동영상 전송을 위하여 터보코드와 EREC알고리즘을 이용한 UEP설계

정회원 심 우 성*, 허 도 근**

Implementation of UEP using Turbo Codes and EREC Algorithm for Video Transmission

Woo-Sung Sim*, Do-Geun Huh** *Regular Members*

요 약

본 논문에서는 무선과 같은 대역 제한되고 잡음의 영향이 심한 환경에서 동영상 부호화를 위해 H.263을 이용하여 비트스트림을 구성하였다. 구현된 비트스트림의 실제 데이터 부분에 대한 UEP를 위해 제안된 EREC알고리즘을 적용하여 EREC서브프레임을 구현한다. 이러한 것은 블록단위의 재동기를 할 수 있어 에러의 전파를 최소로할 수 있고 INTRADC, MVD와 같은 중요 비트위치알 수 있다. 이러한 중요비트 위치를 이용하여 클래스를 분류하고 클래스정보에 의해 가변적인 puncturing 테이블을 설계하였으며 터보 코드의 부호율을 클래스에 따라 다르게 설계하였다. 채널코딩은 터보 코드를 사용하고 인터리버는 EREC 서브프레임 단위의 가변 부호율을 적용시 중요비트의 부가 비트가 제거되지 않고 가변적인 크기이지만 송, 수신단에서 항상 동일하게 설계한다. 시뮬레이션 결과 비트오류확률 측면에서 EEP와 비슷한 부호율을 갖는 UEP는 개선된 결과를 얻을 수 있었고 영상에 적용한 결과 중요비트들의 보호에의해 주관적, 객관적 화질이 좋아짐을 알 수 있었다.

ABSTRACT

In this paper, bitstreams are composed of using H.263 for a moving picture coding in the band-limited and error-prone environment such as wireless environment. EREC sub-frames are implemented by applying the proposed EREC algorithm in order to be UEP for the real data parts of implemented bitstreams. Because those are able to do resynchronization with a block unit, propagation of the error can be minimized, and the position of the important bits such as INTRADC and MVD can be known. Class is separated using the position of these important bits, and variable puncturing tables are designed by the class informations and the code rates of turbo codes are differently designed in according to the class. Channel coding used the turbo codes, and an interleaver to be designed in the turbo codes does not eliminate redundancy bits of the important bits in applying variable code rates of EREC sub-frames unit and is always the same at the transmitter and the receiver although being variable frame size. As a result of simulation, UEP with the code rate similar to EEP is obtained a improved result in the side of bit error probability. And the result of applying it to image knows that the subjective and objective quality have been improved by the protection of important bits.

Ⅰ. 서 론

영상 전송을 위한 소스 코딩의 목적은 높은 압축과 에러 전파를 줄이는 것이다. H.263과 MPEG에서는 소스 압축을 위하여 DCT를 이용하고 양자화하여 비트열로 만들기 위해 허프만 코딩을 이용한 VLC(Variable Length Coding:가변 길이 코딩)를

* 삼성전자 중앙연구소 ** 원광대학교 전자공학과
논문번호 : 99309-0809 접수일자 : 1999년 8월 9일

사용한다. 이러한 방법은 높은 압축율을 얻을수 있지만 무선과 같은 잡음이 심한 환경에서 VLC에 의한 가변 길이는 디코더에서 잘못된 길이의 비트 검출의 문제가 발생할수 있다. H.263의 경우 GBSC, PSC와 같은 동기 신호를 부여하여 이러한 문제를 해결하려 하지만 하나의 픽처나 GOB 단위의 동기이므로 많은 블록의 손실후 재동기 된다[1]. 또한 많은 동기 신호의 부여는 소스 코딩의 목적에 어긋난다. 이러한 문제를 해결하기 위해 ITU-T의 H.324[14]에서 Error Tracking, Independent Segment Decoding(ISD), Reference Picture Selection(RPS)와 ISO의 MPEG-4[ ]에서는 Video Packet Resynchronization, Data Partitioning, Reversible Variable-Length Codes(RVLCs), Header Extension Code(HEC)과 같은 Error resilient방법을 표준화 하였다. 이외에 표준화 이외의 방법으로 EREC(Error Resilient Entropy Coding), ERPC(Error Resilient Positional Coding)방법들이 있다.

채널 코딩은 블럭 코드, 컨벌루션 코드으로 대별되며 일반적으로 블록 코드에 비하여 컨벌루션 코드가 에러를 정정능력이 우수하다. 또한 컨벌루션 코드(convolutional codes) 두 개와 한 개의 인터리버를 병렬로 연결하여 10^{-5}~10^{-6}의 비트오류확률을 만족하는 터보코드(Turbo codes)가 차세대 채널 코딩 방법으로 대두되고 있다. 영상과 같이 낮은 비트오류확률을 요구하는 전송에서 터보코드는 shannon의 정리에 근접하는 채널 용량을 위한 신호대 잡음비의 개선에 있어서 현재까지 알려진 채널 코딩 방법중 가장 효과적인 것으로 알려져 있다[9].

이러한 소스와 채널 코딩은 서로 독립적으로 발전하면서 두 개의 방식이 부분적으로 만족된다면 성능 손실이 없다는 shannon의 분리 법칙에 근거하고 있으며 무한한 지연과 복잡성을 허용한 경우의 결과이다.

따라서 소스와 채널 코딩을 효과적으로 결합하기 위한 JSCC(Joint Source-Channel Coding)가 필요하며[5][6][7][8] 이러한 방법중의 하나가 UEP(Unequal Error Protection)방법이다. 이러한 것은 기존의 채널 코딩 방법이 모든 신호에 대한 동일한 부호율을 적용하는 EEP(Equal Error Protection)방법인 것에 대하여 영상 데이터의 경우 압축 방법에 의해 데이터의 중요도가 다르므로 채널 코딩을 이용할시 소스의 중요도에 따라 부호율을 다르게 하여 중요 비트를 보호하는 UEP가 필요하다. 이중 가장 대표적인 것이 RCPC(Rate Compatible Punctured Convolutional codes)로서 컨벌루션 코드와 고정된 puncturing 테이블을 이용하여 입력 데이터에 대한 여러개의 부가비트를 가변적으로 선택 함으로서 부호율을 다르게 설계하였다[2][8][11]. 하지만 이들은 실제 소스의 구체적인 위치에 대하여 언급되지 않았으며 고정 puncturing 테이블을 사용하였는데 영상 데이터의 경우 가변길이를 사용하므로 중요한 비트로 알려져 있는 DC나 MVD의 위치에 정확히 일치하는 고정된 puncturing 테이블은 불가능하다. 따라서 가변적인 puncturing 테이블을 사용하고 이것이 디코딩과정에서 적은 헤더 정보에 의하여 중요 비트의 위치를 알 수 있도록 EREC 알고리즘을 이용하여 EREC 서브프레임을 구현하고 채널 코딩 에서는 RSC(Recursive Systematic Convolutional codes) 구조를 갖고 있고 컨벌루션 코드보다 일반적으로 성능이 더 우수한 터보코드를 이용한다[9][10]. 터보코드는 연집에러를 산발 에러로 바꾸기 위한 인터리버를 갖고 있는데 EREC 적용에 의한 가변 프레임에 대한 인터리버는 성능 향상 이외에 송, 수신단에서 동일한 인터리버와 중요 클래스의 부호율에 대한 부가 비트가 제거되지 않도록 하는 인터리버 설계가 필요하다.

