특집논문-01-6-2-01

영상 전송을 위한 효율적인 DCT 영역의 트랜스코딩

김 성 진*, 황 인 경*, 정 웅 찬*, 백 준 기*, 김 제 우**, 송 혁*, 백 종 호**

Efficient DCT Domain Transcoding for Video Transmission

Sung Jin Kim*, In Kyung Hwang*, Woong Chan Joung*, Joon Ki Paik*, Je Woo Kim**, Hyok Song** and Jong Ho Paik**

요 약

본 논문에서는 적응적 영상 전송을 위한 효율적인 DCT 영역 비디오 트랜스코딩 알고리듬을 제안한다. 비디오 트랜스코딩은 압축된 비트 스트림을 새로운 전송 대역폭의 제한 조건에 맞춰서 전송의 유연성을 획득하기 위한 기술이다. 이 과정에서 부호화(encoder)와 복호화(decoder)를 거치면서 참조 영상의 차이로 인한 드리프트(drift) 오류가 발생하게 된다. 이러한 문제점을 피하기 위해 비디오 트랜스코딩 방식으로 CPDT(Cascade Pixel-Domain Transcoder)구조를 사용하지만, 이 구조는 많은 계산량을 필요로 하고 구조가 복잡하다는 단점을 가진다. 따라서 본 논문에서는 효율적 비디오 트랜스코딩을 위해 DCT 영역에서 트랜스코딩을 행하는 CDDT(Cascade DCT-Domain Transcoder) 구조를 제안한다. CDDT 구조는 DCT 영역에서 움직임 보상과 부 표본화를 행한다.

Abstract

We propose an efficient DCT-domain video transcoding algorithm for flexible bit-rate video communications. Video transcoding provides communication flexibility by adaptively changing the bit-rate of compressed bit stream. During the transcoding process, adrift error is unavoidable because of the difference between reference images in the series of encoding and decoding. For solving the drift error problem, cascade pixel-domain transcoder (CPDT) has been proposed. CPDT, however, requires highly complex hardware and heavy computational overhead. In this paper we propose a DCT-domain transcoding technique, which enables efficient transcoding without any drift error. The proposed cascade DCT-domain transcoder (CDDT) is realized by new motion compensation and down-sampling methods in the DCT-domain.

I. 서 론

최근들어 화상회의, 주문형 비디오(VOD), 원거리 학습 등과 같은 다양한 멀티미디어 서비스들이 개발되었다. 그러나 이러한 서비스 제공업체와 사용자간의 통일된 네트워크 구조를 구축하지 못하고 있다. 예를 들어, 제공자는 ATM망을 사용자는 PSTN이나 DSL망을 이용하는 경우가 이에 속한다. 이러한 연결 상태는 멀티미디어 서비스를 전송하는 데 있어서 이종네트워크 연결의 구조로 입력되는 비트스트림을 비디오 출력단의 대역폭에 적합한 변환을 필요로 한다. 이런 방식으로 현재 비디오 압축 표준안에는 스케일러블 코딩(Scalable Coding)을 제안하고 있다[1]. 그러나 스케일러블 코딩은 향상계층(Enhancement Layer)의 제한으로 인해 비디오의 화질을 몇 단계로만 표시가 가능하다는 단점이 있어서 송신단과 수신단에 있어서의 더 좋은 화질을 얻는 것이 불가능하다. 따라서 더 좋은 화질을 얻기 위해서 전송하기 전에 디코딩 과정을 거쳐 영상을 재구성한 후 양자화 값을 변화시켜서 수신측에 적합

* 중앙대학교 첨단영상대학원 영상공학과 디지털 영상처리연구실
IP Lab. Dept. of Image Eng., GSAIM, Chung-Ang Univ
** 전자부품연구원 뉴미디어통신연구센터
Korea Electronics Technology Institute (KETI)
※ 이 연구는 한국전자부품연구원의 연구비 지원에 의해서 수행되었습니다

한 비트스트림의 크기로 변환하는 과정이 필요하다. 이러한 과정을 트랜스코딩의 과정이라고 말한다.

비디오 트랜스코딩의 방법으로 여러 가지 고속의 구조들이 제안되었지만, 많은 트랜스코딩 구조들은 드리프트 오류(drift error)라는 열화로 인해 화질의 저하를 발생시킨다. 드리프트 오류는 인코딩 과정에서 움직임 보상을 위해 재구성한 화면과 디코딩 과정에서 재구성한 화면이 같지 않을 때 발생한다[2]. 초기에 재구성한 화면의 드리프트 오류가 작더라도, 이것을 통해 구성된 화면이 다시 다음 프레임을 예측하는데 사용이 되므로 오류가 점점 누적되게 된다.

드리프트 오류를 억제하기 위해 가장 널리 쓰이는 구조로 CPDT(Cascade Pixel-Domain Transcoder)를 들 수 있다[2][3]. 이 구조는 부호화되어 있는 비트스트림을 복호화한 다음 사용자에게 적합한 비트율로 재인코딩하는 구조이다. CPDT의 장점으로 디코딩 후에 재구성된 영상 데이터를 가지고 다시 인코딩할 수 있기 때문에 가장 고 화질의 영상 데이터를 제공할 수 있으며 또한 모든 종류의 트랜스코딩에 사용할 수 있다. 반면에 기본적인 디코딩 과정과 인코딩 과정을 모두 포함하고 있으므로 복잡도가 높고 계산량이 많으며 지연이 생기는 단점이 있다.

본 논문에서는 CDDT(Cascade DCT-Domain Transcoder) 구조를 이용해 CPDT의 단점을 보완해 효율적인 트랜스코딩의 방법에 대해서 설명한다[4]. CDDT 구조는 DCT 영역에서 움직임 보상, 반화소 정밀도 구현 등을 통해서 CPDT에서의 복잡도를 줄일 수 있다. 본 논문의 전반부에서는 CPDT와 CDDT의 구조와 DCT 영역에서의 처리 방법에 대해서 설명하고, 후반부에서는 실험 결과를 제시한다.

