DMB 서비스를 위한 DCT 기반 MPEG-2/H.264 비디오 트랜스코더 시스템 구조

이 주 경[†] · 권 순 영[††] · 박 성 호[†††] · 김 영 주[††††] · 정 기 동[†††††]

요 약

DMB 서비스를 위해 제공되는 대부분의 비디오 컨텐츠는 MPEG-2 규격으로 압축된 채 제공되므로 실제 서비스를 위해서 H.264 규격으로 트랜스코딩을 수행해야 한다. 현재 사용되는 트랜스코딩 방식은 MPEG-2 비트열(bit-stream)의 디코딩과 H.264 규격으로의 인코딩 과정을 연속적으로 수행하는 픽셀 기반 직렬 구조형(CPDT, Cascaded Pixel-Domain Transcoding Architecture)이다. 이 방식은 두 표준의 소스 코드를 수정없이 사용할 수 있으므로 구현이 용이하지만 변환을 위한 처리 시간이 길고 디코딩과 인코딩을 반복하므로 화질의 열화가 발생 할 수 있다.

본 논문에서는 MPEG-2로 압축된 비디오 비트열을 H.264로 트랜스코딩할 때 변환 시간을 향상할 수 있는 DCT 기반의 열린 회로형 트랜스코더 구조(DCT-OPEN)와 변환시간은 CPDT와 유사하지만 화질면에서 우수한 DCT 기반 닫힌 회로형 트랜스코더(DCT-CLOSED) 구조를 제안한다. 제안된 구조에서는 CPDT 방식과 달리 압축 과정의 중간 단계인 DCT(Discrete Cosine Transform)를 이용하여 변환을 수행한다. 이때, MPEG-2와 H.264의 DCT 단위와 방법이 상이하므로 [1, 2]에서 제안된 방식을 이용하여 DCT 간의 변환을 수행한다. 제안된 구조의 성능 평가를 위해 MPEG-2 TM5와 H.264 JM8 코덱을 수정하여 다양한 구조를 구현하였으며 실험 결과 DCT-OPEN의 경우 CPDT에 비하여 계산 복잡도에서 우수하지만 PSNR 성능은 낮게 나타났으며 DCT-CLOSED의 경우 계산 복잡도는 높으나 화질에서 우수한 것으로 나타났다.

키워드 : 비디오 트랜스코더, 트랜스코딩, CPDT, DCT-OPEN, DCT-CLOSED, H.264, MPEG-2

DCT-domain MPEG-2/H.264 Video Transcoder System Architecture for DMB Services

Joo-Kyong Lee[†] · Soon-Young Kwon[††] · Seong-Ho Park[†††] · Young-Ju Kim[††††] · Ki-Dong Chung[†††††]

ABSTRACT

Most of the multimedia contents for DBM services are provided as MPEG-2 bit streams. However, they have to be transcoded to H.264 bit streams for practical services because the standard video codec for DMB is H.264.

The existing transcoder architecture is Cascaded Pixel-Domain Transcoding Architecture, which consists of the MPEG-2 decoding phase and the H.264 encoding phase. This architecture can be easily implemented using MPEG-2 decoder and H.264 encoder without source modifying. However, it has disadvantages in transcoding time and DCT-mismatch problem.

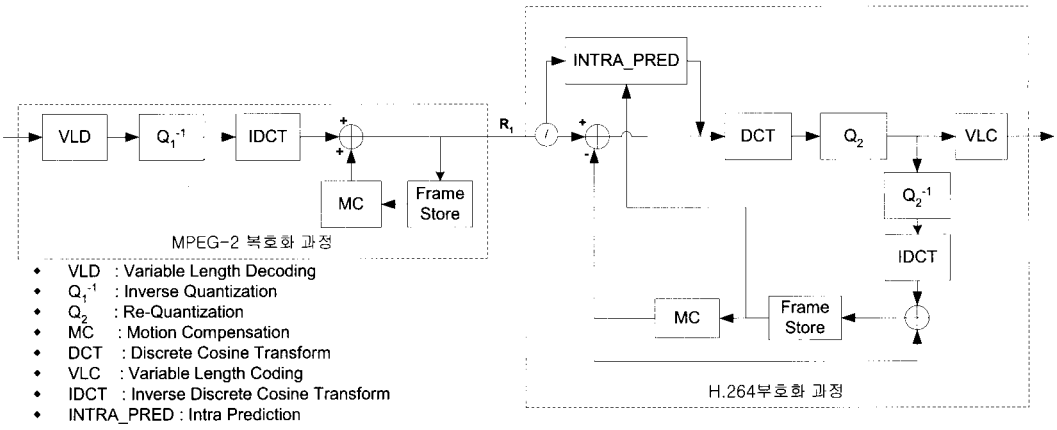
In this paper, we propose two kinds of transcoder architectures, DCT-OPEN and DCT-CLOSED, to complement the CPDT architecture. Although DCT-OPEN has lower PSNR than CPDT due to drift problem, it is efficient for real-time transcoding. On the contrary, the DCT-CLOSED architecture has the advantage of PSNR over CPDT at the cost of transcoding time.

Key Words : Video Transcoder, Transcoding, CPDT, DCT-OPEN, DCT-CLOSED, H.264, MPEG-2

1. 서 론

유·무선 통신 기술의 발전으로 VoIP(Voice on IP), 화상회의, DMB(Digital Multimedia Broadcasting), Wibro(Wireless Broadband)와 같은 고성능 멀티미디어 서비스가 대중화되고 있다. 그 중에서도 위성 DMB의 경우 본격적인 서비스를 제공하고 있으며 전송 비디오 데이터의 압축 표준으로 H.264 baseline을 채택하고 있다. H.264[3]는 비디오 프레임의 압축 효율 향상과 신뢰성있는 전송을 목표로 화상 회의, 화상 전화와 같은 양방향 비디오 통신, 비디오 스트리밍, 브로드캐스팅 등을 주 서비스 대상으로 설계되었다. 또한 [4]의 실험에 의하면 비디오 스트리밍에 대하여 H.264 MP(Main Profile)는 MPEG-2에 비하여 63%, MPEG-3의 ASP (Advanced Simple

※ 본 연구는 정보통신부 정보통신연구진흥원에서 지원하고 있는 정보통신 기초기술연구 지원사업 번호 B1220-0501-0234의 연구결과입니다.
† 준 회 원 : 부산대학교 컴퓨터공학과 병렬멀티미디어연구실 전임연구원
†† 준 회 원 : 부산대학교 대학원 컴퓨터공학과 석사과정
††† 정 회 원 : 부산대학교 정보전산원 조교수
†††† 정 회 원 : 신라대학교 컴퓨터정보공학부 교수
††††† 종신회원 : 부산대학교 전자계산학과 교수
논문접수 : 2005년 8월 9일, 심사완료 : 2005년 9월 4일



(그림 1) MPEG-2/H.264의 CPDT 구조의 예

Profile)에 비하여 37% 정도 압축 효율이 우수한 것으로 나타났다. 그러나 H.264 표준에서는 압축 효율을 높이기 위해 움직임 예측을 위한 가변 블록 크기, 1/4픽셀 지원, 디블로킹 필터링[5], CABAC[6], 4×4 DCT[7], Intra 참조 모드와 같은 새로운 기법 등을 채택하여 기존 비디오 표준과의 호환성이 낮아지게 되었다. 따라서 MPEG-2 방식으로 압축된 콘텐츠를 H.264 방식으로 트랜스코딩하기 위해서 먼저 MPEG-2 비트열을 디코딩한 후, H.264 방식으로 인코딩하는 과정을 수행해야 한다. 이 과정은 디코딩과 인코딩을 반복하는 과정으로 트랜스코딩을 위한 처리 시간이 길어지는 단점이 있다. 이 문제는 비단 DMB 뿐 아니라 다양한 사용자 서비스 환경의 실시간 양방향 통신에서 발생할 수 있는 문제이므로 효율적인 트랜스코딩 기법의 연구가 요구되는 실정이다.