본 논문에서는 동영상에 비트의 중요도에 따른 UEP를 설계하기 위해 ITU-T의 동영상 표준안인 H.263의 비트스트림을 구현 하였다. 구현된 비트스트림에 본 논문에서 제안된 채널 코딩을 고려한 EREC 알고리즘에 의해 중요 비트들의 위치를 알고 위치에 따라 다른 부호율을 갖는 터보코드를 설계하였으며 채널 코딩의 성능 향상을 위해 가변 부호율에 적합한 인터리버를 제안 하였다. 이러한 방법은 기존의 EEP에 비하여 비트오류확률과 주관적, 객관적 화질 측면에서 비교하였다. Ⅱ장에서는 EREC 알고리즘과 중요 비트 위치를 알기위한 EREC 알고리즘과 제안된 EREC 서브프레임에 대하여 설명하며 Ⅲ장에서는 EREC에 의한 가변적인 EREC 서브 프레임에 대한 터보코드에 대하여 설명한다. Ⅳ장에서 실험 및 고찰에 대하여 설명한 후 결론을 맺는다.

Ⅱ. 소스 코딩

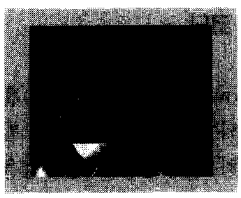
동영상은 VLBC (Very Low Bit rate Coding)을 위하여 하나의 INTRA 픽처와 여러장의 INTER 픽처로 이루어지며 INTRA 픽처는 공간적인 중복성을 줄이기 위한 DCT방법을 이용하고 INTER 픽처는

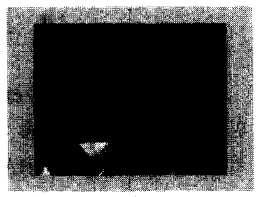
시간적 중복성을 줄이기 위하여 ME/MC(Motion Estimation/Motion Compens ation)와 이전 블록과 현재 블록의 차이값을 DCT하는 방법을 이용한다. H.263의 경우 기본 블럭(8x8)에서 DCT가 수행되며 이것이 4개 모여 하나의 매크로 블록(Macro Block)을 이루며 여기에서 ME/MC가 수행되어 MVD (Motion Vector Data)를 구하고, 11개의 매크로 블록이 하나의 GOB(Group of Block)를 이루고, GOB 단위의 동기가 이루어지며 이것이 9개 모여 하나의 픽쳐를 이룬다[1]. 이러한 구조는 전송시 발생되는 에러에 대하여 다음과 같은 특징을 갖는다.

(1) 허프만 코딩을 이용한 VLC의 에러의 열악성
(2) INTRA 픽쳐의 잡음은 INTER 픽쳐에 전파되며 전체적인 영상의 화질을 저하(INTRA 픽쳐의 중요)
(3) DCT 압축시 DC 정보와 MVD 중요성 결여

(1)의 경우 압축을 위한 VLC은 FLC(Fixed Length Coding:고정 길이 코딩)보다 잡음에 매우 열악하여 잡음에 민감한 영향을 받을 수 있다. 현재의 H.263의 비트스트림에서는 에러가 검출될 때 GOB 단위의 동기비트를 전송하므로 가변 길이일 경우 수신측에서 VLC 테이블 참조와 같은 검출 방법에 의해 에러를 발견하여 주위값의 평균이나 임의의 선택에 의해 에러 숨김(error concealment)을 한다. 하지만 하나의 에러는 VLC의 경우 다음의 비트에 영향을 줄수 있고 이러한 것이 계속 전파되어 결국 하나의 GBSC(Group of Block Start Code)가 발견될 때까지 에러가 전파 될수 있다. 이러한 결과는 음성의 경우 시간적인 shift를 가져오지만 영상은 픽쳐의 커다란 지역에서 공간적인 displace를 가져 온다. 그림 2의 (a)는 INTRA 픽쳐에 에러가 발생하여 GBSC의 재동기 방법에 의한 에러의 영향을 보여준다.

2)의 경우 동영상에서 INTER 픽쳐는 이전 픽쳐에서 MVD와 DCT계수의 차이값을 전송하므로 이전 픽쳐의 잡음은 전파될수 있다. 그림 1의 (b)~(e)는 INTRA픽쳐의 잡음이 INTER픽쳐에 잡음이 섞이지 않더라도 INTER 픽쳐로 전파됨을 보인다.


(a)


(b)


(c)


(d)

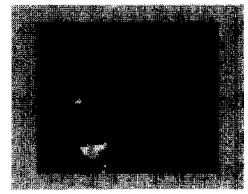

(e)

그림 1. INTRA 픽쳐에 대한 잡음의 전파

(3)의 경우 공간적 압축을 위한 DCT를 수행하면 전체 평균값인 DC 성분이 존재한다. 물론 DC값의 양자화 잡음에 의해 발생되는 블록킹 현상이 존재하지만 전송로의 잡음에 의해서도 DC값의 손실을 가져 올수 있고 이것은 주위 블록과의 블록킹 현상을 심화할 것이다.

(1)을 줄이기 위한 가장 극단적인 방법은 GOB 단위가 아닌 매크로 블록이나 블록 단위의 동기비트의 삽입이다. (2)와 (3)은 낮은 부호율을 갖는 채널 코딩을 하는 것이다. 하지만 이러한 것들은 많은 비트를 요구하게 된다. 따라서 여러 블록을 하나의 프레임으로 전송하며 블록 단위로 재동기 하고 채널 코딩에 비트위치 정보에 대한 사전 정보를 줄수 있는 EREC 알고리즘을 이용한다

1. 일반적인 EREC 알고리즘

일반적인 EREC 알고리즘은 가변길이를 고정 길이로 변환함으로서 가변 길이 에러를 전파되지 못하게 하는 방법이다[3]. EREC 출력 슬롯의 수 N_s, 소스 입력 슬롯의 수 N_b라 할 때 하나의 고정된 크기를 갖는 EREC된 전체 비트를 $T_s = \sum_{j=1}^{N_s} S_j$, 소스 전체비트를 $T_b = \sum_{i=1}^{N_b} b_i$ 이라 정의하자. 여기서 N_s와 N_b는 같은 값일 수도 있고 그렇지 않을 수도 있다. EREC 알고리즘의 방법은 N_b개의 가변적인 블록을 N_s의 고정적인 슬롯에 맞추어 넣는 방법으로 각 슬롯 S_j의 비트수를 결정하는 식은 식 (1)을 이용한다.

