

화상통신에서의 오류전파 제어를 위한 보조모션벡터 코딩기법

이 주 경[†] · 최 태 옥[†] · 정 기 동^{††}

요 약

인터넷을 통해 동영상상을 전송할때, 네트워크의 상태에 따라 패킷 손실이 발생할 수 있다. 패킷 손실이 발생하면 해당 프레임 뿐아니라, 그 프레임에 참조하는 다른 프레임의 화질에도 영향을 미치는 오류전파가 발생한다. 본 논문에서는 전송 중 패킷 손실로 인한 오류전파를 최소화하기 위한 부호화 기법인 AMV(Auxiliary Motion Vector)를 제안한다. AMV 기법에서 부호기는 한 개의 매크로블록을 압축하기 위해 여러 프레임과의 움직임 예측을 수행한 후, 상위 두 개의 매칭블록을 선택한다. 두 블록 중 압축 대상 매크로블록과 더 유사한 블록을 움직임 보상에 이용하며 이 블록을 참조블록 또는 기본블록이라 한다. 나머지 블록을 보조블록이라 하며 참조블록이 손실된 경우 부호기가 참조블록의 대체용으로 이용한다. 보조블록의 정보는 압축된 매크로블록의 헤더에 블록의 모션벡터와 프레임 번호를 삽입함으로써 전송된다. 이 기법은 다른 기법에 비해 압축에 이용되는 참조블록의 수를 최소화하고 보조블록을 이용하여 오류를 복구함으로써 오류전파가 발생하는 범위와 그 강도를 줄이는 장점이 있다. 제안된 기법의 코덱을 구현하기 위해 H.263 표준 소스를 수정하였으며, 다양한 패킷 손실율의 트레이스를 생성하기 위해 시뮬레이션 툴인 NS-2를 이용하였다. 실험 결과 제안된 기법은 전체적으로 H.263 표준에 비하여 높은 성능을 나타냈으며 특히 화면의 변화가 적은 경우 패킷 손실율이 높을수록, 변화가 심한 경우 패킷 손실율이 낮을수록 높은 성능을 나타내었다.

Exploitation of Auxiliary Motion Vector in Video Coding for Robust Transmission over Internet

Joo-Kyong Lee[†] · Tae-Uk Choi[†] · Ki-Dong Chung^{††}

ABSTRACT

In this paper, we propose a video sequence coding scheme called AMV (Auxiliary Motion Vector) to minimize error propagation caused by transmission errors over the Internet. Unlike the conventional coding schemes the AMV coder, for a macroblock in a frame, selects two best matching blocks among several preceding frames. The best matching block, called a primary block, is used for motion compensation of the destination macroblock. The other block, called an auxiliary block, replaces the primary block in case of its loss at the decoder. When a primary block is corrupted or lost during transmission, the decoder can efficiently and simply suppress error propagation to the subsequent frames by replacing the block with an auxiliary block. This scheme has an advantage of reducing both the number and the impact of error propagations. We implemented the proposed coder by modifying H.263 standard coding and evaluated the performance of our proposed scheme in the simulation. The simulation results show that AMV coder is more efficient than the H.263 baseline coder at the high packet loss rate.

키워드 : 보조모션벡터(Auxiliary Motion Vector), 화상 압축(Video Coding), 화상 탄력성(Error Resilience), 오류 전파(Error Propagation), 화상통신(Robust Transmission)

1. 서 론

동영상 압축을 위해 발표된 표준 중에서 MPEG-1, H.261 과 같은 초기 버전의 표준은 동영상 데이터를 저장장치에 효율적으로 저장하는 것에 목표를 두고 있었다. 그러나 인터넷이 대중화되고 동영상의 실시간 전송 비율이 증가함에 따라, 수신된 동영상의 화질에 대한 요구가 매우 높아졌다. 최근 발

표된 H.263+, MPEG-4 표준에는 유무선 네트워크상으로 전송되는 동영상의 화질을 유지하기 위한 다양한 전송 오류제어 기법이 포함되어 있다. 일반적으로 동영상을 압축하기 위하여 프레임내 데이터의 유사성 및 프레임 간의 유사성을 이용하여 DCT와 움직임 보상을 수행한다. 그러나 이러한 기법은 프레임의 일부에서 발생한 오류가 프레임 내의 다른 영역 뿐아니라 연관된 다른 프레임에도 영향을 미치는 단점이 있다. 즉, 네트워크를 통하여 전송되는 동영상의 일부 패킷이 손실되거나 오류가 발생하면 패킷이 속한 프레임 뿐아니라 이후 도착하는 프레임의 화질에도 영향을 미치게 된다. 이러

† 준 회원 : 부산대학교 대학원 전자계산학과
 †† 종신회원 : 부산대학교 전자계산학과 교수
 논문접수 : 2002년 7월 27일, 심사완료 : 2002년 10월 25일

한 문제를 해결하기 위해 전송오류에 관한 많은 연구가 진행되어 왔으며, 그 가운데 일부가 표준으로 채택되기도 하였다.

