978-89-6211-685-4

전송경로에 의한 음성 패킷 복원력 향상 처리기술

일자: 2010년 11월 12일
부서: 슈퍼컴퓨팅본부 융합자원실
제출자: 문정훈
jhmoon@kisti.re.kr

목차

서론

인터넷을 이용하는 음성급 신호 실시간 통신 장치의 데이터 손실을 보완하는 것으로, 특히, 음성급 신호에 의한 패킷 데이터의 손실을 은닉하기 위한 잉여 정보를 전체 전송비트율 증가 없이 전송하고 수신측에서 측정한 음질평가 결과에 의하여 송신측에서 가변비트율과 데이터의 유형을 결정하여 전송하는 음성통신의 패킷 손실을 은닉하는 단말장치 및 방법에 관한 기술이다.

인간의 사회활동과 지적 활동이 활발해 지면서 지정된 상대방에게 전달하거나 의사 전달을 위하여 통신할 경우가 자주 발생하는 동시에 전달하고자 하는 정보의 양이 많아지게 되었다. 또한, 통신시간은 일정하게 정해지지 않고, 필요한때 원하는 상대방과 즉시 접속하여 통신하는 것이 일반적이다. 통신이라 함은, 소리, 기호 등을 이용하여 상대방과 직접 정보 교환할 수 없는 먼 거리 또는 원거리의 상대방과 정보를 교환하는 것을 의미하고, 운송수단 등의 발달에 의하여 통신하고자 하는 상대방이 지구상의 어느 곳에 위치하는지 모르는 경우가 있을 수 있으며, 통신은 비용을 비교적 많이 발생하는 것이 일반적이다. 상대방과 직접 접속하여 의사 및 정보를 전달하는 방식이 실시간 통신방식이고, 전달하고자 하는 정보를 송신하고 상대방이 여유 있는 시간에 확인하여 회신하도록 하는 방식이 비실시간 통신방식이다. 상기와 같이 통신하고자 하는 상대방이 지구상의 어느 곳에 위치하는지에 관계없고 실시간(REALTIME)으로 통신하지 않고 비실시간(NON-REALTIME)으로 통신하며, 전송되는 정보의 량이 매우 많아도 되고 통신비용이 비교적 저렴한 통신방식 중에 하나가 인터넷이다.

상기 인터넷은 다수 경로를 통하여 패킷을 송수신하고, 상기 송수신하는 패킷 데이터 처리에 소정 시간이 소요되며, 망의 특성에 의하여 지연(DELAY)과 지터(JITTER)가 발행하므로 이러한 특성에 민감하지 않은 비실시간 데이터 통신에 사용되어 왔다. 인터넷을 이용하여 음성급 신호를 실시간 송수신하는 인터넷 전화(INTERNET TELEPHONY) 통신 서비스를 제공하기 위해서는 신뢰성(RELIABILITY), 음성품질(VOICE QUALITY), QoS(QUALITY OF SERVICE) 및 표준화(STANDARDIZATION) 등이 보장되어야 하며, 명확성(CLARITY), 지연(DELAY) 및 에코(ECHO) 등의 요소도 포함되고, 어느 하나의 요소에 문제가 있는 경우 전체 음성 품질에 영향을 준다. 그러나 인터넷을 이용하는 경우의 저렴한 통신비용과 전 세계 어디에서나 접속하여 통신할 수 있고, 광대역 통신 시스템 장비 개발 등에 힘입어 오디오 신호가 포함되는 음성신호를 이용한 실시간 통신 기술이 계속 개발 및 발전되고 있다. 이러한 인터넷 통신기술의 발전에 의하여 실시간 오디오 미디어 통신 서비스가 더욱 선호되고 있다. 상기 인터넷은 망 접속에 혼잡 또는 트래픽(TRAFFIC)의 병목현상이 발생하는 경우, 즉 트래픽이 많이 발생하는 경우는 패킷 데이터의 전송 지연이 발생하거나 전송 오류 등에 의한 손실이 발생하므로, 음성급 또는 오디오 신호를 이용하는 실시간 통신 서비스에서는 품질 또는 QoS(QUALITY OF SERVICE)를 적정하게 확보하기 어렵다.

그러므로 실시간 응용(APPLICATION)의 경우, 패킷 손실을 줄이기 위하여 최소한의 성능을 보장하는 기술, 일예로, 대역폭 제한(BANDWIDTH LIMITATION), 패킷 손실 복구(LOSS RECOVERY SCHEME), 대기열(QUEUING), 패킷 분류(PACKET CLASSIFICATION), 전진 에러정정

(FORWARD ERROR CORRECTION: FEC) 등과 같은 기술이 필요하다.일반적으로 인터넷을 통하여 전송되는 패킷 데이터의 스트리밍(STREAMING)에서 약 5% 이상 손실이 발생하고 손실에 의하여 무음 구간이 생성되는 경우 정상적인 실시간 통신이 어려울 정도로 음질 저하를 발생하므로, 인터넷으로 스트리밍 전송되는 패킷 데이터의 손실을 보상하기 위하여 많은 기술이 개발되고 있다. 종래 기술에서는, 손실된 패킷 데이터에 의하여 발생하는 무음구간을 원래의 오디오 신호 파형(WAVEFORM)으로 근사화 시켜 보간(INTERPOLATION)하는 방식을 이용한다. 또한 다른 종래 기술에서는 손실이 발생한 구간에서 잡음을 사용하거나 반복 재생 등과 같은 방식을 이용하여 보간 한다.

일반적으로 오디오 통신신호에 의한 하나의 패킷은 약 20 내지 30 밀리세컨드(MILLISECOND) 시간에 해당하므로 단 하나의 패킷에 손실이 발생하는 경우에도 끊김 간격 또는 무음 구간이 상대적으로 크게 되어 보간(INTERPOLATION)하기 어렵다.
이러한 종래 기술을 보다 개발시킨 것으로, MPEG(MOVING PICTURE EXPERTS GROUP) 오디오 코덱을 위한 압축 영역에서의 보간법 기술을 제시한 것 등이 있다. 또한, PARAMETRIC AUDIO MODELING을 사용하는 방법을 제시한 것으로 등이 있다. 패킷 데이터의 손실을 복원하기 위한 알고리즘을 패킷 손실을 은닉하는(PACKET LOSS CONCEALMENT: PLC) 알고리즘이라고 한다. 상기 PLC 알고리즘은 크게 송신단 기반과 수신단 기반의 두 방법으로 나누어진다. 상기 송신단 기반의 PLC 알고리즘은 수신단에서 수신한 패킷에 손실이 발생한 경우, 이전 패킷에서 함께 전송된 잉여 정보(REDUNDANCY) 또는 부가정보(SIDE INFORMATION)를 손실된 패킷 대신에 이용하도록 하는 것으로, 전진 에러 정정(FORWARD ERROR CORRECTION: FEC), 인터리빙(INTERLEAVING), 재전송(RE-TRANSMISSION) 방법 등이 있다. 또한, 수신단 기반의 PLC 알고리즘은 수신단에서 유효한 패킷 데이터, 즉 손실이 발생된 패킷 이전에 수신된 패킷 데이터와 이후에 수신된 패킷 데이터 등을 이용하여 상기 손실된 패킷 데이터를 복원하는 방법으로, 삽입(INSERTION), 보간(INTERPOLATION), 모델 기반 복원(MODEL-BASED RECOVERY) 방법 등이 있다.

그러나 상기와 같은 종래 기술의 송신단 기반 PLC 알고리즘은 손실이 발생된 패킷 데이터의 복원을 위하여 좋은 결과를 제시하고 있으나, 해당 정보의 추가적인 전송에 의하여 전송 비트율이 증가하고, 이전 프레임 또는 패킷에 대한 인코딩을 다시 처리하여야 되는 등의 처리 지연(PROCESSING DELAY)이 발생하는 문제가 있다. 또한, 상기의 종래 기술에 의한 것으로, 수신단 기반 PLC 알고리즘은 송신단이 제공하는 추가적인 정보가 없어도 수신단에서 자체적이며 독립적으로 손실된 패킷을 복원하는 장점이 있으나, 패킷 손실률이 높아지면 자체적으로 복원하기 어려우므로 음성급 통신신호의 품질(QoS)이 급격히 떨어지는 문제가 있다. 종래 기술은 상관관계가 높은 음성신호에 손실이 발생하는 경우, 손실이 발생한 구간의 데이터를 예측하여 어느 정도의 복구를 할 수 있지만, 음악 등과 같은 광대역 오디오 스트림의 경우에는 패킷 단위 신호 사이의 상관관계를 예측하기 어려워 부가정보 없이 자체적으로 복원하기 어려운 등의 문제가 있다.

따라서 네트워크에서 패킷 손실이 발생할 때 수신단에서 패킷 손실 은닉을 수행할 수 있도록 미리 송신단에서 잉여의 음성 데이터를 전송하는 방법을 고안한 것으로, 여러 프레임에 대한 잉여의 음성 데이터를 생성함에 있어서 여러 종류의 코덱으로 인코딩함으로써 증가된 대역폭을 동일하게 유지할 수 있도록 하였다. 또한 수신단의 패킷 손실 정보는 RTCP (Real Time Control Protocol) 채널을 통해 피드백 하는 방식이다.

잉여의 음성 데이터를 생성함에 있어서 여러 종류의 코덱을 사용하여 인코딩하는 대신에, 가변 비트율 오디오 코덱을 이용하여 메인 프레임과 이전 프레임(잉여 음성 데이터)을 인코딩함으로써 대역폭 증가없이 동일한 대역폭을 유지할 수 있다. 또한 수신단에서는 수신 오디오 스트림에 대해 주기적으로 음질평가를 수행하여 그 결과를 송신단으로 피드백함으로써 송신단의 패킷 손실 은닉 방법에 대한 좀더 명확한 근거를 제시할 수 있다. 이와 동시에, 피드백 정보를 별도의 RTCP 채널을 사용하지 않고, 오디오 데이터 전송 채널인 RTP (Real Time Protocol)의 RTP payload format을 이용하여 전송함으로써 부가적인 채널 overhead가 없다.

기존 입력음성을 고비트율 정보와 저비트율 정보로 인코딩하고 저비트율 정보를 부가 음성 데이터로서 패킷을 구성하여 송신하며, 수신단에서 패킷 손실이 발생할 경우에 부가 음성 데이터를 이용하여 복원하는 방법을 사용하지만 본 기술에서는 송신단은 수신단에서 피드백해주는 음질평가 결과를 기준으로 송신단의 부가 음성 데이터 전송 여부 및 데이터 유형을 결정한다. 또한 송신단에서 부가 음성 데이터를 전송하는 경우에도 추가적인 비트율 증가를 막기 위해 메인 음성 데이터와 부가 음성 데이터의 비트율을 가변하여 전체 비트율이 유지되도록 한다.

– 실시간 오디오 컨퍼런싱에 대한 IP 네트워크의 문제점

인터넷의 대중적인 사용과 광대역 인프라 구조를 기반으로, 인터넷을 통한 음성 및 오디오 통신 서비스는 더욱 선호되고 있다. 그러나 인터넷 즉, IP 네트워크는 기본적으로 파일 전송 및 이메일 등의 비실시간 애플리케이션 제공을 목적으로 구축되었으며, 네트워크 혼잡이 발생하였을 때 종종 데이터 패킷을 지연시키거나 손실시킨다. 따라서 실시간 미디어 서비스의 QoS(Quality of Service)를 보장하기 어렵다. 특히 음성 및 오디오 컨퍼런싱과 같은 실시간 오디오 서비스에 있어서 이는 컨퍼런싱 단말에 지연 및 패킷 손실을 야기할 수 있으며, 일반적으로 패킷 손실이 5% 이상이 되는 경우 대화를 방해할 정도의 음질저하를 가져온다고 인지한다.

음성 및 오디오 컨퍼런싱에 있어서 패킷 손실은 음성 및 오디오의 스트리밍 중에 끊김현상을 발생시킨다. 이러한 끊김으로 발생하는 무음 간격을 채우기 위한 일반적인 방법은 원래의 파형(waveform)을 근사화시키는 것이다. 그러나 일반적으로 하나의 오디오 패킷은 20~30 millisecond 시간에 해당하기 때문에, 단 하나의 손실 패킷에 의해 생기는 끊김 간격이 상대적으로 크며 따라서 보간(interpolation) 하기에 어렵다.

- 패킷 손실을 복원하는 종래의 기술

이러한 패킷 손실을 복원하기 위해 많은 기술들이 연구되어 왔다. 대부분이 음성 신호를 위한 것으로, 종래의 기술들은 잡음 또는 파형의 삽입이나 반복 재생과 같은 단순한 방법으로부터 시작되었다. 이후 보다 진전된 기술로는 MPEG 오디오 코덱을 위한 압축 영역에서의 보간법과 parametric audio modeling을 사용하는 방법(J. Lindblom and P. Hedelin, "Packet loss concealment based on sinusoidal extrapolation," in Proc. ICASSP, pp. 173-176, May 2002.) 등이 있다.

이와 같은 패킷 손실 복원을 위한 알고리즘을 패킷 손실 은닉(Packet Loss Concealment: PLC) 알고리즘이라고 부른다. PLC 알고리즘은 크게 송신단 기반과 수신단 기반의 두 방법으로 나뉠 수 있다. 송신단 기반의 PLC 알고리즘은 패킷 손실 발생 시, 수신단이 손실된 패킷 대신에 사용할 수 있도록 이전 패킷의 잉여 정보(redundancy) 또는 부가 정보(side information)을 전송하는 방식으로, 전진 에러정정(Forward Error Correction: FEC), 인터리빙(interleaving), 재전송(re-transmission) 방법 등이 있다. 수신단 기반의 PLC 알고리즘은 송신단과 무관하게 오직 수신단에서 유효한 데이터, 즉 수신된 이전·이후 패킷 데이터 등을 활용하여 손실된 패킷을 복원해내는 방법으로, 삽입(insertion), 보간(interpolation), 모델 기반 복원(model-based recovery) 방법 등이 있다.

일반적으로 송신단 기반의 PLC 알고리즘은 손실 패킷 복원을 위해 보다 확실한 방법을 제시하지만 추가적인 정보 전송으로 인해 전송 비트율이 증가하고, 이전 프레임에 대한 인코딩을 다시 하는 등의 처리 지연(processing delay)이 높은 단점이 있다. 반면 수신단 기반의 PLC 알고리즘은 송신단의 추가적인 정보 전송 없이 송신단에 독립적으로 자체적인 패킷 손실 복원을 수행하는 장점을 가지지만, 패킷 손실율이 높아지면 자체적으로 신호를 복원하기 어려워져 오디오 품질이 급격히 저하되는 단점이 있다또한 음성인 경우에는 음성신호 간 상관관계가 높아 손실 데이터에 대한 예측이 어느 정도 가능하지만, 음악과 같은 광대역 오디오 스트림인 경우에는 신호 간 상관관계를 예측하기가 어려워 부가 정보 없이 자체적으로 복원하기 어렵다.

- MPEG-2 AAC 오디오 코딩 알고리즘

AAC의 특징을 간략히 정리하면 다음과 같다
8-96 kHz sampling frequency 지원
Multichannel/Multilingual: 최대 8.1 채널, 7개국 부가 음성정보 전송
320 kbit/s (5.1ch)에서 원음과 식별할 수 없는 고품질 제공
256/2048 MDCT, 심리음향 모델, Huffman coding, TNS, Prediction 등 MPEG-1 과 다른 압축 도구(tool)을 사용

MPEG-1 이나 MPEG-2 에서 Layer 라는 개념을 쓴 것과 유사하게 AAC 는 프로파일(profile)이라는 계층이 있다. AAC-main, AAC-LC (Low Complexity), AAC-SSR (Scalable Sample Rate)의 3개의 프로파일로 되어 있으며, 활용분야에 따라 프로파일을 선택하여 사용할 수 있다.