넓은 의미의 트랜스코딩은 코딩된 신호를 다른 신호로 변환하는 것을 의미한다. 그 중에서 비디오 트랜스코딩은 압축된 비디오 데이터를 다른 규격의 비디오로 변환하거나, 비트율, 공간해상도, 프레임율, 오류 탄력성(error-resilience) 등을 조절하는 것을 의미한다[8]. 이 때, 동일 규격 내에서의 트랜스코딩 여부에 따라 동기종 및 이기종 트랜스코딩으로 구분된다. 본 논문에서는 MPEG-2로 압축되어 있는 비디오 컨텐츠를 DMB 서비스를 위해 H.264로 변환하면서 비트율도 함께 조절하는 이기종 트랜스코딩에 초점을 맞춘다.

가장 단순한 구조의 비트율 조절 트랜스코더는 입력 비트열을 완전히 디코딩하여 목표 비트율에 맞게 인코딩하는 과정을 수행하는 구조이다. 이 과정에서 인코딩의 효율을 높이기 위해 움직임 벡터와 매크로블록의 압축 모드를 새롭게 지정할 수 있다[9]. 그러나 이 구조는 인코딩 및 디코딩을 모두 포함하여 트랜스코딩에 소요되는 시간이 높은 단점이 있으므로 입력 비트열의 메타 데이터를 재사용하여 계산량을 줄이는 기법에 대한 연구가 수행되었다[10, 11, 12].

본 논문의 저자는 이미 MPEG-2를 포함한 기존의 동영상 표준과 H.264간의 대표적인 압축 방식의 차이점인 DCT 간 변환(conversion) 기법을 [13]을 통하여 제안한 바 있다. 본 논문에서는 [13]보다 계산량에서 향상된 [1, 2]를 바탕으로 트랜스코딩에 소요되는 시간이 적은 DCT 기반 MPEG-2/H.264 열린 회로형 트랜스코더 구조와 소요시간은 길지만 화질면에서 우수한 DCT 기반 MPEG-2/H.264 닫힌 회로형 트랜스코더 구조를 제안한다.

본 논문의 구성은 다음과 같다. 2장에서는 MPEG-2 비디오 데이터를 H.264 규격으로 트랜스코딩하는 일반적인 구조인 CPDT와 픽셀 기반 열린회로형 트랜스코더 구조(PIXEL-OPEN)를 살펴보고 본 논문에서 제안하는 DCT 기반 트랜스코더에서 이용되는 DCT 변환 기법을 살펴보기로 한다. 3장에서는 제안하는 DCT 기반 MPEG-2/H.264 트랜스코더 구조를 제시하고 4장에서는 실제 비디오 데이터를 이용하여 제안된 구조의 성능을 분석하고 평가한다. 마지막으로 5장에서 결론을 맺고 향후 연구의 방향을 제시한다.

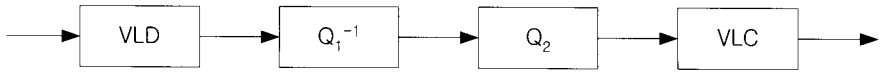
2. MPEG-2/H.264 트랜스코딩 구조와 DCT 변환

2.1 CPDT의 구조

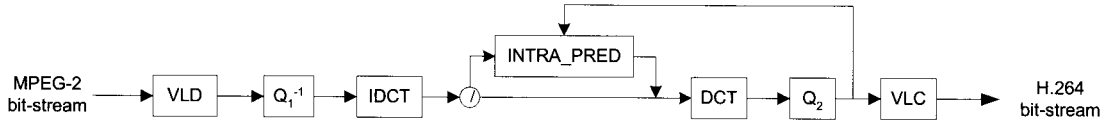
비디오 트랜스코더의 목표는 트랜스코딩을 위한 계산량을 최소화하면서 디코더에서 복원되는 비디오의 화질을 최상으로 유지하는 것이다. 그러나 이 두 요소는 서로 상충되므로 트랜스코딩의 목적에 따라 적절한 선택이 필요하다. (그림 1)에서는 구현이 단순한 픽셀 기반 MPEG-2/H.264 직렬 트랜스코더(CPDT) 구조를 나타내고 있다. 이 구조는 입력 비트열을 디코딩하여 픽셀을 생성하는 단계와 생성된 픽셀(R_1)을 새로운 규격으로 인코딩하는 단계로 구성된다. 이 구조는 이기종 트랜스코딩의 구현이 유리하며 인코딩 과정에서 움직임 벡터, 매크로블록 등의 모드를 새롭게 결정할 수 있어 화질을 높일 수 있다[9]. 그러나 이 구조는 디코더와 인코더를 모두 포함하여 계산량이 높은 단점이 있다. [12, 14, 15]에서는 트랜스코더에 입력되는 비트열의 정보를 재사용하여 계산량을 줄이는 구조를 제안하였다. 현재 위성 DMB를 위한 데이터의 트랜스코딩은 (그림 1)과 같이 MPEG-2의 디코딩과 H.264의 인코딩 과정을 거치게 된다.

2.2 열린회로(open-loop)형 트랜스코더 구조[13]

일반적인 열린 회로형 트랜스코더는 (그림 1)과 달리 디코



(그림 2) 동일 DCT 단위에서의 열린 회로형 트랜스코더 구조



(그림 3) 픽셀기반 MPEG-2/H.264 열린회로형 트랜스코더 구조

딩 과정에서 역움직임 보상(MC)을 수행하지 않고 DCT 단계에서 양자화 과정을 수행하여 비트율을 조절한다. 따라서 (그림 1)의 CPDT에 비하여 트랜스코딩 속도가 빠르다. 이러한 이유는 H.264를 제외한 대부분의 비디오 표준에서 공통적으로 8×8 실수형 DCT를 수행하고 직렬형 구조에서 요구되는 프레임 저장을 위한 메모리나 역DCT가 필요하지 않기 때문이다. (그림 2)는 동일한 단위의 DCT를 사용하는 비디오 표준간의 이기종 트랜스코딩에 적용될 수 있는 열린 회로형 트랜스코더 구조의 예를 보여준다.

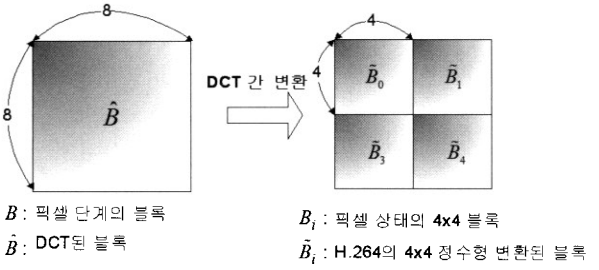
열린회로형 트랜스코더 구조의 단점은 드리프트(drift)를 발생시킨다는 것이다. 드리프트는 트랜스코딩 과정에서 양자화되거나 삭제된 고주파수 데이터로 인하여 인코더에서 코딩을 위해 참조된 프레임의 값과 디코더에서 복원을 위해 참조되는 프레임의 값의 차이로 인해 발생하는 오류(화질의 열화)를 의미한다.

