) g_2(y,v)$$
  - next, 1-D transform along each row of T(x,v)

    $$T(u,v) = \sum_{x=0}^{N-1} T(x,v) g_1(x,u)$$

- matrix form :

  - if kernel g(x, y, u, v) : separable and symmetric

    T=AFA

    Where **F** : N×N image matrix

    A : N×N symmetric transformation matrix with elements $a_{ij} = g_1(i,j)$

    T : resulting N×N transform

  - inverse transform ;

    BTB = **BAFAB**

    If **B** = $\mathbf{A}^{-1}$

    F = **BTB**

    If **B** ≠ $\mathbf{A}^{-1}$

    Approximation $\hat{\mathbf{F}}$ = **BAFAB**

3.5.1 Walsh Transform

- discrete Walsh transform :
  - $N = 2^n$
  - kernel $g(x,u) = \frac{1}{N}\prod_{i=0}^{n-1}(-1)^{b_i(x)b_{n-1-i}(u)}$
  - transform

$$W(u) = \frac{1}{N}\sum_{x=0}^{N-1} f(x)\prod_{i=0}^{n-1}(-1)^{b_i(x)b_{n-1-i}(u)}$$

where $b_k(z)$: the k-th bit in the binary representation of z

ex) if n=3, z=6 (110)

$\rightarrow b_0(z) = 0,\ b_1(z) = 1,\ b_2(z) = 1$

values of the 1-D Walsh transformation kernel (excluding constant $1/N$) for N=8

$N = 8$

u \ x	0	1	2	3	4	5	6	7
0	+	+	+	+	+	+	+	+
1	+	+	+	+	−	−	−	−
2	+	+	−	−	+	+	−	−
3	+	+	−	−	−	−	+	+
4	+	−	+	−	+	−	+	−
5	+	−	+	−	−	+	−	+
6	+	−	−	+	+	−	−	+
7	+	−	−	+	−	+	+	−

- array formed by Walsh transformation kernel
  : symmetric matrix having orthogonal rows and columns

  $\rightarrow$ inverse kernel identical to the forward kernel except for $1/N$

$$h(x,u) = \prod_{i=0}^{n-1}(-1)^{b_i(x)b_{n-1-i}(u)}$$

- inverse Walsh transform

$$f(x) = \sum_{u=}^{N-1} W(u)\prod_{i=}^{n-1}(-1)^{b_i\ x\ b_{n-1-i}\ u}$$

- 2-D Walsh transform :

  - forward kernel ;

$$) = \frac{}{N} \prod_{i=0}^{n-1} -)^{[b_i(x)b_{n-1-i}(u)+b_i(x)b_{n-1-i}(v)]}$$

  - inverse kernel ;

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

  - kernel : separable and symmetric ;

