論 文

Hadamard 변환을 이용한 고속 2차원 DCT에 관한 연구

正會員 全 重 滿* 正會員 崔 源 昊** 正會員 崔 成 男* 正會員 朴 奎 泰*

A Study on Fast 2-D DCT Using Hadamard Transform

Joong Nam JEON,* Won Ho CHOI,** Sung Nam CHOI,* Kyu Tae PARK* *Regular Members*

要 約 본 논문에서는 예측오차의 분포와 비트할당표를 이용하여 계산량을 줄이는 방법으로 직접 2차원 WHT를 계산한 다음 상수행렬을 곱함으로써 2차원 DCT를 계산하는 방법을 제안하였다. 이 2차원 고속 DCT 알고리즘은 곱셈이 마지막 단계에 집중되어 있으며, 따라서 변환계수 중에서 양자화 과정에서 제외되는 변환계수의 수에 비례하여 계산량을 줄일 수 있는 특징이 있다.

계산량 비교를 위하여, 예측오차에 대한 비교할당표에 할당된 DCT 계수만을 구하고자 할 때의 계산량을 산출한 결과, 제안한 방식은 변환부호화에 할당된 화소당 평균 비트율이 0.6비트 이하일 때, 기존의 알고리즘 중에서 가장 계산량이 적은 고속 DCT 알고리즘 보다 계산량이 적게 되어, 변환부호화의 계산량 감축에 기여할 수 있음을 확인하였다.

ABSTRACT In this paper, A new 2-D DCT algorithm is proposed to reduce the computational amount of transform operation using the distribution of the motion compensated error signal and the bit allocation table. In the this algorithm, 2-D Walsh-Hadamard transform is directly computed and then multiplied by a constant matrix. Multiplications are concentrated on the final stage in this algorithm, thus the computational amount is reduced in proportion to the number of transform coefficients that are excluded from quantization.

The computational amount in computing only the DCT coefficients allocated to the bit allocation table is calculated. As the result, the number of multiplications is less than the algorithm known to have the fewest number of computations when less than 0.6 bits per pixel are allocated to transform coding for the motion compensated error image in the case of the proposed algorithm. Thus, it shows that the proposed algorithm is valid in reducing the computational loads of transform coding.

*延世大學校 電子工學科
Dept. of Electronics, Yonsei University
**蔚山大學校 電子및 電算機工學科
Dept. of Electronic Engineering, Woolsan University
論文番號 : 90-23(接受1990. 1. 17)

I. 서 론

DCT는 1차 Markov 신호원에 대하여 KLT (Karhunen-Loeve Transform)와 거의 같은 성능을 보유하고 있기 때문에[1] 데이타 압축, filte-

ring, 특징추출 등에 많이 사용된다.

1974년 Rao에 의해 DCT가 발표[2]된 이후, 그 동안 고속 알고리즘이 여러가지 발표되었으며, 이들은 알고리즘의 원리에 따라 WHT(Walsh-Hadamard Transform)나 FFT(Fast Fourier Transform)를 이용해서 간접적으로 계산하는 방식[3-5], DCT의 커넬행렬(kernel matrix)을 sparse행렬로 직접 분해(factorization)하여 계산하는 방식[6-7], 그리고 순환(recursive) 알고리즘에 의하여 계산하는 방식[8-10]으로 구분될 수 있다. 간접 계산 방식은 다른 변환을 이용하므로 불필요한 연산을 포함하는 수가 있고, 직접 분해 방법은 계산과정에서 커넬 행렬의 차수가 그대로 유지되기 때문에 2차원 알고리즘을 유도하기 어렵다. 그리고 순환 알고리즘들은 낮은 차수의 DCT를 사용해서 높은 차수의 DCT를 수행하기 때문에 2차원 알고리즘을 유도할 수 있으나, 마지막 단계가 순차적(sequential)으로 진행되는 단점이 있다. 그리고, DCT의 커넬행렬을 정수로 근사하여 변환하는 C-행렬변환(C-matrix transform) 도 발표되었다. 지금까지 발표된 고속 DCT 알고리즘들은 변환계수를 모두 구할 때의 계산량 감축을 목적으로 개발되었으며, 변환계수를 모두 사용하지 않는 영상부호화의 특성이나 변환되는 데이타의 특성을 고려하지 않았다.

변환부호화를 영상압축에 이용하는 원리는 일반적으로 변환계수가 주파수가 증가함에 따라 분산이 감소하는 순으로 분포하게 된다. 따라서, 낮은 주파수에 많은 비트를 할당하고, 높은 주파수에 적은 비트를 할당함으로써 효율적인 부호화가 가능하다. 본 연구에서는 DCT 부호화의 계산량 감축을 위하여, 변환에 사용되는 예측 오차 신호가 평균이 0인 라플라시안(Laplacian) 분포라는 점과 변환부호화에서는 비트할당표(bits allocation table)에서 선택된 변환계수만을 양자화 한다는 두가지 특징을 이용하여 계산량을 줄일 수 있는 2차원 고속 DCT 알고리즘을 제안하였다.

Ⅱ. 영상부호화에서 고속 DCT 알고리즘

이 장에서는 기존의 DCT 알고리즘과 이동보상후의 예측 오차신호의 분포와 비트할당표를 이용하여 DCT의 계산량을 감축하는 방법에 관하여 논한다.

Ⅱ-1. 고속 DCT 알고리즘의 비교

DCT는 1974년 Rao등에 의해 제안되었으며[1], 변환 공식과 역변환 공식은 다음과 같이 정의된다.