MPEG-2와 H.264 간의 트랜스코딩은 DCT 단위 및 방법의 차이로 인하여 (그림 2)의 열린 회로형 구조가 그대로 적용될 수 없다. 즉, MPEG-2 표준에서는 8×8 실수형 DCT를 수행하지만 H.264에서는 4×4 정수형 DCT를 수행하므로 (그림 2)와 같은 트랜스코딩을 바로 적용할 수 없게 된다는 것이다. 그러므로 MPEG-2와 H.264 간의 열린 회로형 트랜스코더가 동작하기 위해서는 MPEG-2의 8×8 DCT 블록을 4×4 정수형 DCT 블록으로 변환을 해야하는 과정이 필요하다.

(그림 3)은 MPEG-2의 디코딩 단계에서 역DCT를 수행하여 H.264의 특성에 맞도록 DCT를 새롭게 수행하는 구조를 보여주고 있다. 그림에서 'INTRA_PRED'는 입력되는 블록의 모드가 Intra인 경우에 프레임 내 참조를 수행함을 의미한다. H.264의 Intra 모드 블록에서는 MPEG-2와 달리 4×4 또는 16×16 블록 단위로 이웃하는 블록과의 차이 값을 저장하므로 Intra 블록의 경우 Intra 예측 및 보상을 수행한 후 4×4 정수형 DCT를 수행하게 된다.

2.3 DCT 변환

(그림 3)의 구조는 (그림 2)의 구조에 비하여 역DCT와 DCT를 각각 한 번씩 더 수행해야 하는 단점이 있다. 원인은 앞 절에서 밝혔듯이 MPEG-2와 H.264의 DCT 단위와 방식의 차이 때문이다. DCT 기반 트랜스코딩을 위해 두 DCT 블록 간 변환을 위한 연구가 이미 [1, 2, 13]에서 수행되었다. 특히, [13]에서는 H.264의 4×4 실수형 DCT 변환을 수행하고



(그림 4) 8×8 실수형 DCT 블록을 H.264의 4×4 정수형 블록으로 변환하는 예

양자화 단계의 화질 향상을 위한 양자값 변환을 제안하였으며 [1, 2]에서는 H.264의 구현에 이용되는 4×4 정수형 DCT와 일치하는 변환을 수행하는 기법을 연구하였다. [13]에서는 DCT 변환의 계산복잡도 감소를 위해 빠른 DCT 변환을 제안하였다. 구체적인 내용은 다음과 같다.

MPEG-2에서 H.264로의 변환을 위한 DCT 기반 트랜스코딩이 가능하기 위해서는 (그림 4)와 같이 MPEG-2의 8×8 실수형 DCT 변환 기법으로 압축된 비트열을 H.264의 4×4 정수형 DCT 변환 기법으로 압축된 비트열로 변환하는 과정이 필요하다.

(그림 4)에서 B는 픽셀 기반의 8×8 크기의 Intra 또는 Inter 모드 블록을 의미하며 $\hat{B}$은 B에 8×8 실수형 DCT를 수행하여 생성된 블록을 의미한다. 또한 B_i는 B를 4등분한 하위 블록(sub-block)을 의미하며 $\tilde{B}_i$는 B_i를 H.264의 4×4 정수형 DCT 블록을 나타낸다. H.264의 4×4 정수형 DCT 과정을 행렬의 곱으로 나타내면 (식 1)과 같이 표현할 수 있다. [X]는 픽셀기반의 데이터 블록을 의미하며 실제 H.264의 DCT 과정에서는 X의 양쪽으로 변환행렬 C_f와 그 전치행렬인 C_f^T를 곱하는 것으로 끝난다. 연산자 ⊗는 $(C_f X C_f^T)$의 연산 결

$$Y=(C_f X C_f^T)\otimes R_f$$

$$=\begin{bmatrix}1&1&1&1\\2&1&-1&-2\\1&-1&-1&1\\1&-2&2&-1\end{bmatrix}\begin{bmatrix}1&2&1&1\\1&1&-1&-2\\1&-1&-1&2\\1&-2&1&-1\end{bmatrix}\otimes\begin{bmatrix}a^2&\frac{ab}{2}&a^2&\frac{ab}{2}\\\frac{ab}{2}&\frac{b^2}{4}&\frac{ab}{2}&\frac{b^2}{4}\\a^2&\frac{ab}{2}&a^2&\frac{ab}{2}\\\frac{ab}{2}&\frac{b^2}{4}&\frac{ab}{2}&\frac{b^2}{4}\end{bmatrix}\quad \text{(식 1)}$$

$$,a=\frac{1}{2},\quad b=\sqrt{\frac{2}{5}}\approx 0.6325,\quad d=\frac{1}{2}$$

과 행렬의 각 계수와 R의 동일 위치의 계수를 곱하는 연산자이다. H.264에서는 DCT의 결과를 정수화하기 위해 R_f의 각 인자를 양자화단계에 포함한다[16].

최종적으로 생성되는 DCT 변환식은 (식 2)와 같다. 식에서 A와 A^T는 실수형 8×8 DCT에 이용되는 변환행렬을 의미하며, L_i와 R_i는 블록 B에서 B_i를 추출하기 위해 B의 왼쪽과 오른쪽에 각각 곱해지는 필터링 행렬을 의미한다.

$$\tilde{B}_i = (C_f \cdot I_i \cdot A^T) \cdot \hat{B} \cdot (A \cdot R_i \cdot C_f^T) \quad (0 \le i \le 3) \qquad \text{(식 2)}$$

(식 3)~(식 7)은 (식 2)의 유도과정을 보여준다. H.264의 DCT 블록은 (식 3)과 같이 나타낼 수 있으며 (식 3)의 B_i는 B의 부분 블록이므로 B의 양쪽으로 필터링을 수행하는 L_i와 R_i를 이용하여 (식 4)와 같이 표현할 수 있다. (식 4)에서 행렬 곱의 결합법칙을 적용하여 (식 5)와 같이 나타낼 수 있다. DCT의 직교성을 이용하여 (식 5)의 B를 $\hat{B}$의 형식으로 나타내면 (식 6)과 같이 표현되며 다시 행렬 곱셈의 결합법칙을 이용하면 (식 7)과 같이 표현된다. (식 7)의 $(C_f \cdot L_i \cdot A^T)$와 $(A \cdot R_i \cdot C_f^T)$는 이미 정해진 값이므로 계산된 값을 저장하여 사용 가능하다.

$$\tilde{B}_i = C_f \cdot B_i \cdot C_f^T \quad (0 \le i \le 3) \qquad \text{(식 3)}$$

$$= C_f \cdot (L_i \cdot B \cdot R_i) \cdot C_f^T \qquad \text{(식 4)}$$

$$= (C_f \cdot L_i) \cdot B \cdot (R_i \cdot C_f^T) \qquad \text{(식 5)}$$

$$\leftarrow (\hat{B} = A \cdot B \cdot A^T \Rightarrow B = A^T \cdot \hat{B} \cdot A)$$

$$= (C_f \cdot L_i) \cdot (A^T \cdot \hat{B} \cdot A) \cdot (R_i \cdot C_f^T) \qquad \text{(식 6)}$$

$$= (C_f \cdot L_i \cdot A^T) \cdot \hat{B} \cdot (A \cdot R_i \cdot C_f^T) \qquad \text{(식 7)}$$

그러나 (식 2)의 경우 4개의 부분 블록(B_i)에 대해 각각 작업을 수행하므로 일부 중간 값을 이용하더라도 계산량이 높다. [2]에서는 (식 7)의 과정을 한 개의 변환 행렬로 구성하여 계산량을 줄였으며 빠른 DCT와 유사한 방식으로 빠른 DCT 변환 기법을 제시하여 계산량을 최소화하였다. 본 논문에서는 이 [2]에서 제안한 빠른 변환 방법을 이용하여 DCT 변환을 수행한다.

