Journal of Korea Multimedia Society Vol. 10, No. 8, August 2007(pp. 1039-1051)

초저지연 비디오 통신을 위한 RTP 기반 립싱크 제어 기술에 관한 연구

김병용[†], 이동진[††], 심동규[†††], 권재철[††††]

요 약

본 논문은 비디오통신 시스템에서 초저지연을 달성하면서 립싱크 제어하는 방법을 제안한다. 초저지연 비디오 통신에서 핵심적인 기술은 종단간 지연시간을 줄이는 기술과 립싱크 제어 기술이다. 특히 서비스관점에서 립싱크 제어 기술이 중요한 요인으로 작용하고 있다. 오디오와 비디오의 데이터를 RTP/RTCP 기반으로 패킷을 구성하여 전송하고, 이 패킷을 이용하여 오디오와 비디오의 재생시간을 계산한 후 립싱크 제어를 한다. 본 논문에서는 오디오 데이터가 일정한 간격으로 재생되도록 하고, 오디오가 재생되는 시점에서 가장 근접한 재생시간을 가진 비디오 데이터를 찾아서 재생하는 방법으로 오디오와 비디오간의 립싱크 제어하는 방법을 제안한다. 그리고 종단간 지연시간이 100 ms이하인 초저지연 비디오 통신을 하기 위해서는 송신단의 인코딩 버퍼 제거하여 지연시간을 줄이고, 수신단의 재정렬버퍼 (Reordering Buffer)와 립싱크 버퍼의 크기를 3 프레임으로 처리하여 종단간 지연시간을 최소로 하였다. 실험결과에서 종단간 지연시간이 100 ms이하를 유지하고 오디오와 비디오의 립싱크 제어를 하였다.

A Study on RTP-based Lip Synchronization Control for Very Low Delay in Video Communication

Byoung-Yong Kim[†], Dong-Jin Lee[††], Dong-Gyu Sim[†††], Jae-Cheol Kwon[††††]

ABSTRACT

In this paper, a new lip synchronization control method is proposed to achieve very low delay in the video communication. The lip control is so much vital in video communication as delay reduction. In a general way, to control the lip synchronization, both the playtime and capture time calculated from RTP time stamp are used. RTP timestamp is created by stream sender and sent to the receiver along the stream. It is extracted from the received packet by stream receiver to calculate playtime and capture time. In this paper, we propose the method of searching most adjacent corresponding frame of the audio signal, which is assumed to be played with uniform speed. Encoding buffer of stream sender is removed to reduce the buffering delay. Besides, decoder buffer of receiver, which is used to correct the cracked packet, is resulted to process only 3 frames. These mechanisms enable us to achieve ultra low delay less than 100 ms, which is essential to video communication. Through simulations, the proposed method shows below the 100 ms delay and controlled the lip synchronization between audio and video.

Key words: RTP, RTCP, Low Delay(저지연), Lip Synchronization(립싱크), Video Communication(비디오통신)

1. 서 론

최근 몇 년간 VoIP (Voice over IP)는 패킷망을 기반으로 음성 전송 서비스를 제공하면서 기존 PSTN (Public switched telephone network) 중심의 유선전화 시장을 대체하는 새로운 서비스로 각광을 받고 있다. 하지만, 유선전화에 비하여 QoS (Quality of service)가 보장되지 않는 낮은 통화품질, 통화단

절 현상, 기존 유무선 전화망과 상호접속 미흡으로 인한 착신 서비스 불가능, 그리고 사업자의 임의 착신번호 부여 등의 문제점은 인터넷 전화가 시장에서 주요한 통신서비스로 자리매김 할 수 없게 하는 걸림돌이 되었다. 따라서 최근 네트워크와 VoIP 서비스에 관련된 연구를 통합함으로써 광대역 통신망(BcN)과 같은 대역폭이 넓은 통신망이 발전하고, 기존의 음성 서비스뿐만 아니라 영상 서비스를 제공하는 통합형 멀티미디어 서비스가 발전하였다. 특히 BcN망의 발전은 대량의 데이터 전송에 따른 지연시간을 줄임으로써 영상과 음성의 QoS을 향상시켰다. 그러나 이러한 네트워크의 발전에도 불구하고, 실시간 초저지연 비디오통신 서비스의 상용화 기술은 아직 완벽한 상태에 있지 않은 실정이다. 비디오통신 시스템의 핵심 기술은 고품질의 오디오 및 비디오의 실시간 전송과, 립싱크 제어 기술이다. 일반적으로 저지연 실시간 비디오통신 시스템에서 실시간 데이터 전송을 위해서는, 혼잡제어를 수행하지 않는 UDP (User datagram protocol)와 실시간 전송 기능을 제공하는 RTP (Real time protocol)를 사용하여 고품질의 오디오 및 비디오 서비스를 제공한다. 그러나 각각의 오디오와 비디오 패킷은 서로 다른 경로를 통하여 패킷이 전송이 되기 때문에, 상대 단말기에 도착하기까지 서로 다른 지연시간을 가지게 된다. 또한, 오디오와 비디오 데이터는 인코더와 디코더에서 지연되는 시간이 각각 다르므로 재생되는 시간 또한 서로 다르게 동작한다. 이러한 요소들에 의해 오디오와 비디오의 립싱크가 불일치하여, 서로 의사소통을 하는데 불편을 줄 수가 있다. 따라서 실시간 비디오통신을 위해서는 지연시간을 줄이는 것뿐만이 아니라 오디오와 비디오 사이의 립싱크 제어하는 것도 매우 중요하다. 일반적으로 오디오와 비디오의 재생시간의 차이가 100 ms 이하면, 사람의 눈으로 이를 식별하지 못한다[1].

기존의 립싱크 제어방법은 RTP 패킷에 있는 RTP 타임스탬프를 이용하여 오디오 및 비디오 데이터가 캡처된 시점과 재생되는 시점의 차이 값을 각각 계산하고, 이 두 차이 값의 차를 계산함으로써 오디오와 비디오 사이의 지연시간을 계산한다. 이렇게 계산되어진 지연시간을 비디오 또는 오디오 재생 시간에 추가하여 재생함으로써 오디오와 비디오의 립싱크 제어를 한다. 이 방법은 미디어의 재생 시간에 추가적인 지연이 발생시키므로 음질 및 화질이 떨어지게 된다.

본 논문에서는 이런 문제점을 해결하고자 새로운 립싱크 제어 방법을 제시한다. 오디오는 샘플링 비율 (Sampling rate)이 매우 큰 신호이기 때문에 데이터손실이 발생하면 사람의 귀에 바로 식별되어 음질이 크게 떨어진다. 그러나 비디오는 시간 축에서 비교적 낮은 샘플링 비율을 갖는 신호이기 때문에 데이터손실이 발생하는 경우, 손실된 영상 대신 다음 영상을 재생하여도 몇 프레임으로 재생이 되었는지를 사람의 눈으로는 크게 구별되지 않는다.

이러한 점을 창안하여 본 논문에서는, 오디오 데이터는 일정한 간격으로 재생되도록 하고, 비디오 데이터는 오디오의 재생시간에 맞추어 가장 근접한 비디오의 재생시간을 찾아서 재생하는 방법으로 오디오와 비디오간의 립싱크 제어를 한다. 또한, 초저지연 비디오 통신을 위해 송신단의 인코딩 버퍼를 제거하여 버퍼링에 의해 발생하는 지연시간을 줄이고, 수신단에서는 패킷 순서를 보정하는 재정렬 버퍼와 립싱크를 하기 위한 립싱크 버퍼가 필요하다. 이 버퍼의 크기를 최소로 하기 위해서 3 프레임의 크기로 설정하여 초저지연의 비디오통신 시스템을 구현하였다.