$$v) = \quad _1 \quad) \quad _1 \quad v)$$

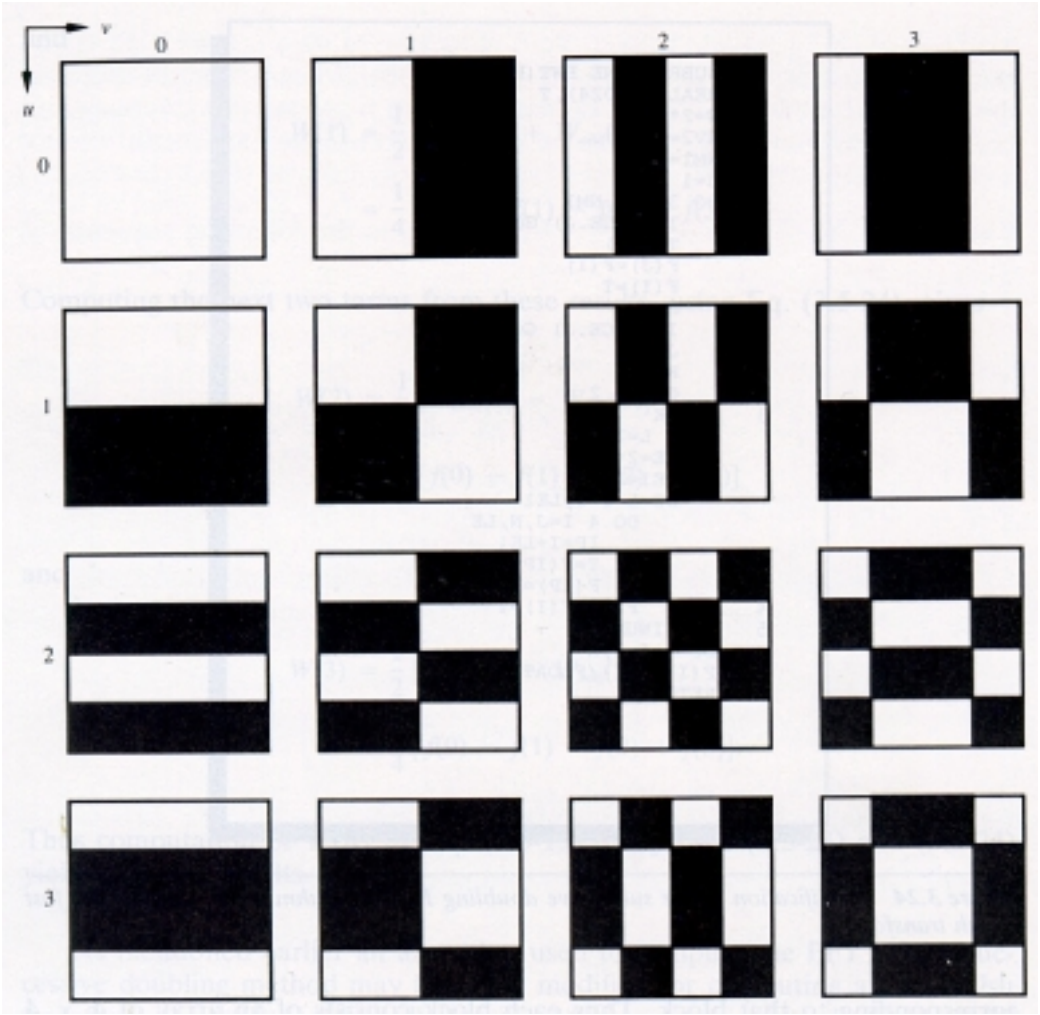
$$= \quad _1 \quad) \quad _1 \quad v)$$

$$= [\frac{1}{\sqrt{N}} \prod_{i=}^{n-1} (-1)^{b_i \; x \; b_{-1-i} \; u}][\frac{1}{\sqrt{N}} \prod_{i=}^{n-1} (-1)^{b_i \; y \; b_{-1-i} \; v}]$$

- fast Walsh transform (FWT) :

  - nearby identical in form to the successive doubling algorithm for FFT
  - only difference : all exponential terms W_n in FFT
    $\rightarrow$ set to 1 in FWT

- Walsh transform : real (smaller computational storage)



the basis functions (kernels) as a function of u and v for computing WT when N=4 (excluding $1/N$).

- computation of WT using the basis function :

  - W(0,0) : obtained by multiplying the image array point-by-point with the 4×4 basis block corresponding to u=v=0, adding the results, and dividing by 4
  - W(1,1) : with the block corresponding to u=1, and v=1

    M
  - W(3,3) : with the block corresponding to u=3, and v=3

- WHT(Walsh-Hadamard T)는 DCT 에 비해 에너지 집중화면은 떨어지지만 적고 하드웨어 구성이 간단하므로 많이 사용된다.)

3.5.2 Hadamard forward Transform

- 1-D Hadamard forward kernel :

$$) = \frac{}{N} -)^{\sum_{i=0}^{n-1} \; i \;\;) \; i \; u)}$$

where summation : performed in module 2 arithmetic.

$b_k(t)$: k-th bit in binary representation of z

- HT ;

$$H(u) = \frac{1}{N} \sum_{x=0}^{N-1} f(x)(-1)^{\sum_{i=}^{-1} b_i(x)b_i(u)}$$

where $N = 2^n$

- inverse kernel ;

- $$u) = -)^{\sum_{i=0}^{n-1} \; i \;\;) \; i \; u)}$$

- inverse HT ;

$$f(x) = \sum_{u=0}^{-1} H(u)(-1)^{\sum_{i=}^{-1} \; i(\;) \; i(u)}$$

- 2-D HT kernel :
  - forward kernel

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

  - reverse kernel : identical to forward kernel

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

- 2-D HT pair ;

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

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

- properties of HT kernel :

  - separable and symmetric

$$g(x,y,u,v)=g_1(x,u)g_1(y,v)=h_1(x,u)h_1(y,v)$$

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

- the matrix produced by 1-D Hadamard kernel for N=8 (except for $1/N$)

Table 3.4

u \ x	0	1	2	3	4	5	6	7
0	+	+	+	+	+	+	+	+
1	+	−	+	−	+	−	+	−
2	+	+	−	−	+	+	−	−
3	+	−	−	+	+	−	−	+
4	+	+	+	+	−	−	−	−
5	+	−	+	−	−	+	−	+
6	+	+	−	−	−	−	+	+
7	+	−	−	+	−	+	+	−

