논문 2011-48SP-1-4

H.264/AVC의 Progressive Mode에서 Thumbnail 영상 생성

(Thumbnail Generation at Progressive Mode of H.264/AVC)

오 형 석*, 김 원 하**

(Hyung Suk Oh and Wonha Kim)

요 약

본 논문에서는 공간 영역과 정수 DCT 영역을 결합하는 hybrid 영역에서 thumbnail 영상을 생성하는 기법을 제안하다. 제안하는 hybrid domain 방법은 잔여 영상 스펙트럼의 DC 값과 복원된 예측 블록의 평균값을 더하여 thumbnail 영상의 화소 값을 구한다. 또한 본 논문에서는 효율적으로 예측 블록의 평균값을 구하기 위하여 블록의 경계화소만을 복원하는 기법을 제안한다. Bit-stream을 복호한 영상을 축소하여 thumbnail 영상을 생성하는 기존의 방법과 비교하여 제안하는 방법은 동일한 thumbnail 영상을 생성하면서 복잡도를 60% 이상 줄인다.

Abstract

In this paper, we develop a method for generating thumbnail images at hybrid domain combined the spatial domain and transform domain. The proposed method generates a pixel of a thumbnail image by adding a DC value of residual transform coefficients and an average value of an estimate block. For effectively calculating average values of estimate blocks, we propose a method for reconstructing the boundary pixels of a block. In comparison to the conventional method of decoding the bit stream then scaling down the decoded images, the developed method reduces the complexity by more than 60% while producing identical thumbnail images.

Keywords: Thumbnail image, H.264/AVC, integer DCT, Video indexing

Ⅰ. 서 론

영상 블록의 스펙트럼의 DC 값은 그 영상 블록의 평균값이 된다는 사실을 바탕으로 부호화된 동영상 bit-stream에서 Thumbnail 영상을 생성하는 일반적인 방법은 I-frame 또는 I-slice에 해당하는 bit-stream에서 추출된 영상 스펙트럼의 DC 값으로 thumbnail 영상의 한 화소 값을 구하는 것이다[1~4]. 이 방법은 동영상 bit-stream에서 영상을 복원하고 다시 축소하여 thumbnail 영상을 생성하는 방법보다 역 변환과 영상의 축소가 필요 없기 때문에 계산량이 적다.

MPEG-2, 4와 같은 실수 DCT를 사용하는 부호화 기법에서는 DCT 계수가 복원 영상의 스펙트럼이 되기 때문에 I-frame을 부호화한 DCT 블록의 DC 계수가 해당 영상 블록의 평균값이 된다[5]. 따라서 MPEG-2와 4에서는 단순히 DC 계수를 추출하여 영상을 생성하면 thumbnail 영상을 획득하게 된다.

H.264/AVC에서는 실수 DCT를 근사화한 modified DCT(MDCT)을 이용하여 영상 신호의 스펙트럼을 얻어낸다[6~8]. 그러나 H.264/AVC에서는 DCT mismatch를 제거하고 정수 연산만을 수행하기 위하여 modified DCT(MDCT)에서 정수 kernel만을 추출한 정수 DCT(IntDCT)를 사용한다[7~8]. 따라서 실제로 부호화되는 정수 DCT계수들은 영상의 스펙트럼이 아니다. 정수 DCT계수로부터 영상의 스펙트럼인 MDCT 계수를 변

* 학생회원, 경희대학교 전자전파공학과
(Department of Electronics & Radio Engr. Kyung Hee University)
** 정회원-교신저자, 경희대학교 전자정보대학
(Department of Electronics and Radio Engr. Kyung Hee University)
※ 본 논문은 2009년도 경희대학교 연구년 지원에 의한 결과임.
접수일자: 2010년9월1일, 수정완료일: 2010년10월12일

환하면 정수화 오류를 발생시킨다[7~8]. 또한 MPEG-2, 4와 달리 H.264/AVC는 I-slice에서도 공간 영역의 prediction인 intra prediction을 수행한다[9]. Intra-prediction은 원 영상 블록을 부호화하지 않고 잔여 영상 블록을 부호화한다. 따라서 H.264/AVC의 I-slice를 부호화한 bit-stream으로부터 구한 MDCT계수의 DC값은 영상 블록의 평균값이 아니라 잔여 영상 블록의 평균값이다. 이와 같은 이유로 H.264/AVC로 부호화된 bit-stream의 스펙트럼을 이용하여 thumbnail 영상을 생성하는 방법은 새롭게 개발되어야 한다.

H.264/AVC의 영상 스펙트럼의 DC값을 이용하여 thumbnail 영상을 생성하기 위하여 MDCT 영역에서 intra prediction을 수행한 대표적인 연구로는 Chen의 연구가 있다[8]. Chen의 연구에서는 공간 영역의 인트라 예측에 필요한 참조 블록(reference block)을 생성하는 행렬을 개발하고, 그 행렬을 MDCT 영역에 적용하였다. 그러나 이 연구는 정수 DCT에서 MDCT로 변환에 발생되는 스케일링 인자(Post Scaling Factor, PF)의 정수화 과정을 고려하지 않았기 때문에 정수화 오류가 발생된다. 또한 H.264/AVC의 intra prediction 구조는 순환(recursive)구조 이기 때문에 정수화 오류가 계속 축적(accumulate)된다. 따라서 Chen의 연구는 실제 시스템에는 적용할 수 없다. Chen의 연구를 이용하여 Kim은 MDCT 계수들로부터 DC 계수들을 추출하여 thumbnail 영상을 생성하였다[10]. 정수화 오류를 제거하기 위하여 Kim은 lookup table(LUT)을 사용하였다. 그러나 LUT는 영상마다 최적화되어야 하고, 정수화 오류는 변환 영역에서 정확한 분석을 할 수가 없기 때문에 LUT는 정수화 오류를 완벽히 제거할 수 없다[11].

본 논문에서 제안하는 방법은 공간 영역과 정수 DCT 영역을 결합하는 hybrid 영역에서 thumbnail 영상을 생성하는 기법을 제안하다. 제안 방법은 정수화 오류를 근본적으로 제거하며, 정수 연산만 수행한다. 따라서 제안 방법은 복잡도를 급격하게 줄일 수 있는 반면 정수화 오류 없이 thumbnail 영상을 생성한다. 제안 방법의 객관적인 평가를 위하여 본 논문에서는 이론적인 복잡도를 분석하고 다양한 영상으로 실험하였다. 제안 방법은 영상을 디코딩하고 해상도를 줄이는 기존의 방법과 비교하여 복잡도를 60%이상 줄이며, 기존의 방법과 같은 thumbnail 영상을 생성한다[12~13].