네트워크상에서 전송 중 발생한 오류를 제어하는 기법은 수행 주체에 따라 부호기에서의 소스코딩 기법, 복호기에서의 오류은닉 기법, 부호기와 복호기가 상호 작용을 수행하는 피드백 채널 기법 등으로 나눌 수 있다. 소스코딩 기법은 압축과정을 거쳐 생성된 비트열(bit-streams)이 오류에 잘 견디도록 코딩하여 프레임 내의 오류전과 영역을 제한하는 기법이다. 대표적인 예로 RVLC[1], EREC[2], ISD[3] 기법을 들 수 있다. RVLC(reversible VLC) 기법은 비트열을 순방향과 역방향으로 모두 복원 가능하도록 한 VLC 기법이다. 이 기법은 비트열에 오류가 발생하면 오류 발생 지점부터 동기화 정보 지점까지의 모든 비트열을 버리는 기존 기법과 달리 동기화 정보 지점에서 오류 발생지점까지 역방향으로 데이터 복원이 가능하다. MPEG-4 표준에서 이 기법을 채용하고 있다. EREC(Error Resilient Entropy Coding) 기법은 VLC 코딩된 블록들을 재배치하여 오류 전과를 제어하는 기법이다. 즉, 길이가 서로 다른 블록들을 주어진 크기의 슬롯에 맞추고 블록의 일부가 남으면 비어있는 슬롯에 삽입하여 재배치함으로써 오류전과를 막는 기법이다. ISD(Independent Segment Decoding)는 H.263+의 옵션으로서 블록이 참조하는 이전 프레임의 영역을 한 개의 GOB(Group of Block)로 제한한다. 이 방법은 참조블록에 오류가 발생하더라도 오류는 한 개의 GOB만 영향을 받도록 한다. 위 세 가지 기법은 압축 효율은 낮아지지만 오류 발생시 화질을 유지할 수 있는 장점이 있다. 수신자의 복호기가 수행하는 오류은닉 기법[4,5]은 블록에 오류가 발생하면 프레임 내의 이웃한 블록 또는 직전 프레임의 블록을 복사하는 기법이다. 오류은닉 기법은 소스코딩이나 피드백 채널 기법을 적용하기 위한 보조수단으로 사용된다. 피드백 채널 기법의 대표적인 예로 RPS(Reference Picture Selection)가 있다. 이 기법에서는 수신자의 복호기가 오류를 발견하면 NACK 메시지를 송신자에 전달한다. NACK 메시지를 받은 송신자의 부호기는 오류가 발생한 프레임 이전에 압축되었던 프레임 중 한 개를 선택하여 현재 프레임의 압축에 이용함으로써 오류전과를 막는다[6,7]. 그러나 이 기법도 NACK 메시지가 수신자에서 송신자로 전송되는 시간동안 발생하는 오류전과를 피할 수 없다[8].

본 논문에서는 복호기가 발생한 오류에 효율적으로 대처하도록 지원하는 소스 코딩 기법인 AMV를 제안한다. 이 기법은 매크로블록을 압축하기 위해 직전 K개의 프레임과 움직임 예측을 수행하여 매크로블록과 가장 유사한 두 개의 블록을 선택한다. 연속적으로 발생하는 패킷 손실에 잘 견딜 수 있도록 두 블록은 서로 다른 프레임에서 선택한다. 선택된 블록중 매크로블록과 더 유사한 블록을 이용하여 움직임

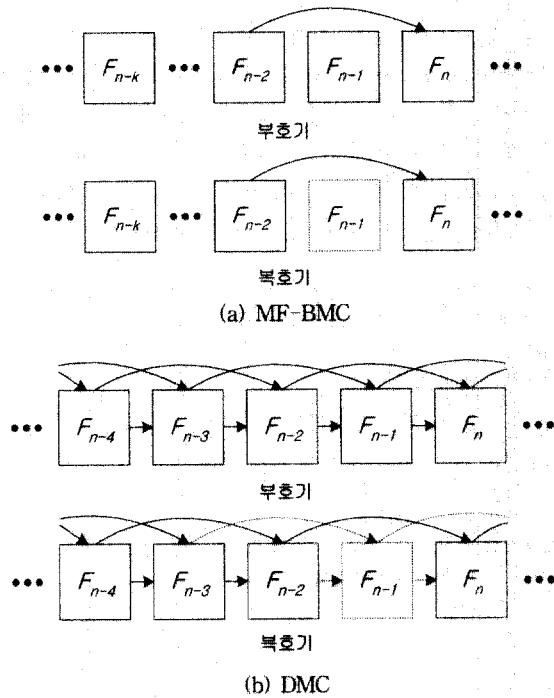
보상을 수행하며 나머지 블록은 움직임 보상에 사용된 블록에 오류가 발생한 경우에 대체되도록 매크로블록의 헤더에 모션벡터와 그 블록의 프레임 번호를 삽입한다. 만약 움직임 보상에 사용된 블록이 전송도중 오류가 발생한다면 송신자의 복호기에서 매크로블록의 헤더에 삽입되었던 두 번째 블록의 모션벡터를 이용하여 원래 매크로블록에 가까운 이미지를 복구할 수 있다. 움직임 보상에 사용한 블록과 보조용 블록이 동일하지 않으므로 원래 매크로블록을 완벽하게 복구할 수는 없지만 여러 프레임에서 원래 매크로블록과 가장 유사한 두 블록을 선택했으므로 단순하면서도 쉽게 원래 매크로블록에 가깝게 복구할 수 있다.

AMV 코더를 구현하기 위해 H.263 TMN 10 소스 코드를 수정하였으며, NS-2를 이용하여 다양한 패킷 손실율의 인터넷 트레이스를 구하였다. 본 논문은 다음과 같이 구성된다. 2장에서는 오류제어를 위해 여러 프레임 또는 블록에서 움직임 예측을 시도한 기존 연구에 관하여 기술하였으며, 3장은 AMV 기법에 관한 상세한 설명을 하였고, 4장은 AMV 부호기/복호기 구현 및 실험 결과를 보인다. 마지막으로 5장은 결론과 향후 과제를 밝힌다.