Ⅱ. CPDT

트랜스코딩의 가장 간단한 구조로 OPLT(Open Loop Transcoder)가 있다[2]. 그림 1은 OPLT 구조를 나타낸다. 이 구조는 지금까지 제안된 트랜스코더 중에서 가장 간단한 구조이며, 디코딩 과정에서 얻은 정보를 부호화 과정에 다시 이용하고 Q1의 양자화 값을 Q2로 변환해 비트율을 변화시키는 방식을 사용하고 있다. 실시간 동작을 요하는 응용분야에서는 거의 이 방식의 트랜스코딩 방식을 사용한다. 그러나 양자화값의 변화로 인해 드리프트 오류가 발생하게 되어 화질의 저하를 나타내는 단점이 있다.

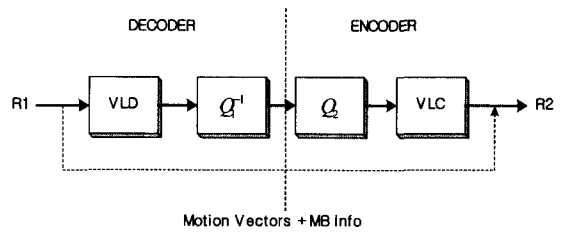


그림 1. 오픈 루프 트랜스코더
Fig. 1. Open Loop Transcoder

OPLT 구조에서와 같이 트랜스코딩을 하는데 있어서 화질의 저하를 발생시키는 가장 큰 문제는 드리프트 오류이다. 이러한 문제점을 해결하기 위해서 제안된 방식이 CPDT 구조이다[2][3]. 그림 2는 CPDT 구조의 트랜스코더를 나타낸다. 그림 2에서 R1의 비트율을 가지는 비트스트림을 VLD(Variable Length Decoding)과정을 통해 디코딩한 후 역양자화 과정과 IDCT 과정을 거쳐서 완벽히 재구성된 화면을 얻게 된다. 이것이 인코딩 과정으로 넘어가 Q2로 양자화 과정을 거치고, 이것을 VLC(Variable Length Coding)과정을 통해 R2의 비트율을 가지는 비트 스트림으로 만들어진다. 여기서 움직임 보상을 위한 예측 메모리가 부호화 과정과 디코딩 과정에 각각 존재하는 분리형 구조로 되어 있기 때문에 디코딩 과정에서 재구성된 화면을 부호화 과정에서 다시 비트스트림으로 만들기 때문에 드리프트 오류가 발생하지 않는다.

그러나 디코딩과 인코딩 과정에서 IDCT와 DCT 연산이 필요하고, 결과적으로 연산의 복잡성이 존재해 실시간 처리를 하기가 힘들다는 단점이 있다. 여기서 움직임 벡터나 매크로블록의 정보는 디코딩 과정에서 얻은 정보를 재사용하지만 IDCT와 DCT 연산이 비디오 압축에 있어서 가장 큰 영향을 미치기 때문에, 그것으로 인한 연산량의 감소 효과를 기대할 수 없다.

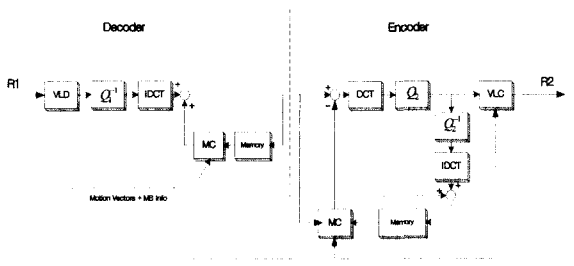


그림 2. CPDT의 블록도
Fig. 2. Block diagram of CPDT

Ⅲ. CDDT 트란스코딩 방식

1. CDDT의 기본 구조

OPLT는 단순한 구조와 지연 시간을 최소화한다는 장점이 있으나 드리프트 오류로 인해 화질의 열화를 가져오게 되어 고화질을 요하는 응용 분야에는 사용이 어렵다. 그리고 CPDT는 가장 일반적인 구조로서 드리프트 오류를 발생시키지 않음으로 가장 좋은 화질을 전송할 수 있지만, 복잡도가 너무 높고 동작 시간이 길어 실시간 처리가 힘들다는 단점을 가지고 있다.

이러한 단점을 보완하기 위해 CPDT보다 복잡도를 줄이고 동작 시간을 단축하며 드리프트 오류를 발생시키지 않음으로 고화질의 영상을 전송할 수 있는 CDDT 구조를 설명한다[4]. 그림 3은 CDDT 구조의 트랜스코더의 블록도를 나타낸다.

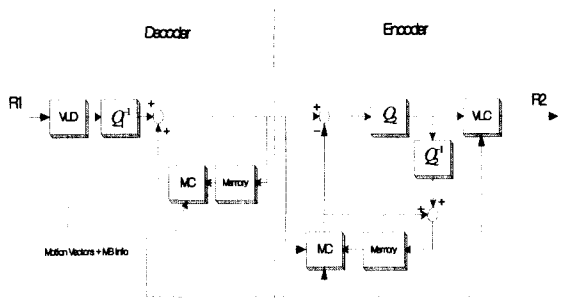


그림 3. CDDT의 블록도
Fig. 3. Block diagram of CDDT

CDDT의 동작은 다음과 같이 이루어진다. CDDT 내의 디코딩 과정에서는 R1의 비트스트림을 입력받아 VLD과정을 거쳐서 DCT 계수를 얻고, 역양자화 과정을 거친다. CPDT 구조에서는 공간영역에서의 동작을 위해서 역양자화된 DCT 계수 값들을 IDCT를 거쳐 공간영역의 화소값으로 변환한 후에 움직임 보상을 하게 되지만, CDDT에서는 IDCT 과정이 필요없이 DCT 계수를 이용해 움직임 보상을 한다. 재구성된 DCT 계수들은 CDDT의 인코딩 과정으로 넘어가게 되고 다시 움직임 보상된 DCT 계수들과의 차분을 구한 후에 양자화 과정을 거쳐 R2의 비트율로 전송한다. 이와 같은 CDDT 구조에서는 CPDT에서 수행하는 DCT 과정과 IDCT 과정을 생략할 수 있다.