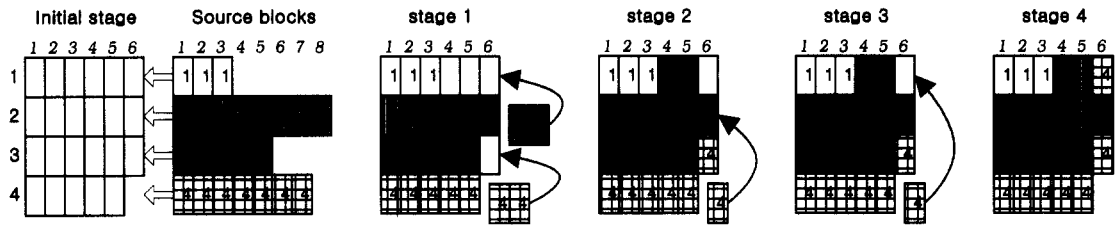


그림 2. 단 방향 offset을 갖는 EREC 알고리즘

$$S_j=\begin{cases}\overline{S}+1 & 1 \le j \le \text{remainder}(T_s / N_s)\\ \overline{S} & \text{remainder}(T_s / N_s) < j \le N_s\end{cases} \quad (1)$$

여기서 $\overline{S}$는 T_s / N_s의 정수이다. 예를 들어 $T_b=23$ 이고 $N_b=4$ 이며 각 블록 길이가 $b_i=(3,8,5,7)$ 이라면 $S_j=\begin{cases}6 & 1\le j\le 3\\ 5 & 3<j\le 4\end{cases}$ 가 된다. 이것은 1,2,3 번째 슬롯에는 6비트를 4번째 슬롯에는 5비트를 의미한다. 이때 마지막 슬롯의 비트가 5비트 이므로 이것을 알리기 위해 EOB(End of Block)를 첨가한다. EREC는 Φ_n을 j번째 블록에서 EREC를 적용하기위한 offset값이라 할 때 $j+\Phi_n$의 블록을 찾아서 빈 블록의 나머지에 채우기 위한 것이며 Φ_n은 붕괴의 종류, 데이터 분포에 따라 단방향, 양방향, 랜덤 탐색방법이 있다[4]. 그림 2는 알고리즘 구현이 용이한 단방향 탐색방법에 의한 예이다.

N_s와 $S_{j+\Phi_n}$는 송, 수신단에서 알고 있고 전송시에는 T_s 만을 전송하면 된다. 디코딩 방법은 인코딩 방법과 비슷하며 각 슬롯에 대하여 EOB 비트를 탐색하여 한 슬롯의 크기를 설정한 후 인코딩 방법과 같은 Φ_n을 이용하되 반대 방향으로 진행하며 VLC 테이블을 참조해 나가며 디코딩 한다.

2. 채널 코딩을 고려한 EREC

기존의 EREC는 하나의 영상단위에 대하여 매크로 블록이나 블록을 하나의 슬롯으로 놓고 하나의 픽쳐에 대하여 EREC를 적용하였다. 따라서 $T_s=T_b$를 만족하지 않으므로 EOB를 첨가하게 되며 이러한 것은 EOB의 잡음에 의해 하나의 픽쳐를 손실할 수 있다. 또한 EREC 알고리즘의 시작 전에 하나의 픽쳐에 대한 모든 블록 데이타를 알고 있어야하므로 큰 메모리와 지연을 요구하고 알고리즘 수행 횟수는 $N_s(N_s-1)$번 수행한다. 이러한 문제를 해결하는 방법으로 본 논문에서는 전체 영상을 각 N_s개의 블록으로 이루어진 서브프레임으로 나눈다. 이것은 $T_s=T_b$를 얻기 위해서 아래의 조건을 만족하도록 다음의 블록을 EREC 서브프레임에 배치하며 $T_s=T_b$가 되는 N_s와 S_j를 구한다.

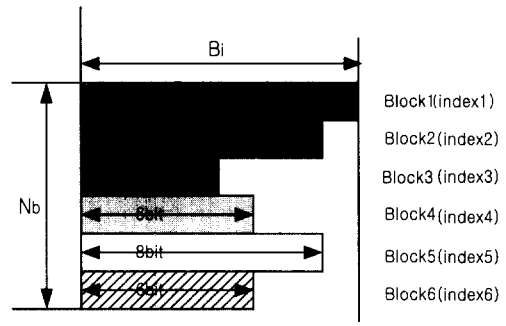
(1) 이웃하는 블록의 비트수가 같은 경우 5개 이상 16개 이하 블록까지 하나의 서브프레임으로 한다.

(2) 이웃하는 블록의 비트수가 같지 않을 경우 각 블록의 비트수를 더하여 진행하며 각 블록의 수(= N_s)로 전체 블록의 비트수를 나누어 나머지가 0일때 까지 진행 한다.

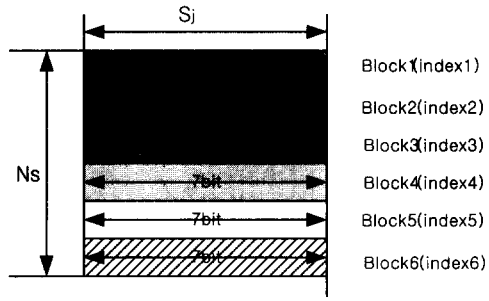
(3) (2)의 경우에 대하여 블록을 16개부터 5개까지 진행한다.

(4) (2)의 경우에 블록을 5개까지 진행하였는데도 $T_s=T_b$를 만족하지 않을 경우 블록 4, 3, 2, 1 에 대하여 (2)를 다시 수행한다.

위와 같은 방법에 의하면 EREC 서브프레임은 최대 16개 블록이며 최소 1개의 블록이 될수도 있다. 그림 3 (a)의 예는 6개 블록이 하나의 서브 프레임인 구조이며 제안된 EREC 알고리즘을 적용하기 이전의 6개의 블록으로 이루어진 가변 길이의 프레임으로 전체 T_b=42이며 N_b=6이므로 그림 3(b)와 같이 6개의 블록에 대하여 S_j=7의 서브프레임이 된다.



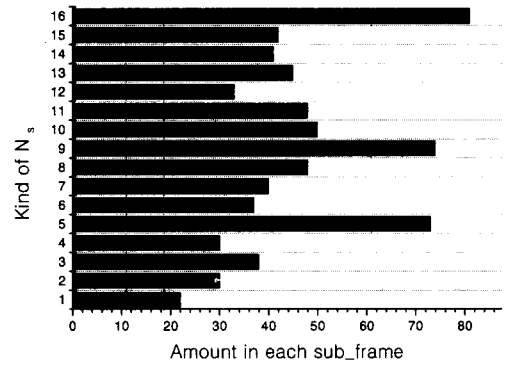
(a) EREC 알고리즘의 입력



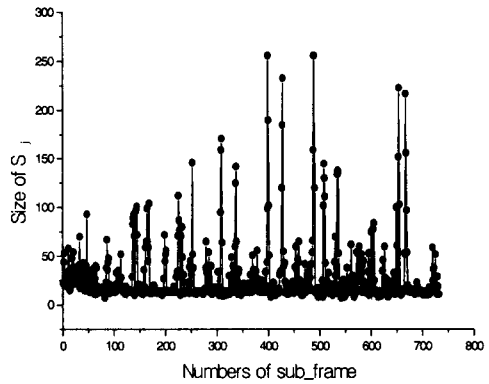
(b) EREC 알고리즘의 출력

그림 3. EREC 서브 프레임

그림 4(a)는 susie 영상의 100 픽쳐에 대해 버퍼 제어된 14픽쳐에 대한 제안된 EREC를 적용할 경우 16가지 서브프레임 종류에 속하는 개수를 나타낸다. 여기서 INTER 픽쳐의 경우 비슷한 움직임 벡터와 DCT 차이값에 의해 같은 크기의 블록이 많이 발생하게 되어 EREC 서브프레임에서 슬롯의 수 N_s가 1인 경우보다 16인 경우가 더 많이 나타남을 알 수 있다. 그림 4(b)는 그림 4(a)와 같은 susie영상에 대해 732개의 EREC 서브프레임의 S_j값을 보였다.



