무손실 음향부호화를 위한 정수 DCT실현기법

(An Integer DCT Algorithm for Lossless Audio Coding)

신재호, 박세형 · 동국대학교 전자공학과
Jaeho Shin, Se Hyoung Park · Dept. of Electronics Engineering., Dongguk University

Abstract

Lifting scheme based integer transforms provides very useful properties on the multimedia coding. An integer transform outputs the integer form when the input has integer value. This doesn't produce quantization errors on coding, so integer transforms are adequate to lossless coding. In this paper, we present an integer DCT algorithm which is able to transform audio signal with longer length. Also the proposed method can be easily implemented recursively even though input is long time. We present the method to overcome the poor approximation which is produced by recursive lifting step. And we have applied the proposed integer DCT to lossless audio coding.

Keywords

Integer DCT, lifting, lossless audio coding

Ⅰ. 서 론

변환(Transform)이란 시간영역 신호를 주파수 영역에서 표현하는 것이므로, 신호해석, 합성, 압축 등 신호처리의 많은 분야에서 사용된다. 특히 DCT(Discrete Cosine Transform), DWT(Discrete Wavelet Transform)는 에너지 집중(Energy Compaction) 특성이 좋아 멀티미디어 신호의 압축에 많이 사용되고 있다. 예로 MPEG과 같은 대부분의 멀티미디어 부호화기는 입력신호를 변환 후 엔트로피 부호화하여 압축을 하는 변환부호화 방식이다.

대부분의 변환은 실수변환이므로 입력이 정수 값을 가질 때에도 변환치는 실수 값을 갖는다. 그러므로 변환부호화에서는 변환된 값을 디지털 형식으로 표현하기 위해 양자화 과정이 필요하게 된다. 이 때 양자화에러가 발생하므로 무손실부호화에 변환부호화 방식을 쓰기 어렵다. 결국 변환부호화를 구현하면, 가역(reversible) 특성이 사라지게 된다. 반면에 정수변환은 입력이 정수 값을 가질 때 변환 값도 정수 값을 가지므로, 정수변환을 이용한 변환부호화기는 실제 구현 시 양자화 과정이 필요 없기 때문에 무손실부호화의 구현이 가능하다. 결국 정수변환에 의한 변환부호화는 실수변환에 의한 부호화와는 달리 가역 특성을 유지할 수 있다.

대부분의 정수변환은 리프팅 기법[1-2]을 이용하여 구현하는데, 가역 변환이 가능하고 빠른 계산을 할 수 있는 장점이 있다. 특히 정수연산이 덧셈과 쉬프트만으로 구현이 가능하므로 빠른 속도로 변환 연산을 수행할 수 있다. Swendels[1]은 리프팅 기법을 이용한 정수 DWT를 제안하였고, 최근에는 리프팅을 이용한 정수 DCT알고리즘이 많이 연구되고 있다. 대

표적인 정수DCT 변환으로는 Y.J. Chen의 IntDCT[3]와 Tran의 BinDCT[4]가 있다. 기존의 정수 DCT변환은 리프팅을 이용한 알고리즘으로서, 리프팅 과정의 부동소수점 곱셈과 반올림(rounding)연산을 $\beta/2^{\alpha}$ 인 근사계수 값을 이용하여 정수연산으로 구현하였다[3-4]. 그러나 기존의 정수DCT 알고리즘은 입력의 개수가 8 또는 16 개인 경우에 적용 가능한 제한적 알고리즘으로, 영상이나 비디오에서 사용하는 8×8, 16×16변환에는 사용하기에 적합하지만 오디오 신호에 적용하기에는 입력의 점수(number of points)가 짧아 적합하지 않다. 또한 직접적으로 IntDCT, binDCT 등의 알고리즘을 확장하여 입력의 점수가 N>>16인 정수 DCT 알고리즘을 구현하는 것은 매우 어렵다. 정수 DCT 알고리즘을 재귀적으로 확장하게 되면, 리프팅 과정의 근사화 에러가 누적되어, 변환의 상관제거 능력이 떨어지게 된다. [5]에서는 정수 DCT를 실수 DCT를 이용하여 구현하였는데, 이는 부동소수점 연산을 하게 되는 단점이 있지만, 근사화 특성이 매우 좋다. 따라서 본 논문에서는 이를 이용하여 재귀적으로 확장할 수 있으면서, 실수변환과 대등한 특성을 갖는 정수 DCT알고리즘을 제안한다. 그리고 이를 무손실 음향부호화에 적용하였다.

먼저 2장에서는 리프팅과 리프팅 행렬을 이용하여 DCT행렬의 인수분해에 과정을 설명하고, 3장에서는 정수 DCT알고리즘을 제안한다. 4장에서는 제안한 정수 DCT알고리즘을 이용하여 무손실 음향 부호화기를 구현하여 제안된 정수 DCT를 평가하고, 마지막으로 결론을 맺는다.

II. 리프팅 기법과 변환

리프팅 기법(lifting scheme)은 Sweldens[1]에 의해서 제안되고 정리되어 FFT, 디지털 필터와 같은 기존의 신호처리 방법에 의존하지 않고 새로운 방법으로 웨이블릿을 구축하여 고속 알고리즘을 연구하는 과정에서 나왔다. 리프팅 기법은 이전의 Donoho의 보간법을 이용한 웨이블릿의 계산과 Lounsberys 등의 다해상도 해석기법을 체계화한 것이다. 그런 다음부터 리프팅을 이용한 웨이블릿 계산과 웨이블릿 변환 연구에 의해 일반적인 신호처리방식으로 자리를 잡게 되었다[6-8].