3. 제안하는 트랜스코더 구조

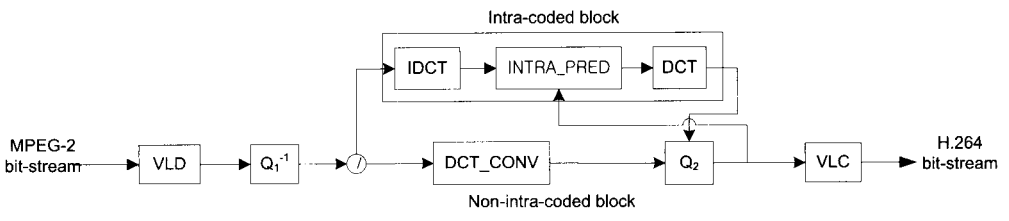
본 논문에서는 앞 절에서 살펴본 DCT 변환을 이용하여 DCT 기반의 열린 회로형 구조와 닫힌 회로형 구조를 제안한다.

3.1 DCT 기반 MPEG-2/H.264 열린 회로형 트랜스코더 구조 (DCT-OPEN)

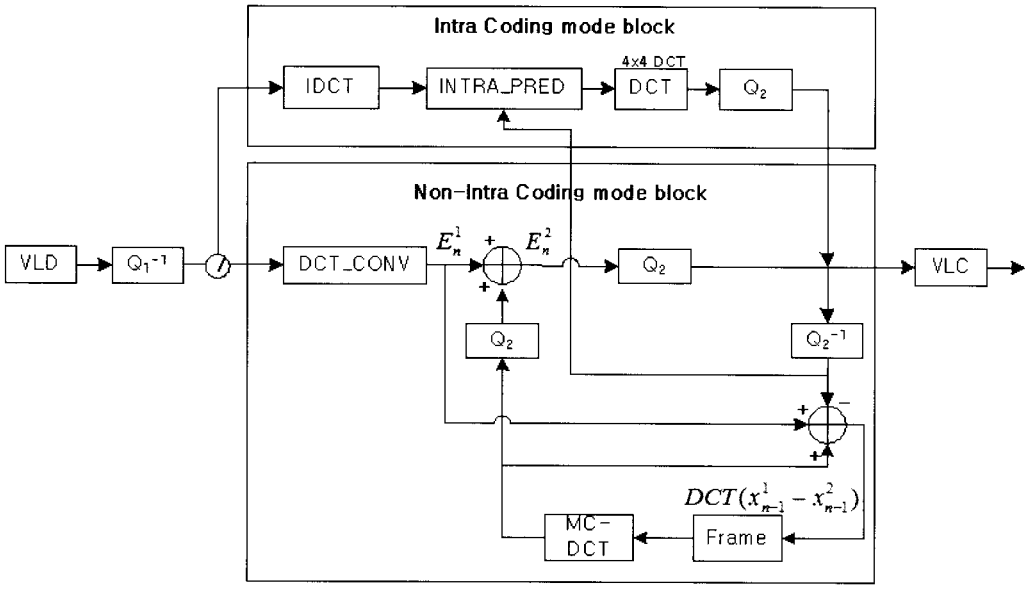
간단한 구조의 MPEG-2/H.264 트랜스코더 구조는 (그림 5)와 같은 열린 회로형 구조로 나타낼 수 있다. H.264의 압축 과정에서는 Intra 모드와 Non-intra 모드의 경우 압축 방식이 다르다. Intra 모드 블록의 경우 기존의 비디오 압축 표준과 달리 해당 블록의 이웃한 블록과의 차이값을 구하는 Intra 예측을 수행한다. 이 과정은 H.264에 채용된 새로운 기법이므로 MPEG-2 규격으로 압축된 데이터를 H.264 규격으로 트랜스코딩하기 위해서는 추가적으로 수행해야 하는 과정이다. 그러나 DCT 상에서 Intra 예측을 수행하는 것은 행렬 연산의 곱을 추가적으로 수행하는 것이므로 픽셀 상태에서 수행하는 것에 비하여 계산의 복잡도가 매우 높다. 즉, DCT 상에서 최상의 블록 모드를 선택하기 위해 각 모드별로 이웃 블록에 DCT를 수행하는 것은 계산량이 너무 높아지는 결과를 초래하게 된다. 따라서 본 논문에서는 Intra 블록에 대해서는 픽셀 기반 트랜스코딩을 수행하도록 한다. (그림 5)의 'INTRA_PRED'에서 이웃한 블록의 역양자화와 역DCT를 수행하여 Intra 예측을 수행하게 된다. 또한, Non-intra 블록의 경우 DCT 변환을 수행하여 DCT 기반 트랜스코딩을 수행한다. 이때, 'DCT_CONV'에서 MPEG-2의 8×8 실수형 DCT 블록을 4개의 H.264의 4×4 정수형 블록으로 변환한다. DCT 변환은 (식 2)를 이용하여 수행된다.

3.2 DCT 기반 닫힌 회로형 트랜스코더 구조(DCT-CLOSED)

(그림 3)과 (그림 5)는 MPEG-2 인코딩에 사용되는 참조 블록의 값과 H.264 디코딩에 사용되는 참조 블록의 값의 차이로 인하여 드리프트가 발생하는 트랜스코더 구조이다. 즉, 트랜스코딩시 새로운 양자화(Q_2)를 수행하여 계수 값이 변하는 이전 프레임을 다음 프레임에서 그대로 참조하여 H.264 디코딩을 수행하기 때문이다. 그러나 (그림 1)의 직렬형 구조에서는 디코딩 후 새로운 인코딩 과정을 수행하므로 드리프트가 발생하지 않는다. 본 절에서는 DCT 상에서 양자화로 인한 오차를 반영하여 드리프트 발생을 최소화하는 DCT 기반 닫힌 회로형 구조를 (그림 6)과 같이 제안한다. 전체 트랜스코딩 구조는 (그림 5)의 DCT-OPEN 구조를 포함하고 있으며 양자화로 인한 참조블록의 차이값을 반영하기 위해 오류값을 보정하는 부분이 추가되었다. 참조 블록의 변화 반영을 위해 [17]에서 증명한 $E_n^2 = E_n^1 + MC(DCT(x_{n-1}^1 - x_{n-1}^2))$가 되도록 설계하였다. 이때, E_n^1은 현재 트랜스코더에 입력된 DCT 상태의 블록을 의미하며 E_n^2는 E_n^1이 참조하는 이전 프



(그림 5) DCT 기반 MPEG-2/H.264 열린 회로형 트랜스코더 구조



(그림 6) DCT 기반 MPEG-2/H.264 닫힌 회로형 구조

레임의 참조블록의 변화를 반영한 값을 의미한다. 또한, $MC(DCT(x_{n-1}^1 - x_{n-1}^2))$는 이전 프레임의 트랜스코딩 입력값과 디코딩 후 값의 차이를 DCT 상태에서 계산하여 저장한 후, 현재 프레임에서 참조하는 블록의 값을 반영함을 의미한다.