2. 관련 연구

네트워크에서 전송중 발생하는 오류가 이후 영상에도 영향을 미치는 동영상의 오류전과를 막기 위해 한 개 이상의 프레임 또는 블록을 움직임 예측 및 움직임 보상에 사용하는 소스코딩 기법이 연구되었다. 첫째로, M. Budagavi[9]는 여러 개의 후보 프레임과 움직임 예측을 수행하는 MF-BMC(Multiframe BMC) 기법을 제안했다. MF-BMC는 현재 프레임의 블록을 압축하기 위해 직전 K개의 프레임과 움직임 예측을 수행하고, 그중 가장 유사한 블록을 선택하여 움직임 보상을 수행한다. 이 기법은 한 프레임 내의 블록들이 서로 다른 프레임 참조함으로써 전송오류로 인한 오류 전과 발생 확률을 감소시킬 수 있으며, 여러 프레임중에서 가장 유사한 블록을 선택함으로써 압축 효율도 높일 수 있다고 한다. 그러나, 특정 프레임에서 오류가 발생하면 그 프레임을 참조하는 블록의 오류 전과를 막을 수 없다는 단점이 있다. C.S Kim[10]은 움직임 보상에 사용되는 모션벡터를 두 개 사용하는 DMC(Double-Vector Motion Compensation) 기법을 제안했다. 이 기법에서는 현재 프레임의 직전 두 프레임과 움직임 예측을 수행하고, 각 프레임에서 압축 대상 매크로블록과 가장 유사한 블록을 선택한다. 선택된 두 블록의 가중치 평균을 구하고 이 값으로 매크로블록과의 움직임 보상을 수행한다. 이 기법의 장점은 참조 블록을 두 개 사용함으로써 한쪽 블록에 오류가 발생하더라도 나머지 블록으로

원래 매크로블록을 복구할 수 있다는 것이다. 복구된 매크로블록의 화질은 가중치에 따라 달라질 수 있다. DMC는 두 블록을 모두 참조하므로 그중 한 개에 오류가 발생해도 적은 양의 오류전파는 불가피하다는 단점이 있다. 마지막으로, Y. Yu[11]는 DMRP(Dynamic Multi-Reference Prediction) 기법을 제안했다. 이 기법은 두 개의 블록을 선택하여 그 평균으로 움직임 보상을 수행한다는 면에서 DMC와 유사하고, 여러 프레임에서 두 블록을 선택한다는 점은 다르다. 여러 후보 프레임에서 참조 블록을 선택하므로 동일한 오류가 발생한다면 DMRP는 DMC보다 오류 전파 가능성이 낮다. 그러나 이 기법 역시 두 블록의 평균을 이용하므로 어느 한쪽 블록에서 오류가 발생하더라도 오류는 전파된다. (그림 1)은 MF-BMC와 DMC 기법의 예를 보여준다. (a)에서 프레임 F_n 은 K 개의 후보 프레임중 가장 유사한 F_{n-2} 를 참조함으로써 직전 프레임 F_{n-1} 의 손실에 관계없이 정상적인 복원이 수행된다. DMC를 적용한 (b)의 경우 각 프레임은 직전 2개의 프레임 모두 움직임 보상에 이용하므로 직전 프레임 F_{n-1} 에 손실이 발생하여도 F_{n-2} 를 이용하여 복구 가능하다.



(그림 1) MF-BMC와 DMC 기법의 압축 과정 및 오류 발생에 따른 복구 예

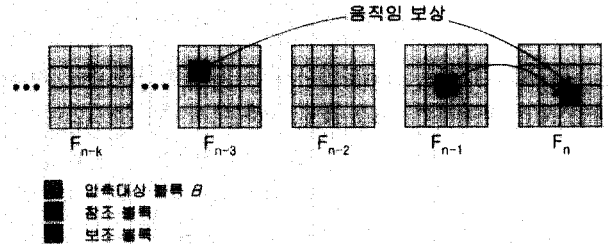
3. AMV를 이용한 오류 제어

관련 연구에서 살펴본 바와 같이 오류전파를 줄이기 위한 방법으로 후보 프레임의 범위를 늘이거나 참조블록의 수를 증가시켜 오류 전파를 줄이는 기법들이 연구되었다. 특히, DMC와 DMRP는 매크로블록당 참조블록 수가 많아져 한

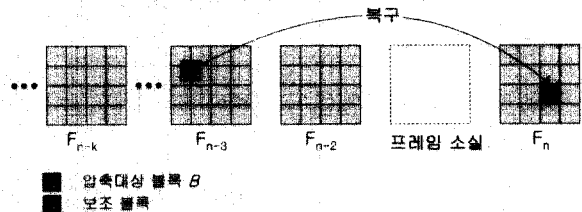
번에 전파되는 오류의 양은 감소하지만 동일 패킷 손실율에 대해 오류 전파의 범위는 더 넓어지게 된다. 이를 해결하기 위해 본 논문에서는 후보 프레임의 범위를 넓히되 참조블록의 수를 한 개로 제한하는 AMV(Auxiliary Motion Vector) 기법을 제안한다. 즉, AMV에서는 연속적인 패킷 오류 발생에 대비해 움직임 예측에 사용되는 후보 프레임의 수를 증가시키고 실제 움직임 보상에 사용되는 참조블록을 한 개로 제한하여 적은 양의 오류 전파가 자주 발생하는 것을 방지한다. 그러나 참조블록에 오류가 발생하면 오류전파가 발생하므로 참조블록과 유사한 보조블록을 이용하여 수신 측에서 복구하도록 한다.