리프팅을 이용한 웨이블릿 변환의 구현은 기존의 FIR필터 또는 필터뱅크를 이용한 웨이블릿 변환 구현 방법을 분할(split), 예측(predict), 그리고 갱신(update)의 3단계의 리프팅 과정으로 분해하여 구현한다. 이는 격자구조(lattice structure)와 유사한 형태이지만, 기존의 웨이블릿 연산 방식보다 절반 정도의 적은 연산량과 적은 메모리를 요구한다. 이외 리프팅의 장점은 리프팅 연산과정 속에 비선형 연산이 들어가도 리프팅의 역 과정에 의해서 비선형 연산의 에러를 복원할 수 있다는 것이다[1-2]. 이는 리프팅 기반의 정수변환의 기본적인 원리로 가역변환을 가능케 한다[9].

일반적으로 DCT, DFT등과 같은 직교변환(orthogonal transform)은 회전행렬(rotation matrix)로 인수분해가 가능하다[10]. 그리고 회전행렬은 다시 리프팅 행렬로 인수분해가 가능하므로, DCT, DFT와 같은 직교변환 행렬은 리프팅 행렬을 이용하여 구현될 수 있다[2-5, 10].

결국 리프팅을 이용한 직교변환의 구현은 리프팅 행렬을 이용하여 각각의 직교변환행렬을 인수분해하는 문제로 국한된다. 또한 변환행렬의 인수분해방법은 유일하지 않기 때문에, 변환행렬의 구현방식의 용이성, 연산량 등의 목적에 따라 많은 방법이 존재할 수 있다.

2.1 행렬표현과 인수분해

리프팅 행렬은 대각(diagonal) 원소가 모두 1이고, 오직 하나의 비대각(nondiagonal) 원소만이 영이 아닌 값을 갖는 정방행렬 L로 정의한다. 리프팅 행렬의 차수가 N이고, i행, j열의 영이 아닌 비대각 원소 값이 m인 경우 리프팅 행렬은 $L_{i,j}(m)(i\neq j)$같이 표기한다. 리프팅 행렬은 역행렬이 존재하고, 리프팅 행렬의 역

행렬은 영이 아닌 원소의 부호를 바꾸는 것으로 간단히 구할 수 있다. 결국 역행렬 역시 리프팅 행렬의 형태를 갖게 된다.

$$L_{i,j}^{-1}(m)=L_{i,j}(-m) \tag{1}$$

리프팅 행렬은 일종의 삼각행렬로서 상삼각, 하삼각 행렬의 형태를 갖는다. 이를 리프팅에서는 다른 이름으로 부르는데, 전자를 예측단계(lifting step)라 하고, 후자는 갱신단계(dual-lifting step)라고도 한다.

리프팅 기법은 리프팅 행렬을 이용하여 행렬의 인수분해를 의미하는데, 행렬의 인수분해는 유클리드(Euclidean) 알고리즘을 이용하여 구할 수 있다[1-2]. 예로 식 (2)는 회전행렬을 리프팅 행렬로 인수분해한 것이다. <그림 1>은 리프팅 행렬로 인수분해 된 회전행렬의 신호흐름을 보여준다.

$$\begin{bmatrix}\cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha\end{bmatrix}=\begin{bmatrix}1 & \frac{\cos\alpha-1}{\sin\alpha} \\ 0 & 1\end{bmatrix}\begin{bmatrix}1 & 0 \\ \sin\alpha & 1\end{bmatrix}\begin{bmatrix}1 & \frac{\cos\alpha-1}{\sin\alpha} \\ 0 & 1\end{bmatrix} \tag{2}$$

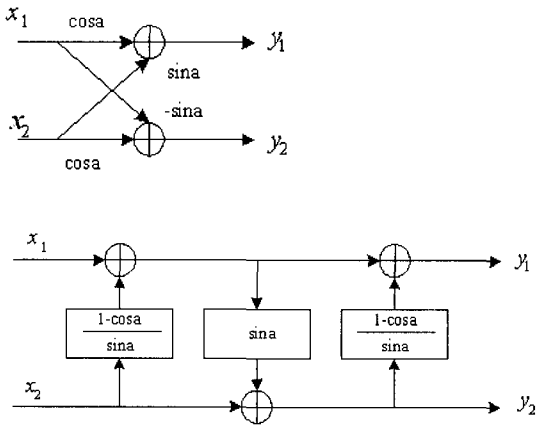

그림 1. 회전행렬과 회전행렬의 리프팅 스텝

Fig. 1. Representation of a Plane Rotation Matrix and its Lifting step

일반적으로 직교변환행렬을 인수분해하게 되면 회전행렬을 포함하게 되고, 이를 다시 리프팅 행렬로 인수분해할 수 있으므로, 결국 직교변환 행렬을 리프팅 행렬로 인수분해할 수 있다. 따라서 식 (2)는 DCT, DFT와 같은 직교행렬의 변환 시 매우 유용하게 이용된다.

