《特別寄稿》

화상처리의 직교변환 기술

李光宰,* 姜承先,* 朴周用,**梁根鎬,***李門浩*
(*전북대학교 정보통신공학과, **서남대학 전자공학과, ***전북대학병원 의공과)

■ 차 례 ■

1. 서 론

DFT(Discrete Fourier Transform)를 시작으로 직교변환 기술은 디지탈 신호처리 분야에서 대단히 중요한 수법이 되고 있다. 직교변환의 구체적인 응용을 위해서는 고속연산법의 존재가 중요하다. DFT의 고속연산법인 FFT(Fast Fourier Transform) 알고리즘의 제안 이후 여러가지 고속연산법이 활발히 연구되고 있다.

Fourier 변환 이후의 대표적인 직교변환으로서는 Walsh-Hadamard 변환, Haar 변환, Slant 변환, 이산적 코사인 변환 등이 알려져 있다. 이들의 기본적 성질은 종래의 고속연산법 및 응용에 관한 많은 문헌에서 다루어져 알기 쉽게 설명되어 있다. 따라서 본 고에서는 최근의 연구 동향을 중심으로 기술한다.

우선 2장에서 직교변환의 고속연산법에 관한 최근의 연구를 개설한다. 다음에 3장에서는 Discrete Linear Base 변환, Alpha 변환 등의 새로운 직교변환에 대해서 설명하며, 변환기술에 관한 최근의 화제를 비교적 간단히 기술한다. 마지막으로 4장에서는 2차원 직교변환의 구체적 계산순서와 계산시간과의 관계에 대하여 기술한다.

2. 직교변환의 고속연산법에 관한 최근 연구

직교변환의 구체적 응용은 고속연산법의 존재가 중요하다. 고속연산법으로는 예를 들면 변환행렬을 소행렬(sparse matrix)의 곱으로 분해, 연산횟수(특히 승산횟수)를 직접계산에 비하여 삭감하는 것에 중점을 두고 있다. 그러나, 화상처리에서의 응용을 고려한 경우에는, 취급하는 데이타량이 방대하므로 단순히 연산횟수뿐만 아니라 필요한 메모리 용량이나 데이타 전송량, 데이타의 병렬변환 등의 여러가지 전후처리 및 그 외의 사항에 대해서도 충분히 고려할 필요가 있다.

본 고에서는 고속연산법에 관한 최근의 연구로서 Cooley-Tukey계의 FFT 알고리즘과는 다른 DFT의 새로운 고속연산법, 임의장 DFT의 고속연산법, Walsh Hadamard 변환(WHT)에 대해

행렬표에 의한 단일취급 이산적 코사인 변환의 고속연산법에 대하여 설명한다.

2.1. DFT의 고속연산법에 관한 최근의 연구

2.1.1. 개 설

이산적 Fourier 변환(DFT)

$$X(k)=\sum_{n=0}^{N-1} x(n)\ W^{nk} \quad \text{(forward)} \qquad (2.1a)$$

$$x(n)=\frac{1}{N}\sum_{k=0}^{N-1} X(K)\ W^{-nk} \quad \text{(inverse)} \qquad (2.1b)$$

단, $W=\exp(-j\ 2\pi/N)$

에 관해서는 Cooley-Tukey에 의한 고속 Fourier 변환 알고리즘의 개발 이후 다양한 요구에 따라 이의 고속 계산법이 연구되고 있다.

고속연산법에 대한 최근 연구 방향의 하나는 DFT를 콘볼루션 계산으로 변환하여 이 콘볼루션을 고속 계산하는 수법이 있다. 또한, Rader[8]는 데이터장 N이 소수일 경우, 데이터의 병렬변환 조작에 의해, DFT가 주기적 콘볼루션으로 치환되는 것을 보였고 Winograd[9]는 주기적 콘볼루션에 필요한 최소 승산횟수를 보였다. 한편, Agarwal과 Cooley[10]는 다항식 수법과 Chinese Remainder Theorem을 이용하여 비교적 데이터장 N이 짧은 경우에 대하여, 이의 최소 승산횟수를 달성하는 새로운 고속 콘볼루션 알고리즘을 제시하였다.

Winograd[11]는 비교적 짧은 데이터장에 대해 이의 새로운 콘볼루션 알고리즘을 DFT에 결부시키는 시도를, 말하자면 Winograd Fourier Transform Algorithm(WFTA)[12][13]을 도출하였다. WFTA에서는 N이 작은(소수 또는 소수의 멱) 경우에 관한 DFT 알고리즘(small N algorithm) 과 함께, 긴 변환에 대하여 small N 알고리즘을 nesting하여 계산하는 방법을 보이고 있다.

한편, Kolba와 Parks[14]는 긴 변환에 대하여 prime factor FFT 알고리즘을 보이고 Winograd의 nested algorithm 보다도 어떤 점에서는 유리함을 밝혔다. 이 수법에 이용되고 있는 short-length DFT 알고리즘은 최소 승산횟수로 콘볼루션을 계산하는 다항식 수법에 의해 직접 유도된다(이에 대하여 small N 알고리즘은 short-lenght DFT 알고리즘에 약간의 수정을 가한 형태로 되어 있다).

이러한 연구 외에 Nusbaumer와 Quandalle[15]은 다항식환상에 정의된 다항식변환 (Polynomial Transform)에 의해, 특히 다차원 DFT를 고속으로 계산하는 방법(1차원 DFT에 대해서도 적용 가능)을 보이고, 많은 경우에 FFT, WFTA 보다도 효율적임을 지적하고 있다. 또한, 긴 DFT를, 짧은 DFT를 조합하는 방법으로 보다 적은 연산횟수로 계산하기 위해 split nesting(가산횟수의 삭감)과 split prime factor (승산횟수의 삭감)의 수법을 보이고 있다.

Burrus[16]은 1차원 배열을 2차원 및 다차원 배열에 매핑(mapping) 하는 방법이 FFT 알고리즘이나 앞서의 고속 콘볼루션 수법의 기초가 됨을 지적하고 독특하며 보다 주기적으로 매핑하기 위한 일반적 조건을 언급하고 있다.

여기서는 DFT와 콘볼루션의 관계, 최소 승산횟수의 콘볼루션을 이용한 DFT, 데이터장 N이 더욱 긴 경우의 변환 수법에 대하여 설명한다.

2.1.2. 콘볼루션에 의한 DFT의 계산

DFT의 정의식(2.1a)을 다음과 같이 표시하자.

$$\left.\begin{aligned} X(0)&=\sum_{n=0}^{N-1} x(n) \\ X(k)&=x(0)+\bar{X}(k)\quad k=1,\cdots,N-1 \end{aligned}\right\} \qquad (2.2)$$

$$\bar{X}(k)=\sum_{n=0}^{N-1} X(n)\ W^{nk} \qquad (2.3)$$

N이 소수인 경우를 고려하자. 우선, 다음 식에 의해 데이타의 병렬변환을 행한다.

$$n=a^m \bmod N \qquad \begin{matrix} n=1,\ 2,\cdots,N-1 \\ m=0,\ 1,\cdots N-2 \end{matrix} \qquad (2.4)$$

단, $a^k \neq 1$, $0<k<N-1$

$a^{N-1}=1$

(결국 a는 1의 원시 (N-1)승근)

식(2, 4)를 이용하여 식(2, 3)을 다시 쓰고, 편의상 m의 부호를 반전하는 방법에 의해 식(2, 3)은 다음 식(2, 5)에 보여진 모양의 주기적 콘볼루션이 된다.

$$\overline{X}(a^l)=\sum_{m=0}^{N-2} x(a^{-m})\ W^{a(l\,m)}\quad l=0,\ 1,\cdots,\ N-2 \tag{2.5}$$

단, a의 지수는 mod(N-1)로 계산된다.

또한, Winograd[11], McClellan과 Rader[17]는 N이 소수 p의 멱(단 $p \neq 2$)인 경우도, DFT가 콘볼루션으로 변환됨을 보였다.

2.1.3. 최소 승산횟수에서의 콘볼루션 계산

N이 작은 경우에 주기적 콘볼루션을 최소 승산횟수로 계산하는 알고리즘은 다항식 수법을 이용하여 설명된다. 긴 N의 2개의 계열 $h_0, h_1, \cdots h_{N-1}$과 $x_0, x_1, \cdots x_{N-1}$의 주기적 콘볼루션을 구하려면, 다음 다항식에서 N개의 계수를 구하면 된다.

$$Y(z)=H(z)\cdot X(z)\ \mathrm{mod}(z^N-1) \tag{2.6}$$

단, $H(z)=\sum_{k=0}^{N-1} h_k z^k$,

$X(z)=\sum_{k=0}^{N-1} x_k z^k$,

다항식 z^N-1이 정(正)계수의 K개의 기약(旣約) 다항식 $Q_i(z)$ (cyclotomic polynomials)의 곱으로 나타낼 수 있으면 다음과 같이 된다.

$$z^N-1=\prod_{i=1}^{k} Q_i(z) \tag{2.7}$$

단, K:1과 N을 포함한 N의 약수의 수

Winograd[9]에 의하면, 식(2.6)의 주기적 콘볼루션에 필요한 최소 승산횟수는 2N-K이다.

식(2.6)의 콘볼루션은 다항식에 적용된 Chinese Remainder Theorem을 이용하여 K개의 짧은 주기적 콘볼루션의 조합으로 변환되며, 계산량의 삭감이 이루어진다.

$$Y(z)=\{\sum_{i=1}^{k} Y_i(z)\ S_i(z)\}\ \mathrm{mod}(z^N-1) \tag{2.8}$$

$$Y_i(z)=H_i(z)\cdot X_i(z)\ \mathrm{mod}\ (Q_i(z))\quad i=1,\cdots,\ K \tag{2.9}$$

$$X_i(z)=X(z)\ \mathrm{mod}\ (Q_i(z))\qquad i=1,\cdots,\ K$$

$$H_i(z)=H(z)\ \mathrm{mod}\ (Q_i(z))\qquad i=1,\cdots,\ K \tag{2.10}$$

다항식 $S_i(z)$ $i=1,\cdots,$ K는 Kronecker delta의 역할을 하며, Euclid 알고리즘을 이용하여 구해진다.

$$S_i(z)=1\ \mathrm{mod}\ Q_i(z)\quad i=1,\cdots,\ K$$
$$S_i(z)=0\ \mathrm{mod}\ Q_j(z)\quad 단,\ j\neq i \tag{2.11}$$

2.1.4. Short length DFT 알고리즘 (최소 승산횟수 콘볼루션을 이용한 DFT의 고속연산법)

Winograd[11]는 z^N-1가 큰 기약수를 갖는 경우, 최소 승산횟수로 주기적 콘볼루션을 계산하는 기존의 알고리즘에서는 다수의 가산이 필요하게 됨을 지적하고 있다. 이 때문에, 주기적 콘볼루션을 거쳐 DFT를 계산하는 방법의 이점을 활용하기 위해서는 N이 작아야 한다. 현재, N으로서는 2, 3, 4, 5, 7, 8, 9, 16이 이용되고 있다(short length DFT 알고리즘).

N=3의 경우에 대해서도, 2.1.2 및 2.1.3에서 기술한 수법을 이용하여 유도된 DFT의 고속 알고리즘을 보면 다음과 같은 형태가 된다.

[short length DFT : N=3]

$a_1=x(1)+x(2)$ $m_1=-1/2\ a_1$ $X(0)=a_3$

$a_2=x(1)-x(2)$ $m_2=-j\frac{\sqrt{3}}{2}a_2$ $X(1)=C_1+m_2$

$a_3=x(0)+a_1$ $C_1=x(0)+m_1$ $X(2)=C_1-m_2$

연산횟수 : 승산 1회, Bit shift 1회, 가산 6회

Winograd의 nested algorithm(2.1.5에 기술)에서는 $W^0(\equiv 1)$에 의한 승산도 고려되기 때문에, W^0에 따른 승산횟수를 줄이는 것이 필요하며 위의 알고리즘에 수정을 가한 다음의 알고리즘이 이용된다.

