H.264에서의 시방향(時方向) 에러은닉 기법

정회원 김 동 형*, 정 제 창**

A Temporal Error Concealment Algorithm with Adaptive Block Size in the H.264/AVC Standard

Donghyung Kim*, Jechang Jeong** ***Regular Members***

요 약

H.264/AVC 동영상 부호화 표준에서는 새로운 부호화 도구들이 추가되었으며 이를 통하여 보다 높은 압축 효율을 보인다. 추가된 부호화 도구들로 인하여 H.264내의 매크로블록은 이전의 부호화 표준들에 비하여 보다 많은 정보를 포함한다. H.264 부호화 표준에서는 하나의 매크로블록이 최대 16개까지의 서로 다른 움직임벡터를 가질 수 있으며 최대 4개의 서로 다른 참조프레임을 가질 수 있다. 또한 다양한 블록크기로의 움직임 추정하며 이는 매크로블록 모드로서 정의된다. 따라서 H.264내의 매크로블록은 기존보다 많은 움직임벡터를 가질 뿐만 아니라 기존에는 없던 참조프레임과 매크로블록 모드의 정보를 새로이 포함하고 있다. 본 논문은 H.264 부호화 표준의 특성을 고려하여 매크로블록이 포함하고 있는 다양한 크기의 움직임 벡터 및 참조 프레임 뿐 아니라 주변 매크로블록들의 모드를 이용하여 서로 다른 블록 크기로 에러를 은닉함으로써 매크로블록이 포함하고 있는 정보를 최대한 활용하고 이를 통한 효율적인 에러은닉 알고리듬을 제안한다. 제안하는 알고리듬은 기존의 시방향 에러은닉 기법과 비교하여 적은 연산양만을 가지면서도 비트율에 관계없이 향상된 객관적 화질을 나타낸다.

Key Words : Error Concealment; H.264; Temporal; Macroblock Mode; OBMA.

ABSTRACT

For the improvement of coding efficiency, the H.264 standard uses new coding tools. Using these coding tools, H.264 has achieved significant improvements from rate-distortion point of view. The adoption of these tools enables a macroblock in H.264 to have more information, sixteen motion vectors, four reference frames and a macroblock mode. In this paper, we present an efficient temporal error concealment algorithm by using not only motion vectors and reference frames but also macroblock mode of neighbor macroblocks. Our algorithm conceals the macroblock error with variable sizes, 16x16, 16x8, 8x16, 8x8 depending on the macroblock modes of neighbor macroblocks. Simulation results show that the proposed method increase the objective quality regardless of bit-rate and block error rate.

Ⅰ. 서 론

H.264/AVC 표준은 ITU-T의 VCEG(Video Coding Experts Group)과 ISO/IEC의 MPEG (Moving Picture Expert Group)의 공동 작업으로 제정되었다[1]. H.264/AVC 비디오 부호화 방식은 이전의 비디오 부호화 표준들(MPEG2, MPEG4 Part2, H.263)에는 없던 새로운 부호화 도구들이 추

* 한양대학교 전자통신전파공학과 영상처리 및 신호처리 연구실 (kimdh@ece.hanyang.ac.kr)
** 한양대학교 전자전기컴퓨터 공학부 (jjeong@ece.hanyang.ac.kr)
논문번호 : KICS2004-12-300, 접수일자 : 2004년 12월 02일
※ 본 논문은 정보통신부의 출연금으로 수행한 IT SoC 핵심설계인력양성사업의 수행결과입니다.

가되었으며, 이러한 도구들은 H.264 부호화 표준이 이전의 비디오 부호화 표준과 비교하여 보다 높은 압축효율을 갖게 하였다. 새롭게 추가된 부호화 도구들을 간단히 살펴보면 다음과 같다.

정수 변환 및 양자화 : H.264에서는 4×4 단위의 정수 변환을 수행하며, 부호기와 복호기 사이의 부정합(Mismatch) 문제가 발생하지 않는다. 이는 8×8 단위의 이산여현변환(Discrete Cosine Transform)을 사용함으로써 부호기와 복호기 사이에서의 부정합 문제를 가지고 있던 기존의 여러 비디오 부호화 표준들과 비교된다. 또한 매크로블록이 인트라 16×16 모드로 선택된 경우 변환수행 단위인 4×4 변환블록의 DC값들은 다시 한번 하다마드(Hadamard Transform) 변환을 수행하며, 이러한 계층적 변환(Hierarchical Transform)을 통해서 보다 높은 압축 효율을 나타낸다. 또한 H.264에서의 양자화 파라미터는 양자화 데이터를 가리키는 하나의 인덱스(Index)로 사용되며 0~51까지 52단계의 값을 가진다. 양자화 스텝 크기(Quantization Step Size)는 양자화 파라미터가 여섯 단계 증가함에 따라 두 배로 증가한다.

엔트로피 부호화 : H.264에서는 프로파일에 따라 두 가지의 엔트로피 부호화 방법을 사용한다. 첫 번째 엔트로피 부호화 기법은 변환 계수를 제외한 대부분의 신텍스(Syntax)를 데이터의 특성에 따라 하나의 코드워드(Codeword) 테이블로 변환 후 EGC (Exponential Golomb Code)로 부호화 하고, 변환계수는 이미 부호화 된 주변 블록들의 특성을 이용하는 CAVLC(Context Adaptive Variable Length Coding)로 부호화 하는 방법이다. 두 번째의 엔트로피 부호화 기법은 기존의 산술 부호화 방식에 컨텍스트 모델링(Context Modeling)을 추가함으로써 추정된 심볼(Symbol) 확률 분포 모델들을 적응적으로 선택하는 방식으로 EGC와 CAVLC를 이용하는 엔트로피 부호화 방식에 비해 높은 복잡도를 가지지만 5~15%의 비트율 감소효과가 있다.