다음은 특별한 형태의 행렬을 리프팅 행렬을 이용하여 인수분해하는 예이다.

$$\begin{bmatrix}c & 0 \\ 0 & 1/c\end{bmatrix}=\begin{bmatrix}1 & c-1 \\ 0 & 1\end{bmatrix}\begin{bmatrix}1 & 0 \\ 1 & 1\end{bmatrix}\begin{bmatrix}1 & 1/c-1 \\ 0 & 1\end{bmatrix}\begin{bmatrix}1 & 0 \\ -c & 1\end{bmatrix} \tag{3}$$

식 (3)은 [1-2] 에서와 같이 인수분해의 해가 항상 유일한 것이 아니다. 이러한 인수분해의 다양한 해가 존재함은 리프팅을 이용한 변환행렬의 인수분해 시, 구현방법과 조건에 따라 다양한 형태를 갖을 수 있음을 말한다. 이와 유사하게 블록 스케일링 행렬 즉 식 (3)의 좌변에서 대각 원소 c가 역행렬이 존재하는 $n\times n$행렬 T이면,

$$\begin{bmatrix}T & 0 \\ 0 & T^{-1}\end{bmatrix}=\begin{bmatrix}-I_n & 0 \\ T^{-1} & I_n\end{bmatrix}\begin{bmatrix}I_n & -T \\ 0 & I_n\end{bmatrix}\begin{bmatrix}0 & I_n \\ I_n & T^{-1}\end{bmatrix} \tag{4}$$

인 $2n\times 2n$ 의 블록(block) 리프팅 행렬로 인수분해 된다[5]. 또한 블록 리프팅 행렬 즉 대각 블록 원소는 단위행렬이고, 비대각원소의 위치에 $n\times n$의 행렬 T가 존재하는 행렬의 경우로, 이러한 블록 리프팅 행렬의 역행렬도 일반적인 리프팅 행렬과 유사한 결과를 갖는다.

2.2 비선형 연산 리프팅 행렬과 근사화

리프팅을 이용한 정수변환은 비선형 연산자가 포함된 리프팅 행렬의 근사화 과정을 통해서 이루어진다. 일반적으로 비선형 연산(또는 양자화)에 의해 발생된 에러는 복원이 안되지만, 비선형 연산자를 포함한 리프팅 행렬은 이러한 에러를 복원할 수 있다. 비선형 연산 에러의 복원은 같은 비선형 연산자를 포함한 리프팅 기법의 역 과정으로 가능하고, 리프팅 기법의 역과정은 리프팅 행렬의 역행렬을 통해 가능하다. 결국 리프팅 기법의 역과정을 통한 비선형 연산 에러의 복원은 정수변환 알고리즘의 기본 원리이다[1-2, 7].

먼저 비선형 연산자 $\lfloor\cdot\rfloor$를 포함한 행렬 $L_{i,j}(\lfloor m\rfloor)$라 하면, 이때 $L_{i,j}(\lfloor m\rfloor)$의 차수

와 같은 길이의 벡터 $\boldsymbol{x}$와의 곱인 벡터 $\boldsymbol{y}$는

$$\hat{\boldsymbol{y}} = L_{i,j}(\lfloor m \rfloor)\boldsymbol{x} \tag{5}$$

이고, 리프팅 스텝의 결과 $\hat{\boldsymbol{y}}$는

$$\begin{aligned} \hat{y}(i) &= x(i) + \lfloor m x(j) \rfloor \\ \hat{y}(k) &= x(k) \ , k \neq i \end{aligned} \tag{6}$$

와 같이 구할 수 있다. 그리고 이의 역과정은 리프팅 행렬의 역행렬 $L_{i,j}^{-1}(\lfloor m \rfloor)$과의 곱이 된다.

$$\boldsymbol{x} = L_{i,j}^{-1}(\lfloor m \rfloor)\hat{\boldsymbol{y}} \tag{7}$$

식 (7)은

$$\begin{aligned} x(i) &= \hat{y}(i) - \lfloor m \hat{y}(j) \rfloor \\ x(k) &= \hat{y}(k) \ , k \neq i \end{aligned} \tag{8}$$

이고, 결국

$$\begin{aligned} x(i) &= x(i) + \lfloor m x(j) \rfloor - \lfloor m \hat{y}(j) \rfloor \\ &= x(i) \ , \quad (x(j) = \hat{y}(j) \ , j \neq i) \end{aligned} \tag{9}$$

가 되어 비선형 연산 과정이 완벽 복원된다. 이와 같이 비선형 연산자 $\lfloor \cdot \rfloor$를 통해 발생한 에러는 리프팅 역행렬을 통해 완전히 복원할 수 있다.

리프팅 행렬의 구현에 필요한 연산과정은 입력벡터와 리프팅 행렬의 비대각원소와의 곱셈과 덧셈으로 구성된다. 즉 입력벡터 $\boldsymbol{x}$의 i번째 원소와 리프팅 행렬의 비대각 원소 m과의 곱과 j번째 $\boldsymbol{x}$의 원소와의 합의 연산이다. 이때 입력벡터가 정수이고, m이 실수일 경우, 리프팅 행렬은 부동소수점(floating point) 연산이 필요하게 된다. 또한 비선형 연산자를 포함한 리프팅 행렬 $L_{i,j}(\lfloor m \rfloor)$의 연산도 부동소수점 연산이 필요하다.

