# 인터넷상의 비디오 데이타 전송에 효과적인 오류 은닉 기법

# (An Effective Error-Concealment Approach for Video Data Transmission over Internet)

김 진 옥 †

(JinOk Kim)

**요 약** 압축한 비디오 데이타를 전송할 때 인터넷과 같이 네트워크 채널이 불안정한 경우 패킷이 분실될 가능성이 높다. 패킷 분실은 연속적 비트 열에 오류가 발생하는 버스트 오류 형태로 일어난다. 본 논문에서는 버스트 오류를 은닉 처리하는데 효과적인 오류 내성 기법을 적용하는 동시에 데이타 숨김을 이용하여 디코더의 계산 복잡도를 줄인 빠른 오류 은닉 방법을 제안한다. 오류 은닉 효과를 높이기 위해, 인코더에서는 네트워크 채널의 버스트 오류에 강건하도록 비디오 데이타에 공간적, 시간적 영역에 대한 오류 내성 기법을 적용한다. 공간적 오류 내성 기법으로는 패킷 분실이 발생한 오류 블록을 분리하는데 효과적인 블록 셔플링을 적용하고 시간적 오류 내성 기법으로는 움직임 벡터의 프레임간 패리티 비트를 데이타 숨김 방법으로 내용 데이타에 삽입, 전송하여 디코더에서 분실된 패킷을 처리한다.

비디오 데이타는 전송 후 디코더에서 오류 은닉 처리하는데 디코더에서 주변 정보를 이용하여 오류 비디오 블록을 보간하는 것은 계산이 복잡하여 비용이 많이 든다. 따라서 본 연구에서는 비디오 인코딩 단계에서 비디오 블록의 에지 특징을 추출 후 이 특징 데이타를 원 데이타에 숨겨 전송하고 전송 시 비디오 데이타가 손상되면 디코더에서 숨겨 온 비디오 블록의 특징을 추출하여 쌍선형 보간법을 통해 전송 시 발생한 오류를 은닉 처리한다. 데이타 숨김을 이용하면 디코더의 계산 복잡도는 낮아진다. 본 논문의 실험 결과는 제안 방법이 비디오의 패킷 분실이 30%에 달하는 경우에도 이를 은닉 처리하여 인지 가능한 품질의 비디오 데이타를 보장한다.

**키워드** : 오류 은닉, 블록 셔플링, 데이타 숨김, 오류 내성

***Abstract*** *In network delivery of compressed video, packets may be lost if the channel is unreliable* like Internet. Such losses tend to occur in burst like continuous bit-stream error. In this paper, we propose an effective error-concealment approach to which an error resilient video encoding approach is applied against burst errors and which reduces a complexity of error concealment at the decoder using data hiding. To improve the performance of error concealment, a temporal and spatial error resilient video encoding approach at encoder is developed to be robust against burst errors. For spatial area of error concealment, block shuffling scheme is introduced to isolate erroneous blocks caused by packet losses. For temporal area of error concealment, we embed parity bits in content data for motion vectors between intra frames or continuous inter frames and recovery loss packet with it at decoder after transmission.

While error concealment is performed on error blocks of video data at decoder, it is computationally costly to interpolate error video block using neighboring information. So, in this paper, a set of feature are extracted at the encoder and embedded imperceptibly into the original media. If some part of the media data is damaged during transmission, the embedded features can be extracted and used for recovery of lost data with bi-direction interpolation. The use of data hiding leads to reduced complexity at the decoder. Experimental results suggest that our approach can achieve a reasonable quality for packet loss up to 30% over a wide range of video materials.

**Key words** : Error Concealment, Block Shuffling, Data Hiding, Error Resilience

† 학생회원 : 성균관대학교 정보통신공학부
jinny@ece.skku.ac.kr

논문접수 : 2002년 9월 24일
심사완료 : 2002년 11월 11일

## 1. 서 론

인터넷의 발달로 디지털 멀티미디어 데이타를 네트워크에 실어 보내는 것은 보편화되었다. 압축된 미디어 데이타는 사소한 비트 오류라도 디코더에서 영상 왜곡을 일으킬 수 있기 때문에 불완전한 네트워크 채널에 민감한 특성이 있다. 인터넷에 비디오 데이타를 전송할 때 데이타가 밀집되는 성질로 인해 패킷이 분실되고, 분실된 데이타로 인해 버스트 오류가 발생[1][2]하기 때문에 오류 은닉 기술과 오류 내성 기술의 적용이 필요하다. 오류 은닉은 손실된 비디오 프레임을 영상 특징인 시간적, 공간적 데이타 중복성을 이용하여 손실되지 않은 것처럼 처리하는 기술로서 주변 데이타간의 상호관계에 기반하여 손상된 영역을 복원한다. 디코더에서 오류 은닉을 수행하는 방법은 일반적으로 두 단계로 이루어진다. 먼저 디코더는 손실 정보의 몇 가지 특징을 예측한다. 이 특징들은 공간 보간에 필요한 에지 정보[3][4]이거나 움직임 모드 또는 시간 오류 은닉을 위한 움직임 벡터이다[5]. 그 다음 디코더는 공간, 변환 영역, 시간 보간을 이용하여 예측한 특징으로부터 손실 정보를 보간한다.

비디오 데이타에 대한 오류 은닉 방법은 시간적 영역의 오류 은닉 방법과 공간적 영역에서의 오류 은닉 방법 두 가지로 나뉜다. 시간적 영역에서의 오류 은닉 방법은 복원된 주변 블록의 정보를 이용해서 손실 블록의 움직임 벡터를 추정한 후 이전 프레임으로부터 손실된 블록을 보상하는 방법이다. 공간적 영역에서의 오류 은닉 방법은 움직임이 많은 영역에서 프레임내 부호화 블록에 효과적인 방법으로 주변 블록을 보간함으로써 오류 블록을 처리하며 높은 복잡도를 보이지만 시각적인 품질 개선 효과는 더 높다.

오류 내성은 인코더에서 디코더의 오류 은닉을 효과적으로 처리하기 위해 손실에 의한 오류의 여파를 최소화하는 기술로 MPEG-4에서는 재동기 마커, 데이타 분리, 가역 VLC와 같은 오류 내성 기술을 제안하여 랜덤 비트 오류를 처리한다. 하지만 패킷 분실은 오류 위치가 알려진 버스트 오류이기 때문에 랜덤 비트 오류 처리 기술을 적용하기에는 부적당하다[6][7]. 따라서 본 연구에서는 불안정한 채널에서 비디오 스트림 전송에 대해 디코더에서 오류 은닉 효과를 개선하는 오류 내성 방법을 제안한다. 공간적 오류 은닉을 처리하기 위한 오류 내성 방법으로 인코딩시 새로운 블록 셔플링 구조를 적용해서 오류 블록만을 분리시켜 이 분리된 블록에서 패킷 분실로 인한 오류가 발생하도록 만든다. 시간적 오류 은닉을 처리하기 위해서는 움직임이 많은 비디오에서 움직임 벡터의 복원은 쉽지 않다는 점을 고려하여 데이타 은닉법을 이용한 연속적인 내부 프레임 또는 프레임간에 움직임 벡터의 프레임간 패리티 비트를 삽입하는 오류 내성 기법을 제안한다. 또한 본 연구에서는 오류 은닉을 위해 주변 블록으로부터 내용 블록을 예측하는 대신 내용 블록 그 자체에서 에지 방향을 추출하여 추출한 정보를 쌍선형 보간[3][4]에 이용하는 방법을 제안한다.