  - the entries is same as for the WT but the order of the rows and columns is different

- Walsh-Hadamard transform :

  - used to denote either WT or HT
  - WT, HT : intermixed in image processing

- Choice of WT or HT :

  - FWT ;
    - ✓ Straightforward modification of FFT ($W_N \to 1$)
    - ✓ Reordering FWT→FHT
  - ✓ Simple recursive relationship for generating the transformation matrices

  - Hadamard matrix of the lowest order (N=2) ;

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

  - The recursive relationship ;

$$\mathbf{H}_{2N} = \begin{bmatrix} \mathbf{H}_N & \mathbf{H}_N \\ \mathbf{H}_N & -\mathbf{H}_N \end{bmatrix}$$

    where $\mathbf{H}_N$: matrix of order $N = 2^n$
  - Transformation matrix ;

$$\mathbf{A} = \frac{1}{\sqrt{N}} \mathbf{H}_N \quad \text{in the N×N case}$$

  - Inverse Hadamard matrix ;
    : identical to forward Hadamard matrix

- Sequency :

  - no. of sign changes along a column of H matrix
  - in Table 3.4
    : 0, 7, 3, 4, 1, 6, 2, 5
  - reordering the H kernel so that the sequency increase as a function of increasing u
    : analogous to the FT

$$g(x,u) = \frac{1}{N}(-1)^{\sum_{i=0}^{n-1} b_i(x) P_i(u)}$$

where $P_0(u) = b_{n-1}(u)$

$P_1(u) = b_{n-1}(u) + b_{n-2}(u)$

M

$P_{n-1}(u) = b_1(u) + b_0(u)$

table 3.5

u \ x	0	1	2	3	4	5	6	7
0	+	+	+	+	+	+	+	+
1	+	+	+	+	−	−	−	−
2	+	+	−	−	−	−	+	+
3	+	+	−	−	+	+	−	−
4	+	−	−	+	+	−	−	+
5	+	−	−	+	−	+	+	−
6	+	−	+	−	−	+	−	+
7	+	−	+	−	+	−	+	−

: column, row : in order of increasing sequency

- Inverse, ordered H kernel :

$$h(x,u) = (-1)^{\sum_{i=0}^{n-1} b_i(x)P_i(u)}$$

- Ordered 1-D HT pair :

  - $$H(u) = \frac{1}{N}\sum_{x=0}^{N-1} f(x)(-1)^{\sum_{i=0}^{n-1} b_i(x)P_i(u)}$$