논문의 구성은 다음과 같다. 제 Ⅱ장에서는 H.264/AVC의 변환영역에서 영상 블록의 스펙트럼을 소개하고, H.264/AVC의 변환영역에서 수행되는 intra prediction을 설명한다. 제 Ⅲ장에서는 H.264/AVC에서 decoding과정 없이 thumbnail 영상을 생성하는 hybrid 영역 방법을 제안한다. 제 Ⅳ장에서는 다양한 동영상을 이용한 실험 결과의 분석을 통하여 제안하는 방법이 이전의 방법에 비하여 동일한 화질의 thumbnail 영상을 생성하면서 복잡도를 60%이상 감소시킨다는 것을 검증한다. 제 Ⅴ장에서는 결론을 논한다.

Ⅱ. H.264/AVC 변환영역에서의 Intra Prediction

본 절에서는 H.264/AVC 변환영역에서의 영상 스펙트럼을 소개하고 변환영역에서 수행되는 intra prediction을 분석한다.

1. H.264/AVC 변환영역에서의 영상 스펙트럼

H.264/AVC에서는 영상 스펙트럼을 DCT를 근사화한 modified DCT(MDCT)를 통하여 얻는다[7~8]. 그러나 정수연산만을 위하여 H.264/AVC는 MDCT kernel로부터 정수만을 추출한 integer DCT(IntDCT)를 수행한다.

$\boldsymbol{x}$를 크기가 4x4인 공간 영역의 블록이라 하고, $\boldsymbol{X}_{\mathbf{MDCT}}$와 $\boldsymbol{X}_{\mathbf{IntDCT}}$를 $\boldsymbol{x}$의 스펙트럼이 되는 MDCT 변환계수 블록, 정수 DCT 계수 블록이라 한다. 이때 다음과 같은 관계가 성립된다.

$$\begin{aligned} \boldsymbol{X}_{\mathbf{MDCT}} &= \left\{\boldsymbol{C}_f \cdot \boldsymbol{x} \cdot (\boldsymbol{C}_f)^t\right\} \otimes \boldsymbol{PF}_f \\ &\equiv \boldsymbol{X}_{\mathbf{IntDCT}} \otimes \boldsymbol{PF}_f \\ \boldsymbol{x} &= (\boldsymbol{C}_i)^t \cdot \left\{\boldsymbol{X}_{MDCT} \otimes \boldsymbol{PF}_i\right\} \cdot \boldsymbol{C}_i \end{aligned} \quad (1)$$

여기서 ⊗은 element multiplication을 의미하며, 정수 forward DCT 행렬 $\boldsymbol{C}_f$, 정수 inverse DCT 행렬 $\boldsymbol{C}_i$와 forward post scaling factor 행렬 $\boldsymbol{PF}_f$, inverse post scaling factor 행렬 $\boldsymbol{PF}_i$는 다음과 같다.

$$\boldsymbol{C}_f = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix} \boldsymbol{PF}_f = \begin{bmatrix} a^2 & ab/2 & a^2 & ab/2 \\ ab/2 & b^2/4 & ab/2 & b^2/4 \\ a^2 & ab/2 & a^2 & ab/2 \\ ab/2 & b^2/4 & ab/2 & b^2/4 \end{bmatrix}$$

$$\boldsymbol{C}_i = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1/2 & -1/2 & -1 \\ 1 & -1 & -1 & 1 \\ 1/2 & -1 & 1 & -1/2 \end{bmatrix}, \boldsymbol{PF}_i = \begin{bmatrix} a^2 & ab & a^2 & ab \\ ab & b^2 & ab & b^2 \\ a^2 & ab & a^2 & ab \\ ab & b^2 & ab & b^2 \end{bmatrix}$$

여기서 $a = 1/2$, $b = \sqrt{2/5}$ 이다.

식(1)로부터 IntDCT 계수 블록은 영상 블록의 스펙트럼이 아니고, H.264/AVC에서 영상 블록의 스펙트럼은 MDCT 계수 블록이라는 것을 알 수 있다.

2. H.264/AVC 변환영역에서의 Intra Prediction

H.264/AVC의 intra prediction은 현재 블록과 현재 블록의 예측 블록의 차이를 인코딩한다. 예측 블록은 현재 블록의 왼쪽, 위 왼쪽, 위쪽 그리고 위 오른쪽에 있는 블록의 경계에 있는 13 또는 33개의 화소들로부터 생성된다. 또한 현재 블록이 인코딩되고 난 후에 현재 블록은 다음 블록의 예측을 위해 다시 복원된다. 따라서 H.264/AVC의 intra prediction은 순환 구조이다.

$\boldsymbol{x}^n$을 n번째 현재 블록이라 하고, $\boldsymbol{m}_n$을 율 왜곡 비용으로 결정된 n번째 예측 모드 행렬이라 한다[9]. 또한 n번째 예측 블록과 $(n-1)$번째 복원 블록은 $\boldsymbol{EstBL}^n$과 $\tilde{\boldsymbol{x}}^{n-1}$으로 표현하고, n번째 residual 블록과 n번째 복원된 residual 블록은 $\boldsymbol{res}^n$과 $\widetilde{\boldsymbol{res}}^n$으로 나타낸다. 그러면 intra prediction은 다음과 같이 표현될 수 있다.

$$\begin{aligned}\boldsymbol{EstBL}^n &= \boldsymbol{m}_n \cdot \tilde{\boldsymbol{x}}^{(n-1)} \\ \boldsymbol{res}^n &= \boldsymbol{x}^n - \boldsymbol{EstBL}^n \\ \tilde{\boldsymbol{x}}^n &= \widetilde{\boldsymbol{res}}^n + \boldsymbol{EstBL}^n\end{aligned} \tag{2}$$

$\boldsymbol{M}_n$을 $\boldsymbol{m}_n$의 MDCT 영역의 행렬표현이라 할 때, 식(2)로부터 MDCT 영역에서의 intra prediction은 다음과 같다.

$$\begin{aligned}\boldsymbol{Res}^n_{MDCT} &= \boldsymbol{X}^n_{MDCT} - \boldsymbol{EstBL}^n_{MDCT} \\ &= \boldsymbol{X}^n_{MDCT} - \boldsymbol{M}_n \cdot \tilde{\boldsymbol{X}}^{(n-1)}_{MDCT} \\ \tilde{\boldsymbol{X}}^n_{MDCT} &= \widetilde{\boldsymbol{Res}}^n_{MDCT} + \boldsymbol{EstBL}^n_{MDCT}\end{aligned} \tag{3}$$

Ⅲ. Thumbnail 영상을 생성하기 위한 Hybrid Domain 방법

본 절에서는 thumbnail 영상을 생성하기 위한 방법을 개발한다. 제안 방법은 H.264/AVC bit-stream을 디코딩하지 않고 정수화 오류 없이 thumbnail 영상을 생성한다. 이를 위해 변환영역에서 복원된 residual 블록의 평균인 residual DC 값을 직접 추출하여 공간영역에서 예측 블록의 평균을 구한 값과 더한다. 본 논문에서 이 방법을 hybrid domain 방법이라 한다.