(a) 16가지 서브 프레임에 대한 서브프레임 수



(b) 서브프레임 수에 대한 S_j의 크기

그림 4. 64kbps에서 susie 영상 100프레임의 N_s와 S_j

수신측에 전송되는 헤더정보는 N_s와 S_j이며 이것에 의해 수신측에서 식 (2)에 의해 T_s비트와 동일한 T_b의 길이를 알 수 있다.

$$S_j \times N_s = T_s \tag{2}$$

N_s는 16를 넘지 않으므로 4비트가 필요하며 S_j의 값은 다양한 크기를 가지므로 정확한 비트로 할 경우 많은 비트수가 첨가된다. 따라서 이전 S_j 값과의 차이값을 보낸다. 이러한 차이값은 128의 값을 넘지 않음을 그림 4(b)의 결과로 알수 있어 7비트를 할당하여 전송한다. 따라서 위와 같은 방법은 $T_s = T_b$를 만족하므로 EOB가 필요 없고 하나의 EREC 서브 프레임을 위하여 11비트가 필요하다. 이러한 것은 여러개의 EREC 서브프레임으로 나누어 전송되므로 하나의 픽쳐에 대하여 EREC를 적용하는 기존의 방법에 비하여 메모리와 지연을 줄일 수 있고 INTRA 픽쳐에 비해 INTER 픽쳐의 경우 각 블록의 비트들이 비슷한 크기로 이루어져 있어 ((1)의 경우) 동일한 크기의 경우 하나의 EREC 서브 프레임으로 결정되므로 EREC 인코더와 디코더에서 복잡성이 감소한다. 디코더에서 차이점은 EOB가 아니라 헤더 정보에 의해 EREC 서브 프레임의 크기를 알수 있다.

이러한 구조의 장점은 수신측에서 N_s와 S_j의 크기를 알고 있다면 S_j 비트후에 하나의 블록들이 존재 하므로 블록 단위의 재동기가 가능하여 에러가 전파되는 것을 최대로 줄일 수 있고 S_j 비트로 이루어진 하나의 슬롯은 각 블록의 처음을 나타내며 이것은 각 입력 데이터의 INTRADC, MVD의 시작 위치를 가리키므로 이러한 위치 정보를 알수 있다는 것이다. 이러한 정보는 INTRADC값이나 MVD에 대한 에러 보호을 위하여 채널 코딩에서 가변적인 puncturing 테이블 정보로서 이용된다.

Ⅲ. 채널 코딩

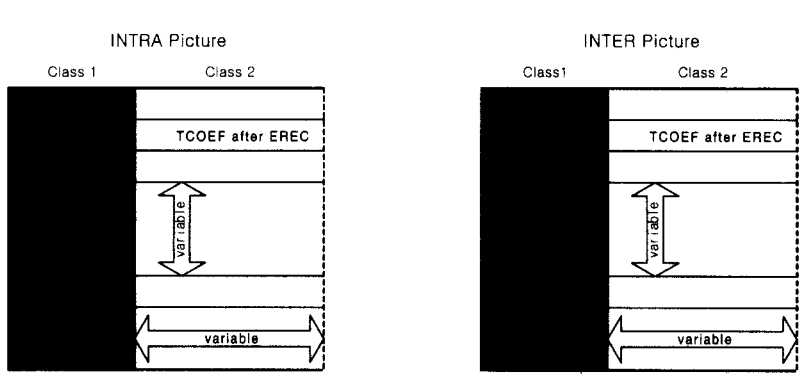
전송하고자 하는 영상 데이타에 대하여 첫 번째 픽쳐인 INTRA 픽쳐는 움직임 벡터와 차이값을 DCT하여 보내는 INTER 픽쳐와는 달리 전체 영상을 정지영상 압축인 JPEG와 비슷하게 모든 블럭에 대하여 DCT하여 보낸다. 하지만 JPEG에서는 8x8 블럭의 DCT계수 중 DC값에 대해 이전 블럭과의 차이값을 보내는 반면에 H.263에서는 고정 8비트

양자화를 수행한다. 이 때문에 JPEG방법은 수신단에서 DC값은 남아 있는 중복성을 이용해 에러 숨김을 수행하기는 쉽다는 장점이 있지만 VLC를 이용하므로 잡음의 영향에 민감하다. H.263은 DC에 대해 고정 비트를 전송하므로 잡음의 영향에는 덜 민감하지만 잘못된 8비트의 데이터를 보내더라도 수신측에서 에러 검출을 하기 어려워 에러 숨김이 용이하지 않다. 이러한 것은 전체적인 화질에 있어 블록킹 현상이 더욱 두드러져 주관적 화질을 나쁘게 만들고 다음의 INTER 픽쳐에 에러를 전파하여 전체 화질에 큰 영향을 미치므로 H.263에서 DC계수에 대하여 더욱 강한 채널 코딩이 요구된다. 또한 MVD의 에러는 INTER 픽쳐의 블록 이동과 찌그러짐이 발생할수 있고 N_s, S_j의 경우 에러가 발생하여 잘못된 EREC 블록 크기를 가져오므로 더욱 강력한 에러 정정 알고리즘이 필요하다. 하지만 상대적으로 비트스트림의 대부분을 차지하는 TCOEF는 고주파 신호로서 전체 화질에 큰 영향을 미치지 않는다. 이러한 특성을 이용하여 소스 코딩된 비트스트림에 대해 채널 코딩이 수행된다.

1. UEP(Unequal Error Protection)

채널 부호는 가장 나쁜 채널 상태를 가정하여 낮은 부호율로 부호화하면 가장 효과적인 처리가 가능하지만 저전송 채널에서의 과도한 부호량은 부적절하다. 따라서 비트의 중요도에 따른 적절한 부호량이 필요하다. 이는 소스 코딩에서 언급 된 바와 같이 N_s, S_j, MVD, INTRADC, Non-INTRADC가 나머지 TCOEF에 비하여 중요하므로 낮은 부호율을 부여하고, 나머지 TCOEF에 대하여 높은 부호율으로 부여함으로서 부호율을 조절한다. 이와 같은 부호율 조절은 전체 비트수에 비하여 INTRADC와 헤더정보가 차지하는 양이 많지 않으므로 전체적인 부호량은 TCOEF부호량을 유지할 수 있다. 따라서 중요도에 따라 두 개의 클래스로 분류하고 중요한 클래스 1에 대하여 부호율 1/3, 클래스 2에 대하여 부호율 1/2의 채널 코딩을 한다. H.263 동영상의 경우 INTRA 픽쳐의 블록은 모두 INTRADC로 8비트 고정 양자화 하지만 INTER 픽쳐의 블록은 DC값이 MV에 따라 8비트일수도 있고(INTER 픽쳐의 intra 매크로 블록) VLC 테이블을 참조(INTER 픽쳐의 inter 매크로 블록) 할 수도 있다. 하지만 VLC를 이용하더라도 DC값의 위치에 있는 정보는 중요하므로 EREC 서브 프레임의 S_j 시작 위치부터 8비트를 모두 클래스 1로 간주한다. MVD(x,y의 MVD)값도 VLC이지만(1~26비트) 전체 MV의 70~80%가 영점 근처에 존재[15]하므로 평균적으로 8비트 안에 존재하게 되어 8비트를 클래스 1로 간주한다.