디코더에서 쌍선형 보간을 수행할 때 오류 복원에 필수적이지만 계산 비용이 많이 드는 주변 정보를 이용한 손실 블록의 예측 대신 주변 정보 추출의 부담을 인코더로 옮겨 인코더에서 블록의 특징 정보를 추출하여 네트워크를 통해 전송하면 디코더는 인코더가 추출한 특징을 이용하여 빠르게 손실 정보의 보간을 수행하는 방법을 제안한다. 인코더에서 특징 추출을 수행하는 것은 인코더가 더 많은 데이타 정보에 접근하기 때문에 더 효과적이다. 인코더의 복잡도는 증가하지만 인코더는 더 많은 계산자원을 가지고 있으며 오프라인에서도 수행할 수 있기 때문이다. 인코더에서 추출한 특징은 미디어 데이타와는 별도 정보로 디코더로 보내는데 이때 데이타 은닉법을 이용한다. 이 방법은 비트율을 높이거나 쓸모없는 데이타를 생성시키지 않고 데이타를 임베드하여 보낼 수 있다.

제안 방법의 실험은 각 8 X 8의 DCT 블록과 16 X 16의 매크로 블럭으로 이루어진 MPEG으로 코딩한 비디오 데이타를 이용한다.

## 2. 관련 연구

### 2.1 오류 제어

네트워크상에 비디오 데이타 전송 시 발생하는 오류 제어 방법으로 디코더에서의 오류를 개선하기 위해 별도 정보를 부가로 보내는 FEC(Forward Error Correction)과 디코더에서 재동기 기능을 개선하기 위해 데이타를 재조직하며 분실된 패킷을 인코더에 재요청하는 ARQ (Automatic Repeat Request)와 같은 채널 코딩 방법을 가장 많이 이용한다. 하지만 FEC는 오류 복구에 필요한 잉여 데이타로 인해 제어하기 어려운 비트 스트림을 만들어 내기 때문에 인코더의 복잡도는 증가하고 대용량 네트워크 대역이 필요하며 지연시간이 증가한다. 또한 ARQ는 지연시간이 엄격한 실시간 응용분야에 적용하기 어렵다. 그렇기 때문에 이 방법들은 인터넷과 같이 버스트 오류가 집중적으로 발생하는 통신 채널의 연속적인 프레임 영역에서의 패킷 분실에 대해서는 그다지

효과적이지 않다. 이에 반해 오류 내성을 이용한 오류 은닉 방법은 이미지와 비디오의 공간적, 시간적 연관 관계를 이용해 오버헤드나 지연을 일으키지 않고 오류를 처리하는 방법으로 인정받고 있다. 특히 손실된 DCT 계수를 예측하기 위해 공간적 시간적인 은닉 방법의 제안[7]은 패킷 손실을 최소화하기 위해 소스 코딩 시 불록 셔플링을 하여 손실 블럭을 격리시킴으로써 오류 복구를 쉽게 한다. 오류 은닉 방법으로 비디오 데이타의 복잡도와 품질간의 여러 가지 상충관계를 다룬 다양한 연구도 제안되고 있다. 선형 보간법을 기반으로 한 알고리즘은 계산적으로는 단순하지만 블록 단위로 연산을 처리하여 인접 블록과 에지 성분이 비연속적으로 나타날 때 발생하는 블록 부산물을 다량 만든다[8]. Zeng [3]과 Sun[4]이 제안한 알고리즘은 손상된 블록을 은닉하기 위해 주변 픽셀에서 연결된 에지 정보를 이용한다. 고르지 않는 블록킹 부산물 효과를 경감시키지만 계산이 복잡하다. Lee[9]는 블록기반의 이미지 코딩 기술로 퍼지논리 기반의 오류 은닉 방법을 제안했다. 대부분의 오류 은닉 방법들은 오류 블럭을 모두 은닉할 때까지 계산이 반복적으로 이루어져야 하고 이 때문에 계산 비용이 높다는 단점이 있다.

이외 재동기화 마커, 데이타 분리, 가역 VLC 코딩과 같은 오류 내성 방법이 랜덤 비트 오류를 처리하기 위해 MPEG-4에서 제안되었으나 이 방법들은 오류 위치가 알려진 버스트 오류인 패킷 분실에는 효과적이지 않다[10][11].

따라서 본 연구에서는 인터넷과 같이 불안정한 패킷 채널에서 버스트 오류를 대해 시간적, 공간적 오류 은닉 효과를 높이는 데이타 셔플링과 움직임 벡터를 보호하는 패리티 비트를 이용한 오류 내성 방법을 적용하고 디코더에서의 계산 부담을 줄이기 위해 인코더에서 블록의 에지 특징을 추출하여 디코더로 전송하면 디코더에서는 쌍선형 보간법을 이용해 낮은 계산 비용으로 오류 은닉을 빠르게 수행하는 방법을 제안한다.

### 2.2 데이타 숨김

데이타 숨김은 사용권한 확인과 데이타 변조를 감지하는 목적으로 사용되고 있다. 그러나 일반적으로 데이타 숨김은 비밀전송, 워터마킹 등과 같이 원 데이타에 다른 목적으로 사용하는 정보를 숨겨 전달하는 방법이다. 데이타 숨김은 목표로 하는 비디오의 품질을 사람이 인지할 정도로만 낮추면서 감지할 수 없는 방법으로 별도의 정보를 전달한다. 다른 목적의 정보를 전달하는 방법은 그 정보를 패킷의 이용자 데이타 필드나 헤더 필드와 같은 곳에 별도로 부가해서 전송한다. 이 방법은 소스 코딩이나 채널 코딩에서 잉여 데이타를 사용하는 것과 유사하다. 잉여 데이타를 사용하는 것과 데이타 숨김의 오류 수정 결과는 동일하지만 데이타 숨김의 이점은 MPEG의 표준형태를 유지하면서 계산상 효과적인 방법으로 정보를 전달한다. 또한 데이타 숨김은 전체 비트율을 유지하면서도 사용자 필드에 별도 정보를 덧붙여 비디오를 더 낮은 비율로 변형 코딩한다. 변형 코딩은 비트율과 시각적 품질간의 상충관계에 복잡한 비율 제어가 관련되어 있기 때문에 이 분야에 관한 많은 알고리즘이 제안되었다[14]. 데이타 숨김은 디지털 데이타에 감지되지 않는 변형을 가하는 것으로 그 적용 방법에 따라 두 가지로 분류한다. 공간적 영역 방법은 몇 개 픽셀을 변형하여 송신자와 정해진 수신자만 그 변형된 위치를 감지하게 하는 방법이다. 주파수 영역 방법은 상관계수의 일부 또는 전체를 바꾸는 방법으로 이미지를 변형하여 에너지가 전체적으로 확산하게 하는 방법이다.