그러나 행렬 $L_{i,j}(\lfloor m \rfloor)$의 $\lfloor m \rfloor$값을 어떻게 설정하는가에 따라 매우 다른 결과를 얻을 수 있다. 입력벡터 $x(i)$가 항상 정수이면 $\lfloor m \rfloor$와의 곱 $\lfloor m x(i) \rfloor$는 실수와 정수의 곱 그리고 라운딩 연산이 필요하게 된다. 그러나 $\lfloor m \rfloor$의 값이 m에 근사한 유리수이면, 부동소수점 연산은 간단한 정수곱(또는 쉬프트 연산과 덧셈)으로도 구현이 가능하다. m의 값이 실수일 경우 리프팅 행렬의 구현은 실수 연산을 필요로 하게 되지만, 이를 피하기 위해 m의 유리수형태의 근사치 $\tilde{m}$ (또는 $RB(m)$)을 다음과 같이 정의하여 정수연산으로 구현할 수 있다.

$$\tilde{m} = RB(m) = \frac{\beta}{\alpha} \ , \ \alpha = 2^{\lambda} \tag{10}$$

이때 α, β, λ는 정수이다[1-9].

III. 정수 DCT의 실현

DCT는 여러 변환 방식 중 상관제거의 최적변환 방식인 KLT(Karhunen-Loe've Transform)와 유사한 성능을 얻을 수 있어 데이터 압축분야에서 많이 사용된다. 이러한 DCT를 고속실현하기 위한 많은 알고리즘이 제안되었다[10][11]. 이들 알고리즘은 현재 JPEG, H.26x, 그리고 MPEG등에서 많이 사용되고 있다. DCT는 그 형태에 따라 I~IV까지 4가지의 형태가 있고, 일반적으로 DCT-II나 DCT-IV가 많이 사용되어지는데 먼저 DCT-II 형태의 변환은 다음의 식 (11)와 같다.

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

$$\alpha(k) = \begin{cases} \frac{1}{\sqrt{2}}, & k = 0 \text{ or } N \\ 1, & otherwise \end{cases} \tag{11}$$

이다. 이때 스케일된 DCT-II의 변환 행렬 C_N^{II}는

$$C_N^{II}=\left[\cos\frac{(2n+1)k\pi}{2N}\right] \quad 0\le k,n\le N-1 \tag{12}$$

로 표현된다. 이와 유사하게 DCT-IV는 다음과 같이 정의되고,

$$X^{(4)}(k)=\sqrt{\frac{2}{N}}\sum_{n=0}^{N-1}x(n)\cos\frac{(2k+1)(2n+1)\pi}{4N} \tag{13}$$

그리고 스케일된 DCT-IV의 변환 행렬 C_N^{IV}는

$$C_N^{IV}=\left[\cos\frac{(2n+1)(2k+1)\pi}{4N}\right] \quad 0\le k,n\le N-1 \tag{14}$$

로 표현된다. 본 논문에서는 식(12), (14)의 스케일된 DCT 행렬에 대하여 인수분해와 리프팅을 적용함으로서 정수 DCT 알고리즘을 제안하였다.

3.1. N점 정수 DCT알고리즘

기존의 리프팅 기반 정수 DCT 고속 알고리즘은 대부분 기존의 실수 DCT 고속 알고리즘을 이용하여 개발되었고, 이들은 입력 신호의 길이 N이 8, 16인 경우로 제한적이어서 영상 신호에는 적합하나 음향 신호의 변환에는 부적합하다. 본 논문에서는 이를 극복하기 위해서 Chen의 인수분해 방식을 이용하였는데, 먼저 식 (12)인 DCT 행렬을 직접계산법을 이용하여 인수분해 하였다. DCT-II 변환행렬 $A_N=C_N^{II}$이라 하면

$$[A_N]=[\ \overline{P}_N]\begin{bmatrix}[A_{N/2}] & 0\\ 0 & [\ \overline{R}_{N/2}]\end{bmatrix}[\ B_N] \tag{15}$$

$[\ \overline{P}_N]$은 permutation 행렬이고, 버터플라이 행렬 $[\ B_N]$은

$$[\ B_N]=\begin{bmatrix}[I_{N/2}] & [\ J_{N/2}]\\ [\ J_{N/2}] & -[I_{N/2}]\end{bmatrix} \tag{16}$$

이다. 단, $[I_M]$는 $M\times M$단위행렬이고, $[J_M]$는 $M\times M$ counter-단위행렬이다. 그리고 $[\overline{R_N}]$는 다음 $[R_N]$행렬의 i행 k열 원소의 순서가 내림차순으로 되어 있는 경우이다. $[R_N]$는

$$[R_N]_{ik}=\cos\left(\frac{(2i+1)(2k+1)\pi}{4N}\right) \tag{17}$$

결국 윗 식 $[R_N]$은 DCT-IV의 변환행렬식이다[10]. 따라서 DCT-II의 직접계산법을 정리하면,

$$C_N^{II}=\begin{bmatrix}C_{N/2}^{II} & 0\\ 0 & C_{N/2}^{IV}\end{bmatrix}\begin{bmatrix}I_{N/2} & J_{N/2}\\ J_{N/2} & -I_{N/2}\end{bmatrix} \tag{18}$$

가 되고, 다시 $C_{N/2}^{II}$는

$$C_{N/2}^{II}=\begin{bmatrix}C_{N/4}^{II} & 0\\ 0 & C_{N/4}^{IV}\end{bmatrix}\begin{bmatrix}I_{N/4} & J_{N/4}\\ J_{N/4} & -I_{N/4}\end{bmatrix} \tag{19}$$

로 인수분해 된다[4,10].