비트스트림 구조에서 실제 데이터 부분인 INTRADC, MVD, Non-INTRADC, TCOEF와 EREC 서브프레임의 헤더 정보인 N_s, S_j에 대해 UEP 방법을 64kbps에서 전송된 susie 영상의 100프레임에 대하여 버퍼 제어된 14프레임에 대한 부호율은 식 (3)과 같다.



(a) INTRA 픽쳐의 EREC 서브 프레임

(b) INTER 픽쳐의 EREC 서브 프레임

그림 5. EREC 서브 프레임에 대한 UEP

$$\frac{\sum_{L=1}^{2} K_L}{\sum_{L=1}^{2} K_L/\rho_L} = \frac{\text{Total bit}}{((\text{Number of frame}\times 11)+(\text{MVD}+\text{INTRADC}))\times 3+(\text{TCOEF})\times 2} \tag{3}$$

$$= \frac{8052+44352+161960}{((8052+44352)\times 3)+(161960\times 2)} = \frac{214364}{481132} \fallingdotseq 0.4455$$

여기서 L은 클래스를 표현하고 K_L은 각 클래스의 비트수이며 ρ_L은 클래스의 부분 부호율을 나타낸다. 전체적인 부호율은 클래스 1의 부호율을 적용하는 비트수가 적으므로 클래스 2와 비슷한 부호율을 유지함을 알수 있다.

2. 터보코드와 부호율 조정

터보코드는 컨벌루션 코드 두 개가 병렬로 연결되고 하나의 인터리버로 구성된다. 일반적인 경우 부호율은 고정적으로 이루어진다. 그림 6은 1/2의 고정적인 부호율을 갖는 터보코드의 인코더이다.

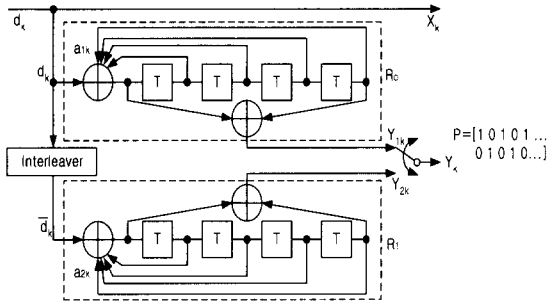


그림 6. 1/2 부호율의 Turbo 인코더

puncturing 테이블 P는 보통 고정길이의 프레임에 의해 정해지지만 소스 코딩에서의 VLC는 항상 고정된 위치에 클래스 1의 정보가 존재하지 않으므로 중요 비트에 대한 가변 부호율을 적용하기 위해서는 테이블이 항상 가변적으로 변해야 된다. 따라서 기존의 구조 그대로를 유지하며 부호율을 조정할수 있도록 클래스 1과 클래스 2의 비트 위치에 따라 가변 프레임 단위의 puncturing 테이블을 설정한다. 입력 X에 대한 인코더 RSC1과 입력 비트 X를 인터리빙한 후 RSC2에 의해 각 X_k, Y_{1k}, Y_{2k}가 만들어진다. 이때 비트들의 중요도에 따라 부호율이 조정될 경우 puncturing 테이블에 의해 부호율을 조정할 수 있다. 즉 출력 코드 C_k는 입력 X_k에 대하여 $C_k=[X_k|Y_{1k}|Y_{2k}]$로 표현된다. 여기서 각 부호율 조정은 Y_{1k}와 Y_{2k}를 puncturing 테이블에 의해서 전송될건지 아닌지를 결정할 수 있으며 본 논문에서는 클래스 1에 대하여는 Y_{1k}와 Y_{2k}의 부가 비트를 X_k와 같이 보내는 1/3, 클래스 2에 대해서는 Y_{1k}와 Y_{2k}의 부가 비트중 하나만을 선택적으로 X_k와 보내는 1/2의 부호율로 전송한다. 하나의 EREC 서브 프레임에 대한 두 개의 클래스 1과 클래스 2에 대한 클래스 구분과 puncturing 테이블은 그림 7과 같다.

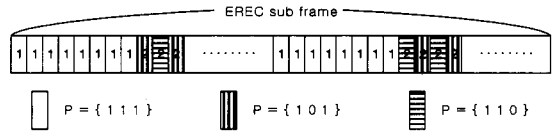


그림 7. 하나의 EREC 서브 프레임에 대한 puncturing 테이블

이러한 puncturing 테이블은 EREC 서브 프레임 단위로 이루어 지며 채널 코딩을 고려한 EREC 알고리즘시 구현된 바와 같이 임의의 $N_s \times S_j$ ($N_s=1,2,3,\ldots$)비트 후에는 항상 클래스 1성분이 존재하므로 이러한 위치 정보를 이용하여 puncturing 테이블을 구성한다. 이러한 것은 인코더에 의한 puncturing 테이블은 디코더에 전송되지 않더라도 N_s와 S_j의 정보에 의해 puncturing 정보를 알수 있다. 이렇게 구현된 전송 비트스트림을 디코더 하기 위한 방법으로 SOVA(Soft-Output Viterbi Algorithm)와 MAP(Maximum A Posteriori)알고리즘의 반복 복호 방법이 있다. MAP 알고리즘은 정보 비트의 사후 확률값을 계산하여 이 확률값이 최대가 되도록 복호하는 반면 SOVA는 모든 경로에 대하여 최대 경사 경로(Maximum Likelihood path)와 이에 근접하는 경쟁 경로(Competitor path)와의 관계를 통하여 연판정값을 산출해 낸다. 일반적으로 SOVA 알고리즘이 MAP알고리즘에 비하여 하드웨어 복잡성은 낮으나 에러정정 능력이 낮은 것으로 알려져 있다[10]. 따라서 본 논문에서는 MAP 알고리즘을 이용한 반복 복호법을 이용하여 전송된 비트스트림을 디코딩한다.

3. 인터리버

터보코드는 에러 정정 능력의 향상을 위하여 인터리버를 갖고 있다. 인터리버 설계시 데이터가 모두 동일한 중요도를 갖고 있다면 위치에 상관없는 블럭 인터리버, 랜덤 인터리버를 이용할수 있다. 하지만 중요도에 따라 부호율 조정을 할 경우 puncturing 테이블을 고려한 인터리버가 필요하다[4]. 따라서 본 논문에서 인터리버 설계시 고려 되어야 할 사항에 대하여 알아보자.