AAC-main 프로파일은 AAC 표준안에서 제안하는 압축 도구 중에 AAC-SSR 프로파일에서 사용되는 이득 조절(gain control)을 제외한 모든 도구를 사용한다. AAC-LC 프로파일의 경우, AAC-main 프로파일에서 Prediction 부분이 제외되고, TNS의 계수를 12개로 제한하는 등, 복잡도가 높은 도구를 사용하지 않는다. AAC-SSR 프로파일은 AAC-LC 프로파일에 hybrid filter bank (IPQF + divided IMDCT) 를 사용한 것이 특징이고 다른 프로파일에는 사용되지 않는 gain control 이 사용된다. AAC-LC 프로파일은 AAC-main 프로파일에 비해서 음질의 차이는 많지 않지만 복잡도에서 많은 이득을 얻을 수 있기 때문에 가장 널리 사용되고, MPEG-4 AAC, AAC+(HE-AAC), Enhanced AAC+ 등 상위 버전의 기초가 된다. MP3는 192 kits/s 에서 원음과 투명성(transparency)을 가지고, MPEG-1 Layer 2는 256 kbit/s에서 투명성을 가지는 것에 비해, AAC는 128 kits/s 에서 투명성을 가지게 된다. AAC의 구조도는 그림1과 같다.

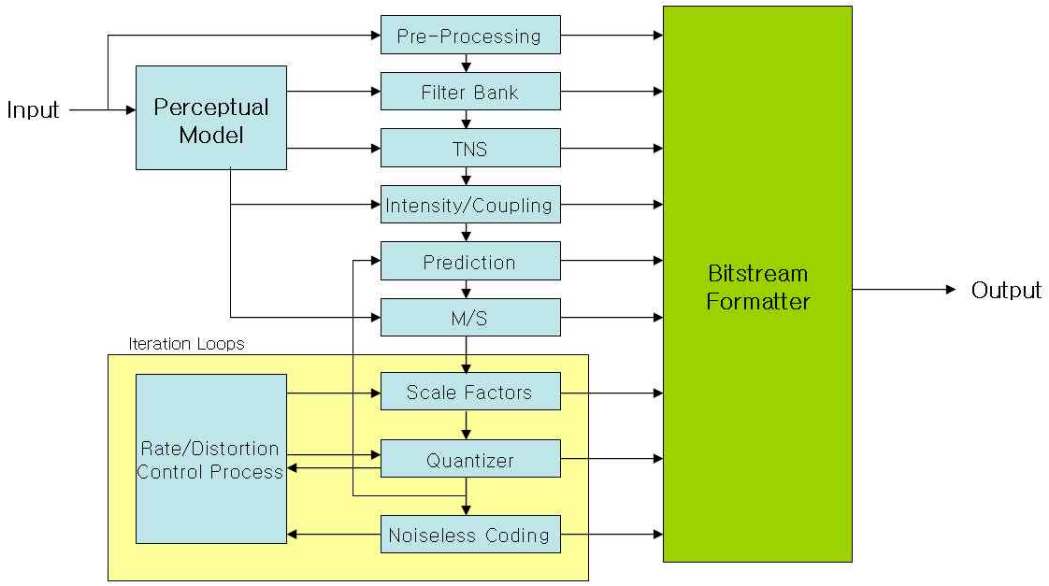


그림 1 MPEG-2 AAC encoder structure

· 심리음향 모델 (Psychoacoustic Model)

AAC에서 심리음향 모델은 MP3에 사용되는 심리음향 모델과 유사하다. 심리음향을 통해 윈도우의 종류, SMR (Signal Masking Ratio) , PE (Perceptual Entropy) , 최대 허용 에러 에너지, 한 프레임에서 사용 가능한 비트 등을 구한다.

· 필터뱅크 (Filterbank)

AAC에서 T/F 변환을 위하여 TDAC (Time Domain Aliasing Cancellation) MDCT를 사용한다. 이것은 Dolby 사의 AC3 코덱과 유사하다. 기존 MPEG-1 layer3와 다른 점은 다음과 같다.

256/2048 MDCT filterbank

KBD (Kaiser-Bessel Derived), Sine window

AAC는 다른 코덱보다 긴 256/2048의 크기를 가지는 윈도우를 사용한다. 2048윈도우를 통해 steady-state signal 처리하기 위해 23Hz 주파수 해상도를 가진다. 256 윈도우를 통해서는 2.7msec의 시간 해상도를 가진다. 256의 short 윈도우에서 long 윈도우로 바뀌기 위해서는 다음 그림2에서 보는 것과 같이 start 윈도우 및 stop 윈도우가 필요하다.

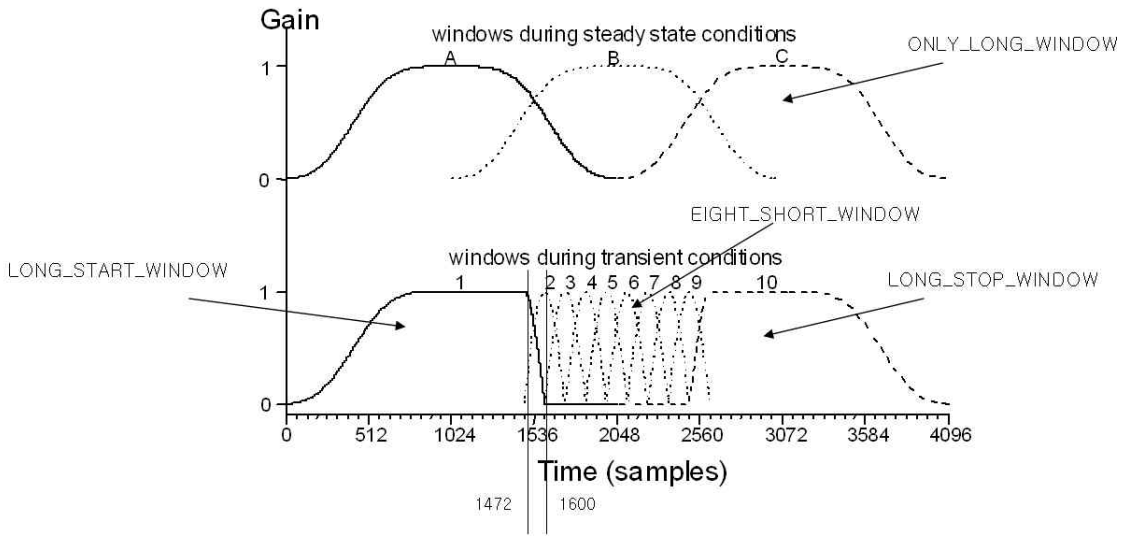


그림 2 Window switching

2048 MDCT를 할 경우 pre-echo가 발생하게 된다. pre-echo는 주파수 영역에서 데이터를 양자화하기 때문에 프레임의 전체적인 시간 영역에서 양자화 잡음이 일어나기 때문이다. 이를 위해서 그림 2와 같은 transient 성분을 가지는 데이터를 처리할 때는 256 MDCT를 총 8회 사용하는 block switching을 사용한다. 이를 위해서 AAC에서는 심리음향 모델에서 인지 엔트로피 (perceptual entropy) 를 이용하여 256/2048 block switching을 조절할 것을 ISO 표준안에서 권고하고 있다. 하지만 표준안보다 효율적인 처리를 위해 이보다 입력 데이터에 HPF (High Pass Filter)를 통과시켜 고주파 에너지를 가지고 판단하는 방법이 최근 많이 사용되고 있다.

· TNS (Temporal Noise Shaping)

pre-echo는 데이터가 없는 부분에 잡음이 발생하기 때문에 음질에 큰 영향을 미친다. filterbank에서 block switching을 통해 pre-echo를 일부 제거할 수 있지만, short 윈도우 프레임 안의 pre-echo는

제거하진 못한다. 이것을 제거하기 위해 TNS가 사용된다. TNS의 기본적인 개념은 예측기를 통한 envelope 성질과 시간/주파수 영역간의 쌍대성(duality)을 이용하여 설명할 수 있다. 구조도는 그림 3, 그림4와 같다.

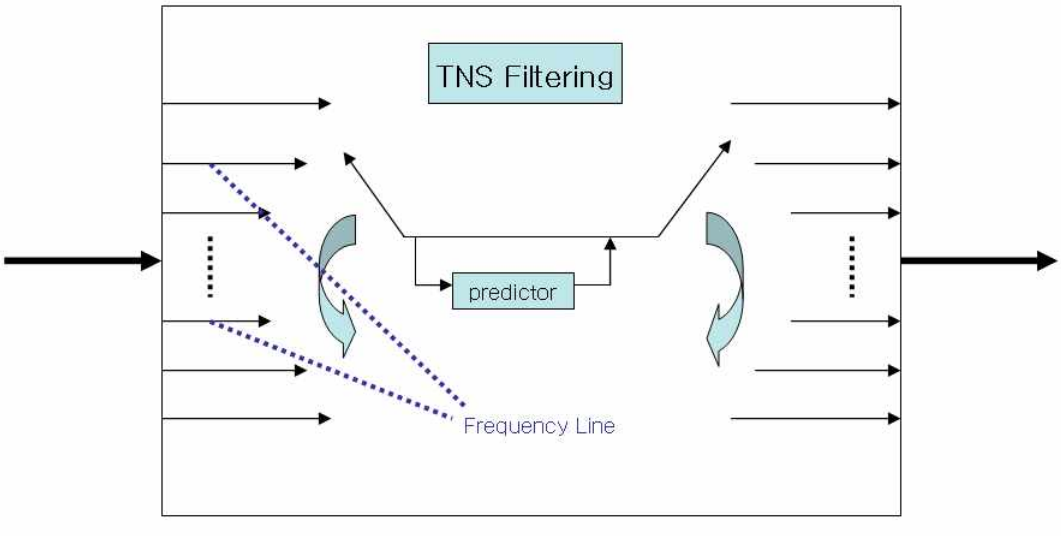


그림 3 TNS structure

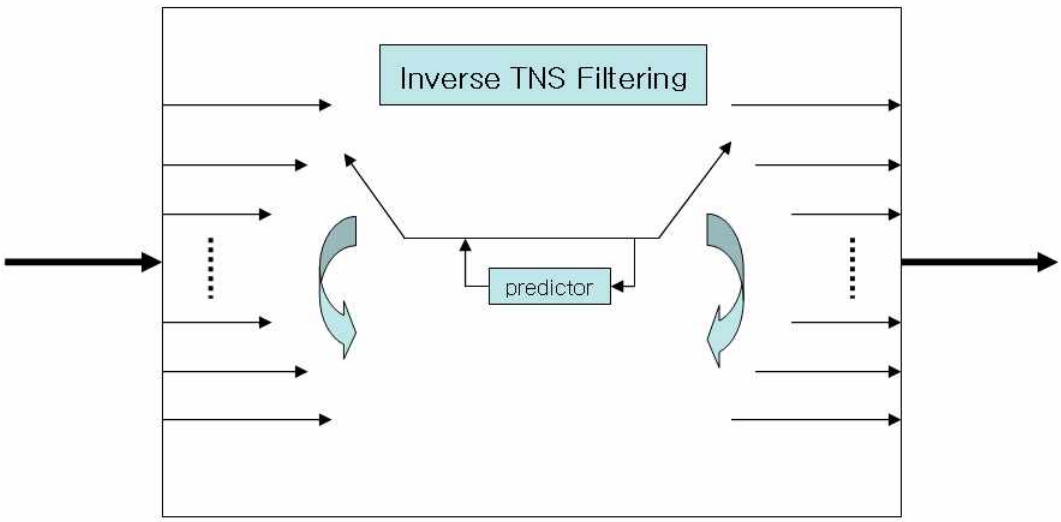


그림 4 Inverse TNS structure

음성신호처리에서 사용하는 예측은 주파수 축에서 예측기의 envelope와 residual의 주파수 값의 곱으로 나타난다. 이와 동일하게 주파수 축에서 예측을 사용하면 시간 축에서 예측기의 envelope와 residual의 시간 값의 곱으로 나타난다.

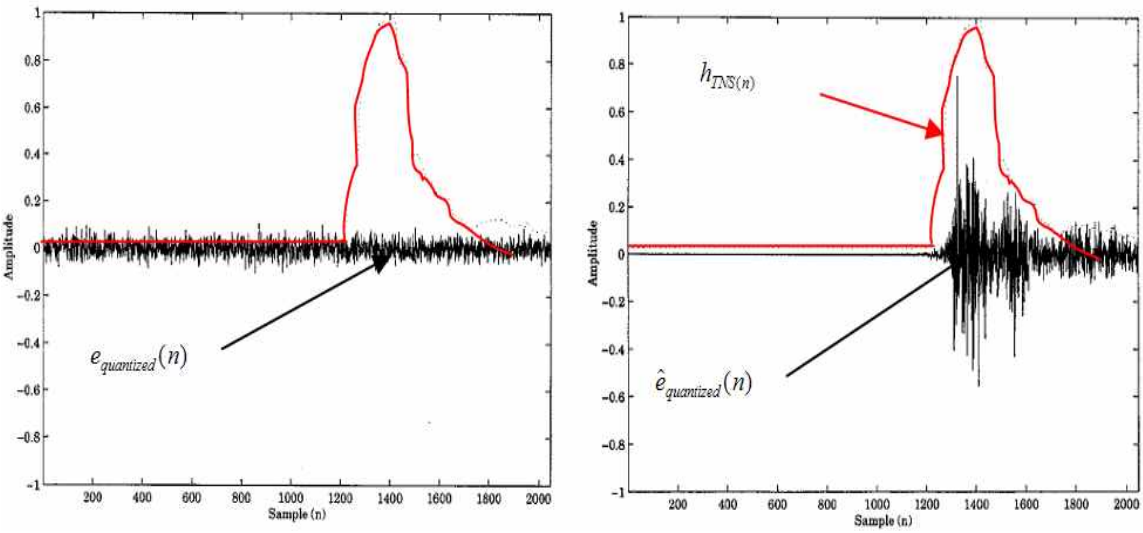


그림 5 Quantization error related TNS

Pre-echo는 주파수 축에서 신호를 양자화하기 때문에 생기는 프레임 내부 전반적인 양자화 잡음이다. 하지만 TNS를 사용할 경우 무음구간에서는 양자화 잡음에 0에 가까운 값을 곱해주기 때문에 transient한 신호를 처리하더라도 pre-echo를 많이 제거한 신호를 얻을 수 있다. TNS 최대 차수는 main 프로파일에서 short windows의 경우 7, long windows의 경우 20을 사용하고, LC 프로파일에서 short windows 의 경우 7, long windows의 경우 12를 사용한다. 또한 모든 scale factor band 에 대해 TNS를 적용하는 것이 아니라 다음 표에서 나타난 것처럼 최대 TNS scale factor band를 정한다. 저주파 성분이 필터의 안정성을 보장하지 못하기 때문에 2.5 kHz 이상에 해당하는 scale factor band에서 최대 TNS scale factor band까지 TNS를 적용한다.

Sampling Rate [Hz]	Long windows	Short windows
96000	31	9
88200	31	9
64000	34	10
48000	40	14
44100	42	14
32000	51	14
24000	46	14
22050	46	14
16000	42	14
12000	42	14
11025	42	14
8000	39	14

표 1 Processing structure and method of SBR

TNS 인코딩 과정을 세부적으로 알아보면 다음과 같다. 먼저 Levinson-Durbin 알고리즘을 이용하여 반사계수와 예측 이득을 구한다. 예측 이득이 1.4를 넘을 경우 TNS를 사용하고, 1.4를 넘지 않을 경우에는 TNS를 사용하지 않는다. 예측 이득이 1.4를 넘을 경우 구한 반사계수를 양자화한다. LPC 계수를 양자화하지 않고 반사계수를 양자화하는 이유는 반사계수가 LPC 계수에 비해 양자화 에러에 강인하기 때문이다. 양자화 과정 후 역과정을 통해 복원된 반사계수 중에 0.1보다 작은 반사 계수는 0으로 내림을 한다. 이러한 과정을 통해서 TNS 예측기의 차수를 조절한다. 0.1보다 큰 반사계수를 이용하여 예측 계수를 구한다. 마지막으로 예측 계수를 이용하여 TNS 필터링 과정을 거친다. 이때 사용되는 필터는 MA (Moving Average) 필터를 사용하고, 디코딩 과정에서는 AR (Auto Regressive) 필터를 사용한다.