본 연구에서는 블록의 에지 정보와 움직임 벡터를 유지하기 위한 방법으로 적용한 패리티 비트를 디코더로 전송하기 위해 주파수 영역 방법의 홀짝 삽입방법의 데이타 숨김을 적용한다[15][16]. 고주파수의 DCT 상관계수에는 블록의 에지 정보를 숨기고 저주파 계수에는 패리티 비트를 숨겨서 디코더로 전송한다.

## 3. 버스트 오류에 효과적인 오류 보정 방법

### 3.1 제안 구조

그림 1은 원천 MPEG 비디오 스트림에서 오류 은닉 처리한 MPEG 스트림을 생성하는데 제안 오류 보정 구

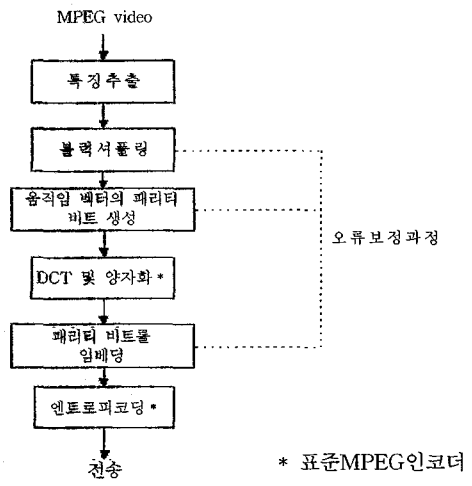



그림 1 인코더의 오류 보정 처리 흐름도

조를 적용한 과정을 보여준다. 인코더에서 MPEG 스트림은 블록으로 분리된 후 블록의 특징이 추출된 후 셔플링 된다. DCT와 양자화가 이루어진 다음 주파수 방식의 삽입이 행해진다. 만약 프레임이 I 픽처(미리 세팅된 픽처그룹의 첫번째 프레임)라면 프레임 방식의 패리티 비트들이 I 프레임에 삽입된다. 그 다음 엔트로피 코딩되는데 재귀 동기화 방식의 코드 워드가 블록 각 스캔 라인의 처음에 붙여져 한 블록의 전송 오류는 한 개 라인만을 손상시킨다. 엔트로피 코딩 후 새롭게 생성된 MPEG 스트림이 패킷으로 만들어져 전송된다. 디코더에서는 패킷 번호를 이용해 손상된 블록을 찾는 오류 감지가 먼저 수행되고 블록의 특징 정보를 연관 블록에서 추출한 후 역 셔플링을 한 다음에 특징 정보를 이용하여 쌍선형 보간으로 손실 블록을 복구한다.

### 3.2 공간적 오류 은닉을 위한 블록 셔플링 방법

3.2.1 소스 코딩과 채널 코딩

MPEG 표준을 기반으로 한 연구에서 압축한 비디오 비트 스트림에 재동기 워드와 같은 오류 내성 기술을 삽입할 경우 소스 코딩이 채널 코딩이 데이타 오류를 처리하는데 더 효과적이다[15].

소스 코딩의 블록 셔플링은 오류 비트들이 어떤 열로 표현되는 버스트 오류를 오류 비트 위치간에 아무런 상관성이 없는 랜덤 비트 오류로 만드는 것을 목적으로 하는 채널 코딩의 비트 인터리빙과 유사하지만 그 차이는 디코더에 있다. 소스 디코더는 랜덤 블록 오류의 복원을 위해 남아있는 잉여 데이타를 사용하는 반면 채널 디코더는 FEC 코드를 이용하여 랜덤 비트 오류를 복원한다. MPEG 비디오의 구조는 보통 비트 오류를 블록 오류로 만드는데 코딩된 비디오에는 언제나 약간의 잉여 데이타가 있기 때문이다. 인터리빙 방식의 채널 디코더는 버스트 오류보다는 랜덤 오류 비트를 처리하는데 효과적이고 버스트 오류를 처리하는데는 소스 블록 셔플링이 채널 인터리빙보다 더 효과적이다. 모든 표준 비디오 코딩 구조는 모두 블록 기반이기 때문에 같은 평균 비트 오류일 경우 소스 디코더가 랜덤 오류보다는 버스트 오류를 더 잘 처리한다.

3.2.2 오류 분산 블록 셔플링 방법

인터넷에서 전송된 패킷은 분실될 경우 그 분실 위치가 노출되는 버스트 오류이다.

버스트 오류는 코딩 블록을 심하게 손상시켜 오류 은닉이 불가능하게 만들기도 한다. 따라서 버스트 오류를 처리하는 동시에 프레임 내에서 손상된 블록들을 가능한 멀리 분리하기 위해 규칙적이거나 임의적으로 이루어 지는 치환 과정인 블록 셔플링 방법을 적용한다. 오류가 분리되지 않으면 영상에서 불필요한 부산물이 두드러지지 않기 때문에 디코더에서 오류 은닉을 처리하기 어렵다. 블록 셔플링의 다른 잇점은 오류를 분산시킨다는 점이다. 집중적으로 발생한 오류는 움직임 보상을 통해 다음 프레임에 더 잘 파급되기 때문에 블록을 셔플링함으로써 오류 파급 효과를 낮춰 오류 발생을 줄인다. 랜덤 블록 셔플링은 오류를 격리하는데 오류가 발생한 블록이 4% 이하여야 효과적이기 때문[16]에 인터넷 응용 분야에 적용하기 어렵다. 랜덤 블록 셔플링이 제안한 방법인 짝-홀수 블록 셔플링 또는 양방향 블록 인터리빙(2BI) 방법[17]은 그림 2(a)와 같다. 모든 블록은 '1' 또는 '2'로 레이블 된다. 각 줄의 모든 '1' 블록을 차례로 인코딩하고 다음 줄의 '1' 블록들을 인코딩한다. 모든 '1' 블록들을 인코딩한 후 '2' 블록들을 같은 방법으로 인코딩하면 각 '1' 블록은 4개의 '2' 블록들로 싸이게 되고 각 '2' 블록들은 4개의 '1' 블록들로 싸인다. 이렇게 되면 이 방법을 적용한 블록의 분실율이 50%에 달하게 되어 이미지 전체에서 복원할 수 없는 오류를 유발하게 된다.