3.1 AMV의 예

(그림 2)에서 현재 프레임 F_n 의 매크로블록 B 를 압축하기 위해 K 개의 후보 프레임과 움직임 예측을 각각 수행하여 각 프레임마다 B 와 가장 유사한 블록을 선택한다. 선택 기준은 3.2절에서 상세히 설명한다. 선택된 K 개의 블록 중에서 B 와 가장 유사한 블록 두 개를 다시 선택한다. (그림 2)에서 선택된 블록은 F_{n-1} 과 F_{n-3} 프레임에 속하며 각각 B_{n-1} 과 B_{n-3} 라고 가정한다. 선택된 두 블록중 B 와 더 유사한 B_{n-1} 의 블록과 움직임 보상을 수행하여 압축하고, 압축된 B 의 헤더에 나머지 블록인 B_{n-3} 의 모션 벡터를 삽입한다. 이때, B_{n-1} 을 참조블록 또는 기본블록이라 하고 복호기가 참조블록을 이용할 수 없는 경우에 사용될 B_{n-3} 을 보조블록이라 한다. (그림 3)은 수신측 복호기가 복호화를 수행할때 참조블록 B_{n-1} 이 손실되어 B 를 제대로 복호화 할 수 없는 경우를 보여준다. 이 경우 복호기는 보조블록 B_{n-3} 을 사용해 원래 블록 B 에 가까운 값을 복구한다.



(그림 2) 부호기의 참조블록을 이용한 압축과 보조 블록의 삽입



(그림 3) 복호기의 보조블록을 이용한 블록 B 의 복구

표준 동영상 압축 알고리즘과 유사하게 AMV는 움직임 예측을 통해 참조블록을 선택하고 움직임 보상을 수행한다. AMV는 여러 후보 프레임에서 움직임 예측을 수행하므로 연속적인 전송오류 발생시 전파되는 오류를 줄일 수 있으며 실제 움직임 보상에 사용되는 블록은 한 개이므로 여러 참조블록을 사용하는 경우보다 오류 전파 가능성을 낮출 수 있다. 그러나 후보 프레임의 수가 많아지면 참조블록과 보조블록 선택을 위한 계산량이 증가하므로 계산시간과 화질을 모두 만족하는 후보 프레임의 수인 K 를 결정해야 한다.

3.2 부호기에서의 처리과정

식 (1)은 기존의 H.263에서 참조 블록을 선택하기 위한 비융합수인 SAD(Sum of Absolute Difference)를 여러 후보 프레임으로 확장 적용한 식이다. SAD 함수 값은 압축대상 매크로블록과 후보 프레임내 특정 블록과의 차이 값에 절대치를 적용하고 합을 계산한 것이다. 현재 프레임을 F_n , 압축대상 매크로블록을 B_i 이라고 하고, B_i 의 크기를 $b \times b$ 픽셀이라고 하자. 여기에서 l 은 F_n 내 매크로블록의 순서를 의미한다. $F_n(x+u, y+v)$ 는 프레임의 좌측 상단을 기준으로 $(x+u, y+v)$ 만큼 떨어진 곳의 픽셀 값을 의미한다. (x, y) 는 B_i 의 좌측 상단의 좌표를 의미하고, (u, v) 는 B_i 내에서 (x, y) 로부터 떨어져 있는 픽셀의 상대 위치를 나타낸다. k 번째 후보 프레임의 움직임 예측 범위는 $F_{n-k}(x+i+u, y+j+v)$ 와 같이 나타낼 수 있다. 여기에서 i, j 는 움직임 예측 범위인 $-p \leq i, j \leq p$ 영역에 존재하게 된다. 3.1절에서 언급한 바와 같이 후보 프레임의 범위인 K 를 신중히 선택해야 한다.

$$SAD_k(i, j) = \sum_{u=0}^{b-1} \sum_{v=0}^{b-1} |F_n(x+u, y+v) - F_{n-k}(x+i+u, y+j+v)| \quad (1)$$

$1 \leq k \leq K, -p \leq i, j \leq p$

식 (1)은 각 후보 프레임마다 움직임 예측 범위내에 존재하는 모든 블록과의 SAD 값을 구하는 식이다. 식 (2)는 식 (1)에서 계산된 SAD 값에 대해 각 프레임마다 최소 값을 선택한다. k 번째 프레임의 최소 값을 sad_k 라고 하고 K 개의 sad_k 집합을 S 라 지정한다. 요약하면, 식 (2)에서는 후보 프레임마다 B_i 과 가장 유사한 블록을 선택하고 그 집합을 S 라 한다. 식 (3)에서는 집합 S 의 원소들을 SAD 값을 기준으로 오름차순으로 정렬한다.

$$sad_k = \text{Arg min } SAD_k(i, j), 1 \leq k \leq K, -p \leq i, j \leq p \quad (2)$$

$$S = \{sad_1, sad_2, \dots, sad_k\}$$

$$Ssad^1 \leq Ssad^2 \leq \dots \leq Ssad^K, Ssad^K \in S \quad (3)$$

움직임 보상을 수행하기 위해 식 (3)에서 B_i 과 가장 유사한

두 블록의 SAD 값 즉, $Ssad^1$ 과 $Ssad^2$ 를 선택하고, 해당 블록을 각각 기본블록과 보조블록이라 한다. 기본블록을 $B_i^?$ 라고 정의하고, B_i 을 위한 움직임 보상에 이용한다. 보조블록을 B_i^* 라고 정의하며 B_i^* 에 전송중 오류가 발생한 경우에만 복호기가 B_i 의 복구용으로 이용한다. 부호기는 복호기가 보조블록을 사용할 수 있도록 보조블록의 정보를 함께 전송해야 하는데, 압축된 B_i 의 헤더에 보조블록의 프레임 번호와 모션 벡터를 함께 삽입하여 전송한다. 이 때문에 표준 코덱에 비해 데이터의 크기가 증가하는 단점이 있지만, 동영상의 비트율이 증가할수록 이 오버헤드는 상대적으로 줄어들고, 패킷 손실이 발생하면 일반 동영상에 비해 높은 화질을 유지할 수 있다. 마지막으로 부호기는 B_i 과의 움직임 보상($B_i - B_i^*$), DCT, 엔트로피 코딩을 수행한다. 이 과정은 H.263 표준의 DCT와 엔트로피 코딩 과정을 따르기로 한다.