· M/S 스테레오 코딩 (M/S Stereo Coding)

스테레오 채널을 압축할 때 left 채널과 right 채널의 신호가 유사할 경우 left 채널과 right 채널을 독립적으로 압축하는 것보다 두 채널의 합과 차이를 압축하는 것이 효율적이다. 이를 위해 mid/side 채널은 다음과 같은 식으로 구한다.

$$M = \frac{L+R}{2} \; and \;\; S = \frac{L-R}{2}$$

M/S 스테레오 코딩은 left, right, mid, side 신호의 에너지를 비교하여 left, right 채널을 압축하는 것이 효율적인지, mid, side 채널을 압축하는 것이 효율적인지 결정하게 된다. 그리고 scalefactor band 단위로 M/S 스테레오 코딩을 사용할 것인지, 아닌지를 결정한다. 모든 scalefactor band는 intensity stereo coding과 M/S 스테레오 코딩을 둘 다 사용하지 못한다.

· 음압 스테레오 코딩 (Intensity Stereo Coding)

스테레오 코딩에서 인간이 소리를 인식하는데 있어서 고주파 신호는 주로 고주파 성분의 에너지에 의존한다. 그렇기 때문에, 스테레오 코딩에 있어서 6 kHz가 넘는 고주파 성분에 대해서 두 채널 모두 고주파 성분을 압축하는 것이 아니라, 한 채널의 고주파 성분과 에너지 값을 압축하여 압축효율을 높인다. 해당 scale factor band의 에너지 차이는 다음과 같이 계산한다.

$$is\_position[sfb] = n\,\mathrm{int}\left[2 \cdot \log_2\left(\frac{E_l[sfb]}{E_r[sfb]}\right)\right]$$

또한 아래의 식을 통하여 고주파 성분의 대표 값을 정한다. 이 대표 값은 right 채널에 scalefactor 대신 3 bit로 보내고, 15번 Huffman codebook을 이용하여 noiseless coding 과정을 거친다.

$$spec_i[i] = (spec_l[i] + spec_r[i]) \cdot \sqrt{\frac{E_l[sfb]}{E_s[sfb]}}$$

여기서 구한 $spec_i[i]$를 $spec_l[i]$에 대입하고 $spec_r[i]$는 0으로 설정을 한다. 또한 right 채널에 고주파 성분이 없어지기 때문에, prediction 상태를 강제적으로 "Off"시킨다.

· 예측 (Prediction)

예측은 블록(block)과 블록 사이에 유사성이 클 때, 두 블록 사이의 차이를 보내는 inter-frame prediction 방법이다. 예측 계수를 압축 과정과 복원 과정에서 모두 구하는 backward adaptation prediction을 사용하기 때문에 예측계수를 전송하지 않고, 예측기를 통과한 여기신호(residual)를 양자화 한다. 과거 두 프레임에 대한 스펙트럼 정보를 이용하여 계산하기 때문에, 4096개 (프레임 수(2) * 프레임 당 샘플 수(2048))의 예측기가 필요하다. 이는 코덱의 복잡도를 매우 높이는 요인이 된다. (AAC-main 프로파일의 50%)

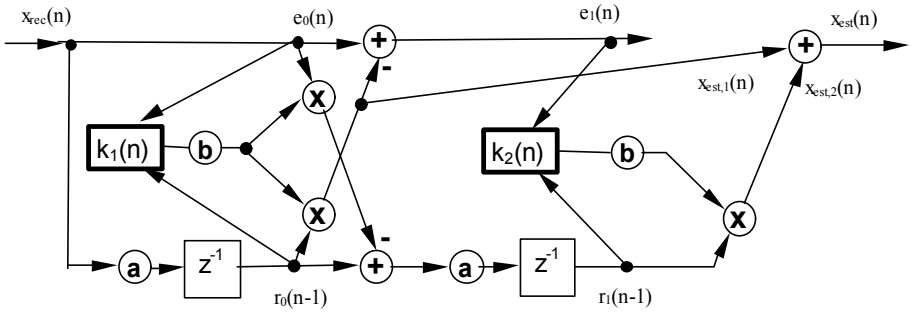


그림 6 Prediction structure

· 양자화 (Quantization)

양자화 하는 과정은 기존 MP3와 동일하다. 비균일(nonuniform) 양자화를 하며 Inner, Outer Iteration loop를 통해서 양자화의 step size를 조절하여 최적의 비트를 할당하는 방법을 찾는다. AAC에서 사용하는 양자화기는 다음과 같다.

$$x_{quant}(i) = \mathrm{int}((abs(mdct(i)) \times 2^{(-\frac{1}{4} \times (common\_scalefac - scalefactor))})^{\frac{3}{4}} + 0.4054)$$

가장 적합한 common_scalefac와 scalefactor를 구하기 위하여 Iteration loop를 사용한다. 크게 inner loop와outer loop로 나뉘며 각 특징을 살펴보면 다음과 같다.

Inner iteration loop

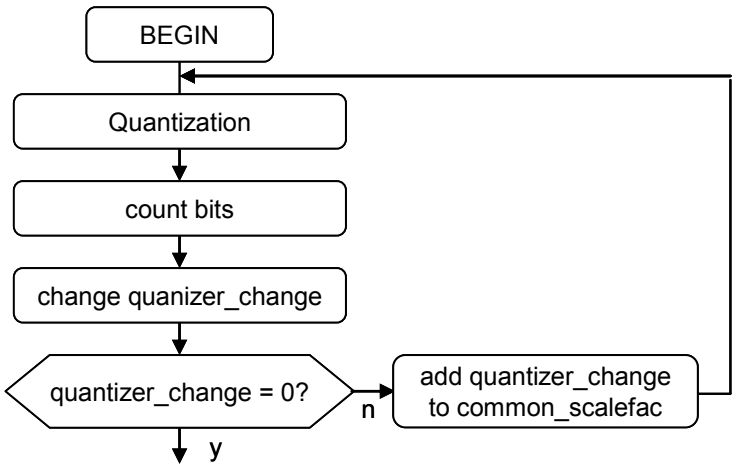


그림 7 Inner loop

Inner loop에서는 주파수 성분을 양자화 한 후 사용한 비트를 계산한다. 양자화에 사용한 비트 수와 현재 프레임에서 사용 가능한 비트 수를 비교하면서 가장 적절한 양자화기의 global step size를 찾는다.

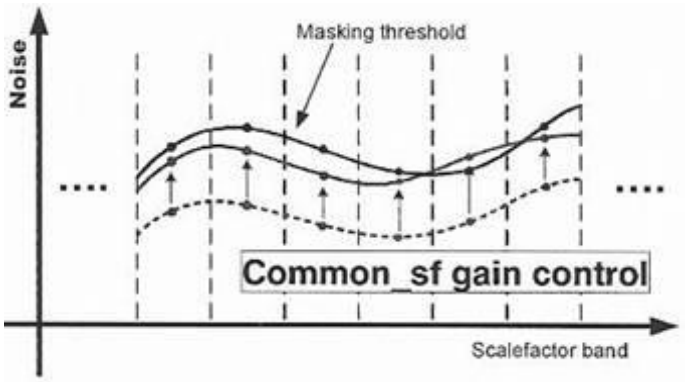


그림 8 Total gain control of inner loop

Outer iteration loop

Outer loop에서는 inner loop에서 계산한 양자화 과정을 통해 양자화 잡음을 계산한다. 각 밴드 별로 계산하여 최대 허용 잡음 에너지보다 양자화 잡음이 클 경우 해당 scale factor band의 step size를 줄여서 다시 양자화하게 된다.

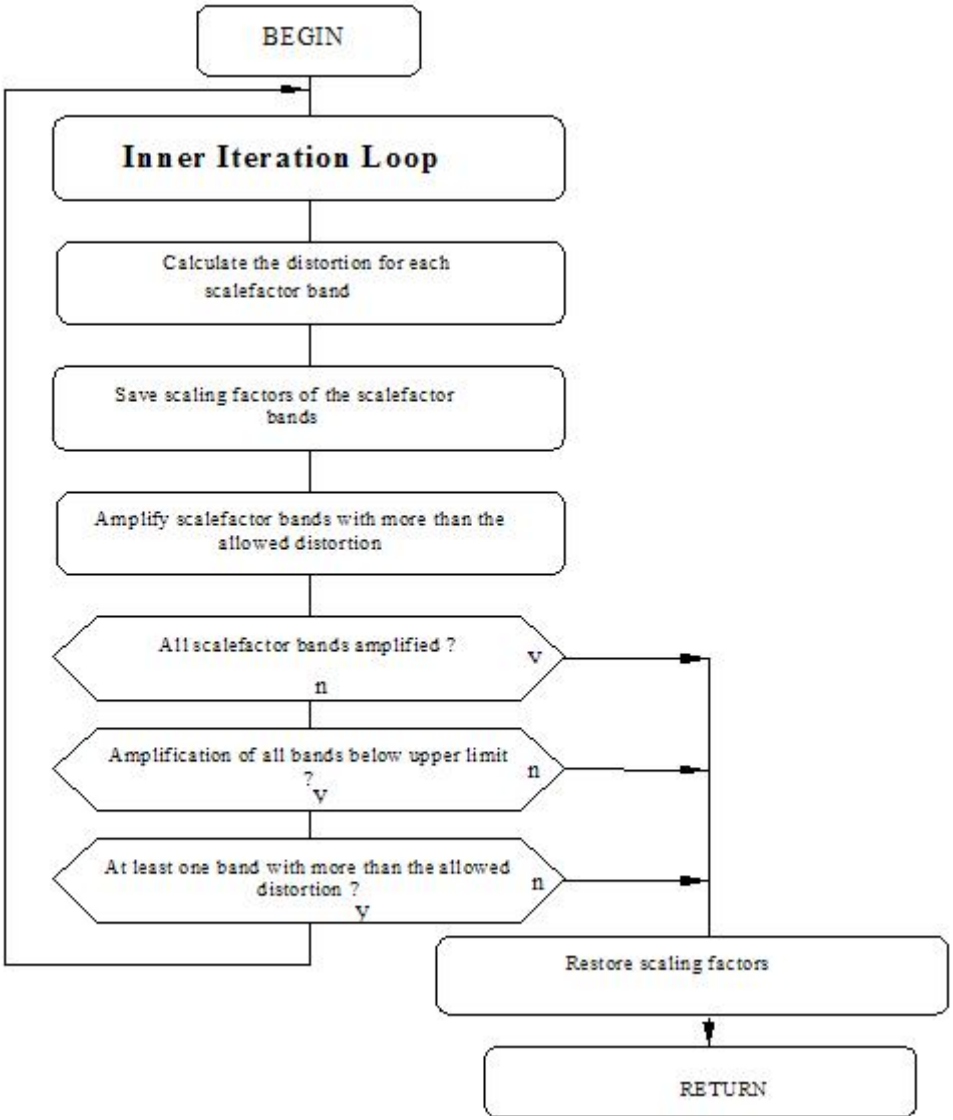


그림 9 Outer iteration loop

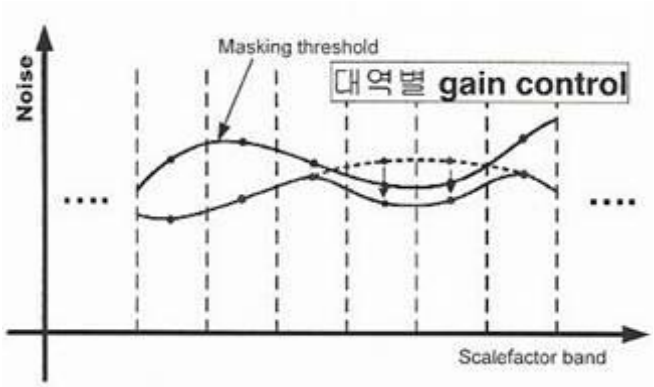


그림10 Bandwidth gain control of outer loop

양자화 과정은 다음 세 가지 조건 중 하나를 만족하면 중단된다.
scalefactor band 모두 허용되는 noise를 넘지 않는 경우
다음 iteration이 scalefactor band 중 어떤 band에서 최대 허용되는 값을 넘기게 만드는 경우
다음 iteration에서 모든 scalefactor band의 증대가 발생하는 경우

· 무손실 (Lossless Coding)

무손실 압축은 MP3와 다르게 12개의 허프만 코드북을 이용하여 scale factor band 단위로 압축을 하며 scalefactor값 역시 차분 허프만 코딩으로 압축한다. 또한 비트 사용을 줄이기 위해서 sectioning, 그룹화 및 인터리빙 등을 사용한다. 무손실 압축을 하기 위해 short window를 사용한 프레임에 대해서 그룹화 및 인터리빙 과정이 필요하다. 그림 11은 그룹화 및 인터리빙 과정을 표현하고 있다.

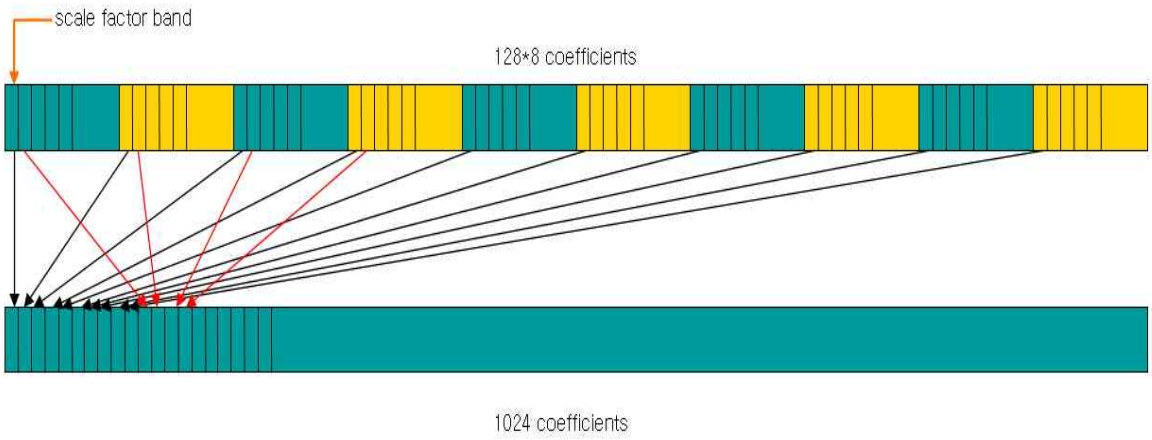


그림 11 Grouping and interleaving

그림 11 에서 보는 것과 같이 일반적으로 short window를 scalefactor band 단위로 interleaving해주게 될 경우 고주파에 0값을 가지는 scalefactor band들이 한쪽으로 모이게 되고, 이러한 효과는 압축효율을 높인다. 이 과정을 마친 후 각 scalefactor band에 대해 허프만 코딩을 한다. AAC에서 허프만 코딩은 모두 12개의 코드북을 이용하여 압축된다. 그리고 scale factor 중 global gain은 8비트 양수로 압축하고, scalefactor는 차분 허프만 코딩을 이용하여 압축한다.
다음의 표는 AAC에서 사용하는 허프만 코드북의 특징을 나타낸 것이다.

