MPEG-2에서 H.264로의 Transcoding 과정에서 DCT 계수를 H.264 변환 계수로 변환하는 효율적인 알고리듬

준회원 김 용 재*, 정회원 이 창 우**

An Efficient Algorithm for the Conversion of DCT Coefficients to H.264 Transform Coefficients in MPEG-2 to H.264 Transcoding

Yong-jae Kim* *Associate Member*, Chang-woo Lee** *Regular Member*

요 약

가장 최근에 표준화된 H.264/AVC 동영상 압축 부호화 방식은 기존의 방법인 MPEG-2에 비해 우수한 압축 효율을 보이지만, 아직 많은 응용분야에서 MPEG-2가 적용되고 있기 때문에 MPEG-2에서 H.264/AVC로의 변환 기법 개발이 필요하다. 본 논문에서는 이러한 변환을 위해 가장 첫 번째 단계로 볼 수 있는 DCT 계수에서 H.264 변환 계수로의 변환을 위한 효율적인 방법을 제안한다. 이러한 변환은 영상의 압축을 위한 변환기술로 MPEG-2에서는 8×8 discrete cosine transform (DCT)를 사용하고 있고, H.264/AVC 표준안에서는 4×4 integer transform을 사용하고 있기 때문에 필수적이다. 제안하는 알고리듬에서는 부호화 성능을 유지하면서 효율적으로 계산량을 감소시킬 수 있는 방법을 제시하고 계산량 분석과 모의실험을 통해서 이를 입증한다.

Key Words : Video Transcoding, H.264/AVC, MPEG-2, Integer Transform, Discrete Cosine Transform

ABSTRACT

The H.264/AVC video coding standard provides higher coding efficiency compared to the conventional MPEG-2 standard. Since a lot of videos have been encoded using MPEG-2, the format conversion from MPEG-2 to H.264 is essential. In this paper, we propose an efficient method for the conversion of DCT coefficients to H.264/AVC transform coefficients. This conversion is essential, since 8×8 DCT and 4×4 integer transform are used in MPEG-2 and H.264/AVC, respectively. The mathematical analysis and computer simulation show that the computational complexity of the proposed algorithm is reduced compared to the conventional algorithm, while the loss caused by the conversion is negligible.

Ⅰ. 서 론

H.264/AVC 영상 부호화 방식은 기존 영상 부호화 방식인 MPEG-1,2 또는 H.263에 비해 높은 압축 효율을 보인다. 그러나 많은 동영상 정보가 MPEG-2 방식을 이용하여 부호화되어 있기 때문에 MPEG-2 표준으로 부호화된 정보를 H.264 표준으로 변환하는 트랜스코딩 기법이 많이 연구되고 있다[1].

MPEG-2에서는 동영상 정보를 압축하기 위해 공간 영역에서 8×8 블록 단위의 DCT(discrete cosine

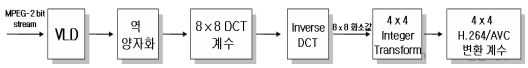

그림 1. 공간 영역에서 DCT 계수에서 H.264 정수 변환 계수로 변환

Fig. 1. The conversion of DCT coefficients to H.264/AVC transform coefficients in spatial domain

* 가톨릭대학교 대학원 컴퓨터공학과 신호처리 연구실, ** 가톨릭대학교 정보통신전자공학부 (changwoo@catholic.ac.kr)
논문번호 : KICS2007-04-169, 접수일자 : 2007년 4월 10일, 최종논문접수일자 : 2007년 7월 26일

transform)가 이용되는 반면에, H.264/AVC에서는 4×4 블록 단위의 정수 변환이 이용되는 차이점을 보인다[2]. 두 표준간의 호환을 위해 DCT 계수를 공간 영역으로 변환하고 이를 다시 정수 변환 영역으로 변환하는 방법을 생각할 수 있다. 이에 대한 블록도를 그림 1에 도시하였는데 공간 영역에서의 변환을 위해 먼저 IDCT 과정이 필요함을 알 수 있다. IDCT와 정수 변환 연산을 차례로 수행해야 하는 계산상의 복잡도를 피하기 위해 Jun Xin 등은 DCT 영역에서 직접 정수 변환 계수로 변환하는 방법을 제안하였다[3]. 또한, Bo Shen은 8-point DCT 행렬의 재구성을 통해 계산상의 복잡도를 줄이는 방법을 제안하였고, 이를 개선한 방법이 Park 등에 의해 제안되었다[4,5]. 최근의 Gao Chen 등이 제안하는 방법에서는 DCT 계수의 특성을 이용하여 DCT 블록들을 분류하여 계산량을 줄이는 방법을 제안하였으며[6], Chen Chen 등은 이미지 크기를 줄이면서 DCT 계수를 정수 변환 계수로 변환하는 방법[7]을 제안하였다. 이러한 알고리듬들은 기존의 방법에 비해 DCT 영역에서 정수 변환 영역으로 직접 변환이 이루어짐에 따라 IDCT 과정이 필요가 없으므로 계산량을 줄일 수 있고, IDCT 과정의 오차를 줄일 수 있는 장점을 가진다.

본 논문에서는 정수 변환은 4×4 블록 단위로 변환이 이루어지므로 8×8 DCT 계수에서 변환하는 것보다는 4×4 DCT 계수 블록에서 변환시키는 것이 더 효율적이라는 점을 착안하여 8×8 DCT 계수를 4×4 DCT 계수로 변환한 후 이를 다시 4×4 정수 변환 계수로 변환하는 알고리듬을 제안한다. 또한, DCT 계수에서 정수 변환 계수로의 전환에 필요한 계산량을 최대한 줄일 수 있는 효율적인 방법도 제시한다. 그리고 계산량 분석과 모의실험을 통해서 제안하는 알고리듬이 기존의 알고리듬과 비교하여 성능을 저하시키지 않으면서 계산량을 줄일 수 있음을 입증한다.