[short length DFT(small-N algorithm) : N=3]

$$a_1=x(1)+x(2) \quad m_1=(-\frac{1}{2}-1) \quad a_1=-\frac{2}{3}a_1$$

$$X(0)=m_3$$

$$a_2=x(1)-x(2) \quad m_2=-j\frac{\sqrt{3}}{2}a_2 \quad X(1)=C_1+m_2$$

$$a_3=x(0)+a_1 \quad m_3=W^0\cdot a_3=1\cdot a_3 \quad X(2)=C_1-m_2$$

$$C_1=m_3+m_1$$

연산횟수 : 승산 2회, W^0에 의한 승산 1회, 가산 6회

행렬을 이용하면 다음과 같이 된다.

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & -\frac{2}{3} & 0 \\ 0 & 0 & -j\frac{\sqrt{3}}{2} \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 1 & -1 \end{bmatrix} \begin{bmatrix} x(0) \\ x(1) \\ x(2) \end{bmatrix} \quad (2.12)$$

일반적으로 small-N 알고리즘은 다음 식으로 표현된다.

$$X=D_Nx=S_NC_NT_Nx \quad (2.13)$$

단, D_N : W의 멱을 요소로 하는 DFT의 변환행렬

T_N, S_N : 0, ±1만으로 되는 incidence 행렬, 각각 입력, 출력으로의 가산처리를 나타낸다.

C_N : 승산계수(실수 또는 순허수)를 포함한 대각행렬.

2.1.5. 데이타장 N이 큰 경우의 변환 수법

데이타장 N이 큰 경우에 대해서는 short length DFT 알고리즘을 조합하여 계산하는 방법이 제안되고 있다. 여기서는 Winograd[11]에 의한 nested algorithm과 Kolba, Parks[14]에 의한 prime factor algorithm에 대하여 설명한다.

[1] Winograd's nested algorithm[11-13]

N이 서로 소인 L개의 인수의 적(積)

$$N=N_1\times N_2\times\cdots\times N_L \quad (2.14)$$

으로 분해되고, 각 인수 N_l가 "small-N" 인수에 대응하게 되어, 식(2.13), (2.14)에 대하여 WFTA 의 기본식 (2.15)가 얻어진다.

$$X'=(SN_LCN_LTN_L^*\cdots^*SN_1CN_1TN_1)x'$$
$$=(SN_L^*\cdots^*SN_1)(CN_L^*\cdots^*CN_1)(TN_L^*\cdots^*TN_1)x \quad (2.15)$$

단, *는Kronecker적, x', X'는 각각 데이타의 병렬변환 R_I, R_F를 거친 입출력 벡터이다. 또한, R_I, R_F는 가역적인 매핑 조작은 되지 않는다.

계산의 편의상(가산횟수의 삭감), 입출력 벡터를 L차원 행렬로 바꿔 쓴 z_N, Z_N에 대한, 다음의 nested form을 표시했다.

$$Z_N=SN_L(SN_{L-1}\cdots(SN_{L2}(SN_1C\circ TN_1$$
$$\cdot(TN_2\cdots(TN_{L-1}(TN_Lz_N)^t)^t\cdots)^t)^{t^{-1}})^{t^{-1}}\cdots)^{t^{-1}} \quad (2.16)$$

t, t^{-1}은 각각 TN_l, SN_l에 관계되는 conjugate transpose를 나타낸다. C는 행렬이지만, 대각행렬은 아니다. ∘는 요소만의 적을 나타낸다.

Winograd's nested algorithm에서 전체적인 승산횟수 M_N는 인수 N_l에 대한 small-N 알고리즘의 승산횟수를 MN_l로 하여,

$$MN=MN_L\cdot MN_{N-1}\cdots MN_1 \quad (2.17)$$

로 된다. 가산횟수에 대해서는 N이 식 (2.14)의 형으로 분해될 때, 대응하는 small-N 알고리즘의 적용 순서에 의존한다. 가산횟수를 줄이기 위해, small-N의 상대적 순위로서는 우선 9, 다음으로 5, 7, 16의 순으로 되어 나머지 2, 3, 4, 8의 순서는 임의로 된다.

역시, nested algorithm의 특징은 모든 승산을 입력, 출력부에서의 가산처리 안쪽에 nesting하고 있다는데 있다.

[2] prime factor FFT algorithm[14]

길 $N=N_1\times N_2\times\cdots\times N_L$($N_i$은 서로소)의 1차원 DFT를 Chinese Remainder Mapping을 이용하여 L차원 DFT로 바꾸고, 각 인덱스(index)에 대한 DFT(short length DFT이다)를 고속 콘볼루션을 거쳐 계산하는 방법이다. 이 경우에는 nested 알고리즘과는 달리 승산횟수는 인덱스에 대한 DFT로 분산된다. 또한, short length DFT 알고리즘으로는 2.1.3에서 기술한 최소 승산횟수

표 2.1. Short Length DFT에 필요한 연산횟수[10].

N	Prime Factor FFT			Nested Algorithm		Adds
	Multiplies	Shifts	Adds	Multiplies	W^0 multipies	
2	0	0	2	0	2	2
3	1	1	6	2	1	6
4	0	0	8	0	4	8
5	4	2	17	5	1	17
7	8	0	36	8	1	36
8	2	0	26	2	6	26
9	8	2	49	10	2	49

표 2.2. DFT 알고리즘의 비교[10](각종 데이타장 N에 대한 연산횟수)

N	Nested Algorithm			Prime Factor FFT		Radix 2 FFT	
	Factors	Multiplies	Adds	Multiplies	Adds	Multipies	Adds
30	5·3·2	68	384	68	384		
32	2^5					102	830
126	9·7·2	424	3,312	512	2,920		
128	2^7					774	5,662
252	9·7·4	848	7,128	1,024	6,344		
256	2^8					1,926	13,726
315	9·5·7	1,292	11,286	1,784	8,812		
360	9·5·8	1,152	9,492	1,396	8,708		
504	9·7·8	1,704	15,516	2,300	13,948		
512	2^9					4,614	32,286
840	5·7·8·3	2,592	24,804	4,244	23,177		
1,024	2^{10}					10,758	74,270
1,260	9·5·7·4	5,168	50,184	7,136	40,288		
2,048	2^{11}					24,283	167,966
2,520	9·5·7·8	10,344	106,667	15,532	86,876		

콘볼루션에서 직접 유도된 수법을 이용한다.

Prime factor FFT 알고리즘은 mixed radix type이며, 문헌 14)에서는 IBM 370 FORTRAN을 이용하여 Singleton[22]의 mixed radix FFT와의 계산시간을 비교하여, 약 50%로 됨을 밝히고 있다.

또한, Kolba, Parks[14]는 prime factor FFT가 다음과 같은 점에서 매력적임을 지적하고 있다. 즉,

① 모든 승·가산수는 nested 알고리즘과 같은 정도.

② nested 알고리즘에 비해 메모리 용량과 데이타의 축적이 적어진다.

③ 프로그래밍이 용이하다.

[3] DFT 알고리즘의 연산횟수 비교

앞서 기술한 nested 알고리즘과 prime factor FFT 알고리즘에 적용되는 short length DFT에 필요한 연산횟수는 표 2.1과 같다. 또한, N이 큰 경우에 대하여 nested algorithm, prime factor FFT 알고리즘 및 raidx 2 FFT 알고리즘 각각에 대하여 연산횟수를 나타내면 표 2.2와 같다.

2.1.6. WFTA에 관한 기타 화제

WFTA는 복소수 데이타의 DFT에 관한 고속연산법이며, N점 데이타의 변환에 대하여 기수 2의 FFT에서는 $0(N \log_2 N)$의 승산이 필요한 것에 대해 $0(N)$의 승산횟수로 된다는 특징이 있다. 그러나, 필요한 메모리 용량이, FFT의 2.25N에 대하여 최대 7N으로 증가한다는 문제가 있다. 이 WFTA에 대해서는 앞서 기술한 연구 외에도 다양한 검토가 이루어지고 있다.

WFTA 프로그램 작성에는 FORTRAN 서브루틴과 small-N 알고리즘의 적용순서등을 나타낸 표 등, 구체적 수법이 실려있는 Zohar[18]의 문헌을 참고하기 바란다. 또한 Parsons[19]는 실수 데이타의 WFTA에 대하여 검토하고 있다.

Patterson과 McClellan[20]은 fixed-point 연산의 경우에 대하여 오차해석을 행하고 WFTA는 FFT보다도 오차가 많고 일반적으로 FFT와 같은 정도의 오차로 하려면 데이타 표현을 위한 1비트 내지 2비트가 여분으로 필요한 점을 지적하고 있다. 한편, WFTA에 관해서 small-N 알고리즘의 적용순서가 연산횟수, 중간결과를 축적하고 있기 위한 메모리용량, 계산오차에 대한 영향을 언급하고, 이들을 동시에 최소화할 수 없음을 지적하고 있다.

Morris[21]는 범용계산기를 이용하여 부동소수점 연산으로 프로그램을 작성한 경우, WFTA에서는 FFT에 비해 승산횟수는 감소하지만, 메모리 레지스터간, 레지스터·레지스터간 데이타 전송의 증대, 데이타의 병렬 변환 등이 전체적인 계산시간에 영향을 주어 연산횟수만에 의해 비교가 위험한 것임을 지적하고 있다.

2.2. 임의장 DFT의 고속연산법

일반적으로 DFT를 고속으로 계산하기 위해서는 데이타장 N이 highly composite(多數 因數의 積)으로 되어야 한다. 임의의 N에 대해서는 다음에 기술한 것처럼 N'점($N' \geq 2N-1$)계열의 콘볼루션, 또는 상관을 거쳐 고속으로 계산하게 된다.

N점 데이타 x(n)의 이산적 Fourier 변환 X(k)는 (2.1a)와 (2.1b)와 같다. 순변환의 경우에는 chrp-Z 변환 계산으로 되어, 다음과 같이 행해진다[23].

$$nk = \frac{\{k^2 + n^2 - (k+n)^2\}}{2} \tag{2.18}$$

라 하면 식(2.1a)는

$$X(k) = W^{n^2/2} \left\{ \sum_{n=0}^{N-1} \left(x(n)\, W^{n^2/2} \right) W^{\frac{-(k-n)^2}{2}} \right\} \tag{2.19}$$

로 되어, $\hat{x}(n) = x(n)\, W^{n^2/2}$과 $h(n) = W^{n^2/2}$과의 콘볼루션 계산으로 된다. 이 콘볼루션은 $\hat{x}(n)$에는 0을 추가하고, $h(n)$에는 $h(n) = h(N'-n+$

1), n=N'-N+1,…,N'-1로 구성되는 데이타를 부가하여 N'점(N'≥2N-1, 2의 멱으로 된다) 계열로 하여, 종래의 기수 2의 FFT를 거쳐 고속으로 계산된다. 역변환의 경우에는 순변환용의 프로그램을 이용하여 계산하게 되지만[2], 여기서는 상관을 이용하여 계산하는 수법에 대하여 소개하고 있다.

$$-nk=\{k^2+n^2-(k+n)^2\}/2 \qquad (2.20)$$

로 하면, 식(2.1b)는

$$x(n)=\frac{1}{N}W^{n^2/2}\left\{\sum_{k=0}^{N-1}\left(X(k)W^{k^2/2}\right)W^{\frac{-(k+n)^2}{2}}\right\} \qquad (2.21)$$

로 된다. $\hat{X}(k)=X(k)W^{k^2/2}$과 $H(k)=W^{k^2/2}$와의 상관계산으로 된다. 이 상관은 $\hat{X}(k)$에는 0을 추가하고, H(k)에는 $H(k)=W^{k^2/2}$, k=N,…, 2N-2 및 H(k)=0, k=2N-1,…,N'-1을 추가하여 N'점(N'≥2N-1, 2이 멱이 된다)계열로서, 기수 2 FFT를 거쳐 고속으로 계산된다.