(그림 4)는 프레임내 매크로블록 압축 과정을 의사코드로 작성한 것이다. 함수 $\text{min_sad}()$ 는 현재 매크로블록과 후보 프레임들 입력받아 매크로블록과 가장 유사한 블록을 되돌려준다. MV 는 식 (2)에서 생성된 집합 S 의 자료구조로 $\text{min_sad}()$ 의 결과로 선택된 블록의 위치와 SAD 값으로 구성된다. MV 에서 참조블록을 선택한 후, 부호기는 압축모드를 결정해야 한다. 즉, 매크로블록에 대해 프레임 간 압축(inter-type coding)을 수행할지, 프레임내 압축(intra-type coding)을 수행할지 결정해야 한다. 이 모드는 압축효율에 따라 결정되며 프레임 간 압축이 효율적인 경우에만 AMV가 적용되고 그렇지 않으면 프레임 내 압축을 수행한다.

```

for each macroblock  $B_i$  in a frame  $F_n$ 
   $MV \leftarrow \emptyset, k = 1; B_i^? \leftarrow \emptyset$ 
  for preceding  $K$  frames
     $MV \leftarrow MV \cup \text{min\_sad}(B_i, F_{n-k})$ 
     $k = k + 1$ 
  end for /* 각 후보프레임과 움직임 예측 수행 */
   $Pblock = \text{min\_mv}(MV)$  /* 참조블록 선택 */
   $MV \leftarrow MV - Pblock$ 
   $Ablock = \text{min\_mv}(MV)$  /* 보조블록 선택 */
   $mode = \text{determine\_coding\_mode}(Pblock)$ 
  if ( $mode = \text{INTER\_TYPE}$ )
     $B_i^? = \text{insert\_AMV}(B_i^?, Ablock)$  /* 보조모션벡터 삽입 */
     $B_i^? \leftarrow B_i^? + DCT(B_i - B_i^?)$  /* 움직임보상과 DCT */
  else
     $B_i^? = \text{intra\_coding}(B_i)$ 
  end for
    
```

(그림 4) 부호기에서의 AMV 알고리즘

3.3 복호기에서의 처리과정

부호기에서 압축된 동영상이 인터넷과 같은 네트워크를 통하여 수신자에게 전송되면 복호기는 부호화 과정의 역으로 작업을 수행한다. 이를 복호화라 하며 엔트로피 디코딩과 IDCT

를 수행하여 움직임 보상된 블록($B_i - B_i^f$)에 이미 도착한 참조블록 B_i^f 를 합하여 원래의 매크로블록에 가까운 \overline{B}_i 을 복원하게 된다. 이 과정은 H.263 표준 복호화 과정과 동일하며 차이점은 참조블록 B_i^f 에 오류가 발생하여 더 이상 블록 B_i 을 복원할 수 없을때 블록 \overline{B}_i 과 유사한 블록으로 복구하는 방법이 있다. 식 (4)는 AMV 기법의 복호화 방법을 보여준다. 복호기는 기본블록 B_i^f 이 정상이면 이 블록을 이용하여 원래 블록으로 복원한다. 그러나 B_i^f 에 오류가 발생하면 그 블록이 속한 프레임에 간단한 오류은닉 기법을 적용하고, \overline{B}_i 을 복원하기 위해 B_i^f 대신 B_i^c 를 이용함으로써 원래 매크로블록 이미지에 가깝게 복구한다. 만약, B_i^c 에도 오류가 발생했다면, 원래 블록에 가까운 이미지를 복원할 수 없게 된다. 그러나 식 (2)와 식 (3)에 의해 참조블록과 보조블록이 서로 다른 프레임에 속하므로 두 블록에 모두 오류가 발생할 확률은 매우 낮다. 만약 그런 경우가 발생한다면 B_i^f 에 대한 오류 은닉 기법만 적용한다.

$$\overline{B}_i = \begin{cases} IDCT + B_i^c, & \text{only if } B_i^f \text{ is corrupted} \\ IDCT + B_i^f, & \text{otherwise} \end{cases} \quad (4)$$

(그림 5)에서는 AMV 복호기의 복호화 과정을 의사코드로 나타내었다.

```

for each compressed block  $B_i^f$ 
  if ( $B_i^f$  is uncorrupted)
     $\overline{B}_i = IDCT(B_i^f) + B_i^f$ 
  else if ( $B_i^c$  is uncorrupted)
     $\overline{B}_i = IDCT(B_i^c) + B_i^c$ 
  else
     $\overline{B}_i = IDCT(B_i^f) + conceal\_block(B_i^c)$ 
end for
    
```

(그림 5) 복호기의 복호과정

4. 구현 및 실험 결과

제안된 기법의 성능 평가를 위해 표준 H.263의 소스코드를 수정하여 참조블록과 보조블록을 선택하는 부분을 추가하였다. 복호기는 참조블록에서 오류를 발견하면 보조블록을 이용하여 복호화를 수행한다. 실험을 단순화하기 위해 GOB 단위로 패킷을 생성하고, 오류 발생도 패킷단위로 발생한다고 가정한다. H.263 표준에서 QCIF 화일의 GOB는 11개의 매크로블록으로 구성되며 GOB마다 동기화 정보를 삽입할 수 있다. 따라서 본 연구의 실험에서는 한 프레임이 9개의 패킷으로 생성된다.