루프 필터 : 블록 기반의 움직임 추정 및 변환 그리고 양자화 기법은 부호기 설계에 많은 이점을 가져오지만 블록화 현상(Blocking Artifact)을 수반한다. H.264는 이러한 블록화 현상의 효율적인 개선을 위해 부호화 및 복호화 과정 내(內)에 루프 필터를 사용하여 블록화 현상을 제거한다. 이는 부호화 및 복호화 과정과 독립적인 후처리(Post-Processing)의 수행을 통해서 블록화 현상을 제거하는 기존의 부호화 표준들과 비교된다.

인트라 예측 부호화 : H.264 부호화 기법은 공간영역에서의 인트라 예측 부호화 과정을 포함한다. 공간 내의 예측 부호화 방식은 휘도(Luminance) 성분의 경우 16×16크기 및 4×4크기로 수행되며, 각각 4가지와 9가지의 공간내의 방향성을 고려한 예측모드를 가지고 있고, 색차(Chrominance) 성분은 16×16 크기의 휘도성분과 같은 4가지의 예측모드를 가지고 있다. 휘도 성분에 대한 공간영역에서의 16×16 크기와 4×4 크기의 인트라 예측은 영상의 평탄한 영역뿐만 아니라 세밀한 영역에서도 높은 압축 효율을 갖게 한다. 공간 영역(Spatial Domain)에서의 인트라 예측 부호화 방식은 변환 영역(Transform Domain)에서 휘도 및 색차 신호에 대한 인트라 예측 부호화를 수행하는 MPEG-4의 표준과 비교된다.

다중 참조 영상 및 1/4 화소 단위의 움직임 추정 : H.264에서는 참조 영상으로서 이전의 재구성된 여러 장의 프레임을 사용한다. 이러한 다중 참조 영상의 사용은 단일 참조 영상만을 사용하는 경우에 비해 보다 높은 압축 효율을 가지게 하며, 특히 반복적인 영상이나 다른 객체에 의해 가려져 있다가 나타난 영역의 부호화 경우에 더욱 두드러진 성능을 보인다. 또한 H.264에서는 보다 정확한 움직임 추정을 위해 보간(Interpolation) 기법을 이용하여 1/4 화소 단위로 움직임을 추정한다. 이때 1/2 화소의 보간 방식으로는 6탭 필터를 사용하며, 1/4 화소 단위의 보간 방법은 쌍선형 필터(Bilinear Filter)를 사용한다.

다양한 블록크기의 움직임 추정 : H.264에서는 블록의 크기를 다양화하여 16×16, 16×8, 8×16, 8×8, 8×4, 4×8, 4×4의 크기로 움직임을 추정한다. 이는 기존에 16×16 및 8×8 크기로만 움직임을 보상하던 H.263 및 MPEG-4 기술과 대비된다. 다양한 크기의 블록 단위, 특히 8×8 부-매크로블록(Sub Macroblock) 이하의 크기인 8×4, 4×8, 4×4 블록 크기의 움직임 추정은 보다 세밀한 영상 영역에서도 높은 압축 효율을 가질 수 있게 한다.

그 밖에도 적응적 프레임/필드 부호화(Adaptive Frame/Field Coding) 및 NAL(Network Abstraction Layer)등의 부호화 도구들이 추가되었다. 이렇게 새롭게 추가된 부호화 도구들은 H.264 부호화 표준이 이전의 여러 동영상 부호화 표준들에 비해 보다 높은 압축 효율을 가질 수 있게 한다. 다섯 장의 참조 프레임의 사용과 32 화소의 움직임 추정 영역을 사용하여 메인 프로파일로 부호화 한 경우, H.264 표

준은 MPEG-2와 비교하여 약 48~78%까지의 비트율 감소효과가 있다고 알려져 있다[2].

이중 다양한 블록크기의 움직임 추정기법은 기존의 부호화 표준과 비교하여 보다 많은 계산량을 요구하지만 그에 따라 보다 세밀한 단위로 움직임을 추정함으로써 부호화 효율을 높이는데 상당한 영향을 미치는 부호화 도구이다. 이렇게 다양한 블록 크기를 사용하여 움직임을 추정함으로써 나타나는 또 다른 특징은 기존의 동영상 표준에 비해 보다 많은 움직임 벡터를 생성한다는 것이다. 즉, 기존의 동영상 표준이 16x16 크기를 갖는 하나의 매크로블록에 대해 하나의 움직임 벡터만을 갖는 반면 H.264는 16x16 크기의 매크로블록 하나가 가질 수 있는 움직임 벡터의 개수는 최대 16개이다. 이는 H.264에서의 오류은닉 시 사용될 수 있는 주변 매크로블록의 움직임벡터 정보의 양이 기존의 부호화 표준에 비해 많음을 나타낸다. 또한 여러 장의 참조영상을 사용함으로써 각 매크로블록은 움직임벡터와 함께 참조프레임의 정보 또한 포함하게 된다. 따라서 H.264내 하나의 매크로블록은 최대 16개의 서로 다른 움직임벡터와 최대 4개의 참조프레임 정보 그리고 움직임 추정의 단위를 정의하는 매크로블록 모드 정보를 포함하게 된다.