임의 사이즈의 2차원 DFT에 대해서도 마찬가지의 알고리즘을 이용하게 된다. N×N점 DFT를 계산하는 데는 N'×N'점 DFT(또는 2N회의 N'점 DFT)를 3회 실행하기 때문에 계산시간이나 메모리 용량은 증대하지만 임의 크기의 화상에 적용된다는 이점이 있다.

2.3. WHT 고속연산법의 행렬표현에 의한 단일 취급

Walsh-Hadamard 변환(WHT)에 관해서는 여러가지로 정의되고 또한 많은 고속연산법이 발표되어 있다[1-25]. 이에 대하여 Fino와 Algazi[24]는 행렬표현을 이용하여 WHT의 단일 취급법을 제시했다.

우선, order 2^n의 Handamard ordered Walsh-Handamard (WH) 행렬 $[H_h(n)]$에 대해서 고려하고 있다. $[H_h(n)]$은 order 2^n의 WH행렬 $[H_2]$를 이용하여 다음과 같이 표현된다.

$$[H_h(n)]=[H_2]*[H_h(n-1)]=[H_2]^{*n} \qquad (2.22)$$

단, $[H_2]=\frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}$

*n은 n회 연속하여 Kronecker 적을 취함을 나타낸다.

Order 2^n의 Paley 및 Walsh order의 WH 행렬을 각각 $[H_p(n)]$, $[H_w(n)]$이라 하면 이들은 다음과 같은 $[H_h(n)]$으로부터 유도된다.

$$[H_p(n)]=[H_h(n)][BR]=[BR][H_h(n)] \qquad (2.23)$$

$$[H_w(n)]=[BI][BR][H_h(n)]$$
$$=[BI][H_h(n)][BR] \qquad (2.24)$$

단, [BR], [BI]는 각각 bit 역순 병렬변환, bit inversion 병렬변환을 행하는 행렬이다.

따라서, $[H_h(n)]$에 의한 변환의 전후에서 적당한 데이타의 병렬변환을 실시하는 것으로 다른 ordering에 대응한 WHT 결과를 얻게 된다.

화상처리에의 응용은 $[H_w(n)]$을 이용한 Walsh-ordered WHT(WWHT)가 유용하며, sequency순의 변환결과가 얻어진다. WWHT의 고속연산법으로서는, (a) 식(2.24)에 근거하여 고속 Hadamard-ordered WHT를 거쳐 계산하는 방법과 (b) 입력 데이타의 bit reversed 병렬변환을 거친 후 $[H_h(n)]$에 수정을 가한 행렬에 의해 고속연산을 행하는 Manz의 알고리즘[25]이 알려져 있다. (b)에 대해서도 행렬표현으로 부의 도출이 가능하며 식(2.24)에 비해 bit inversion 병렬변환이 불가능한 부분만 계산시간의 점에서 유리하다. 그러나 (a), (b) 어느 것을 이용하여도 큰 차이는 없다.

2.4. DCT의 고속연산법 : DCT 프로세서를 위한 알고리즘[47-56]

DCT는 그림 2.1과 같은 변환부호화(Trans-

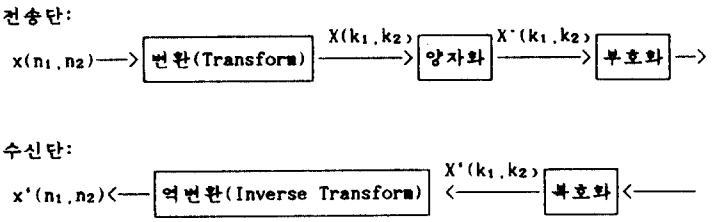


그림 2.1. 변환 부호화

form coding) 중에 하나이다. 변환 부호화는 파형부호화(waveform coding) 보다 계산은 많지만 적은 비트(low bit rate)로 전송하는 분야에서 수행 능력이 더 뛰어나다. 이 변환은 영상화소간의 중복도를 줄여 많은 양의 에너지를 갖는 변환 계수가 나타난다.

이러한 특성을 에너지집중이라 하며, 낮은 에너지를 갖는 계수에 대해서는 적은 비트를 할당하므로써 낮은 전송 비트로 영상의 부호화를 가능하게 하지만 많은 계산량으로 인해 하드웨어 구현이 어렵다.

여러가지 변환 부호화가 에너지 집중도와 계산 요구량을 고려하여 제시되었다. 그중 KLT(Karhunen Loeve Transform)가 에너지 집중면에서 최적의 변환으로 알려져 있다. 하지만 KLT는 데이타를 불규칙적으로 처리해야 된다는 것과 그에 대한 공분산(covariance)을 알고 있다는 전제하에 변환이 이루어졌으나 실제 영상에 있어서 불규칙 처리라는 것은 부정확하며 공분산도 알 수가 없고 추측될 수 밖에 없다. 또한 계산에 있어서 효과적인 알고리즘이 없어서 $N_1 \times N_2$에 대하여 $(N_1 \times N_2)^2$ 승산기와 $(N_1 \times N_2)^2$의 가산기가 필요하다. 이러한 이유로 KLT는 영상 부호화에서 좀처럼 사용하지 않는다.

DFT는 FFT라는 효과적인 알고리즘과 양호한 에너지집중으로 많이 사용되었다. DCT는 이러한 DFT의 에너지집중 현상을 더욱 발전시킨 것으로 영상부호화에 가장 널리 쓰인다.

먼저, DFT와 DCT의 에너지집중에 대하여 비교하기 위하여 DFT의 N=4일 때, DFT의 x(n)의 값과 주기 N다마 반복한 x'(n)이 각각 그림 2.2(a), 2.2(b)일 때 그림에서 보는 바와 같이 x(n)의 처음과 끝의 연결부에서 불연속(discontinuity)성이 나타나는데 이는 높은 주파수를 형성하여 에너지 집중을 감소시킨다. 불연속점을 없애기 위해 N개를 대칭(Symmetric)적인 형태로 하여 2N개의 새로운 수열 y(n)을 만든다. 앞의 수열과 이 수열을 주기 2N 마다 반복하여 만든 y(n)을 나타내면 각각 그림 2.2(c), 2.2(d)와 같다. DCT는 y(n)에 대해 DFT

표 2.3. DCT의 성질

$x(n_1, n_2) \leftrightarrow X_x(k_1, k_2)$: $x(n_1, n_2)$를 DCT 한 것.

$y(n_1, n_2) \leftrightarrow X_y(k_1, k_2)$: $y(n_1, n_2)$를 DCT 한 것.

1. 선 형 성

   $ax(n_1, n_2)+by(n_1, n_2) \leftrightarrow aX_x(k_1, k_2)+bX_y(k_1, k_2)$

2. 분리 가능성

   $x(n_1, n_2)=x(n_1)\times(n_2) \leftrightarrow X_x(k_1, k_2)=X_{x1}(k_1)X_{x2}(k_2)$

3. 대 칭 형

   $x(n_1, n_2)^* \leftrightarrow X_x(k_1, k_2)^*$: *은 복소수

   실수 $x(n_1, n_2) \leftrightarrow$ 실수 $X_1(k_1, k_2)$

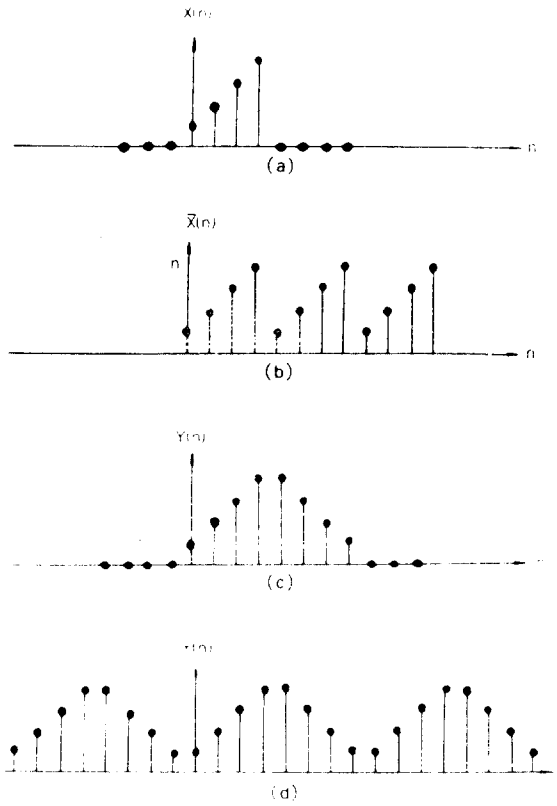


그림 2.2 DFT와 DCT의 에너지집중

를 거친 형태로 불연속적인 점을 갖지 않아서 에너지집중이 개선된다. 변환 부호화에는 이 외에도 Haar, DST(Discrete Sine Transform), 아마다르등이 있으나 압축효율은 DCT에 비하여 좋지 못하다.

DCT에서 두 개의 변수에 상수를 곱한 것은 변환후에 상수를 곱한 것과 같으며 커널로 표현된 코사인은 곱의 형태로 나누어질 수 있어서 1차원의 전치로서 2차원적인 구조를 얻을 수 있으며 앞에서 보듯이 대칭적인 성질을 가졌다. 이러한 여러가지 성질은 변환과 역변환(Inverse Transform)의 연산 알고리즘에서 효과적으로 쓰인다. 표 2.3에 이러한 성질을 정리하여 나타내었다.

2.5. DCT 알고리즘

먼저 간단한 1차원 DCT를 전개하여 일반적인 DCT를 이해하는데 도움을 주기 위해 1차원 DFT와 연관하여 DCT를 유도하고 여러 알고리즘을 간략히 기술한다. 자세한 수식은 부록에 수록했다.

2.5.1. DFT를 이용한 DCT

x(n)을 $0 \le n \le N-1$ 이외는 0을 갖는 N개의 값의 수열이라 할 때 DCT의 여러 형태중 우수대칭(even symmetric)인 구조로 하여 DFT와 연관시켜 보자. N개의 x(n)을 2N개의 y(n)으로 하고, y(n)을 DFT하여 Y(k)을 얻고, Y(k)로부터 C_k를 구한다. 이것을 간단히 나타내면 다음과 같다.

$$\begin{array}{cccccc} N\text{개} & 2N\text{개} & & 2N\text{개} & & N\text{개 DCT} \\ x(n) & \leftrightarrow & y(n) & \xrightarrow{\text{DFT}} & Y(k) & \rightarrow C_k \end{array}$$

첫 단계인 x(n)과 y(n)의 관계를 나타내면

$$y(n)=x(n)+x(2N-1-n) \qquad (2.26)$$

$$= \begin{cases} x(n) & 0 \le n \le N-1 \\ x(2N-1-n) & N \le n \le 2N-1 \end{cases}$$

이다. 예를 들어 N=4일 때 x(n), y(n)은 그림 2.3과 같이 나타낼 수 있으며,

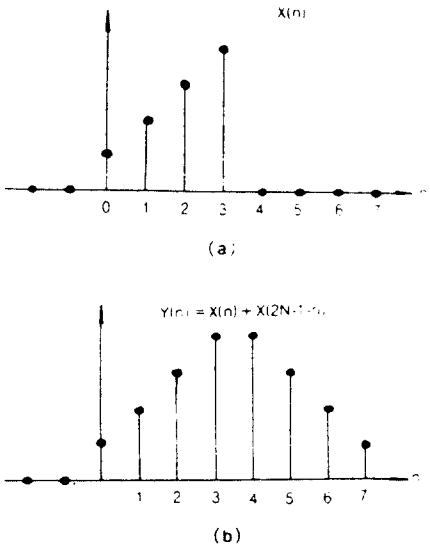
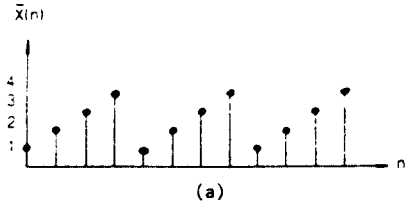


그림 2.3 N=4일 때 x(n)과 y(n)

여기서 수열 y(n)은 $n=N-\frac{1}{2}$ 점에 대하여 대칭적이다. 그리고 x(n)의 주기적인 구조 (그림 2.4a)는 불연속이나, y(n)의 주기적인 구조(그림 2.4b)는 더 이상 불연속이 아니다.