Codebook index	n-Tuple size	Maximum absolute value	Sign values
0		0	
1	4	1	yes
2	4	1	yes
3	4	2	no
4	4	2	no
5	2	4	yes
6	2	4	yes
7	2	7	no1
8	2	7	no
9	2	12	no
10	2	12	no
11	2	16 (ESC)	no

표 2 Huffman codebook

각 scalefactor band에 대하여 허프만 코딩 작업을 마친 후 sectioning 과정을 한다. Sectioning은 근접 scalefactor band에 대해 하나의 허프만 코드북을 가지고 압축하였을 때 비트 수가 scalefactor band단위로 압축한 비트 수의 합보다 많다면 하나의 코드북으로 압축하는 과정을 말한다. Sectioning의 과정을 pseudo-code로 살펴보면 다음과 같다.

```
for(all sfb)
{
If count_bit[huff1]+count_bit[huff2] > count_bit[huff1,2]
        section1,2 are merged
}
for(all sfb)
{
If count_bit[huff1,2]+count_bit[huff3,4] > count_bit[huff1,2,3,4]
        section2,3,4 are merged
}
...
```

– 패킷 손실 은닉 알고리즘

프레임 삭제 은닉 알고리즘으로도 알려져 있는 패킷 손실 은닉 (PLC: Packet Loss Concealment) 알고리즘은 오디오 시스템의 전송 손실을 숨긴다. 여기에서의 오디오 시스템은 송신단에서 입력신호를 부호화하고 패킷화하여 네트워크로 전송하고, 반대로 수신단에서는 수신된 패킷을 복호화하여 출력신호를 재생하는 시스템을 말한다. G.723.1, G.728, G.729 와 같은 CELP (Code Excited Linear Prediction) 기반의 표준 코덱들은 대부분이 표준안에 수록된 PLC 알고리즘들을 가지고 있다.
PLC 알고리즘의 목표는 수신된 비트 스트림 내의 사라진 데이터를 은닉하기 위해 합성된 음성 신호를 만들어내는 것이다. 이상적으로, 합성된 신호는 사라진 신호와 동일한 음색과 스펙트럼 특성들을 가지고, 부자연스러운 인공효과를 발생하지 않는다. 음성 신호는 종종 국부적으로 정상적(stationary) 이기 때문에, 적당한 근사치를 생성하기 위해서는 지나간 히스토리(history)를 사용하는 것이 가능하다. 만일 삭제 구간이 너무 길지 않고 또한 신호가 급격하게 바뀌는 영역에 들어가지 않는다면, 삭제 구간은 은닉 후에 인식되지 않게 된다.

G.711 Appendix I 에서 제안하는 패킷 손실 보상 알고리즘은 히스토리 버퍼 (history buffer) 에 저장되어 있는 48.75 msec 의 이전 음성 신호로부터 피치(pitch) 구간을 계산한 후 계산된 피치 구간을 반복해서 삽입하는 피치구간삽입 방식이다. 패킷 손실 보상을 위해 복호화된 출력 신호는 히스토로 버퍼에 저장되며 히스토리 버퍼는 차후 손실된 구간에서의 피치를 계산하고 손실구간 동안의 신호를 추출하는데 사용된다. 만약 패킷 손실이 발생하면 히스토리 버퍼의 내용을 피치 버퍼로 복사한 후 피치 버퍼의 음성으로부터 피치를 계산한다. 부드러운 음성 연결을 위해 계산된 피치의 1/4 지점을 삼각 윈도우를 이용하여 OLA (OverLap and Add) 를 사용하여 피치 구간을 반복하여 삽입한다. 그림12는 연속적인 손실에 대한 G.711 패킷 손실 은닉 알고리즘의 수행 동작을 보여준다.

손실을 은닉하지 않는 G.711 시스템에 PLC 알고리즘을 추가하기 위해서는 오직 수신단의 변화만이 요구된다. G.711 권고안에서는 G.711로 인코딩되는 오디오 데이터는 8 kHz 로 샘플링되고 10 ms 프레임 (80 샘플)들로 분할된다고 가정한다. 약간의 파라미터들을 조절함으로써, 패킷 크기 또는 샘플링율은 조절될 수 있다.

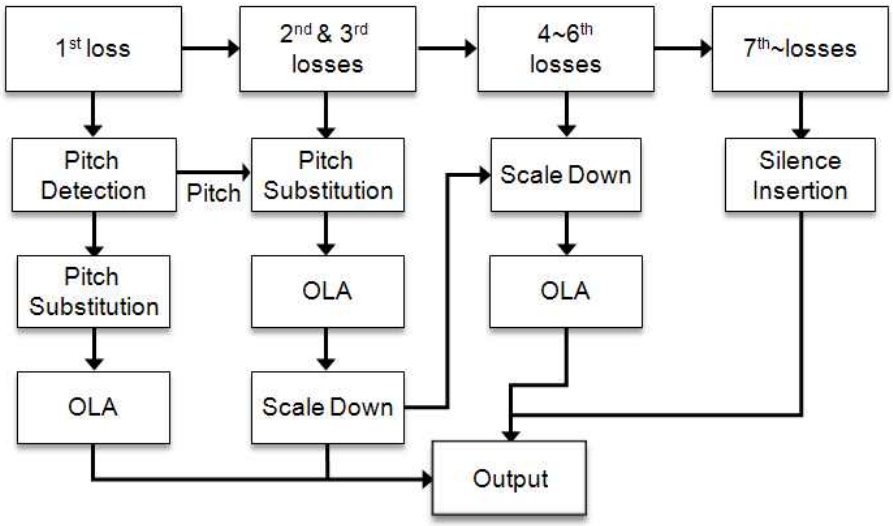


그림 12 Frame erasure concealment algorithm for G.711

· 완전한 프레임 (Good Frames)

일반적인 동작 동안에 수신단은 수신 패킷을 복호하고 오디오 포트로 출력을 전송한다. PLC를 지원하기 위해서는 완전한 프레임을 처리하기 위한 두 가지의 작은 변화가 수신단에서 이루어진다.

복호된 출력의 복사본은 48.75 ms (390 샘플) 길이의 원형 히스토리 버퍼 저장된다. 히스토리 버퍼는 현재의 피치 주기를 계산하고 삭제된 파형을 추출하기 위해 사용된다.

출력은 오디오 포트로 보내지기 전에 3.75 ms (30 samples) 만큼 지연된다. 이와 같은 알고 리즘 지연은 삭제구간 시작점에서의 OLA 에 사용되며, PLC 알고리즘 코드가 실제 신호와 합성 신호 사이의 부드러운 천이 (transition)를 수행할 수 있도록 해준다.

· 불완전한 첫 번째 프레임 (First Bad Frame)

삭제구간의 시작점에서, 원형 히스토리 버퍼는 피치 버퍼라고 불리는 사용하기에 편리한 비원형 버퍼로 복사된다. 피치 버퍼의 데이터는 삭제 구간을 위해 사용된다. 삭제 구간이 10 ms 보다 긴 경우에는, lastq 버퍼라고 불리는 최신의 1/4 피치 주기의 추가적인 복사가 수행된다.

· 피치 검파 (Pitch Detection)

먼저, 피치 주기는 히스토리 버퍼에 있는 최신의 10 ms 음성과 5ms (40 샘플) 에서 15 ms (120 샘플) 까지 탭(taps)에서의 이전 음성에 대해 정규화 교차 상관 (normalized cross-correlation) 의 peak 를 찾음으로써 예측된다. 이에 상응하는 주파수는 200 에서 66 Hz 이다. 피치 범위는 G.728의 post-filter 에서 사용된 범위에 기반하여 선택된다. G.728이 2.5 ms (20 샘플)의 낮은 경계를 사용하는 반면에, 여기에서는 단 하나의 10 ms 삭제 프레임에서 동일한 피치 주기가 두 번 이상 반복되지 않도록 하기 위해 20 샘플로 증가시킨다. 복잡도를 낮추기 위해, 피치 예측은 두 단계로 계산된다. 첫째, 2:1 로 decimation 된 신호에서 대략적인 검색 (coarse search) 이 수행되고 나서, 대략적인 검색의 peak 근처에서 정밀 검색 (finer search) 가 수행된다. 정밀 검색의 생략은 약간의 음질을 저하시

키면서 복잡도를 낮출 수 있다. wavelength 는 이러한 계산의 출력 값을 나타내는데 사용된다.
WSOLA (Waveform Shift Overlap Add) 로부터, 정규화 교차 상관 함수는 비정규화 교차 상관 또는 교차 평균 크기차 (AMDF: cross-Average Magnitude Difference Function) 으로 대체될 수 있으며 전체 성능 결과가 비슷하게 얻어지는 것으로 알려져 있다.

· 초기 10 ms 합성신호 생성 (Synthetic Signal Generation for First 10 ms)

삭제구간 처음의 10 ms 에 대해, 최상의 결과는 감쇠가 없는 마지막 피치 주기로부터 합성신호를 생성함으로써 얻어진다. 처음의 10 ms 동안에는 피치 버퍼의 오직 최신 1.25 피치 주기만이 사용된다. 실제 신호와 합성 신호 간에 부드러운 천이를 보장하기 위해, 또한 피치 주기가 여러 회 반복될 때에 부드러운 천이를 보장하기 위해, 마지막 피치 주기와 그와 인접한 피치 주기 사이의 1/4 피치 주기 상에 삼각 윈도우를 사용하는 OLA 가 수행된다. 1/4 파형길이에 대해서는, 피치 버퍼 끝으로부터 1.25 피치 주기 지점에서 시작되는 신호는 상승 기울기 (up-sloping ramp) 에 의해 곱해지고, 이는 하강 기울기 (down-sloping ramp) 에 의해 곱해져서 lastq 버퍼의 마지막 0.25 피치 주기에 더해진다. 복잡도가 주요 이슈가 아닌 경우에는 모든 OLA 동작에서의 삼각 윈도우는 해밍 윈도우로 대치한다.

OLA 결과는 피치 버퍼의 끝과 히스토리 버퍼의 끝단(tail)을 모두 대체한다. 또한 이는 마지막 완전한 프레임의 끝 부분에 대한 원래의 신호를 대체하는 수신단 출력이 된다. 이는 다음 프레임의 삭제 여부를 알게 될 때까지 마지막 프레임의 끝 부분 출력을 하지 않음으로써 알고리즘 지연을 발생한다. 삭제가 생기면, 마지막 완전한 프레임의 끝 부분에 대한 신호는 합성된 신호로의 부드러운 천이를 위해 OLA 를 통하여 수정된다.
삭제된 동안의 10 ms를 위한 합성 신호는 피치 버퍼 끝에서 한 피치 주기만큼 포인터를 옮긴 샘플을 복사하여 출력함으로써 생성된다. 피치 주기가 10 ms 보다 짧은 경우, 즉 포인터가 피치 버퍼 끝을 벗어나는 경우에는 포인터를 정확히 한 피치 주기 전으로 설정한다. 피치 주기가 짧은 경우 (주파수가 높은 경우) 에 피치 버퍼의 마지막 피치 주기는 10 ms 삭제 구간 동안 여러 회 반복된다.
삭제가 일어나는 동안에 히스토리 버퍼는 합성된 출력에 의해 갱신된다. 이러한 방법으로 히스토리 버퍼는 항상 부드럽고 연속되는 신호를 가지게 된다. 이러한 연속성은 "불완전 프레임 (bad frame), 완전한 프레임 (good frame), 불완전 프레임 (bad frame)" 시퀀스가 발생할 때에 중요하다.

· 10 ms 이후의 합성 신호 생성 (Synthetic Signal Generation after 10 ms)

다음 프레임이 동시에 삭제되는 경우, 삭제구간은 최소 20 ms 길이가 되어 추가적인 처리가 필요하다. 짧은 삭제 구간에 대해 하나의 피치 주기를 반복하는 것이 좋은 효과를 내는 것과는 달리, 긴 삭제 구간에서는 부자연스러운 하모닉 인공효과 (beeps)를 야기한다. 특히 이러한 현상은 삭제구간이 무성음 영역에 들어가는 경우 또는 정지 상태와 같이 급격한 천이가 생기는 영역에 들어가는 경우에 확연해진다. 이러한 인공효과는 삭제구간 처리로써 신호를 합성하는데 사용되는 피치 주기 수를 늘림으로써 확연하게 감소된다는 것이 실험을 통해 입증되었다. 더 많은 피치 주기를 재생하는 것은 신호의 변이를 증가시킨다. 비록 피치 주기가 원 신호에서 발생된 순서대로 재생되지 않는다 해도,

출력은 결과적으로 자연스럽게 된다. 삭제 구간으로 진입하는 10 ms 에서는 신호를 합성하는데 사용되는 피치 주기 수는 2개로 증가하며, 20 ms 에서는 세 개의 피치 주기가 추가된다. 20 ms 이상의 삭제에 대해서는 피치 버퍼의 추가적인 수정이 발생하지 않는다.
피치 버퍼에서 사용되는 피치 주기의 개수가 증가할 때, 합성된 신호에서의 부드러운 천이가 중요하다. 이는 현존하는 두 번째와 세 번째 삭제 프레임의 시작지점 1/4 피치 주기에 대해 피치 버퍼의 출력을 계속하고, 피치 버퍼를 갱신하고, 버퍼 포인터를 정확한 위상에 동기화 시키고, 그러고 나서는 새로운 피치 버퍼로부터의 출력으로 OLA 를 수행함으로써 얻어진다.
피치 버퍼는 피치 주기의 개수가 증가하는 경우를 제외하고, 처음 삭제 프레임 동안에 정확히 갱신된다. 예를 들면, 두 번째 삭제 프레임의 시작에서, 1/4 파형길이에 대해 피치 버퍼 종단으로부터 2.25 피치 주기에서 시작되는 신호는 상승 기울기에 의해 곱해지고, 이는 하강 기울기에 의해 곱해진 lastq 버퍼에 있는 1/4 파형길이에 더해진다. OLA 결과는 피치 버퍼에 있는 마지막 1/4 파형길이를 대체한다. 현재 출력 포인터의 위상을 유지하기 위해 피치 주기는 사용된 첫 번째 피치 주기에 있을 때까지 포인터로부터 추출된다.

· 감쇄 (Attenuation)

G.729와 G.728 Annex I 와 같은 다른 PLC 알고리즘과 같이, 긴 삭제에 대해서는 신호를 감쇄시키는 것이 필요하다. 삭제가 길수록 합성된 신호는 실제 신호로부터 보다 더 발산된다. 감쇄 시키지 않으면 합성된 신호 세그먼트가 독립적으로 자연스럽게 들리는 경우라 하더라도 어떠한 타입의 소리를 지나치게 길게 유지함으로써 이상한 인공효과가 생긴다. 두 번째 10 ms 의 시작점에서, 합성된 신호는 10 ms 당 20% 의 비율을 가지는 기울기로 선형적으로 감쇄된다. 60 ms 후에는 합성신호는 0이 된다.

· 삭제 후 첫 번째의 완전한 프레임 (First Good Frame after an Erasure)

삭제 구간 이후의 첫 번째 굿 프레임에서, 합성된 삭제음성과 실제 신호 간에 부드러운 천이가 필요하다. 이를 수행하기 위해, 피치 버퍼로부터의 합성된 음성은 삭제 구간이 지난 후에 계속되며, 이 후 OLA 를 사용하여 실제 신호와 혼합된다. OLA 길이는 피치 주기와 삭제 길이 모두에 의존한다. 10 ms 의 짧은 삭제인 경우에는 1/4 파형길이의 윈도우가 사용된다. 긴 삭제 구간에 대해서는 최대 프레임 크기 (10 ms) 까지 삭제구간 10 ms 마다 4 ms 만큼 윈도우를 증가시킨다.

– iLBC 음성 코딩 알고리즘

iLBC (Internet Low Bit-rate Codec) 는 2003년 IETF에 의해 표준화된 음성 코덱이다. iLBC는 VoIP (Voice over IP) 서비스에 적합하며 저작권 무료인 음성 코덱으로 현재 유럽의 VoIP의 음성 코덱으로 사용되고 있으며 그 활용 범위가 점차 넓어지고 있다. 특히 iLBC는 기존 음성 코덱과 비교하여 패킷 손실에 강인하게 만들어졌기 때문에 패킷 손실이 큰 이동 전화 및 네트워크 환경에서 활용 가능성이 높아지고 있다. iLBC는 블록 독립적인 LPC (Block-independent LPC) 기반의 음성 코덱

이다. 기본적으로 20 ms, 30 ms 두 가지의 프레임 길이를 제공하고 입력신호는 샘플링 주파수가 8 kHz 인 신호를 사용한다. iLBC는 20ms 일 때 15.2 kbit/s, 30ms 일 때 13.3 kbit/s 의 전송률을 가지도록 설계되었다.
전체적인 iLBC의 부호화 알고리즘을 정리하면 다음 그림13과 같다.