<표 1> 실험에 사용되는 동영상의 특징

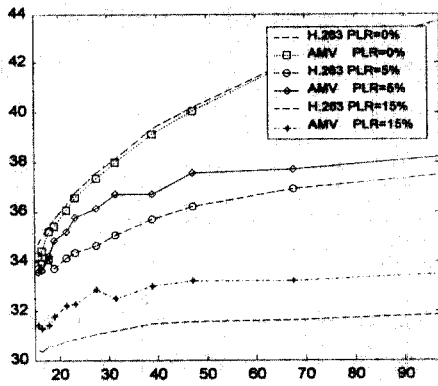
파일명	akiyo.qcif	foreman.qcif
동영상 포맷	QCIF (176×144)	QCIF (176×144)
재생률	원 영상	30 fps
	압축영상	10 fps
움직임 정도	낮 음	높 음

AMV 부호기는 동영상의 첫 프레임을 I 프레임으로 압축하고, 나머지 프레임을 P 프레임으로 생성한다. 또한 초당 10 프레임의 재생률로 동영상을 생성하며 실험을 위해 I 프레임은 전송도중 오류가 발생하지 않는다고 가정한다. 다양한 네트워크 상태에서 AMV의 성능을 측정하기 위해 네트워크 시뮬레이터 NS-2를 이용하여 패킷 손실율을 2%~29%까지 다양하게 변화시켜 인터넷 트래이스를 생성하였다. AMV 부호기의 성능 평가 단위로 PSNR (Peak-signal-to-noise ratio)을 사용하였다. 실험에 사용된 QCIF 파일은 4:2:0 YUV 모드로 한 개의 매크로블록이 4개의 8×8 휘도 블록과 2개의 8×8 색상 블록으로 구성된다. 실험에서는 휘도 블록의 평균 PSNR을 성능평가 단위로 사용한다. PSNR은 다음과 같이 정의 된다[12].

$$PSNR [dB] = 10 \cdot \log_{10} \frac{255^2}{\frac{1}{N} \sum_{i=1}^N (x_i - \hat{x}_i)^2} \quad (5)$$

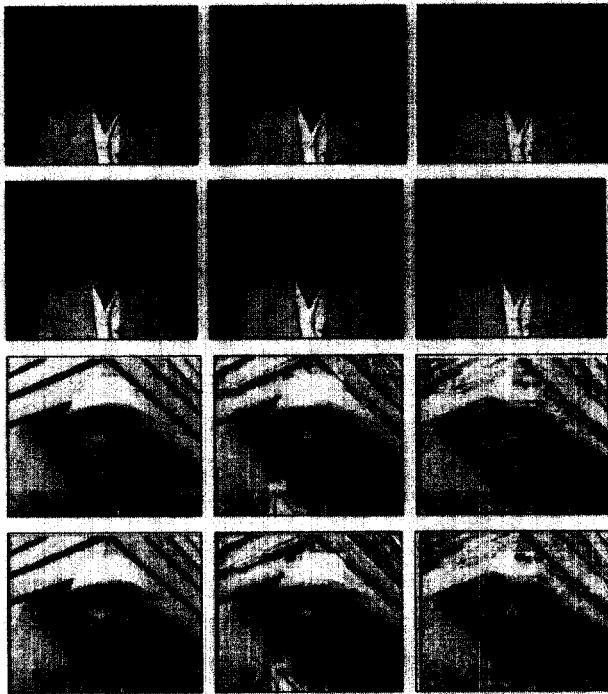
식 (5)에서 x_i 는 압축된 프레임의 픽셀 값이며, \hat{x}_i 는 복원된 값이다. 전체적으로 소스의 픽셀 값과 복원된 픽셀 값의 차이가 크면 PSNR은 감소한다. 이 PSNR은 복호화된 원래 이미지의 객관적인 화질 측정 단위이다.

실험은 움직임이 거의 없는 Akiyo와 변화가 심한 Foreman 동영상을 대상으로 실행되었으며 제안된 알고리즘의 R-D 성능을 H.263 표준과 각각 비교하였다. (그림 6)은 단위시간(초)에 생성되는 스트림의 양(비트율이라고 한다)을 15Kbps~97Kbps로 변화시키면서 0%, 5%, 15%의 패킷 손실이 발생했을때의 성능을 비교하였다. 전송된 패킷의 오류가 없는 경우 H.263이 AMV에 비해 평균 0.2dB 정도의 PSNR이 높게 나타난다. 이러한 이유는 AMV 기법에서는 부가정보를 전송하므로 동일 비트율에서 H.263의 성능이 더 높게 나타나게 된다. 손실율이 5%인 경우 AMV가 H.263에 비해 각각 0.9dB(Akiyo)와 1.4dB(Foreman)로 변화가 심한 동영상에서 더 높은 성능을 보인 반면, 15% 패킷 손실율에서는 Akiyo의 경우에 1.4dB, Foreman은 0.1dB 정도 높은 성능을 보였다. 이 실험을 통하여 AMV는 패킷 손실율이 낮을 때 움직임이 많은 동영상 복구에 효율적이며, 패킷 손실율이 높은 경우 움직임이 적은 동영상에 더 효율적임을 알 수 있었다.



(그림 6) 비트율과 패킷 손실율에 따른 성능 비교

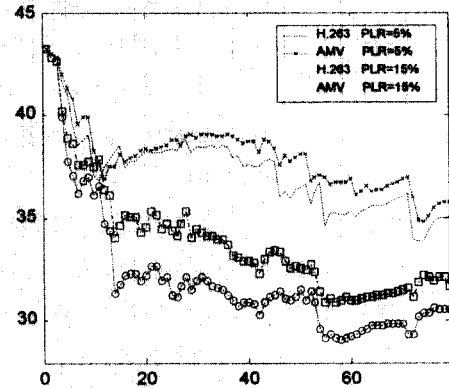
(그림 7)은 H.263 표준 압축기법과 AMV를 Akiyo와 Foreman에 적용하여 주관적 성능을 비교한 것이다. 각 열은 왼쪽에서부터 0%, 5%, 15%로 패킷 손실율을 적용하여 복구된 프레임들을 보이고 있으며 Akiyo의 경우 30번째 프레임들, Foreman의 경우 80번째 프레임들을 보여준다. 각 행은 H.263 표준과 AMV를 보여준다. (그림 6)에서 살펴본 바와 같이 Akiyo의 경우 15%의 패킷 손실율에서 얼굴선이 더 선명함을 알 수 있고, Foreman의 경우 5% 패킷 손실율에서 배경의 선이 분명함을 알 수 있다.