본 연구에서 제안하는 셔플링 패턴은 오류를 이미지의 전체 영역으로 분산시켜 중요한 연결 블록들이 동시에 분실될 수 있는 가능성을 낮춘다. 이 셔플링 패턴을 '오류 분산 셔플링'이라 부른다. 오류 분산 셔플링은 오류를 효과적으로 분산하고 압축할 때는 계산 비용을 줄인다. 오류 분산 셔플링은 송신된 패킷에서 블록의 모든 열과 행은 분실될 수 있는 가능성이 동일하다는 점을 고려했다.

| | |
|---|---|
| 12121212<br>21212121<br>12121212<br>21212121 | 18352746<br>61835274<br>46183527<br>74618352 |

(a) 2BI (b) 오류 분산 셔플링

그림 2 셔플링 패턴

그림 2(b)는 제안 셔플링 패턴이다. $N \times M$ 크기의 이미지가 $8 \times 8$ 크기의 블록으로 DCT 변환되었다고 할 때 이 패턴은 다음과 같은 방법으로 생성한다. $N$ 개의 열과 $M$ 개의 행으로 구성된 매크로 블럭의 이미지에서 $M > N$이라고 가정한다. 좌측 상위 모서리부터 시작하여 $(i,i)$블록을 "1"로 번호를 붙인다. $N$ 블록을 "1"로 한 다음 $(1,N+1)$블록부터 시작하여 모든 $(i,i+N)$블록을 "2"로 번호를 붙인다. 이미지의 우측 끝까지 번호를 붙였다면 레이블 $k+1$을 $(1,j+1)$블록부터 할당한다. 만약 $(1,j+1)$에 이미 번호가 지정되었

다면 $k+1$을 $(1,j')$블록에 할당한다. $j'$은 $j+1$과 가장 큰 값으로 지정된 인접한 행 인덱스의 중간 값이다. 만약 이 역시 지정되어 있다면 $k+1$을 $(1,j'')$에 할당한다. $j''$은 $j+2$와 가장 큰 값으로 번호가 붙여진 인접한 행 인덱스의 중간 값이다. 모든 블록에 번호가 붙여질 때까지 이 방법을 취한다. 이 셔플링 방법은 버스트 오류가 발생한 손실 패킷 상의 오류 블록을 격리시킨다.

2BI와 같은 고정된 셔플링 패턴은 긴 비디오 스트림에서 오류를 발생시키는 쓸모없는 부산물 데이타를 만들 수 있다. 그래서 제안 방법과 같이 방향이 다른 셔플링 패턴을 혼합하여 적용하면 부산물 데이타를 줄일 수 있다.

### 3.3 시간적 오류 은닉을 위한 움직임 벡터 보호 방법

움직임 벡터는 시간적 영역에서의 오류 은닉을 위해 필수적이다 그래서 많은 오류 은닉 방법이 손상된 움직임 벡터를 복원하는데 제안되었다. 그러나 손상된 영역이 그 주변과 연관 관계가 없을 경우 움직임 벡터를 복원하는 것은 매우 어려운 일이다. 디코더의 오류를 개선하기 위해 별도 정보를 부가로 보내는 FEC는 움직임을 정확히 복원하기 때문에 디코더 오류 복구에 가장 많이 이용한다. 그러나 인터넷과 같은 네트워크를 통한 데이타 전송 시 발생하는 버스트 오류를 처리하기 위해서는 많은 잉여 교정 코드가 필요한데 이 때문에 패킷은 더 많이 분실될 수 있다. 본 연구에서는 움직임 벡터를 복원하는데 명확한 오류 수정 비트를 전송하는데 데이타 숨김 방법을 적용한다. 이 방법은 높은 데이타량에도 불구하고 감지 불가능한 데이타 숨김 구조로 비디오 압축에서도 유지 가능하다.

제안 방법에서 오류 수정은 패리티 비트를 통해 이루어진다. Song[18]은 매크로 블럭 그룹 전체에 패리티 비트를 삽입하는 방법을 제안했다. 이 방법은 손실된 움직임 벡터가 프레임에서 오직 블록 그룹 한 개와 관련이 있기 때문에 프레임에서 블록 그룹이 연속적으로 손실될 수 있는 인터넷 전송에는 적절하지 않다. 그래서 본 연구에서는 미리 정해진 프레임간 그룹의 움직임 벡터에 대해 프레임 지향 패리티 비트를 생성하는 방법과 코딩 구조 및 지연 요구에 따라 생성한 패리티 비트를 연속적인 프레임 내 또는 프레임간에 삽입하는 방법을 제안한다. 프레임에 패리티 비트를 삽입하는 데는 데이타 숨김을 이용함으로써 데이타 코딩 비율을 변하게 않게 유지한다. 이 구조는 버스트 오류 환경에서 손실된 움직임 벡터를 잘 복원한다. 프레임간에서는 P 프레임만 앵커 프레임으로 사용되고 이 프레임은 오류 확산에 민감하기 때문에 이 프레임들에만 움직임 벡터 보호 방법을 적용한다. MPEG에서 움직임 벡터는 호프만 코딩으로 표현된다. 패리티 비트를 계산하기 위해 먼저 각 P 프레임의 움직임 벡터의 코딩된 비트를 라인 단위로 만든다. 만약 매크로 블럭에 움직임 벡터가 없다면 매크로 블럭에 "0"을 할당한다. 픽처그룹의 $k$번째 P 프레임의 $j$번째 줄에 $i$번째 움직임 벡터를 $V_j^{(k)}(i)$와 같이 정의하면 GOP의 움직임 벡터의 패리티 비트는 다음과 같이 계산한다. 움직임 벡터 보호를 위한 패리티 비트의 생성은 식 (1)과 같다.

$$p_j(i) = V_j^{(1)}(i) \oplus V_j^{(2)}(i) \oplus \cdots \oplus V_j^{(N_p)}(i) \quad (1)$$

여기에서 $\oplus$는 각 자리 벡터 비트의 합을 mod(2) 연산함을 의미하고 $p_j(i)$는 P 프레임 전체인 $N_p$에 대한 $j$번째 줄의 $i$ 번째 패리티 비트를 의미한다. 각 자리 비트의 합을 mod(2) 연산하면 합이 짝수이면 '0' 패리티가, 홀수이면 '1'패리티 비트가 생성된다. 각 움직임 벡터들의 라인 길이가 동일하지 않기 때문에 긴 라인과 짧은 라인을 맞추기 위해 짧은 라인의 빈 부분에는 '0' 비트를 삽입한다. 이 구조는 각 비트 자리마다 발생하는 비트 오류 하나를 수정하기 위해 하나의 패리티 비트를 사용한다. 디코더에서는 이 패리티 비트를 움직임 벡터와 비교하여 오류가 발생하면 $N_p$비트들에서 비트 오류를 한 비트씩 감지할 수 있다.