이를 요약하면, N점의 C_N^{II}은 $C_{N/2}^{II}$과 $C_{N/2}^{IV}$로 인수분해 되어 진다. 그리고 $C_{N/2}^{II}$은 재귀적으로 C_L^{II}과 C_L^{IV}로 인수분해 되어 진다. 위의 알고리즘은 공통적으로 단위 행렬, permutation 행렬, 버터플라이 행렬과 같은 기본적인 행렬을 갖게 된다. 결국 $C_2^{II}, C_2^{IV}, C_4^{IV}\cdots C_{N/2}^{IV}$는 모두 회전 행렬이므로 모두 리프팅 행렬로 인수분해 될 수 있고, 리프팅 행렬의 nonzero 비대각 원소를 근사화시켜 정수변환 알고리즘을 구현하게 된다[4,10,11,13].

binDCT는 N=8 or 16인 경우의 예로 들 수 있다. <그림 2>는 직접인수분해를 이용한 Tran[4]

의 binDCT 알고리즘의 구조를 보여준다.

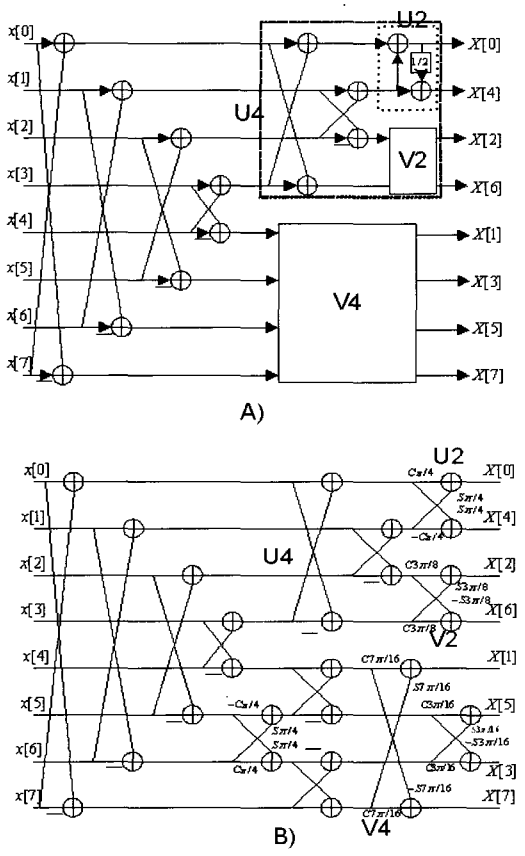

그림 2. A) Chen의 인수분해의 알고리즘 블록
B) 8포인트 Chen의 DCT 인수분해 구조
Fig 2. A) Algorithm block of Chen's factorization
B) Structure of 8 point Chen's factorization of DCT

식(18), (19)은 재귀적 형태를 잃지 않으면서 근사화 성능을 높이기 위해서 본 논문에서는 Geiger[5]가 제안한 정수 MDCT(or 정수 DCT-IV) 알고리즘을 적용하였다. 식 (14)에서 정의한 DCT-IV 행렬은, C_N^{IV}는 $N/2$길이의 $C_{N/2}^{IV}$로 인수분해 될 수 있다[5]. 즉

$$C_N^{IV}=T\begin{bmatrix} C_{N/2}^{IV} & 0 \\ 0 & C_{N/2}^{IV} \end{bmatrix} MQP \qquad (20)$$

로 분해되어 진다[5,10,14].

이때 행렬 T는 $N\times N$ 회전행렬로서

$$\begin{bmatrix} T_{k,k} & T_{k,N-1-k} \\ T_{N-1-k,k} & T_{N-1-k,N-1-k} \end{bmatrix} = \begin{bmatrix} \cos(\frac{2k+1}{4N}\pi) & -\sin(\frac{2k+1}{4N}\pi) \\ -\sin(\frac{2k+1}{4N}\pi) & -\cos(\frac{2k+1}{4N}\pi) \end{bmatrix}$$

$$k=0,\cdots,N/2-1$$

$$T_{k,l}=0 \qquad otherwise \qquad (21)$$

로 정의되고, $N\times N$ 행렬 M은

$$M=\frac{1}{\sqrt{2}}\begin{bmatrix} I_{N/2} & I_{N/2} \\ -I_{N/2} & I_{N/2} \end{bmatrix} \qquad (22)$$

이다. P는

$$P_{4k,4k}=P_{4k+1,4k+1}=P_{4k+2,4k+3}=P_{4k+3,4k+2}=1$$

$$k=0,\cdots,N/4-1$$

$$P_{k,l}=0 \qquad otherwise \qquad (23)$$

이고, $N\times N$ Q는

$$Q_{k,2k}=Q_{k,2k}=1 \quad k=0,\cdots,N/2-1$$

$$Q_{k,l}=0 \qquad otherwise \qquad (24)$$

이다. 그림 3은 길이 N=8인 경우 C_8^{IV}를 보여준다.