(그림 7) H.263과 AMV 기법을 Akiyo(위쪽)와 Foreman(아래쪽)에 적용하여 복구된 프레임의 화질 비교

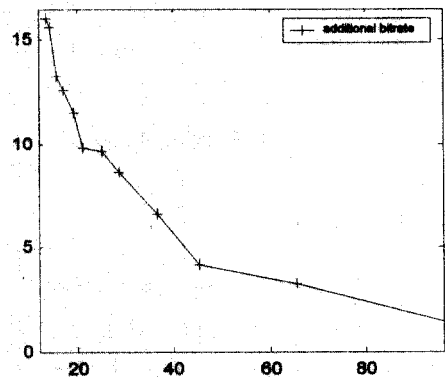
(그림 8)은 Akiyo 동영상을 압축하여 패킷 손실율이 5%, 15%인 경우 복구된 동영상 프레임 열의 PSNR 변화를 비교한 것이다. 실험에서 생성된 비트열의 평균 비트율은 67.44

Kbps이다. 전체적으로 AMV의 성능감소 폭이 H.263 표준에 비하여 적고 안정적임을 알 수 있다.



(그림 8) 동영상내 프레임의 성능 비교

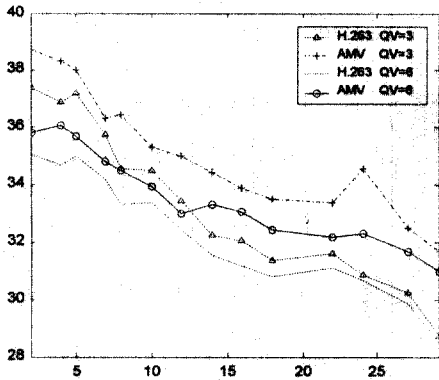
(그림 9)는 H.263 표준에 대한 AMV의 비트율 오버헤더를 백분율로 나타낸 것이다. X축은 H.263 표준 부호기가 초당 생성하는 데이터의 크기를 나타내고, Y축은 AMV의 오버헤더를 나타낸다. X축의 좌표 값은 양자화 계수를 변화시켜 생성된 비트율을 나타낸 값이다. 양자화 계수가 클수록 비트율이 증가하므로 프레임에서 일정한 크기를 차지하는 보조블록 정보의 비율이 낮아져서 오버헤더는 감소하는 경향을 보인다. 그림에서 AMV의 오버헤더는 최대 16%에서 최저 2%까지 낮아진다. 비트율이 낮은 경우 AMV의 오버헤더의 비율이 상대적으로 높아지므로 압축효과와 화질간의 적절한 조화를 고려해야 한다. (그림 9)에서 비트율이 43Kbps 인 지점에서 AMV의 오버헤더는 4%가 되는데, 화질과 압축효율이 적절히 조합된 지점이다.



(그림 9) H.263 표준에 대한 AMV의 오버헤드

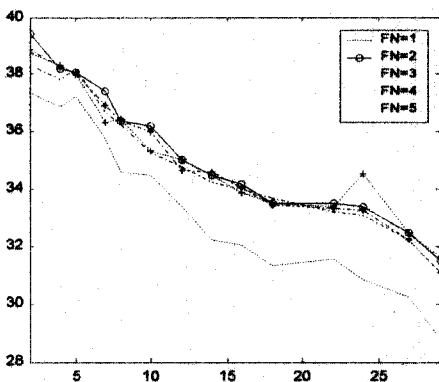
(그림 10)은 패킷 손실율이 2%~29%까지 변환때, 양자화 계수(QV)를 3과 6으로 하여 PSNR을 비교한 것이다. 화면의 변화가 적은 Akiyo에 대해 양자화 계수에 관계없이 AMV가 평균적으로 PSNR이 높으며 패킷 손실율이 12%이상인 경우

에는 오히려 양자화 계수가 6인 AMV 부호기가 더 효율적이다. 이것은 H.263 기본 압축방식으로 생성된 데이터의 절반의 데이터로 더 높은 화질을 보인다는 것을 의미한다. 화면의 변화가 심한 Foreman의 경우 패킷 손실율이 증가할수록 급격한 PSNR 감소를 보이며 AMV가 H.263 표준에 비해 약간 우수한 성능을 보인다.



(그림 10) 패킷 손실율의 변화에 따른 성능 비교

마지막으로 후보 프레임의 수를 조절하여 PSNR의 변화를 살펴보았다. 일반 표준 코덱의 압축 과정에서 움직임 예측이 차지하는 계산량이 전체 계산량의 약 30%~40%를 차지한다. AMV는 지정된 수의 후보 프레임을 이용해 각 후보 프레임과 움직임 예측을 수행하므로 후보 프레임 수가 많으면 계산량이 증가하고 압축을 수행하는 시스템의 CPU, 메모리 용량에 따라 실시간 압축이 어려워 질 수 있다. 그러므로 실시간 압축을 위해 적당한 후보프레임 수를 지정하는 것은 매우 중요하다. (그림 11)에서는 후보프레임의 수와 비디오 화질과의 관계를 파악하기 위해 후보 프레임 수를 조절하여 실험하였다. 실험 결과 후보 프레임의 수가 한 개인 경우를 제외하고 나머지의 경우 성능 변화가 거의 없었다. 그러므로 프레임 수를 두 개로 지정함으로써 실시간 압축과 성능이라는 요구사항을 모두 만족 시킬 수 있다.