(1) 가변적 크기인 EREC 서브프레임에 대한 인터리버이므로 가변적 크기에 대한 인터리버

(2) 송신단에서 특별히 인터리버에 대한 정보가 전송되지 않더라도 수신단에서 동일한 인터리버

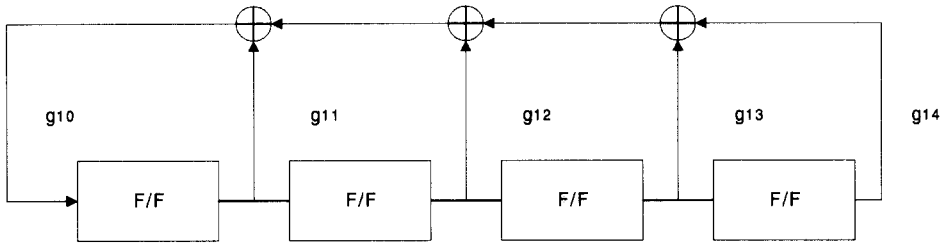


그림 8. 4개의 메모리를 갖는 PN 부호 발생기

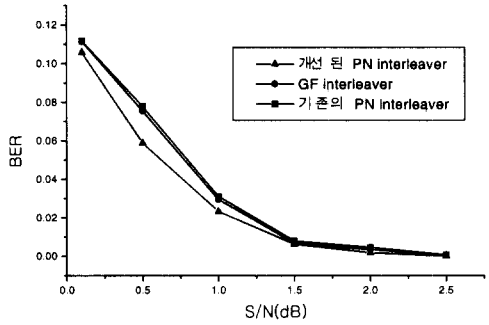
(3) 중요비트의 부가 비트가 손실되지 않도록 puncturing 테이블을 고려한 인터리버

위와 같은 고려 사항중 (1)과 (2)를 만족하기 위해 개선된 PN인터리버를 사용하고 (3)을 만족하기 위해 Mod-2 인터리버의 개념을 이용하여 각 클래스를 별도로 인터리버를 행한 후 puncturing 정보에 의해서 각 클래스의 인터리버값을 하나씩 가져오는 방법을 이용하였다. 즉 EREC 서브프레임을 클래스 1과 클래스 2로 나누는 Mod-2개념과 각 클래스의 가변적인 크기에 대하여 개선된 PN인터리버를 사용하였다.

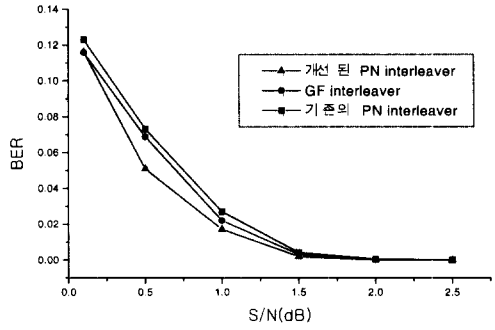
3-1. 개선된 PN인터리버

일반적인 PN인터리버는 그림 8과 같은 PN부호 발생기에서 메모리 수가 v일 때 2^v-1의 주기를 갖는 랜덤 비트를 발생시킨다.

이러한 특성을 인터리버에 사용하기 위해 각 메모리의 상태를 이용하여 랜덤 수를 발생시킨다. 하지만 PN인터리버는 고정된 2^v-1개의 크기에만 적용되므로 이것을 가변적 크기에 적합하도록 블록 인터리버와 pruning(절삭)방법을 이용한다. 먼저 입력 비트수가 N이라면 N보다 한단계 큰 2^v-1개의 PN인터리버를 발생시킨다. 이렇게 발생된 PN인터리버는 $P\times Q$ 블록에 행 방향으로 입력되고 블록의 마지막 값은 2^v값으로 대치한다. 만들어진 블록에대하여 열방향으로 읽어오며 N보다 큰 수는 절삭하는 방법을 이용한다. 이러한 방법은 GF인터리버[16]에서 가변 길이에 따른 고정된 인터리버가 송,수신단에서 일정하도록 하는 방법과 비슷하며 단지 PN인터리버의 사용함으로서 GF인터리버처럼 lookup테이블이 필요 없다는 점이 다르다. 즉 입력의 크기만 정해지면 크기에 맞는 인터리버가 항상 송, 수신단에서 일정하게 유지될수 있고 실험 결과 이러한 방법은 그림 9와 같이 GF인터리버보다 짧은 프레임에서 성능이 개선됨을 알수 있었다.



(a) 150 프레임



(b) 350 프레임

그림 9. 개선된 PN 인터리버와 GF 인터리버의 비교

3-2. 클래스를 고려한 인터리버

가변적인 부호율을 갖는 터보 코드의 인터리버 고려 조건중 (3)번째를 고려하기 위해서 Mod-2인터리버의 개념을 짝, 홀수 비트가 아닌 클래스 1, 2에 따라 고려 하였다. 예를 들어 X={1,2,3,4,5,6,7,8}이고 C_1={1,2,5,6}, C_2={3,4,7,8}이며 P=[1 1 0 0 1 1 0 0]일 때 본 논문에서 제안한 중요도를 고려한 인터리버는 그림 10의 $P(\pi(x)_{class})$이며 고려하지 않은 경우는 $P(\pi(x))$이다. 중요도에 따른 부호율을 고려하지 않은 인터리버의 경우 RSC1의 $P(x)$출력과 인터리버된 $\pi(x)$를 이용한 RSC2의 출력에 대하여 그림 10(b)처럼 재배열 할 경우에 puncturing 테

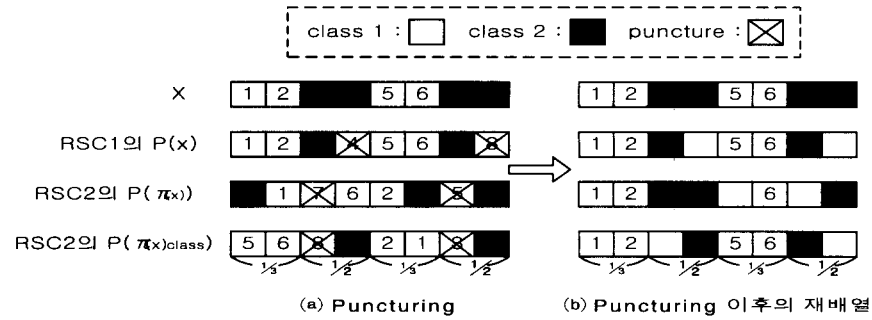


그림 10. 중요도를 고려한 인터리버

이블 P에 의해 클래스 1에 속하는 5의 부가 비트가 고려되지 않는다. 이러한 것은 중요한 클래스에 대해 부가 비트를 잃게 되어 디코더에서 에러 정정에 좋지 않은 결과를 가져온다.

따라서 $P(\pi(x)_{class})$처럼 비트의 중요도에 따라 puncturing 테이블을 설계하고 테이블에 의한 중요 비트의 위치를 알고 있으므로 이를 이용하여 클래스들의 위치에 따라 같은 클래스들에서의 인터리버를 설계하였다. 즉 클래스 1과 클래스 2에 속하는 비트의 위치에 따라 같은 클래스의 데이터 값들끼리 인터리빙한다. 각 클래스별로 인터리버 된 값들은 클래스 1의 인터리빙 된 것과 클래스 2의 인터리버 된 값을 puncturing 테이블 순서에 따라 선택함으로서 새로운 인터리버를 설계할 수 있다. 위와 같은 방법은 그림 10(b)의 $P(\pi(x)_{class})$에 있는 것처럼 UEP 방법에 적합한 인터리버로서 중요 비트의 부가 비트값이 puncturing 되어 제거되지 않는다. 즉 5의 비트에 대한 부가 비트가 puncturing 되지 않음으로서 중요비트의 부가비트에 의한 디코더에서의 성능 개선을 얻을 수 있다. 이러한 성능 개선의 결과는 Ⅳ장에서 보였다.