본 논문의 구성은 다음과 같다. 2절에서는 RTP/RTCP와 기존의 립싱크 제어 방법에 대해서 설명하고, 3절에서는 제안한 립싱크 제어시스템에 대해서 설명한다. 4절에서는 제안한 립싱크 제어시스템에 대한 실험결과를 분석하고, 5절에서는 결론을 제시한다.

※ 교신저자(Corresponding Author) : 심동규, 주소 : 서울시 노원구 월계동 광운대학교 화도관 635(139-701), 전화 : 02)940-5470, FAX : 02)941-6470, E-mail : dgsim@kw.ac.kr
접수일 : 2007년 4월 10일, 완료일 : 2007년 7월 16일
† 정회원, 광운대학교 컴퓨터공학과
(E-mail : kby_car@kw.ac.kr)
†† 정회원, 광운대학교 컴퓨터공학과
(E-mail : woji@kw.ac.kr)
††† 광운대학교 컴퓨터공학과
†††† KT 미래기술연구소
(E-mail : jckwon@kt.co.kr)
※ 본 연구는 KT 미래연구소에서 지원한 "IP기반 양방향 영상통신을 위한 립싱크 제어 기법 연구"와 일부 "서울시 산학협력사업"을 통하여 이루어졌습니다.

2. RTP/RTCP와 기존의 립싱크 제어 방법

RTP 패킷은 미디어의 데이터를 지속적으로 전송하고, RTCP는 전송된 RTP 패킷을 효율적으로 제어함으로써 고품질의 음성과 영상을 제공하기 위한 것이다. 본 장에서는 실시간 비디오통신에서 사용하는 RTP/RTCP와 이를 이용한 기존의 립싱크 제어 방법에 대하여 다룬다.

2.1 RTP/RTCP

일반적으로 실시간 비디오 통신시스템에서는 데이터 전송을 위해 UDP기반의 RTP (Real-time protocol)와 RTCP (Real-time control protocol)를 사용한다[2,3]. 그림 1은 RTP 패킷의 구조를 보여준다. RTP 패킷은 미디어 데이터와 미디어의 기본적인 정보를 포함한다. RTP 패킷에서 립싱크 제어를 위하여 마커비트 (M bit)와 시퀀스 넘버 (Sequence number), RTP 타임스탬프 (Timestamp)를 사용하고, RTP의 다른 부가적인 정보들은 패킷의 유효성을 검사하거나 여러 사람과 비디오 통신을 할 경우에 사용한다. 비디오 데이터의 마커 비트는 분할된 프레임의 마지막 패킷에 표시된다. 비디오 스트림에서 마커비트가 1이라는 것은 한 프레임을 다 전송하였으니 다른 패킷을 계속 기다리지 않고 한 프레임을 디코딩을 하여도 된다는 것을 어플리케이션에게 알려주는 기능을 한다. 오디오 데이터의 마커 비트는 오디오의 침묵기간(silence period)이 끝난 후 다음에 전송되는 패킷에 표시된다. 다시 말해서, 오디오 스트림을 전송할 때 전송될 패킷의 마커비트가 1이라는 것은 데이터가 전송되지 않는 침묵기간 후에 전송

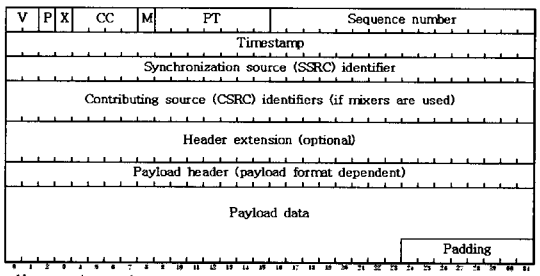


그림 1. RTP 패킷의 구조

된 첫 번째 패킷을 의미한다. 시퀀스 넘버는 패킷이 손실되거나 순서에 어긋나게 전송된 경우 수신자에게 데이터의 순서를 알려주기 위해서 사용된다. 이는 16비트 양의 정수 값이고 한 패킷이 전송된 후 다음 패킷의 시퀀스 넘버는 이전 패킷의 시퀀스 넘버보다 하나 증가한 숫자를 갖게 된다. 또한, 시퀀스 넘버의 초기 값은 임의적으로 선택되고, 시퀀스 넘버의 최대 값에 도달하면 시퀀스 넘버는 다시 영으로 돌아간다. 이러한 랩 어라운드 (Wrap-around)는 자주 발생하기 때문에 어플리케이션은 랩 어라운드를 고려한 패킷 수신을 해야 한다. RTP 타임스탬프는 패킷 내에서 미디어 데이터의 첫 번째 옥텟 (Octet)을 위한 샘플링 순간을 나타내고 이는 미디어 데이터의 재생과 관련된 스케줄링을 하는데 사용된다. 32비트 양의 정수 값인 RTP 타임스탬프는 미디어의 샘플링 비율에 따라 그 값이 증가한다. 즉, 시퀀스 넘버처럼 단계적으로 1씩 증가하는 것이 아니라 미디어의 샘플링과 관련되어 증가하게 되는 것이다. 이 때 RTP 타임스탬프의 값은 연속적인 시퀀스의 형태여야만 한다. 그러나 최대 값에 도달하면 RTP 타임스탬프의 값은 다시 0으로 설정하는 랩 어라운드가 일어나고, 초기 값 설정은 시퀀스 넘버와 같은 방법으로 설정한다. RTP 타임스탬프의 랩 어라운드는 RTP 동작의 일반적인 부분으로써 모든 어플리케이션 (Application)에 의해 다루어져야 한다. RTP 패킷에서 미디어 코덱을 사용할 경우에는 RTP 페이로드 해더 (payload header)에 미디어 코덱의 정보를 보내진다[4].

RTCP 패킷은 RTP 전송에 참여하는 어플리케이션들 간에 분실된 패킷 수, 지터 간격, 패킷과의 지연시간 등의 정보를 교환함으로써, 해당 응용 프로그램의 적합한 서비스 품질을 평가하고, 이에 따른 적응성 있는 인코딩을 제공하도록 한다. 즉, RTCP 패킷은 RTP 패킷을 제어하기 위해서 사용되는 프로토콜로서, RR (Receiver report), SR (Sender report), SDES (Source description), APP (Application), 그리고 BYE 이렇게 5가지 종류의 패킷으로 구성되어 있다. 비디오 서비스 응용 프로그램은 RR과 SR 패킷에 따라 립싱크 제어를 한다.

RR 패킷은 송수신에 대한 통계 정보를 송신자에게 알려주기 위해서 전송되는 패킷이다. 그림 2는 RR 패킷의 구조를 보여준다. SSRC_n은 32비트 필드로 RTCP 패킷에 포함된 각 수신자 리포트 블록