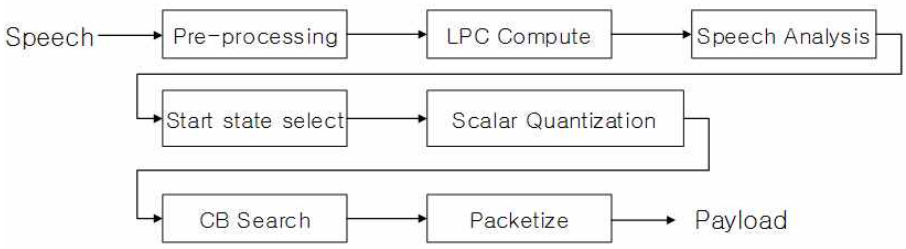


그림 13 Encoding algorithm of iLBC

우선 전처리 과정으로 고역통과필터 (high pass filter) 과정을 거친다. 그 후 LPC (Linear Prediction Coefficient) 를 계산한 다음 계산된 LPC를 이용하여 음성의 여기신호 (residual signal)를 구한다. 한 개의 프레임은 5ms에 해당하는 4개의 서브블록으로 나뉜다. 각각의 서브블록에 대한 여기신호를 구한 후 에너지가 큰 연속된 2개의 서브블록인 시작상태 (start state) 을 선택한다. 이렇게 선택된 2개의 서브블록 중 57 샘플에 대해 양자화 한다. 양자화 과정 후 시작상태 블록에서 양자화하지 않은 23개의 샘플과 나머지 2 서브블록에 대해 적응코드북 (ACB: Adaptive CodeBook) 방식을 적용하여 코드북 인덱스와 이득을 구하고 이들을 양자화 한다. 그 후 원 신호와 부호화한 23개의 샘플 및 나머지 2 서브블록의 샘플들을 복원하여 그 차이를 구한 다음, 차이에 대한 코드북 인덱스와 이득을 구하는 과정을 거치게 된다. 총 3번에 걸쳐 코드북 인덱스와 이득을 구한 후 양자화 된 모든 데이터를 패킷화 하는 과정으로 iLBC 부호화 과정이 끝나게 된다.

· 전처리 과정

음성 코덱은 입력된 신호에서 50 Hz 이하의 잡음과 DC 성분을 제거하기 위해 전처리 과정을 거치게 된다. 이 과정을 위해 iLBC 에서는 90 Hz의 차단주파수를 가지는 다음 식과 같은 고역통과필터를 사용한다.

$$H(z)=\frac{0.92727436-1.8544941z^{-1}+0.92727436z^{-2}}{1+1.9059436\,z^{-1}-0.9114024z^{-2}}$$

· LPC 분석, 양자화 및 여기신호 생성

전처리된 신호를 이용하여 LPC를 구하는 과정으로 현재 입력된 샘플 160개와 이전 프레임에서 사용된 80개의 샘플들을 합하여 총 240개 샘플들에 대해 LPC 분석을 실시한다. 이전 프레임이 없는 첫 프레임의 경우 이전 프레임의 샘플을 0으로 가정하고 LPC 분석을 한다. LPC 분석을 하기 위한 첫 단계로 다음 그림14와 같은 윈도우를 적용한다.

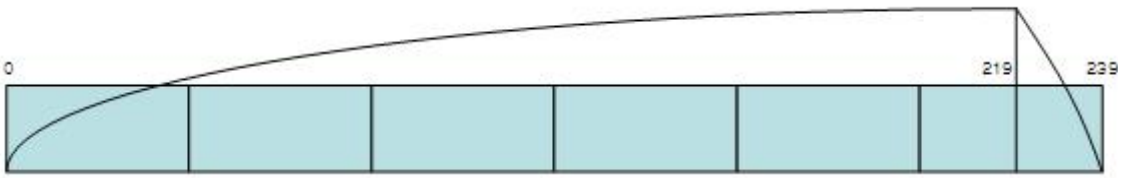


그림 14 Window for LPC analysis

그림 14의 윈도우는 0번 샘플부터 219번 샘플까지는 Hamming 윈도우와 유사한 윈도우를 적용하고, 220번 샘플부터 239번 샘플까지는 cosine 윈도우를 적용한다. 윈도우는 다음 식과 같이 정의된다.

$$w(i)=\begin{cases}(\sin(\frac{\pi(i+1)}{441}))^2 \ , & i=0,\ldots,219\\ \cos((i-220)\frac{\pi}{40}) \ , & i=220,\ldots,239\end{cases}$$

이렇게 얻어진 신호를 이용하여 자기상관계수 (autocorrelation sequence) 를 구하고 자기상관계수는 Levinson-Durbin 알고리즘을 통해 10차 LPC로 변환된다. 이렇게 구해진 LPC는 또다시 양자화에 적합한 LSF (Line Spectrum Frequency) 로 변환한다. 이때 변환된 LSF를 $\omega_{crrent\_frame}(i), 1<i\le 10$ 라 표시한다. 변환된 LSF의 1차에서 3차는 각각 6 bits, 4차에서 6차는 각각 6 bits, 7차에서 10차는 각각 7 bits 스칼라 양자화기를 통해 양자화 된다. 이어서 역양자화 과정 (inverse quantization) 을 통해 얻어진 양자화 된 LSF, $\tilde{\omega}_{crrent\_frame}(i), 1<i\le 10$, 는 다음 식을 만족하는지를 확인하여 안정도 (stability) 조건을 확인한다.

$$\tilde{\omega}_{crrent\_frame}(i-1) < \tilde{\omega}_{current\_frame}(i) \ , \quad 0<i\le 10$$

여기서 $\tilde{\omega}_{crrent_frame}(i)$는 현재 프레임에서 i 번째 양자화 된 LSF 계수이다. 안정성을 확인된 LSF를다음 식과 같이 이전 프레임의 LSF와 보간 (interpolation)을 통해 서브블록에 해당하는 LSF를 계산한다.

$$\tilde{\omega}_{subblock(n)}(i)=\frac{4-n}{4}\tilde{\omega}_{old\_frame}(i)+\frac{n}{4}\tilde{\omega}_{current\_frame}(i), n=1,2,3,4 \ and \ i=1,\cdots,10$$

여기서 n은 서브블록의 번호이고, $\tilde{\omega}_{old_frame}(i)$는 이전 프레임의 i 번째 LSF 계수, $\tilde{\omega}_{crrent_frame}(i)$는 현재 프레임의 i 번째 LSF, $\tilde{\omega}_{subblock(n)}(i)$은 n번째 서브블록의 i 번째 LSF 계수를 나타낸다. 양자화 되지 않은 LSF에 대해서도 위의 식을 적용하여 각 서브블록의 LSF, $\omega_{subblock(n)}(i), n=1,2,3,4 \text{ and } i=1,\ldots,10$, 를 얻는다.

위와 같이 계산된 LSF를 이용하여 각 서브블록에 대해 양자화 된 LPC 필터와 양자화하지 않은 LPC 필터를 위와 아래의 식이 각각 다음과 같이 생성된다.

$$A_n(z)=1+\sum_{i=1}^{10}a_n(i)z^{-i}$$

$$\widetilde{A}_n(z) = 1 + \sum_{i=1}^{10} \widetilde{a}_n(i) z^{-i}$$

여기서 $A_n(z)$는 양자화하지 않은 n 번째 서브블록의 LPC 필터이고, $\widetilde{A}_n(z)$는 양자화된 n 번째 서브블록의 LPC 필터, $a_n(i)$는 양자화하지 않은 n 번째 서브블록의 I 번째 LPC 계수, $\widetilde{a_n}(i)$는 양자화된 n 번째 서브블록의 i 번째 LPC 계수를 나타낸다. 각 서브블록 샘플에 해당하는 두 가지 여기신호를 계산한다.

· 시작상태 검색

위의 LPC 분석, 양자화 및 여기신호 생성 과정을 통해 얻은 여기신호를 이용하여 시작상태의 위치를 검색한다. 시작상태는 여기신호의 에너지가 큰 2개의 연속된 서브블록으로 선택한다. 시작상태를 찾기 위해 우선 각 서브블록의 여기신호에 다음 그림15와 같은 윈도우를 적용한다.

그림 15 Window for the start state search

그림 15에서와 같이 윈도우는 0번 샘플부터 4번 샘플은 각각 {1/6, 2/6, 3/6, 4/6, 5/6}을, 35번 샘플부터 39번 샘플은 각각 {5/6, 4/6, 3/6, 2/6, 1/6}을 적용하고 나머지 5번 샘플부터 34번 샘플까지는 모두1의 값을 적용한다. 이렇게 윈도우를 이용하여 다음과 같은 식으로 각 서브블록의 여기신호에 대한 에너지를 계산한다.

$$E(n) = \sum_{i=0}^{39} W_s(i)\, e_n^2(i), n = 1,2,3,4$$

여기서 $W_s(i)$는 <Figure 5-3>에서 같이 i 번째 샘플의 윈도우 값을 나타내고, $e_n(i)$는 n번째 서브블록의 i 번째 여기신호, $E(n)$은 n 번째 서브블록의 에너지를 나타낸다.

식 (5-7)에서 구한 $E(n)$을 이용하여 에너지가 큰 2개의 연속된 서브블록을 나타내는 시작상태를 선택하고 시작상태의 위치는 다음과 같이 2 bits로 부호화된다.

Start State = 1: 시작 상태가 서브블록 1, 2로 구성될 때

Start State = 2: 시작 상태가 서브블록 2, 3로 구성될 때

Start State = 3: 시작 상태가 서브블록 3, 4로 구성될 때

· 시작상태 중 57 샘플 선택

시작상태를 결정한 후, 시작상태에 해당하는 80개의 샘플의 여기신호에 대해 양자화를 한다. 이때 모든 샘플을 양자화 하는 것 대신에 압축효율을 높이기 위해서 57개의 샘플을 선택한다. 이를 선택하는

방법은 시작상태로 선택된 80개의 샘플 중 앞에서부터 57개 샘플의 에너지를 구한 것과 뒤에서부터 57개 샘플의 에너지를 구한 것을 비교하여 에너지가 큰 57개의 샘플을 선택하고 다음과 같이 1bit을 이용하여 어느 것을 선택 했는지를 부호화한다.

Start_first = 1: 앞에서부터 57개 샘플을 선택

Start_first = 0: 뒤에서부터 57개 샘플을 선택

이렇게 선택된 57개의 샘플은 스칼라 양자화기를 통해 부호화되며 나머지 23개의 샘플은 나머지 서브블록과 함께 ACB를 통해 부호화된다.

· 샘플 정규화

57개의 샘플을 양자화하기 전에, 57개의 샘플의 크기를 표준화한다. 이를 위해 스케일링(scaling) 하는 과정이 필요하다. 먼저 57개의 샘플 중 가장 큰 값을 선택하여 다음식과 같이 변환하여 6-bits 양자화기를 통해 부호화하고 양자화 된 스케일 값으로 57개의 샘플을 정규화 한다.

$$scale = \log_{10}(Max\_value)$$

여기서 Max_value는 57개의 샘플 중 가장 큰 값을, scale은 계산된 스케일 값을 나타낸다.

· 샘플 양자화

정규화된 여기신호 57 샘플은 다음 <Figure 5-4>과 같이 3-bit DPCM 양자화기 방식으로 양자화 된다.

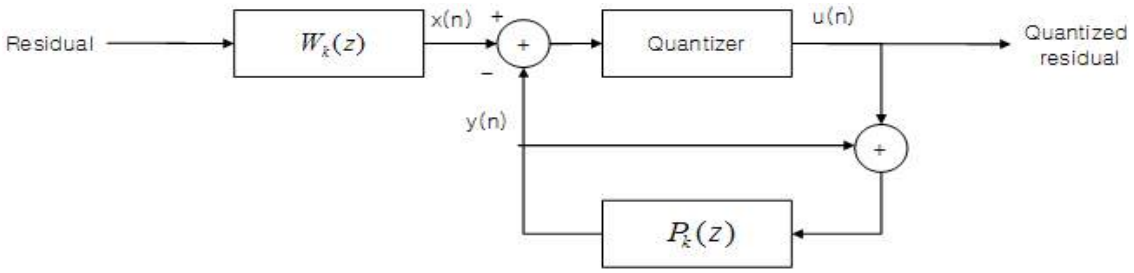


그림16 DPCM 3-bits quantizer for 57 sample

여기서 $W_n(z)$는 n번째 서브블록의 심리 가중 필터 (perceptual weighting filter) 이고, 다음 식과 같다.

$$W_n(z) = \frac{1}{A_n(\frac{z}{\lambda})}$$

여기서 $A_n(z)$는 n번째 서브블록의 양자화하지 않은 LPC 필터를 나타내고, λ는 0.4222값을 가진다. 그리고 $P_n(z)$는 n번째 서브블록의 예측 필터 (prediction filter) 로 다음 식과 같다.

$$P_n(z) = 1 - \frac{1}{W_n(z)} \quad .$$

· 코드북 메모리 생성

57개의 샘플을 부호화된 두개의 서브블록의 나머지 부분인 23개의 샘플과 남은 두개의 서브블록에 대한 부호화 과정을 거친다. 이를 위해서는 57개의 샘플과 같은 양자화방식이 아닌 코드북을 이용하여 부호화한다. 부호화는 23개의 샘플을 먼저 부호화하고 시작상태보다 시간상 뒤에 있는 서브블록, 시작상태보다 시간상 앞에 있는 서브블록의 순서로 부호화한다. 이 부호화에 사용되는 코드북은 57개의 샘플 및 부호화 대상 샘플보다 이전에 부호화된 샘플로 생성된다. 코드북을 생성하기 위해 만든 코드북 메모리의 길이는 부호화하는 대상에 따라 다음과 같이 두가지로 나뉜다.

85 샘플 : 잔여샘플 (23개의 샘플)을 위한 코드북 메모리의 길이

147 샘플 : 40개의 샘플을 위한 코드북 메모리의 길이

코드북 메모리를 구성하는 방법을 예를 들어 설명하면 다음과 같다. 시작 상태가 2, 3 번째 서브블록이고, 뒤에서부터 57개의 샘플을 부호화하였다고 가정하면 부호화 순서는 그림 17과 같다.

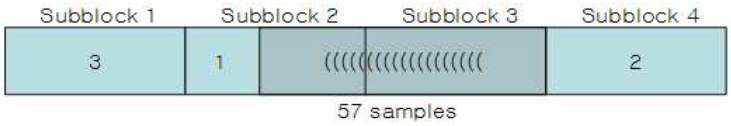


그림 17 Codebook memory generation order

양자화하려는 여기신호의 샘플이 시작상태보다 시간적으로 이전에 있을 경우, 즉 Start_first=0 인 경우, 이미 양자화된 57개의 샘플에 28개의 zero를 채운 샘플을 합해서 전체 85개의 샘플들을 배열한다. 코드북 메모리로부터 이후 8번 코드북 생성 과정을 통해 23개의 샘플을 양자화 한다. 두 번째로 4번째 서브블록 양자화 해야 한다. 이때 147개의 코드북 메모리가 필요하다. 시작상태에 속한 80개의 양자화 된 샘플과 67개의 zero를 채운 샘플을 합해서 전체 147개의 샘플들을 배열한다. 그리고 상기에서와 같은 방법으로 4번째 서브블록의 40개의 샘플을 모두 양자화 한다.

마지막으로 첫 번째 서브블록의 40개의 샘플을 양자화하기 위해 147개의 코드북 메모리를 생성한다. 지금까지 양자화 된 120개의 샘플 (2,3,4번째 서브블록의 샘플) 과 27개의 zero를 채운 샘플을 합해서 전체 147개의 샘플들을 배열한다. 그리고 40개의 샘플에 대해 양자화를 한다.