본 논문의 구성은 다음과 같다. 본 논문의 2장에서는 DCT 계수를 정수 변환 계수로 전환하기 위한 기존의 알고리듬들을 설명하고, 3장에서는 DCT 계수를 정수 변환 계수로 전환하기 위해서 제안하는 알고리듬의 방법을 구체적으로 기술하며, 제안하는 알고리듬의 계산량을 줄이는 보다 효율적인 방법에 대해 설명한다. 4장에서는 여러 테스트 영상에 적용한 결과를 영상 성능과 계산량 측면에서 분석하고, 마지막으로 5장에서 결론을 맺는다.

Ⅱ. 기존의 알고리듬

그림 1에 제시한 IDCT와 정수 변환 연산을 차례로 수행해야 하는 기존 알고리듬의 계산상의 복잡도를 줄이기 위해 Jun Xin 등이 제안한 DCT 영역에서 정수 변환 계수로 직접 변환하는 방법을 설명하면 다음과 같다[3]. 그림 2에 Jun Xin 등에 의해 제안된 DCT 계수를 정수 변환 계수로 직접 변환하는 방법을 도시하였다. 그림에서 볼수 있는 것과 같이 MPEG-2 비트열에서 얻은 매크로 블록 단위의 8x8 DCT 계수 블록 4개를 Jun Xin 등에 의해 제안된 변환 행렬을 이용하여 매크로 블록 단위의 4x4 정수 변환 계수 블록 16개로 직접 변환 시킬 수 있고, 4번째 블록의 C는 제안된 변환 행렬을 의미하며 그 값은 [3]의 연구에 제시되어 있다. 이와 같이 직접 변환을 통해 IDCT 연산을 생략함에 따라 계산량을 감소시킬 수 있을 뿐만 아니라, IDCT 과정에서 발생하는 오차를 없애는 장점을 가진다.

또한, Bo Shen은 직접 변환 행렬을 이용하는 방법과는 다르게 8x8 DCT 행렬을 재구성하여 DCT 계수에서 정수 변환 계수로의 전환에 필요한 계산량을 줄이는 방법을 제안하였다[4]. 즉, DCT 계수를 정수 변환 계수로 변환시키는 위의 식 (1)에서 DCT 행렬 T_8을 Merhav 등에 의해 제안된 방법[8]을 이용하여 재구성하였다. 식 (1)에서 T_8은 8×8 DCT 행렬을, T_8^t은 T_8의 전치 행렬을 나타내고, $H_{4\times4}$과 $H_{4\times4}^t$은 정수 변환 행렬과 그 전치 행렬을 각각 의미하며, $Y_{8\times8}$은 변환된 정수 변환 계수를 의미한다. Bo Shen이 제안한 방법을 토대로 계산량을 더욱 향상시키는 개선된 방법이 Park 등에 의해 제안되었다[5]. 이 방법은 식 (1)에서 $H_{4\times4}$을 대각 행렬 $D_{4\times4}$와 수정된 4×4 DCT 행렬 $T_{4\times4}^m$의 곱으로 근사화하여 나타내고 대각 행렬을 위한 연산은 양자화과정에서 수행함으로써 계산량을 더욱 줄이는 알고리듬이다. 여기서 $T_{4\times4}^m$은 정확한 4×4

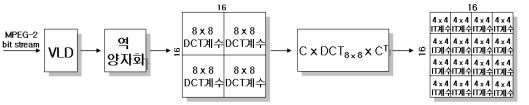

그림 2. DCT 영역에서 DCT 계수를 정수 변환 계수로 변환
Fig. 2. The conversion of DCT coefficients to H.264/AVC transform coefficients in DCT domain

$$Y_{8\times 8}=\begin{pmatrix} H_{4\times 4} & O_{4\times 4} \\ O_{4\times 4} & H_{4\times 4} \end{pmatrix} \times (T_8^t \cdot DCT_{8\times 8} \cdot T_8) \times \begin{pmatrix} H_{4\times 4}^t & O_{4\times 4} \\ O_{4\times 4} & H_{4\times 4}^t \end{pmatrix} \quad (1)$$

DCT 행렬이 아니라 수정된 행렬이며, 정수 변환 행렬 $H_{4\times 4}$, 대각행렬 $D_{4\times 4}$,그리고 수정된 4×4 DCT 행렬 $T_{4\times 4}^m$의 값은 식 (2)~(4)와 같다.

$$D_{4\times 4}=\begin{pmatrix} 2 & 0 & 0 & 0 \\ 0 & 3.1623 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 3.1623 \end{pmatrix} \quad (2)$$

$$T_{4\times 4}^m=\begin{pmatrix} 0.5 & 0.5 & 0.5 & 0.5 \\ 0.6325 & 0.3162 & -0.3162 & -0.6325 \\ 0.5 & -0.5 & -0.5 & 0.5 \\ 0.3162 & -0.6325 & 0.6325 & -0.3162 \end{pmatrix} \quad (3)$$

$$H_{4\times 4}=\begin{pmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{pmatrix} \quad (4)$$

마지막으로 가장 최근의 Gao Chen 등이 제안한 방법은 앞에서 언급한 것과 같이 Jun Xin 등이 제안한 방법에서 DCT 계수 블록을 분류하는 방법인데 최선의 경우에 계산량이 큰 감소를 얻게 되는 알고리듬을 보여주고 있다[6]. 그러나 이 방법은 일반적으로 적용되기에는 단점을 가지고 있고 오히려 최악의 경우로 DCT 계수가 분류될 시 계산량은 Jun Xin 등이 보여주는 것과 같게 나타난다.