Intra prediction과정에서 복원된 영상 블록의 평균값, 즉, thumbnail 영상의 한 화소인 Avg_rec은 다음과 같이 구하여 진다.

$$Avg\_rec = Avg\_res + Avg\_EstBL$$

여기서 Avg_res과 Avg_EstBL은 residual 블록의 평균값과 예측 블록의 평균값을 말한다.

1. Residual 블록의 평균값 계산, Avg_res

식(3)으로부터 Q_s가 양자화 스케일이라 하면 복원된 residual 블록의 스펙트럼은 다음과 같다[8].

$$\begin{aligned}\widetilde{\boldsymbol{Res}}^n_{MDCT} &= \partial\left(\frac{\boldsymbol{Res}^n_{MDCT}}{Q_s}\right)\cdot Q_s \\ &= \partial\left(\frac{\boldsymbol{Res}^n_{\mathrm{IntDCT}} \otimes \boldsymbol{PF}_f}{Q_s}\right)\cdot Q_s \\ &\equiv \boldsymbol{Z}^n \cdot Q_s\end{aligned} \tag{4}$$

여기서 $\partial(\boldsymbol{X})$는 $\boldsymbol{X}$에 가장 가까운 정수이고 $\boldsymbol{Z}^n$은 n번째 양자화된 residual MDCT 계수 블록이다.

정수화 연산을 위하여 H.264/AVC의 역양자화 과정은 residual MDCT 계수 블록 $\widetilde{\boldsymbol{Res}}^n_{MDCT}$ 대신 residual 정수 계수 블록 $\widetilde{\boldsymbol{Res}}^n_{\mathrm{IntDCT}}$을 생성한다. 역양자화 과정은 rescaling 행렬 $\boldsymbol{V}$와 $\boldsymbol{Z}^n$의 곱에 의해 구현된다. 여기서 rescaling 행렬 $\boldsymbol{V} = Q_s \cdot \boldsymbol{PF}_i \cdot 64$은 rounding 연산에 의해 정수화 된다. 양자화 스케일 Q_s는 양자화 파라미터 QP 가 6씩 증가할 때마다 2의 배수만큼 증가한다. Mapping 함수 $Q_s(QP)$는 다음과 같다[7~8].

$$Q_s(QP) = \frac{M(\mathrm{mod}(QP,6))}{2^6}\cdot 2^{floor(QP/6)} \tag{5}$$

여기서 $\mathrm{mod}(M,N)$은 modular 연산이고, M은 다음과 같은 mapping table을 가진다.

$$\begin{aligned}&\{M(0),M(1),M(2),M(3),M(4),M(5)\} \\ &\quad = \{40,44,52,56,64,72\}\end{aligned}$$

따라서 rescaling 행렬 $\boldsymbol{V}$을 사용하여 $\boldsymbol{Z}^n$으로부터 $\widetilde{\boldsymbol{Res}}^n_{\mathrm{IntDCT}}$을 얻기 위한 역양자화 과정은 다음과 같다.

$\widetilde{\boldsymbol{Res}}^n_{\mathrm{IntDCT}}$

$$= Z^n \otimes \left\{ \partial(Q_s(QP) \cdot PF_i \cdot 2^6) \cdot 2^{floor(QP/6)} \right\} \quad (6)$$

식(6)에서 보듯이 역양자화에서 발생하는 정수화 오류는 rescaling 행렬 V의 PF_i 성분들에 기인한다. 그러나 $\{(i,j)\}=\{(0,0),(2,0),(0,2),(2,2)\}$의 위치에 있는 PF_i 성분들은 1/4이기 때문에 그 위치에 있는 성분들은 정수화 오류를 포함하고 있지 않다. 따라서 그 위치에 대응하는 rescaling 행렬 V의 성분들은 다음과 같다.

For $\{(i,j)\}=\{(0,0),(2,0),(0,2),(2,2)\}$

$$V(i,j) = \partial(Q_s(QP) \cdot PF_i(i,j) \cdot 64)$$

식(5)을 대입하면,

$$= \partial\left(\frac{M(\mathrm{mod}(QP,6))}{2^6} \cdot 2^{floor(QP/6)} \cdot \frac{1}{4} \cdot 64 \right)$$

$$= \partial\left(\frac{M(\mathrm{mod}(QP,6))}{4} \cdot 2^{floor(QP/6)} \right) \quad (7)$$

식 (7)의 값들은 정수이기 때문에

$$= M(\mathrm{mod}(QP,6)) \cdot 2^{floor(QP/6)-2}.$$

식 (7)로부터 $\{(i,j)\}=\{(0,0),(2,0),(0,2),(2,2)\}$에 대하여 정수화 오류는 0이다. 따라서 residual 계수 블록의 DC 값이 residual 블록의 평균값과 같다는 사실을 고려하여 Avg_res는 다음과 같다.

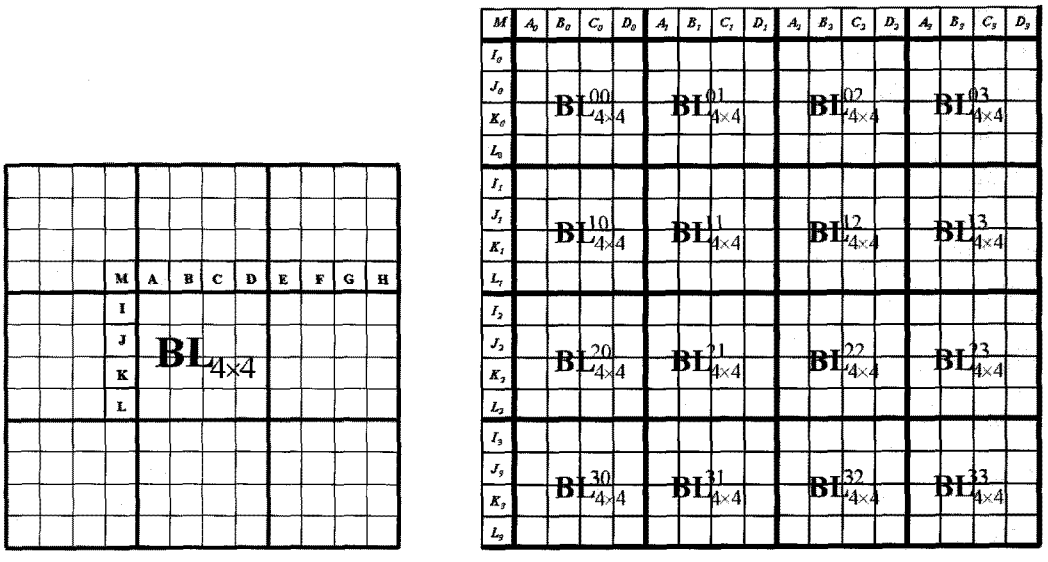
$$Avg\_res = \widetilde{Res}^{(0,0)}_{MDCT} = \frac{\widetilde{Res}^{(0,0)}_{\mathrm{IntDCT}}}{2^{4+floor(QP/6)}} \quad (8)$$

식 (8)에서 복원된 residual 계수 블록의 DC 값 $\widetilde{Res}^{(0,0)}_{\mathrm{IntDCT}}$은 rounding에 의한 오류를 가지고 있지 않으므로 residual 블록의 평균값은 변환영역에서 직접 추출할 수 있다.