DCT 영역에서의 동작은 여러 가지의 장점을 가지고 있다. 첫째로, 비트스트림을 디코딩하여 화면을 만들고 다시 구성된 화면을 인코딩하여 비트스트림을 만드는 과정을 없앨 수 있다는 것이다. 이것은 DCT와 IDCT 과정을 제거하는 동시에 지연시간 또한 줄일 수 있다. 둘째로, DCT와 IDCT 과정을 제거함으로 인해 DCT와 IDCT의 반복으로 인한 반올림 오차를 줄일 수 있게 되고 화질의 향상을 기대할 수 있다. 셋째로, 재구성된 영상의 비트율보다 더 적은 비트율을 갖는 비트스트림으로 구성되기 때문에 복잡도 및 계산량이 적어진다. 이것은 인코딩된 비트스트림이 '0'의 값을 많이 갖기 때문에 가능해진다. 물론 이것은 DCT 영역에서 어떤 동작을 하느냐와 압축률, '0'이 아닌 움직임 벡터의 개수 등에 따라 차이가 생길 수 있지만, 공간 영역에서의 동작에 비해서는 매우 단순해지게 된다.

그림 2를 통해서 CDDT는 CPDT와 매우 유사한 구조를 가지고 있다는 것을 알 수 있다. CPDT와 마찬가지로, 디코딩과 인코딩 과정에서 각각의 예측 메모리가 따로 존재하는 분리형 구조로 되어 있기 때문에 드리프트 오류가 생기지 않기 때문에 고화질의 영상 데이터의 전송이 가능하다. 또한 DCT 영역에서의 동작을 통해 DCT 및 IDCT 과정을 제거할 수 있으므로 복잡도가 훨씬 줄어드는 반면에, 트랜스코더 내의 인코딩 과정에서는 디코딩 과정에서 얻어지는 움직임 벡터 및 매크로블록 관련 정보를 재 사용하기 때문에 트랜스코더 내의 부호화 과정에서는 움직임 보상만 이루어진다. 이로 인해서 부호화 과정의 복잡도 및 계산량을 줄일 수 있고, 트랜스코더 내부의 DCT 및 IDCT 과정을 제거하여 DCT/IDCT 과정에서 발생하는 정밀도의 제약으로 발생할 수 있는 오차도 줄일 수 있게 되어 더욱 고화질의 영상을 얻을 수 있게 된다.

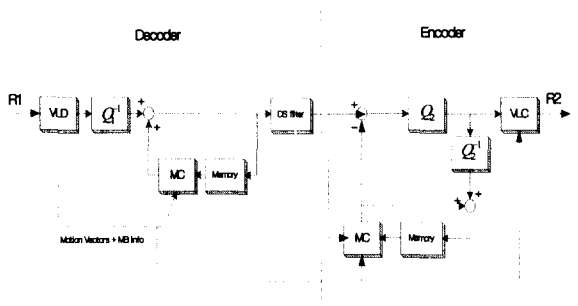


그림 4. 부표본화 필터가 포함된 CDDT의 블록도
Fig. 4. Block diagram of CDDT including down sampling filter

그림 4는 영상의 크기를 감소시키는 응용에 사용되는 부표본화 필터가 포함된 CDDT의 블록도이며, 그림 4에서와 같이 트랜스코더 내의 디코딩 과정과 인코딩 과정 사

이에 부표본화 필터의 삽입이 가능하여 특별한 변경 없이도 그대로 사용이 가능하다.

2. DCT 영역에서의 움직임 보상

DCT 영역에서 동작하는 트랜스코더를 설계하는데 있어서 DCT 영역에서의 움직임 보상은 필수적인 기술이다. 그림 5(a)는 공간 영역에서의 움직임 보상을 하는 디코더의 일반적인 블록도이다.

그림 5(a)는 식 (1)로 표현될 수 있다.

$$P_{rec}(t,x,y) = e(t,x,y) + P_{rec}(t-1, x-m_x, y-m_y) \qquad (1)$$

여기서 $P_{rec}(t,x,y)$, $e(t,x,y)$, m_x, 그리고 m_y 는 각각 t 프레임의 x, y 좌표에서의 재구성된 영상, 차분 영상, x 방향으로의 움직임 벡터와 y 방향으로의 움직임 벡터를 말한다.

DCT의 선형적인 성질을 이용하여 양변에 DCT를 취하면 위의 식 (1)은 다음 식 (2)와 같이 표현될 수 있다.

$$DCT(P_{rec}(t,x,y)) = DCT(e(t,x,y)) + DCT(P_{rec}(t-1, x-m_x, y-m_y)) \qquad (2)$$

이 결과를 다시 IDCT를 취하면 식 (1)과 같은 결과를 얻게 되며, 이와 같은 DCT 영역에서의 움직임 보상과정은 그림 5(b)와 같다[5][6].