1차원 변환공식:

$$F(n)=\frac{2e(n)}{N}\sum_{j=0}^{N-1} f(j)\cos\frac{(2j+1)n\pi}{2N} \qquad (1)$$

1차원 역변환공식:

$$f(j)=\sum_{k=0}^{N-1} e(n)F(n)\cos\frac{(2j+1)n\pi}{2N} \qquad (2)$$

j, k=0,1, , N-1

$$e(k)=\begin{cases}1/\sqrt{2} & k=0\text{일때}\\ 1 & k=1,2,\cdots,N-1\text{ 일때}\end{cases}$$

이 변환 및 역변환 과정은 변환부호화에서 가장 계산량이 많이 소요되는 부분이다. 따라

표 1. 고속 DCT의 알고리즘의 분류표
Table 1 Classification table of the fast DCT algorithms

방식	원 리	참 고 문 헌		N=8일 때의 곱셈수	
		1차원	2차원	1차원(8×1)	2차원(8×8)
1	WHT 이용	[3]	없음	20	×
2	FFT 이용	[4,5]	[5]	12c	320c
3	DCT 커넬을 직접분해	[6]	없음	16	×
4	DCT 커넬을 순환적으로 분해	[8,10]	[8]	12	144

* c는 복소수 계산

서, 실제 응용에서는 계산량을 줄이기 위한 고속 알고리즘을 사용하게 되며, 지금까지 발표된 고속알고리즘들을 계산량 감축 원리에 따라 표1과 같이 분류할 수 있다.

Ⅱ-2. 오차신호 분포를 이용한 계산량 감축

이동보상후 발생되는 오차신호에는 화면내 물체의 비선형 이동, 탐색영역밖으로의 이동 그리고 회로상의 잡음(noise)등에 의하여 예측이 불가능한 화소들이 존재하며, 이들을 중요화소(significant pixel)라 하고, 중요화소가 포함된 블럭을 중요블럭(significant block)이라 한다. 잡음에 의한 중요화소는 불규칙(random)하게 분포하며, 그외 물체 이동에 의한 중요화소는 물체의 경계선을 따라 존재하고 라플라시안(Laplacian)분포이다.

이와 같이 오차영상에는 예측이 정확한 화소들이 많이 포함되고 있고 이들은 주로 편중되어 존재한다는 것을 알 수 있으며, 이를 다음과 같이 계산량 감축에 이용할 수 있다. 예측오차에는 중요화소가 많지 않으므로 중요블럭은 sparse 행렬로 취급할 수 있다. 방식1의 알고리즘은 먼저 고속 Hadamard 변환한 후에 식6의 행렬을 곱하여 DCT 계수를 구한다. 원리는 다음과 같다. 식1와 식2을 행렬(matrix)의 형태로 표현하면

$$\mathbf{F}=\mathbf{C}\,\mathbf{f} \tag{3}$$

$$\mathbf{f}=\mathbf{C}^{T}\,\mathbf{F} \tag{4}$$

이다. 여기서, **C**는 크기가 N×N인 DCT 변환 커넬의 행렬이고, ^{T}는 전치행렬(transpose matrix) 이고, **f**는 $[f_0, f_1,\cdots,f_{N-1}]^T$, F는 $[F_0, F_1,\cdots, F_{N-1}]^T$인 열행렬(column matrix)이다. 크기가 N×N인 Hadamard 변환 행렬을 **H**라 하면, $\mathbf{H}=\mathbf{H}^{-1}$이므로 식3는 다음과 같이 쓸 수 있다.

$$\begin{aligned}\mathbf{F}&=\mathbf{C}\,\mathbf{H}\,\mathbf{H}\,\mathbf{f}\\&=\mathbf{S}\,\mathbf{H}\,\mathbf{f}\end{aligned} \tag{5}$$

이때, S는 sparse 행렬이 되고, **Hf**는 **f**의 Hadamard 변환이다. N=8일때의 S는 다음과 같다. 문헌[3]은 C,H를 비트 반전(bit reverse)한 것으로, 식(6)과는 계수의 배치가 다르다.

$$S=\begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0.108 & 0.375 & 0 & 0.907 & 0 & 0 & -0.075\\
0 & 0 & 0 & 0 & 0 & 0.383 & 0.923 & 0\\
0 & 0.214 & 0.768 & 0 & -0.318 & 0 & 0 & 0.513\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0.318 & -0.513 & 0 & 0.214 & 0 & 0 & 0.768\\
0 & 0 & 0 & 0 & 0 & 0.923 & -0.383 & 0\\
0 & 0.907 & -0.075 & 0 & -0.180 & 0 & 0 & -0.375
\end{bmatrix} \tag{6}$$

N=8인 경우의 방식1의 곱셈수는 식 6의 실수부의 수와 같은 20회이다. 이와 같이, 이 방식은 곱셈수가 다른 방식보다 많다는 단점이 있으며, 더구나 영상데이타는 2차원이므로, N×N의 2차원 데이타를 열과 행으로 나누어 2N회의 1차원 변환으로 처리하여야 한다.

그림 1는 고속 Hadamard 변환의 흐름도이며, 일반적인 고속 알고리즘의 흐름도를 역순으로 표현한 것으로서 Hadamard 변환에서는 변환과정과 역변환 과정이 같기 때문에 역순으로 계산해도 같은 결과를 얻는다.