2. 예측 블록의 평균값 계산, Avg_EstBL

예측 블록은 intra prediction mode에 따라 이전에 복원된 블록들의 경계에 있는 13 또는 33개의 화소들로부터 생성된다. 따라서 예측 블록의 평균값은 13 또는 33개의 화소들로부터 직접 구할 수 있다. <표 1>은 각 mode에 따라 13 또는 33개의 화소들로부터 직접 구하는 식을 보여준다. <그림 1>의 (a)와 (b)는 4x4 intra prediction mode와 16x16 intra prediction mode에 대하여 13개와 33개의 참조 화소를 나타낸다. 또한 <그림 1>의 (b)는 4x4블록들의 그룹으로써 16x16블록을 표현하였다.

복원된 블록들의 화소들이 thumbnail image 생성에 필요하지 않더라도 현재의 intra decoding은 예측 블록을 생성하기 위해 전체 블록들을 복원한다[11]. 따라서



(a) (b)

그림 1. 블록 크기에 따른 참소 화소들. (a) 4x4 intra prediction을 위한 13개의 참조 화소들. (b) 16x16 intra prediction을 위한 33개의 참조 화소들.

Fig. 1. Reference pixels in accordance with block sizes. (a) The 13 reference pixels for the 4x4 intra prediction. (b) The 33 reference pixels for 16x16 intra prediction.

표 1. 예측 블록의 평균값을 계산하는 수식
Table 1. Formulas for calculating the averages of estimate blocks

블록	예측모드	예측 블록 평균값
4x4	0:vertical	$(4\cdot A+4\cdot B+4\cdot C+4\cdot D)/16$
	1:horizontal	$(4\cdot I+4\cdot J+4\cdot K+4\cdot L)/16$
	2:DC	$(A+B+C+D+I+J+K+L)/8$
	3:diagonal down-left	$(A+4\cdot B+8\cdot C+12\cdot D+14\cdot E+12\cdot F+8\cdot G+5\cdot H)/64$
	4:diagonal down-right	$(14\cdot M+12\cdot A+8\cdot B+4\cdot C+D+12\cdot I+8\cdot J+4\cdot K+L)/64$
	5:vertical-right	$(11\cdot M+16\cdot A+15\cdot B+10\cdot C+3\cdot D+5\cdot I+3\cdot J+K)/64$
	6:horizontal-down	$(11\cdot M+5\cdot A+3\cdot B+C+16\cdot I+15\cdot J+10\cdot K+3\cdot L)/64$
	7:vertical-left	$(3\cdot A+10\cdot B+15\cdot C+16\cdot D+13\cdot E+6\cdot F+G)/64$
	8:horizontal-up	$(3\cdot I+10\cdot J+15\cdot K+36\cdot L)/64$
16x16	0:vertical	$avg_{i0}=(4\cdot A_i+4\cdot B_i+4\cdot C_i+4\cdot D_i)/16$ $avg_{i1}=(4\cdot A_i+4\cdot B_i+4\cdot C_i+4\cdot D_i)/16$ $avg_{i2}=(4\cdot A_i+4\cdot B_i+4\cdot C_i+4\cdot D_i)/16$ $avg_{i3}=(4\cdot A_i+4\cdot B_i+4\cdot C_i+4\cdot D_i)/16$ $avg_{ij}(i=0,1,2,3)$는 <그림 1>.(b)에 있는 BL_{ij}의 평균이다.
	1:horizontal	$avg_{0j}=(4\cdot I_j+4\cdot J_j+4\cdot K_j+4\cdot L_j)/16$ $avg_{1j}=(4\cdot I_j+4\cdot J_j+4\cdot K_j+4\cdot L_j)/16$ $avg_{2j}=(4\cdot I_j+4\cdot J_j+4\cdot K_j+4\cdot L_j)/16$ $avg_{3j}=(4\cdot I_j+4\cdot J_j+4\cdot K_j+4\cdot L_j)/16$ $avg_{ij}(i,j=0,1,2,3)$는 <그림 1>.(b)에 있는 BL_{ij}의 평균이다.
	2:DC	$\begin{pmatrix}A_1+B_1+C_1+D_1+I_1+J_1+K_1+L_1\\+A_2+B_2+C_2+D_2+I_2+J_2+K_2+L_2\\+A_3+B_3+C_3+D_3+I_3+J_3+K_3+L_3\\+A_4+B_4+C_4+D_4+I_4+J_4+K_4+L_4\end{pmatrix}/32$
	3:plane	$ih=\begin{Bmatrix}(A_3-C_2)+2\cdot(B_3-B_2)+3\cdot(C_3-A_2)+4\cdot(D_3-D_1)\\+5\cdot(A_4-C_1)+6\cdot(B_4-B_1)+7\cdot(C_4-A_1)+8\cdot(D_4-M_1)\end{Bmatrix}$ $iv=\begin{Bmatrix}(I_3-K_2)+2\cdot(J_3-J_2)+3\cdot(K_3-I_2)+4\cdot(L_3-L_1)\\+5\cdot(I_4-K_1)+6\cdot(J_4-J_1)+7\cdot(K_4-I_1)+8\cdot(L_4-M_1)\end{Bmatrix}$ $ib=(5\cdot ih+32)\gg 6,\ ic=(5\cdot iv+32)\gg 6,\ iaa=16\cdot(D_4+L_4)$
		$avg_{ij}=2iaa-11ib-11ic+32,\ avg_{ij}=2iaa-3ib-11ic+32$ $avg_{ij}=2iaa+5ib-11ic+32,\ avg_{ij}=2iaa+13ib-11ic+32$ $avg_{ij}=2iaa-11ib-3ic+32,\ avg_{ij}=2iaa-3ib-3ic+32$ $avg_{ij}=2iaa+5ib-3ic+32,\ avg_{ij}=2iaa+13ib-3ic+32$ $avg_{ij}=2iaa-11ib+5ic+32,\ avg_{ij}=2iaa-3ib+5ic+32$ $avg_{ij}=2iaa+5ib+5ic+32,\ avg_{ij}=2iaa+13ib+5ic+32$ $avg_{ij}=2iaa-11ib+13ic+32,\ avg_{ij}=2iaa-3ib+13ic+32$ $avg_{ij}=2iaa+5ib+13ic+32,\ avg_{ij}=2iaa+13ib+13ic+32$ 여기서, $avg_{ij}(i,j=0,1,2,3)$는 <그림 1>.(b)에 있는 BL_{ij}의 평균이다.

효율적으로 예측 블록의 생성에 필요한 화소들을 복원하기 위하여 본 논문에서는 블록의 경계에 있는 7개의 화소들만 복원하는 기법을 제시한다.