MPEG-2가 제정된 이후로 시방향 에러은닉에 관한 많은 연구가 이루어져 왔다. Keiu는 손실된 매크로블록의 움직임을 추정함으로써 추정된 움직임 벡터를 이용하여 기준블록을 복사하여 에러를 은닉하는 기법을 제안하였다[3]. 하지만 움직임 벡터의 추정이 정확하지 않은 경우 주변블록과 많은 차이를 보이게 되는 단점을 가진다. 이와 유사한 방법으로서 Suh와 Kwon은 각각 주변 매크로블록 움직임 벡터의 평균값[4] 및 중간값[5]을 사용하여 에러를 은닉하는 기법을 제안하였으며 이는 손실된 매크로블록과 이웃한 매크로블록의 움직임벡터간의 상관도가 높다는 가정이 전제되어야 한다. Jung은 손상된 매크로블록의 주변 경계화소들을 이용하여 움직임 벡터를 추정하는 BMA (Boundary Matching Algorithm) 기법을 개선한 수정된 BMA 기법을 제안하였다[6]. 하지만 이는 높은 복잡도를 갖는 단점을 가진다. 위의 여러 시방향 에러은닉 기법들은 16x16 크기로 움직임 추정을 수행하는 이전의 부호화 표준들에 적합한 알고리듬이다. Zheng은 다양한 블록크기로 움직임을 추정하는 H.264에서의 보다 많은 움직임 벡터 정보를 이용하여 화소단위의 움직임벡터 복원을 통하여 에러를 은닉하였지만 이 역시 다중 참조 프레임 및 매크로블록 모드 정보는 고려하지 못하고 있다[7].

본 논문은 H.264의 부호화 특성을 고려하여 매크로블록이 포함하는 최대 16개의 서로 다른 움직임 벡터와 최대 4개의 서로 다른 참조프레임 뿐만 아니라 손실된 블록 주변의 매크로블록 모드에 따라 다양한 크기로 에러를 은닉하는 보다 효율적인 시방향 에러은닉 기법을 제안한다.

논문의 구성은 2장에서 현재 H.264에서의 시방향 에러은닉 기법을 기술하고 3장에서 본 논문에서 제안하는 적응적 블록 크기를 갖는 에러은닉 알고리듬을 기술한다. 4장에서는 제안하는 알고리듬의 적용을 통하여 알고리듬의 타당성을 보이고 마지막 절에서 결론을 맺는다.

Ⅱ. H.264/AVC에서의 시방향 에러은닉기법

2.1 BMA 및 OBMA 에러은닉 기법

BMA(Boundary Matching Algorithm)기법은 동영상 부호화 표준에서 시방향 움직임추정을 통한 에러은닉에 가장 널리 사용되는 방법이다. 이는 인접한 화소들 간에 큰 상관성을 가지는 자연영상의 특징을 이용한 알고리듬으로서 에러가 발생한 매크로블록의 인접화소들과 주변 매크로블록의 움직임벡터로부터 추정된 블록의 가장자리 화소들 간의 차이값을 최소로 하는 움직임벡터를 선정함으로써 에러를 은닉한다. 아래 식(1-4)은 손실된 매크로블록의 상하좌우에서의 인접화소들간의 차이를 비용값으로 산출한 수식이며 이를 이용하여 산출한 각 움직임 벡터값에서의 최종 비용값은 식 (5)와 같다.

$$BM_T = \sum_{x=x_0}^{x_0+15} |P_{x,y_0-1} - P^r_{x+M_x,y_0+M_y}| \quad (1)$$

$$BM_B = \sum_{x=x_0}^{x_0+15} |P_{x,y_0+16} - P^r_{x+M_x,y_0+15+M_y}| \quad (2)$$

$$BM_L = \sum_{y=y_0}^{y_0+15} |P_{x_0-1,y} - P^r_{x_0+M_x,y+M_y}| \quad (3)$$

$$BM_R = \sum_{y=y_0}^{y_0+15} |P_{x_0+16,y} - P^r_{x_0+15+M_x,y+M_y}| \quad (4)$$

$$Cost_{BM} = BM_T + BM_B + BM_L + BM_R \quad (5)$$

이때 $P_{x,y}$는 현재 프레임의 (x,y) 위치에서의 화소값을 의미하며, $P^r_{x,y}$는 참조프레임의 (x,y) 위치에

서의 화소값을 나타낸다. M_x, M_y는 에러가 발생한 블록에 인접한 주변 매크로블록으로부터 얻어낸 움직임벡터정보이다.

OBMA(Overlapping BMA)의 경우는 인접화소간의 차이값 대신에 참조프레임에서 움직임벡터를 고려한 후 동일 위치간의 차이값을 비용값으로 사용함으로써 화면내의 에지성분을 어느 정도 고려할 수 있다는 장점을 가진다. **OBMA**를 사용하는 경우 손실된 매크로블록의 상하좌우에서의 비용값은 식 (6-9)와 같으며 하나의 움직임 벡터에 대한 최종 비용값은 식 (10)과 같다.

$$OBM_T = \sum_{x=x_0}^{x_0+15} |P_{x,y_0-1} - P^r_{x+M_x,y_0-1+M_y}| \quad (6)$$

$$OBM_B = \sum_{x=x_0}^{x_0+15} |P_{x,y_0+16} - P^r_{x+M_x,y_0+16+M_y}| \quad (7)$$

$$OBM_L = \sum_{y=y_0}^{y_0+15} |P_{x_0-1,y} - P^r_{x_0-1+M_x,y+M_y}| \quad (8)$$

$$OBM_R = \sum_{y=y_0}^{y_0+15} |P_{x_0+16,y} - P^r_{x_0+16+M_x,y+M_y}| \quad (9)$$

$$Cost_{OBM} = OBM_T + OBM_B + OBM_L + OBM_R \quad (10)$$

2.2 H.264/AVC에서의 시방향 에러은닉기법

H.264/AVC 이전의 대부분의 부호화 표준들은 16x16단위로 움직임을 추정함으로써 하나의 매크로블록이 가질 수 있는 움직임 벡터의 수는 하나로 한정되어 있었다. 반면 H.264에서는 움직임 추정을 다양한 블록크기로 수행함으로써 하나의 매크로블록은 최대 16개의 움직임 벡터를 가질 수 있다. 또한 여러 장의 참조프레임을 사용함으로써 각 매크로블록은 참조프레임의 정보도 포함하고 있다. 프레임참조의 최소단위는 8x8 블록으로 수행되기 때문에 하나의 매크로블록이 가질 수 있는 최대 참조프레임의 수는 4개이다.