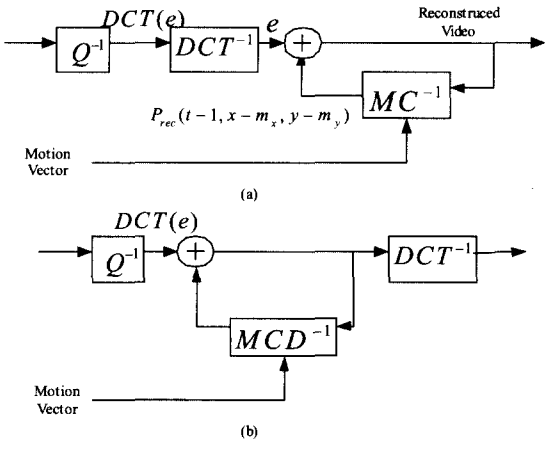


그림 5. (a) 전통적인 MC-DCT 기반 비디오의 디코더 (b) DCT 영역에서의 움직임 보상
Fig. 5. (a) MC-DCT based video decoder (b) Motion compensation in DCT domain

공간 영역에서의 움직임 보상은 이전 참조 영상으로부터 움직임 벡터만큼 이동하여 매크로블록 크기 만큼의 데이터를 복사하여 현재의 매크로블록의 데이터를 채우면 되지만, DCT 영역에서의 움직임 보상은 이와 같은 방법을 사용할 수 없다. 따라서 DCT 영역에서의 움직임 보상을 가능하게 하기 위해서는 공간 영역에서의 움직임 보상 과정을 수식화하여 이를 DCT 영역의 움직임 보상에 적용해야 한다. 공간영역에서의 움직임 보상 과정을 다르게 표현하면 윈도윙(windowing)과 쉬프팅(shifting)이라는 방법으로도 가능하다.

DCT 영역에서의 움직임 보상은 이와 같은 윈도윙과 쉬프팅이라는 조금 더 특별한 방법으로 행해지게 된다. 어느 한 블록의 좌측 상단의 데이터를 우측 하단으로 옮기기 위해서는 데이터를 복사하여 붙이는 방법이 있으며 또 다른 방법으로는 그림 6에서와 같이 데이터 블록의 앞과 뒤에 적절한 행렬을 곱해서 이동시키는 방법이 있다. 움직임 보상을 가능하게 하는 행렬은

$$H_1 = \begin{bmatrix} 0 & 0 \\ I_h & 0 \end{bmatrix}, \quad H_2 = \begin{bmatrix} 0 & I_w \\ 0 & 0 \end{bmatrix} \qquad (3)$$

이고, 여기서 I_h와 I_w는 각각 w와 h만큼 움직이고자 하는 블록의 수직과 수평 성분의 크기를 가지는 대각 성분이 1이고 나머지는 0인 행렬을 나타낸다. 그림 6에서 데이터 블록에 H_1을 곱하게 되면 선택된 블록은 좌측 하단으로 이동하게 되고, 여기에 H_2를 곱하게 되면 우측 하단으로 이동하게 되어 좌측 상단의 블록이 우측 하단으로 이동하게 된다. 여기에서 H_1과 H_2는 DCT를 연산을 위한 기본 단위의 크기인 8×8의 크기를 갖게 된다[5][6].

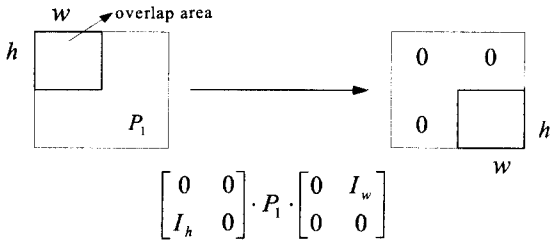


그림 6. 단순한 행렬 곱에 의한 블록의 이동
Fig. 6. Block shifting by simple matrix multiplication

하나의 블록을 위와 같이 윈도윙과 쉬프팅을 이용하여 움직이는 데에는 움직임 벡터에 따라 3 가지의 경우가 생기게 된다. 첫 번째 경우는 움직임 벡터가 가로 및 세로로

모두 '0' 이거나 모두 블록 크기의 배수인 경우이다. 이와 같은 경우에는 윈도윙과 쉬프팅의 동작이 필요없으며, 공간에서와 같이 블록을 복사하여 사용하면 된다. 두 번째의 경우는 가로 또는 세로로 어느 한 방향의 움직임 벡터가 '0' 또는 블록 크기의 배수인 경우로, 이와 같은 경우에는 H_1 또는 H_2를 하나만 사용하여 블록을 재구성하게 된다. 마지막 경우는 위의 2가지 경우를 제외한 것으로, 하나의 블록을 재구성하기 위해 H_1과 H_2를 모두 사용하게 된다.

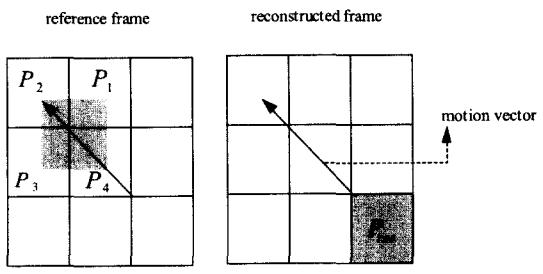


그림 7. 블록의 움직임보상
Fig. 7. Motion compensation for one block

그림 7은 하나의 블록을 위와 같이 윈도윙과 쉬프팅을 이용하여 이동시키는 예를 보여주고 있으며, 가장 복잡한 경우인 움직임 벡터가 모두 '0'이 아니고 블록 크기의 배수도 아닌 경우의 예를 보여준다. 하나의 블록을 이동하여 재구성하기 위해서는 4번의 동작이 필요하다. 이와 같은 동작은 다음 식(4)와 같이 표현할 수 있다.

$$P_{rec} = \sum_{i=1}^{4} H_{i1} P_i H_{i2} \tag{4}$$

DCT 영역에서의 동작은 아래 식(5)와 같이 나타낼 수 있다.

$$DCT(P_{rec}) = \sum_{i=1}^{4} DCT(H_{i1}) DCT(P_i) DCT(H_{i2}) \tag{5}$$

여기서 P_i는 참조영상의 해당 블록을 나타낸다.