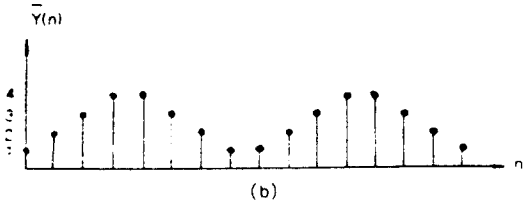


그림 2.4 그림 2.3의 x(n)과 y(n)에 대한 주기적인 구조

y(n)과 Y(k)의 관계는 식(2.26)과 같이 되고, 식(2.26)을 전개하면 식(2.27)과 같이 된다.

$$Y(k)=\underbrace{\sum_{n=0}^{2N-1} y(n)\, W_{2N}^{kn}}_{\text{2N개 DFT}} \qquad 0 \le k \le 2N-1 \tag{2.26}$$

$$W_{2N} = \exp\left(-j\,\frac{2\pi}{2N}\right)$$

$$Y(k)=\sum_{n=0}^{N-1} x(n)\, W_{2N}^{kn} + \sum_{n=N}^{2N-1} x(2N-1-n)\, W_{2N}^{kn}$$

$$0 \le k \le 2N-1 \tag{2.27}$$

그리고, 변수의 변화와 수학적 계산의 결과 식(2.28)는 식(2.29)가 된다.

식(2.28)에 의해 x(n)의 N개, DCT, C_k는 Y(k)로 부터 얻어진다.

$$Y(k) = W_{2N}^{k/2} \sum_{n=0}^{N-1} 2x(n)\ \cos\frac{\pi}{2N} k(2N+1)$$

$$0 \le k \le 2N-1 \tag{2.28}$$

$$C_k = \begin{cases} W_{2N}^{k/2}\ Y(k) & 0 \le k \le 2N-1 \\ 0 & \text{otherwise} \end{cases} \tag{2.29}$$

식 (2.28), (2.29)로 부터

$$C_k = \begin{cases} \sum_{n=0}^{N-1} 2x(n)\ \cos\frac{\pi}{2N} k(2N+1) \\ 0 \end{cases}$$

$$\begin{matrix} 0 \le k \le N-1 \\ \text{otherwise} \end{matrix} \tag{2.30}$$

이며, 식(2.30)이 DCT의 정의식이다. 그리고 DCT인 C_k는 데이타가 실수이면 실수값을, 복소수이면 복소수 값을 갖는다.

2.5.2. DCT의 여러 알고리즘

1974년 K.R. Rao 등에 의해 발표된 DCT는 식(2.31a)로 정의된다.

$$\begin{cases} X(0)=\frac{\sqrt{2}}{N} \sum_{n=0}^{N-1} x(n) \\ X(k)=\frac{2}{N} \sum_{n=0}^{N-1} x(n)\ \cos\frac{\pi}{2N} k(2n+1) \end{cases}$$

$$k=1,\ 2,\ 3,\cdots,N-1 \tag{2.31a}$$

(2.31a)의 정규화 :

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

$$k=1,\ 2,\ 3,\cdots,N-1 \tag{2.31b}$$

DCT는 많은 계산량이 필요하므로 그후 고속 알고리즘들이 DCT계산에 사용되었다. 그러한 알고리즘은 크게 나누어서 간접적인 계산과 직접적인 계산으로 나눌 수 있으며 간접적인 계산은 DFT(or FFT)나 WHT, DHT를 이용한 형태로, DCT 연산을 위한 여분의 연산이 들어 있는 형태이고 직접적인 계산은 Chen, Wang, Lee,

Hou에 의해 이루어졌다. 이러한 고속 알고리즘을 정리하여 표 2.4에 나타내었다.

표 2.4 DCT의 고속 알고리즘

	DFT를 사용한 형태		
	N.Ahmed et al (1974)	M.J. Narasimha et al (1978)	M. Vetterli et al (1984)
알고리즘의 개요	2N점 FFT를 사용	N점 FFT를 사용	순환(Recursive) FFT를 이용
곱셈수	$4N(\log_2 2N+1)$	$\frac{1}{4}$ (log N−N+2)의 복소수 곱셈	$\frac{1}{2}$ (N log N)
장점	의의 : 처음으로 DCT를 DFT와 관련시켜 계산	기존의 FFT를 사용하여 구현이 용이하다.	빠르고 순환 (Recursive)
단점	곱셈수가 많다.	느리다.	복소수 index mapping

	D. Hein et al (1987)	H.S Malvar (1987)	직접 계산 W.H chen et al(1977)
알고리즘의 개요	WHT(Walsh Hadamard Transform)이용	DHT(Discrete Hadamard Transform) 이용	DCT행렬을 sin, cos 행렬과 ±1 행렬로 분해
곱셈수	사용된 WHT 알고리즘에 따라 다르다. N=8일 때 20번	사용하는 FFT와 같다.	$N\log_2 N-\frac{3}{3}$ N+4
장점	쉽게 구현할 수 있고 간단한 mapping	FFT를 사용하여 구현할 수 있다.	상당히 빠르다.
단점	변환 매트릭스 필요 N≥16이면 느리다.	느리다.	복소수 mapping 비순환(nonrecursive)

	직 접 계 산	
	B.G. Lee(1984)	H.S. Hou(1987)
알고리즘의 개요	$\frac{1}{2}$ IDCT를 이용하여 계산	순환적인 형태
곱셈수	$\frac{1}{2}$ (N logN)	$\frac{1}{2}$ (N logN) 다중화→(N−1)
장점	빠르다.	빠르고 순환적이며 간단한 map
단점		다중화 없이는 느리며 쉬프트 필요.

[1] Ahmed et al.의 알고리즘

DCT 수열(2.31b)식을 DFT와 연관시켜 계산하면(부록 1),

$$X(k)=RE\left[\exp\left(-j\frac{k}{2N}\pi\right)\underbrace{\sum_{n=0}^{2N-1}x(n)\,W_{2N}^{-nk}}_{\text{2N점 DFT}}\right] \quad (2.32)$$

단, $W_{2N}=\exp\left(-j\frac{2\pi}{2N}\right)$

Re{·}은 {·}의 실수부를 나타낸다.

위의 식(2.32)에 의하면 N점 DCT는 2N점 DFT(FFT 이용)를 이용하여 계산할 수 있다. 이 계산은 DFT을 이용하는 DCT 연산의 기초가 된다.

[2] Narasimha, Peterson의 알고리즘

데이타 길이 N을 짝수로 하고, 새로운 수열 y(n)을 정의한다.

$$\left.\begin{aligned} y(n)&=x(2n)\\ y(N-1-n)&=x(2n+1)\end{aligned}\right\}\ n=0,\ 1,\cdots,(N/2-1) \quad (2.33)$$

(2.34)식을 (2.32)식에 대입하고 정돈하면

$$X(k)=\sum_{n=0}^{N-1}y(n)\ COS\left[\frac{\pi(4n+1)k}{2N}\right]$$

$$=RE\left[\sum_{n=0}^{N-1}y(n)\exp\left(j\frac{\pi(4n+1)k}{2N}\right)\right]$$

$$=RE\left[\exp\left(j\frac{k\pi}{2N}\right)\underbrace{\sum_{n=0}^{N-1}y(n)\exp\left(\frac{2\pi nk}{N}\right)}_{\text{N점 IDFT}}\right] \quad (2.34)$$

이다. 즉, N점 DCT를 입력의 재배열과 N점 IDFT(Inverse DFT)를 이용하여 계산하였다. (2.34)식의 [·]부분의 계산은 [·]의 k번째의 값과 (N−k)번째의 값의 관계가 실수와 허수에 해당하여 처음 N/2 부분의 IDFT의 계산으로 DCT를 구할 수 있다. 따라서 곱셈수는 radix-2 FFT의 절반이 되며 표 2.4에 나타낸 것과 같다(부록 2).

위의 알고리즘의 계산과정을 그림 2.5에 간단히 표시하였다.

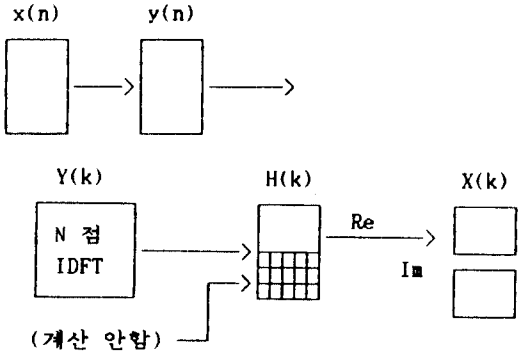


그림 2.5 N점 IDFT를 이용한 N점 DCT의 계산

[3] Vetterli 알고리즘

입력수열 x(n)에 대한 DFT X(k)의 실수와 허수의 값은 다음과 같다(부록 3).

$$RE[X(k)]=\underbrace{\sum_{n=0}^{N/2-1}x(2n)\ COS\left[\frac{2\pi nk}{N/2}\right]}_{\text{N/2 DFT}}+\underbrace{\sum_{n=0}^{N/4-1}(x(2n+1)+x(N-2n-1))\ COS\left[\frac{2\pi(2n+1)k}{4x(N/4)}\right]}_{\text{N/4 DCT}} \quad (2.35a)$$

$$IM[X(k)]=\underbrace{\sum_{n=0}^{N/2-1}x(2n)\ SIN\left[\frac{2\pi nk}{N/2}\right]}_{\text{N/2 DFT}}+\underbrace{\sum_{n=0}^{N/4-1}(x(2n+1)+x(N-2n-1))\ SIN\left[\frac{2\pi(2n+1)k}{4x(N/4)}\right]}_{\text{N/4 DCT}} \quad (2.35b)$$

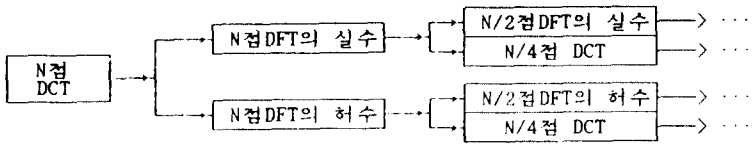


그림 2.6. DFT를 이용한 순환적인 구조

위의 실수와 허수부분의 첫항은 x(n)의 N/2 DFT의 실수와 허수부분이며 두번째항은 N/4 DCT이다.

식(2.53)와 cos 함수의 성질을 이용하면 (2.31b)식은 다음과 같이 된다.

$$X(k) = \cos\left[\frac{2\pi k}{4N}\right] \mathrm{Re}[Y(k)] - \sin\left[\frac{2\pi k}{4N}\right] \mathrm{Im}[Y(k)] \qquad (2.36)$$

여기서 Y(k)는 (2.33)식에 정의한 수열 y(n)의 N점 DFT이다. 즉, (2.37)식은 N점 DFT의 나누어짐과 cosine 덧셈정리를 이용하여 DCT 수열을 순환적인 형태로 만들고 N=2 또는 N=4일 때까지 계속해서 N을 나누어 가는 방법이다. 이러한 과정을 요약하여 그림 2.6에 나타내었다.