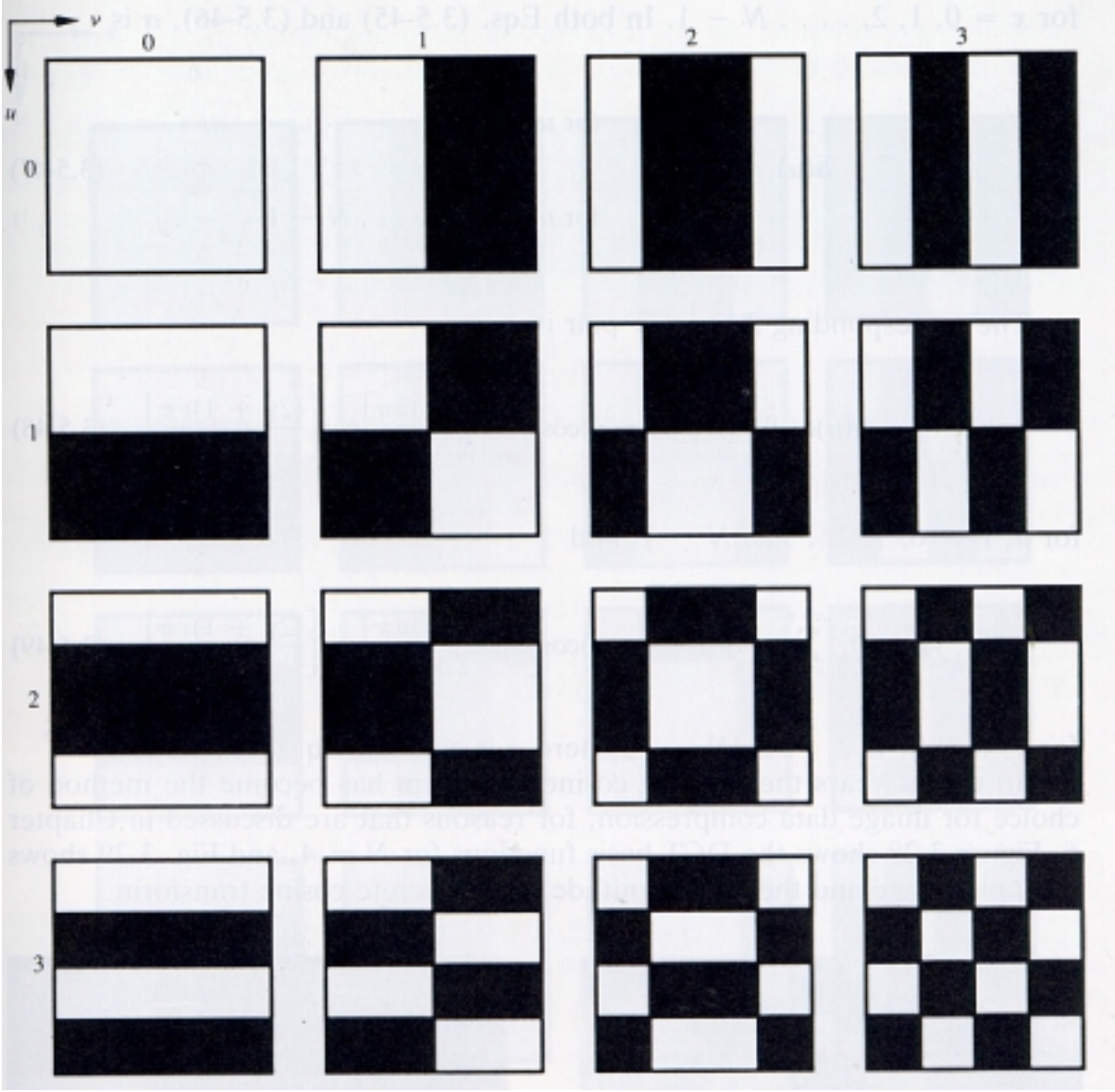
    $$f(x) = \sum_{x=0}^{N-1} H(u)(-1)^{\sum_{i=0}^{n-1} b_i(x)P_i(u)}$$

  - 2-D kernel ;
    : separable and symmetric

- Ordered 2-D HT pair :

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

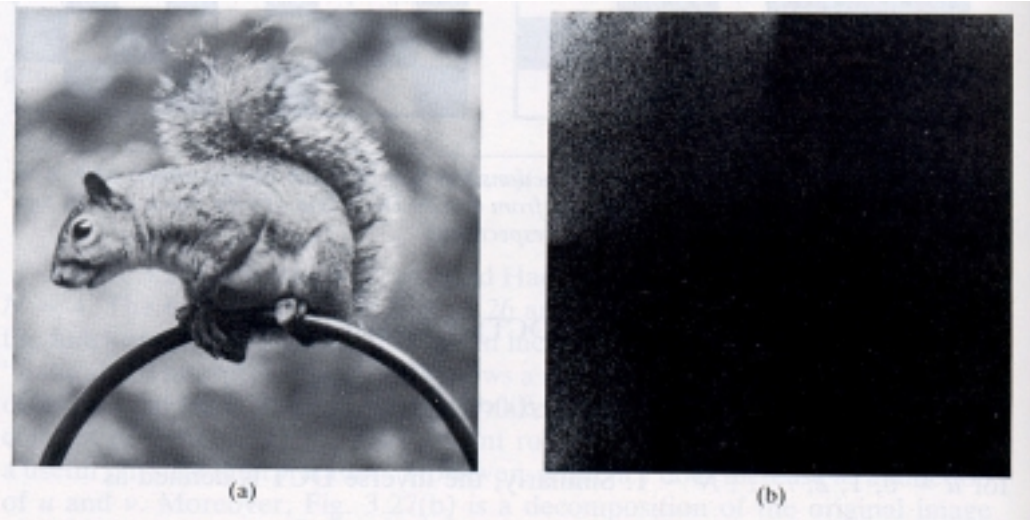
$$f(x,y)=\frac{1}{N}\sum_{u=0}^{N-1}\sum_{v=0}^{N-1}H(u,v)(-1)^{\sum_{i=0}^{n-1}[b_i(x)P_i(u)+b_i(y)P_i(v)]}$$



: ordered H basis functions (kernels) for N=4

- ordered increasing sequency

HTed image (log. Magnitude)



(b) : decomposition of original image function, based on basis functions that are simply +1 or –1 instead of the more complex sine and cosine functions used in FT