<그림 2>.(a)는 4x4 예측 블록을 생성하기 위한 참조 화소들을 보여준다. <그림 2>.(a)에서 x^n은 인코딩되기 위한 현재 블록이다. A~M 화소들은 블록 x^n을 예측하기 위한 참조 화소들이다. 블록 x^n이 인코딩된 후에 복원된 블록 $\tilde{x}^n$의 오른쪽 경계 화소들은 $x^{(n+1)}$을 예측하기 위하여 I,J,K,L로써 사용된다. 복원된 블록 $\tilde{x}^n$의 아래쪽 경계 화소들은 블록 x^m을 예측하기 위하여 E,F,G,H화소가 되며, 또한 블록 $x^{(m+1)}$을 예측하기 위하여 A,B,C,D화소가 된다. 블록 $x^{(m+2)}$을 예측하기 위한 M화소는 복원된 블록 $\tilde{x}^n$의 오른쪽 아래의 화소이다. <그림 2>.(b)는 복원된 블록 $\tilde{x}^n$의 7개의 경계화소들이 어떻게 다음 블록들의 참조화소로 이용되는지를 보여준다. 유사하게 16x16 intra prediction mode에 대한 33개의 참조화소들은 4x4 블록들의 7개의 경계화소들의 조합으로 표현될 수 있다. 따라서 어떠한 intra prediction mode에서든지 예측블록은 복원된 블록의 7개 경계 화소들의 조합으로 구성될 수 있다.

이러한 사실로부터 블록의 7개 경계화소들만을 복원시키는 방법을 개발한다. 식(2)에서 7개의 경계화소들만을 복원하기 위해서는 7개 residual 경계화소들과 7개 예측 경계화소들이 복원되어야 한다. <그림 2>.(b)에서 보듯이 7개 예측 경계화소들은 이전에 복원된 7개 경계화소들의 조합으로 생성될 수 있기 때문에 7개 예측 경계화소들을 생성하는 복잡도는 4x4 예측블록을 생성하는 것과 비교하여 약 60% 줄어든다.

식(1)에서 복원된 residual 블록 $\widetilde{res}$는 정수 IDCT로부터 다음과 같이 계산된다[12~13].

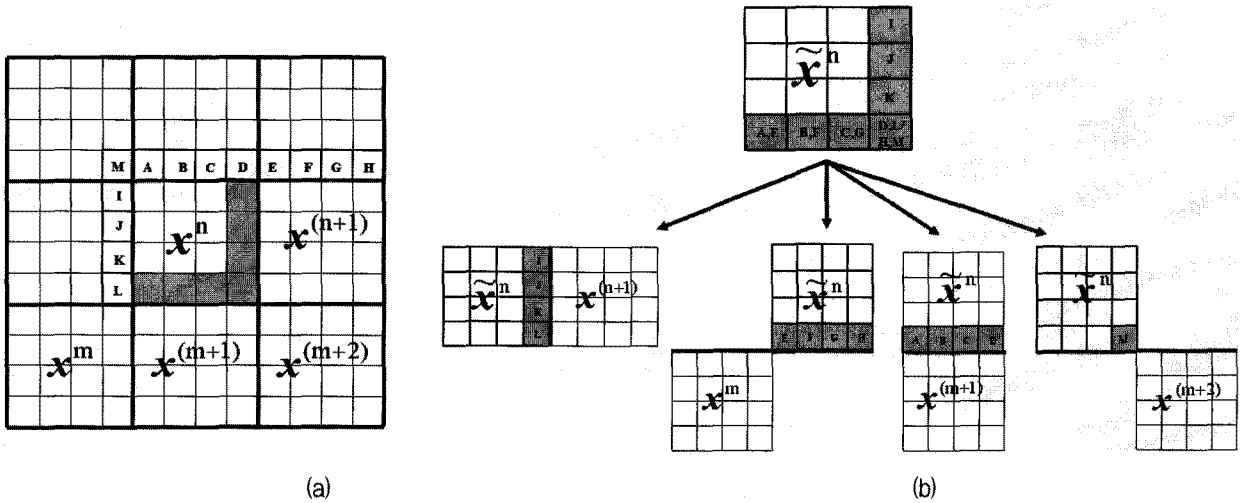
$$\widetilde{res} = C_i^t \cdot \widetilde{Res}_{\mathrm{IntDCT}} \cdot C_i$$

또한 위의 식은 7개의 residual 경계화소들만을 구하기 위하여 7개 residual 경계화소들과 관련된 연산만을 수행함으로써 복잡도를 급격히 줄일 수 있다. $\widetilde{res}^{(i,j)}$와 $\widetilde{Res}^{(i,j)}_{\mathrm{IntDCT}}$, $(i,j=0,1,2,3)$를 $\widetilde{res}$와 $\widetilde{Res}_{\mathrm{IntDCT}}$의 성분이라 할 때, 7개 경계화소들의 위치는 다음과 같다.
$\{(i,j)\}=\{(0,3),(1,3),(2,3),(3,0),(3,1),(3,2),(3,3)\}$
7개 residual 경계화소들에 대한 정수 IDCT는 다음과 같이 구하여 진다.

$$\begin{bmatrix}\widetilde{res}_{03}\\ \widetilde{res}_{13}\\ \widetilde{res}_{23}\\ \widetilde{res}_{33}\end{bmatrix} = C_i^t \cdot \begin{bmatrix}H_0\\ H_1\\ H_2\\ H_3\end{bmatrix} = \begin{bmatrix}H_0+H_1+H_2+H_3/2\\ H_0+H_1/2-H_2-H_3\\ H_0-H_1-H_2+H_3/2\\ H_0-H_1+H_2-H_3/2\end{bmatrix} \tag{9}$$

$$\begin{bmatrix}\widetilde{res}_{30}\\ \widetilde{res}_{31}\\ \widetilde{res}_{32}\end{bmatrix}^T = \begin{bmatrix}V_0\\ V_1\\ V_2\\ V_3\end{bmatrix}^T \cdot C_i = \begin{bmatrix}V_0+V_1+V_2+V_3/2\\ V_0+V_1/2-V_2-V_3\\ V_0-V_1-V_2+V_3/2\end{bmatrix}$$

여기서 horizontal 요소들은



(a) (b)

그림 2. 4x4 예측 블록을 생성하기 위한 참조 화소들. (a) 블록 을 예측하기 위하여 사용되는 13개의 참조 화소들. (b) 복원된 블록 에서 다음 블록을 예측하기 위한 7개 경계 화소들.