[4] WHT를 이용한 알고리즘

D.Hein은 Walsh-ordered WH 행렬 H_w의 성질을 이용하여 $N=2^n$의 데이타의 DCT가 Walsh-ordered WHT의 결과에 다음의 변환행렬 T(k)을 곱하는 방법으로 구해짐을 보였다(부록 4).

$$T(k) = [C'][H_w']^T \qquad (2.37)$$

단, ^{T} : 전치(transpose)를 의미한다.

여기서 [C'], [H_w']은 각각 BRO(bit reversed order)한 DCT, WHT의 커널을 의미한다. 그리고 T(k)는 orthonormal한 행렬인 C와 H_w의 합이므로 orthonormal 행렬이며, 블럭대각선(Block diagonal) 구조이다.

계산에 있어서 왈쉬 아다마르 변환이 곱셈이 없음을 이용하여 왈쉬변환의 결과에 변환행렬 T(k)와 행렬계산함으로써 곱셈이 감소되는 계산을 할 수 있다. N=8일 때 직접 계산에 의한 곱셈이 64회이나, 왈쉬변환을 이용하면 24회의 곱셈이 필요하게 된다. 이 때 변환행렬 T(8)은 다음과 같다.

$$[T_8]_{BRO} = \begin{bmatrix} [1.0] & & & \\ & [1.0] & & \\ & & \begin{bmatrix} 0.923 & 0.383 \\ -0.383 & 0.923 \end{bmatrix} & 0 \\ & 0 & & \begin{bmatrix} 0.907 & -0.075 & 0.375 & 0.180 \\ 0.214 & 0.768 & -0.513 & 0.318 \\ -0.318 & 0.513 & 0.768 & 0.214 \\ -0.180 & -0.375 & -0.075 & 0.907 \end{bmatrix} \end{bmatrix} \qquad (2.38)$$

[5] Chen 등에 의한 고속 알고리즘

입력수열 {x(n)}을 열 벡터(column vector)로 표시하면 변환수열은 다음과 같이 표시된다.

$$[X] = [C_N][x] \qquad (2.39)$$

여기서 $[C_N]$은 N×N 행렬이고 정규화한(normalize) 요소이다. 이해를 돕기 위해서 먼저 $[C_2]$와 $[C_4]$을 보이면 식(2.40a)이고, 행렬 $[C_4]$의 나누어진 구성형태는 식(2.40b)이다.

$$[C_1]=\begin{bmatrix}\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}}\\ C_4^1 & C_4^3\end{bmatrix},\quad [C_4]=\begin{bmatrix}\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}}\\ C_4^1 & C_4^3 & C_4^5 & C_4^7\\ C_4^2 & C_4^6 & C_4^{10} & C_4^{14}\\ C_4^3 & C_4^9 & C_4^{15} & C_4^{21}\end{bmatrix} \tag{2.40a}$$

$$[C_4]=\begin{bmatrix}1&0&0&0\\0&0&0&1\\0&1&0&0\\0&0&1&0\end{bmatrix}\begin{bmatrix}\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}}\\ C_4^1 & C_4^3 & C_4^5 & C_4^7\\ C_4^2 & -C_4^6 & C_4^{10} & -C_4^{14}\\ C_4^3 & C_4^9 & -C_4^{15} & -C_4^{21}\end{bmatrix}$$

$$=\begin{bmatrix}1&0&0&0\\0&0&0&1\\0&1&0&0\\0&0&1&0\end{bmatrix}\begin{bmatrix}\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 & 0\\ C_4^1 & C_4^3 & 0 & 0\\ 0 & 0 & -C_4^1 & C_4^3\\ 0 & 0 & C_4^3 & C_4^1\end{bmatrix}\begin{bmatrix}1&0&0&1\\0&1&1&0\\0&1&-1&0\\1&0&0&-1\end{bmatrix} \tag{2.40b}$$

단, $C_k^l=\cos[i\pi/k]$은 코사인 계수를 간략히 하기 위해서 사용되었다.

위의 예에서 보는 바와 같이 $[C_4]$은 $[C_2]$로 표현되는 것을 이용하여, 2의 제곱인 N에 대하여 $[C_N]$을 일반화하여 나타내면 식(2.41a)와 같다.

$$[C_N] = [P_N]\begin{bmatrix}C_{N/2} & 0\\ 0 & R_{N/2}\end{bmatrix}[B_N] \tag{2.41a}$$

단.

$$[B_N] = \begin{bmatrix}I_{N/2} & I_{N/2}\\ I_{N/2} & -I_{N/2}\end{bmatrix} \tag{2.41b}$$

$$[R_{N/2}] = \left[\cos\frac{(2n+1)(2k+1)\pi}{2N}\right]_{n,k=0,1,\cdots,N-1} \tag{2.41c}$$

$[P_N]$은 데이타의 짝수 열에 대하여 순차적으로, 홀수의 열에 대하여 감소적인 차수로 배열하는 역할을 하며, DCT수열 $[X]$를 BRO에서 NO(natural order)로 바꿔준다. 버터플라이 행렬인 $[B_N]$은 단위행렬의 합과 역 단위행렬로 표현할 수 있다. 행렬 $[R_{N/2}]$은 $[R_{N/2}]$의 열과 행을 바꿈으로써 얻어지며, $[P_{N/2}]$은 DCT-4의 변환행렬로 더 이상 순환이 이루어지지 않지만 $(2\log_2 N-3)$개의 영이 많은 행렬의 곱으로 만드는 방법으로 구한다. 그리고 $[C_N]$은 $[C_2]$일 때까지 순환을 계속하여 적은 곱셈으로 계산할 수 있다. 즉, Chen의 고속 알고리즘은 행렬 분해(matrix decomposition)에 기본을 두고 있다. 그림 2.7은 이 방법을 사용한 것으로 N=8에 대한 순방향 신호흐름도이다.

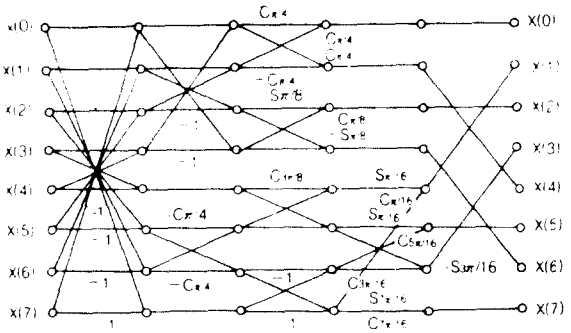


그림 2.7 Chen 알고리즘의 신호흐름도

[6] Lee 알고리즘

Lee는 IDCT(inverse DCT)에 대한 알고리즘을 제시했다.

정규화한 IDCT :

$$x(k) = \sum_{n=0}^{N-1} X(n)\ \cos\left[\frac{\pi(2k+1)n}{2N}\right] \tag{2.42}$$

이 식에서 x(k)의 n을 짝수번째와 홀수번째로 나누어 표시하면 식(2.44)과 같이 되는데 여기서 g(k)는 짝수번째의 k에 대한 N/2점 IDCT이다.

$$x(k) = g(k) + h'(k)$$
$$x(N-k) = g(k) - h'(k)$$
$$k=0,1,\cdots,N/2-1$$

단.

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

$$h'(k) = \sum_{n=0}^{N/2-1} X(2n+1) \cos\left[\frac{\pi(2k+1)(2n+1)}{2N}\right] \quad (2.43)$$

그리고 h'(k)를 수학적 계산을 이용하여 다른 N/2점 IDCT의 새로운 h(k)을 구한다. 이러한 g(k), h(k)와 DCT 커널인 코사인 함수의 성질을 이용하여 식(2.43)을 정리하면 식(2.44)가 된다(부록 5).

$$x(k) = g(k) + (1/2\, C_{2N}^{(2k+1)})\, h(k)$$

$$x(N-k) = g(k) - (1/\, C_{2N}^{(2k+1)})\, h(k)$$

$$k=0,1,\cdots,N/2-1 \quad (2.44)$$

단.

$$C_{2N}^{(2k+1)2n} = \cos\frac{\pi(2k+1)}{2N}$$

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

식(2.44)는 N점 IDCT을 N/2점 IDCT의 합/차로 나타낼 수 있음을 나타내며, 이러한 과정의 반복으로 IDCT를 더욱 더 분해할 수 있다.

DCT 수열은 직교변환이므로 IDCT의 전치로서 얻어질 수 있는데, 이것은 IDCT 신호흐름도에서 화살표의 방향을 반대로 하여 얻을 수 있다. 이 알고리즘 FFT와 유사한 전개방식을 사용하였으며, 따라서 FFT신호 흐름도와 비슷하다. 그림 2.8은 N=8일 때의 신호흐름도이다.

최근 서울대 계측제어공학과 조남익, 이상욱 교수 팀에서 2차원 고속 DCT를 1차원 DCT로 쉽게 처리하는 기법을 IEEE, Trans. on CAS, Vol. 38, No. 3(1991. 3)에 발표하였다. 이에 대한 분석은 지면 관계로 생략한다.

[7] 순환적인 DCT

식(2.34)과 같은 입력수열 재배열을 통해서 새로운 DCT 커널을 생각할 수 있고, 커널의

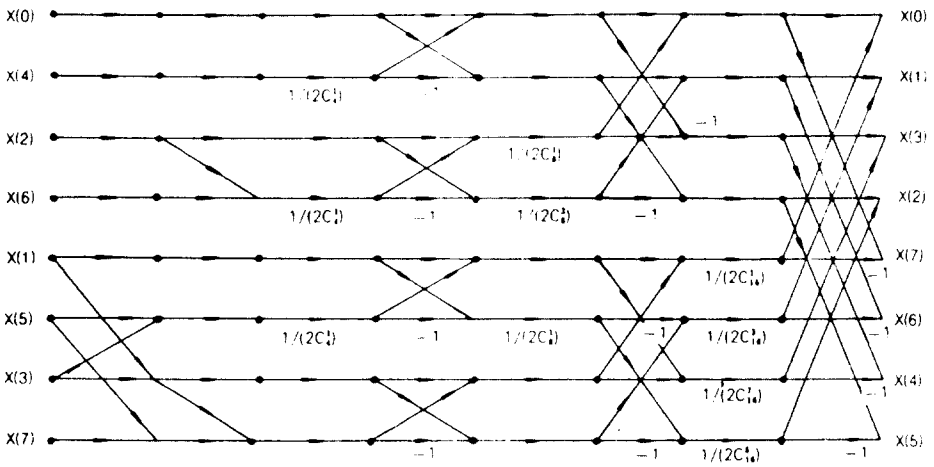


그림 2.8 Lee 알고리즘의 신호 흐름도

대칭적인(symmetric) 성질을 발전시켜 DCT를 순환적인 구조로 만들었다. 이해를 쉽게 하기 위해서 먼저 N=4, N=8에 대한 커널을 나타내면

N=4일 때

$$\begin{bmatrix} X(0) \\ X(2) \\ X(1) \\ X(3) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ \alpha & -\alpha & \alpha & -\alpha \\ \beta & -\delta & -\beta & \delta \\ \delta & \beta & -\delta & -\beta \end{bmatrix} \begin{bmatrix} x(0) \\ x(2) \\ x(3) \\ x(1) \end{bmatrix}$$

N=8일 때

$$\begin{bmatrix} X(0) \\ X(4) \\ X(2) \\ X(6) \\ X(1) \\ X(5) \\ X(3) \\ X(7) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ \alpha & -\alpha & \alpha & -\alpha & \alpha & -\alpha & \alpha & -\alpha \\ \beta & -\delta & -\beta & \delta & \beta & -\delta & -\beta & \delta \\ \delta & \beta & -\delta & -\beta & \delta & \beta & -\delta & -\beta \\ \sigma & \mu & -\tau & \varepsilon & -\sigma & -\mu & \tau & -\varepsilon \\ \mu & \tau & -\varepsilon & \sigma & -\mu & -\tau & \varepsilon & -\sigma \\ \varepsilon & -\sigma & \mu & \tau & -\varepsilon & \sigma & -\mu & \tau \\ \tau & \varepsilon & \sigma & \mu & -\tau & -\varepsilon & -\sigma & -\mu \end{bmatrix} \begin{bmatrix} x(0) \\ x(2) \\ x(4) \\ x(6) \\ x(7) \\ x(5) \\ x(3) \\ x(1) \end{bmatrix}$$