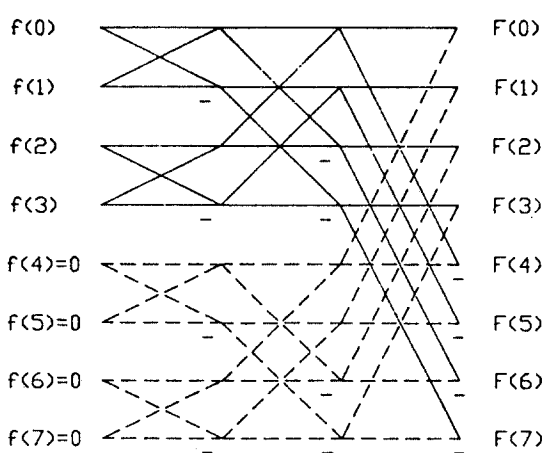


그림 1. 고속 Hadamard 변환의 흐름도
Fig. 1 Signal flow diagram for the fast Hadamard transform

그림 1에서와 같이 변환하려는 데이타의 값이

$$f(j)\neq 0 \quad j=0,1,2,3$$
$$f(j)=0 \quad j=4,5,6,7 \qquad (7)$$

이라고 할 때, 방식 1의 알고리즘은 첫단이 인접 화소간의 연산이므로, 그림 1의 점선으로 표시된 부분의 연산은 제거할 수 있다. 그러나, 방식 3과 방식 4의 알고리즘은 첫단이 f(j)와 f(N-1-j)의 쌍으로 되어 있으므로 오차신호에 0이 편중되어 있다는 점을 적용하기 곤란하다.

Ⅱ-3. 비트할당표를 이용한 계산량 감축

변환 부호화에서 실제로 부호화되는 계수는 비트할당표에 의해 선택되어진 변환계수만을 부호화한다. 화소당 평균비트율이 0.5와 2.0 bpp (bits per pixel)인 경우, 8×8 크기의 블럭에 대한 비트할당표의 예를 그림2에 제시하였다. 이 그림에서 비트가 할당한 변환 계수는 평균비트율에 따라 약 6—25%이고, 이들만이 양자화되고 부호화된다. 따라서, 변환과정에서 비트할당된 변환계수만을 구하여 부호화하더라도 변환부호화의 전체성능은 변하지 않는다.

1차원 신호의 변환부호화에서 변환계수를 선택적으로 계산할 경우, 방식1, 방식3, 방식4의 알고리즘에 의한 계산량을 비교해 보자. 1차원 데이타의 크기가 8이고, 그림 3과 같이 낮은 주파수 영역의 4개의 변환계수에 비트가 할당되었다고 가정한다. 방식2는 복소수 연산이므로 고려대상에서 제외한다.

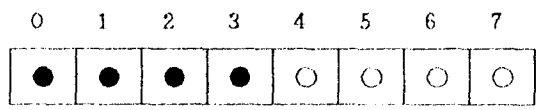
8 4 3 2 · · · ·
4 3 2 1 · · · ·
2 1 1 · · · · ·
1 · · · · · · ·
· · · · · · · ·
· · · · · · · ·
· · · · · · · ·
· · · · · · · ·

(a) 0.5 bpp인 경우

8 7 6 5 3 3 2 1
7 6 5 4 3 2 1 ·
5 5 4 3 3 2 1 ·
4 3 3 3 2 1 1 ·
3 2 2 1 1 1 · ·
2 2 2 1 1 · · ·
2 1 1 · · · · ·
1 1 · · · · · ·

(b) 2.0 bpp인 경우

그림 2. 비트할당표
Fig. 2 Bit allocation table



● : 비트할당 됨
○ : 비트할당 되지 않음

그림 3. 1차원 변환부호화의 비트할당표의 예
Fig. 3 An example of bit allocation table for the 1-dimensional transform coding

그림3의 비트할당표에서 선택된 변환계수만을 계산하고자 할 때, 방식4의 알고리즘의 곱셈이 전체 흐름도에 균일하게 분포되어 있고, 또 마지막 단계가 순차적으로 계산되므로 선택적으로 부호화하기 곤란하다. 그러나 방식1의 알고리즘은 정수 연산이 포함되어 있고 식7에서 비트가 할당된 열(row)를 곱하면 되므로 위의 4열의 곱셈만 하면 되고, 방식3의 알고리즘은 곱셈이 마지막 단계에 집중되어 있으므로, 그림3의 비트가 할당된 계수를 제외한 나머지 연산(그림에 ○로 표시된 부분)은 제거할 수 있으므로, 오히려 계산량이 방식4의 알고리즘 보다 줄어든다.

표 2. 방식1, 3, 4의 계산량 비교
Table 2 Comparision of the number of calculations between the method 1, 3 and 4

방식	계수를모두계산		계수를 선택적으로 계산	
	곱 셈	덧 셈	곱 셈	덧 셈
1	20	24+14	10	24+ 7
3	16	0+26	9	0+22
4	12	0+29	12	0+29

단, 덧셈수는 (정수 연산수 + 실수 연산수) 임.

지금까지는 1차원 데이타인 경우를 고찰하였으나, 영상부호화에서는 2차원 데이타를 처리하여야 한다. 이런 관점에서, 방식1의 알고리즘으로 2차원 데이터를 처리하기 위해서는 다음식을 계산하여야 한다.

$$\mathbf{F}=\mathbf{S}\ \mathbf{H}\ \mathbf{f}\ \mathbf{H}^T\ \mathbf{S}^T \tag{8}$$

단, **F**, **S**, **f**, **H**의 크기는 N×N 임.