H.264는 8x8 단위의 참조 프레임 정보 및 4x4 단위의 움직임 벡터정보를 가지는 주변 매크로블록의 정보를 이용하여 에러를 은닉하며 그림 1은 이를 도시(圖示)한다.

그림 1에서 보는 바와 같이 H.264에서 시방향 에러은닉 시 사용하는 움직임 벡터정보로서 상하좌우 방향으로 인접한 8x8 블록의 움직임 벡터를 사용한다. 8x8블록의 크기의 단위로 움직임벡터를 선정하는 주된 이유는 8x8 단위로 참조영상을 선택하는 H.264의 특징 때문이다.

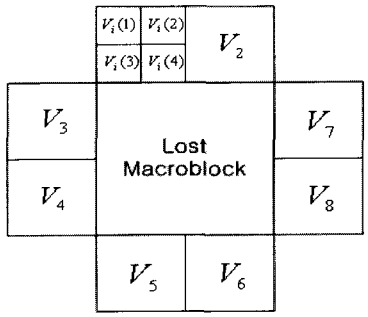


그림 1. 인접 매크로블록의 움직임 벡터정보 및 참조 프레임 정보 $V_i = (M_x, M_y, Ref)$

Fig 1. The motion vectors and reference frames of neighbor macroblocks $V_i = (M_x, M_y, Ref)$

H.264에서의 시방향 에러은닉 기법은 기본적으로 BMA에 기반하며 이때 사용되는 움직임 벡터 후보군은 아래와 같다.

$$V_{M_x,M_y,Ref} = [ZM, V_1, V_2, ..., V_8]$$
$$where, V_i = Mean(V_i(1) + V_i(2) + V_i(3) + V_i(4)) \quad (11)$$

식 (11)의 ZM은 바로 이전 프레임 같은 위치의 블록을 복사하는 경우를 말한다. 식 (11)에서 보는 바와 같이 H.264에서는 시방향 에러은닉 시 하나의 매크로블록에 대해서 최대 9개의 움직임벡터를 사용한다. 만일 슬라이스 에러가 발생한 경우에 우측 매크로블록의 정보는 사용할 수 없게 되며, 좌측 매크로블록의 경우 이미 에러가 은닉된 경우는 7개의 움직임 벡터를 사용하며, 만일 좌측 매크로블록의 정보 또한 없는 경우 5개의 움직임벡터를 사용하여 에러를 은닉하게 된다.

주변 매크로블록으로부터 움직임벡터 후보군이 결정되면 아래 수식을 통하여 에러 매크로블록의 최종 움직임 벡터를 선정하며 이를 이용하여 에러를 은닉하게 된다.

$$(v_x, v_y, ref) = \underset{(M_x, M_y, Ref)}{Arg} \; Min(Cost_{BM}(V_{M_x,M_y,Ref})) \quad (12)$$

식 (12)의 v_x, v_y 및 ref는 비용값을 최소로 하는 움직임 벡터 및 참조 영상을 나타낸다. 비용값을 최소로 하는 움직임 벡터 및 참조영상이 결정되면 H.264는 최종적으로 아래의 수식을 이용하여 에러를 은닉한다.

$$MB_{conceal}(x_i, y_j, t) = MB(x_i + v_x, y_j + v_y, t-1-ref) \quad (13)$$
$$i, j = 0, 1, \cdots, 15 \quad ref = 0 \sim 4$$

식 (13)은 손실된 매크로블록의 주변 매크로블록 정보로부터 최종 움직임 벡터 및 참조프레임이 결정된 후 결정된 참조프레임에서 결정된 움직임벡터에 위치한 블록을 가져옴을 의미한다. 5장의 참조프레임을 사용하는 경우 ref값은 0~4까지의 값 중 하나를 가질 수 있다.

Ⅲ. 제안하는 알고리듬

3.1 H.264 매크로블록의 모드

H.264에서는 이전의 부호화 표준과 다르게 다양한 블록 크기로 움직임을 추정한다. 그림 2는 움직임 추정에 사용되는 다양한 블록 크기를 갖는 매크로블록의 모드를 나타낸다.

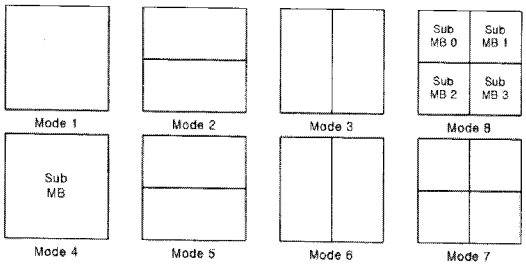


그림 2. 움직임 추정에 사용되는 다양한 크기의 매크로블록 모드

Fig. 2. The macroblock mode with the VBS (Variable Block Size) for the motion estimation

그림 2에서 Mode1, Mode2, Mode3, Mode8은 16x16의 매크로블록 크기를 나타내며, Mode4~Mode7은 8x8크기의 부 매크로블록 크기를 나타낸다. 즉, 하나의 매크로블록은 Mode1, Mode2, Mode3, Mode8의 블록 모양으로 움직임을 추정하며 Mode8이 선택되는 경우 각 부-매크로블록은 다시 Mode4~Mode7의 블록 모양으로 움직임을 추정한다.