Fig. 2. Reference pixels for constructing a 4x4 estimate block. (a) Constitution of 13 reference pixels used for estimating the block . (b) The 7 boundary pixels at the reconstructed block for estimating next blocks.

$$\begin{bmatrix} H_0 \\ H_1 \\ H_2 \\ H_3 \end{bmatrix} = \begin{bmatrix} \widetilde{Res}^{00}_{IntDCT} & \widetilde{Res}^{01}_{IntDCT} & \widetilde{Res}^{02}_{IntDCT} & \widetilde{Res}^{03}_{IntDCT} \\ \widetilde{Res}^{10}_{IntDCT} & \widetilde{Res}^{11}_{IntDCT} & \widetilde{Res}^{12}_{IntDCT} & \widetilde{Res}^{13}_{IntDCT} \\ \widetilde{Res}^{20}_{IntDCT} & \widetilde{Res}^{21}_{IntDCT} & \widetilde{Res}^{22}_{IntDCT} & \widetilde{Res}^{23}_{IntDCT} \\ \widetilde{Res}^{30}_{IntDCT} & \widetilde{Res}^{31}_{IntDCT} & \widetilde{Res}^{32}_{IntDCT} & \widetilde{Res}^{33}_{IntDCT} \end{bmatrix} \cdot \begin{bmatrix} 1 \\ -1 \\ 1 \\ -\frac{1}{2} \end{bmatrix}$$

$$= \begin{bmatrix} \widetilde{Res}^{00}_{IntDCT} - \widetilde{Res}^{01}_{IntDCT} + \widetilde{Res}^{02}_{IntDCT} - \widetilde{Res}^{03}_{IntDCT}/2 \\ \widetilde{Res}^{10}_{IntDCT} - \widetilde{Res}^{11}_{IntDCT} + \widetilde{Res}^{12}_{IntDCT} - \widetilde{Res}^{13}_{IntDCT}/2 \\ \widetilde{Res}^{20}_{IntDCT} - \widetilde{Res}^{21}_{IntDCT} + \widetilde{Res}^{22}_{IntDCT} - \widetilde{Res}^{23}_{IntDCT}/2 \\ \widetilde{Res}^{30}_{IntDCT} - \widetilde{Res}^{31}_{IntDCT} + \widetilde{Res}^{32}_{IntDCT} - \widetilde{Res}^{33}_{IntDCT}/2 \end{bmatrix},$$

그리고 vertical 요소들은

$$\begin{bmatrix} V_0 \\ V_1 \\ V_2 \\ V_3 \end{bmatrix}^T = \begin{bmatrix} 1 \\ -1 \\ 1 \\ -\frac{1}{2} \end{bmatrix}^T \cdot \begin{bmatrix} \widetilde{Res}^{00}_{IntDCT} & \widetilde{Res}^{01}_{IntDCT} & \widetilde{Res}^{02}_{IntDCT} & \widetilde{Res}^{03}_{IntDCT} \\ \widetilde{Res}^{10}_{IntDCT} & \widetilde{Res}^{11}_{IntDCT} & \widetilde{Res}^{12}_{IntDCT} & \widetilde{Res}^{13}_{IntDCT} \\ \widetilde{Res}^{20}_{IntDCT} & \widetilde{Res}^{21}_{IntDCT} & \widetilde{Res}^{22}_{IntDCT} & \widetilde{Res}^{23}_{IntDCT} \\ \widetilde{Res}^{30}_{IntDCT} & \widetilde{Res}^{31}_{IntDCT} & \widetilde{Res}^{32}_{IntDCT} & \widetilde{Res}^{33}_{IntDCT} \end{bmatrix}$$

$$= \begin{bmatrix} \widetilde{Res}^{00}_{IntDCT} - \widetilde{Res}^{10}_{IntDCT} + \widetilde{Res}^{20}_{IntDCT} - \widetilde{Res}^{30}_{IntDCT}/2 \\ \widetilde{Res}^{01}_{IntDCT} - \widetilde{Res}^{11}_{IntDCT} + \widetilde{Res}^{21}_{IntDCT} - \widetilde{Res}^{31}_{IntDCT}/2 \\ \widetilde{Res}^{02}_{IntDCT} - \widetilde{Res}^{12}_{IntDCT} + \widetilde{Res}^{22}_{IntDCT} - \widetilde{Res}^{32}_{IntDCT}/2 \\ \widetilde{Res}^{03}_{IntDCT} - \widetilde{Res}^{13}_{IntDCT} + \widetilde{Res}^{23}_{IntDCT} - \widetilde{Res}^{33}_{IntDCT}/2 \end{bmatrix}.$$

본 논문에서는 식(9)을 7-integer IDCT (7-IntIDCT)라 칭한다. 7-IntIDCT의 연산은 16개 성분들의 연산 대신 7개만을 연산하기 때문에 연산량을 줄일 수 있다. 또한 7-IntIDCT의 연산 구조는 대각선 방향의 계산이 필요 없고, 내부의 연산 연관성이 없는 non-recursive 구조이기 때문에 순환 연산에 대한 복잡도를 줄일 수 있다.

<표 2>는 7-IntIDCT와 기존의 4x4 fast와 direct 정수 IDCT의 비교를 보여준다. 기존의 4x4 fast 정수 IDCT와 비교할 때, 7-IntIDCT는 덧셈 연산에 대하여 42%, 곱셈 연산에 대하여 54%, memory access에 대하여 84% 줄어든다. 또한 7-IntIDCT는 direct IDCT와 비교할 때, 덧셈 연산에 대하여 77%, 곱셈 연산에 대하여 80%, memory access에 대하여 56% 줄어든다. Non-recursive 구조를 채택하면서 7-IntIDCT는 4x4 fast IDCT에 비하여 memory access를 급격하게 줄이며, 제안 방법이 더 적은 화소들을 복원하기 위한 연산만을 필요로 하므로 direct IDCT에 비하여 연산량을 급격히 줄인다.

표 2. 4x4 IDCT와 제안한 7-IntIDCT의 복잡도
Table 2. Complexity of 4x4 integer IDCT and proposed 7-IntIDCT.

	덧셈	곱셈	M/A	Structure
Fast IDCT	96	48	176	Recursive
Direct IDCT	240	112	64	Non-recursive
7-IDCT	56	22	28	Non-recursive

3. 제안 방법의 분석

<그림 3>은 제안 방법의 전반적인 구조를 보여준다. 양자화된 residual 정수 계수 블록 Z는 I-slice bit-stream으로부터 variable length decoding(VLD)을 통하여 구하여 진다. 또한 역 양자화 과정을 통하여 residual 정수 계수 블록 $\widetilde{Res}_{IntDCT}$은 생성이 되고,