· 코드북 생성

위와 같이 생성된 코드북 메모리를 이용하여 양자화 하고자 하는 샘플에 해당하는 코드북을 다음 그림 18과 같이 생성한다. 우선, 시작상태에 있는 23개의 잔여샘플을 양자화하기 위해 코드북 메모리의 마지막 샘플부터 23개의 샘플을 사용하여 첫번째 코드워드 (codeword)를 생성한다. 그리고 두 번째 코드워드는 마지막에서 두 번째 샘플부터 23개를 선택하여 생성된다. 이와 같은 방식으로 마지막

64번째 코드워드는 앞에서 23번째 샘플부터 0번째 샘플을 이용하여 생성되며 잔여 샘플에 대한 코드북 생성을 완료한다. 다른 서브블록 내에 있는 샘플들을 양자화하기 위해서는 코드북 메모리의 마지막 샘플부터 40개씩을 사용하여 코드북을 생성한다. 그 외의 과정은 잔여샘플의 코드북 생성 과정과 동일하다.

상기와 같이 코드북 메모리를 이용하여 생성하는 코드북을 기본 코드북이라 하며 기본 코드북은 잔여샘플 64개, 서브블록샘플 108개의 코드워드를 가진다.

그림 18 Codebook generation

· 코드북 확장

8번 코드북 생성 과정에서 생성한 기본 코드북으로 샘플을 부호화하기에 어려움이 많으므로 코드북 확장 과정을 통해 최종적으로 잔여샘플을 양자화하기 위해서는 코드북 생성 과정에서 만든 64개의 코드워드의 코드북을 128개의 코드워드를 갖는 코드북으로, 서브블록을 위해서는 108개의 코드워드에 20개의 코드워드를 더해서 128개로 만든 후 256개의 코드워드를 갖는 코드북으로 확장한다. 그리고 이를 코드북 확장이라 부른다. 코드북 확장에 사용되는 방법은 코드북 증강 (codebook augmentation), 코드북 확장 (codebook expansion) 으로 나뉜다. 코드북 증강에 의한 확장된 코드북은 서브블록의 양자화를 위한 코드북에 적용되는 방법으로 20개의 코드워드가 더해진다.

그림19는 코드북 증강 방법에 의해 코드워드를 한개 늘리는 방법을 보여 주고 있다. 코드북 생성 과정에 얻어진 147개의 샘플로 구성된 코드북 메모리 $(c_0, ..., c_{146})$ 로 부터 우선 20개의 샘플을 가지고 온다. $(c_0, ..., c_{19})$ 그리고 처음 5개의 샘플 $(c_0, ..., c_4)$ 과 21번째에서 25번째의 샘플 $(c_{20}, ..., c_{24})$을 선형보간을 한다. 이렇게 선형보간된 샘플, $(\overline{c_{20}}, ..., \overline{c_{24}})$ 을 증강된 코드워드의 21번째에서 25번째의 원소로 한다. 그리고 $(c_{25}, ..., c_{29})$의 15개의 샘플을 가지고 와서 증강된 코드워드를 다음과 같이 구성한다.

$(c_0, ..., c_{19}, \overline{c_{20}}, ..., \overline{c_{24}}, c_{25}, ..., c_{39})$

두 번째로 증강된 코드워드는 $(c_0, ..., c_{19}, \overline{c_{20}}, ..., \overline{c_{24}}, c_{25}, ..., c_{39})$이며 여기서 $(\overline{c_{19}}, ..., \overline{c_{23}})$은 $(c_0, ..., c_4)$와 $(c_{19}, ..., c_{23})$을 선형보간하여 만든다. 이러한 방법을 반복하여 최종적으로 20개의 코드워드를 증강한다.

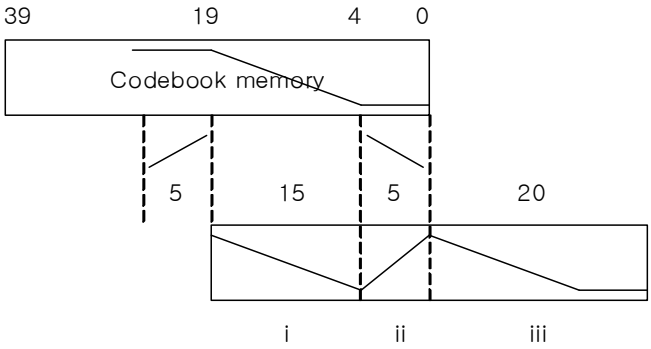


그림 19 Codebook increasing method - 1st codebook

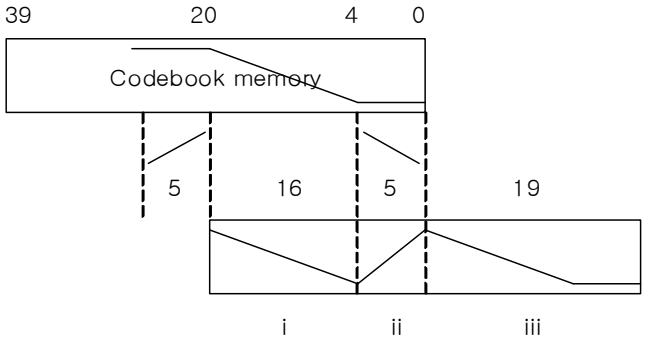


그림 20 Codebook increasing method - 2nd codebook

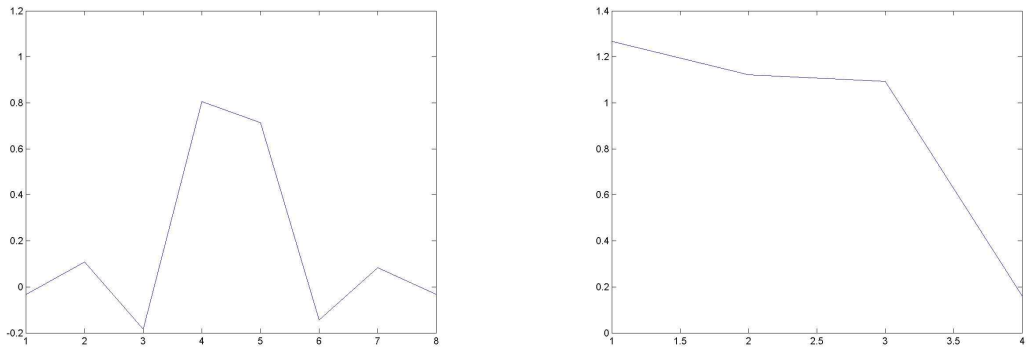
코드북 확장은 코드북 증강을 포함하여 생성된 코드북에 다음 식과 같은 필터를 이용하여 생성한다.

$$exp\_cb(k) = \sum_{i=0}^{7} cbfiltersTbl(i)\ base\_cb(k-i+4)$$

여기서 $base_cb(k)$는 k번째 코드워드를, $cbfiltersTbl(i)$는 i번째 필터 계수를 나타내고 $exp_cb(k)$는 k번째 확장된 코드워드를 나타낸다. 코드북 확장에 사용되는 필터의 계수는 다음과 같으며 <Figure 5-12>와 같은 특성을 가진다.

cbfiltersTbl[8] = { -0.033691, 0.083740, -0.144043, 0.713379, 0.806152, -0.184326, 0.108887, -0.034180};

이를 통하여서 최종적으로 잔여샘플을 위해 128개의 코드북, 서브블록의 샘플을 위해256개의 코드북이 생성된다.



< Time domain > < Frequency domain >

그림 21 Characteristic of codebook expansion filter

· 코드북 검색 및 이득 계산

본 과정에서는 상기의 과정을 통해 제작된 코드북을 검색하여 여기신호에 가장 적합한 코드워드의 인덱스를 찾는다. 검색방법은 다단계 (multi-stage) 검색이며 인덱스 검색과 함께 가장 적합한 이득 또한 계산해 낸다. 검색의 단계는 총 3단계로 구성되며 각 단계별로 가장 적합한 코드워드와 이득을 찾는다. 우선 첫 번째 단계에 대해 다음의 과정을 수행한다.

첫째, 양자화하려는 여기신호와의 오차를 최소화하는 코드워드의 인덱스와 이때의 이득은 다음 식을 만족해야 한다. 즉

$$\min_{g,c_i(n)} \sum_{n=0}^{N-1} (e(n) - g c_i(n))^2$$

여기서 $e(n)$은 양자화하려는 여기신호, $c_i(n)$는 i번째 코드워드, g는 이득, N은 샘플 수로 시작상태의 나머지 여기신호에 대해서는 23, 다른 서브블록에 대해서는 40이다. 식은 다음 식 을 최대화하는 $c_i(n)$를 찾는 것과 동일하게 된다. 특히, 1단계의 경우 $\sum_{n=0}^{N-1} e(n) c_i(n)$이 양수이어야 한다.

$$\frac{\left(\sum_{n=0}^{N-1} e(n) c_i(n) \right)^2}{\sum_{n=0}^{N-1} c_i^2(n)}$$

둘째, 이때의 최적화된 이득은 다음 식 (5-14)로 표현되며, 이 이득의 범위는 $|g| \leq 1.3$ 으로 제한된다.

$$g = \frac{\sum_{n=0}^{N-1} e(n)c_i(n)}{\sum_{n=0}^{N-1} c_i^2(n)} \quad .$$

각 단계의 코드워드의 인덱스 및 이득에 할당된 비트수는 표3과 같다. 제 2단계 및 제 3단계의 탐색을 위해 여기신호 $e(n)$은 다음 식과 같이 갱신된다.

$$e_2(n) = e(n) - \hat{g}\hat{c}_i(n)$$

여기서 $\hat{g}$은 바로 전단계 탐색에서 얻은 이득값이고 $\hat{c}_i(n)$은 바로 전단계 탐색에서 얻은 코드워드값이다. $e_2(n)$으로 부터 7번 과정부터 10번 과정을 수행하여 제 2단계에 대한 코드북을 검색한다. 마지막으로 지금까지의 과정을 반복하여 제3단계 코드북 검색을 수행한다.

· 비트스트림 생성

위와 같은 알고리즘을 이용하여 최종적으로 비트스트림을 생성한다. 생성된 비트스트림은 표3과 같은 구조를 가진다.

Parameter		Bits class <1,2,3>	Parameter		Bits class <1,2,3>
LSF	Split 1	6 <6,0,0>	Indices for CB sub block 1	Stage 1	8 <7,0,1>
	Split 2	7 <7,0,0>		Stage 2	7 <0,0,7>
	Split 3	7 <7,0,0>		Stage 3	7 <0,0,7>
	Subtotal	20 <20,0,0>		Subtotal	22 <7,0,15>

표 3 Bitstream of 15.2 kbps iLBC coder

– 기술적 과제

IP 네트워크 상의 오디오 컨퍼런싱 시스템에 대해, 가변 비트율 오디오 코덱을 이용한 잉여 정보(redundancy)의 가변 비트율 인코딩 및 전송하는 기능과 수신단의 실시간 음질측정 및 피드백하는 기능을 이용한 패킷 손실 은닉 방법 및 장치를 고안함으로써 패킷 손실 환경에서의 오디오 품질을 향상시킬수 있다.

· 품질 향상을 위한 전체 오디오 컨퍼런싱 시스템

종래 기술의 문제점을 감안한 것으로서, 특히, 인터넷으로 실시간 전송되는 음성급 신호에 손실이 발생하는 경우 전체적인 전송 비트율의 증가 없이 잉여 정보를 전송하여 손실 구간의 음성신호 패킷을 복구 또는 은닉하는 것으로, 음성통신의 패킷 손실을 은닉하는 단말장치 및 방법을 제안한다.

수신측에서 음성신호 패킷을 수신하고 음질평가한 결과를 이용하여 송신측에서 가변비트율을 결정하고 부가 음성 데이터의 유형에 의한 잉여 정보를 결정하여 송신하도록 하는 음성통신의 패킷 손실을 은닉하는 단말장치 및 방법을 제공한다.

인터넷을 통하여 오디오 신호로 실시간 통신하는 단말기에 있어서, 실시간 입력되는 오디오 신호와 잉여정보를 전체 비트율이 증가하지 않는 가변비트율로 인코드하고 음질평가 결과를 기록하여 실시간 전송 포맷으로 변환하며 패킷으로 인터넷에 송신하는 송신단 및 인터넷을 통하여 실시간 오디오 패킷을 수신하고 음질평가 결과를 확인하여 송신단에 피드백하며 가변비트율로 디코딩한 패킷을 순차 기록하고 지터를 제거하며 음질평가 결과를 피드백하고 재생된 음성신호를 출력하는 수신단이 포함되는 구성을 제시한다. 송신단은 실시간 인가되는 오디오 신호를 샘플링하여 입력하는 사운드 캡쳐부; 입력하는 오디오 신호를 디지털 신호로 인코딩하고 전체 비트율이 증가되지 않는 상태에서 이전 패킷에 대한 잉여 정보를 포함하여 인코딩하는 오디오 인코딩부; 오디오 인코딩부가 인코딩한 오디오 신호에 수신단이 피드백한 음질측정결과를 포함 기록하여 실시간 전송포맷으로 변환하는 실시간 포맷부 실시간 포맷부가 포맷한 오디오 신호를 인터넷으로 실시간 전송할 패킷으로 변환하여 송신하는 실시간 패킷부를 포함하여 구성되는 것을 특징으로 한다.

또한, 수신단은 인터넷을 통하여 상대편 단말기가 송신한 실시간 패킷을 수신하는 실시간 수신부, 실시간 수신부가 수신한 패킷을 분석하여 음질평가 결과를 확인하고 송신단에 피드백하는 실시간 분석부, 실시간 분석부에서 인가되는 신호로부터 가변비트율을 분석하여 수신된 패킷을 디코딩하는 오디오 디코더부, 디코딩된 패킷을 순서번호대로 기록하며 지터를 제거하는 버퍼부, 버퍼부에 저장된 패킷을 순차적으로 인가받고 설정된 소정 숫자 주기 마다 음질평가 결과를 측정하여 송신단에 피드백하는 품질측정부, 품질측정부로부터 인가되는 오디오 신호를 출력하는 오디오 렌더부를 포함하여 구성되고 오디오 인코딩부는 입력되는 오디오 신호와 잉여정보를 전체 비트율이 증가하지 않는 가변비트율로 인코딩하는 구성으로 이루어진다. 가변비트율은 수신단으로부터 확인되어 피드백된 음질평가 결과를 분석하여 확인되는 구성으로 이루어지고, 가변비트율은 상대방 단말기로부터 패킷에 포함되어 수신된 음질평가 결과를 분석하여 확인하는 구성으로 이루어진다. 실시간 포맷부는 수신단이 피드백한 측정 음질평가 결과를 페이로드 포맷의 예측 영역(MOS)에 기록하는 구성으로 이루어지고, 오디오 디코더부는 실시간 분석부의 신호를 분석하여 가변비트율을 확인하고 확인된 가변비트율로 디코딩하는 구성으로 이루어지는 것으로 한다. 수신부가 수신한 신호는 현재 패킷과 이전 패킷에 의한 잉여 정보 중에서 적어도 하나 이상이 포함되는 구성으로 이루어지는 것을 특징으로 한다. 잉여정보는 이전 패킷의 그 이전 패킷에 대한 잉여 정보가 더 포함되어 이루어지는 구성을 특징으로 한다. 가변비트율은 엠펙2(MPEG-2) 에이에이시(AAC)에 의하여 이루어지는 구성되고, 지터는 인터넷에서 발생하는 변이로 이루어지는 구성을 가진다.. 버퍼부는 디코딩된 패킷 신호를 버퍼에 기록하므로 상기 지터를 제거하는 구성으로 이루어지고, 품질측정부는 버퍼부로부터 설정된 소정 숫자 주기의 오디오 패킷 신호를 인가 받은 후에 음질평가 결과를 측정하는 구성으로 이루어지는 것을 특징으로 한다. 음질평가는 아이티유티(ITU-T) 표준 피563(P.563)의 논-인트루시브

(NON-INTRUSIVE) 방식을 이용하여 평가되는 객관적 결과로 이루어지게되며, 품질측정부는 음질평가의 결과를 1 내지 5 의 범위 값으로 표현하는 것을 특징으로 한다. 음질평가의 결과 값은 송신단의 실시간 포맷부에 피드백되어 기록되는 구성으로 이루어지고, 실시간 분석부는 확인된 음질평가 결과를 송신단에 피드백하는 구성으로 이루어지는 것을 특징으로 한다. 오디오 렌더부는 버퍼부로부터 인가되는 패킷 단위의 오디오 신호를 사운드 카드를 통하여 실시간으로 재생하고 출력하는 구성으로 이루어지는 것을 특징으로 한다.