식8의 가운데 식 $\mathbf{H}\ \mathbf{f}\ \mathbf{H}^T$는 2차원 Hadamard 변환이며, 직접 계산하는 알고리즘이 개발되어 있으나 식6의 sparse 행렬 **S**를 열과 행으로 곱해야 하므로 변수를 선택적으로 계산하기 곤란하다.

1차원 변환에서 방식3의 알고리즘은 N개의 변환계수 중 m개만을 구하고자 할 때, 2(N-m)회의 곱셈과 (N-m)회의 덧셈을 줄일 수 있다. 그러나 이 알고리즘은 변환 행렬 부분이 다시 크기가 N/4인 작은 행렬로 분해되지 않기 때문에, 2차원 변환을 직접 구하는 알고리즘의 개발이 어렵다.

따라서, 본 논문에서는 방식1의 알고리즘을 2차원 데이터에도 선택적으로 계산할 수 있는 알고리즘으로 개선한다.

Ⅲ. Hadamard 변환을 이용한 2차원 고속 DCT 알고리즘의 제안

이 장에서는 방식1의 알고리즘을 2차원으로 확장한다. 이를 위하여 우선 고속 2차원 Hadamard 변환 알고리즘에 대하여 설명하고, DCT 커넬 행렬에 WHT 커넬 행렬을 곱하여 구한 식6의 행렬 **S**를 일반항으로 표시하여 2차원에 적용하도록 한다.

Hadamard 변환의 커넬행렬은 다음과 같은 특징이 있다.

$$\mathbf{H}_2=\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix} \tag{9}$$

$$\mathbf{H}_N=\begin{bmatrix}\mathbf{H}_{N/2} & \mathbf{H}\mathbf{H}_{N/2}\\ \mathbf{H}_{N/2} & -\mathbf{H}_{N/2}\end{bmatrix} \tag{10}$$

$$\mathbf{H}_N=\ \mathbf{H}_N^T \tag{11}$$

길이가 N×N인 데이터 **f**의 2차원 Hadamard 변환은 다음식으로 정의된다.

$$\mathbf{F}_h\ =\ \mathbf{H}_N\ \mathbf{f}\ \mathbf{H}_N^T \tag{12}$$

식12의 행렬 **H**, **f**를 길이가 N/2×N/2인 네개의 부행렬로 분해하여 정리하면 다음식을 구할 수 있다.

$$\mathbf{F}_h=\begin{bmatrix}\mathbf{H}_{N/2} & \mathbf{H}_{N/2}\\ \mathbf{H}_{N/2} & -\mathbf{H}_{N/2}\end{bmatrix}\begin{bmatrix}\mathbf{f}_{00} & \mathbf{f}_{01}\\ \mathbf{f}_{10} & \mathbf{f}_{11}\end{bmatrix}\begin{bmatrix}\mathbf{H}_{N/2} & \mathbf{H}_{N/2}\\ \mathbf{H}_{N/2} & -\mathbf{H}_{N/2}\end{bmatrix}$$

$$=\begin{bmatrix}\mathbf{F}_{00}+\mathbf{F}_{01}+\mathbf{F}_{10}+\mathbf{F}_{11} & \mathbf{F}_{00}-\mathbf{F}_{01}+\mathbf{F}_{10}-\mathbf{F}_{11}\\ \mathbf{F}_{00}+\mathbf{F}_{01}-\mathbf{F}_{10}-\mathbf{F}_{11} & \mathbf{F}_{00}-\mathbf{F}_{01}-\mathbf{F}_{10}+\mathbf{F}_{11}\end{bmatrix} \tag{13}$$

단,

$$\begin{bmatrix}\mathbf{F}_{00}\\ \mathbf{F}_{01}\\ \mathbf{F}_{10}\\ \mathbf{F}_{11}\end{bmatrix}=\begin{bmatrix}\mathbf{H}_{N/2} & \mathbf{f}_{00} & \mathbf{H}_{N/2}\\ \mathbf{H}_{N/2} & \mathbf{f}_{01} & \mathbf{H}_{N/2}\\ \mathbf{H}_{N/2} & \mathbf{f}_{10} & \mathbf{H}_{N/2}\\ \mathbf{H}_{N/2} & \mathbf{f}_{11} & \mathbf{H}_{N/2}\end{bmatrix} \tag{14}$$

고속 2차원 Hadamard 변환은 식12-식14로 정의된다. 즉, 크기가 N/2×N/2인 4개의 식14를 계산함으로서 길이가 N×N인 Hadamard 변환을 할 수 있으며, 이 때의 계산량은 정수 덧셈 $2N^2\log_2 N$회다.

$$FC_N(n)=\sum_{j=0}^{N-1} f(j)\ \cos\frac{2^k(2j+1)n\pi}{2N} \tag{15a}$$

$$FS_N(n)=\sum_{j=0}^{N-1} f(j)\ \sin\frac{2^k(2j+1)n\pi}{2N} \tag{15b}$$

단, $k=0,1,\cdots,\log_2 N$

위 식을 고속 Hadamard 변환 알고리즘과 같이 f(j)의 인접한 데이터의 연산을 먼저 처리하도록 표현하면 다음식을 구할 수 있다.