Ⅲ. DCT 계수를 정수 변환 계수로 변환하는 효율적인 알고리듬

3.1 제안하는 알고리듬

2장에서 설명한 것과 같이 DCT 계수를 정수 변환 계수로 전환하기 위해 DCT 영역에서 정수 변환 영역으로 직접 변환하는 방법은 DCT 영역에서 공간 영역으로 변환하고 다시 정수 변환 영역으로 변환하는 방법에 비해 IDCT 연산이 필요하지 않기 때문에 더 효율적이다. 따라서 본 절에서는 DCT 영역에서 정수 변환 영역으로 직접 변환하는 효율적인 알고리듬을 제안한다. 또한, MPEG-2 비트열에서 얻어지는 매크로 블록 단위의 8×8 DCT 블록을 4×4 정수 변환 블록으로 바로 전환하는 기존의 방법에 비해서 제안하는 알고리듬은 먼저 8×8 DCT 블록을 4×4 DCT 블록으로 변환하는 과정을 추가하여 4×4 DCT 블록에서 4×4 정수 변환 블록으로 변환이 이루어지게 함으로써 효율성을 극대화하였다. 제안하는 알고리듬의 전체적인 블록도를 그림 3에 도시하였다. 제안하는 방법에서는 그림 2에 도시한 기존의 방법과는 다르게 서로 다른 크기의 DCT 블록간의 관계를 이용하여[9], 8×8 DCT 블록을 4×4 DCT 계수 블록 4개로 변환하는 과정이 추가되었다. DCT 계수에서 정수 변환 계수로의 전환에 있어 입력을 8×8 DCT 계수라 하고 출력을 4×4 정수 변환 계수로 가정하면, 제안하는 알고리듬을 구현하기 위한 변환 행렬을 식 (5)와 같이 표현할 수 있다.

$$Y_{8\times 8}=\begin{pmatrix} H_{4\times 4} & O_{4\times 4} \\ O_{4\times 4} & H_{4\times 4} \end{pmatrix}\begin{pmatrix} T_{4\times 4}^t & O_{4\times 4} \\ O_{4\times 4} & T_{4\times 4}^t \end{pmatrix} S_{(2,4)}^t \cdot DCT_{8\times 8} S_{(2,4)} \begin{pmatrix} T_{4\times 4} & O_{4\times 4} \\ O_{4\times 4} & T_{4\times 4} \end{pmatrix}\begin{pmatrix} H_{4\times 4}^t & O_{4\times 4} \\ O_{4\times 4} & H_{4\times 4}^t \end{pmatrix} \quad (5)$$

앞에서 설명한 것과 같이, 식 (5)에서 $H_{4\times 4}$는 H.264/AVC에서 압축 변환 기술로 이용되는 4×4 블록 단위의 정수 변환 행렬을 의미하고, $H_{4\times 4}^t$는 $H_{4\times 4}$의 전치 행렬을 의미하며, $T_{4\times 4}$와 $T_{4\times 4}^t$는 4×4 DCT 발생 행렬과 그 전치 행렬을, $DCT_{8\times 8}$은 MPEG-2 디코딩 후 입력으로 들어오는 8×8 블록 단위의 DCT 계수 값을, 마지막으로 $Y_{8\times 8}$은 결과 행렬 값으로 정수 변환 계수 값을 각각 의미한다. 그리고 $S_{(2,4)}$은 8×8 DCT 계수 블록을 4×4 DCT 계수 블록 4개로 변환하는 행렬을 의미하고, $S_{(2,4)}^t$은 $S_{(2,4)}$의 전치 행렬을 나타낸다. $O_{4\times 4}$는 4×4 단위의 영행렬이다. $S_{(2,4)}$행렬은 8×8 행렬이며, Mukherjee 등에 의해 제안된 변환 행렬을 이용하였다[10].

식 (5)를 풀어 설명하면, 먼저 $S_{(2,4)}^t DCT_{8\times 8} S_{(2,4)}$의 행렬간의 연산을 통하여 8×8 DCT 계수 블록을 4×4 DCT 계수 블록 4개로 변환한다. 행렬 $S_{(2,4)}$의 값은 (6)과 같고, $1/\sqrt{2}$을 위한 연산은 양자화

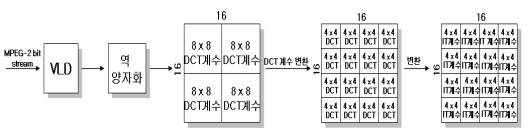

그림 3. DCT 계수를 정수 변환 계수로 변환하기 위하여 제안하는 알고리듬
Fig. 3. The proposed algorithm for conversion of DCT coefficients to H.264/AVC transform coefficients

과정에 포함시킴으로써 계산량을 줄일 수 있다.

$$S_{(2,4)} = \frac{1}{\sqrt{2}} \times \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0.9061 & 0.4158 & -0.0747 & 0.0229 \\ 0 & 1 & 0 & 0 \\ -0.3182 & 0.7911 & 0.5132 & -0.0976 \\ 0 & 0 & 1 & 0 \\ 0.2125 & -0.3524 & 0.7682 & 0.4904 \\ 0 & 0 & 0 & 1 \\ -0.1802 & 0.2777 & -0.3753 & 0.8658 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ -0.9061 & 0.4158 & 0.0747 & 0.0229 \\ 0 & -1 & 0 & 0 \\ 0.3182 & 0.7911 & -0.5132 & -0.0976 \\ 0 & 0 & 1 & 0 \\ -0.2125 & -0.3524 & -0.7682 & 0.4904 \\ 0 & 0 & 0 & -1 \\ 0.1802 & 0.2777 & 0.3753 & 0.8658 \end{pmatrix} \tag{6}$$

이와 같은 과정을 통해 4×4 DCT 계수 블록 4개를 얻게 되고, 4-point IDCT 과정과 정수 변환을 수행함으로써 우리가 얻고자하는 4×4 블록 정수 변환 계수를 얻는다. 4-point DCT 행렬 $T_{4\times4}$는 (7)과 같다.

$$T_{4\times4} = \begin{pmatrix} 0.5 & 0.5 & 0.5 & 0.5 \\ 0.6533 & 0.2706 & -0.2706 & -0.6533 \\ 0.5 & -0.5 & -0.5 & 0.5 \\ 0.2706 & -0.6533 & 0.6533 & -0.2706 \end{pmatrix} \tag{7}$$