H.264에서는 매크로블록의 모드를 선택하기 위해 움직임벡터 비용함수(MVcost), 참조 프레임 비용함수(REFcost) 그리고 율-왜곡 비용함수(RDcost)를 사용한다.

$$MVcost = WeightedCost(f, mvbits[(cx << s) - px] + mvbits[(cy << s) - py]) \quad (14)$$
$$where, f : lambda\ factor$$

움직임 벡터 비용함수는 식 (14)와 같다. 여기에서, cx 및 cy는 현재 매크로블록의 움직임 벡터를 나타내고, s는 움직임 벡터의 추이(推移) 양을 나타내며, px 및 py는 예측된 움직임 벡터를 나타낸다. $WeightedCost$ 함수는 f 값을 사용하여 움직임 벡터 예측 오차의 부호화에 사용되는 비트량을 인수로 받아 비용 값을 반환한다.

$$REFcost = WeightedCost(f, refbits(ref)) \quad (15)$$
$$where, f : lambda\ factor$$

참조 프레임 비용 함수는 식 (15)와 같다. 여기에서 ref는 선택된 참조 프레임을 가리키며, $WeightedCost$ 함수는 f값을 사용하여 선택된 참조 프레임 값을 부호화 하는데 필요한 비트량을 인수로 받아 비용 값을 반환한다. 일반적으로 움직임 벡터 비용에 비하여 매우 적은 값을 갖는다.

$$RDcost = Distortion + lambda \times Rate \quad (16)$$

율-왜곡 비용 함수는 식 (16)과 같다. 율-왜곡 최적화 기법이 사용되는 경우 율-왜곡 비용 함수는 Mode8내의 부-매크로블록 모드의 선택 및 전체 매크로블록 모드의 최종 선택 시 사용되며 이전의 두 비용함수에 비해 상대적으로 높은 복잡도를 갖는다. 식 (16)내의 왜곡(Distortion) 값은 각 매크로블록 모드에서의 SNR을 구함으로써 계산할 수 있으며, 율(Rate) 값은 매크로블록의 부호화가 끝나는 시점에서의 압축된 비트열(bitstream)의 비트율을 나타낸다.

H.264에서의 매크로블록 모드 결정은 율-왜곡 최적화 기법의 사용 여부에 따라 두 가지로 나뉜다. 만일 율-왜곡 최적화 기법이 사용되는 경우에는 율-왜곡 비용값을 최소로 하는 매크로블록 모드를 결정하게 되며 율-왜곡 최적화 기법이 사용되지 않는 경우에는 움직임벡터 비용값과 참조프레임 비용값의 합이 최소가 되는 매크로블록 모드를 선택한다.

3.2 적응적 시방향 에러은닉

앞서 기술한 H.264내의 비용함수 중 율-왜곡 비용값과 움직임벡터 비용값은 어느 정도 비례하는 성질을 가지는데 이는 움직임 추정을 통한 예측오차의 크기가 비트율에 직접적으로 영향을 주기 때문이다. 즉 율-왜곡 최적화 기법이 사용 여부와 관계없이 매크로블록의 모드가 움직임 추정후의 예측오차의 크기와 연관되어 있음을 의미한다. 가령 매

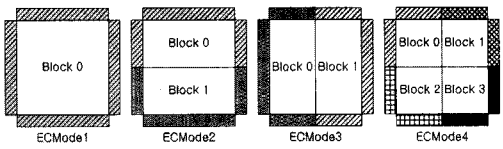


그림 3. 다양한 크기를 가지는 네 가지의 에러은닉모드
Fig. 3. The four error concealment modes with variable block sizes

크로블록이 Mode2로 선택되었다면 두 개의 16x8 블록 크기로 움직임 추정을 수행하는 것이 최소의 예측오차를 발생한다고 기대할 수 있다.

따라서 제안하는 알고리듬은 오류가 발생한 매크로블록의 모드를 인접한 매크로블록의 모드로서 예측하고 이를 이용하여 에러은닉에 사용되는 블록의 크기를 가변적으로 선택함으로써 보다 효율적으로 에러를 은닉한다. 에러은닉 모드는 그림 3과 같이 네 가지 중 하나로 선택되며 각 에러은닉모드는 서로 다른 블록 크기로 에러를 은닉한다.

네 가지의 에러은닉 모드는 서로 다른 블록 단위로 에러를 은닉하고 OBMA와 동일한 방법으로 비용값을 산출하며 각 에러은닉모드에 따라 서로 다른 개수의 참조 화소수를 갖는다.

가. 에러은닉모드-1 (ECMode1)

에러은닉모드-1은 에러가 발생한 매크로블록의 상하 블록중 하나 이상이 세로방향으로의 나눔성분을 가지고 있는 Mode3 또는 Mode8 이외(以外)의 매크로블록 모드를 가지며, 좌우 블록 중 하나 이상이 가로방향으로의 나눔 성분을 가지고 있는 Mode2 또는 Mode8 이외의 매크로블록 모드를 가질 때 선택되며 이 경우 기존의 H.264에서 채택하고 있는 방법과 동일하게 16x16 단위로 에러를 은닉하며 이때 사용되는 비용값은 식 (10)과 동일하다.

비용값 산출에 사용되는 움직임 벡터 후보군 또한 기존의 방법과 동일하게 그림 1에서 나타난 바와 같이 주변 8개 8x8 단위의 평균 움직임 벡터 및 참조프레임과 ZM 벡터를 사용한다.

나. 에러은닉모드-2 (ECMode2)

에러은닉모드-2는 에러가 발생한 매크로블록의 좌우 블록 모두가 가로 방향으로의 나눔 성분을 가지고 있는 Mode2 또는 Mode8의 매크로블록 모드를 가지며, 상하 블록중 하나 이상이 세로 방향으로 나눔 성분을 가지고 있는 Mode3 또는 Mode8 이외의 매크로블록 모드를 가지는 경우 선택된다. 이 경우 에러은닉은 두개의 16x8의 블록 크기로 수행되며 각 세분화된 블록에 대한 비용값은 아래의 수식을 사용한다.

$$Cost_{Block0} = \sum_{x=x_0}^{x_0+15}(|P_{x,y_0-1} - P^r_{x+M_x,y_0-1+M_y}|) + \sum_{y=y_0}^{y_0+7}(|P_{x_0-1,y} - P^r_{x_0-1+M_x,y+M_y}| + |P_{x_0+16,y} - P^r_{x_0+16+M_x,y+M_y}|) \quad (17)$$

$$Cost_{Block1} = \sum_{x=x_0}^{x_0+15}(|P_{x,y_0+16} - P^r_{x+M_x,y_0+16+M_y}|) + \sum_{y=y_0+8}^{y_0+15}(|P_{x_0-1,y} - P^r_{x_0-1+M_x,y+M_y}| + |P_{x_0+16,y} - P^r_{x_0+16+M_x,y+M_y}|) \quad (18)$$