$$FC_N(n)=\cos\frac{2^k n\pi}{2N}\sum_{j=0}^{N/2-1}\{f(2j)+f(2j+1)\}\cos\frac{2^{k+1}(2j+1)n\pi}{2N}+\sin\frac{2^k n\pi}{2N}\sum_{j=0}^{N/2-1}\{f(2j)-f(2j+1)\}\sin\frac{2^{k+1}(2j+1)n\pi}{2N} \quad (16a)$$

$$FS_N=\cos\frac{2^k n\pi}{2N}\sum_{j=0}^{N/2-1}\{f(2j)+1)\}\sin\frac{2^{k+1}(2j+1)n\pi}{2N}-\sin\frac{2^k n\pi}{2N}\sum_{j=0}^{N/2-1}\{f(2j)-2j+1)\}\cos\frac{2^{k+1}(2j+1)n\pi}{2N} \quad (16b)$$

식16a에서 k=0일 때가 정규화 계수를 제외한 DCT 정의식이다. 여기에 Σ항의 수가 하나만 남을 때까지 식16a와 식16b를 순환적으로 적용할 수 있으며, 그 결과 Σ항에는 Hadamard 변환식이 남게 되며, cos( )항과 sin( )항의 곱으로 구성된 n의 함수가 유도된다. 여기에 n=0, 1,…,N−1을 대입하여 구한 값을 행렬 **S**의 열에 놓으면 식6과 같이 같은 sparse 행렬을 구할 수 있다. **S**의 요소(element)를 S(n,j)라 하면

$$S(n,j)=\prod_{i=0}^{\log_2 N} cs\left(\frac{2^i n\pi}{2N}\right)=cs\left(\frac{2n\pi}{2N}\right)\ cs\left(\frac{4n\pi}{2N}\right)\cdots cs\left(\frac{n\pi}{2}\right) \quad (17)$$

(최하위비트 ~ 최상위비트: i의 2진수 표현; 짝수패리티)

단,

$$cs(\ )=\begin{cases}\cos(\ ) & \text{비트값이 0일 때}\\ (-1)^k\sin(\ ) & \text{비트값이 1일 때}\end{cases} \quad (18)$$

이며 S(n,j)는 다음과 같은 특징이 있다.

(1) $(\log_2 N+1)$개의 cos( )항과 sin( )항의 곱으로 구성된다.

(2) j를 2진수로 표현할 때의 각 비트 값이 0이면 cos( )이고, 비트값이 1이면 sin( )항이다.

(3) 식18의 k는 최하위비트(least significant bit)부터 계산하여 1이 홀수번째일 때는 0이고, 짝수번째일 때는 1이다. 단, 짝수 패리티항의 k값은 항상 0이다.

위 법칙에 의하여 계산한 크기가 8×1인 n번째 열행렬을 S_n이라 하면 식6의 **S**는

$$\mathbf{S}=[\mathbf{S}_0,\mathbf{S}_1,\ldots,\mathbf{S}_{N-1}]^T \quad (19)$$

$$\mathbf{S}_n^T=\begin{bmatrix}\cos(n\pi/16) & \cos(n\pi/8) & \cos(n\pi/4) & \cos(n\pi/2)\\ \sin(n\pi/16) & \cos(n\pi/8) & \cos(n\pi/4) & \sin(n\pi/2)\\ \mathrm{con}(n\pi/16) & \sin(n\pi/8) & \cos(n\pi/4) & \sin(n\pi/2)\\ \sin(n\pi/16) & \cos(-n\pi/8) & \cos(n\pi/4) & \cos(n\pi/2)\\ \sin(n\pi/16) & \sin(n\pi/8) & \cos(n\pi/4) & \sin(n\pi/2)\\ \cos(n\pi/16) & \cos(n\pi/8) & \sin(-n\pi/4) & \cos(n\pi/2)\\ \cos(n\pi/16) & \sin(n\pi/8) & \sin(-n\pi/4) & \cos(n\pi/2)\\ \sin(n\pi/16) & \sin(-n\pi/8) & \sin(n\pi/4) & \sin(n\pi/2)\end{bmatrix} \quad (20)$$

으로 표현할 수 있다. 따라서 방식1의 알고리즘은 다음과 같이 요약할 수 있다.

(1) 차원 데이터를 Hadamard 변환한다.

$$\mathbf{F}_h=\mathbf{H}_N\ \mathbf{f} \quad (21)$$

(2) $\mathbf{S}_n$을 곱하여 n번째 DCT 계수를 구하다.

$$\mathbf{F}(n)=\mathbf{S}_n\ \mathbf{F}_h \quad (22)$$

지금까지 설명한 1차원 알고리즘을 다음과 같이 2차원으로 확장할 수 있다. 단 $\mathbf{S}_2=\mathbf{S}_m^T\ \mathbf{S}_n$로 정의한다.

(1) 2차원 데이터를 Hadamard 변환다.

$$\mathbf{F}_h=\mathbf{H}_N\ \mathbf{f}\ \mathbf{H}_N^T \quad (23)$$

(2) $\mathbf{F}_h$의 원소에 크기가 N×N인 $\mathbf{S}_m^T\mathbf{S}_n$의 각 원소인 $\mathbf{S}_2(i,j)$를 곱한 후 0이 아닌 원소를 모두 더하여 (m, n)번째 DCT 계수를 구한다.

$$\begin{aligned} F(m,n) &= S_2(0,0)\,F_h(0,0)+S_2(0,1)\,F_h(0,1)+\cdots\cdots+S_2(0,N-1)\,F_h(0,N-1) \\ &\quad +S_2(1,0)\,F_h(1,0)+\cdots\cdots \qquad +S_2(1,N-1)\,F_h(1,N-1) \\ &\quad \vdots \qquad\qquad\qquad\qquad \vdots \\ &\quad +S_2(N-1,0)\,F_h(N-1,0)+\cdots\cdots \qquad +S_2(N-1,N-1)\,F_h(N-1,N-1) \\ &= \sum_{m=0}^{N-1}\sum_{n=0}^{N-1} S_2(m,n)\cdot F_h(m,n) \end{aligned} \tag{24}$$