위와 같이 DCT 영역에서의 움직임 보상 과정은 이전 프레임의 DCT 계수 값에 적절한 움직임 보상 행렬 H_1과 H_2 DCT 변환한 행렬을 곱해서 나타낼 수 있다. 이러한 $DCT(H_1)$과 $DCT(H_2)$는 고속 연산을 위해 미리 버퍼에 저장해 둔 상태로 사용 된다.

DCT 영역에서의 움직임 보상을 빠르게 수행하기 위한 고속 알고리듬으로는 식 (5)를 재구성하여 처리 속도를 향상시킨 방법[7], 고속 DCT 알고리듬인 DCT factorization을 이용한 방법[8][9], 미리 계산된 pre-matrix와 post-matrix를 근사화 시켜 쉬프팅과 덧셈만으로 처리가 가능하게 한 방법[4], 그리고 움직임 보상 과정에서 공통된 정보를 이용하여 계산량을 줄이는 방법[10]이 있다. 본 논문에서는 미리 계산된 행렬을 이용한다.

DCT 영역에서 움직임 보상을 위해서는 반화소의 움직임 보상이 필요하다. 공간 영역에서의 반화소 움직임 보상을 위해서는 일반적으로 선형 보간이 사용된다. 한 방향만으로 반화소의 움직임이 발생하였다면, 그와 같은 방향으로 이웃하는 두 화소의 평균값을 취하게 되고, 두 방향 모두 반화소의 움직임이 발생하였다면 두 방향으로 이웃하는 두 화소씩, 4개 화소의 평균값을 취하게 된다.

하나의 블록에서 수평 방향으로의 평균을 구하는 과정을 행렬의 곱의 형태로 표현하면

$$P_i^v = P_i f \tag{6}$$

이고, 여기서 P_i, P_i^v, 그리고 f는 각각 선형 보간 이전의 블록 데이터, 수평 방향으로 선형 보간의 결과, 그리고 선형 보간 필터이다. DCT 영역에서의 반화소 움직임 보상은 수평, 수직 방향으로 +1, -1의 움직임 벡터가 발생할 경우로 분리할 수 있다. 아래의 행렬은 식 (6)의 필터 f를 수평, 수직 방향으로 반화소 움직임 보상을 할 때 사용하는 필터를 나타낸다[4].

$$f_1 = 1/2\begin{bmatrix} 1&0&0&0&0&0&0&0\\ 1&1&0&0&0&0&0&0\\ 0&1&1&0&0&0&0&0\\ 0&0&1&1&0&0&0&0\\ 0&0&0&1&1&0&0&0\\ 0&0&0&0&1&1&0&0\\ 0&0&0&0&0&1&1&0\\ 0&0&0&0&0&0&1&1 \end{bmatrix} \quad f_2 = 1/2\begin{bmatrix} 0&0&0&0&0&0&0&1\\ 0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0 \end{bmatrix}$$

$$f_3 = 1/2\begin{bmatrix} 1&0&0&0&0&0&0&0\\ 1&1&0&0&0&0&0&0\\ 0&1&1&0&0&0&0&0\\ 0&0&1&1&0&0&0&0\\ 0&0&0&1&1&0&0&0\\ 0&0&0&0&1&1&0&0\\ 0&0&0&0&0&1&1&0\\ 0&0&0&0&0&0&1&2 \end{bmatrix} \quad f_4 = 1/2\begin{bmatrix} 2&0&0&0&0&0&0&0\\ 1&1&0&0&0&0&0&0\\ 0&1&1&0&0&0&0&0\\ 0&0&1&1&0&0&0&0\\ 0&0&0&1&1&0&0&0\\ 0&0&0&0&1&1&0&0\\ 0&0&0&0&0&1&1&0\\ 0&0&0&0&0&0&1&1 \end{bmatrix}$$

$$F_i = DCT(f_i) \tag{7}$$

첫 번째 경우는 수평 방향으로 반 화소 움직임 벡터가 +1만큼 발생한 경우로 식 (8)과 같이 나타낼 수 있다. 첫 번째 블록과 세 번째 블록의 우측 경계 값은 이웃하는 두 번째, 네 번째 블록의 좌측 경계 값과의 평균을 취하게 되고, 두 번째와 네 번째 블록의 우측 경계 값은 자기 자신의 값을 갖게 된다.

B_i는 매크로블록 내의 블록을 의미하며, i는 각 블록의 번호를 나타낸다. 번호는 좌측 상단이 1, 우측 상단이 2, 좌측 하단이 3 그리고 우측 하단의 블록이 4이다. $B_i{}^h$는 수평 방향으로 반 화소 움직임 보상한 결과의 블록을, $B_i{}^v$는 수직 방향으로 반 화소 움직임 보상한 결과의 블록을 나타낸다.

$$B_i{}^h=\begin{cases} B_i F_1{}^h + B_{i+1} F_2{}^h, & i=1,3 \\ B_i F_3{}^h, & i=2,4 \end{cases} \quad (8)$$
$$F_1{}^h=F_1,\ F_2{}^h=F_2,\ F_3{}^h=F_3$$

두 번째 경우는 수직 방향으로 반 화소 움직임 벡터가 +1만큼 발생한 경우로 식 (9)와 같이 나타낼 수 있으며, 첫 번째와 두 번째 블록의 하단 경계 값은 세 번째 블록과 네 번째 블록의 상단 경계 값과의 평균을 취하게 되고, 세 번째와 네 번째 블록의 하단 경계 값은 자기 자신의 값을 갖게 된다.