3.2.1 Intra 블록

H.264의 Intra 블록은 프레임 내의 이웃한 블록의 계수 값을 이용하여 Intra 예측을 수행하고 그 중에서 압축 효율이 높은 예측 모드를 이용하여 압축을 수행한다. 이 과정은 기존의 비디오 압축 표준에서 제공하지 않는 기능으로 H.264로의 트랜스코딩시 추가적으로 수행해야 한다. 그러므로 DCT 상에서 Intra 예측을 수행하려면 Intra 예측의 각 모드마다 이웃 블록의 값을 모드에 맞게 DCT 상태로 변환(transform)해야 한다. 이 과정은 MPEG-2의 8×8 DCT 블록을 원하는 형태의 값으로 변환하는 과정이므로 계산량이 매우 높은 과정이다. 이 문제를 해결하기 위해서는 DCT 상에서의 모드 결정을 위한 통계적 기법 등을 추가해야 하지만 픽셀 블록과의 특성이 전혀 달라 통계적 기법을 적용하기 어려운 실정이다. 따라서, 본 논문에서는 DCT-OPEN에서의 처리와 동일하게 픽셀기반 Intra 예측을 수행하여 계산량 증가를 줄이기로 한다. 또한, Intra 블록의 경우 이전 프레임을 참조하는 것이 아니라 현재 프레임 내 이웃한 블록을 참조하므로 보정 단계가 요구되지 않는다. 다만 'INTRA_PRED'에서 이웃 블록의 값을 참조하기 위한 처리 과정이 수행되어야 한다.

3.2.2 Non-intra 블록

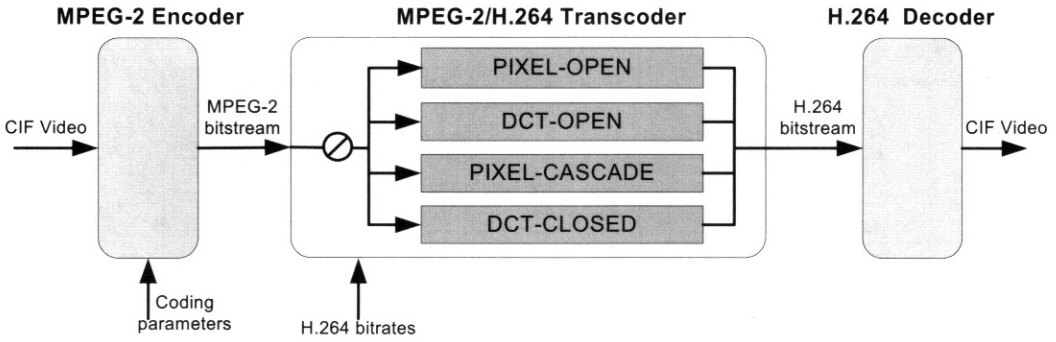
열린 회로형 구조에서는 시간이 흐를수록 트랜스코딩으로 인한 오류가 누적된다. 그러므로 각 프레임 별 오류 보정을 위해 트랜스코더의 입력 데이터를 복원한 값과 출력 데이터를 복원한 값의 차이를 저장하여 다음 프레임에서 참조하는 블록이 갱신하도록 해야 한다. 이를 위해 입력 블록을 DCT 상에서 복호화한 값을 구하기 위해 'DCT_CONV'에서 H.264의 DCT 블록으로 출력되는 값을 역양자화(Q_2^{-1})한 값과의 차이를 구하다. 이때, 버퍼에 저장되어 있는 이전 프레임의 누적 오류 값을 MC-DCT하여 차이값 보정에 반영한다. [17]에서 밝혔듯이 DCT상에서의 보정은 이론상 정확하지만 컴퓨터상에서의 처리시 실수처리 등의 문제로 미미한 오류가 발생할 수 있다.

4. 구현 및 실험 결과

제안된 DCT-OPEN과 DCT-CLOSED의 성능을 평가하기 위해 PIXEL-OPEN과 PIXEL-CASCADE 구조를 구현하여 PSNR, 실제 이미지, 계산복잡도 측면에서의 성능을 분석하였다.

4.1 실험 환경 및 방법

(그림 7)은 성능 평가를 위해 구현된 트랜스코더 시스템의 구조를 나타낸다. 그림에서 MPEG-2 인코더는 MPEG-2 Test Model(TM5)의 인코더[18]를 실험 환경에 맞게 수정한 것이며 MPEG-2/H.264 트랜스코더는 MPEG-2 TM5의 디코더와 H.264의 Joint Model 8(JM8) 인코더[19]를 수정하여 구현하였다. (그림 7)의 트랜스코더에서 PIXEL-OPEN은 (그림 3), DCT-OPEN은 (그림 5), PIXEL-CASCADE는 (그림 1), DCT-CLOSED는 (그림 6)의 구조를 각각 구현한 것이다. 실험에 사용된 디코더, 인코더, 트랜스코더는 C로 작성되어 Windows XP 운영체제하에서 Visual C++6.0으로 컴파일되고 실험되었다. DCT-OPEN과 DCT-CLOSED 구조의 Inter 모드 매크로블록에서는 MPEG-2에서 생성된 움직임 벡터를 그대로 이용하므로 동일한 조건에서의 실험을 위해 PIXEL-OPEN, PIXEL-CASCADE 구조에서도 MPEG-2의 움직임 벡터를 그대로 이용하도록 구현하였다.



(그림 7) 실험을 위한 트랜스코더 시스템의 구조

〈표 1〉 MPEG-2 인코더의 입력 데이터의 특징

인자		설명	
비디오 포맷		CIF(352×288)	
컬러 샘플링		4:2:0	
비디오	이름	Mobile	Tempete
	움직임 정도	보통	비교적 낮음

〈표 2〉 MPEG-2 부호화를 위한 인자

인자	설명	인자	설명
GOP(N,M)	(15,1)	비트율	50~400 KB/S
프레임율(fps)	30	양자화 행렬	default
Profile/Level	Main/Main	1/2화소 지원	지원하지 않음
총 프레임 수	200	예측 모드	프레임

실험에 사용된 데이터의 특징은 <표 1>과 같다.