이때 각 블록에 대해 고려하는 움직임 벡터 후보군은 아래와 같다.

$$Block0 : \{ZM, V_1, V_2, V_3, V_7\}$$
$$Block1 : \{ZM, V_4, V_5, V_6, V_8\}$$

다. 에러은닉모드-3 (ECMode3)

에러은닉모드-3은 에러가 발생한 매크로블록의 상하 블록 모두가 세로 방향으로의 나눔 성분을 가지고 있는 Mode3 또는 Mode8의 매크로블록 모드를 가지며, 좌우 블록중 하나 이상이 가로 방향으로 나눔 성분을 가지고 있는 Mode2 또는 Mode8 이외의 매크로블록 모드를 가지는 경우 선택된다. 이 경우 에러은닉은 두개의 8x16 블록 크기로 수행되며 각 세분화된 블록에 대한 비용값은 아래의 수식을 사용한다.

$$Cost_{Block0} = \sum_{y=y_0}^{y_0+15}(|P_{x_0-1,y} - P^r_{x_0-1+M_x,y+M_y}|) + \sum_{x=x_0}^{x_0+7}(|P_{x,y_0-1} - P^r_{x+M_x,y_0-1+M_y}| + |P_{x,y_0+16} - P^r_{x+M_x,y_0+16+M_y}|) \quad (19)$$

$$Cost_{Block1} = \sum_{y=y_0}^{y_0+15}(|P_{x_0+16,y} - P^r_{x_0+16+M_x,y+M_y}|) + \sum_{x=x_0+8}^{x_0+15}(|P_{x,y_0-1} - P^r_{x+M_x,y_0-1+M_y}| + |P_{x,y_0+16} - P^r_{x+M_x,y_0+16+M_y}|) \quad (20)$$

각 블록에 대해 고려하는 움직임 벡터 후보군은 아래와 같다.

$$Block0 : \{ZM, V_1, V_3, V_4, V_5\}$$
$$Block1 : \{ZM, V_2, V_6, V_7, V_8\}$$

라. 에러은닉모드-4 (ECMode4)

에러은닉모드-4는 에러가 발생한 매크로블록의 상하 블록 모두가 세로 방향으로의 나뉨 성분을 가지고 있는 Mode3 또는 Mode8의 매크로블록 모드를 가지며, 좌우 블록 모두가 가로방향으로의 나뉨 성분을 가지고 있는 Mode2 또는 Mode8을 가지는 경우 선택된다. 이 경우 에러은닉은 네 개의 8x8 블록 크기로 수행되며 각 세분화된 블록에 대한 비용값은 아래의 수식을 사용한다.

$$Cost_{Block0} = \sum_{x=x_0}^{x_0+7} (|P_{x,y_0-1} - P^r_{x+M_x,y_0-1+M_y}|) + \sum_{y=y_0}^{y_0+7} (|P_{x_0-1,y} - P^r_{x_0-1+M_x,y+M_y}|) \quad (21)$$

$$Cost_{Block1} = \sum_{x=x_0+8}^{x_0+15} (|P_{x,y_0-1} - P^r_{x+M_x,y_0-1+M_y}|) + \sum_{y=y_0}^{y_0+7} (|P_{x_0+16,y} - P^r_{x_0+16+M_x,y+M_y}|) \quad (22)$$

$$Cost_{Block2} = \sum_{x=x_0}^{x_0+7} (|P_{x,y_0+16} - P^r_{x+M_x,y_0+16+M_y}|) + \sum_{y=y_0+8}^{y_0+15} (|P_{x_0-1,y} - P^r_{x_0-1+M_x,y+M_y}|) \quad (23)$$

$$Cost_{Block3} = \sum_{x=x_0+8}^{x_0+15} (|P_{x,y_0+16} - P^r_{x+M_x,y_0+16+M_y}|) + \sum_{y=y_0+8}^{y_0+15} (|P_{x_0+16,y} - P^r_{x_0+16+M_x,y+M_y}|) \quad (24)$$

각 블록에 대해 고려하는 움직임 벡터 후보군은 아래와 같다.

$$Block0 : \{ZM, V_1, V_3\}$$
$$Block1 : \{ZM, V_2, V_7\}$$
$$Block2 : \{ZM, V_4, V_5\}$$
$$Block3 : \{ZM, V_6, V_8\}$$

3.3 연산량 비교

에러은닉 기법에서는 객관적 화질만큼이나 복잡도 또한 중요한 성능으로 고려되어야 한다. 이는 복호화시에 에러블록을 포함하고 있는 프레임은 에러은닉과정이 끝난 후 다음 프레임을 복호화 하는 과정에서 참조프레임으로서 사용되기 때문이다. 따라서 높은 복잡도를 가지는 에러은닉 알고리듬은 실시간 복호화에 상당한 영향을 미칠 수 있다.

제안하는 알고리듬은 주변 매크로블록의 모드에 따라 네 가지의 블록 크기로 에러은닉을 수행한다. 각 에러은닉모드에 따라 서로 다른 연산량을 가지며 상하좌우의 모든 매크로블록의 정보가 사용가능한 경우 하나의 매크로블록에 대해서 각 에러은닉 모드에 따른 계산양은 아래 표와 같다.

표 1. 에러은닉모드에 따른 연산량 비교
Table 1. The comparison of the complexity of ECModes

ECModes	Addition	Subtraction	Comparison
ECMode1	9	576	9
ECMode2	10	320	10
ECMode3	10	320	10
ECMode4	12	192	12

만일 모든 에러블록이 에러은닉모드-1으로만 선택되는 경우 모든 에러 블록은 16x16 단위로 에러은닉을 수행하며 이 경우 기존의 H.264에서의 시방향 에러은닉 기법과 같은 연산량을 가진다. 에러은닉모드-2, 에러은닉모드-3, 에러은닉모드-4의 경우에는 고려하는 움직임 벡터군을 에러은닉 모드에 따라 제한함으로써 에러은닉모드-1 보다 적은 계산량을 갖는다.