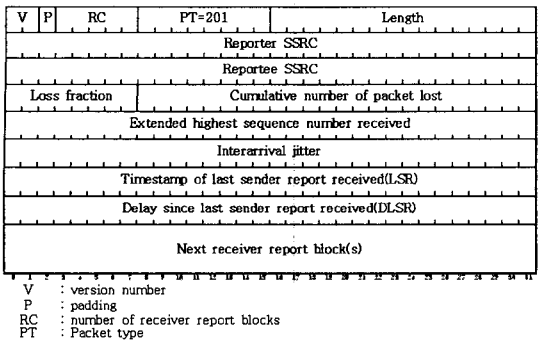


그림 2. RTCP RR 패킷 구조

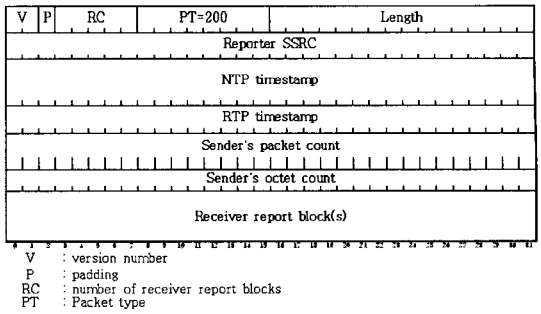


그림 3. RTCP SR 패킷 구조

(Receiver report block)의 동기화 소스 식별자를 나타낸다. 손실 비율 (Loss fraction)은 이전의 SR이나 RR 패킷이 전송된 이후 SSRC_n으로부터 손실된 RTP 데이터 패킷의 비율을 나타낸다. 손실된 전체 패킷 개수 (Cumulative number of packets lost)는 SSRC_n으로부터 패킷을 받기 시작한 이후 손실된 모든 RTP 패킷의 총수를 나타낸다. 확장된 시퀀스 넘버 (Extended highest sequence number received)는 시퀀스 넘버가 32비트 필드로, 상위 16비트는 랩 어라운드가 발생한 수를 나타내고, 하위 16비트는 SSRC_n로부터 수신된 RTP 데이터 패킷의 가장 큰 시퀀스 넘버를 나타낸다. 지터 (Interarrival jitter)는 패킷 도착 간격의 변화를 나타내는 값으로 이 값이 작을수록 패킷의 도착속도가 일정한 것이고, 이 값이 클수록 패킷이 불규칙하게 도착하는 것이다. LSR (Timestamp of last sender report received)은 가장 최근에 도착한 SR 패킷의 64비트 NTP 타임스탬프의 중앙 부분 32비트를 나타낸다. 만일 SR을 아직 받지 않은 상태라면 이 값은 0으로 설정된다. DLSR (Delay since last sender report received)는 SSRC_n으로부터 가장 최근에 받은 SR 패킷과 RR 패킷이 전송되기까지의 지연시간을 1/65536의 단위로 나타낸다. LSR와 DLSR은 라운드-트립 (Round-trip) 시간을 계산하는데 사용된다. 다음 수신 보고 블록 (Next receiver report block)은 RTCP 헤더 바로 뒤에 추가되고, 수신자 리포트 패킷과 동일한 블록 구조를 갖는다.

SR 패킷은 송수신에 대한 통계 정보를 수신자에게 알려주기 위한 패킷으로 그림 3은 SR 패킷의 구조를 보여준다. RR 패킷과 SR 패킷의 차이점은 SR 패킷만이 20바이트의 송신자 정보 섹션을 가진다는 점이다. 이는 데이터를 전송하는 경우에 사용되며, 자신이 보낸 데이터에 대한 정보를 포함한다. NTP 타임스탬프는 64비트의 필드로 SR이 보내질 때의 실제 시간을 나타내고, 1900년 1월 1일 (GMT)을 기준으로 한다. RTP 타임스탬프는 32비트의 필드로 NTP 타임스탬프에 대응하지만, RTP 미디어 시간의 단위로 표현된다. 이 값은 RTP 타임스탬프 계수의 실제 시간 간의 관계를 이용해서 해당 NTP 타임스탬프로부터 계산된다. 송신자 패킷의 수 (Sender's packet count)는 송신단에 의하여 전송이 시작된 이후부터 이 SR 패킷이 생성될 때까지 전송된 RTP 패킷의 총수를 나타낸다. 만약 SSRC가 변경되면 이 값은 0으로 설정된다. 송신자 패킷의 바이트 (Sender's octet count)는 송신자 패킷의 수와 같지만 여기에는 헤더와 패딩 바이트를 포함하지 않은 실제 보내진 정보의 총 바이트 수이다. 수신 보고 블록 (Receiver report block)은 RTCP 헤더 바로 뒤에 추가되고 수신자 리포트 패킷과 동일한 블록 구조를 갖는다.

2.2 기존의 립싱크 제어 방법

일반적으로 오디오와 비디오의 립싱크 제어는 위에서 살펴본 RTP/RTCP 패킷내의 RTP 타임스탬프와 NTP 타임스탬프를 이용한다[5-7]. 이를 이용하여 오디오 및 비디오 데이터가 캡처된 시점과 재생되는 시점의 차이 값을 각각 계산하고, 두 차이 값의 차를 비디오 또는 오디오 재생 시간에 추가하여 재생함으로써 오디오와 비디오의 립싱크 제어를 한다.

그림 4는 기존의 립싱크 제어를 위한 타이밍도이다. 립싱크를 제어하기 위해서는 각각의 캡처된 시간과 재생시간을 계산하여야 한다. RTP 패킷의 캡처

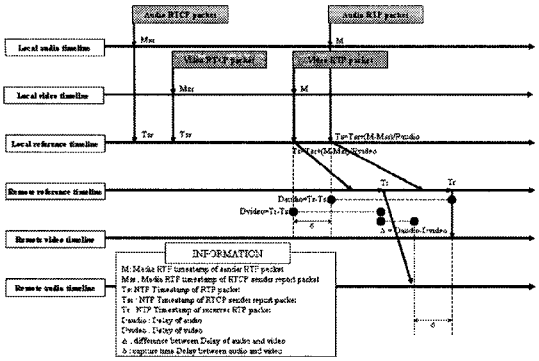


그림 4. 기존 립싱크 제어를 위한 타이밍도

된 시간은 절대적인 시스템 시간이어야 하는데, RTP 패킷만으로는 절대적인 시스템 시간을 알 수 없다. 왜냐하면 RTP 패킷에 있는 RTP 타임스탬프는 임의의 값으로 시작되어 미디어의 샘플링 비율에 따라 증가된 값이기 때문이다. 하지만 RTCP 패킷의 NTP 타임스탬프가 있으면 RTP 패킷의 RTP 타임스탬프에 대한 절대적인 시스템 시간을 알 수가 있다. 따라서 각각의 캡처된 시간은 NTP 타임스탬프에 RTP 타임스탬프를 매핑 함으로써 계산을 하는데, 이 과정은 식 (1)에 나타나 있다.

$$T_s = T_{sr} + (M - M_{sr})/Rate \quad (1)$$

T_{sr}은 RTCP 패킷에 있는 NTP 타임스탬프, M은 RTP 패킷에 있는 RTP 타임스탬프, M_{sr}은 RTCP 패킷에 있는 RTP 타임스탬프, $Rate$는 미디어의 클럭 비율, 그리고 T_s는 캡처된 시간을 나타낸다. 미디어의 지연시간 (D)은 식 (2)를 통해, 식 (1)에서 구한 캡처된 시간 (T_s)과 미디어가 재생시간(T_r)의 차이 값으로 계산된다. 여기서, 미디어의 지연시간에는 네트워크에서 생기는 지연시간과 미디어 코덱에서 발생할 지연시간이 포함되어 있다.

$$D = T_r - T_s \quad (2)$$

오디오와 비디오의 지연시간의 차이 (Δ)는 위 식에서 구한 오디오의 지연시간 (D_{audio})과 비디오의 지연시간 (D_{video})의 차이 값이 된다.

$$\Delta = D_{audio} - D_{Video} \quad (3)$$

식 (3)을 통해서 오디오와 비디오의 지연시간의 차이 (Δ)를 구할 수 있다. 이것은 오디오와 비디오의 네트워크와 시스템에서 발생한 지연시간의 차이가 된다. 여기서, Δ의 값이 음수인 경우, 오디오 재생 시간을 Δ값만큼 지연시킨 후 재생을 하고, Δ의 값이 양수인 경우에는 비디오 재생 시간을 Δ값만큼 지연시킨 후 재생함으로써, 오디오와 비디오의 립싱크 제어를 한다.

3. 제안된 립싱크 제어시스템

그림 5는 UDP/IP를 기반으로 하는 초저지연 비디오 통신시스템의 구성도를 나타낸다.