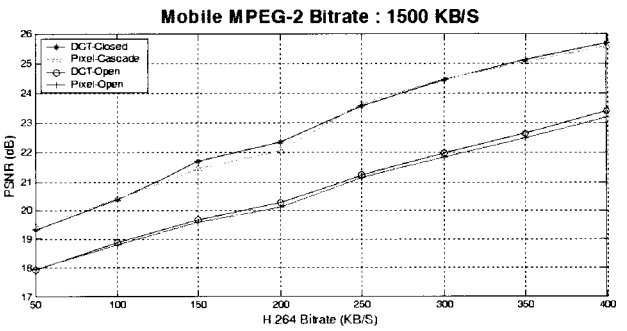
전체 실험 과정은 (그림 7)과 같이 MPEG-2의 인코딩, 트랜스코딩, H.264 디코딩 과정으로 구성되며 MPEG-2 인코더에 사용된 인자는 다음과 같다(<표 2> 참조). 각 프레임의 부호화 모드를 결정하는 GOP(Group of Pictures)의 크기와 I, P 프레임 간격을 N=15와 M=1로 지정하여 'IPPPP…' 형식으로 압축되도록 지정하였다. 이 경우 GOP내의 P 프레임은 직전에 압축되어 디코딩된 프레임을 참조하므로 오류의 누적을 확인하기 편리하다. 또한 비트율에 따른 PSNR을 비교하기 위해 MPEG-2 인코더에서는 TM5의 비트율 조절 기법을 그대로 적용하여 초당 50KB~1500KB까지 다양하게 실험하였다. H.264의 경우 rate control을 위한 양자값(Qstep)을 결정하는 시점이 DCT 전 단계이므로 DCT 기반 트랜스코더에서 기존 H.264의 비트율 조절 기법을 그대로 적용할 수 없다. 본 논문에서는 DCT 상에서 프레임 단위로 Qstep을 결정하기 위한 통계적인 정보를 이용하여 비트율 조절을 수행하였다. 이 경우 생성되는 비트열 크기가 초당 400KB 이내가 되도록 실험하였다.

4.2 화질의 비교

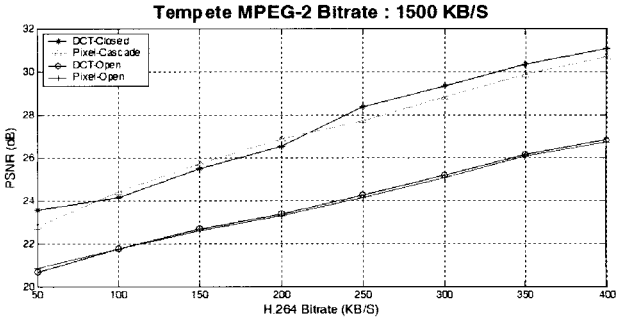
본 절에서는 비트율 변화에 따른 비디오 화질을 PSNR(Peak Signal to Noise Ratio), 실제 이미지로 비교하기로 한다.

(그림 8)은 비트율 변화에 따른 Mobile 비디오의 PSNR을 비교한 것이다. 그림에서 DCT-CLOSED와 PIXEL-CASCADE의 성능이 PIXEL-OPEN, DCT-OPEN에 비하여 평균 2dB 이상 높은 것으로 나타났으며 H.264의 압축 비트율이 높아질수록 간격이 더 커짐을 알 수 있다. DCT-CLOSED의 성능이 PIXEL-CASCADE보다 0.3dB 정도 높거나 거의 유사함을 알 수 있는데 이는 MPEG-2의 비트율보다 상대적으로 낮은 비트율로 압축될 때, 역DCT와 DCT를 두 번 수행하는 것보다 DCT 변환을 수행하는 것이 화질에 영향을 미친 것으로 분석된다. 이것은 DCT-OPEN과 PIXEL-OPEN 간의 PSNR 관계에도 동일하게 나타나는 현상이다.

(그림 9)는 Tempete 비디오에 대해 비트율 변화에 따른 화질 변화를 측정한 것이다. Mobile에 비하여 상대적으로 비디오 내 객체의 움직임이 적은 Tempete의 경우에도 Mobile과 거의 동일한 결과를 볼 수 있다.



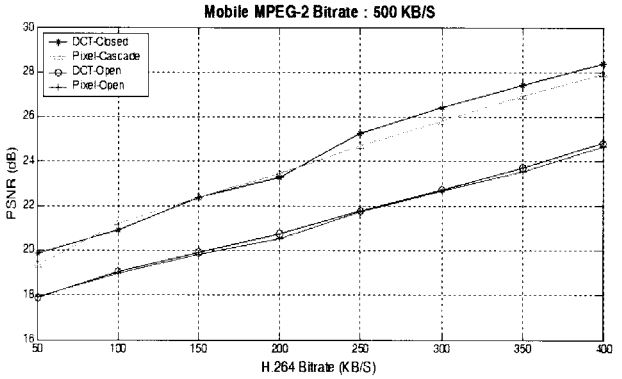
(그림 8) MPEG-2 압축 비트율이 1500KB/S인 경우 Mobile의 트랜스코딩 비트율 변화에 따른 PSNR 변화



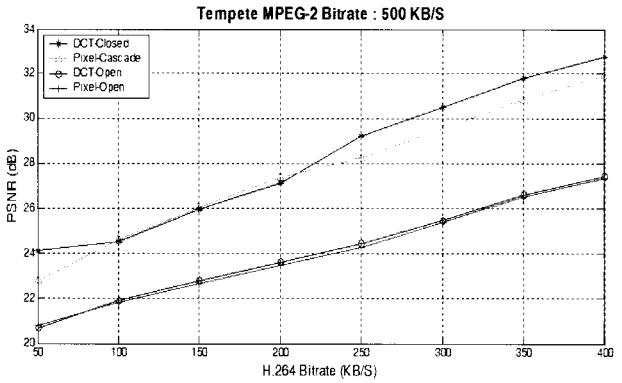
(그림 9) MPEG-2 압축 비트율이 1500KB/S인 경우 Tempete의 트랜스코딩 비트율 변화에 따른 PSNR 변화

(그림 10)과 (그림 11)은 MPEG-2의 비트율이 500KB/S일 때 Mobile과 Tempete의 화질을 비교한 것이다. DCT-CLOSED의 경우 MPEG-2의 압축 비트율이 1500KB/S일 경우보다 최대 1dB까지 화질이 우수함을 알 수 있으며 DCT-OPEN의 경우 압축 비트율에 관계없이 거의 동일한 결과를 보여줌을 알 수 있다.

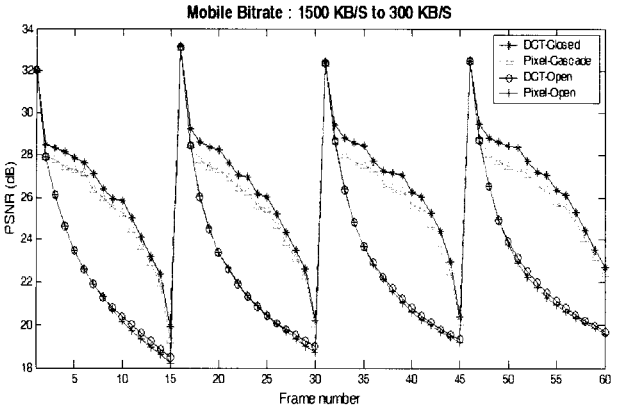
(그림 12)와 (그림 13)은 Mobile과 Tempete에 대하여 각각 MPEG-2 압축 비트율을 1500KB/S, 500KB/S, 트랜스코딩 비트율을 300KB/S, 100KB/S로 설정하여 비디오의 각 프레임의 최종 PSNR을 비교한 것이다. 결과는 이전 그림에서 살펴본 바와 같이 DCT-CLOSED의 화질이 우수한 것으로 나타났으며 비디오의 움직임이 상대적으로 높은 Mobile의 PSNR 감소율이 Tempete에 비하여 높은 것으로 나타났다.