기존의 연구에서는, 8×8 DCT 행렬과 정수 변환 행렬을 곱하여 하나의 변환 행렬로 구현함으로써 8×8 DCT 계수를 직접 정수 변환 계수로 변환하는 방법을 제안하였다. 그러나 본 논문에서는 행렬 $H_{4\times4}$과 $T^t_{4\times4}$의 곱에서 양자화 계수로 포함시킬 수 있는 대각 행렬을 구하고 행렬 $S_{(2,4)}$을 수정하여 계산량을 줄이는 방법을 제안한다. 그런데, 제안하는 방법에서 식 (5)를 이용하여 DCT 영역을 정수 변환 영역으로 변환한다면 IDCT 과정을 수행하게 되므로 오히려 비효율적이다. 따라서 행렬의 곱을 효율적으로 구현하여 IDCT 과정을 없애고, 계산량을 최대한 줄이기 위해 $S_{(2,4)}$을 다음의 과정을 통해 새롭게 분해하여 정의할 수 있다.

먼저, 식 (5)에서 행렬 $H_{4\times4}$와 $T^t_{4\times4}$의 곱의 결과는 식 (8)과 같다.

$$H_{4\times4} \cdot T^t_{4\times4} = \begin{pmatrix} 2 & 0 & 0 & 0 \\ 0 & 3.1544 & 0 & -0.2242 \\ 0 & 0 & 2 & 0 \\ 0 & 0.2242 & 0 & 3.1544 \end{pmatrix} \tag{8}$$

식 (8)에서 얻은 결과 값은 다음과 같이 두 행렬 값의 곱으로 다시 표현할 수 있다.

$$H_{4\times4} \cdot T^t_{4\times4} = \begin{pmatrix} 2 & 0 & 0 & 0 \\ 0 & 3.1544 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 3.1544 \end{pmatrix} \times \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & -0.0711 \\ 0 & 0 & 1 & 0 \\ 0 & 0.0711 & 0 & 1 \end{pmatrix} \tag{9}$$

식 (9)에서 왼쪽의 4×4 행렬은 대각행렬로서 이를 위한 연산은 양자화 과정에 포함시킬 수 있으므로, 식 (9)에서 오른쪽 4×4 행렬을 $R_{4\times4}$라 정의하고 이 행렬과 행렬 $S_{(2,4)}$와의 관계만 고려하면 된다. 식 (6)의 행렬 $S_{(2,4)}$에서 $1/\sqrt{2}$을 위한 연산은 양자화 과정에 포함시킬 수 있고, 나머지 행렬의 1열부터 4열까지의 8×4 행렬을 S_L로 5열부터 8열까지의 8×4 행렬을 S_R로 다시 표현하면, 8×8 행렬 $S_{(2,4)}$은 [S_L S_R]와 같다. 여기서 S_L과 S_R의 대칭성을 이용하여 S_L은 A+B로 S_R은 A-B로 나타낼 수 있고, A와 B는 다음과 같이 정의할 수 있다.

$$A = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0.4158 & 0 & 0.0229 \\ 0 & 0 & 0 & 0 \\ 0 & 0.7911 & 0 & -0.0976 \\ 0 & 0 & 1 & 0 \\ 0 & -0.3524 & 0 & 0.4904 \\ 0 & 0 & 0 & 0 \\ 0 & 0.2777 & 0 & 0.8658 \end{pmatrix} \tag{10}$$

$$B = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0.9061 & 0 & -0.0747 & 0 \\ 0 & 1 & 0 & 0 \\ -0.3182 & 0 & 0.5132 & 0 \\ 0 & 0 & 0 & 0 \\ 0.2125 & 0 & 0.7682 & 0 \\ 0 & 0 & 0 & 1 \\ -0.1802 & 0 & -0.3753 & 0 \end{pmatrix} \tag{11}$$

MPEG-2 비트열에서 얻은 8×8 DCT 계수의 1열을 Z[1]~Z[8]이라 할 때, 식 (5)에서 1차원 계산만을 고려한다면, 다음과 같이 쓸 수 있다.

$$H \cdot T^t_4 \cdot S^t_{(2,4)} \cdot \begin{bmatrix} Z[1] \\ \sim \\ Z[8] \end{bmatrix} \tag{12}$$

다시 이 식은 식 (9)~(11)에 의해 다음과 같이 나타낼 수 있다.

$$R_{4\times4} \cdot \begin{bmatrix} A^t + B^t \\ A^t - B^t \end{bmatrix} \cdot \begin{bmatrix} Z[1] \\ \sim \\ Z[8] \end{bmatrix} \tag{13}$$

A^t와 B^t 행렬은 4×8 행렬로 A와 B의 전치행렬이다. 여기서 1차원 계산은 결국 열벡터 Z[1]~Z[8]과의 연산이므로 $R_{4\times4}$와 A^t의 곱, 그리고 $R_{4\times4}$와 B^t의 곱의 결과는 다음과 같다.

$$R_{4\times4}\cdot A^t_{4\times8}=\begin{pmatrix}1&0&0&0&0&0&0&0\\0&0.4142&0&0.7980&0&-0.3873&0&0.2161\\0&0&0&0&1&0&0&0\\0&0.0525&0&-0.0414&0&0.4653&0&0.8855\end{pmatrix} \quad (14)$$

$$R_{4\times4}\cdot B^t_{4\times8}=\begin{pmatrix}0&0.9061&0&-0.3182\\0&0&1&0\\0&-0.0747&0&0.5132\\0&0&0.0711&0\end{pmatrix}\begin{pmatrix}0&0.2125&0&-0.1802\\0&0&-0.0711&0\\0&0.7682&0&-0.3753\\0&0&1&0\end{pmatrix} \quad (15)$$

따라서 $R_{4\times4}\cdot A^t\cdot\begin{bmatrix}Z[1]\\\sim\\Z[8]\end{bmatrix}$의 계산을 위해서는 8회의 곱셈과 6회의 덧셈이 필요하며, $R_{4\times4}\cdot B^t\cdot\begin{bmatrix}Z[1]\\\sim\\Z[8]\end{bmatrix}$의 계산량을 위해서는 곱셈 수 10과 덧셈 수 8을 요구한다. 그러므로 DCT 계수에서 정수 변환 계수로의 변환에 필요한 1차원 계산량은 18회의 곱셈과 14회의 덧셈이 필요하며, 이에 대한 2차원의 경우, 288의 곱셈 수와 224의 덧셈 수로 총 512의 계산량이 요구된다.