위의 $S_m{}^T S_n$에는 값이 같은 원소들도 포함되어 있다. N=8인경우 0이 아닌 값을 갖는 요소수 중에서 값이 같은 원소를 제외한 실제 곱셈수를 그림4에 나타내었으며, 전체의 합계는 364회이다. 그러나 영상부호화에서 비트가 할당된 변환계수만을 구하고자 할 때는 이보다 상당히 줄어든다.

m/n	0	1	2	3	4	5	6	7
0	1	4	2	4	1	4	2	4
1	4	10	6	12	4	12	6	12
2	2	6	3	6	2	6	3	6
3	4	12	6	10	4	10	6	12
4	1	4	2	4	1	4	2	4
5	4	12	6	10	4	10	6	12
6	2	6	3	6	2	6	3	6
7	4	10	6	12	4	12	6	10

합계 : 364

그림 4. 행렬 $S_m{}^T S_n$내의 0이 아닌 요소수
Fig. 4 Number of nonzero elements in the matrix $S_m{}^T S_n$

Ⅳ. 영상부호화에서의 계산량 비교

실제로 영상을 이동보상과 DCT의 복합부호화하여 본 논문에서 제안하는 DCT 알고리즘의 계산량을 산출하였다. Ellen 영상과 Cronkite 영상 각 16화면으로부터 이동보상한 후의 오차영상 중 중요블럭을 DCT 변환하고, 각 계수의 분산(variance)에 비례하여 식25에 의하여 비트할당[14]을 하였다.

$$R_k = R + \frac{1}{2}\log_2 \frac{\sigma_k^2}{\left[\prod_{j=0}^{N-1}\sigma_j^2\right]^{1/N}} \tag{25}$$

R_k : k번째 변환계수의 양자화 비트수
R : 블럭당 평균비트수
σ_k : k번째 변환계수의 분산

그림7은 Ellen 영상과 Cronkite 영상에서 구한 변환계수 분산값의 분포도이며, DC 계수를 중심으로 라플라시안 분포의 형태이다. 이 분포에 대하여 식25로 화소당 비트율이 0.5-1.0으로 할당한 비트할당표를 그림 8에 표시하였다.

	0	1	2	3	4	5	6	7
0	19.15	10.61	6.28	3.65	1.93	0.94	0.43	0.22
1	11.56	7.41	4.37	2.61	1.39	0.76	0.33	0.18
2	7.47	4.47	3.18	1.96	1.80	0.66	0.29	0.18
3	4.09	2.56	1.96	1.27	0.81	0.47	0.23	0.18
4	2.10	1.6	1.12	0.78	0.55	0.35	0.19	0.14
5	1.06	0.92	0.45	0.31	0.23	0.16	0.16	0.12
6	0.69	0.61	0.43	0.27	0.25	0.16	0.14	0.10
7	0.47	0.47	0.33	0.02	0.17	0.15	0.01	0.09

그림 7. 변환 계수의 분산 분포
Fig. 7 Variance distribution of the transform coefficents

3 2 2 1 1 · · ·	3 2 2 2 1 1 · ·
2 2 2 1 1 · · ·	2 2 2 1 1 · · ·
2 2 1 1 1 · · ·	2 2 2 1 1 · · ·
1 1 1 1 · · · ·	2 1 1 1 1 · · ·
1 1 1 · · · · ·	1 1 1 · · · · ·
1 · · · · · · ·	1 1 · · · · · ·
· · · · · · · ·	· · · · · · · ·
(a) 0.5 bpp	(b) 0.6 bpp

3 3 2 2 1 1 · ·
3 2 2 2 1 1 · ·
2 2 2 1 1 1 · ·
2 1 1 1 1 · · ·
1 1 1 1 · · · ·
1 1 · · · · · ·
1 · · · · · · ·
· · · · · · · ·

(c) 0.7 bpp

3 3 2 2 1 1 · ·
3 2 2 2 1 1 · ·
2 2 2 1 1 1 · ·
2 2 1 1 1 · · ·
2 1 1 1 1 · · ·
1 1 1 · · · · ·
1 1 · · · · · ·
· · · · · · · ·

(d) 0.8 bpp

3 3 2 2 2 1 · ·
3 2 2 2 1 1 · ·
3 2 2 2 1 1 · ·
2 2 2 1 1 1 · ·
2 1 1 1 1 1 · ·
1 1 1 · · · · ·
1 1 · · · · · ·
1 1 · · · · · ·

(e) 0.9 bpp

3 3 3 2 2 1 1 ·
3 3 2 2 1 1 · ·
3 2 2 2 1 1 · ·
2 2 2 1 1 1 · ·
2 2 1 1 1 1 · ·
1 1 1 1 · · · ·
1 1 1 · · · · ·
1 1 · · · · · ·

(f) 1.0 bpp

그림 8. 비트 할당표
Fig. 8 Bits allocation table

표3은 그림8의 비트 할당표를 사용하였을 때 DCT 연산에 소되는 곱셈수이다. 64개의 계수를 모두 계산하는 경우는 방식4의 계산량이 가장 적지만, 화소당 비트율이 0.6비트 이하인 경우는 방식1의 계산량이 방식3 보다 적고, 방식3의 경우는 0.8비트 이하일 때 방식4보다 적다.