$$B_i{}^v=\begin{cases} F_1{}^v B_i + F_2{}^v B_{i+2}, & i=1,2 \\ F_3{}^v B_i, & i=3,4 \end{cases} \quad (9)$$
$$F_1{}^v=(F_1)^T,\ F_2{}^v=(F_2)^T,\ F_3{}^v=(F_3)^T$$

수평 방향으로 반화소 움직임 벡터가 -1만큼 발생한 경우도 식 (10)과 같이 나타낼 수 있다. 두 번째 블록과 네 번째 블록의 좌측 경계값은 이웃하는 첫 번째, 세 번째 블록의 우측 경계 값과의 평균을 취하게 되고, 첫 번째와 세 번째 블록의 좌측 경계값은 자기 자신의 값을 갖게 된다.

$$B_i{}^h=\begin{cases} B_i F_1{}^h + B_{i-1} F_2{}^h, & i=2,4 \\ B_i F_3{}^h, & i=1,3 \end{cases} \quad (10)$$
$$F_1{}^h=(F_1)^T,\ F_2{}^h=(F_2)^T,\ F_3{}^h=(F_4)$$

수직 방향으로 반화소 움직임 벡터가 -1만큼 발생한 경우로 식 (11)와 같이 나타낼 수 있다. 세 번째 블록과 네 번째 블록의 상단 경계 값은 이웃하는 첫 번째와 두 번째 블록의 하단 경계 값과의 평균을 취하게 되고, 첫 번째와 두 번째 블록의 상단 경계 값은 자기 자신의 값을 갖게 된다.

$$B_i{}^v=\begin{cases} F_1{}^v B_i + F_2{}^v B_{i-2}, & i=3,4 \\ F_3{}^v B_i, & i=1,2 \end{cases} \quad (11)$$
$$F_1{}^v=F_1,\ F_2{}^v=F_2,\ F_3{}^v=F_4$$

3. DCT 영역에서의 부표본화

비트율의 변화를 통해 비트스트림의 크기를 줄이는 방법이외에 효율적인 비디오 전송을 위한 방법으로는 비디오의 해상도 축소를 통한 방법이 있다. 일반적인 방법은 위에서 다음에 설명되는 필터들을 사용하면 DCT 영역에서 해상도를 줄이거나 늘릴 수 있다

아래 행렬은 비디오의 크기를 가로, 세로 각각 1/2로 만들 때 필터로 사용된다.

$$H_1=H_3=\begin{bmatrix} 0.5 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.5 & 0.5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.5 & 0.5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0.5 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$H_2=H_4=\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.5 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.5 & 0.5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.5 & 0.5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0.5 \end{bmatrix}$$

위의 행렬의 원소들이 0.5로 되어 있으므로 인접한 두 화소사이의 평균값을 나타낼 수 있다. 가로, 세로의 해상도를 줄이는 것이므로 해상도를 줄이고자 하는 비디오의 블록에 해상도 감소 행렬을 곱하게 되면

$$B=H_1B_1W_1+H_1B_2W_2+H_2B_3W_1+H_2B_4W_2 \quad (12)$$

이고, 여기서 W_i는 $W_i=(H_i)^T$으로써 얻어 낼 수 있다[6].

DCT 영역에서의 동작은 아래와 같다.

$$DCT(B)=\sum_{i=1}^{4} DCT(H_i)DCT(B_i)DCT(W_i) \quad (13)$$

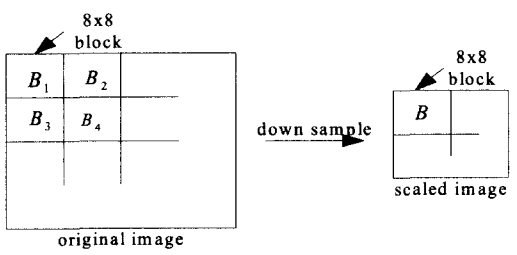
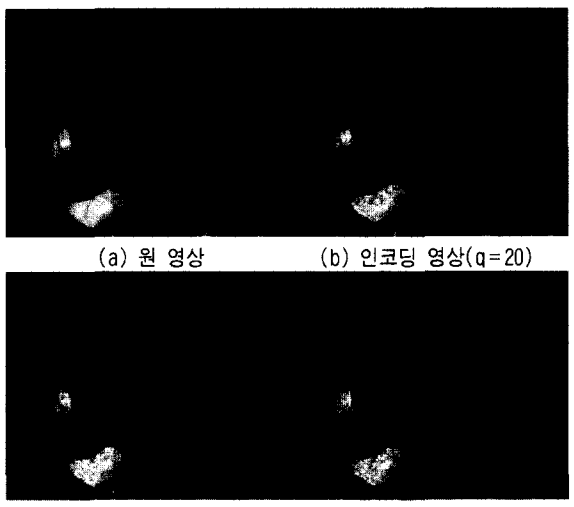


그림 8. 부표본화 샘플링 방법
Fig. 8. Down sampling method

DCT 영역에서의 고속 동작을 위해서 행렬 $DCT(H_i)$와 $DCT(W_i)$을 미리 계산해 버퍼에 저장해 놓는다.

Ⅳ. 실험 결과

고화질 영상의 전송이 가능하다고 알려진 CPDT와 제안된 방식인 CDDT와의 화질에 대한 성능을 비교하기 위한 실험 조건으로는 최초 인코더의 양자화 값은 10으로, 트랜스코더 내의 인코더의 양자화 값은 20을 사용하였으며 45번째까지의 프레임에 대해 실험을 수행 하였다. 또한 비교를 위해 원래의 영상을 20으로 직접 압축한 결과를 참고로 표시하였다.