$\alpha=1/\sqrt{2}$, $\beta=C_8^1$, $\delta=S_8^1$, $\sigma=C_{16}^1$, $\varepsilon=C_{16}^3$, $\mu=S_{16}^1$, $\tau=S_{16}^3$

(2.45)

이다. 윗 식을 살펴보면 열(row) 길이의 절반을 중심으로 윗쪽은 데이타 길이가 절반인 커널이고 아래쪽은 부호가 서로 반대인 새로운 행렬이다. 따라서 N에 대하여 일반적인 구조를 생각할 수 있다.

$$T(N) = \begin{bmatrix} T(N/2) & T(N/2) \\ D(N/2) & -D(N/2) \end{bmatrix} \quad (2.46)$$

단, D=RLR T Q

여기서 D(N/2)은 행렬분해에 의해 T(N/2), BRO을 수행하는 R행렬, 로우 삼각행렬 L 및 대각행렬(diagonal) Q의 곱으로 이루어져 있다. 위의 식을 이용하여 DCT 수열을 나타내면 식(2.47)과 같다.

$$\begin{bmatrix} (X_e)_{BRO} \\ (X_o)_{BRO} \end{bmatrix} = \begin{bmatrix} [T_{N/2}] & \\ [K_{N/2}][T_{N/2}][Q_{N/2}] & \\ & [T_{N/2}] \\ & -[K_{N/2}][T_{N/2}][Q_{N/2}] \end{bmatrix} \begin{bmatrix} (X_e) \\ (X_o)_{BRO} \end{bmatrix} \quad (2.47)$$

단, R : reverse, K=RLR

즉, 이 알고리즘은 N점 DCT계산을 N/2점 DCT의 계산으로 얻을 수 있음을 이용하여, DCT 수열을 기본적인 DCT까지 계속 순환하여 얻는다. 따라서 곱셈과 덧셈수가 적어 빠른 계산을 할 수 있다. 그림 2.9는 이 구조를 구현한 것이다.

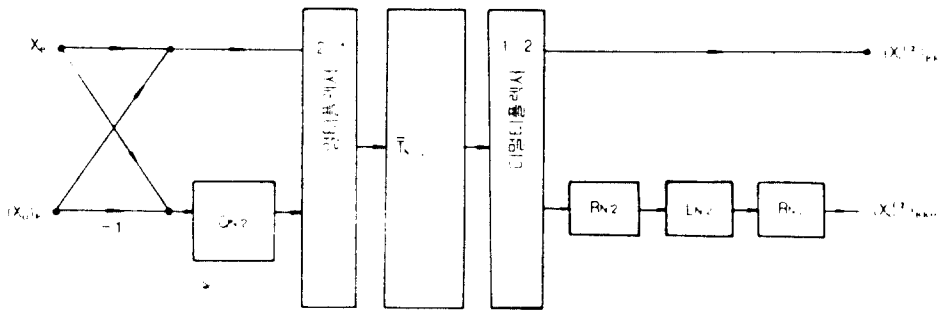


그림 2.9 다중화를 이용한 N점 DCT의 순환적인 구조

3. 새로운 직교변환 수법

이산적 코사인 변환(DCT)과는 달리, Karhunen Loven 변환(KLT)을 근사하여 더욱 고속연산법을 가지는 수법으로서, Discrete Linear Base 변환[34]과 Alpha 변환[35]이 발표되었다. 어느 것도 비교적 작은(4×4, 8×8 정도) 화상 압축용으로 이용되고 있지만, 다른 변환과의 비교 등에 관해서는, 앞으로 검토가 필요하다.

우선, Walsh-Hadamard 변환(WHT)을 기본으로 다른 직교변환을 유도하는 수법에 관한 연구도 활발히 이루어지고 있다[36][38]. 이것은 이론적으로도 흥미있는 문제이다. 단, 실제 화상처리에의 이용에 대해서는, WHT에서 변환용 행렬의 축적, 계산시간등의 점에서 비교적 작은 화상에의 적용에 국한하여 고려되고 있다.

3.1. Discrete Linear Base 변환(DLBT)[34]

Discrete Linear Base 변환(DLBT)은 WHT의 계산의 용이성과 KLT의 우수한 성능과의 양호한 trade off를 언급하고 있다.

DBLT의 기저 벡터는, 다음의 조건을 만족하는 형태로 생성된다.

i) N개의 벡터 $V_1, \cdots V_N \in R_N$은 서로 직교한다.

ii) 벡터의 성분은 정수치이며 공통인수를 갖지 않는다.

iii) 각 벡터는 even 또는 odd

단, $V = (a_1, .., a_N)$은

even , if $a_i = a_{N+1-i}$

odd , if $a_i = -a_{N+1-i}$

$(i = 1, .., N)$

생성된 벡터는 Sequency순으로 N=4인 경우의 예를 다음에 보인다.

[예] r = 1, s = -2 (문헌 34를 참조)

$$\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \begin{bmatrix} 3 \\ 1 \\ -1 \\ -3 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \\ -1 \\ 1 \end{bmatrix} \begin{bmatrix} 1 \\ -3 \\ 3 \\ -1 \end{bmatrix}$$

역시, R^N의 벡터가 2개의 서차원 부분공간, R^{N1}, R^{N2}($N = N_2 \cdot N_2$)의 기저벡터를 이용하여 표현된다는 원리에 근거한 고속연산법이 존재한다. 연산량에 대해서는, 가령 6×6 화소의 화상에 대해 KLT에서는 2,592회의 부동소수점 연산이 필요한 것에 대해 DLBT에서는 864회의 정수연산으로 끝난다.

3.2. Alpha 변환(AT)[35]

Alpha 변환은 신호의 WHT을 계산한 후에 신호의 통계적 성질에 따라 선택된 파라메터 α를 포함한 행렬을 곱해 KLT에 근사한 결과를 얻는 수법이다.

N=2의 경우에 대해서 설명한다. 입력 데이타의 WHT 결과를 (h_1, h_2)라 하면 Alpha 변환은 식(3.1)로 주어진다.

$$\begin{bmatrix} g_1 \\ g_1 \end{bmatrix} = \begin{bmatrix} h_1 \\ h_2 \end{bmatrix} \begin{bmatrix} A_2 \end{bmatrix},$$

$$A_2 = \frac{1}{\sqrt{1+\alpha^2}} \begin{bmatrix} 1 & \alpha \\ -\alpha & 1 \end{bmatrix} \qquad (3.1)$$

Alpha 행렬 (unitary)

여기서, 파라메터 α는, $F(\alpha) = \overline{g_1^2} / (\overline{g_1^2} + \overline{g_2^2})$이 최대가 되도록 선택된다. 역시, order 2의 Alpha 변환은 KLT와 엄밀히 일치한다.

[예] order 4의 Alpha 행렬은 다음의 2가지가 존재한다.

$$A_4^1 = \begin{bmatrix} 1 & 0 & \alpha & 0 \\ 0 & 1 & 0 & -\alpha \\ -\alpha & 0 & 1 & 0 \\ 0 & \alpha & 0 & 1 \end{bmatrix}$$

$$A_i^2 = \begin{bmatrix} 1 & 0 & 0 & \alpha \\ 0 & 1 & -\alpha & 0 \\ 0 & \alpha & 1 & 0 \\ -\alpha & 0 & 0 & 1 \end{bmatrix} \quad (3.2)$$

3.3. WHT를 기본으로 한 기타 직교변환의 계산법

WHT를 거쳐 DFT를 계산하는 수법에 대해서는 이전부터 검토되어 왔다. 문헌 36)에는 이 수법에 대하여 연구동향 및 문헌 리스트가 소개되어 있다.

우선, 최근 Hein과 Ahmed[37]는, Walsh-ordered WH 행렬 $[H_W(n)]$의 성질을 이용하여 $N=2^n$ 점 데이타의 DCT가 Walsh-ordered WHT의 결과에 다음의 변환행렬 $[T(n)]$을 곱하는 방법으로 구해짐을 보였다.

$$[T(n)] = [\hat{\Lambda}(n)]\,[\hat{H}_W(n)]^t \quad (3.3)$$

단, $[\hat{\Lambda}(n)]$, $[\hat{H}_W(n)]$은 각각 행을 비트 역순으로 병렬변환한 DCT 행렬, WH 행렬이다. t는 전치(轉置)를 나타낸다. 여기서, $[T(n)]$은 orthonormal 이며, block diagonal 구조를 갖는다.

문헌 37)에는 N=8의 경우에 대하여 기술하고 있고, 본 수법에 의하면 DCT의 직접 계산에 비하여 연산횟수가 약간 감소하게 된다(예를 들면 승산횟수, 64회→20회로 감소). 본 수법은 8 이외의 N에 대해서도 일반화되어, 64 이하의 N에 대해서는 2N점 FFT를 이용하는 수법[28]보다도 계산시간이 단축된다고 보고되어 있다.

Jones와 Hein[38]은 Fourier 변환, slant 변환, DCT, WHT를 포함하여 일반적으로 이용되는 직교변환이 Even / Odd Transform*,**이며, Even / Odd Transform끼리는 소행렬을 곱하는 방법으로 상호변환됨을 지적하고 있다.

3.4. 국소적 DFT를 순차 계산하는 경우의 효율적 계산법

Tamimoto[47]는, (N×N)점 화상에 대하여, 1화소씩 중심을 옮기면서, (M×M)점 창에서 국소적인 2차원 DFT를 순차계산하기 위한 효율적 수법을 보이고 있다. 각 창마다에 단순히 2차원 DFT를 행하는데는 $O(N^2M^2 \log M)$의 계산이 필요하다. 이에 대해, 창의 끝 부분에서 행 방향까지는 열방향의 1차원 DFT 결과에 착안하여, 계산수법을 공부함으로써, $O(N^2M^2)$의 계산으로 끝나게 된다. 이 수법은 texture의 기술(記述) 등, 화상해석에서 유용하게 응용된다. 편의상, 1차원의 경우에 대하여 설명한다. 데이터 $f_n(n=0, 1,\cdots)$의 부분계열(f_h, $f_{h+1},\cdots,f_{h+N-1}$)의 Fourier 변환의 n번째 항을 $F_n^{(k)}$라 하면,

$$F_n^{(k)} = \sum_{m=k}^{k+N-1} f_m \exp\left(-j2\pi \frac{(m-k)n}{\pi}\right) \quad (3.4)$$

이 때, 다음의 관계식이 성립된다.

$$F_n^{(k+1)} = \exp(j2n\pi/N)\,(F_n^{(k)} - f_k + f_{k+N}) \quad (3.5)$$

식(3.5)에 의하면, 긴 N의 변환창을 1점씩 옮겨서, N점 변환을 갱신하려면, 2N회의 가산과 N회의 복소승산이 필요하다.

[4] 2차원 직교변환에 필요한 계산시간의 비교

2차원 직교변환의 경우, 변환을 위한 기본적인 승산, 가산회로가 동일하여도, 행방향 및 열방향에 대한 계산순서에 따라서 전체로서의 계산시간

주) *Even / Odd Transform에서는 변환 벡터의 반수가 even vector이며, 나머지 반이 odd vector이다.
**Even / Odd Vector 계수가 무상관이 되도록 한 상관행렬을 가지는 데이타 클래스에 대해서는 KLT도 Even / Odd Transform이 된다.