이 구성도는 본 논문에서 제안한 립싱크 제어시스템을 구성하기 위한 전체 시스템을 나타낸 것으로, 크게 캡처 모듈, RTP/RTCP 모듈, 그리고 재생 모듈로 구성되어 있다[8,9].

비디오 캡처모듈에서는 비디오 캡처 API을 이용하여 RGB형식의 QCIF (176×144) 영상을 초당 30 프레임 캡처하고, 캡처된 프레임마다 RTP 타임스탬프를 저장한다. 캡처된 프레임은 전처리 과정을 통하

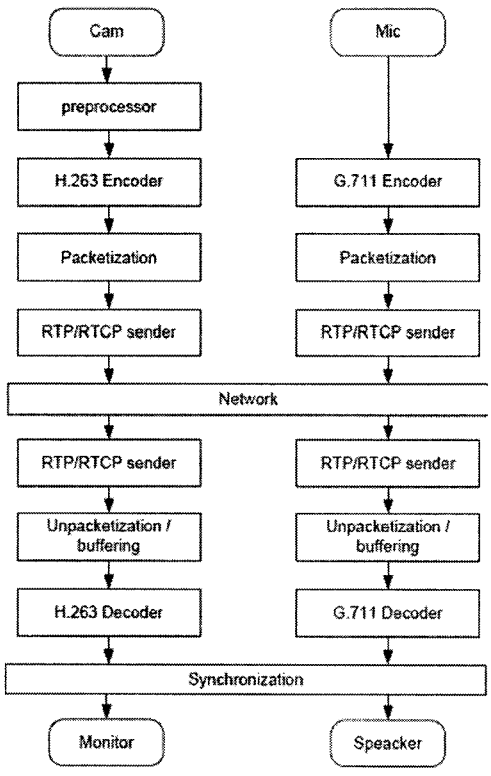


그림 5. 초저지연 비디오통신 시스템 구성도

여 RGB 형식에서 YUV (4:2:0) 형식으로 변환됨으로써, 공간영역의 중복성이 제거된다. 오디오 캡처모듈에서는 오디오 캡처 API을 이용하여 약 20 ms마다 320 byte 단위의 16비트 PCM데이터를 연속적으로 샘플링하고, G.711 음성코덱을 사용하여 데이터를 압축을 한다[10]. G.711 음성코덱은 16비트 PCM 데이터를 8비트 유럽방식 (A-law)이나 북미방식 (U-law) PCM 데이터로 변환을 하는데, 국내 및 유럽에서는 A-law방식을 사용한다. G.711 음성코덱을 사용하면 8kHz, 16비트, 모노, short형의 데이터가 8kHz, 8비트, 모노, unsigned char형으로 변환된다. 즉, 한 프레임의 16비트 320 byte PCM 데이터가 8비트 160 byte PCM 데이터로 압축이 된다.

RTP/RTCP 모듈에서는 비디오 RTP 데이터의 경우, 한 슬라이스 단위로 구성된 페이로드 데이터에 RTP 헤더를 추가하여 RTP 패킷을 구성하고, RTP 비디오 포트번호인 9996번 포트를 통하여 상대 단말기로 패킷을 전송한다. 오디오 RTP 패킷의 경우, A-law방식의 160 byte PCM 페이로드 데이터에 RTP 헤더를 추가하여 RTP 패킷을 구성하고, RTP 오디오 포트번호인 9998번 포트를 통하여 상대 단말기로 패킷을 전송한다. 여기서, 포트 번호는 임의적으로 짝수 번호를 설정한 것이기 때문에 다른 포트 번호를 설정하고자 할 경우, 사용하지 않는 짝수 포트 번호로 설정하면 된다.

RTP 패킷이 지속적으로 전송이 되는 것에 비해, RTCP 패킷은 주기적으로 전송된다. 비디오 RTCP 패킷은 RTCP 비디오 포트번호인 9997번을 통해 상대 단말기로 전송되고, 오디오 RTCP 패킷은 RTCP 오디오 포트번호인 9999번을 통해 전송된다. RTCP 패킷 역시 RTP 패킷과 마찬가지로 다른 포트 번호를 설정하고자 할 경우, 사용하지 않는 홀수 포트 번호로 설정하면 된다.

재생 모듈에서 오디오의 경우, 패킷이 수신 되면 재생시간을 계산한 후에 이 정보와 데이터를 큐에 삽입한다. 오디오는 일정하게 약 20 ms단위의 간격으로 재생되어야 하기 때문에, 이전 오디오 데이터의 재생 시간과 현재 재생될 오디오 데이터의 재생 시간 간의 차가 20 ms 이상이면, 현재 재생될 오디오 데이터를 버리고 이전 오디오 데이터를 재생한다. 재생 시간의 차가 20 ms 이하이면, 현재 재생될 오디오 데이터는 다음에 재생될 데이터와 함께 재생된다. 이렇게 일정한 간격으로 오디오 데이터를 재생할 때, 8비트 PCM 데이터를 16비트 PCM 데이터로 변환하는 역양자화 과정을 거친 후 데이터를 재생한다. 비디오의 경우, 한 프레임이 여러 패킷으로 분할되어 전송되기 때문에 한 프레임이 모아지기 전까지 버퍼에 저장하고, 버퍼에 한 프레임이 모아지면 해당 프레임이 재생시간 정보와 데이터를 큐에 삽입한다[11]. 오디오 데이터가 일정한 간격으로 재생이 되는 것에 비해, 비디오 데이터는 오디오 데이터가 재생되는 시간과 가장 근접한 비디오 데이터를 RGB 형식으로 변환한 후 화면에 재생한다.

본 논문에서는 약 100 ms의 초저지연 비디오 통신 시스템을 위해, 송신단의 인코더 버퍼를 제거하여 버퍼링에 의한 지연시간을 줄였고, 수신단에서 재배열 버퍼와 립싱크 버퍼의 크기를 3 프레임의 크기로 설정하여 지연시간을 최소로 하였다. 송신단에서는 패킷이 생성이 되는 즉시 패킷을 상대 단말기에 전송을 하기 때문에 인코더 버퍼를 하지 않아도 비디오 통신에 크게 영향을 주지 않는다. 하지만, 수신단에서는 네트워크로 전송된 패킷의 순서가 변경되었거나 잘못 수신된 경우, 이를 보정하기 위한 재배열 버퍼와 비디오와 오디오의 동기화를 위한 립싱크 버퍼가 필요하다. 초저지연의 비디오 통신시스템을 구현하기 위해 이들 버퍼의 크기를 3 프레임의 크기로 설정하였다.

그림 6은 본 논문에서 제안된 립싱크 제어 타이밍도이다. 재생시간은 송신자와 수신자간에 생기는 시스템 시간의 차를 보정하는 매핑 오프셋 (Mapping offset)과 네트워크에서 생기는 지터 (Jitter)를 이용해서 구할 수 있다. 송신자의 NTP 타임스탬프와 수신자의 NTP 타임스탬프는 서로 다른 절대적인 시스템 시간을 사용하게 된다. 이렇게 되면 지연시간

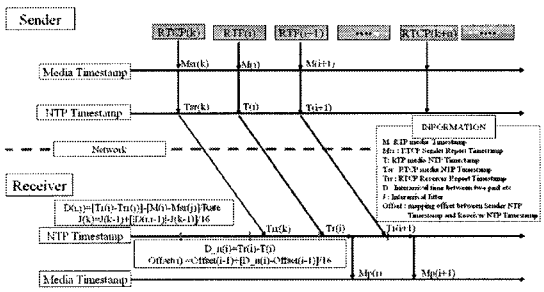


그림 6. 제안된 립싱크 제어 타이밍도

이 얼마나 발생하는지 정확히 판단할 수 없으므로, 각 절대적인 시스템 시간에 따른 차이를 보정해주어야 한다[12]. 매핑 오프셋 값 (*Offset*)은 식 (4)의 계산 방법을 통하여 얻을 수 있다.

$$D_n(i) = T_r(i) - T(i)$$
$$Offset(i) = Offset(i-1) + [D_n(i) - Offset(i-1)]/16 \quad (4)$$

여기서, T는 RTP 패킷이 캡처된 시간, T_r는 RTP 패킷이 수신된 시간, D_n은 캡처된 시간과 수신된 시간간의 차이, 그리고 *Offset*은 매핑 오프셋을 나타낸다. 매핑 오프셋이 급격하게 증가하거나 급격하게 감소하는 것을 방지하기 위해서, 캡처된 시간과 수신된 시간간의 차이와 이전 오프셋 값 간의 차를 16으로 나눈 값에, 이전에 구한 오프셋 값을 더한 결과를 매핑 오프셋 값으로 설정한다. 매핑 오프셋 값은 RTP 패킷이 수신 될 때마다 계산되고 이를 보정한다.