생성된 패리티 비트를 다음 연속 I 프레임의 양자화된 DCT 계수에 삽입하여 디코더로 전송한다. 두드러져 보이는 부산물 데이타를 덜 만들기 위해 DC와 저주파의 AC 계수는 바꾸지 않고 데이타 삽입에 사용하는 상관계수를 선택하는 데는 시각 인지 모델을 이용한다. 프레임간에 패리티 비트를 적용하는 것은 최소의 패리티 비트를 이용하여 같은 프레임에서 연속적으로 발생하는 오류를 보호하는 장점이 있다. 균일하지 않은 블록의 삽입 용량을 균일하게 만들기 위해 셔플링은 삽입 전에 적용한다. 만약 비디오의 삽입 용량이 작으면 움직임 벡터를 선택적으로 삽입하거나 한 개 슬라이드에 움직임 벡터의 평균이나 합만을 삽입한다. 한 개 패리티 비트 집합을 삽입하기 위해 한 개 이상의 프레임을 이용할 수도 있다. 디코더에서는 패킷 번호로 오류 비트의 위치를 감지한다. 감지된 오류 블록은 쌍선형 보간을 통해 은닉 처리된다.

## 4. 쌍선형 보간을 이용한 빠른 오류 은닉

### 4.1 블록 특징 추출

본 논문에서는 디코더에서 인터넷을 통해 전송된 데이타의 오류 은닉을 효과적으로 실시간으로 빠르게 처

리하기 위해 주변 블록에서 손실 블록의 공간 정보를 예측하는 것을 인코더에서 처리하도록 한다. 이 방법은 인코더에서 8 x 8로 나눈 블록 A에서 블록의 에지 특징을 뽑아 주변 블록의 스트림내 사용자 데이타로 만들어 디코더에 전송한다. 블록의 일부 또는 전체가 손실되면 디코더에서는 주변 블록에 삽입된 블록 A의 특징과 블록 A의 주변 블록이 오류를 은닉하는 데 이용된다. 이 때 이미 인코더에서 블록 A의 특징이 추출되어 왔기 때문에 디코더의 손실 블럭 A의 오류 은닉 계산 부담은 경감한다.

본 논문에서 적용한 오류 은닉 방법중의 하나인 쌍선형 보간법은 주변 픽셀에서 손실되거나 손상된 데이타의 지역적 기하 정보를 뽑아내어 내용 종속적인 보간에 이용한다. 주변의 정확하게 수신한 블록으로부터 손상된 블록의 에지 방향과 같은 지역적 기하 정보를 예측한 다음 정확하게 수신된 주변 블록을 이용하여 에지를 따라 손상된 블록을 보간한다. 비디오에서 손실된 영역은 패킷 번호로 감지한다. 패킷 기반 데이타 채널에서 손실된 영역은 블록 형태이다.

내용 블록의 에지는 작은 블록에서 직선과 유사하다. 에지 방향을 결정하기 위해서 본 연구에서는 로버츠 기울기 연산자[4]를 적용한다. 픽셀의 지역 에지의 기울기 성분은 먼저 (2)과 (3)에 의해 계산한다.

$$g_y = u_{i+1,j-1} - u_{i-1,j-1} + 2u_{i+1,j} - 2u_{i-1,j} + u_{i+1,j+1} - u_{i-1,j+1} \quad (2)$$

$$g_x = u_{i-1,j+1} - u_{il,j-1} + 2u_{i,j+1} - 2u_{i,j-1} + u_{i+1,j+1} - u_{i+1,j-1} \quad (3)$$

$g_x$와 $g_y$는 수평, 수직 기울기이다. 좌표 $(i,j)$에서 기울기 벡터의 크기와 방향은 식 (4)과 (5)이다.

$$g(i,j) = \sqrt{g_x^2 + g_y^2} \quad (4)$$

$$\theta(i,j) = \tan^{-1}(\frac{g_y}{g_x}) \quad (5)$$

픽셀 위치 $(i,j)$는 만약 $g_{i,j}$가 임계값 $t$를 초과하면 에지 점이 된다. 만약 거기에 아무런 에지 점이 없으면 내용 블럭은 에지가 없는 블록이다. 그렇지 않으면 에지 블록이다. 에지 블록이면 이 에지 블럭에 대한 에지 방향은 $0^0$-$180^0$ 범위에서 동일한 간격으로 배치된 방향인 $m$ 중 하나로 식 (6)과 같이 양자화된다.

$$q^{\theta}_{ind}(i,j) = \left[ \frac{\theta(i,j) + \pi/(2m)}{\pi/m} \right] \quad (6)$$

여기서 $q^{\theta}_{ind}(i,j) \in \{0, \cdots m-1\}$

한정된 삽입 용량 때문에 내용 블럭이 한 개 이상의 에지를 가졌다 해도 한 개 에지만이 별도의 정보로 전송된다. 식 (7)의 다수결 투표 방법을 이용하여 한 개의 에지를 선택한다.

$$q^{\theta}_{ind} = \{k \mid \sum_{q^{\theta}_{ind}(p,q)=k} g(p,q) \geq \sum_{q^{\theta}_{ind}(p,q)=l} g(p,q), l=0,\cdots,m-1, l \neq k\} \quad (7)$$

$$q^{\theta} = q^{\theta}_{ind} \times \pi/m + \pi/2m \quad (8)$$

$(p,q)$는 내용 블록에서 에지 포인트의 좌표이고 $q^{\theta}$는 에지의 각도이며 $q^{\theta}_{ind}$는 동위 인덱스이다. 인코더에서 에지를 쉽게 감지하기 위해 내용 블록중 두 개의 최근접 주변 층을 합하여 블록 크기를 확장한다. 인코더에서 에지를 쉽게 감지하여 디코더로 보내면 오류 은닉시 쓸모없이 발생하는 부산물 데이타를 줄일 수 있다. 내용 블록의 에지 블럭 유무를 표시하기 위해서는 한 비트가 필요하다. 에지 방향 인덱스 $q^{\theta}_{ind}$는 $b=[\log_2^m]$비트로 표시한다. 그래서 한 비트를 에지가 없는 블럭에, $1+b$ 비트를 에지 블록에 할당하여 추출한 내용 블록의 특징을 각 프레임 연관 블록의 DCT 잔여량내에 별도 정보로 삽입한다[13][14]. 내용 블록의 특징을 추출한 다음 내용 블록은 선형 데이타로 연결되어 셔플링한다. 다음 DCT와 양자화 단계를 거친다.

### 4.2 데이타 삽입 및 추출

불록의 특징 비트를 내용 데이타에 삽입하는 것은 MPEG 표준 양자화 단계 후에 수행한다. 고주파의 DCT 계수를 변형하면 눈에 띄는 데이타 부산물을 덜 만들기 때문에 추출한 특징 정보를 상관 블록의 DCT의 고주파 성분에 삽입하는데 AC성분의 첫번째와 두번째 대각선 계수를 제외한 성분에 삽입한다. 본 연구에서는 데이타 숨김의 홀짝 삽입 방법[19]을 적용하여 특징 정보 비트가 '0'이면 상관 계수는 '0'을 더해 짝수로 만들고, 특징 정보 비트가 '1'이면 상관계수는 '1'을 더해 홀수로 만든다. '0'으로 양자화된 상관계수는 변경되지 않는다.