3.2 제안하는 알고리듬의 개선

3.1절에서 DCT 계수에서 정수 변환 계수로의 1차원상의 변환을 위해 18회의 곱셈과, 14회의 덧셈이 필요함을 알 수 있었는데, 본 절에서는 곱셈 수를 더 줄일 수 있는 개선된 방법을 제안한다. 우선 식 (14)에서 마지막 열의 4번째 행의 원소인 0.8855을 두 번째 행의 원소 0.2161의 배수인 0.8644로 근사화하면 이는 쉬프트 연산으로 대치할 수 있다. 마찬가지로, 식 (15)에서 마지막 열의 세 번째 행의 원소인 -0.3753을 첫 번째 행의 원소 -0.1802의 두 배인 -0.3604로 근사화하면, DCT 계수와의 곱 연산 시 쉬프트 연산으로 대치시킴으로써 곱셈 량을 줄일 수 있다. 또한, 식 (15)에서 7번째 열의 두 번째 행의 원소 값인 -0.0711을 0으로 근사시킴으로써 곱셈 수와 덧셈 수를 줄일 수 있다. 이와 같은 두 가지 방법을 통해 식 (14)~(15)에서 얻은 값을 다시 수정하면, 다음과 같다.

$$\mathrm{mod}ified(R_{4\times4}\cdot A^t_{4\times8})=\begin{pmatrix}1&0&0&0&0&0&0&0\\0&0.4142&0&0.7980&0&-0.3873&0&0.2161\\0&0&0&0&1&0&0&0\\0&0.0525&0&-0.0414&0&0.4653&0&0.8644\end{pmatrix} \quad (16)$$

$$\mathrm{mod}ified(R_{4\times4}\cdot B^t_{4\times8})=\begin{pmatrix}0&0.9061&0&-0.3182\\0&0&1&0\\0&-0.0747&0&0.5132\\0&0&0.0711&0\end{pmatrix}\begin{pmatrix}0&0.2125&0&-0.1802\\0&0&0&0\\0&0.7682&0&-0.3604\\0&0&1&0\end{pmatrix} \quad (17)$$

따라서 식 (16), (17)로 수정된 행렬로 1차원 DCT 계수와의 계산량을 구하면 곱셈은 15회가 필요하고 덧셈은 13회가 필요하며, 곱셈 수가 줄어드는 대신 3회의 쉬프트 연산이 추가된다. 2차원의 경우, 곱셈 수는 240, 덧셈 수는 208, 쉬프트 연산 수는 48이 된다. 이에 대한 기존 방법과의 비교 분석은 4장에서 다룬다. 마지막으로, 위의 수정된 행렬에 의해 식 (6)에서의 행렬 $S_{(2,4)}$의 원소들이 바뀌게 되는데 이는 다음과 같이 구한다. $R_{4\times4}$와 A^t의 곱, 그리고 $R_{4\times4}$과 B^t의 곱에서 $R_{4\times4}$의 역행렬을 구하고, 이 역행렬을 통해 A^t와 B^t의 값을 구한 후, 두 행렬의 합과 차에 의해 수정된 행렬 $S_{(2,4)}$를 구할 수 있으며 그 결과는 다음과 같다.

$$modified\ S_{(2,4)}=\begin{pmatrix}1&0&0&0\\0.9061&0.4158&-0.0747&0.0229\\0&1&0&0\\-0.3182&0.7911&0.5132&-0.0976\\0&0&1&0\\0.2125&-0.3524&0.7682&0.4927\\0&0&0&1\\-0.1802&0.2761&-0.3604&0.8447\end{pmatrix}\begin{pmatrix}1&0&0&0\\-0.9061&0.4158&0.0747&0.0229\\0&-1&0&0\\0.3182&0.7911&-0.5132&-0.0976\\0&0&1&0\\-0.2125&-0.3524&-0.7682&0.4927\\0&0&0&-1\\0.1802&0.2761&0.3604&0.8447\end{pmatrix} \quad (18)$$

이와 같이 식 (6), (14)~(15)을 식 (16)~(18)로 수정이 가능한 이유는 MPEG-2 디코딩에서 얻은 8×8 DCT 계수의 고주파 부분의 값이 0에 가까운 값이므로 이 부분과의 연산은 근사화하여도 전체 성능에 있어 큰 영향을 미치지 않기 때문이다. 그러나 근사화를 통해 H.264 표준 복호 시 화질 열화의 가능성이 존재하게 되는데 근사화의 결과는 수정된

행렬 $S_{(2,4)}$에만 영향을 주고 정수 변환 행렬에는 영향을 주지 않으므로 정수 변환 행렬을 근사화한 기존의 방법에 비해 효율적이며 이는 다음 장의 화질 성능 분석에서 확인할 수 있다.