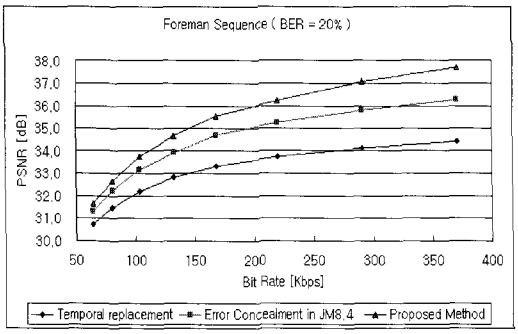
위의 표에서 뺄셈개수의 차이는 단순한 감산회수 이상의 의미를 갖는다. 이는 H.264에서의 움직임 추정이 1/4 화소단위로 수행되며 후보움직임벡터가 1/2 또는 1/4 단위를 가질 때 참조프레임을 보간하여 뺄셈을 수행하기 때문에 보간에 사용되는 복잡도를 함께 고려하면 실제 각 모드간의 복잡도는 많은 차이를 보이게 된다.

따라서 제안하는 알고리듬은 주변 매크로블록의 블록모드를 이용하여 적응적으로 에러은닉 블록 크기를 선택하고 각 에러은닉블록모드에 따라 움직임 벡터 후보군을 제한함으로써 기존의 H.264에서 사용되는 에러은닉기법보다 상대적으로 낮은 복잡도를 갖는다.

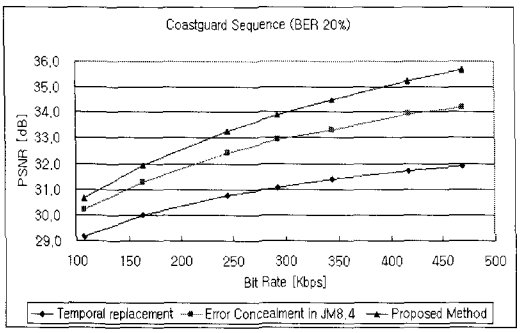
Ⅳ. 실험 및 결과 분석

알고리듬의 타당성을 보이기 위한 실험은 QCIF 크기의 시퀀스 17개에 대해서 제안하는 알고리듬을

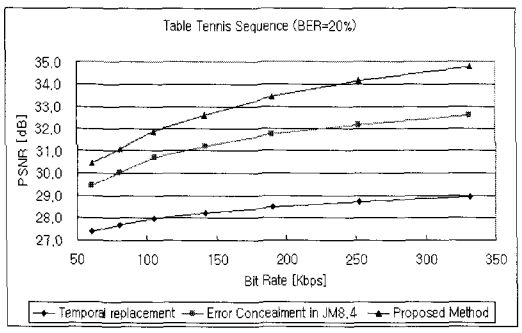
적용하여 에러은닉을 수행하였으며 본 논문에서는 비교적 큰 움직임을 가지는 Foreman, Coastguard, Table tennis시퀀스에 대한 결과를 기술한다. 움직임의 거의 없는 news, claire등의 영상의 경우 시방향 블록 복사기법 또는 기존의 H.264내의 시방향 에러은닉 기법의 사용만으로 좋은 성능을 보이며 그 경우에도 제안하는 알고리듬은 기존의 방법보다 높은 성능을 나타낸다. 제안하는 알고리듬은 H.264 참조 소프트웨어(JM 8.4)에 적용하였으며 시방향 에러은닉의 성능을 비교하기 위해 각 시퀀스의 최초 100 프레임을 IPPP 구조로 부호화 한 결과를 복호기 입력으로 사용하였다. 그림 4는 매크로블록의 오류율(誤謬率)이 20%일때 각 시퀀스에서의 비트율에 대한 오류 은닉후의 PSNR을 나타내며 비교를 위하여 시방향 블록 복사 및 H.264내의 시방향 에러은닉을 사용한 경우를 함께 도시하였다.



(a) Foreman Sequence 100 프레임 (BER=20%)



(b) Coastguard Sequence 100 프레임 (BER=20%)



(c) Table tennis Sequence 100 프레임 (BER=20%)

그림 4. 매크로블록 오류율이 20%일때의 실험결과
Fig 4. The simulation results (BER=20%)

그림 4의 결과에서 보는바와 같이 제안한 알고리듬은 비트율에 상관없이 시방향 블록 복사 기법뿐만 아니라 H.264에서의 시방향 에러은닉방법 보다도 높은 성능을 보이는 것으로 나타난다.

아래의 표 2~표 4는 서로 다른 비율의 매크로블록 오류가 발생했을때 BER에 따른 PSNR을 나타낸다.

표 2. Foreman 영상에 대한 BER에 따른 PSNR
Table 2. The PSNRs of Foreman Sequence (100frame) for variable BERs

QP	Original	PSNR	BER			
			5%	10%	15%	20%
24	39.07	$PSNR_{ZM}$	36.79	35.80	34.71	33.74
		$PSNR_{JM}$	37.51	36.77	36.12	35.26
		$PSNR_{PRO}$	38.06	37.55	36.89	36.25
22	40.64	$PSNR_{ZM}$	37.67	36.48	35.22	34.14
		$PSNR_{JM}$	38.58	37.67	36.80	35.82
		$PSNR_{PRO}$	39.33	38.68	37.80	37.06
20	42.06	$PSNR_{ZM}$	38.33	36.98	35.59	34.41
		$PSNR_{JM}$	39.46	38.36	37.41	36.29
		$PSNR_{PRO}$	40.38	39.64	38.58	37.73