삽입된 데이타를 디코더에서 추출하는 것은 삽입 과정을 반대로 한다. 디코더에서 수신한 비트 스트림은 먼저 역 엔트로피 코딩된다. 만약 역 엔트로피된 스트림이 I 스트림이고 이전 P 프레임에 오류가 있으면 삽입된 데이타가 추출된다. 표준 DCT및 양자화 과정을 거친 데이타의 상관 계수에서 짝수 계수에는 '0'이 홀수 계수에는 '1'의 특징 비트가 삽입되었으므로 이 특징 정보를 추출한다. 역 양자화 과정과 역 DCT 과정이 표준 MPEG 디코더에 의해 실행된다.블록들은 그 다음 원래 순서대로 역 셔플링한다. 마지막으로 특징정보를 이용한 쌍선형 오류 은닉이 이루어진다. 그림 3은 오류 은닉 처리 플로우이다.

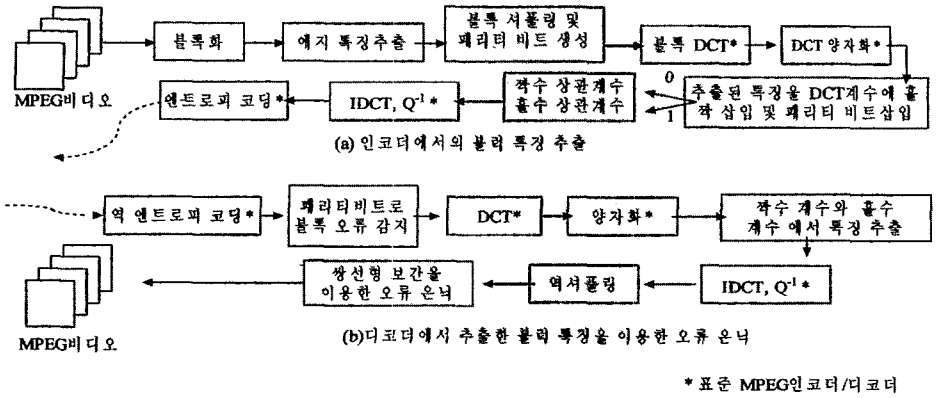



그림 3 오류 은닉 처리 플로우

### 4.3 쌍선형 오류 은닉

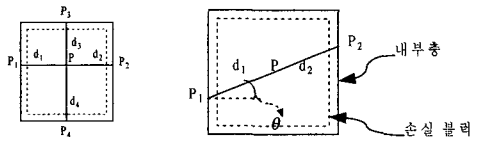
오류 은닉은 비디오 디코딩 후에 이루어진다. 수신한 패킷의 패킷 번호를 통해 어떤 패킷이 손실되었는지 파악한다. 손실된 패킷의 I 프레임 내 오류에 대해 쌍선형 보간법을 적용한다. 만약 디코더에서 손실 블록이 감지되면 이 블록 타입과 에지 특징은 인코더에서 추출한 호스트 블록의 특징과 타입을 삽입한 연관 블록으로부터 추출하고 그림 4와 같이 분실 블록을 재현하기 위해 쌍선형 보간을 수행한다. 만약 분실 블록이 에지가 없는 블록이면 그림 4(a)와 같이 픽셀 P의 쌍선형 보간을 수행한다.

$$p=\frac{\left[\frac{d_2}{d_1+d_2}P_1+\frac{d_1}{d_1+d_2}P_2+\frac{d_4}{d_3+d_4}P_3+\frac{d_3}{d_3+d_4}P_4\right]}{2} \quad (9)$$

식 (9)의 $P_1, P_2, P_3, P_4$는 주변 블록에 삽입된 블록의 특징 정보이다. 만약 손실 블럭이 에지 블록이면 에지 방향 θ에 따라 P의 쌍선형 보간은 그림 4(b)와 같이 이루어진다.

$$P=\frac{d_2}{d_1+d_2}P_1\frac{+d_1}{d_1+d_2}P_2 \quad (10)$$

식 (10)의 $P_1, P_2$는 주변 블록의 내부층에서 2개의 최근접 픽셀로부터 선형적으로 보간된다. $d_i=|P-P_i|$이다.




(a)에지가 없는 블록 (b) 에지 블록

그림 4 쌍선형 보간

P 프레임에 대해서는 GOP의 각 P 프레임에서 같은 위치에 있는 슬라이스에서 한 개 슬라이스만 손실되었다면 정확하게 수신된 다음 I 프레임 또는 P프라임의 같은 위치 슬라이스를 이용하여 움직임 벡터를 복원한다.

만약 오류가 발생한 블록의 특징을 삽입한 연관 블록이 손실되었다면 미디안 필터를 이용하여 인접한 움직임 벡터로부터 손실된 움직임 벡터를 보간하거나 H.263에서 제안된 것과 같이 중첩된 움직임 보상 방법을 이용하여 시간적 방법의 오류 은닉을 처리한다.

## 5. 실험 및 고찰

인터넷에서 패킷 손실을 시뮬레이션 하기 위한 방법으로 상태 마코프 모델을 이용한다. 채널은 평균 패킷 손실 확률($M_l$)과 평균 버스트 길이($L_b$)를 설정한다. 비디오는 변동 길이의 패킷으로 만든다. 각 패킷은 재동기 마커가 있는 한 개 매크로 블록 라인으로 구성한다. 제안 구조의 효과를 보여주기 위해 패턴을 인코딩하고 다양한 프레임 크기로 비디오 전역을 테스트하는 과정이 필요하다.

### 5.1 채널 모델

인터넷은 두 개의 상태로 구성된 마코프 체인으로 모델링한다[20]. 이 모델에서 채널은 두 개 상태중 하나가 되는데 좋은 상태는 "0"으로 나쁜 상태는 "1"로 모델링한다. 상태 "0"에서 패킷은 정확하게 수신되고 상태 "1"에서는 패킷이 손실된다. 모델은 천이 매트릭스 로 묘사된다.