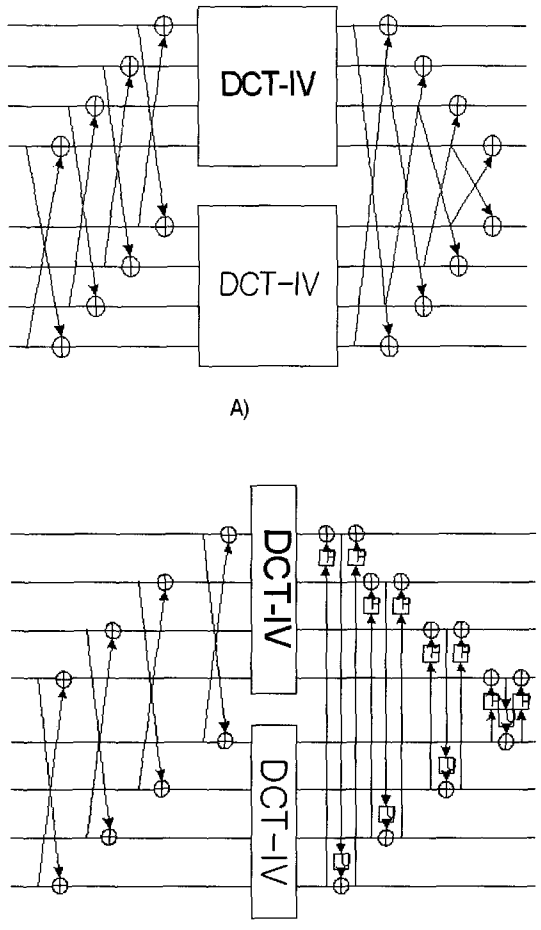

그림 3. A) 정수 DCT(식 21)의 블록도
B) 그림 A)의 리프팅 표현
Fig. 3. A) Block diagram of integer DCT eq.(21)
B) Representaion of A) with lifting step

여기서 C_N^{IV}는 $N/2$길이의 2개의 $C_{N/2}^{IV}$와 리프팅 행렬로 구현이 가능하므로 이는 C_L^{IV}의 재귀적 방법으로 구현될 수 있다. 즉, 식 (12)는 다음과 같이 재귀적으로 구현이 가능하다. 그림 4는 N=8인 경우에 대한 예이다.

$$C_N^{II}=\begin{bmatrix} C_{N/2}^{II} & 0 \\ 0 & C_{N/2}^{IV} \end{bmatrix}\begin{bmatrix} I_{N/2} & J_{N/2} \\ J_{N/2} & -I_{N/2} \end{bmatrix} \quad (25)$$

$$C_{N/2}^{II}=\begin{bmatrix} C_{N/4}^{II} & 0 \\ 0 & C_{N/4}^{IV} \end{bmatrix}\begin{bmatrix} I_{N/4} & J_{N/4} \\ J_{N/4} & -I_{N/4} \end{bmatrix} \quad (26)$$

$$C_{N/2}^{IV}=L\begin{bmatrix} C_{N/4}^{IV} & 0 \\ 0 & C_{N/4}^{IV} \end{bmatrix}MQP \quad (27)$$

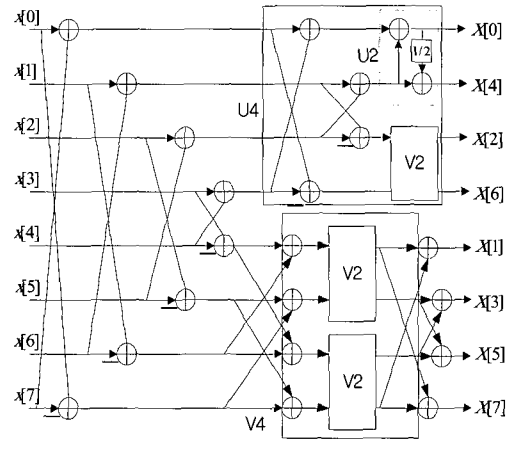

그림 4. 직접 인수분해를 이용한 재귀적 알고리즘
Fig. 4. Recursive algorithm of using Direct's factorization

그러나 식 (25)-(27)을 이용한 정수 DCT의 재귀적 방법은 근사화 성능의 문제점이 있다. 정수 DCT는 실수 DCT의 근사화된 결과를 얻으므로, 근사화된 리프팅 행렬 $L_{i,j}(RB(m))$의 α, β 값 선택에 따라 근사화 성능의 차이가 있다. 게다가 알고리즘이 재귀적 형태를 갖게 되면 근사화오차가 누적되어 실수 DCT의 특징이 사라지게 된다.

따라서 본 논문에서는 이를 극복하기 위해서 실제 구현시 $C_{N/2}^{IV}$ 의 내부는 실수연산을 하고, 회전행렬 T 를 리프팅 행렬과 비선형 연산 과정을 통함으로서 정수 DCT를 구현하였다. 이러한 방법은 내부적으로 실수연산을 사용하여 $C_{N/2}^{IV}$ 내부를 재귀적으로 구성하지 않아 근사화 성능을 높일 수 있고, 기존의 고속 DCT-IV알고리즘을 이용함으로 쉽게 정수 변환을 구현할 수 있다.

Ⅳ. 무손실 음향부호화에 응용

무손실 변환부호화 방식에서 정수 변환은 매우 유용하다. 대다수 변환부호화의 입력신호는 유한 비트로서 8, 16, 24비트의 해상도를 가진다. FFT와 DCT 등은 이론적으로는 가역 변환이지만 디지털 신호의 유한비트 표현으로 인해 가역변환이 불가능해져 무손실 부호화에 부적합하다.

반면에 리프팅 기반 정수 변환은 가역 변환이면서 출력이 정수 값을 가지기 때문에 무손실 부호화에 적합하다. 특히 LTAC[15]과 같이 실수변환을 이용한 무손실 부호화 구조와 비교해서 실수변환의 역변환 과정이 필요없게 되며, 복호화기와 부호화기의 복잡도가 동일하여 연산량을 크게 줄일 수 있다.