음성통신의 패킷 손실을 은닉하는 방법에 있어서, 음성통신 단말기의 송신단에 의하여 오디오 신호를 입력하고 실시간 캡처하는 과정; 캡처된 오디오 신호를 확인된 음질평가 결과로 분석된 가변비트율에 의하여 잉여정보와 함께 인코딩하는 과정, 인코딩된 오디오 신호와 측정된 음질평가 결과를 실시간 전송 포맷으로 변환하는 과정 및 실시간 전송 포맷의 오디오 신호를 패킷으로 변환하여 인터넷에 송신하는 과정을 포함하는 구성을 제시한다. 상기 인코딩하는 과정은 확인된 음질평가 결과에 의하여 잉여정보를 인코딩하지 않고, 인코딩하는 과정은 확인된 음질평가 결과에 의하여 하나의 잉여정보 또는 두 개의 잉여정보를 함께 전체 비트율이 증가하지 않는 상태에서 인코딩한다. 하나의 잉여정보는 현재 인코딩할 패킷의 이전 패킷에 대한 정보 인 것을 특징으로 한다. 두 개의 잉여정보는 현재 인코딩할 패킷의 이전 패킷과 그 이전 패킷에 대한 정보이고, 측정된 음질평가 결과는 상대방으로부터 수신된 오디오 패킷을 설정된 소정 숫자 단위로 누적하여 음질평가한 결과이다.

음성통신의 패킷 손실을 은닉하는 방법에 있어서, 음성통신 단말기의 수신단에 의하여 오디오 패킷을 실시간 수신하고 분석하여 확인된 음질평가 결과를 송신단에 피드백하는 과정; 음질평과 결과로부터 분석된 가변비트율로 수신된 패킷을 디코딩하고 순서번호에 의하여 버퍼에 저장하는 과정; 및 버퍼에 저장되고 설정된 소정 숫자 단위의 패킷으로부터 측정된 음질평가를 송신단에 피드백하는 동시에 실시간 오디오 신호로 변환하여 출력한다. 음질평가 결과는 수신된 패킷을 분석하여 기록된 음질평가 결과를 확인하고, 가변비트율은 확인된 음성평가 결과로부터 분석하여 확인한다. 음질평가는 버퍼에 설정된 소정 숫자 단위로 패킷이 누적 기록되는 주기마다 측정한다. 또한, 상기 오디오 신호의 출력은 음질평가의 측정과 관계없이 버퍼에 순서대로 기록되는 오디오 패킷을 인가받아 변환하고 출력한다.

음성통신의 패킷 손실을 은닉하는 방법에 있어서, 음성통신 단말기의 수신단에 의하여 상대방으로부터 수신된 패킷을 분석하고 음질평가 결과를 확인하는 단계. 분석된 음질평가 결과에 의하여 전송비트율을 가변하고 캡처된 오디오 신호와 잉여 정보를 코딩하는 단계; 수신된 패킷을 설정된 소정 주기 단위로 누적하여 객관적인 음질평가를 측정하는 단계 및 측정된 음질평가 결과를 송신단에 피드백하여 포맷의 지정 영역에 기록하고 송신하는 단계가 포함되는 구성이다. 잉여정보를 코딩하는 단계는 분석된 음질평가 결과에 의하여 잉여 정보를 코딩하지 않거나 이전 패킷의 잉여정보를 코딩하거나 이전 패킷과 그 이전 패킷의 잉여정보를 코딩하는 것 중에서 선택된 어느 하나이다.

오디오 컨퍼런싱 시스템(Audio Conferencing System: ACS)의 구현은 양방향 오디오 전송을 통한 음성 및 오디오 통신을 목적으로 한다. 본 발명에서의 오디오 컨퍼런싱 시스템은 음성 및 오디오의 효율적인 전송을 위해 MPEG-2 AAC(Advanced Audio Coding)와 같은 가변 비트율 오디오 코덱을 적용한다. 또한 음성 및 오디오 비트스트림은 MPEG 오디오에 관한 IETF(Internet Engineering Task Force) draft 표준에 정의된 RTP(Real Time Protocol) payload format(M. Kretschmer, A. Basso, M. R. Civanlar, S. R. Quackenbush, and J. H. Snyder, RTP payload format for MPEG-2 and MPEG-4 AAC streams, IETF Draft, July 2001.)을 수정 및 활용하며, 오디오 컨퍼런싱 시스템은 두 사용자를 직접 연결내지는 하나의 공통 멀티캐스트그룹 안의 여러 사용자를 동시에 연결한다.

그림 21은 오디오 컨퍼런싱 시스템의 전체 구성도를 보여준다. 그림 21에서 101과 102는 구현하고자 하는 오디오 컨퍼런싱 시스템 단말을 나타낸 것으로, 각각은 송신단(103, 105)과 수신단(104, 106)으로 나뉠 수 있으며, 송신단과 수신단 내부의 블록들은 두 단말이 인터넷을 통해 오디오 컨퍼런싱을 수행하기 위한 단계별 처리과정을 나타낸다.

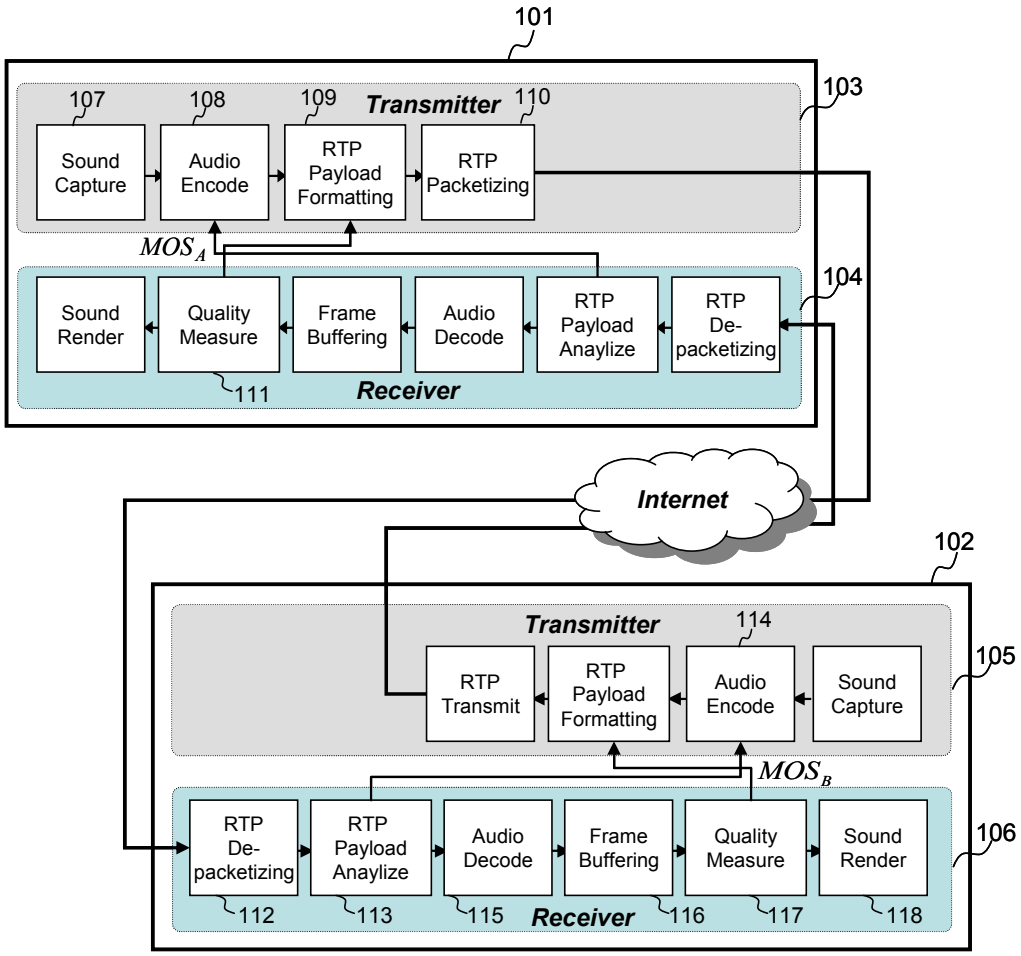


그림 21 오디오 컨퍼런싱 시스템 구성도

먼저 오디오 컨퍼런싱 시스템 A(101)의 송신단(103)은 오디오를 실시간으로 입력받아(107) 오디

오 인코딩(108)을 수행한다. 이 때 오디오 인코딩의 경우, MPEG-2 AAC와 같은 가변 비트율을 지원하는 코덱을 적용한다. 이로써 송신단이 패킷 손실 은닉을 위해 이전 프레임에 대한 잉여 정보 전송을 수행하게 되는 경우에 현재 프레임과 이전 프레임 잉여 데이터의 전송 비트율을 조절하여, 전체 전송 비트율이 증가되지 않고 유지될 수 있도록 한다. 오디오 인코딩을 통해 생성된 오디오 프레임은 RTP payload formatting(109) 과정을 통해, 실시간 전송에 적합한 포맷을 갖추게 된다. 이는 프레임 연속성 및 패킷 손실 복원에 필요한 정보와 다중 오디오 프레임의 전송 및 그에 대한 정보를 제공함으로써 가능해진다. 또한 이 과정에서, 송신단은 수신단으로 부터 이전까지 수신된 오디오 스트림에 대한 음질 측정(111) 결과를 입력받아 RTP payload format의 예측 필드(prediction field)에 기록한다. 이로써 향후 상대편 단말의 송신단(105)은 수신된 패킷의 RTP payload format의 예측 필드를 분석하여 잉여 정보의 인코딩 여부 및 유형을 결정하게 된다. 다음으로는 RTP payload format으로 표현된 오디오 프레임을 최종적으로 RTP 패킷화하여(110) 네트워크로 전송한다.

이에 대해 오디오 컨퍼런싱 시스템 B(102)의 수신단(106)은 인터넷 망으로부터 수신되는 RTP 패킷을 수신하고(112), RTP payload 분석(113)을 수행한다. 이 과정에서 상대편 단말에서의 음질평가 결과를 추출하여 송신단의 오디오 인코딩 블록(114)으로 피드백한다. 이후에는 오디오 디코딩(115)을 수행한다. 이 때 가변 비트율 오디오 코덱은 인코딩된 비트율을 판별하여 그에 맞게 디코딩한다. 이후에는 네트워크에서 발생하는 지연 변이(jitter)를 제거하기 위해 프레임 버퍼링(116)을 거쳐 규칙적인 간격으로 재생을 수행할 수 있다. 이 때 RTP payload format의 순서번호(그림 22의 209)를 참조하여 순차적으로 버퍼에 쌓는다. 또한 버퍼링된 오디오 데이터는 음질평가(117)에 사용된다. 즉, 일정 수의 오디오 프레임이 디코딩될 때까지 버퍼링한 후 음질평가를 실시한다. 음질평가 방법으로 ITU-T(International Telecommunications Union - Telecommunication Standardization Sector) 표준 P.563 과 같은 non-intrusive 방식의 객관적 음질평가 방법을 사용하며, 평가 결과는 1에서 5 사이의 MOS(Mean Opinion Score)로 표현한다. 매 프레임이 아닌 일정 프레임 수 주기마다 수행되는 음질평가 블록과 무관하게 디코딩된 매 프레임은 사운드카드를 통해 실시간으로 재생(118)된다.

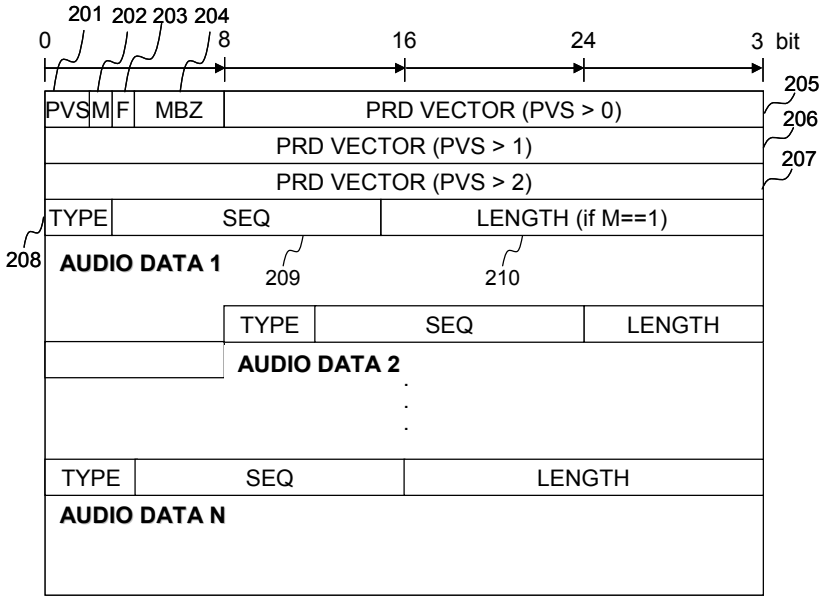


그림 22 RTP Payload Format

· 송신단의 잉여 정보 전송 및 수신단의 음질 피드백을 위한 RTP payload format

IETF의 RFC(Request For Comments) draft에서 정의된 MPEG-2 및 MPEG-4 AAC 오디오를 위한 RTP payload format을 근간으로 음질평가 방법을 송수신단에서 교환하는 format을 고안한다. 그림22는 IETF의 RFC draft에 정의된 원래의 RTP payload format을 보여준다. 이 포맷은 다음의 정보를 가짐으로써 패킷 손실 은닉 및 패킷 동기화 능력을 갖는다. 첫째, 프레임이 네트워크에서 손실되는 경우 이를 복원하기 위한 정보, 둘째, 프레임이 여러 RTP 패킷에 분할(fragmentation) 되었는지에 대한 정보, 셋째, 프레임을 그룹화 및 인터리빙(interleaving)을 했는지에 대한 정보, 넷째, 실제 프레임 데이터이다.

그림 22의 상세 필드를 보면, 먼저 'PVS'(201)는 32 비트의 'PRD VECTOR'(205, 206, 207)가 몇 개 있는지를 나타낸다. 'PRD VECTOR'는 각 프레임에 대한 PQ(Predictability Quantifier) 정보를 비트 이동으로 합쳐놓은 것으로, PQ의 값은 다음과 같은 의미를 갖는다. '0'은 다른 AAC 프레임을 이용하여 예측될 수 없음을, '1'은 이전 프레임으로부터 또는 다음 프레임으로부터 예측될 수 있음을, '2'는 이전 프레임 또는 다음 프레임 또는 이전/이후 양 프레임으로부터 예측될 수 있음을 의미한다. 프레임이 여러 RTP 패킷에 분할되었는지에 대한 필드인 'F'(203)는 현재 패킷에 포함된 프레임의 분할여부를 나타낸다. 또한 프레임이 그룹화 또는 인터리빙 되었는지에 대한 정보는 'M'(202)으로, 현재 패킷에 몇 개의 프레임이 연속되었는지를 나타낸다. 마지막으로, 실제 프레임 데이터를 나타내는 필드는 프레임 타입, 순서, 길이 정보를 포함한 데이터 블록으로, 'TYPE'(208)은 데이터의 타입을, 'LENGTH'(210)는 프레임 길이를, 'SEQ'(209)는 프레임의 순서번호를 나타낸다. 여기에서 'TYPE'이 '0'이면 인코딩된 원본 데이터를, '1'이면 인코딩된 원본 데이터이지만 패킷 복원을 위한 잉여 정보를, '2'이면 비트율 감소를 위해 높은 압축율로 압축된 인코딩 데이터임을 나타낸다. 또한 'LENGTH'는 바이트 수로 나타낸다.