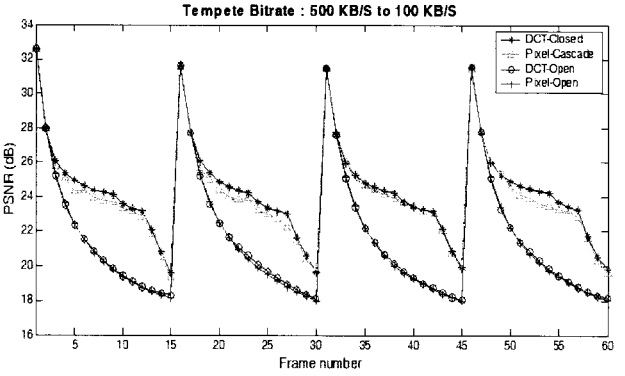
(그림 10) MPEG-2 압축 비트율이 500KB/S인 경우 Mobile의 트랜스코딩 비트율 변화에 따른 PSNR 변화



(그림 11) MPEG-2 압축 비트율이 500KB/S인 경우 Tempete의 트랜스코딩 비트율 변화에 따른 PSNR 변화



(그림 12) MPEG-2 압축 비트율: 1500KB/S, 트랜스코딩 비트율: 300KB/S인 경우의 Mobile 동영상의 프레임 별 PSNR 비교



(그림 13) MPEG-2 압축 비트율: 500KB/S, 트랜스코딩 비트율: 100KB/S인 경우의 Tempete 동영상의 프레임 별 PSNR 비교

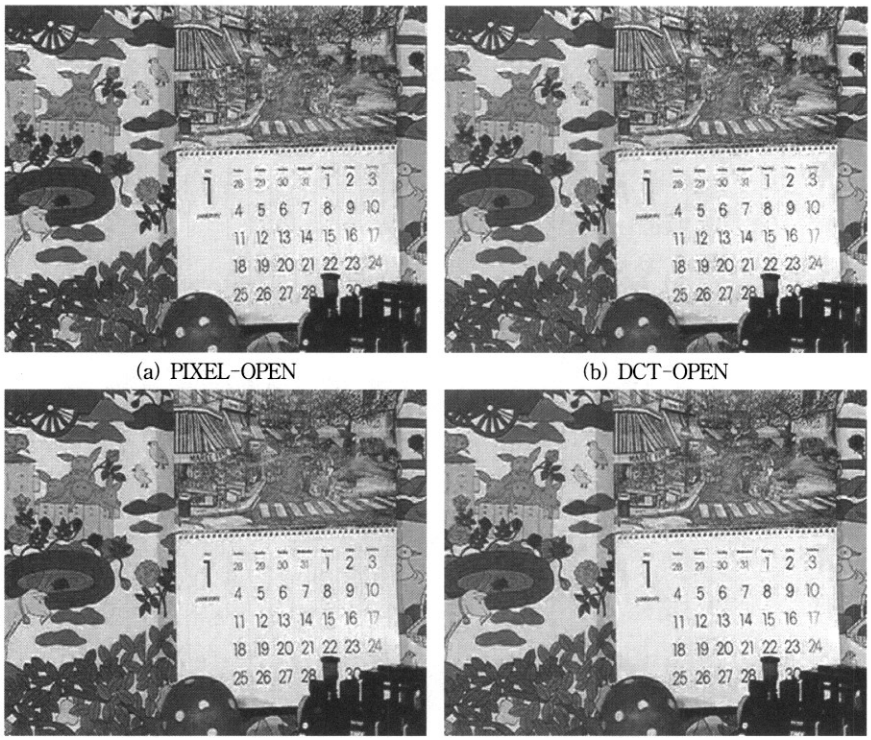
(그림 14)는 각 구조별로 MPEG-2 압축 비트율이 1500KB/S이고 트랜스코딩 비트율이 300KB/S인 Mobile 비디오의 18번째 프레임의 이미지를 나타낸 것이다. 그림을 보면 DCT-OPEN의 경우 PIXEL-OPEN과 거의 유사하지만 각 그림의 외곽선이 약간 더 선명함을 알 수 있다. DCT-CLOSED와 PIXEL-CASCADE의 경우 DCT-OPEN, PIXEL-OPEN에 비하여 전체적으로 번짐 현상이 향상되어 달력의 숫자 부분과 배경화면이 훨씬 선명하게 나타남을 알 수 있다. 또한, DCT-CLOSED의 경우 PIXEL-CASCADE에 비하여 이미지 내의 각 객체의 외곽선이 선명하게 나타남을 볼 수 있다.

4.3 계산 복잡도 비교

본 절에서는 PIXEL-OPEN, DCT-OPEN, PIXEL-CASCADE, DCT-CLOSED 구조의 계산 복잡도를 곱셈과 덧셈 연산 횟수 위주로 비교 분석한다. 연산 횟수의 비교를 단순화하기 위해 각 트랜스코더에서 공통으로 수행하는 역 엔트로피코딩(VLD), 엔트로피코딩(VLC)을 비교 대상에서 제외한다. Intra 모드 블록의 경우 모든 구조에서 유사한 트랜스코딩 과정을 거치므로 본 논문에서는 비교대상에서 제외하고 Non-intra 블록의 계산량을 비교하기로 한다.

DCT와 역DCT의 경우 Chen의 빠른 DCT[20] 기법을 사용한다고 가정하며 DCT 변환의 경우 [2]에서 제안한 기법을 사용한다. <표 3>은 실험에 사용된 각 구조에서 트랜스코딩을 수행할 때 8×8 블록을 트랜스코딩하기 위해 사용되는 연산자 수를 각각 비교한 것이다.

표에서 DCT-OPEN이 PIXEL-OPEN에 비하여 덧셈 연산 수에서 약간 유리한 것으로 나타났다. DCT-CLOSED와 PIXEL-CASCADE의 경우 DCT-CLSOED의 곱셈 연산자 수가 PIXEL-CASCADE에 비하여 약 2배 정도 많다. 이러한 DCT-CLOSED에서 수행되는 MC-DCT에서 행렬 곱셈 연산



(a) PIXEL-OPEN (b) DCT-OPEN

(c) PIXEL-CASCADE (d) DCT-CLOSED

(그림 14) MPEG-2 압축 비트율: 1500KB/S, 트랜스코딩 비트율: 300KB/S인 경우 Mobile 비디오의 18번째 이미지의 화질 비교

이 수행되기 때문이다. 그러나 <표 3>은 8×8 블록 1개를 트랜스코딩할 때 발생하는 연산의 횟수이므로 한 프레임을 트랜스코딩할 때 PIXEL-CASCADE와 DCT-CLOSED 간 차이는 크게 줄어들게 된다. 왜냐하면 특정 프레임을 압축할 때 많은 부분이 Intra 모드로 압축되기 때문이다.

〈표 3〉 각 구조별 8×8 블록 트랜스코딩 계산 복잡도

트랜스코더 구조	Non-intra 블록의 계산 복잡도		
	곱셈	덧셈	시프트
PIXEL-OPEN	448	800	256
DCT-OPEN	448	736	192
PIXEL-CASCADE	512	1184	384
DCT-CLOSED	1,088	1,360	320

5. 결 론

본 논문에서는 DMB 서비스를 위해 이미 MPEG-2로 압축되어 있는 비디오 컨텐츠를 H.264 규격으로 트랜스코딩하기 위한 트랜스코더 구조를 제안하였다. 제안된 구조 중 DCT-OPEN은 계산복잡도가 낮아 실시간 처리에 유리하며 DCT-CLOSED는 계산량은 높으나 화질에서 유리한 구조이다. DCT-OPEN의 경우 양자화로 인한 참조 블록의 변화를 반영하지 않으므로 드리프트 현상이 발생하게 되며 DCT-CLOSED에서는 보정을 수행하여 드리프트 발생 원인을 제거하였다. 계산량 측면에서 DCT-CLOSED의 경우 MC-DCT 수행으로 인해 불리한 것으로 나타났으나 프레임 전체적인 처리 시간은 PIXEL-CASCADE와 유사할 것으로 예상된다. 향후 연구 과제로는 DCT 기반 트랜스코딩의 화질 향상을 위해 비트율 제어 알고리즘을 연구하여 제안된 구조에 적용하는 것이다.