(그림 11) 후보 프레임 수와 패킷 손실율에 따른 성능 비교

5. 결 론

본 논문에서는 인터넷을 통하여 동영상을 전송할 때 발생하는 전송오류의 영향을 최소화하기 위한 기법인 AMV(Auxiliary Motion Vector) 기법을 제안하였다. AMV에서는 참조되는 블록들의 연속적인 오류발생에 따른 오류전파를 줄이기 위해 여러 프레임중에서 압축대상 매크로블록과 가장 유사한 두 개의 블록을 선택한다. 또한 기존의 기법과 달리 움직임 보상에 사용되는 블록을 한 개로 제한하는 대신, 참조블록에 오류가 발생하면 보조블록을 이용하여 복구한다. 이 기법의 장점은 오류전파의 발생횟수와 한 번의 오류 전파가 미치는 영향을 줄일 수 있다는 것이다. 실험 결과 AMV는 화면의 변화가 적은 동영상에 대해서는 비트율이 높고 패킷 손실율이 높을수록 효율적이며 변화가 심한 동영상에 대해서는 비트율이 낮고 패킷 손실율이 낮을때 효율적임을 알 수 있었다. 특히, Akiyo 동영상에 대해 네트워크의 패킷 손실율이 12% 이상이고 양자화 계수가 6인 경우, H.263 기본 압축방식으로 생성된 데이터의 1/2 이하의 비트율로 더 높은 PSNR을 보여주었다. AMV의 단점은 여러 후보 프레임에서 움직임 예측을 수행하므로 일반 표준 코덱에 비하여 계산량이 증가하고 데이터의 크기가 증가한다는 점이다. 실험 결과 직전 2개의 프레임을 후보 프레임으로 선정한 경우가 실시간 압축과 화질 면에서 적합함을 알 수 있었고, 양자화 계수에 따라 오버헤드는 변하므로 적당한 양자화 계수를 선택 가능하다. 앞으로의 연구 과제는 AMV를 인터넷 뿐만 아니라 모바일 네트워크에서 적용 가능성을 실험하고, 발전시키는 것이다.

참 고 문 헌

- [1] Y. Takashima, M. Wada, and H. Murakami, "Reversible variable length codes," *IEEE Trans. Communications*, Vol. 43, pp.158-162, Feb./Mar./Apr., 1995.
- [2] D. W. Redmill and N. G. Kingsbury, "The EREC : An error resilient technique for coding variable-length blocks of data," *IEEE Trans. Image Processing*, Vol. 5, pp.565-574, Apr., 1996.
- [3] S. Wenger, G. Knorr, J. Ott, F. Kossentini : "Error resilience support in H.263+," *IEEE Trans. on circuit and System for Video Technology*, Vol.8, No.6 pp.867-877, Nov., 1998.
- [4] S. Aign, "Error concealment for MPEG-2 video," *Signal Recovery Techniques for Image and Video Compression and Transmission*, A. K. Katsaggelos and N. P. Galatsanos, editors, Kluwer Academic Publishers, ch.8, pp.235-268, 1998.

- [5] H. Sun and W. Kwok, "Concealment of damaged block transform coded images using projections onto convex sets," IEEE Trans. Image Proc., Vol.4, No.4, pp.470-477, Apr., 1995.
- [6] T. Nakai, and Y. Tomita : "Core Experiments on Feedback channel Operation for H.263+," ITU-T SG15 contribution LBC 96-308, Nov., 1996.
- [7] Wada M. "Selective Recovery of Video Packet Loss Using Error Concealment," IEEE J. Select. Areas in Commun. 7(5) : pp.807-814, June, 1989.
- [8] B. Girod and N. Farber, "Feedback-based error control for mobile video transmission," Proceedings of the IEEE, pp. 1707-1723, October, 1999.
- [9] Budagavi, M., Gibson, J.D, "Multiframe Video Coding for Improved Performance over Wireless channels," IEEE Trans. Image Processing. Vol.10. pp.252-265, Feb., 2001.
- [10] Chang-Su Kim, Rin-Chul Kim, Sang-Uk Lee, "Robust Transmission of Video Sequence Using Double-Vector Motion Compensation," IEEE Trans. On circuit and Systems for video technology. Vol.11, No.9, Sept., 2001.
- [11] Yang Yu, Xuelong Zhu : "Dynamic Multi-reference Prediction in Video Coding for Improved Error Resilience over Internet," IEEE Pacific Rim Conference on Multimedia 2001, pp.102-109, Oct., 2001.
- [12] E. Müller, T. Strutz, "Scalable Wavelet-based Coding of Color images," Proc. of APEIE '98. Vol.10, pp.29-35. Sept., 1998.



이 주 경

e-mail : jklee@melon.cs.pusan.ac.kr
 1996년 부산대학교 전자계산학과(학사)
 1998년 부산대학교 대학원 전자계산학과
 (이학석사)
 1998년~2001년 한국전력공사 근무
 2001년~현재 부산대학교 전자계산학과
 박사과정

관심분야 : 멀티미디어, 동영상 코덱, 오류제어



최 태 욱

e-mail : tuchoi@melon.cs.pusan.ac.kr
 1997년 동의대학교 전산통계학과(학사)
 1999년 부산대학교 대학원 전자계산학과
 (이학석사)
 2002년 부산대학교 대학원 전자계산학과
 박사

관심분야 : VoIP, 멀티미디어, QoS



정 기 동

e-mail : kdchung@melon.cs.pusan.ac.kr
 1973년 서울대학교(학사)
 1975년 서울대학교 대학원(이학석사)
 1986년 서울대학교 대학원(이학박사)
 1990년~1991년 MIT대학 교환 교수
 1995년~1997년 부산대학교 전자계산소
 소장

1999년~2001년 부산대학교 BK21 사업 단장
 1978년~현재 부산대학교 전자계산학과 교수
 관심분야 : 병렬처리, 멀티미디어