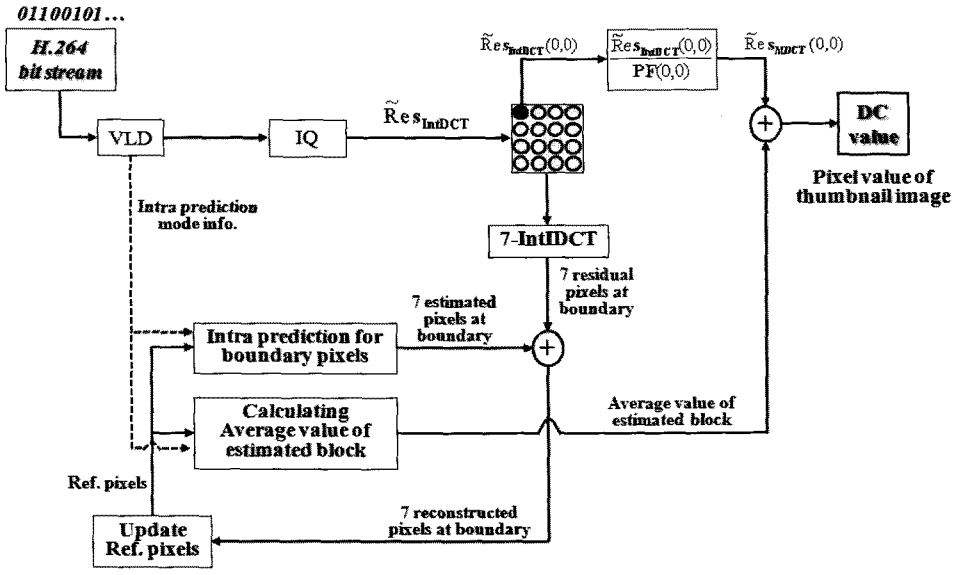


그림 3. 제안 방법의 전반적인 구조.
Fig. 3. The overall scheme for the proposed thumbnail image generation method.

residual 블록의 평균인 Avg_res는 residual 정수 계수 블록의 DC 값 $\widetilde{Res}_{\mathrm{IntDCT}}(0,0)$으로 직접 계산될 수 있다. 4x4 블록에 대한 7개의 residual 화소들은 7-IntIDCT에 의해 복원되고, 4x4 블록에 대한 7개의 예측 화소들은 이전에 복원된 7개의 화소들의 조합으로부터 생성된다. 따라서 현재 블록의 7개의 화소들은 7개의 residual 화소들과 예측 화소들의 덧셈에 의하여 생성된다. 예측 블록의 평균인 Avg_EstBL은 <표 1>에 의해 계산된다. 따라서 현재 블록의 평균값은 Avg_res와 Avg_EstBL의 합이 된다.

제안 방법의 특징은 다음과 같다. 첫 번째, 제안 방법은 transform 영역과 spatial 영역을 같이 사용하는 hybrid 영역 방법이며, 두 번째, 블록의 7개의 화소만을 복원시키기 위하여 7-IntIDCT를 사용한다. 마지막으로 제안방법은 모든 예측 블록을 복원시키지 않고 7개의 예측 화소들만 복원시킨다.

<표 3>은 이론적인 복잡도를 기존의 방법과 비교한다. 제안 방법의 복잡도는 7-IntIDCT 연산량, 7개의 예측 화소들을 생성하는 연산량과 예측 블록의 평균값을 계산하는 연산량의 합이다. 반면 기존 방법의 복잡도는 4x4 블록의 정수 IDCT, 4x4 예측 블록을 생성하는 연산량과 예측 블록의 평균값을 계산하는 연산량의 합이다. <표 3>에서 보듯이, 제안 방법은 덧셈 연산은 27.7%, 곱셈은 46.1%, memory access는 80%를 평균적으로 줄인다[15].

표 3. 각 예측 모드에 따른 thumbnail 영상의 한 화소 생성에 필요한 이론적인 연산량 비교

Table 3. Comparison to theoretical complexity of one pixel in thumbnail image at each intra prediction.

Mode		기존의 방법			제안 방법		
블록	예측	덧셈	곱셈	M/A	덧셈	곱셈	M/A
4x4	0	128	49	196	88	23	33
	1	128	49	196	88	23	33
	2	136	50	200	96	24	38
	3	148	62	200	108	37	43
	4	149	63	201	109	37	44
	5	154	65	200	114	39	43
	6	154	65	200	114	39	43
	7	153	64	199	113	38	42
	8	143	58	196	103	32	39
16x16	0	128	49	196	88	23	33
	1	128	49	196	88	23	33
	2	136	50	200	96	24	38
	3	241	74	225	201	48	61

Ⅳ. 실 험

제안하는 hybrid 영역 기법과 부호화된 bit-stream에서 영상을 복원하고 공간 영역에서 블록의 평균값으로 thumbnail 영상을 생성하는 기존의 방법을 비교하였다.

Thumbnail 영상의 화질을 비교적 명확하게 비교하기 위하여 HD(1920x1080)의 해상도인 시험 동영상들을 사용하였다. H.264/AVC 부호화는 MPEG의 JM9.8를 사용하였고, Profile은 main profile으로 하였다[16].

따라서 thumbnail의 영상 크기는 480x270이 된다. 또한 미세한 화질의 차이도 평가하기 위하여 비교적 고화질에 생성하도록 양자화 계수 QP는 20으로 하였다.

<그림 4>은 기존의 방법과 제안하는 방법으로 생성된 thumbnail 영상의 화질을 비교한다. 공간영역 방법과 제안 방법으로 생성된 thumbnail 영상들은 완벽하게 동일하기 때문에 <그림 4>.(b)에서 보듯이 두 thumbnail 영상의 수치적인 차이도 0이 된다. 따라서 제안하는 방법은 기존의 방법과 정확히 일치하는 thumbnail 영상을 생성한다는 것을 알 수 있다.

<표 4>은 시험 영상들을 복호화하여 thumbnail을 생성하는 방법과 제안하는 hybrid 영역 방법의 연산량을 비교한 것이다. <표 4>의 복잡도는 intra prediction mode의 실제 빈도수와 <표 3>에서 제시된 각 mode에서 요구되는 연산량을 곱한 것이다. 표에서 보듯이, 제안하는 방법의 복잡도는 기존의 방법에 비하여 평균적으로 덧셈 연산은 26.9%, 곱셈 연산은 45.5%, memory access는 79.9% 감소하는 것을 알 수 있다.

본 실험에서는 3GHz 32-bit pentium processer에서

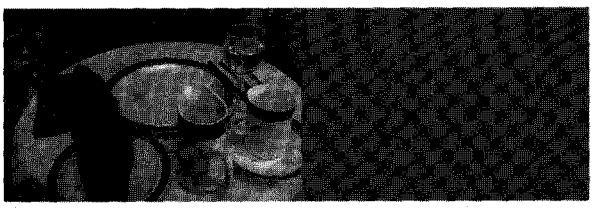


그림 4. 영상의 크기가 1/4로 줄어든 thumbnail 영상. (a) 제안 방법으로 생성된 thumbnail 영상. (b) 제안 방법과 기존의 방법으로 생성된 thumbnail 영상의 차이영상.

Fig. 4. Thumbnail images of which the image resolution is reduced to 1/4 for each direction. (a) Thumbnail image generated by the proposed method. (b) Difference image between images generated by the proposed method and the conventional method.

표 4. Thumbnail 영상의 한 화소 생성에 필요한 복잡도 비교

Table 4. Complexity comparison needing one pixel creation of thumbnail image.