에는 큰 차이를 발생하며, 특히, 화상 사이즈가 큰 경우에는, 이에 의한 영향이 현저하게 된다. 고속 기억용량이 한정되어 있는 경우에는, 앞서 제안된 무전치 2차원 변환법에 의해, 효율적으로 계산을 행하게 된다. 여기서는, 이 수법이 가상기억 계산기에서 대규모 화상의 변환을 행하는 경우에도 대단히 유용함을 보인다. 우선, 다음 계산수법에 대해서도 비교를 행한 결과를 보인다. 또한, 직교변환으로서는 기수 2의 FFT 및 Hadamard-ordered WHT를 다룬다.

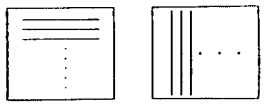
4.1. 2차원 직교변환의 계산순서

2차원 직교변환의 대표적 계산순서로서, 무전치 연산법을 포함하여 그림 5.1에 보인 4가지를 다룬다.

4.2. 계산시간 측정용 프로그램의 구성

[1] Hadamard-ordered WHT : 2차원 랜덤 데이타를 이용하여, 타입 1~4의 계산순서에 대하여, 버터 플라이 연산(실수 가감산만)에 소요되는 시간을 측정한다.

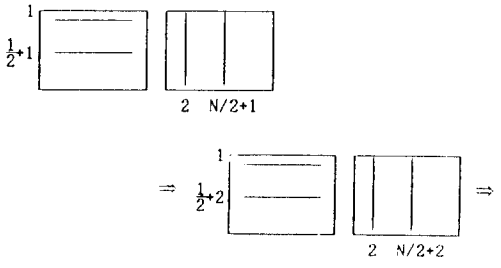
[type 1] 행방향 연산 → 열방향 연산



◇기본적 고찰 방법

◇1차원 직교변환을 단순히 행방향, 열방향에 순차 적용

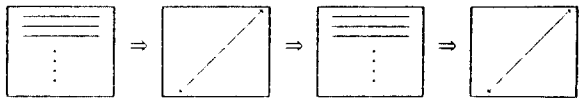
[type 2] 행과 열방향 교대로 버터플라이 연산을 수행한다.



◇화상배열은 정방향 행렬로 하는 것이 바람직하다.

◇어떤 화상과 어떤 화상에 버터플라이 연산을 행하는가를 결정하기 위한 계산량이 적어진다.

[type 3] 행방향 연산 → 전치 → 행방향 연산 → 전치

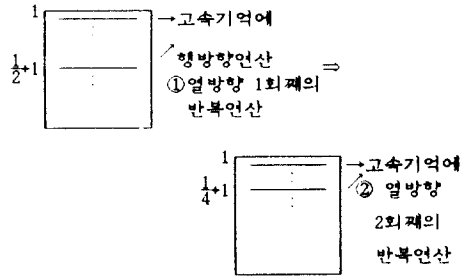


◇전치처리가 필요

◇정방향 행렬이 아니면 전치는 번거롭게 된다.

◇정방향 행렬이면 고속 전치 알고리즘을 사용할 수 있지만, 어쨌든 연산에 시간이 걸린다(여기서는 단순한 전치를 채용).

[type 4] 무전치 2차원 변환법(무전치 연산법)



◇고속기억 용량이 작아도 가능

◇화상배열은 구형(矩形)행렬로 가능

그림 5.1 2차원 직교변환의 계산순서

[2] FFT : 기수 2의 Sande-Tukey FFT 알고리즘[2]을 이용한다. 변환용 원화상으로서는, 실수에 난수를, 허수에 0을 대입한 것을 사용한다.

또한, W의 멱, 따라서 sin, cos의 계산 및 bit reversed order에 대해서는 미리 작성

한 table을 참조하여 처리하게 된다. 계산시간으로서는, 버터플라이 연산과 bit reversed order 병렬변환에 필요한 시간을 추정한다.

5 결 론

직교변환에서의 최근의 화제에 대해서도, 화상처리를 염두에 두고, 고속연 산법과 새로운 수법을 중심으로 기술하였다. 각각의 수법에 대한 자세한 것은 원문을 참조하기 바란다. 직교변환에 관해서는, 고속연산법의 개발, 새로운 변환기술의 개발과 함께 활발한 연구가 진행되고 있다.

특히, DFT에 관해서는, DFT를 콘볼루션 계산으로 변환하여 콘볼루션을 어떤 방법(다항식 수법등)으로 고속 계산하려는 움직임이 현저하며 Winograd의 알고리즘(WFTA)도 이러한 흐름을 반영한 것이다. 한편, HDTV, G4 FAX, VCR 등의 영상신호를 ISDN으로 전송하기 위해서는 많은 데이타 압축이 필요하다. 이 때 DCT는 화상정보 압축기술로서 절대적으로 필요하며 H.261 표준으로 이용되고 있다.

參 考 文 獻

1. N. Ahmed and K.R. Rao : Orthogonal Transforms for Digital Signal Processing, p. 623, Springer-Verlag, 1975.
2. E.O. Brigham : The Fast Fourier Transform, p. 252, Prentice-Hall, 1974(宮川, 今井譯 : "高速 Fourier 變換", p. 262, 科學技術出版社, 1978).
3. 尾上守夫 : "直交變換技術－高速フーリエ變換を中心に－", テレビジョン, Vol. 31, No. 9, pp. 712~721, 1977. 9.
4. 宮川, 他 : デイヅタル信號處理, p. 270, 電子通信學會, 1975.
5. W.K. Pratt : Digital Image Processing, p. 750, Jhon Wiley & Sons, Inc., 1978.
6. P.A. Wintz : "Transform Picture Coding", Proc. IEEE, Vol. 60, No. 7, pp. 809~820, July 1972.
7. 직교변환 관계의 중요한 논문은 IEEE Selected Paper Series에 잘 정리되어 있으므로 참조하기 바람.
8. C.M. Rader : "Discrete Fourier Transforms when the number of data samples is prime", Proc. IEEE, Vol. 56, No. 6, pp. 1107~1108, June 1968.
9. S. Winograd : "Some bilinear forms whose multiplicative complexity depends on the fields of constants", IBM T.J. Watson Res. Ctr., Yorktown Height, NY, IBM Res. Rep., RC-5569, Oct. 10, 1975.
10. R.C. Agarwal and J.W. Cooley : "New algortihms for digital convolution", IEEE Trans., Vol. ASSP-25, No. 5, pp. 392~410, Oct. 1977.
11. S. Winograd : "On computing the discrete Fourier Transform", Proc. Nat. Acad. Sci. USA, Vol. 73, No. 4, pp. 1005~1006, Apr. 1976.
12. H.F. Silverman : "An introduction to programming the Winograd's Fourier transform algortihm(WFTA)", IEEE Trans., Vol. ASSP-25, No. 2, pp. 152~165, Apr. 1977.
13. 飯塚, 田中 : "フーリエ變換を高速に實行する最近の技術", 日經エレクトロニクス, No. 117, pp. 60~91, 1978. 1. 9.
14. D.P. Kolba and T.W. Parks : "A prime factor FFT algorithm using high-speed convolution", IEEE Trans., Vol. ASSP-25, No. 4, pp. 281~294, Aug., 1977.
15. H.J. Nussbaumer and P. Quandalle : "Fast computation of discrete Fourier transforms using polynomial transforms", ibid., Vol. ASSP-27, No. 2, pp. 169~181, Apr. 1979.
16. C.S. Burrus : "Index mappings for multidimensional fourmulation of the DFT and convolution", ibid, Vol. ASSP-25, No. 3, pp. 239~242, June 1977.
17. J. McMlellan and C.M. Rader : "There is something much faster than the fast Fourier transform", Seminar Notes, Oct. 21, 1976.(문헌 14를 참조).
18. S. Zohar : "A precison of Winograd's discrete Fourier transform algorithm", IEEE Trans., Vol. ASSP-27, No. 4, pp. 409~421. Aug. 1979.

19. T.W. Patterson : "A Winograd-Fourier transform algorithm for realvalued data", ibid, Vol. ASSP-27, No. 4, pp. 398~402, Aug. 1979.
20. R.W. Patterson and J.H. McClellan : "Fixed point error analysis of Winograd Fourier transform algorithms", ibid., Vol. ASSP-26, No. 5, pp. 447~455, Oct. 1978.
21. L.R. Morris : "A comparative study of time efficient FFT and WFTA programming for general purpose computers", ibid, Vol. ASSP-26, No. 2, pp. 141~150, Apr. 1978.
22. R.C. Singleton : "An algorithm for computing the mixed radix fast Fourier transform", ibid, Vol. AU-17, No. 2, pp. 93~103, June 1969.
23. L.R. Rabiner, R.W. Schafer and C.M. Rader : "The chirp z-transform algorithm and its application", Bell Syst. Tech. J., Vol. 48, No. 5, pp. 1249~1292, May-June 1969.
24. B.J. Fino and V.R. Algazi : "Unified matrix treatment ofthe fast Walsh-Hadamard transforms", IEEE Trans., Vol. C-25, No. 11, pp. 1142~1146, Nov. 1976.
25. J.W. Manz : "A sequency-ordered fast Walsh transform", ibid., Vol. AU-20, No. 3, pp. 204~205, Aug. 1972.
26. W.K. Pratt, W.H. Chen and L.R. Welch : "Slant transform image coding", ibid., Vol. COM-22, No. 8, pp. 1075~1093, Aug. 1974.
27. J.P. Crettez : "The relationship between two fast slant transforms in ordered form", Proc. of the 4th Int. Joint Conf. on Pattern Recog., pp. 445~448, 1978.
28. M.H. Lee, M. Kaveh, "Fast Hadamard Transform Based on a Simple Matrix Factoriz ation", IEEE Trans. on ASSP, Vol. 34, No. 6, pp. 1666~1667, 1986.
29. K.S. Shangmugam : "Comments on 'Discrete cosine transform', ibid., Vol. C-24, No. 7, pp. 759, July 1975.
30. W.H. Chen, C.H. Smith and S.C. Fralick : "A fast computational algortihm for the discrete cosine transform", ibid., Vol. COM-25, No. 9, pp. 1004~1009 Sept. 1977.
31. M.J. Narashima and A.M. Peterson : "On the computation of the discrete cosine transform", ibid., Vol. COM-26, No. 6, pp. 934~936, June 1978.
32. R.M. Haralick : "A storage efficient way to implement th discrete cosine transform", ibid., Vol. C-25, No. 7, pp. 764~765, July 1975.
33. W.H. Chen and C.H. Smith : "Adaptive coding of monochrome and color images", ibid., Vol. COM-25, No. 11, pp. 1285~1292, Nov. 1977.
34. R.M. Haralick and K.S. Shangmugam : "Computation study of a discrete linear basis for image data compression", ibid., Vol. SMC-4,No. 1, pp. 16~27, Jan. 1974.
35. G.E. Lowitz : "The Alpha transform : a Walsh-Hadamard generation which adapts to image stastics", Proc. of the 4th Int. Joint Conf. on Pattern Recog., pp. 441~444, Nov. 1978.
36. R. Kiti and K.H. Siemens : "Discrete Fourier transform via Walsh transform", IEEE Trans., Vol. ASSP 27, No. 3, pp. 288, June 1979.(이 분야의 연구동향 및 문헌리스트가 실려있다)
37. D. Hien and N. Ahmed : "On a real-time Walsh-Handamard/cosine transform image processor", ibid., Vol. EMC-20, No. 3, pp. 453~457, Aug. 1978.
38. H.W. Jones and D.N. Hein : "The Karhunen-Loeve, discrete cosine, and related transforms obtained via the Hadamard transform", International Telemetering Conference/USA, pp. 87~98, 1978.
39. R.C. Agarwal and C.S. Burrus : "Number theoretic transform to implement fast digital convolution", Proc IEEE, Vol. 63, No. 4, pp. 550~560, Apr. 1975.
40. R.C. Agarwal and C.S. Burrus : "Fast convolution using Fermat number transforms with applications to digital filtering", IEEE Trans., Vol. ASSP-22, No. 2, pp. 87~97, Apr. 1974.
41. C.M. Rader : "Discret convolution via Mersenne transforms", ibid., Vol. C-21, No. 12, pp. 1269~1273, Dec. 1972.
42. I.S. Reed, T.K. Truong, Y.S. Kwoh and E.L. Hall : "Image processing by transforms over a finite field", ibid., Vol. C 26, No. 9, pp. 8

74~881, Sept. 1977.