만약 채널이 $n^{th}$시간 단위에서 좋은 상태라면 $s_n$="0"이고 반대의 경우 $s_n$="1"이다. $s_n$은 천이 매트릭스 형태의 이진 마코프 프로세스이다

$$M=\begin{pmatrix} M[s_n=0|s_n=0] & M[s_n=0|s_n=1] \\ M[s_n=1|s_n=0] & M[s_n=1|s_n=1] \end{pmatrix}=\begin{pmatrix} m_{00} & m_{01} \\ m_{10} & m_{11} \end{pmatrix} \quad (11)$$

마코프 체인은 정적임을 가정한다. 평균 버스트 손실 확률을 식 (12)로 표현하고

$$M_l=\frac{m_{01}}{m_{10}+m_{01}} \quad (12)$$

평균 버스트 길이는 식 (13)으로 채널 조건을 만든다.

$$L_b=\frac{1}{m_{10}} \quad (13)$$

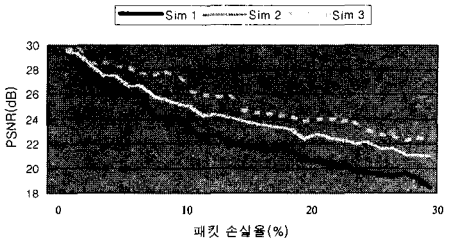
### 5.2 실험

실험은 많은 움직임이 있는 'football' 비디오를 이용했다. 연속 움직임은 초당 10 프레임의 QSIF(176 X 112) 크기의 MPEG-1으로 인코딩했다. GOP 프레임은 IPPP으로 설정했다. B 프레임은 오류 확산에 영향을 미치지 않으므로 2B 프레임은 생략하였다. 코딩 효율을 보여주는 데는 고정된 양자화기를 적용했다. I 프레임에 대한 양자화 크기 팩터는 10이고 프레임간 그룹의 수는 4로 설정했다. 움직임 벡터는 15픽셀 크기 이내 이고 정확도는 반 픽셀이다. 인트라 블록은 오버헤드 때문에 P 프레임으로 한정한다. 비트 전송율은 256Kbps이다. 실험에서는 세 가지 시뮬레이션을 수행하고 그 결과를 비교했다. 각 시뮬레이션은 셔플링 구조, 공간 오류 은닉 방법, 시간 오류 은닉 방법 등 세 부분으로 행했다. 표 1에 각 시뮬레이션 방법을 열거했다. 시뮬레이션 1의 블록 복사는 같은 공간위치를 점유한 이전 프레임의 블록으로 손상 프레임을 대치하는 방법이다. 시뮬레이션 3의 공간적 오류 은닉 방법에서는 인코더에서 내용 블록의 에지 정보를 추출해서 디코더로 보내 디코더에서는 이 정보를 이용하여 쌍선형 보간을 수행했다. 시간적 오류 은닉의 경우 움직임 벡터 패리티 비트를 삽입하는 방법을 적용하면서 패리티 비트가 손실될 경우에는 미디언 필터를 적용했다.

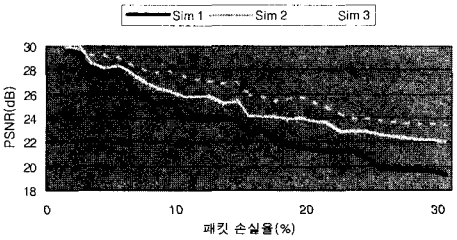
표 1 시뮬레이션 방법

| | | |
|---|---|---|
| | 블럭 복사 | 블럭 복사 |
| | 2B1 인터리빙 + 쌍 선형 보간 | 미디안 필터 |
| | 제안 셔플링 + 에지 정보 추출 + 쌍선형 보간 | 움직임 벡터 패리티 비트 삽입 + 미디안 필터 |

먼저 비트율은 시뮬레이션 1과 비교하여 시뮬레이션 2와 시뮬레이션 3이 0.35%, 1.02%로 증가되었다. 이것은 제안 알고리즘이 무시할 정도의 낮은 압축율을 보임




(a) 랜덤 손실




(b) Lb = 2인 버스트 오류

그림 5 시뮬레이션 방법에 따른 평균 PSNR

을 보여준다. 그림 5에서는 다섯 가지의 다른 채널 조건을 이용하여 시뮬레이션 했을 때 $P_l$함수로써 원본 이미지와 변화된 이미지의 색상 변화를 나타낸 수치인 평균 PSNR(Peak Signal to Noise Ratio)을 보여준다. 인터넷 송신에서 전형적인 버스트 오류 길이가 2일 때와 랜덤 오류 손실의 경우를 시뮬레이션 했다. 실험 결과 랜덤 손실의 경우 시뮬레이션 3이 시뮬레이션 1에 대해 3.81dB의 PSNR를, 시뮬레이션 2에 대해서는 1.98dB의 PSNR을 보였다. 버스트 오류에 대해서는 각각 3.37dB, 1.63dB의 PSNR을 보였다.

영상차이는 그림 6에서 확인할 수 있다.

(a) I 프레임의 블록 손실 패턴

(b) 오류 은닉 처리한 I 프레임

그림 6 "풋볼" 비디오의 오류와 복구

그림 6(a)는 I 프레임에서 3번째 프레임이 손실되었을 때 손실 블록의 패턴이다. 그림 6(b)는 오류가 발생한 I 프레임을 복구한 결과이다. 각 프레임에는 좌측 프레임에는 시뮬레이션 방법 1을 적용했고 중간 프레임에는 시뮬레이션 방법 2, 우측 프레임에는 시뮬레이션 방법 3을 적용했다.

그림 7은 재현된 I 프레임의 PSNR과 P프레임에서 아무런 오류가 발생하지 않았다는 가정 아래 I 프레임에 종속된 4개 P프레임의 PSNR이다. 시뮬레이션 3의 결과는 다른 두 시뮬레이션 방법보다 더 우수하다. I 프레임에서 훑어진 오류는 오류 은닉 품질을 개선할 뿐만 아니라 다른 종속 프레임에 대한 오류 파급 효과를 줄인다. 제안된 데이타 삽입에 의한 움직임 벡터 보호 방법이 I 프레임의 이미지 객체의 품질을 나쁘게 하거나 종속적 P 프레임에서 오류를 발생하게 할 수도 있으나 잉여로 인한 인지적인 잇점 덕택에 영상 품질 저하는 거의 감지되지 않음을 알 수 있다. 만약 연속적 I 프레임에 삽입할 여력이 있다면 더 강력한 오류 수정 코드를 적용할 수 있다.

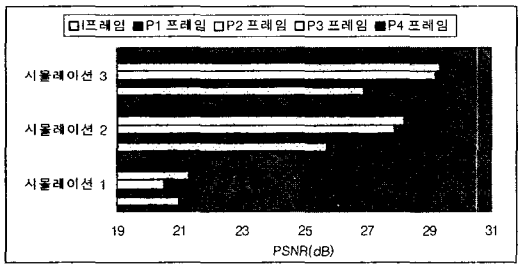



그림 7 오류 은닉한 I 프레임과 연속 P 프레임의 PSNR

그림 8은 오류 은닉한 시뮬레이션 이미지의 품질 결과이다. 여기서 무오류는 오류가 없는 표준 MPEG 비디오를 이용했을 때의 이미지 품질이다. 전체적으로 시뮬레이션 3의 결과 이미지의 품질이 높음을 알 수 있다.