그림 23은 수정된 RTP payload format을 나타낸 것이다. 즉, 프레임의 예측 정보를 실을 수 있는 필드인 'PRD VECTOR'에는 수신단에서 주기적으로 측정되는 음질평가 결과인 MOS(303)를 기록하도록 한다. 'PRD VECTOR' 기록되는 값의 범위는 1에서 5 사이에 해당된다. 음질평가 결과가 실렸는지의 여부는 'PVS' 값을 참조함으로써 판별할 수 있다. 즉, 'PVS' 값이 '1'이상이면 'PRD VECTOR'가 존재하며 그에 따라 음질평가 결과가 피드백 되었는지 여부를 확인할 수 있다. 또한 송신단에서 이전 프레임에 대한 잉여 정보를 전송하게 되는 경우에는 메인 프레임 다음의 오디오 데이터 영역(305, 306)을 이용하여 그룹화한 후 전송한다.

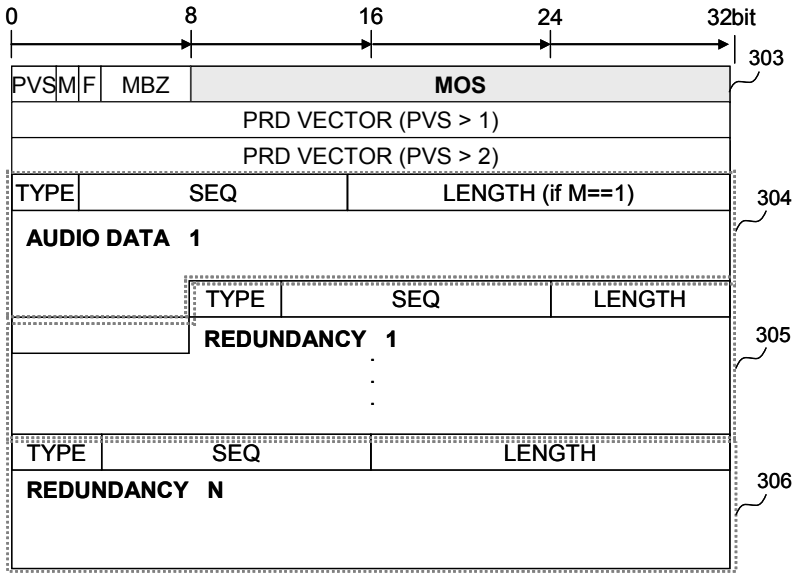


그림 23 수정된 RTP Payload Format

·가변 비트율 오디오 코덱과 음질측정에 기반한 패킷 손실 은닉 방법

앞서 설명한 오디오 컨퍼런싱 시스템의 패킷 손실 은닉 방법은 전진 에러정정 기법 중의 하나인 잉여 정보 코딩 방식에 기초한다. 그림 24 송신단에서의 잉여 정보 인코딩 방법을 결정하기 위한 정보로 활용되는 음질평가 결과를 얻기 위해, 수신단에서 실시간 음질평가를 수행하는 방법을 나타낸 것이다. 우선, 수신단은 RTP 패킷 분석(401) 및 RTP payload format 분석(402)을 수행한 후, 압축되어 전송된 오디오 데이터를 디코딩(402)한다. 이 때, 패킷 손실이 발생한 경우에는 디코딩할 데이터가 NULL인 상태가 되어 디코딩 및 재생 과정이 생략된다. 다음의 음질평가 과정은 손실된 패킷을 포함한 총 N개의 패킷이 도달하여 이에 대한 raw 오디오 데이터가 생성될 때마다 수행된다. MOS로 표현되는 음질평가를 위해서는 광대역의 오디오 신호를 다운 샘플링(403)한다. 다운 샘플링된 오디오 데이터는 실제 음질평가 블록(404)으로 입력된다. 음질평가 방법으로는 ITU-T 표준 P.563과 같이 참조음원(reference) 없이 독립적으로 음질평가를 수행할 수 있는 non-intrusive한 평가방법을 채택한다. 최종적으로 수신단의 음질평가 장치는 N개의 RTP 수신에 대해 하나의 MOS 출력을 도출한다.

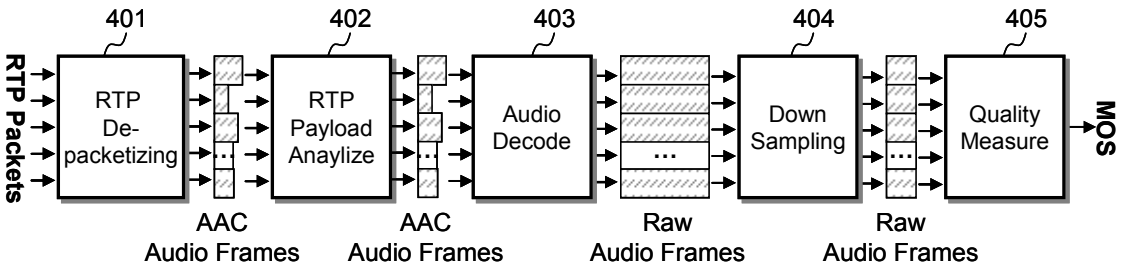


그림 24 RPT Packet 전송

그림 25는 송신단이 상대편 수신단으로 부터 피드백된 음질평가 결과에 따라 잉여 정보 인코딩을 차별적으로 수행하는 방법을 보여준다. 도면에서 MOS_{N_i}는 상대편 수신단으로 부터 피드백된 음질평가 결과이다. 그림의 501, 502, 503과 같이 음질평가 결과인 MOS 값의 범위에 따라 네트워크에서의 패킷 손실 정도를 판단한다. 즉, MOS가 기준치 이하로 낮으면 잉여 정보 생성을 늘리고(504), 기준치 이상이 되면 잉여 정보 생성을 줄이는(505) 형태이다. 이 때, 경우에 따른 전송 비트율을 동일하게 유지하기 위해 메인 프레임과 잉여 정보의 전송 비트율을 가변하면서 인코딩한다. 다시 말하면, 원래의 전송 비트율 $BitRate_F$을 현재 프레임과 이전 프레임에 각각 $BitRate_M$와 $BitRate_R$로 나누어서 할당한다. 즉, $BitRate_F = BitRate_M + BitRate_R$ 또는 $BitRate_F = BitRate_M + BitRate_{R1} + BitRate_{R1}$등의 형태가 된다. 이와 같이 몇 가지의 음질 기준을 두고, 그에 따라 가변 비트율 오디오 코딩을 통해 현재 프레임 및 잉여 정보 코딩을 수행한다.

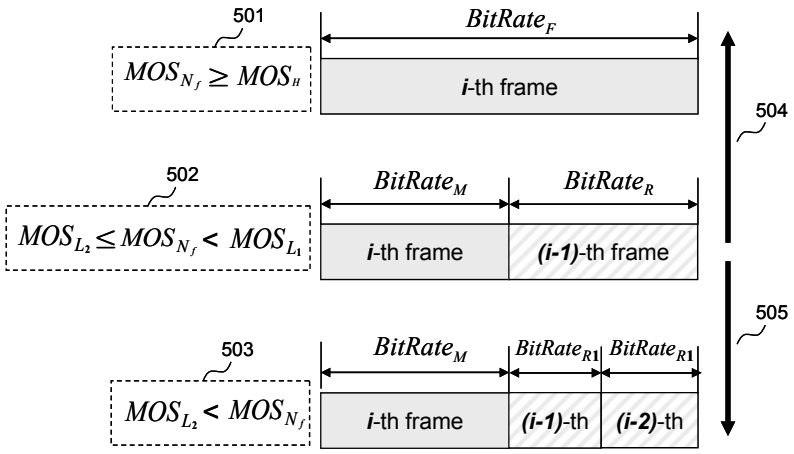


그림 25 Encoding

그림 6는 A와 B 두 지점간의 오디오 컨퍼런싱에서의 전체 패킷 손실 은닉 장치의 동작을 보여준다. A 지점의 송신단(601)는 실시간으로 오디오 인코딩을 수행하여 인코딩된 프레임 데이터를 RTP 전송하고 B 지점의 수신단(602)는 이를 디코딩하여 프레임 단위로 버퍼에 순차적으로 넣는다. 이 때 N개의 프레임이 수신될 때마다 그에 대한 음질측정을 수행하여 그 결과를 A 지점의 송신단으로 피드백한다.

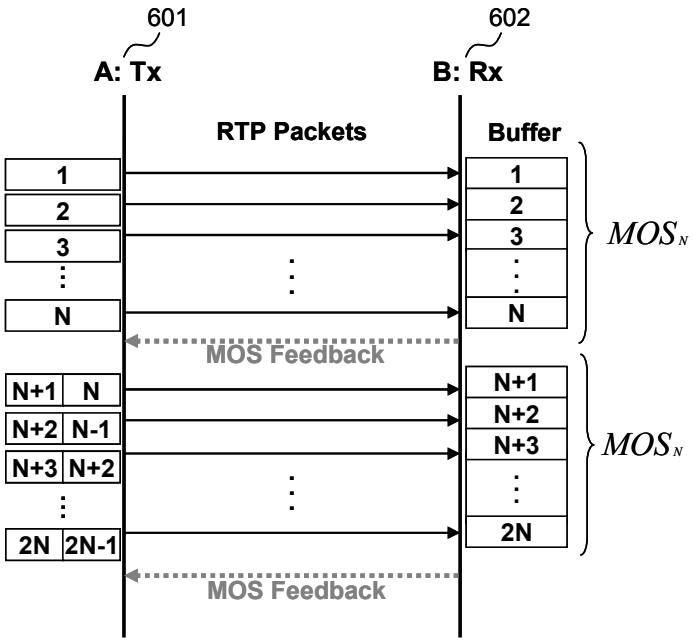


그림 26 Packet feedback operation

· 구현 및 성능평가

먼저 오디오 컨퍼런싱 시스템 구현을 위해, 공개소스 기반의 다자간 음성협업 소프트웨어인 RAT(Robust Audio Tool)(http://www-mice.cs.ucl.ac.uk/multimedia/software/rat/)와 MPEG-2 AAC 공개소스(http://sourceforge.net/projects/faac/)를 이용하였다. 먼저 RAT 구조에 맞게 MPEG-2 AAC 인코딩/디코딩 API(Application Program Interface) 라이브러리를 구현하고, 이를 RAT 프로그램에 추가하여 오디오 컨퍼런싱 시스템을 구현하고, 이후에는 고안한 패킷 손실 은닉 장치를 RAT에 추가적으로 구현하였다.

성능평가를 위해, MPEG-2 AAC 인코딩을 위한 음원은 32 kHz 스테레오로 설정하고, 인코딩 비트율은 146 kbit/s로 하였다. 음질평가를 위한 다운샘플링 팩터는 1/4 로 하여 8 kHz 모노의 음질평가용 수신 음성을 얻었다. 또한 수신단의 음질평가 주기 N은 8초 길이인 250 프레임으로 구현하고, 음질 기준치는 음성통화가 가능한 수준인 3.6으로 설정하였다. 송신단의 잉여 정보 인코딩을 위한 가변 비트율은 현재 프레임과 이전 프레임에 각각 1/2 씩 동일하게 할당하는 것으로 실험하였다.

음질 실험은 A와 B 두 지점을 두고 A 지점 송신단에서 실험음원을 RTP 전송한 후, B 지점 수신단에서는 이에 대한 음질평가를 수행하는 시뮬레이션 환경에서 이루어졌다. 실험 음원은 SQAM(Sound Quality Assessment Material)(http://sound.media.mit.edu/mpeg4/audio/sqam/) 중에서 스테레오이고 32 kHz로 샘플링된 여성 보컬음성 32초를 사용하며, 고안한 패킷 손실 은닉 방법의 적용에 대한 음질 비교를 수행하기 위해 무손실 음원과 손실 음원을 각각 준비하였다. 특히, 손실 패턴은 ITU-T 표준 G.191에 정의된 Gilbert-Elliot 모델을 이용하여 7% 손실로 제작하였다. 따라서 음질 비교는 무손실로 음원을 전송하는 경우와 7% 손실로 음원을 전송할 때의 고안 PLC 알고리즘을 적용하기 전과 후 각 경우에 대해 수행하였다.

그림 7은 이에 대한 결과를 나타낸 것으로, 7% 패킷 손실 환경에서 고안한 패킷 손실 은닉 방법을 적용한 결과, MOS 값을 2.702에서 3.580로 24%의 음질개선 효과를 확인하였다. 즉, 패킷 손실 환경에서는 오디오 코덱의 비트율을 낮추고 그에 상응하는 비트율을 이전 프레임의 잉여 정보 인코딩에 사용하는 것이 동일한 자원을 소비하면서도 음질을 개선할 수 있다.

조건 / 구간(초)	무손실	7% 손실	
		PLC 적용 전	PLC 적용 후
0 - 8	4.610	3.149	3.149
8 - 16	3.755	2.246	3.450
16 - 24	3.515	2.336	3.393
24 - 32	4.377	3.078	4.327
평균	**4.064**	**2.702**	**3.580**

표4 측정결과

결론

본 기술로서 이전의 송신단 기반 패킷 손실 은닉 방법이 잉여 정보 전송함에 있어서 전송 비트율을 증가시키는 점과 RTCP와 같은 별도의 전송채널을 이용하여 패킷 손실 정보를 피드백 하는 오버헤드를 극복할 수 있다. 또한 수신단의 실시간 음질평가를 통해 패킷 손실에 의한 음질저하 정도를 음질평가 결과에 견주어 판별함에 따라, 패킷 손실 은닉을 수행할 때에 보다 명확한 근거를 제시할 수 있다. 따라서 본 발명은 보다 효율적이고 명확한 패킷 손실 은닉 방법으로 활용될 수 있고 더불어 패킷 손실 환경에서의 오디오 컨퍼런싱 시스템의 오디오 품질을 향상시킬 수 있다.

참고문헌

[1] "Error concealment 래개 compressed digital audio" P. Lauber and R. Sperschneider, in Aes 111th Convention, NY, PP 1-11 Sept. 2001

[2] "Error Mitigation in mpeg-audio packet communication systems"S. Qauckenbush and P. Driessen, in Aes 115th Convention, NY, pp. 1-11, oct. 2003

[3] "Packet loss concealment basd on sinusoidal extrapoltion" J. Lindblom and P. Hedelin, in proc. ICASSP, pp. 173-176, May 2002

[4] "Adaptive recovery techniques for real-time audio streams" W.T. Liao, J.C. Chen, M.S. Chen, in proc. IEEE Inforcom, Anchorage, AK, Vol. 2 pp. 815-823, Apr. 2001

[5] "Packet loss concealment for auto streaming based on the gapes and mapes algorithms" H. Ofir and D. Malah, in IEEE 24th Convention of electrical and Eletronics Engineering, Israel, pp. 280-284, Nov. 2006

[6] I. Kouvelas, O. Hodson, V. Hardman, and J. Crowcroft, "Redundancy control in real-time Internet audio conferencing," in *Proc. International Workshop on Audio-Visual Services over Packet Networks (AVSPN97)*, Aberdeen, Scotland, Sept. 1997.

[7] P. Lauber and R. Sperschneider, "Error concealment for compressed digital audio," in AES 111th convention, NY, pp. 1-11, Sept. 2001.; S. Quackenbush and P. Driessen, "Error mitigation in MPEG-audio packet communication systems," in AES 115th convention, NY, pp. 1-11, Oct. 2003.

[8] W.-T. Liao, J.-C. Chen, and M.-S. Chen, "Adaptive recovery techniques for real-time audio streams," in Proc. IEEE INFOCOM, Anchorage, AK, vol. 2, pp. 815-823, Apr. 2001.