(a) 원 영상 (b) 인코딩 영상(q=20)

(c) CPDT 결과영상 (d) CDDT 결과영상

그림 9. Suzie 영상의 45번째 프레임의 트랜스코딩 결과영상
Fig. 9. Transcoding result of Suzie sequence(45th frame)

그림 9는 QCIF크기의 Suzie Sequence에 대한 실험결과 영상을 보여주고 있으며, 그림 10은 이 실험에 대한 PSNR 그래프를 나타낸다. 그림 9(a)는 45번째 프레임의 원 영상을, 그림 9(b)는 양자화 값 20으로 직접 압축한 영상을, 그림 9(c)는 CPDT의 결과영상을 그리고 그림 9(d)는 제안된 CDDT의 결과 영상을 나타낸다. 실험 결과 영상을 눈으로 판단하기에는 개인적인 시각과 관점에 차이가 생겨 동일한 결과에 대해서도 다른 견해를 나타낼 수 있으므로, 그림 10과 같은 PSNR 값을 비교한다.

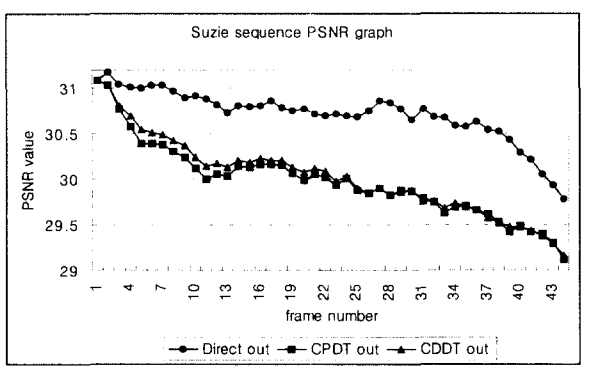


그림 10. Suzie 영상을 이용한 트랜스코딩 결과의 PSNR 그래프
Fig. 10. The PSNR graph of transcoding result using Suzie seq

표 1. 여러 가지 실험영상에 대한 트랜스코딩 결과 평균 PSNR
Table 1. Average PSNR of Transcoding using some experimental image

	Direct out	CPDT	CDDT	(CDDT-CPDT)
Suzie (QCIF)	30.7128	29.9788	30.0240	0.0452
Foreman (QCIF)	28.141	27.1169	27.1432	0.0263
Miss USA (QCIF)	33.9574	33.2087	33.285	0.0763
Akiyo (CIF)	32.963	32.2945	32.3427	0.0482

그림 10에서 각 프레임에서의 PSNR값을 비교해 보면, 제안된 방식의 결과가 기존의 방식보다 약간의 화질 개선이 이루어진 것을 알 수 있다. 표 1은 몇 가지의 다른 영상에 대한 평균 PSNR을 보여주고 있으며, 0.02 ~0.07dB 정도의 화질 개선이 이루어진 것을 볼 수 있다.

제안된 CDDT에 부표본화 필터가 삽입된 경우의 실험은 가로 및 세로 방향으로 1/2씩 크기를 줄이는 실험을 하였으며, 여기에는 CIF 크기의 영상을 QCIF 크기로 줄이는 방법을 사용하였다. 사용된 영상은 Foreman과 Akiyo, Football Sequence를 사용하였다.

표 2에서 Case 1은 4개의 움직임 벡터의 평균을 구해 하나의 움직임 벡터를 만들었을 때의 결과를 나타내고, Case 2는 하나의 움직임 벡터와 나머지 3개와의 유클리

표 2. CDDT에 부표본화 필터가 삽입되었을 경우의 움직임벡터 선정에 따른 평균 PSNR
Table 2. Average PSNR according to motion vector selection in case of down sampling filter in CDDT

	Case 1	Case 2	Case 3	Case 4
Foreman (CIF)	25.6264	25.9299	25.7276	25.5164
Football (CIF)	23.2954	23.2438	23.4217	23.3599
Akiyo (CIF)	28.5483	28.5044	28.6435	28.5228

디언 거리를 계산해 최소가 되는 움직임 벡터를 선택하였을 경우의 결과를, Case 3은 같은 방향을 갖는 움직임 벡터들의 평균을 구한 결과를, Case 4는 4개의 움직임 벡터 중에서 임의로 하나를 선택한 경우로 이 실험에서는 좌측 상단의 움직임 벡터를 선택하였을 경우의 실험 결과를 보여주고 있다. Foreman 영상에서는 Case 2가, 나머지 경우에는 Case 3이 가장 좋은 결과를 보여주고 있다. 그러나 Case 2는 나머지 영상에서 가장 나쁜 결과를 보여주고 있으므로 편차가 크게 나타나므로 Case 3이 가장 좋은 결과를 나타낸다고 할 수 있다. 그러나 이 방법은 움직임 벡터를 선정하는 데에 추가적인 계산이 필요하므로, 고속을 요하는 경우에는 추가적인 계산이 필요하지 않는 Case 1과 Case 4를 사용하게 된다. Case 4는 Football 영상에서는 Case 1보다 좋은 결과를 얻었지만 대부분의 경우에는 Case 1이 더 좋은 결과를 보이므로, 고속의 움직임 벡터의 선정이 요구되는 경우에는 Case 1을 사용하게 된다.

V. 결 론

본 논문에서는 DCT 영역에서 동작하는 효율적인 트랜스코더에 대해 제안을 하였다. DCT 영역에서 동작하는 트랜스코더는 기존의 공간 영역에서 동작하는 트랜스코더에 비해 화질 면에서나 복잡도 면에서나 좋은 결과를 가져올 수 있었다.

DCT 영역에서 동작하는 트랜스코더를 구현하기 위해 DCT 영역에서의 동작에 대해서 소개를 하고 트랜스코더에 사용되어 여러 가지 필요한 기능을 수행할 수 있게 하였다. 또한 이러한 DCT 영역에서의 기술들은 트랜스코더 이외에도 압축된 비트열의 상태에서 조작을 필요로 하는 영상 합성이나 영상회의 등의 여러 가지 응용 분야에서 사용이 가능하다.