비디오 통신시스템에서 RTP 패킷은 네트워크를 통해 전송되므로 네트워크에 의한 지터가 발생하게 된다[13]. 지터 (J)는 식 (5)으로 정의될 수 있다[2].

$$D(i,j) = [T_r(i) - T_r(j)] - [T(i) - T(j)] \quad (5)$$
$$J(k) = J(k-1) + [|D(i,i-1)| - J(k-1))]/16$$

여기서, Tr는 RTP 패킷이 수신된 시간, D는 캡처된 시간과 수신된 시간간의 차이, 그리고 J는 네트워크의 지터를 나타낸다. 지터는 매핑 오프셋과 같은 방법으로, 급격하게 감소하거나 증가하는 것을 방지하도록 지터 값을 설정한다.

위에서 구한 매핑 오프셋과 지터, 그리고 식 (1)을 통해 구한 캡처된 시간을 이용하여 식 (6)을 통해 재생 시간 (T_r)을 계산할 수 있다.

$$T_r(i) = T_{sr}(k) + D_s + J(k) + Offset \quad (6)$$

여기서, T_{sr}은 캡처된 시간, D_s는 시스템 지연시간, J는 지터, *Offset*은 송신자의 NTP 타임스탬프와 수신자의 NTP 타임스탬프를 보정하기 위한 값을 나타낸다.

그림 7은 립싱크 제어하는 순서도를 보여준다. 각각 RTP 패킷은 캡처된 시간, 오프셋, 그리고 지터를 이용하여 재생시간을 계산한다. 수신된 RTP 데이터는 계산된 재생시간과 함께 각각의 큐에 저장이 된다. 오디오는 일정하게 재생 간격을 유지하면서 재생이 되어야 음질이 저하되지 않는다. 따라서 일정한 간격을 유지하기 위해서는 오디오 큐에서 노드를 가져올 때, 재생시간과 현재시간의 차이가 -18 ms ~ +18 ms 이내에 도착한 노드의 경우는 오디오를 재생하고, 재생시간이 현재시간보다 +18 ms 차이가 발생한 노드의 경우는 노드를 버린다. 그리고 재생시간이 현재시간보다 -18 ms 차이가 발생한 노드는 오디오 큐에 다시 저장한다. 오디오는 일정한 간격으로 재생하면서 비디오에게 이벤트 메시지와 오디오의 재생된 시간을 전달해 준다. 비디오는 오디오에서

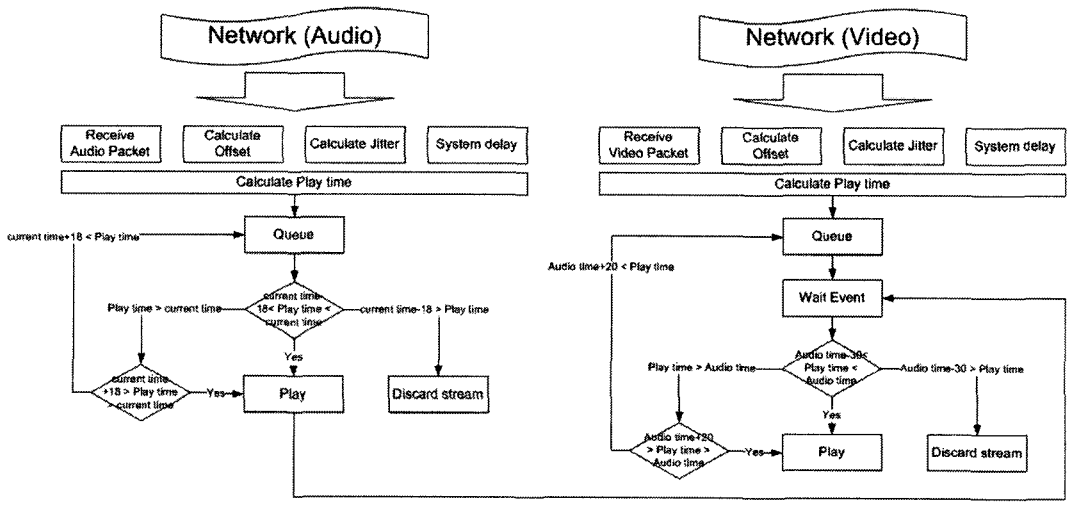


그림 7. Playout 계산과정

보내진 오디오의 재생시간을 이용하여 비디오 큐에서 오디오의 재생시간과 가장 근접한 노드를 찾아 재생한다. 오디오의 재생시간과 가장 근접한 노드를 찾기 위해서는 비디오 큐에서 노드를 가져올 때, 오디오의 재생시간과 비디오의 재생시간의 차이가 -30 ms와 +20 ms 사이에 있는 경우는 비디오를 재생하고, 오디오의 재생시간과 비디오의 재생시간이 -30ms 이하의 차이가 나는 경우에는 노드를 버린다. 그리고 오디오의 재생시간과 비디오의 재생시간이 +20ms 이상의 차이가 나는 경우에는 비디오 큐에 다시 저장한다. 위와 같은 방법으로 오디오와 비디오의 재생시간을 반복적으로 스케줄링 함으로써, 오디오와 비디오의 립싱크 제어를 한다.

4. 실험결과

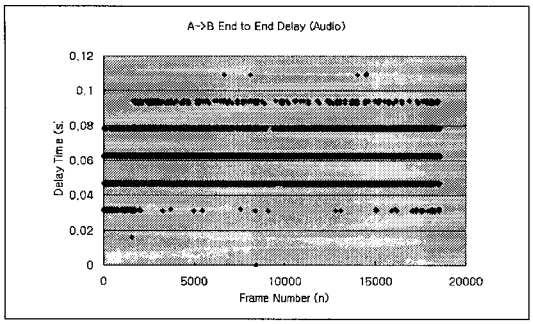
본 실험에서는 립싱크가 일치하는 가를 알아보기 위하여 오디오와 비디오의 종단간 평균 지연시간을 분석하고, 캡처된 시간과 재생된 시간의 차이를 비교하였다. 본 실험에서는 데이터의 송·수신을 위해 두 대의 컴퓨터를 이용하였고, 각각의 컴퓨터 시스템의 사양은 다음과 같다.