Ⅳ. 성능 분석

본 장에서는 제안하는 알고리듬의 성능을 분석하기 위해서 표준 CIF 동영상에 대한 간단한 모의실험 결과를 제시하고 그림 4와 같은 방법의 실험 환경에서 각 알고리듬에 대한 성능을 비교 분석한다. 또한, 기존 방법과의 계산량 비교 분석을 통해 제안하는 알고리듬의 효율성을 확인한다. 먼저 기존의 방법들과 본 논문에서 제안하는 알고리듬의 성능을 다음과 같이 분석할 수 있다. CIF 테스트 영상을 각 알고리듬을 이용하여 DCT 영역에서 정수 변환 영역으로 변환한 후 원 영상과 정수 변환 영역을 공간 영역으로 변환한 영상의 비교를 통해 얻은 PSNR 값을 표 1에 제시하였는데 실험 결과는 각 영상에서 첫 번째 프레임만을 선택하여 실험한 결과 값이다. Park 등에 의해 제안된 방법과 제안하는 알고리듬을 개선한 방법에 의한 PSNR 결과 값은 공간 영역에서의 DCT 계수를 정수 변환 계수로 변환하는 방법에 의한 PSNR 값과 비교하면 많은 차이를 보이지 않는다. 또한, 3.1절에서 제안하는 알고리듬에 의해 얻은 PSNR 값에 비해 3.2절에서 제안하는 계산량을 줄이는 개선된 방법으로 얻어낸 PSNR 값이 더 좋은 결과를 보였다. 이는 3.2절에서 설명한 것과 같이 DCT 계수의 고주파 부분의 값이 0에 근사한 성질을 이용하여 (18)에서 제시한 수정된 행렬 $S_{(2,4)}$의 값이 식 (6)의 Mukherjee 등에 의해 제안된 변환 행렬 값보다[10], 8×8 DCT 계수값에서 4×4 DCT 계수값으로 더 정확하게 변환시킬 수 있다는 사실을 뒷받침한다.

또한, 양자화에 의한 영향을 분석하기 위해서 그림 4에 도시한 것과 같이 8×8 블록을 DCT, 양자화(Q1) 그리고 역 양자화(IQ1)를 한 후, 각 알고리

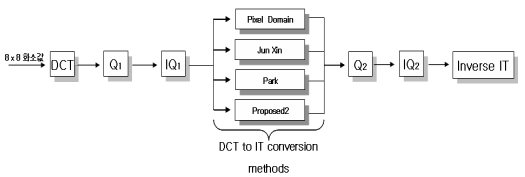

그림 4. 성능 분석을 위한 실험 환경
Fig. 4. Simulation settings

표 1. 각 알고리듬에 대한 영상과 원 영상과의 차이를 통한 성능 비교(proposed 1: 3.1절에서 제안하는 방법, proposed 2: 3.2절에서 제안하는 방법)
Table 1. performance comparison for difference of original image and image of each algorithm (proposed 1: proposed method in 3.1, proposed 2: proposed method in 3.2)

Algorithm	PSNR			
	Carphone	Claire	Forman	Miss_am
공간 영역에서의 변환	51.48	52.49	51.44	51.97
Jun Xin[3]	49.92	49.04	49.68	49.29
Park[5]	51.12	51.18	51.04	51.16
Proposed 1	49.93	49.00	49.62	49.30
Proposed 2	50.77	50.03	50.53	50.63

듬을 이용하여 DCT 영역에서 정수 변환 영역으로 변환하고 다시 정수 변환 영역에서 양자화(Q2), 역 양자화(IQ2)를 실행한 후 마지막으로 역 정수 변환을 하여 각 알고리듬의 성능을 비교하였다. 또한, 8×8 블록을 DCT 변환 후 양자화단계에서 양자화 크기(Q1)를 8로 하여 실험하였으며, x축의 QP는 H.264에서의 양자화 크기(Q2)에 대응되는 양자화 파라미터 값을 의미한다. 실험 결과는 그림 5와 6에 도시하였고 각각은 352×288 크기의 Miss_am 영상과 Carphone 영상을 이용하여 각 테스트 영상의 30프레임에 대한 평균의 값을 나타낸 것이다. 그림에서 Y축 값은 영상의 공간 영역에서의 변환 방법을 기준으로 하여 DCT 영역에서 변환하는 각 알고리듬들과의 PSNR 차이값을 나타낸다. 따라서 PSNR 차이값이 높다는 것은 더 좋은 영상 성능을 보이는 것을 의미하며, 공간 영역에서의 변환보다 DCT 영역에서의 변환 방법이 더 우수한 성능을 보

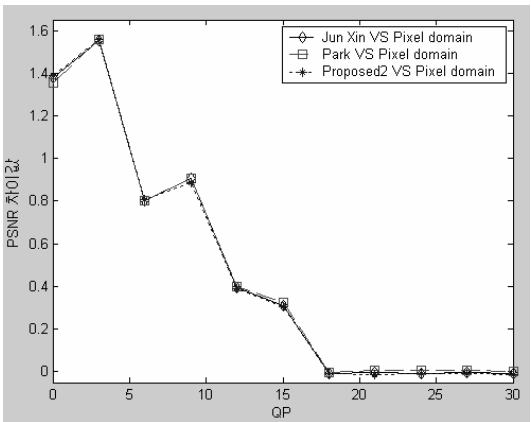

그림 5. Miss_am 영상에 대한 각 알고리듬과 공간 영역에서의 변환 방법과의 PSNR 차이값
Fig. 5.Relative PSNR difference each algorithm vs. spatial domain conversion for Miss_am CIF sequence

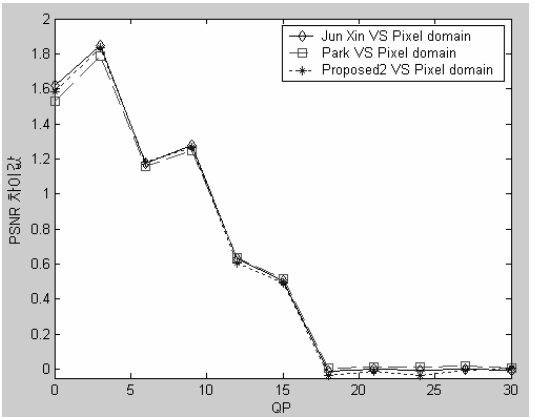

그림 6. Carphone 영상에 대한 각 알고리듬과 공간 영역에서의 변환 방법과의 PSNR 차이값
Fig. 6. Relative PSNR difference each algorithm vs. spatial domain conversion for Carphone CIF sequence