43. H.C. Andrews and C.L. Petterson : "Singular value decomposition(SVD) image coding", ibid., Vol. COM-24, No. 4, pp. 425~432, Apr. 1976.
44. H.C. Andrews and C.L. Petterson : "Outer product expansions and their uses in digital image processing", ibid., Vol. C-25, No. 2, pp. 140~148, Feb. 1976.
45. N.K. Bose and K.A. Prebhu : "Two-dimensional discrete Hilber transform and computational complexity aspects in its implementation" ibid., Vol. ASSP-27, No. 4, pp. 356~360, Aug. 1979.
46. K.R. Rao, MA., Narashimhan and K. Revuluri : "Image data processing by Hadamard-Harr"
47. J.S. Lim, Two-dimensinal signal and image processing, Englewood Cliffs, NJ : Prentice-Hall, 1989.
48. K.R. Rao and P.Yip, Discrete Cosine Transform, Academic Press, 1990.
49. N. Ahemd, T. Natarajan, K.R. Rao, "Discrete Cosine Transform", IEEE Trans. on Computers, Vol. C-24, pp. 90~93, Jan. 1974.
50. W.H. Chen, C.H. Smith and S.C. Fralick, "A Fast Computational Algorithm for the Discrete Cosine Transform", IEEE Trans. Commun., Vol. COM-25, No. 9, pp. 1004~1009, Sep. 1977.
51. M.J. Narasimha and A.M. Peterson, "On the Computation of the Discrete Cosine Transform", IEEE Trans. on Comm., Vol. COM-26, pp. 934~936, June 1978.
52. D. Hein and N. Ahmed, "On a real-time Walsh-haramard cosine transform image processor", IEEE Trans. Electromag. Compat., Vol. EMC-20, pp. 453~457, Aug. 1978.
53. M. Vetterli and H. Nussbaumer, "Simpe FFT and DCT algorithms with reduced number of sperations", Signal Processing, Vol. 6, pp. 267~278, Aug. 1984.
54. B.G. Lee, "A New Algorithm to Compte the Discrete Cosine Transform", IEEE Trans. on ASSP., Vol. ASSP-32, No. 6, pp. 1243~1245, Dec. 1984.
55. H.S. Hou, "A fast recusive algorithm for computing the DCT", IEEE Trans. on ASSP., Vol. ASSP-35, pp. 1455~1461, Oct. 1987.
56. R.J. Clark, "Transform Coding fo Images", Academic Press, 1985.
57. 이문호, The Weighted Hadamard / Discreter Cosine Transforms for image coding and their Systolic Array Processing, 도서출판 해외 마케팅 센타, 1990.
58. Nam Ik Cho, Sang Uk Lee, "Fast Algorithm and Implementation of 2-D Discrete Cosine Transform", IEEE Trans. on CAS, Vol. 38, No. 3, pp. 297~305, 1991.

부 록

1. Ahmend et al.의 알고리즘에서 식(2.32) 유도

데이타 수열 {x(n) : n=0, 1,…N-1}에 대한 DCT 수열 {X(k) : k=0, 1,…N-1}은 (2.31b)을 다시 쓰면

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

N점 DFT :

$$DFT_k = \sum_{n=0}^{N-1} x(n) \exp(-j \frac{2\pi nk}{N}) \qquad (A1.1)$$

$$RE(DFT)_k = \sum_{n=0}^{N-1} x(n) \cos \frac{2\pi nk}{N} \qquad (A1.2)$$

2N점 DFT :

$$DFT_k = \sum_{n=0}^{2N-1} x(n) \exp(-j \frac{2\pi nk}{2N}) \qquad (A1.3)$$

DFT와 DCT를 연관시키기 위해 먼저 DFT를 나타내면 식(A1.1)과 같이 되고 이것의 실수 부분을 취한 것이 식(A1.2)이다. 위의 DFT식을 길이 2N으로 계산하면 식(A1.3)과 같으며, 이 부분을 DCT 수열을 구하는데 이용한다.

$$X(k) = RE \sum_{n=0}^{N-1} x(n) \exp(-j \; k(n + \frac{1}{2}) \; \frac{\pi}{N}$$

$$= RE \; \exp(-j \frac{\pi k}{2N}) \sum_{n=0}^{N-1} x(n) \exp(-j \; \frac{kn\pi}{N}) \qquad (A1.4)$$

$$= RE \; \exp(-j \frac{\pi k}{2N}) \; \underbrace{\sum_{n=0}^{2N-1} x(n) \exp(-j \frac{kn\pi}{N})}_{\text{2N점 DFT}} \qquad (A1.5)$$

단, 식(A1.5)은 식(A1.4)이 N부터 2N-1까지 영(zero)으로 채워진 것이다.

2. Narasimha, Peterson의 알고리즘에서 식(2.34)의 계산

$$X(k) = RE \left[\exp(j \frac{k\pi}{2N}) \underbrace{\sum_{n=0}^{N-1} y(n) \exp(\frac{2\pi nk}{N})}_{H(k)} \right]$$

$$H(k)= \exp\left(j \frac{k\pi}{2N} \right) \frac{\sum_{n=0}^{N-1} y(n) \exp\left(\frac{2\pi nk}{N} \right)}{Y(k)} \qquad (A2.1)$$

$$H(N-k)=j[H(k)^*] \qquad (A2.2)$$

단, * : 공액 복소수

식(2.34)에서 [ ] 부분을 H(k)라 하고 N−K일 때의 값이 H(k) 값의 공액복소수의 허수 부분이므로 식(2.34)는 다음 식과 같이 쓸 수 있다.

$$X(k)= Re\ [H(k)] \qquad (A2.3)$$

$$X(N-k)= Im\ [H(k)] \qquad (A2.4)$$

즉, H(k)의 N/2점 복소수 계산에 의해 X(k) 값을 계산할 수 있다.

3. Vetterli의 알고리즘에서 식(2.35), (2.36)의 유도
기본적인 성질 :

$$DFT(k) = \cos DFT(k) - j \sin DFT(k) \qquad (A3.1)$$

$$DCT(-k) = DCT(k) \qquad (A3.2)$$

$$DCT(2N-k) = -DCT(k) \qquad (A3.3)$$

$$\cos DFT(N-k) = \cos DFT(k) \qquad (A3.4)$$

$$\sin DFT(N-k) = -\sin DFT(k) \qquad (A3.5)$$

위의 식(A3.1)에서 코사인 성분은 DFT의 실수 부분의 값으로 우함수의 특성을 가졌으며, 식(A3.4)을 이용하여 홀수번째의 것은 DCT로 표현 가능하며 다음과 같이 나타낼 수 있다.

$$RE[X(k)] = \sum_{n=0}^{N/2-1} x(2n) \cos\left[\frac{2\pi nk}{N/2} \right] + \sum_{n=0}^{N/4-1} (x(2n+1)+x(N-2n-1)) \cos\left[\frac{2\pi(2n+1)k}{4(N/4)} \right]$$

4. WHT을 이용한 알고리즘에서 식(2.37)의 유도

$$X(k)=[C]\ x(n) \tag{A4.1}$$
$$W(k)=[H_w]\ x(n) \tag{A4.2}$$
$$X'(k)=[C']\ x(n) \tag{A4.3}$$
$$W'(k)=[H_w']\ x(n) \tag{A4.4}$$

단, [C] : DCT 커널, $[H_w]$: WHT 커널, ' : BRO(bit-reversed-order)

위의 처음 두 식은 각각 DCT수열, 왈쉬 변환한 수열을 나타내며 나머지 두 식은 처음식을 BRO 취한 형태이다. 위의 식과 왈쉬 아다마르가 orthonoraml 한 성질을 이용하여 왈쉬변환 후 변환행렬을 곱함으로써 DCT 수열을 얻을 수 있다.

$$X'(k)=\underbrace{[C']\ [H_w']^T}_{T(k)}\ \underbrace{[H_w']\ x(n)}_{W'(k)} \tag{A4.5}$$

5. Lee 알고리즘에서 식(2.43)에서 식(2.44)의 유도

먼저 식(20)에서 g(k)가 N/2 IDCT로 나타낼 수 있음을 보이고, h'(k)로 N/2 IDCT을 나타내기 위해서 코사인 성분을 곱하여 새로운 h(k)을 만드는 과정을 전개하겠다.

g(k)의 코사인 성분 :

$$\cos\left[\frac{\pi(2k+1)2n}{2N}\right]=\cos\left[\frac{\pi(2k+1)\ n}{N}\right]=\underbrace{\cos\left[\frac{\pi(2k+1)\ n}{2*(N/2)}\right]}_{N/2\ \text{IDCT 커널}} \tag{A5.1}$$

다른 N/2 IDCT 커널 :

$$\cos\left[\frac{\pi(2k+1)}{2N}\right]*\cos\left[\frac{\pi(2k+1)(2n+1)}{2N}\right]$$
$$=\cos\left[\frac{\pi(2k+1)\ 2n}{2N}\right]+\cos\left[\frac{\pi(2k+1)2(n+1)}{2N}\right] \tag{A5.2}$$

$$\cos\left[\frac{\pi(2k+1)}{2N}\right]*h'(k)=\sum_{n=0}^{N/2-1}X(2n+1)\cos\left[\frac{\pi(2k+1)\ 2n}{2N}\right]$$
$$+\sum_{n=0}^{N/2-1}X(2n+1)\cos\left[\frac{\pi(2k+1)(2n+1)}{2N}\right] \tag{A5.3}$$

식(A5.2)는 다른 N/2 IDCT 커널이고 식(A5.3)은 식(A5.2)의 식에 합을 취한 형태이다. 식(A5.3)의 우측 둘째 항은 수학적 계산을 통하여 첫째 항과 코사인 부분이 같도록 만들어 우측 부분이 새로운 N/2 IDCT h(k)가 되도록 한다. 따라서 h(k)와 h'(k)를 관계지움으로써 식(2.43)으로부터 식(2.44)을 얻을 수 있다.

李 光 宰

저자약력

- 1986 : 전북대 전자공학과(학사)
- 1990 : 전북대 대학원 전자과(석사)
- 1987～현재 : 한국방송공사 전주총국 기술국

姜 承 先

저자약력

- 1988～현재 : 전북대학교 정보통신공학과

朴 周 用

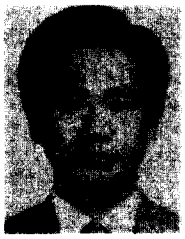
저자약력

- 1982 : 전북대 전자공학과(학사)
- 1986 : 전북대 전자공학과(석사)
- 1988～1990 : 전북대 전자공학과 박사과정중
- 1988～현재 : 전북대 정보통신공학과 조교

梁 根 鎬

저자약력

- 1989 : 전북대 전자공학과(학사)
- 1991 : 전북대 대학원 전자과(석사)
- 1990～현재 : 전북대학병원 의공과 연구원



李 門 浩

저자약력

- 일본동경대 전자과(공박)
- 미국 미네소타 주립대 전기과 포스트닥터
- 독일 하노버대학 전자과 초청교수
- 전기통신 기술사(1982) 및 유·무선설비기사 1급
- 1970~1980 : 남양 MBC 송신소장
- 1980~현재 : 전북대 정보통신공학과 교수