- 컴퓨터 A :
  - Intel(r) Core(TM)2 CPU 6400 2.13GHz/RAM 2.00GB
  - NAVIDIA GeForce 7600 GS
  - SoundMAX Integrated Digital HD Audio
  - 오디오 드라이브 버전 (5.10.1.4530)
- 컴퓨터 B :
  - Intel(R) Pentium(R) 4 CPU 3.00GHz/RAM 1.00GB
  - NVIDIA GeForce 6600
  - C-Media High Definition Audio Device
  - 오디오 드라이브 버전 (5.12.1.8)

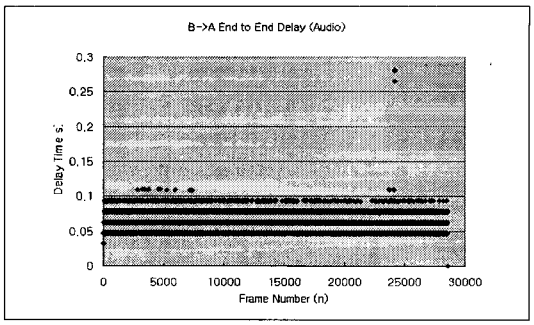
컴퓨터 A의 네트워크 상태는 평균 다운로드 속도가 44.1 Mbps, 평균 업로드 속도가 68.2 Mbps이다. 컴퓨터 B의 네트워크 상태는 평균 다운로드 속도가 60.5 Mbps, 평균 업로드 속도는 58.9 Mbps이다. 네트워크 상태의 확인은 한국정보사회진흥원 품질 측정 시스템 (http://speed.nia.or.kr)을 이용하였다. RTCP SR과 RR 패킷은 5초 간격으로 상대 단말기에 전송이 되고, 본 실험에서는 비디오의 코덱을 사용하지 않았기 때문에 비디오의 코덱에 의해 발생할 수 있는 지연시간 즉, 시스템에 의한 지연시간을 20 ms로 가정하였다. A->B로 표기한 것은 컴퓨터 A에서 컴퓨터 B로의 데이터 전송을, B->A는 컴퓨터 B에서 컴퓨터 A로의 전송을 나타낸다.

그림 8은 기존의 립싱크 제어 방법을 이용한 종단간 지연시간을 나타내는 그래프이다. 종단간 지연시간을 측정하는 방법은 RTP 패킷마다 캡처부터 전송까지의 지연시간과 수신부터 재생까지의 지연시간을 계산하고, 송신단의 시퀀스 넘버와 수신단의 시퀀스 넘버가 일치하는 것을 찾은 후, 두 지연시간의 합으로 구한다. 그림 8은 종단간 지연시간을 측정하는 방법을 이용하여 RTP 패킷마다 캡처된 시간부터 재생되기까지의 종단간 지연시간을 그래프로 나타낸 것이다. 네트워크에서 발생하는 지연시간은 가변적인 외부요인이므로 종단간 지연시간에 고려되지 않았다. 오디오의 경우 하나의 패킷단위로 지연시간을 측정하였고, 비디오의 경우 한 프레임이 여러 개의 패킷으로 분리되어 전송되므로, 한 프레임에 대한 종단간 지연시간을 측정하였다[14]. 그림 8의 (a)와 (b)에서 보듯이, 오디오의 평균 지연시간은 약 60ms이고, 그림 8의 (c)와 (d)에서 보듯이 비디오의 평균 지연시간은 약 30ms이다. 기존의 립싱크 제어 방법은 식 (2)와 식 (3)을 통해 오디오와 비디오의 지연시간의 차이를 계산하고, 그 계산된 시간만큼 오디오 또는 비디오를 지연시켜 제어한다. 따라서 오디오의 지연이 발생되면 오디오의 재생시간을 일정한 간격으로 유지하지 못하여 오디오 음질이 저하된다.

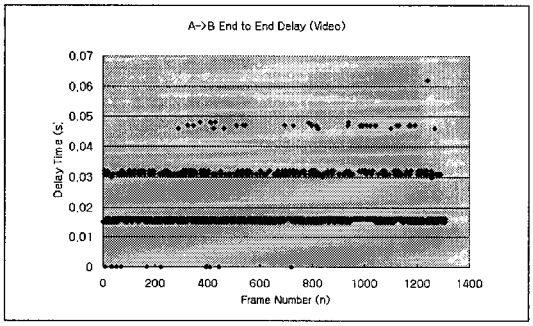
그림 9는 제안한 립싱크 제어의 종단간 지연시간을 나타내는 그래프이다. 종단간 지연시간은 그림 8과 같은 방법으로 측정하였다. 그림 9의 (a)와 (b)에서 보듯이, 오디오의 평균 지연시간은 약 60 ms이고, 그림 9의 (c)와 (d)에서 보듯이 비디오의 평균 지연시간은 약 30 ms이다. 송신단의 패킷 전송은 오디오와 비디오 캡처 API을 이용하여 캡처된 데이터를 인코딩하고, 이 데이터에 RTP 헤더를 추가하여 RTP 패킷으로 구성한 후, UDP을 통하여 수신단으로 전송한다. 따라서 송신단에서는 인코딩 버퍼와 같이 버퍼링 과정을 처리하는 부분이 없으므로, 지연시간이 발생하지 않는다. 하지만, 수신단에서는 패킷의 순서를 보정하는 재배열 버퍼와 오디오와



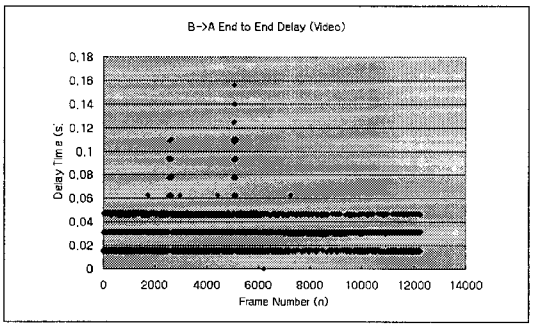
(a) 오디오 지연시간 (A->B)



(b) 오디오 지연시간 (B->A)

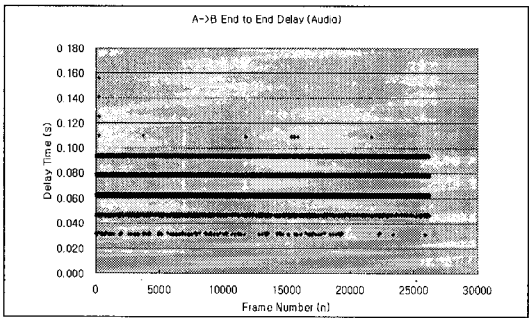


(c) 비디오 지연시간 (A->B)

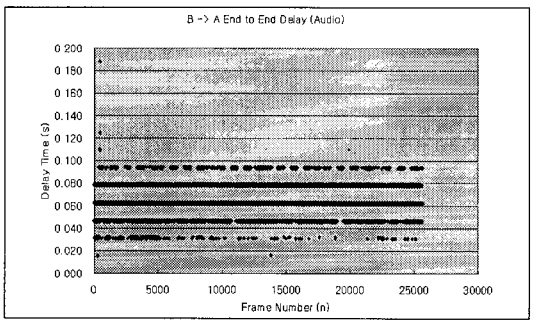


(d) 비디오 지연시간 (B->A)

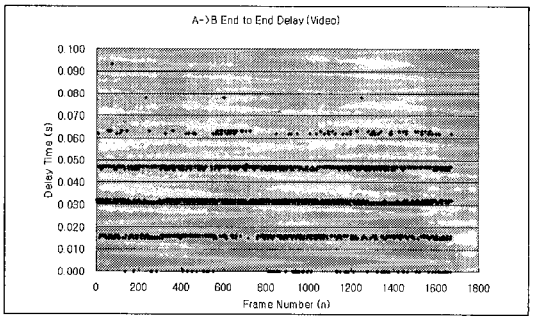
그림 8. 기존의 립싱크 제어 방법을 이용한 종단간 지연시간



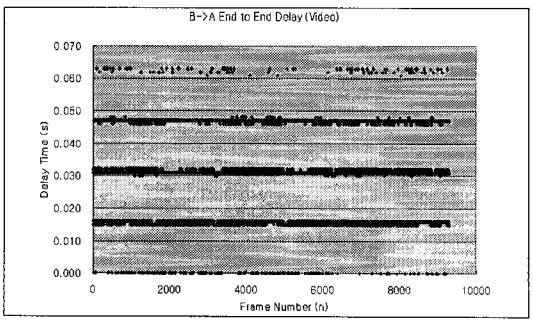
(a) 오디오 지연시간 (A->B)