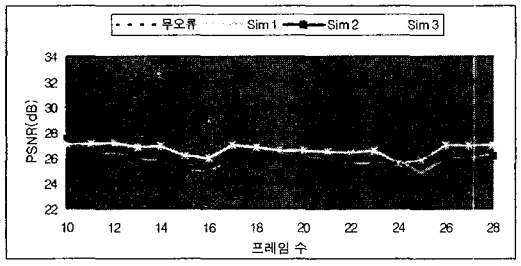



그림 8 오류 은닉 결과 비교

## 6. 결 론

본 연구에서는 인터넷을 통해 비디오 데이타 전송 시 발생하는 버스트 오류에 대해 오류 은닉 효과를 높이는 방법을 제안했다. 이를 위해 인코더에서 데이타를 셔플링하여 비디오에서 국지적으로 발생한 오류를 전체로 퍼지게 함으로써 인터넷을 통해 전송한 후에도 손실 블록이 프레임에 미치는 영향을 적게 하는 공간적 오류 내성 방법과 움직임 벡터에 패리티 비트를 삽입하여 데이타 손실로 인한 오류를 디코더에서 빨리 인식 처리할 수 있는 시간적 오류 내성 방법을 제안했다. 또한 디코더에서의 오류 은닉을 빠르고 효과적으로 처리하기 위해 디코더에서 손실 블록의 오류 은닉을 위해 주변 블록에서 필요 정보를 예측하는 대신 인코더에서 블록의 특징 정보를 추출하여 DCT 상관계수에 은닉, 삽입하여 전송하는 방법을 제안했다. 디코더에서의 오류 은닉은 오류가 발생한 블럭에 대해 전송되어 온 블록의 특징정보를 쌍선 보간법을 적용하여 처리하였으므로 계산 부담이 줄었다. 움직임 벡터의 패리티 정보와 인코더에서 추출한 데이타 블록의 특징은 데이타 숨김 방법을 통해 전송하였기 때문에 전체 데이타 양에는 영향을 미치지 않았다.

실험 결과는 제안 방법을 적용하면 패킷 손실율이 25~30%까지 높아져도 높은 비디오 품질로 복원됨을 보여준다. 제안 방법은 MPEG 표준 인코더와 디코더의 비디오의 구조와 내용을 이용하였으며 특히 패킷 손실을 조합하는데 유용하다. 향후 연구에서는 제안 방법을 다양한 네트워크 채널 환경에서 여러가지 MPEG, MPEG2 비디오 데이타에 적용하여 신뢰성을 높일 계획이며 인코더에서 다양한 에지 정보를 삽입하여 전송하는 데이타 숨김에 대한 연구도 계속할 것이다.

## 참 고 문 헌

[1] J. M. Boyce and R. D. Gaglianello, "Packet loss effects on MPEG video sent over the public Internet," ACM Multimedia'98, pp.181-190, 1998.

[2] U. Horn, K. Stuhlmuller, M. Link and B. Girod, "Robust Internet video transmission based on scalable coding and unequal error protection," Image Communication, Special Issue on Real-time Video over the Internet, pp. 77-94, 15(1-2), Sept., 1999.

[3] W. Zeng and B. Liu, "Geometric structure-based error concealment with novel application in block-based low bit rate coding," IEEE Transaction on CSVT, pp. 648-665, June 1999.

[4] H. Sun and W. Kwok, "Concealment of damaged block transform coded images using projections onto convex sets," IEEE Transactions on Image Processing, vol 4, pp. 470-477, April 1995.
[5] W. Lam, A. Reibman and B. Liu, "Recovery of lost or erroneously received motion vectors," ICASSP'93, pp.304-315, 1993.
[6] 김진옥, 황대준, "인터넷상에서 비디오 데이타 전송시 오류 내성에 관한 연구", 2002 추계 한국정보과학회 학술집, 2002.
[7] Y. Wang and Q. F. Zhu, "Error control and concealment for video Communication: an overview," Proceedings of IEEE, vol. 86, no. 5, pp. 974-997, May 1998.
[8] S. Aign and K. Fazel, "Temporal and spatial error concealment techniques for hierachical MPEG-2 video codec," Globlecom'95, vol. 3, pp. 1778-1783, 1995.
[9] X. Lee, Y. Zhang, and A. Leon-Garcia, "Information loss recovery for block-based image coding techniques-a fuzzy logic approach," IEEE Trans. Image Processing, vol. 4, no. 3, pp. 259-273, March 1995.
[10] ISO/IEC 14496-2:1999, "Information Technology-Coding of Audio/Visual Objects," Part 2:Visual.
[11] C. L. Buhan, "Software-embedded data retrieval and error concealment schemes for MPEG-2 video sequences," Proceedings of SPIE Conference on Electronic Imaging, Digital Video Compression: Algorithms and Technologies 1996, vol. 2668, pp. 384-391, USA, Feb. 1996.
[12] W. Bender, D. Gruhl, N. Morimoto and A. Lu, "Techniques for data hiding," IBM Systems Journal, vol. 35, no.3/4, pp.313-336, 1996.
[13] I. Cox, J. Kilian, T. Leighton, T. Shamoon, "Secure Spread Spectrum Watermarking for Multimedia," IEEE Trans. On Image Processing, vol.6, no.12, pp.1673-1687, 1997.
[14] G. Langelaar, I. Setawan and R. Lagendijk, "Watermarking digital image and video data," IEEE Signal Processing, vol. 17, no. 5, pp. 20-46, Sept. 2000.
[15] S. Gringeri, R. Egorov, K. Shuaib, A. Lewis and B. Basch, "Robust compression and transmission of MPEG-4 video," in ACM MM99 Electronic Proceedings, June 1999.
[16] K. I. Chan, J. Lu and J. C. Chung, "Block shuffling and adaptive interleaving for still image transmission over rayless fading channels," ITVT, vol.48, no.3, May 1999.
[17] Q. F. Zhu, Y. Wang and L. Shaw, "Coding and cell-Loss recovery in DCT-based packet video," IEEE Transaction on CSVT, vol. 3, no. 3, pp. 248-258 June, 1993.
[18] J. Song and K. J. R. Liu, "A data embedding scheme for H.263 compatible video coding," ISCAS'99, vol. 4, pp.390-393, 1999.
[19] M. D. Swanson, M. Kobayashi, A. H. Tewfik, "Multimedia data-embedding and watermarking technologies," Proceedings of IEEE, vol. 86, pp. 1064-1087, June, 1998.
[20] L. N. Kanal and A. R. K. Sastry, "Models for channels with memory and their applications to error control," Proceedings of the IEEE, vol. 66, no. 7, pp. 724-244, 1978.

김 진 옥
정보과학회논문지 : 컴퓨팅의 실제
제 8 권 제 3 호 참조