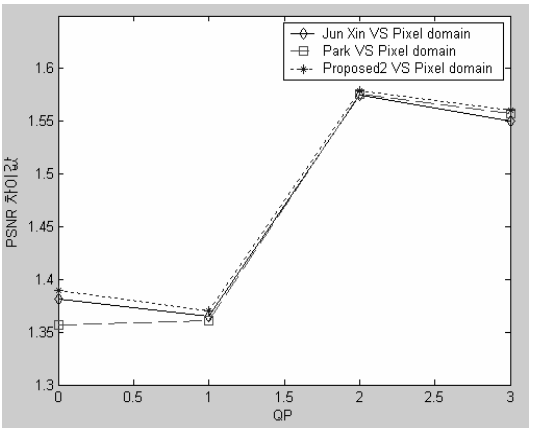

그림 7. 그림 5에서 QP가 작을 때에 대한 확대 그림
Fig. 7. A magnification picture of fig 5 as QP is low

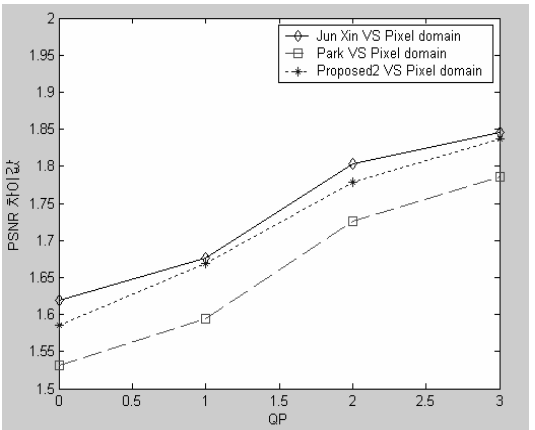

그림 8. 그림 6에서 QP가 작을 때에 대한 확대 그림
Fig. 8. A magnification picture of fig 6 as QP is low

표 2. DCT 계수에서 정수 변화 계수로의 변환을 위한 각 알고리듬의 계산량 비교
Table 2. Computation comparison of each algorithm for conversion of DCT coefficients to integer transform coefficients.

Algorithm	곱셈 수	덧셈 수	쉬프트 수	총 연산수	계산상의 단계
공간영역에서의 변환	256	672	64	992	6
Jun Xin[3]	352	352	0	704	2
Park[5]	80	400	64	544	4
Gao Chen[6]	352(132)	352(132)	64(0)	704(264)	2
Proposed 1	288	224	0	512	3
Proposed 2	240	208	48	496	3

임을 입증한다. 그러나 QP값이 커질수록 양자화 크기가 커지기 때문에 각 알고리듬을 통해 얻은 PSNR의 차이가 거의 나타나지 않는다. 따라서 QP값이 작을 때에 대한 PSNR 차이값을 확인할 필요가 있다. 그림 7에서는 QP값에 따라 모든 알고리듬의 PSNR 차이가 크게 나타나지 않지만 그림 8에서는 뚜렷한 차이를 볼 수 있으며, 두 실험 모두에서 제안하는 알고리듬이 Park 등에 의해 제안된 방법보다 영상 성능에 있어 더 우수함을 볼 수 있다.

마지막으로 DCT 영역에서 DCT 계수를 정수 변환 계수로 변환하는 각 알고리듬의 계산량을 다음과 같이 비교 분석할 수 있다. 먼저, DCT 계수를 공간 영역으로 변환한 후 정수 변환 계수로 변환하는 방법과 DCT 영역에서의 정수 변환 영역으로 직접 변환하는 방법에 대한 계산량을 표 2에 제시하였다. 표 2는 각 알고리듬에 대해 곱셈 수, 덧셈 수, 쉬프트 수, 전체 계산량 그리고 계산상의 단계를 각각 나누어 표현하였고, 그에 대한 결과 값은 2차원 연산에 대한 것이다. DCT 영역에서의 변환이 IDCT 연산을 이용한 공간 영역에서의 변환과 유사한 성능을 보이면서 계산량을 상당히 줄일 수 있다는 것을 확인할 수 있다. 제안하는 알고리듬과 Jun Xin 등이 제안한 방법을 비교할 때, 복잡도가 상당히 개선된 것을 표를 통해 확인 할 수 있다. Jun Xin 등이 제안한 변환 행렬을 본 논문에서 제안하는 알고리듬에서 많은 영향을 미치는 행렬 $S_{(2,4)}$와 비교할 때, 두 행렬 모두 대칭성을 가지고 있지만 계산에 있어 용이한 0과 1의 원소 값을 행렬 $S_{(2,4)}$이 더 많이 가지고 있다는 점에서 계산량 감소의 주요한 원인이 된다. 또한, DCT 계수의 고주파 부분과의 곱 연산이 요구되는 부분을 쉬프트 연산이 될 수 있도록 수정한 3.2절에서 제안한 알