(b) 오디오 지연시간 (B->A)



(c) 비디오 지연시간 (A->B)

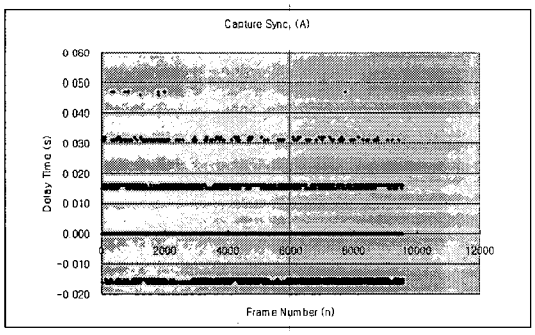


(d) 비디오 지연시간 (B->A)

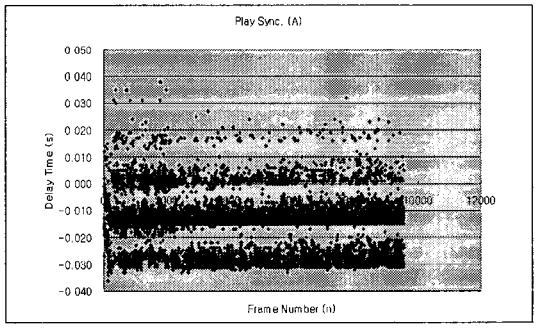
그림 9. 제안한 립싱크 제어의 종단간 지연시간

비디오의 동기화를 위한 립싱크 버퍼가 필요하기 때문에 지연시간이 발생한다. 오디오의 평균 지연시간은 립싱크 제어를 위해 일정한 패킷단위로 버퍼링을 하는 과정에서 발생하는 지연을 의미한다. 즉, 약 20 ms의 지연을 가지는 3개의 패킷을 버퍼링하므로, 약 60 ms의 평균 지연시간을 가지게 된다. 본 실험에서는 오디오 데이터를 +15 ~ -15 ms로 일정하게 재생하도록 하였고, 오디오와 비디오의 재생시간의 차가 -30 ~ 20 ms 일 때 재생하도록 설정하였다. 그러므로 비디오의 평균 지연시간은 약 30 ms가 된다. 여기서, 비디오의 평균 지연시간은 시스템에 의한 지연시간 (20 ms)을 강제적으로 더해준 것이기 때문에, 실질적으로 약 10 ms의 지연시간이 발생한 것이다. 그러나, 비디오 통신시스템은 네트워크의 지연시간과 비디오 코덱에 의한 지연시간에 따라 종단간 지연시간이 달라질 수 있다. 제안한 방법은 오디오의 재생시간을 일정한 간격을 유지하면서 립싱크 제어를 하기 때문에 기존의 방법에 비해서 음질이 향상된다.

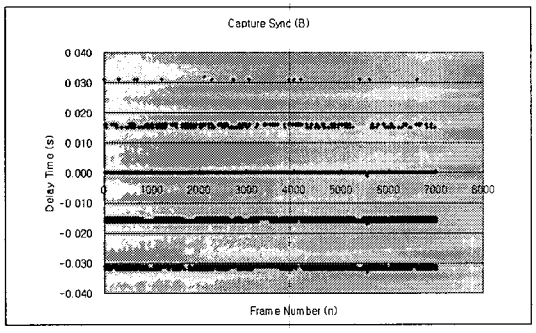
그림 10은 오디오와 비디오 각각의 캡처된 시간과 재생된 시간의 차이를 비교한 그래프이다. 그림 10의 (a)와 (c)는 오디오의 RTP 패킷에 있는 시퀀스 넘버와 비디오의 RTP 패킷에 있는 시퀀스 넘버가 일치하는 패킷을 찾은 후, 오디오와 비디오의 캡처된 시간의 차이에 대한 분포를 나타낸 그래프이고, 그림 10의 (b)와 (d)는 수신단에서 비디오의 RTP 시퀀스 넘버를 기준으로 오디오의 RTP 시퀀스 넘버와 일치하는 패킷을 찾아, 이 두 패킷이 재생된 시간의 차이에 대한 분포를 나타낸 그래프이다. 여기서, 오디오와 비디오의 캡처시간은 식 (1)을 통해, 오디오와 비디오의 재생시간은 식 (6)을 통해 구할 수 있다. 그림 10에서 오디오와 비디오가 캡처된 시간의 차이와 재생된 시간의 차이를 비교해 보면, 캡처된 시간의 차이와 재생된 시간의 차이가 거의 비슷하다는 것을 확인할 수 있다. 여기서, 오디오와 비디오의 캡처된 시간 차이가 일정한데 비해, 재생시간의 차이는 분산되어 있는 것을 볼 수 있다. 이는 지터 및 매핑오프셋 값을 보정함으로써 생기는 현



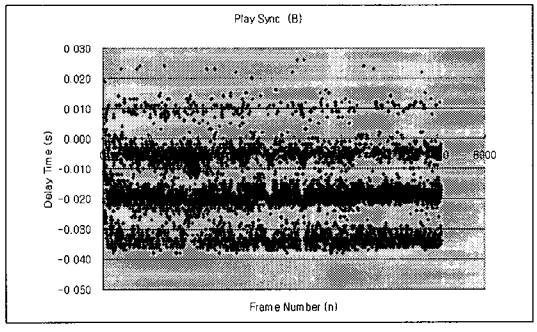
(a)오디오와 비디오의 캡처시간의 차이 (A)



(b)오디오와 비디오의 재생시간의 차이 (A)



(c)오디오와 비디오의 캡처시간의 차이 (B)



(d)오디오와 비디오의 재생시간의 차이 (B)

그림 10. 제안한 립싱크 제어시간

상이다. 하지만, 여기서 발생한 재생시간의 차이는 100 ms이하이기 때문에 사람이 이를 식별하지 못한다. 기존의 립싱크 제어 방법은, 오디오 및 비디오 데이터가 캡처된 시점과 재생되는 시점의 차이 값을 각각 계산하고, 이 두 차이 값을 이용하여 오디오와 비디오 사이의 지연시간을 계산한 후, 이 지연시간을 비디오 또는 오디오 재생 시간에 추가하여 재생함으로써 오디오와 비디오의 립싱크 제어를 한다. 이 같은 방법은 미디어의 재생 시간에 추가적인 지연이 발생시키므로 음질 및 화질이 떨어지게 된다. 이런 문제점을 해결하고자 제안한 시스템에서는 오디오의 재생시간을 일정하게 하고, 이 오디오 시간에 가장 근접한 비디오의 재생시간을 찾아서 재생함으로써 추가적인 지연이 발생하지 않고, 음질 및 화질에도 영향을 주지 않는다.

그림 11은 오디오와 비디오에 대한 립싱크 제어가 일치하는 것을 파악하기 위해, 그림 10에서 나타낸 캡처시간과 재생시간의 일부분을 확대한 그래프이다. 송신단에서 오디오와 비디오의 캡처시간의 차이만큼 수신단에서 오디오와 비디오의 재생시간의 차이를 유지하면, 립싱크가 이루어진다. 그림 11을 통해 오디오와 비디오의 캡처된 시간 차이의 간격과 오디오와 비디오의 재생된 시간 차이의 간격이 거의 일치하는 것을 확인할 수 있다. 따라서 본 논문에서 제안한 시스템은 초저지연 양방향 통신 시스템의 목표인 100 ms이하의 지연시간을 달성함과, 동시에 오디오 재생시간에 비디오 재생시간을 맞추어 재생함으로써, 오디오와 비디오의 립싱크 제어를 할 수 있다.