표 3. 8×8 DCT계산의 곱셈수
Table 3 Number of multiplications for 8×8 DCT

비 트 할당율	할당된element수 / 전체 element수	방식 1 (제안1)	방식 2	방식 3	방식 4
	100.0%	364	320c	256	144
.5 bpp	23 / 64=35.9%	107		256	
.6 bpp	26 / 64=40.6%	127		256	
.7 bpp	30 / 64=46.9%	151		256	
.8 bpp	33 / 64=51.5%	171		256	
.9 bpp	37 / 64=57.8%	199		256	
1.0 bpp	40 / 64=62.5%	210		256	

※c는 복소수 계산.

예측오차신호의 DCT 부호화에서는 화소당 비트율이 0.5비트이면 충분한 화질을 유지할 수 있다. 제안된 방법은 변환하려는 오차영상에

(a) Ellen

(b) Cronkite

그림 9. 분산계수를 구하는데 사용한 영상
Fig. 9 Images for obtaining variance coefficient

0이 많이 포함되어 있을 때 유리하다. 예측오차의 절대값이 임의의 임계값보다 작은 화소의 값을 0으로 하였을 때, 중요블럭을 크기가 4×4인 네개의 부영역으로 분할하여, 0이 아닌 값이 포함된 부영역의 수에 따라 4개의 부류로 나누어 각 부류의 분포를 조사하였다. 예를들면, 부류1은 네개의 부영역중에 하나의 부영역에만 0이 아닌 값이 포함되고 나머지 3개의 부영역은 모두 0으로 채워진 경우이다.

그림12는 전체 중요블럭을 100%라 할때 각 부류가 차지하는 비율을 그래프로 표현한 것으로서, 복호화된 영상이 안정된 상태에 도달한 10번째 화면 이후의 실험 데이터로 추출한 것이다. 임계값이 증가함에 따라 0을 포함하지 않은

블럭 수가 증가하게 되며, 임계값이 5일 때는 각 부류의 발생률이 거의 같다.

방식1에 Hadamard 변환에 소요되는 정수덧셈은 $2N^2\log_2 N$이므로 N=8인 경우는 384회이지만, 4×4의 부영역이 모두 0인 경우는 $2\times4^2\log_2 4$=64회의 덧셈을 생략할 수 있다. 각 부류의 발생율이 모두 같다고 할 때는 평균 64×(1+2+3)/4=192회의 정수덧셈을 생략할 수 있다.

V. 결 론

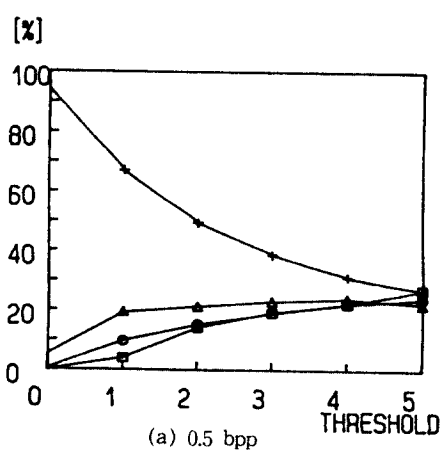
본 연구에서는 이동보상후의 예측오차의 특성과 비트할당표를 이용하여 고속 2차원 DCT알고리즘을 제안하였다.

본 논문에서 고속 2차원 DCT 알고리즘은 변환계수를 모두 구하고자 할 때는 DCT 커넬을 순환적으로 분해하여 계산하는 방식4의 알고리즘보다 계산량이 많지만 비트할당표에서 선택된 계수만을 구하고자 할 때는 계산량이 감소한다. Hadamard 변환을 이용한 방식에서는 변환에서 제외되는 sparse 행렬들의 원소수 만큼 곱셈이 감소된다.

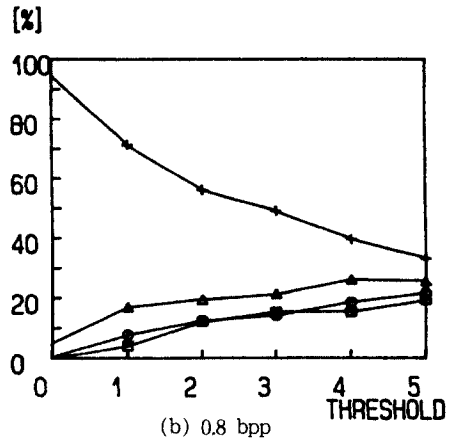
제안된 방법은 비트할당표가 미리 구해져 있는 부호화에서는 사용할 수 있으나, 변환영역에서의 적응 부호화(adaptive coding)에는 사용할 수 없다는 단점이 있다. 그러나, 제안한 방법에서 DCT 계수를 구하는 중간 단계인 Hadamard 변환 계수에 의하여 여러개의 비트할당표 중 하나를 선택하여 부호화하는 적응부호화가 가능하며, 이 단계까지는 정수 덧셈으로 되어 있고, 또한 임계값을 도입하고 예측 오차블럭에 값이 0인 화소가 많이 존재하는 점을 이용하면 정수 계산량이 약 25% 정도 감소된다는 장점이 있다.

또한 본 알고리즘에서 마지막 단계에 곱해지는 상수행렬은 여러개의 cos( )항과 sin( )항의 곱으로 구성되어 있으므로 이들 중에는 무시할 수 있을 정도로 작은 값들이 존재하며, 이들을 0으로 취급하여 알고리즘을 적용하면 계산량을 더욱 줄일 수 있다.