시험 영상	방법	operations		
		덧셈	곱셈	M/A
Table setting	기존방법	146.3	56.4	200.3
	제안방법	106.3	30.4	39.7
Playing cards	기존방법	152	58	201.8
	제안방법	112	32	41.3
Rolling tomatoes	기존방법	147.7	56.9	200.5
	제안방법	107.7	30.9	40.1

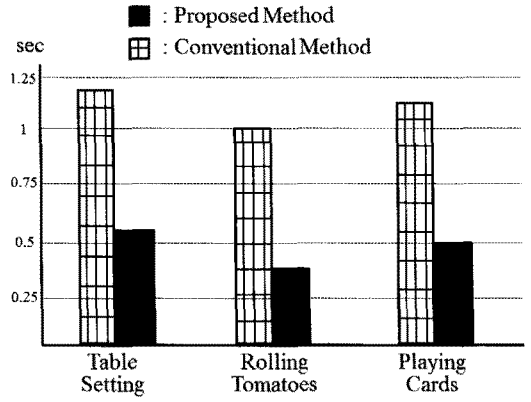


그림 5. Thumbnail 영상 생성의 수행시간 비교

Fig. 5. Actual execution times for generating a thumbnail image.

제안 방법과 기존 방법의 실제 실행 속도를 측정하였다. <그림 5>는 실행 속도를 측정한 결과이며, <그림 5>에서 보듯이 제안 방법은 기존의 방법보다 약 60% 빠르다는 것을 확인할 수 있다.

V. 결 론

본 논문에서는 H.264/AVC로 부호화된 bit-stream의 I-slice로부터 thumbnail 영상을 생성하는 기법을 제안하였다. 제안하는 방법은 잔여 영상의 평균값과 예측 블록의 평균값을 더하여 thumbnail 영상의 화소값을 구한다. 잔여 영상의 평균값은 잔여 영상 스펙트럼의 DC을 이용하여 구하고, 예측 블록의 평균값은 참조화소들로부터 직접 구하기 때문에 제안 방법은 hybrid domain에서 수행된다. 예측블록의 평균값을 위한 참조화소들은 이전에 코딩된 블록들의 경계에 있는 7개의 화소들만 복원하여 intra prediction mode와 블록 크기에 맞게 조합된다. 따라서 제안 방법은 블록의 경계에 있는 7개의 화소들만 복원하여 참조화소들을 구성하기 때문에 블록들을 모두 복원하는 기존의 방법보다 계산량을 현저히 낮출 수 있다.

실험에서는 개발된 기법을 HD 영상에 적용하였을 때 평균적으로 덧셈 연산은 26.9%, 곱셈 연산은 45.5%, memory access는 79.9%의 계산량 감소를 보여주며, 기존의 방법으로 생성한 thumbnail 영상과 정확히 동일하다는 것을 증명하였다. 또한 기존의 방법보다 약 60% 빠르다는 것을 확인하였다.

참 고 문 헌

[1] 오형석, 김원하, 구준모, "임의의 직교 블록 변환 영역에서 영상 특성에 적합한 필터를 사용한 영상 해상도 변환", 대한전자공학회, 제 45권 SP편, 제 5호, 2008년 9월.

[2] B. L. Yeo and B. Liu, "On the extraction of DC sequences from MPEG compressed video", IEEE ICIP, vol. II, pp. 260-263, 1995.

[3] ISO/IEC, JTC1/SC29/WG11, Information Technology-coding of Audio-Visual objects, Part 2: Video, FDIS 14496-2 Oct. 1998.

[4] W. B. Pannebaker and J. L. Mitchell, JPEG Still Image Data Compression Standard, Van Nostrand, New York, 1993.

[5] J. Song, B. Yeo, "Fast Extraction of Spatially Reduced Image Sequences from MPEG-2 Compressed Video", IEEE Trans. Circ. and Syst. Video Tech., vol. 9, No. 7, pp.1100-1114, Oct. 1999.

[6] H. S. Malvar, A. Hallapuro, M. Karczewicz and L. Kerofsky, "Low-Complexity Transform and Quantization in H.264/AVC", IEEE Trans. Circ. and Syst. Video Tech., vol. 13, No. 7, pp.598-603, July 2003.

[7] E. Iain and G. Richardson, H.264 and MPEG-4 Video Compression, Wiley, 2004.

[8] Draft ITU-T Recommendation and Final Draft International Standard of Joint Video Specification (ITU-T Rec. H.264 | ISO/IEC 14496-10 AVC), 2003.

[9] Chen Chen, Ping-Hao, Chen H, "Transform-Domain Intra Prediction for H.264", IEEE ISCAS, pp.1497-1500, May 2005.

[10] E. S. Kim, T. W. Um, and S. J. Oh, "A fast thumbnail extraction method in H.264/AVC video stream", IEEE Trans. Consum. Electron., vol. 55,

No. 3, pp. 1424-1430, Aug. 2009.
[11] C. Chen, P. H. Wu, and H. Chen, "Rounding mismatch between spatial domain and transform domain", IEEE Trans. Circ. and Syst. Video Tech., vol. 16, No. 10, pp. 1286-1293, Oct. 2006.
[12] I. Hamzaoglu, O. Tasdizen, E. Sahin, "An Efficient H.264 Intra Frame Coder System", IEEE Trans. Consum. Electron., Vol. 54, No. 4, Nov. 2008.
[13] 오형석, 신동인, 김원하, "H.264/AVC에서 고속 I Slice 부호화/복호화 방법", 대한전자공학회, 제 46권 CI편, 제 2호, 2009년 3월.
[14] Z. Cheng, C. Chen, B. Liu, and J. Yang, "High Throughput 2-D Transform Architectures for H.264 Advanced Video Coders", IEEE Asia-Pacific Conference on Circuits and Systems, pp. 1141-1144, Dec. 2004.
[15] C.H. Roth and L.K. John, Digital Systems Design using VHDL, Thomson, Toronto, 2008.
[16] H.264/AVC Reference Software Version 9.8 http://iphome.hhi.de/suehring/tml, 2007.

저 자 소 개

오 형 석(학생회원)
2005년 경희대학교 전자공학과 학사졸업
2007년 경희대학교 일반대학원 전자전파공학과 석사졸업
2009년 9월~2010년 8월 University of California San Diego (UCSD) 교환학생
2007년~현재 경희대학교 일반대학원 전자전파공학과 박사 과정
<주관심분야 : 영상/동영상부호화, 동영상신호처리>

김 원 하(정회원)
1985년 연세대학교 전자공학과 학사졸업
1988년 Univ. of Wisconsin-Madison 전기공학과 석사졸업
1997년 Univ. of Wisconsin -Madison 전기공학과 박사졸업
1996년 1월~7월 (미) Motorola, 연구원
1997년 8월~2000년 2월 Los Alamos National Lab. 연구원
2003년 8월 명지대학교 정보통신공학과 조교수
2009년 9월~2010년 8월 University of California San Diego(UCSD) 방문교수
2003년 8월~현재 경희대학교 전자정보대학 전자전파공학과 교수
<주관심분야 : 영상/동영상부호화, 동영상신호처리>