Ⅳ. 실험 결과 및 고찰

제안된 알고리즘의 성능 평가를 위해 H.263의 표준에 따라 QCIF 크기 영상의 INTRA 픽쳐에 대한 비트스트림을 구성한 후 제안된 방법에의해 다양한 크기를 갖는 EREC 서브프레임을 구현하고 이러한 서브프레임의 정보에 따라 puncturing 테이블을 설계하고 puncturing 테이블 정보에 따라 가변 부호율을 갖는 터보코드를 설계하였다. 이때 서브 프레임 내의 각 클래스에 속하는 비트의 위치에 따라 가변적으로 발생되고 서브 프레임의 크기만 알면 송, 수신단에서 동일한 인터리버를 갖고 중요 비트의 부가 비트를 제거 되지 않도록 하는 인터리버를 구현하였다. 터보 인코더에 의해 부호화된 비트들에 대하여 BPSK 변조하였고 변조된 신호에 채널 상의 잡음으로 AWGN 잡음을 부가 하였으며 터보코드의 디코더는 MAP 알고리즘을 이용하였다.

그림 11는 64kbps에서 susie 영상의 100개의 픽쳐에서 AWGN 잡음에 대한 각 신호대 잡음비에 대하여 기존의 EEP와 본 논문에서 제안한 UEP 방법에 대한 비트오류확률을 보였다. 여기서 점선은 전체 비트오류확률이며 실선은 클래스 1에 대한 비트오류확률이다. 이것은 낮은 SNR에서는 비슷하지만 높은 SNR에서는 약 0.3~0.5dB 정도 비트오류확률이 개선되고 특히 클래스 1의 경우 낮은 SNR이나 높은 SNR에서 평균적으로 0.5dB 향상되어 잡음에 대하여 보호됨을 알 수 있다. 그림 12는 본 논문에서 제안한 인터리버(C-random) 특성을 알아보기 위해 클래스를 고려하지 않은 랜덤 인터리버와 클래스를 고려한 랜덤 인터리버를 사용할때의 비교

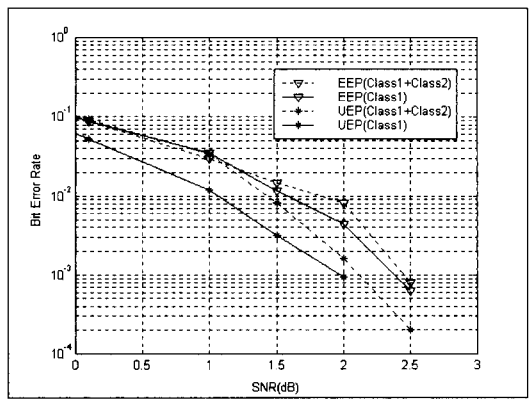


그림 11. UEP와 EEP의 비트에러 확률
(부호율 : EEP=0.5 UEP=0.4455, 4 stage,
G_1=31, G_2=17 , 5회 반복, MAP 디코딩)

이다. 전체적인 비트오류확률이 제안된 인터리버가 좋아짐을 알수 있고 특히 클래스 1에 속하는 비트들은 모든 부가 비트를 고려한 제안된 인터리버가 평균적으로 0.3dB정도 좋아짐을 알 수 있다. 하지만 비트오류확률의 개선은 UEP가 EEP에 비하여 낮은 부호율에 의해 채널코딩한 것이므로 당연한 결과라 할 수 있지만 주관적 화질이나 PSNR에서는 많은 차이를 나타낸다.

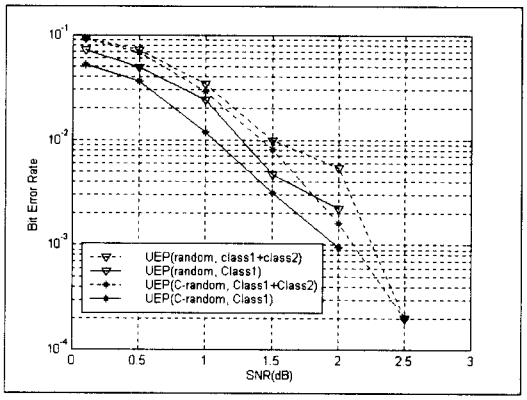


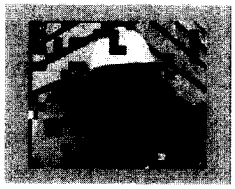
그림 12. 인터리버 성능
(부호율 : UEP=0.4455, 4 stage,
G_1=31, G_2=17 , 5회 반복, MAP 디코딩)

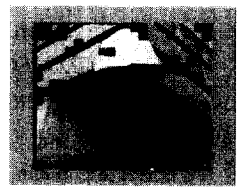
그림 13(a), (b)는 1.5dB의 신호대 잡음비에서 INTRA 픽처에 대한 EEP와 UEP에 대하여 잡음이 발생된 블럭이며 DC의 경우 특별한 오류검출을 사용하지 않으므로 AC성분의 VLC 테이블의 참조에 의해 에러로 검출된 블록이다. 정확한 DC의 전송은 UEP에 의한 방법이 에러가 적음을 알기 위하여 그림 13(a)와 (b)에서 잘못된 블럭에 대하여 DC값으로 대체한 것이 그림 13(c)와 (d)이다. DC의 대체에도 불구하고 블록킹 현상이 심한 부분은 양자화 오차에 의한 잡음 일수도 있지만 DC비트의 잡음에 의해 발생되기도 한다. 그림 14는 2.0dB의 신호대 잡음비에 대한 susie 영상의 결과이다. 이때 디스플레이를 위해 H.263의 헤더 정보(PSC, GBSC, MCBPC, CBPY..)와 N_s 와 S_i'은 에러 없이 전송되었다고 가정하고 비트오류 확률의 계산에는 고려되었다.

픽셀의 강도값에 의해 판단되는 객관적 PSNR은 비트오류확률과는 다르게 영상에 대한 화질을 판단하므로 64kbps에서 100픽처의 susie 영상의 버퍼제어된 14 픽쳐에 대한 PSNR을 보인 것이 그림 15이다. UEP 방법이 EEP에 비하여 S/N=1.5에서 평균 0.96dB, S/N=2.0에서 평균 1.35dB, S/N=2.5에서 평균 2.59dB 정도가 PSNR 측면에서도 개선되었다. 이러한 것은 그림 11의 비트오류확률에 비하여 좋은 결과를 얻을 수 있는데 DC나 MVD의 낮은 비트오류확률이 PSNR에 영향을 미쳐 객관적 측면에서 개선됨을 알 수 있다.