고리듬의 복잡도를 보면, 3.1절에서 제안한 알고리듬에 비해 쉬프트 연산이 추가되지만, 곱셈 수와 전체 계산량을 줄이는 효과를 얻었다. 그러나 제안하는 알고리듬의 개선된 방법과 Park 등에 의해 제안된 방법을 비교해 볼 때, 본 논문에서 제안한 개선된 알고리듬이 전체 계산량에 있어 향상되었음을 확인할 수 있고 계산 단계에 있어서도 절감되는 효과를 얻었지만, 곱셈 수에서는 아직 많은 차이를 보이고 있다. 곱셈 계산량 측면에서 제안하는 알고리듬이 높은 복잡도를 보이지만, Park 등에 의해 제안된 알고리듬은 8×8 DCT 행렬의 재구성과 정수 변환 행렬의 재구성을 통해 여러 개의 단계로 구성되기 때문에, 행렬의 재구성과 변환에 필요한 행렬값들을 저장할 메모리 공간의 필요와 행렬값들을 이용하기 위한 많은 메모리 접근 횟수의 필요는 하드웨어 구현 시 단점으로 작용할 수 있다. 이는 MPEG-2에서 H.264로의 트랜스코딩에서 인트라 모드 결정과 예측이 결합되어 트랜스 코더를 이룰 때 인트라 모드 결정과 예측에서도 상당한 메모리를 필요로 하게 되는데 DCT 계수를 정수 변환 계수로 변환하는데에 계산 단계를 많이 필요로 한다면 전체적으로 메모리가 많이 필요하다는 단점이 있다. 그리고 정수 변환 행렬을 근사화한 부분에서는 원 영상을 H.264 부호기로 부호화한 결과와 다를 수 있고 H.264 표준으로 복호 시 여러 프레임에 걸쳐 복호화가 진행될 경우 정수 변환 행렬의 근사화로 화질 열화를 발생시킬 수 있는 단점을 가진다. 이러한 가능성은 정수 변환 행렬을 근사화한 것이기 때문에 본 논문에서 제안하는 방법에 비해 발생 가능성이 더 높게 나타날 수 있다. 또한, 표 2에서와 같이 Gao Chen 등이 제안한 방법[6]의 경우 DCT 계수 블록에 따라 그 특성이 달라지는데 최선의 경우에서는 DCT 계수의 고주파 부분을 0으로 생각하여 곱셈 수와 덧셈 수가 표 2의 괄호 안에 제시한 것과 같이 모두 132회로 총 계산량은 264회로 상당히 줄어들지만, 최악의 경우에는 Jun Xin 등이 보여준 계산량과 같은 결과를 얻게 되므로 일반적으로 적용하기에는 단점을 가지게 된다. 그리고 이 방법에서도 DCT 계수의 고주파 부분을 0으로 근사화한 것으로 화질 열화의 가능성이 본 논문에서 제안하는 알고리듬에 비해 높을 수 있다.

Ⅴ. 결 론

본 논문에서는 DCT 계수를 정수 변환 계수로 직접 변환하는 효율적인 방법을 제안하였다. 제안하는 알고리듬은 매크로 블록 단위의 8×8 DCT 계수에서 직접 4×4 단위의 정수 변환 계수로 전환하지 않고, 8×8 DCT 블록을 정수 변환과 같은 블록 크기인 4×4 DCT 블록 4개로 변환한 후 정수 변환 계수로 변환하는 방법이다. 또한, DCT 계수의 고주파 부분이 0에 근사하다는 성질을 이용하여 보다 효율적으로 개선된 알고리듬을 제안하였다. 개선된 알고리듬은 기존의 방법과 비교하여 전체 성능을 비슷하게 유지하는 동시에 복잡도를 크게 줄일 수 있는 방법이다. CIF 영상을 이용한 모의 실험과 복잡도 분석을 통해서 제안하는 알고리듬의 효율성을 확인하였다. 또한, 제안하는 알고리듬은 8×8 DCT 계수, 4×4 DCT 계수를 모두 사용할 수 있다는 특성 때문에 DCT 계수를 이용하여 인트라 모드를 결정할 시 그 활용성이 높다. 따라서 제안하는 알고리듬은 MPEG-2에서 H.264로의 효율적인 변환에 이용될 수 있다.

참 고 문 헌

〔1〕 A.Vetro, Christopoluos, C, and huifang Sun, "Video transcoding architectures and techniques: an overview", *IEEE Signal Processing Magazine*, pp. 18-29, Mar, 2003.

〔2〕 H.S. Malvar, A. Hallapuro, M. Karczewicz, and L.Kerofsky, "Low-complexity transform and quantization in H.264/AVC", *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 598-603, July, 2003.

〔3〕 J. Xin, A. Verto, and H. Sun, "Converting DCT coefficients to H.264/AVC transform coefficients ", *Technical Report of Motsubishi Electric Research Lab.,* TR 2004-058, June, 2004.

〔4〕 B, Shen, "From 8-tap DCT to 4-tap integer-transform for MPEG to H.264/AVC transcoding", *Proc. 2004 IEEE ICIP,* pp. 115-118, Oct. 2004.

〔5〕 C. Y. Park and N. I. Cho, "A fast algorithm for the conversion of DCT coefficients to H.264 transform coefficients", *IEEE International Conference on Image Processing 2005*, III.664-III.667, Sept. 2005.

〔6〕 G. Chen, S. Lin, and Y. Zhang, "A fast co-

efficients convertion method for the transform domain MPEG-2 to H.264 transcoding", *IEEE International Conference on Digital Telecommunications 2006,* pp.13-17 April 2006.

〔7〕 C. Chen, P-H.Wu, and H.Chen, "MPEG-2 to H.264 transcoding", *Picture Coding Symposium,* 15-17, Dec 2004.

〔8〕 N. Merhav and V. Bhaskaran, "Fast algorithm for DCT-domain image down-sampling and for inverse motion compensation", *IEEE Trans. on Circuits and Systems for Video Technology,* vol. 7, no. 3, June 1997.

〔9〕 Jiang, J, and Feng, G, " The spatial relationship of dct coefficients between a block and its sub-blocks", *IEEE Trans. Signal Process.*, vol. 50, no. 5, pp. 1160-1169, May 2002.

〔10〕 Mukherjee, J., and Mitra,S.K. "Arbitrary resizing of images in DCT space", *IEEE Proc.-Vis. Image Signal Process.*, vol. 152, no. 2, pp. 155-164, April 2005.

김 용 재 (Yong-jae Kim) 준회원

2006년 8월 가톨릭대학교 컴퓨터전자공학부 학사 졸업

2006년 8월~현재 가톨릭 대학교 컴퓨터공학과 석사과정

<관심분야> 신호처리, 영상통신, 영상처리

이 창 우 (Chang-woo Lee) 정회원

현재 가톨릭대학교 정보통신전자공학부 교수

<관심분야> 영상통신, 영상처리

제 32권 7월호 통신이론 및 시스템 참조