참 고 문 헌

[1] 강진미, "MPEG-2에서 H.264/AVC로의 변환을 위한 DCT 기반 트랜스코더 구조", 공학석사 학위논문, 부산대학교 대학원 컴퓨터공학과, 2005년 2월.

[2] Jun Xin, Anthony Vetro, Huifang Sun, "Converting DCT Coefficients to H.264/AVC Transform Coefficients," PCM 2004: pp.939-946, Nov., 2004.

[3] SO/IEC 14496-10:2003, Coding of Audiovisual Objects -Part 10: Advanced Video coding. 2003 and ITU-T Recommendation H.264: Advanced video coding for generic audiovisual services.

[4] J. Ostermann, J. Bormans, P. List, D. Marpe, M. Narroschke, F. Pereira, T. Stockhammer, and T. Wedi, "Video Coding with H.264/AVC: Tools, Performance. and Complexity," IEEE Circuits and Systems. Magazine, Vol.4, pp.7-28, 2004.

[5] P. List, A. Joch, J. Lainema, G. Bjontegaard and M. Karczewicz "Adaptive deblocking filter," IEEE Transactions on Contents on Circuits and Systems for Video Technology, Vol.13, pp.614-619, Jul., 2003.

[6] D. Marpe, T. Wiegand and H. Schwarz, "Context-Based Adaptive Binary Arithmetic Coding in the H.264/AVC Video Compression Standard," IEEE Transactions on Contents on Circuits and Systems for Video Technology, Vol.13, No.7, pp.620-636, Jul., 2003.

[7] H. Malvar, A. Hallapuro, M. Karczewicz, and L. Kerofsky, "Low-complexity transform and quantization in H.264/AVC," IEEE Transactions on Contents on Circuits and Systems for Video Technology, Vol.13, No.7, pp.598-603, Jul., 2003.

[8] A. Vetro, C. Christopoulos and H. Sun "An overview of transcoding architectures and techniques," IEEE Signal Processing Magazine, pp.18-29, Mar., 2003.

[9] H. Sun, W. Kwok and J. Zdepski, "Architectures for MPEG compressed bitstream scaling," IEEE Transaction On Circuits and Systems for Video Technology, Vol.6, pp.191-199, Apr., 1996.

[10] Y. Nakajima, H. Hori and T. Kanoh, "Rate conversion of MPEG coded video by requantization process," Proceeding of IEEE International Conference of Image Processing, pp.408-411, 1995.

[11] A. Eleftheriadis, "Dynamic rate shaping of compressed digital video," Ph.D. dissertation, Dept. Elec. Eng., Columbia Univ., New York, Jun., 1995.

[12] N. Bjork and C. Christopoulos, "Transcoder architectures for video coding," IEEE Transactions on Consumer Electronics, Vol.44, pp.88-98, Feb., 1998.

[13] Joo-Kyong Lee and Ki-Dong Chung, "Quantization/DCT Conversion Scheme for DCT-Domain MPEG-2 to H.264/AVC Transcoding," IEICE Trans. Commun., Vol. E88-B, No.7, Jul., 2005.

[14] P. Assuncno and M.Ghanbari, "Post-processing of MPEG-2 coded video for transmission at lower bit-rates," Proceeding of IEEE international conference of Acoustics, Speech and Signal Processing, pp.1998-2001, 1996.

[15] G. Kessman, R. Hellinghuizen, F. Hoeksma and G. Heidman, "Transcoding of MPEG bitstreams", signal processing : Image Communications, Vol.8, No.6, pp. 481-500, Sept. 1996.

[16] I. E.G. Richardson, 'H.264 and MPEG-4 Video Compression : Video Coding for Next-generation Multimedia', Wiley, 2003.

[17] P. Assuncno and M.Ghanbari, "A Frequency-Domain Video Transcoder for Dynamic Bit-Rate Reduction of MPEG-2 Bit Streams," IEEE Transactions on Circuits and Systems for Video Technology, Vol.8, No.8, Dec., 1998.

[18] http://www.mpeg.org/MPEG/MSSG/tm5/

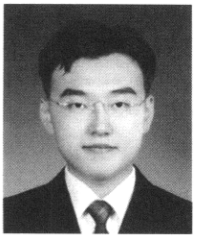
[19] http://iphome.hhi.de/suehring/tml/download/

[20] W.H. Chen, C.H. Smith, S.C. Fralick, "A Fast Computation Algorithm for The Discrete Cosine Transform," IEEE Trans. Commun., Vol.COM-25, pp.1004-1009, 1977.

이 주 경

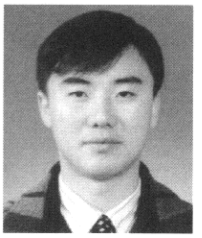
e-mail : jklee@melon.cs.pusan.ac.kr
1996년 부산대학교 전자계산학과(학사)
1998년 부산대학교 전자계산학과 (이학석사)
1998년~2001년 한국전력공사 근무
2005년 부산대학교 대학원 전자계산학과 (이학박사)
관심분야 : 멀티미디어 데이터 압축, 오류제어, 임베디드 시스템

권 순 영



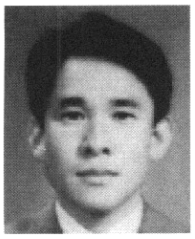
e-mail : ksy2020@melon.cs.pusan.ac.kr
2004년 부산대학교 전자전기정보컴퓨터공학부(학사)
2004년~현재 부산대학교 컴퓨터공학과 석사과정
관심분야 : 멀티미디어 데이터 압축, 임베디드 시스템

박 성 호



e-mail : shpark@cs.pusan.ac.kr
1996년 부산대학교 전자계산학과(학사)
1998년 부산대학교 전자계산학과 (이학석사)
2002년 부산대학교 전자계산학과 (이학박사)
2002년~현재 부산대학교 정보전산원 조교수
관심분야 : 멀티미디어 통신, 인터넷 캐싱, IP Telephony

김 영 주



e-mail : yjkim@silla.ac.kr
1988년 부산대학교 계산통계학과 (이학사)
1990년 부산대학교 계산통계학과 (이학석사)
1990년~1995년 큐닉스컴퓨터 응용시스템 연구소 근무
1999년 부산대학교 전자계산학과(이학박사)
2000년~현재 신라대학교 컴퓨터정보공학부 교수
관심분야 : 멀티미디어, 병렬파일시스템, 분산처리

정 기 동

e-mail : kdchung@pusan.ac.kr
1973년 서울대학교(학사)
1975년 서울대학교 대학원(석사)
1986년 서울대학교 대학원 계산통계학과 (이학박사)
1990년~1991년 MIT, South Carolina 대학 교환 교수
1995년~1997년 부산대학교 전자계산소 소장
1978년~현재 부산대학교 전자계산학과 교수
1997년~현재 부산대학교 대학원 멀티미디어 협동과정 교수
관심분야 : 병렬처리, 멀티미디어, 임베디드시스템