그러나 리프팅 기반 정수변환은 원래의 실수변환을 근사화한 결과이므로 실수변환에 비해 상관제거 능력이 떨어지게 된다. 이는 정수변환의 구현방식, 근사화 정도에 따라 성능이 다르게 나타난다. 결국 무손실 부호화에서는 정수 변환이 실수 변환에 가장 근사화된 결과를 가질 때 가장 좋은 성능을 나타낼 수 있다. 본 논문에서는 제안한 리프팅 기반 고속 정수 DCT를 이용하여 무손실 음향부호화기를 구현하고 이를 이용하여 음향신호를 압축부호화 하였다. <그림 5>의 A)는 제안한 방식, B)는 실수변환을 이용하는 LTAC의 무손실 부호화 과정이다.

무손실 음향부호화기는 프레이밍, 상관제거기, 엔트로피 부호화기, 그리고 비트 스트림 패킹의 4과정으로 구성하였다. 그리고 엔트로피 부호화기로는 Golomb-Rice 부호화기를 이용하였다. Golomb-Rice부호화기는 프레임 별로 독립적으로 부호화하기 때문에 이를 위해 정수 DCT된 신호 $c[k]$ 를 16~64개의 블록별로 나누어 각각 독립적으로 엔트로피 부호화한다[12-16]. 표 1과 같이 정수DCT의 입력길이와 엔트로피 부호화 블록의 길이를 세 가지(IDL1, IDL2, IDL3) 경우로 나누어 각각 무손실 음향부호화를 시행하였다.

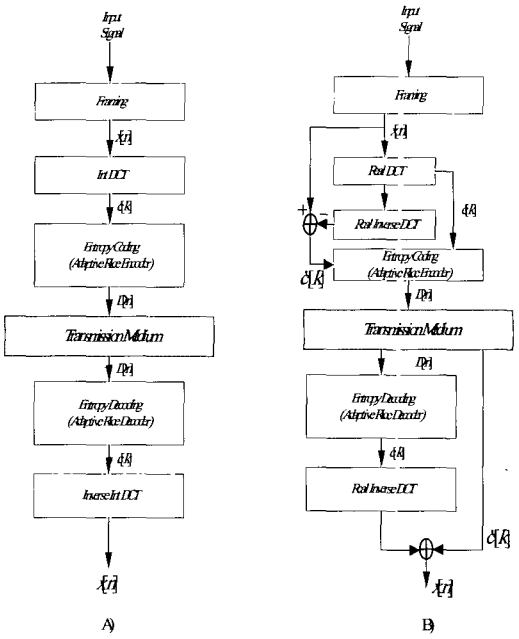

그림 5. A) 제안한 정수 DCT 무손실 음향 부호화와 B) 실수 DCT를 이용한 무손실 부호화 (LTAC)

Fig. 5. A) Block of proposed lossless audio coder using integer DCT
B) Block of lossless audio coder using real transform(LTAC)

표 1. IDL1형, IDL2형과 IDL3형의 파라미터
Table 1. Parameters of IDL1, IDL2, and IDL3 types

Type	Frame Size	Segment Size
IDL1	4096	32
IDL2	4096	64
IDL3	2048	16

따라서 제안한 방식의 무손실 부호하기를 실험하기 위해서 MAX 9(Sony Ltd.)앨범에서 추출한 데이터를 사용하였는데, 실험에 사용된 데이터는 44.1kHz의 샘플링 주파수를 갖고, 16비트의 해상도, 그리고 모노 채널에 대해서 실시하였다. 데이터의 압축률은 식 (28)를 이용하여 계산하였다.

$$CompressionRatio(\%) = 100 \times \frac{Compresesed\ file\ Size\ in\ byte}{Original\ file\ Size\ in\ byte} \quad (28)$$

<표 2>는 시뮬레이션 결과로 기존의 실수 DCT에 대한 정수 DCT의 근사화 정도와 성능을 확인하기 위하여 본 논문에서는 [15]에서 소개한 LTAC과 비교하였다.

표 2. 제안된 알고리즘의 압축률[%]
Table 2. Compression ratio[%] of proposed algorithms

Track No.	LTAC [%]	IDL1 [%]	IDL2 [%]	IDL3 [%]
Track1	77.21	77.47	77.19	78.71
Track2	66.28	68.99	68.61	69.61
Track3	76.85	77.39	76.97	78.16
Track4	71.74	72.60	72.19	74.08
Track5	57.31	61.11	60.83	62.55
Track6	70.84	71.63	71.29	72.57
Track7	64.29	68.19	67.86	69.32
Track8	71.65	73.75	73.34	74.46
Track9	64.44	67.99	67.65	69.71
Track10	68.15	69.21	68.78	71.06
Track11	69.98	72.60	72.22	73.87
Track12	67.71	69.94	69.50	71.29
Track13	59.73	62.04	61.68	63.78
Track14	59.92	62.07	61.74	63.83
Track15	62.49	66.68	66.40	68.18
Track16	65.18	69.03	68.83	70.38
Track17	61.43	63.92	63.75	65.95
Average	66.31	69.09	68.75	70.44

<표 2>에서 보는 바와 같이 무손실 부호화의 성능에 큰 차이가 없음을 보였다. 따라서 기존의 실수 DCT를 이용하는 LTAC보다 최소 0.2% 최대 4%정도의 압축률의 차이를 보이지 않음을 알 수 있었다. 그러나 두 방법의 연산량은 같은 고속 실수 DCT알고리즘을 사용한다는 가정아래 제안한 방식의 부호화기는 LTAC에 비해 절반의 DCT 연산과 N번의 덧셈을 줄일 수 있고 복호화기에서는 N번의 덧셈을 줄일 수 있어, 하드웨어 감축 및 실시간 처리에 적합함을 보여준다.