제안된 방법을 이용한 실험에서는 화질면에서는 0.2~0.7 dB 정도의 화질의 개선이 있었고, 트랜스코더에서 많은 계산량을 차지하고 있는 움직임 보상과정이 최대 68% 정도 계산량의 감소가 있었음을 알 수 있었다. 기존의 트랜스코더에서의 큰 단점이던 복잡한 구조와 많은 계산량을 제안된 트랜스코더가 DCT 영역에서 수행을 함으로서 구조를 단순화 시키면서 계산량을 줄이고 다른 트랜스코더에서의 단점인 드리프트 오류의 발생은 제안된 트랜스코더가 CPDT 구조처럼 디코더 블록과 인코더 블록이 분리되어 있어 발생하지 않게 하였다.

참고 문헌

[1] ITU-T, ISO/IEC 13818-2, "Information Technology - Generic Coding of Moving Pictures and Associated Audio Information", Nov. 1994.

[2] N. Bjork and C. Christopoulos, "Transcoder Architectures for Video Coding," *IEEE Trans. Consumer Electron.*, vol. 44, pp. 88-98, Feb. 1998.

[3] J. Youn and M. Sun, "Motion vector refinement for high-performance transcoding," *IEEE Trans. Multimedia*, vol. 1, pp. 30-40, march. 1999.

[4] P. Assuncao and M. Ghanbari, "Post-processing of MPEG2 coded video for transmision at lower bit rates," *ICASSP'96*, vol. 4, pp. 1998-2001, May 1996.

[5] S.-F. Chang and D. G. Messerschmitt, "Manipulation and compositing of MC-DCT compressed video," *IEEE J. Select. Areas Commun*, vol. 13, pp. 1-11, Jan. 1995.

[6] S.-F. Chang and D. G. Messerschmitt, "A new approach to decoding and compositing motion compensated DCT-based images," *Proc. IEEE ICASSP '93*, pp. V.421-V.424, April 1993.

[7] Y. Shibata, Z. Chen and R. Campbell, "A fast degradation-free algorithm for DCT block extraction in the compressed domain," *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing*, vol. 6, pp. 3185-3188, March 1999.

[8] Y. Arai, T. Agui and M. Nakajima, "A fast DCT-SQ scheme for images," *Trans. IEICE*, vol. E 71, pp. 1095-1097, Nov. 1988.

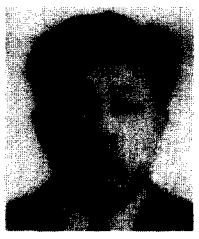
[9] N. Merhav and V. Bhaskaran, "Fast algorithm for DCT-domain image down -sampling and for inverse motion compensation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, pp. 468-476, June 1997.

[10] J. Song and B. Yeo, "A fast algorithm for DCT-domain inverse motion compensation based on shared information in a macroblock," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 10, pp. 767-775, Aug. 2000.

저 자 소 개

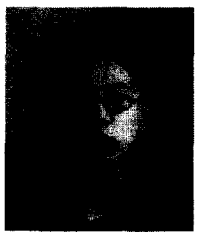
김 성 진

2000년 2월 : 중앙대학교 전자전기공학부 졸업 (공학사)
2000년 2월~현재 : 중앙대학교 첨단영상대학원 영상공학과 석사과정
주관심분야 : 영상 압축, 영상 통신, 영상 복원



황 인 경

1995년 2월 : 중앙대학교 전자공학과 졸업 (공학사)
1999년 9월~2001년 8월 : 중앙대학교 첨단영상대학원 영상공학과 졸업 (석사)
1995년 1월~1996년 : LG반도체 연구원
19996년~현재 : I&C 테크놀러지 연구원
주관심분야 : 영상 압축, 영상 통신

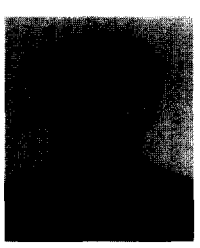


정 웅 찬

1999년 2월 : 중앙대학교 전자공학과 공학사
2001년 9월 : 중앙대학교 첨단영상대학원 석사과정
주관심분야 : Augmented Reality

백 준 기

1990년 : 노스웨스턴대학교 전기공학박사
1990년 : 삼성전자 선임연구원
1993년 : 중앙대학교 전자공학과 조교수
1999년 : 중앙대학교 첨단영상대학원 부교수
2001년 : 테네시주립대학 방문교수
주관심분야 : 영상통신, 영상복원, 3차원영상처리, 영상감시시스템

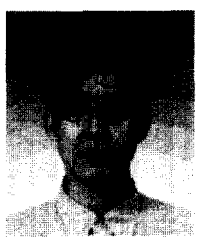


김 제 우

1997년 2월 : 서울시립대학교 제어계측공학과 졸업 (공학사)
1999년 2월 : 서울시립대학교 제어계측공학과 졸업 (공학석사)
1999년 1월~현재 : 전자부품연구원 전임연구원
주관심분야 : 영상 압축 알고리즘, 영상통신시스템 관련 알고리듬, 3차원 영상통신 알고리즘

송 혁

1999년 2월 : 광운대학교 제어계측공학과 졸업 (공학사)
2001년 2월 : 광운대학교 전자공학과 졸업 (공학석사)
2000년 12월~현재 : 전자부품연구원 연구원
주관심분야 : 영상압축 하드웨어, 영상통신, 로봇 영상시스템



백 종 호

1994년 2월 : 중앙대학교 전기공학과 졸업(공학사)
1997년 2월 : 중앙대학교 전기공학과 졸업(공학석사)
1997년 1월~현재 : 전자부품연구원 전임연구원
주관심분야 : 영상 및 방송 통신, 유무선 통신시스템