V. 결 론

대역 제한되고 에러의 영향에 열악한 무선환경에서 동영상의 효율적인 전송을 위해서는 소스 코딩의 특성을 고려한 채널 코딩이 요구된다. 이를 위해 제안된 EREC 방법에 의해 소스의 중요도에 따른 클래스를 분류하고 각 클래스에 따라 다른 부호율


(a)EEP에 의해 잡음블럭의 표현


(b)UEP에 의해 잡음블럭의 표현


(c)EEP에 의한 잡음의 DC값 대체

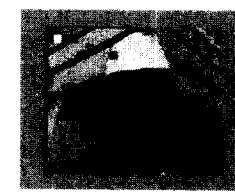
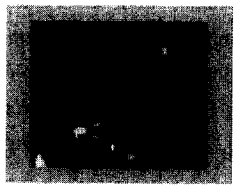
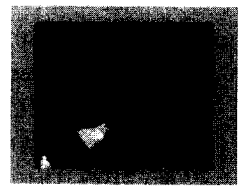
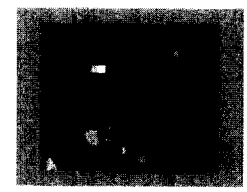

(d)UEP에 의한 잡음의 DC값 대체

그림 13. S/N=1.5에서 EEP와 UEP방법의 잡음 영향


(a)EEP에 의해 잡음블럭의 표현


(b)UEP에 의해 잡음블럭의 표현


(c)EEP에 의한 잡음의 DC값 대체

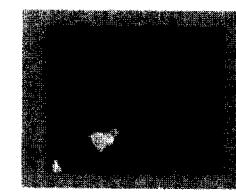

(d)UEP에 의한 잡음의 DC값 대체

그림 14. S/N=2.0에서 EEP와 UEP방법의 잡음 영향

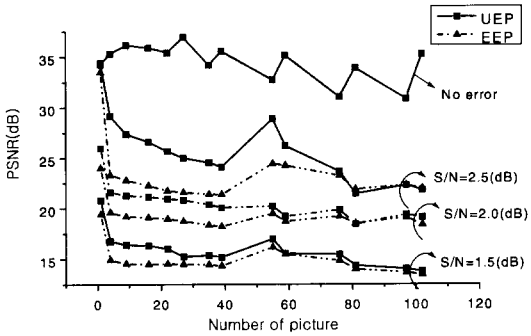


그림 15. 64kbps에서 100 프레임의 susie 영상에 대한 PSNR

조정과 이에 적합한 인터리버를 설계하였다. 이때 클래스 분류의 정보를 채널 코딩에 이용하기 위해 소스 코딩에서는 채널 코딩을 고려한 EREC 알고리즘을 제안하였다. 이러한 것은 블록 단위의 재동기가 가능하여 에러가 전파되는 것을 줄일수 있고 블록단위로 처리되므로 블록의 시작위치를 알 수 있다. 또한 중요비트 위치는 중요도에 따라 DC나 MVD와 같은 중요 클래스와 TCOEF와 같은 덜 중요한 클래스로 구분하여 채널 코딩에서 클래스에 따라 가변적인 puncturing 테이블을 만들 수 있다. 채널 코딩에서는 이러한 가변적인 puncturing 테이블을 이용한 터보코드를 이용하였으며 소스의 중요도를 고려한 인터리버를 설계하였다. 실제 H.263의 비트스트림을 설계하여 시뮬레이션해 본 결과 전체적인 비트오류확률이 약 0.3~0.5dB정도 개선됨을 알수 있었고 특히 채널 코딩에서 낮은 부호율을 갖는 중요 클래스의 비트들이 나머지 비트들에 비하여 보호 되어 주관적 화질이 개선됨을 알 수 있었다. 또한 객관적 PSNR 측면에서도 중요 비트가 보호됨에 따라 각 S/N에 대하여 평균적으로 0.9~2.6dB 정도 개선됨을 알수 있었다. 이러한 것은 제한된 채널 용량을 갖는 무선과 같은 채널에서 높은 부호율을 유지하며 중요 비트들을 보호하므로 무선 환경의 영상통신에 적합함을 확인하였다. 향후 연구 개발 방향은 실제 데이터 부분뿐만 아니라 헤더정보를 포함한 H.263의 전체 비트스트림을 고려 해야 되고 가변적인 비트스트림의 전송을 위한 전송 프로토콜 접목이 요구된다.

참 고 문 헌

[1] ITU-T Rec. H.263, "Video coding for low bitrate communication," Mar., 1996.

[2] ITU-T Rec. H.223 Annex C, "Multiplexing Protocol for Low Bitrate Mobile Multimedia Communication over Highly Error-Prone Channels," Sep., 1997.

[3] D. W. Redmill and N. G. Kingsbury, "The EREC : An Error-Resilient Techniques for Coding Variable-Length Block of Data," *IEEE Trans. on Comm.*, vol. 5, no. 4, pp. 565-574, 1996.

[4] N. T. Cheng and N. G. Kingsbury, "The ERPC : An efficient error-resilient technique for encoding positional information of sparse data," *IEEE Trans. on Comm.*, vol. 40, pp. 140-148, 1992.

[5] W. Xu, J. Hagenauer and J. Hollmann, "Joint Source-Channel Decoding Using the Residual Redundancy in Compressed Images," *IEEE Trans. on Comm.*, vol. 44, pp. 142-148, 1996.

[6] G. Caire, E. Biglieri, "Parallel Concatenated Codes with Unequal Error Protection," *IEEE Trans. on Comm.*, vol. 46, no. 5, pp. 565-567, 1998.

[7] M. Srinivasan, R. Chellappa and P. Burlina, "Adaptive Source-Channel Subband Video Coding for Wireless Channels," *IEEE Signal Processing Society 1997 Workshop on Multimedia Signal Processing, Electronic Proc.*, 1997.

[8] F. Burkert, G. Caire, J. Hagenauer and T. Hindelang, ""Turbo" Decoding with Unequal Error Protection applied to GSM speech," *in Proc., GLOBECOM'96, London, U.K*, pp. 2044-2048, 1996.

[9] C. Berrou and Glavieux, "Near optimum error correcting coding and decoding: Turbo-codes," *IEEE Trans. on Comm.*, vol. 44, pp. 1261-1271, 1996.

[10] J. Hagenauer, P. Robertson, and L. Papke, "Iterative(Turbo) decoding of systematic convolutional codes with the MAP and SOVA algorithm," *ITG Tagung, Codierung fur Quelle, Kanal und Ubertragung, Frankfurt, Germany*, pp. 21-29, Oct., 1994.

[11] J. Hagenauer, "Rate-Compatible Punctured Convolutional Codes(RCPC codes) and their

Application," *IEEE Trans. on Comm.*, vol. 36, pp. 389-400, Apr., 1988.

[12] S. A. Barbulescu and S. S. Pietrobon, "Interleaver design for Turbo codes," *Electronic Letters,* vol. 30, pp. 2107-2108, Dec., 1994.

[13] S. Dolinar and D. Divsalar, "Weight Distributions for Turbo Codes Using Random and Nonrandom Permutations," *TDA Report 42-122*, pp. 56-65, Aug., 1995.

[14] ITU-T Rec. H.324, "Terminal for Low Bitrate Multimedia Communication," 1995.

[15] L. M. Po and W. C. Ma, "A Novel Four-step Search Algorithm for Fast Block Motion Estimation," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 6, pp. 313-317, June, 1996.

[16] M. Eroz and A. R. Hammons Jr., "On the Design of Prunable Interleavers for Turbo-Codes," *IEEE Proc. of the 49th Technology Conf.*, vol. 2, pp. 1669-1673, 1999.

심 우 성(Woo-sung Sim) 정회원

1994년 2월 : 원광대학교 전자공학과 졸업
1996년 2월 : 원광대학교 전자공학과 석사 졸업
1996년 3월 : 원광대학교 전자공학과 박사과정
2000년 5월~현재 : 삼성전자 중앙연구소
<주관심 분야> 영상처리, 영상통신, 통신시스템

허 도 근(Do-geun Huh) 정회원

1975년 2월 : 울산대학교 전자공학과 졸업
1980년 2월 : 경희대학교 전자공학과 석사
1990년 3월 : 경희대학교 전자공학과 박사
1980년~현재 : 원광대학교 전자공학과 교수
<주관심 분야> 영상처리, 영상통신, 통신시스템