$PSNR_{ZM}$: *Temporal Replacement*
$PSNR_{JM}$: *Error Concealment in JM*8.4
$PSNR_{PRO}$: *Proposed Method*

표 3. Coastguard 영상에 대한 BER에 따른 PSNR
Table 3. The PSNRs of Coastguard Sequence (100frame) for variable BERs

QP	Original	PSNR	BER			
			5%	10%	15%	20%
26	35.53	$PSNR_{ZM}$	33.81	32.61	32.18	31.39
		$PSNR_{JM}$	34.86	34.06	33.79	33.33
		$PSNR_{PRO}$	35.27	34.82	34.70	34.49
25	36.54	$PSNR_{ZM}$	34.46	33.10	32.62	31.73
		$PSNR_{JM}$	35.74	34.81	34.46	33.94
		$PSNR_{PRO}$	36.22	35.65	35.48	35.23
24	37.17	$PSNR_{ZM}$	34.85	33.40	32.87	31.93
		$PSNR_{JM}$	36.19	35.15	34.81	34.21
		$PSNR_{PRO}$	36.80	36.14	35.95	35.68

표 4. Table tennis 영상에 대한 BER에 따른 PSNR
Table 4. The PSNRs of Table tennis Sequence (100frame) for variable BERs

QP	Original	PSNR	BER			
			5%	10%	15%	20%
28	35.56	$PSNR_{ZM}$	32.11	30.51	29.64	28.49
		$PSNR_{JM}$	34.05	33.04	32.57	31.74
		$PSNR_{PRO}$	34.80	34.18	33.97	33.44
26	36.83	$PSNR_{ZM}$	32.65	30.88	29.96	28.72
		$PSNR_{JM}$	34.85	33.57	33.13	32.18
		$PSNR_{PRO}$	35.79	35.01	34.83	34.15
24	38.26	$PSNR_{ZM}$	33.19	31.25	30.27	28.94
		$PSNR_{JM}$	35.71	34.27	33.73	32.62
		$PSNR_{PRO}$	36.89	35.87	35.57	34.79

위의 결과에서 보는바와 같이 제안하는 알고리듬은 모든 비트율 및 BER에 대해서 기존의 방법에 비교하여 최대 2.17 dB까지의 객관적 화질 향상을 가져온다.

Ⅳ. 결 론

H.264 내의 매크로블록은 4x4 블록 단위까지의 움직임 벡터 및 8x8 단위의 참조프레임, 그리고 움직임추정 및 보상에 사용하는 블록의 크기를 정의하는 매크로블록 모드의 정보를 포함한다. 이전의 부호화 표준에서 보다 많은 정보를 포함하는 H.264 표준의 부호화 특성을 고려하여 추가된 정보들을 시방향 에러은닉 기법에 활용함으로써 보다 효율적인 시방향 에러은닉 기법을 제안하였다.

제안한 알고리듬은 손실된 매크로블록의 주변 매크로블록이 포함하고 있는 정보를 이용하여 에러를 은닉하고자 하는 매크로블록의 모드를 추정하고 이를 이용하여 에러은닉에 사용되는 블록의 크기를 적응적으로 선택함으로써 H.264에서의 시방향 에러은닉 기법보다도 적은 연산량을 가지면서도 보다 높은 객관적 화질을 나타낸다.

참 고 문 헌

[1] JVT G050r1, "Draft ITU-T recommendation and final draft international standard of joint video specification (ITU-T Rec. H.264/ISO/IEC 14496-10 AVC)," May 2003.

[2] Thomas Wiegand, Heiko Schwarz, Anthony Joch, Faouzi Kossentini, "Rate-Constrained Coder Control and Comparison of Video Coding Standard," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 13, pp. 688-703, July 2003.

[3] L. H. Kieu and D. N. Ngan, "Cell-loss concealment techniques for layered video codecs in an ATM networks," *IEEE Trans. Image Processing*, vol. 3, no. 5, pp. 666-677, 1994

[4] J. Suh and Y. Ho, "Recovery of motion vectors for error concealment," *IEEE TENCON*, pp. 750-753, June 1999.

[5] D, Kwon and P. Driessen, "Error concealment techniques for H.263 video transmission," *IEEE*, pp. 276-279, February 1999.

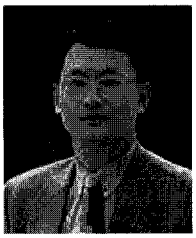
[6] Young H. Jung, Yong-goo Kim, and Yoonsik Choe, "Robust error concealment algorithm using iterative weighted boundary matching criterion," *Proc. ICIP*, pp. 384-387, 2000.

[7] Jinghong Zheng and Lap-Pui Chau, "A motion vector recovery algorithm for digital video using lagrange interpolation," *IEEE Trans. on Broadcasting*, vol. 49, pp. 383-389, Dec. 2003.

김 동 형(Donghyung Kim) 정회원

1999년 2월 충북대학교 전자공학과 졸업
2001년 8월 충북대학교 전자공학과 석사
2002년 3월~현재 한양대학교 전자통신전파공학과 박사과정
<관심분야> 영상처리 및 영상압축

정 제 창(Jechang Jeong) 정회원



1980년 2월 서울대학교 전자공학과 졸업
1982년 2월 KAIST 전기전자공학과 석사
1990년 미국 미시간대학 전기공학과 공학박사
1980~1986 KBS 기술연구소 연구원(디지털 TV및 뉴미디어 연구)
1990~1991 미국 미시간대학 전기공학과 연구교수 (영상 및 신호처리 연구)
1991~1995 삼성전자 멀티미디어 연구소(MPEG, HDTV, 멀티미디어 연구)
1995~현재 한양대학교 전자전기컴퓨터공학부 교수 (영상통신 및 신호처리 연구실)
1998년 11월 27일 과학기술자상 수상
1998년 12월 31일 정보통신부장관상 표창
<관심분야> 영상처리 및 영상압축