5. 결 론

본 논문에서는 초저지연 달성을 위한 립싱크 제어 시스템에 관한 새로운 방법을 제안하였다. 오디오와 비디오가 캡쳐된 시간의 차이만큼 수신된 데이터가 재생되도록 제어하는 기존의 립싱크 제어방법은 추가적인 지연시간이 발생하고, 음질 및 화질이 떨어지는 문제점을 가지고 있다. 이러한 문제점을 해결하고자 본 논문에서는 오디오의 재생시간을 일정하도록 제어하고, 오디오의 재생시간에 가장 근접한 비디오의 재생시간을 찾아 비디오 데이터를 재생함으로써 오디오와 비디오의 립싱크 제어를 하였다. 대역폭이 넓은 네트워크 상태에서 제안한 립싱크 제어 방법을 실험한 결과, 평균 60 ms의 종단간 지연을 유지하면서, 오디오와 비디오의 재생시간의 차이가 100 ms이하에서 재생되는 것을 확인할 수 있었다. 100 ms는 일반적으로 사람이 오디오와 비디오 사이의 시간적인 오차를 구분할 수 없을 정도의 차이로 주관적인 테스트를 통해 이를 확인할 수 있었다. 앞으로 네트워크상에서 패킷의 손실이 일어났을 경우 이를 효과적으로 보정하여 시스템의 QoS를 향상시키는 연구가 진행될 예정이다.

참 고 문 헌

[1] R. Ramjee, J. Kurose, D. Towsley, and H. Schulzrinne, "Adaptive Playout Mechanisms for Packetized Audio Applications in Wide Area Networks," *INFOCOM '94. Network-*

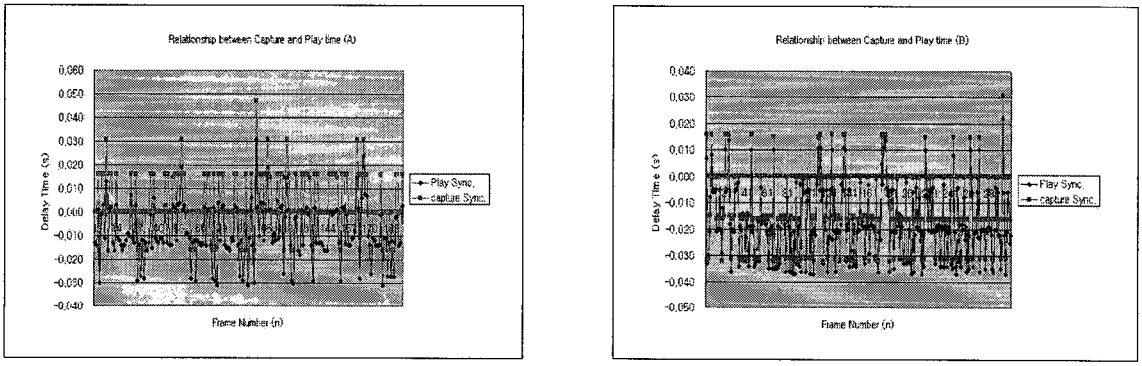


그림 11. 제안한 립싱크 제어의 캡처시간과 재생시간의 관계

ing for Global Communications. 13th Proceedings IEEE, Vol. 2, pp. 680-688, June 1994.

[2] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications," *Internet Engineering Task Force*, RFC1889, Jan. 1996.

[3] H. Schulzrinne, "RTP Profile for Audio and Video Conferences with Minimal Control," *Internet Engineering Task Force*, RFC 3551, July 2003.

[4] C. Bormann, L. Cline, G. Deisher, T. Gardos, C. Maciocco, D.Newell, J. Ott, G. Sullivan, S. Wenger, and C. Zhu, "RTP Payload Format for the 1998 Version of ITU-T Rec. H.263 Video (H.263+)," *Internet Engineering Task Force*, RFC 2429, Oct. 1998.

[5] I. Kouvelas, V. Hardman, and A. Watson, "Lip synchronization for use over the Internet: analysis and implementation," *IEEE Global Telecommunications Conference*, Vol. 2, pp. 893-898, Nov. 1996.

[6] K. H. Lee, D. H. Kim, M. G. Kang, K. H. Han, S. M. Park, and S. H. Kung, "An implementation of control protocol for multipoint audio-video teleconferencing systems," *Information Networking, 1998. (ICOIN-12) Proceedings., Twelfth International Conference*, pp. 38-41, Jan. 1998.

[7] 김찬우, 서광덕, "휴대단말기의 비디오 오디오 동기 장치 및 방법," 대한민국, 출원번호 2004-0052619, 출원일자 2004. 7. 7.

[8] 김찬우, 박성준, 서광덕, "화상 전화 단말기에서의 효율적인 오디오/비디오 동기화 방법," 한국컴퓨터종합학술대회, Vol. 32, No. 1(A), pp. 355-357, 2005.

[9] H. Jinzenji and K. Hagishima, "Real-time audio and video broadcasting of IEEE GLOBECOM'96 over the Internet using new software," *IEEE Communications Magazine*, Vol. 27, pp. 34-38, Apr. 1997.

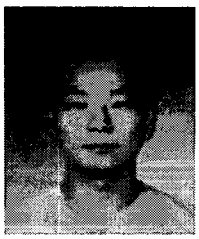
[10] ITU-T "PULSE CODE MODULATION (PCM) OF VOICE FREQUENCIES," Recommendation G.711, Nov. 1988.

[11] M. Narbutt and L. Murphy, "Adaptive Playout Buffering for Audio/Video transmission over the Internet," *Proceedings of the 17th IEE UK Teletraffic Symposium*, Vol. 27, pp. 1-6, May 2001.

[12] Orion Hodson, Colin Perkins, and Vicky Hardman, "Skew Detection and Compensation for Internet Audio Applications," *Proceedings of the IEEE International Conference on Multimedia and Expo*, Vol. 3, pp. 1687-1690, July 2000.

[13] M. Narbutt and L. Murphy, "Adaptive Playout Buffering for H.323 Voice over IP applications," *Proceedings Irish Signals and Systems Conference 2001*, pp. 201-206, June 2001.

[14] J.-C. Bolot, "End-to-end packet delay and loss behavior in the Internet," *Computer Comm. Review*, Vol. 23, pp. 289-298, Sept. 1993.

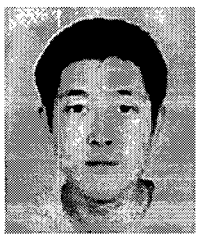


김 병 용

2006년 광운대학교 컴퓨터공학과 학사 졸업

2006년~현재 광운대학교 컴퓨터공학과 석사 과정

관심분야 : 비디오통신시스템, 화질/음질 측정

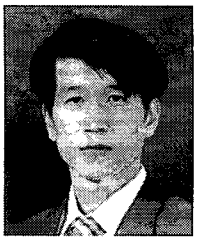


이 동 진

2006년 광운대학교 컴퓨터공학과 학사 졸업

2006년~현재 광운대학교 컴퓨터공학과 석사 과정

관심분야 : 비디오통신시스템, DMB 시스템



심 동 규

1999년 서강대학교 전자공학과 공학박사
1999년~2000년 (주)현대전자
2000년~2002년 (주)바로비젼
2002년~2005년 Univ. of Washington
2005년~현재 광운대학교 컴퓨터공학과 (조교수)
관심분야 : 영상신호처리, 영상압축, 컴퓨터 비젼

권 재 철

1986년 한양대학교 전자공학과 학사 졸업
1988년 한국과학기술원 전기및 전자공학과 석사 졸업
2003년 한국과학기술원 전기및 전자공학과 박사 졸업
1988년~현재 (주)KT 미래기술연구소 연구전문 수석연구원
주관심분야 : 멀티미디어통신, 패킷비디오, 양방향 영상통신