본 논문에서 제안한 고속 DCT 알고리즘은 정지영상의 변환 부호화에도 사용할 수 있으며, 블럭크기가 8×8인 경우 낮은 주파수 영역의 6×6보다 적은 수의 변환계수를 부호화할 때 방식4의 알고리즘보다 계산량이 적어진다. 따라



(a) 0.5 bpp



(b) 0.8 bpp

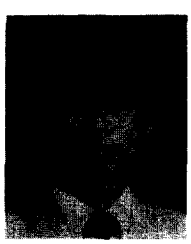
+class 4 ⊙class 2
▲class 3 ▣class 1

그림 12. 중요블럭에 0이 아닌 값이 포함된 부영역의 점유율
Fig. 12 The percentage of the number of subregions containing nonzero pixels in the significant blocks

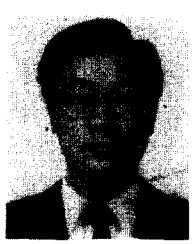
서, 변환 계수의 특정 영역을 부호화 하는 지역 부호화(zonal coding)에 적용할 경우 유리하다.

참고문헌

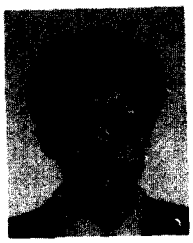
1. R. J. Clark, "Relation between the Karhunen-Loeve and cosine transform," IEEE PROC., Vol. 128, No. 6, Nov. 1981.
2. N. Ahmed, T. Natarajan and K. R. Rao, "Discrete cosine transform," IEEE Trans. on Comput., Vol. C-23, No. 1, pp.90-93, Jan. 1974.
3. D. Hein and N. Ahmed, "On a real-time Walsh-Hadamard / Cosine transform image processor," IEEE Trans. on Electormag. Compat, Vol. EMC-20, No. 3, pp.453-457, Aug. 1978.
4. M. J. Narashma and A. M. Perterson, "On the comptation of discrete cosine transform," IEEE Trans. Commun., Vol. COM-26, No.10, pp. 966-968, Oct. 1 978.
5. B. D. Tseng and W. C. Miler, "On computing the discrete cosine tranform," IEEE Trans. on Comput., Vol. C-27, No. 6, pp. 934-936, June. 1978.
6. W. H. Chen, C. H. Smith and S. C. Fralick, "A fast computational algorithm for the discrete cosine transform," IEEE Trans. on Commun., Vol COM-25, No. 9, pp. 1004-1008, Sept. 1977.
7. F. A. Kamangar and K.R. Rao, "Fast algorithms for the 2-D discret cosine transform," IEEE Trans. on Comput., Vol. C-31, No. 9, Sept. 1982.
8. B. G. Lee, "A new algorithm to compute the discrete cosine trasfom," IEEE Trans. on ASSP., Vol. ASSP-32, No. 6, pp. 1243-1245, Sept. 1977.
9. M. A. Hague, "A two-dimensional fast cosine transform," IEEE Trans. on ASSP., Vol. ASSP-33, No. 6, pp. 1532-1539, Dec. 1985.
10. H. S. Hou, "A fast recursive algorithm for computing the discrete cosine transform," IEEE Trans. on ASSP., Vol. ASSP-35, No. 6, pp. 1445-1461, Oct. 1987.
11. R. J. Clarke, *Transform Coding of Image*, Acadmic Press Inc., 1985.
12. J. Markoul, "A fast consine transform in one and two dimensions," IEEE Trans. on ASSP., Vol. ASSP-28, No. 1, pp. 27-34, Feb. 1980.
13. S. Kappagantula and K. R. Rao, "Motion compensated predictive coding," IEEE Trans. on Commun., Vol COM-33, No. 9, Sep. 1985.
14. N. S. Jayant, Peter Noll, Prentice-Hall, INC., 198 4.



全 重 南(Joong Nam JEON) 正會員
1959年 2 月19日生
1981年 2 月：延世大學校 電子工學科 卒業
1985年 8 月：延世大學校 大學院 電子工學科 卒業(工學碩士)
1990年 2 月：延世大學校 大學院 電子工學科 卒業(工學博士)



崔 源 昊(Won Ho CHOI) 正會員
1956年 2 月 9 日生
1978年 2 月：延世大學校 電子工學科 卒業
1980年 2 月：延世大 大學院 電子工學科 卒業(工學碩士)
1979年12月～1984年12月：第一精密(株) 研究開發室
1984年12月～1986年 2 月：三星HP(株) R&D Project Manager
1988年 2 月：延世大 大學院 電子工學科 博士課程 修了
1986年 3 月～現在：嘉山大學校 電子및電算機工學科 助教授.



崔 成 男(Sung Nam CHOI) 正會員
1965年 8月25日生
1987年 2月：亞洲大學校 電子工學科 卒業
1989年 2月：延世大學校 大學院 電子工學科 卒業(工學碩士)
1989年 3月～現在：延世大學校 大學院 電子工學科 碩士課程

朴 圭 泰(Kyu Tae PARK) 正會員
1933年 6月11日生
1953年～1957年：延世大學校電氣工學科 卒業
1957年～1964年：延世大學校 大學院 電子工學科 卒業(工學碩士)
1962年～1964年：英國 LONDON UNIVERSITY, MSC(工學碩士)
1967年～1969年：英國SOUTHAMPTON UNIVERSITY, Ph. D.(工學博士)
1970年～現在：延世大學校 教授