<표 3>은 제안한 방식과 실수 DCT를 사용한 LTAC과의 연산량을 비교하였다. 여기서 O_{DCT}는 입력이 N인 DCT연산이다.

표 3. 프레임당 LTAC과 제안한 방식의 연산량 비교
Table 3. Comparison of computation complexity per 1 frame between LTAC and proposed method

		LTAC	제안한 방식
압축률		평균 66.31%	평균68.75%
연산량	부호화기	$2\times O_{DCT}$연산 N번의 덧셈	O_{DCT}연산
	복호화기	O_{DCT}연산 N번의 덧셈	O_{DCT}연산

V. 결 론

리프팅 기법은 이제 하나의 기본 신호처리 이론으로서, 기존의 방식과 달리 여러 유용한 결과를 얻을 수 있다. 실제 리프팅의 구현과정은 단일 프로세서와 적은 메모리의 사용을 가능케 하고, 비선형 연산을 포함한 리프팅 행렬의 역변환을 통해서 정수변환을 구현할 수 있다. 본 논문에서는 리프팅을 이용한 Long point 정수DCT를 제안하였는데, 기존의 고속 알고리즘을 이용하므로 구현이 쉽다. 또한 기존의 정수 DCT의 단점을 보완하여, 입력이 긴 경우에도 가능한 알고리즘을 제안하였다. 그리고 이때 발생되는 근사화 문제를 해결하기 위한 방법도 제시하였다. 마지막으로 제안한 정수 DCT알고리즘을 적용한 무손실 음향 부호화기를 구현하여 제안한 정수 DCT방식이 실제 무손실 음향 부호화에 사용되어도 좋은 성능을 나타냄을 보였다.

[참 고 문 헌]

[1] W. Sweldens, "The lifting scheme: A custom-design construction of biorthogonal wavelet," *Appl. Comput. Harmon. Anal.*, vol. 3, no. 20, pp.186-200, 1996.

[2] I. Daubechies and W. Sweldens. "Factoring wavelets transforms into lifting steps," *Technical Report, Bell Labatatories,* Lucent Technologies, 1996.

[3] Y. Chen, S. Oraintara, and T. Nguyen, " Integer Discrete Consine Transform (IntDCT)," *IEEE Trans. on Signal Processing,* Feb. 2000.

[4] T. Tran, " The BinDCT: Fast Multiplierless Approximation of the DCT," *IEEE Signal Processing Letters,* vol. 7, no. 6. Jun. 2000.

[5] R. Geiger, Y. Yokotani, and G. Schuller, "Improved Integer transforms for lossless audio coding," *11st AES Convention Preprint,* 5471, 2001.

[6] P. Vaidyanathan, *Multirate Systems and Filter Banks,* Prentice-Hall, Englewood Cliffs, N.J., 1993.

[7] M. Vetterli and J. Kovacevic. *Wavelets and Subband Coding,* Prentice Hall, Englewood Cliffs, N.J., 1995.

[8] 유훈, "리프팅 기반 신호 종속적 웨이블릿 변환과 영상 처리에의 응용", *한양대학교 대학원 박사논문,* Dec. 2002.

[9] W. Sweldens, "The lifting scheme: A construction of second generation wavelets", *SIAM J. Math. Anal.*, vol. 29, no. 2. pp.511-546, 1997.

[10] K. Rao and P. Yip, *Discrete Cosine Transform : Advantages and Applications,* New York : Academic, 1990.

[11] Y. Zeng, and L. Cheng, " Intger DCTs and Fast Algorithms," *IEEE Trans. on Signal Processing,* vol. 49, no. 11, Nov. 2001.

[12] 박세형, 신재호, "Hybrid 무손실 음향 부호화기의 설계", *전자공학회 논문지,* 제41권, SP-6호, Nov. 2004.

[13] S. Park, S. Park, and J. Shin, "A lossless and lossy audio compression using prediction model and wavelet transform," *Proc. of ITC-CSCC'02,* Aug. 2002.

[14] L. Cheng, H. Xu, and Y. Luo, " Integer discrete cosine transform and its fast algorithm", *Electronics Letters,* vol. 37. no. 1. Jan. 2001

[15] M. Purat, T. Liebchen, and P. Noll, "Losselss transform coding of audio signals," *Proc. 101nd AES Conv.*, Munich, Germany, 1997.

[16] T. Robinson, "SHORTEN: Simple lossless and near-lossless waveform compression," *Cambridge Univ. Eng. Dept., Cambridge, UK, Tech. Rep.* 156, 1994.

Biography

신 재 호

1979년 서울대학교 전자공학과 졸업
1982년 서울대학교 대학원 전자공학과(공학석사)
1987년 서울대학교 대학원 전자공학과(공학박사)
1988년-현재 동국대학교 전자공학과 교수
디지털미디어기술연구소장

<주관심분야> DSP, Audio Coding, Watermarking
<이메일> jhshin@dongguk.edu

박 세 형

1997년 동국대학교 전자공학과 졸업
1999년 동국대학교 대학원 전자공학과(공학석사)
2005년 서울대학교 대학원 전자공학과(공학박사)
현재 삼성전자(주) 책임연구원

<주관심분야> Audio Coding, Watermarking
<이메일> sehyoung.4u.park@samsung.com