실시간 동영상 오버레이 멀티캐스트 시스템

준회원 강 호 종*, 정회원 송 황 준**, 준회원 민 경 원***

Real-time Overlay Video Multicast System

HoJong Kang* *Associate Member*, HwangJun Song** *Regular Member*,
KyungWon Min*** *Associate Member*

요 약

본 논문에서는 인터넷을 통한 동영상 오버레이 멀티캐스트 시스템을 제안한다. 제안한 시스템은 실시간 동영상 전송에 적합한 오버레이 멀티캐스트 트리 구성 알고리즘과 H.263+ 부호화율 제어 알고리즘을 고려한다. 오버레이 멀티캐스트 트리는 멀티캐스트 구성원들 간의 평균 시간 지연을 최소화하도록 구성하고, H.263+ 부호화율 제어 알고리즘은 시각적 성능을 향상시키기 위해 공간적, 시각적 품질을 동시에 제어한다. 두 가지 알고리즘은 유기적으로 결합하여 실제 인터넷상에서 효과적인 동영상 오버레이 멀티캐스트 시스템을 구성한다.

Key Words : Overlay multicast, Delay, H.263+, Rate control, Video streaming

ABSTRACT

This paper presents an overlay video multicast system over the Internet. The proposed system consists of two parts, i.e. overlay multicast tree suitable for the real-time video delivery and H.263+ rate control adaptive to overlay multicast tree. Overlay multicast tree is constructed to minimize the average time delay of members, and H.263+ rate control pursues a tradeoff between spatial and temporal qualities to enhance the human visual perceptual quality. Two systems are integrated and tested over the real Internet. And experimental results are provided to show the performance of the proposed system.

Ⅰ. 서 론

현재, 인터넷은 멀티미디어 통신 분야에서 중요한 역할을 수행한다. 인터넷에 접속된 호스트의 수는 급속히 증가하고 있으며, 인터넷을 통한 멀티미디어 서비스 요구 역시 매우 빠르게 증가하고 있다. 일반적으로 QoS (quality of service)와 멀티캐스트는 다양한 형태의 멀티미디어 서비스를 제공하고 네트워크의 성능을 향상시키기 위한 핵심 사항이다[1]. QoS는 IP계층에서의 도움 없이 응용계층에서 단독으로 제공할 수 없기 때문에 관련 연구가 IP계층을 중심으로 이루어지고 있다. DiffServ (differentiates services), IntServ(integrated service), RSVP(resource reservation protocol) 그리고 MPLS(multiprotocol label switching)과 같은 프로토콜들이 그러하다. IP 멀티캐스트[2]는 다수의 구성원들에게 데이터를 전송할 때 가장 효과적인 방법이다. 동일한 데이터를 구성원 수만큼 복사하지 않고 최소 요구로 중복을 줄일 수 있기 때문이다. 그러나 IP 멀티캐스트는 제어신호의 과다 발생과 라우터에서 경로

※이 논문은 정부(교육인적자원부)의 재원으로 학술진흥재단의 지원을 받아 수행됨(R08-2004-000-10084-0).
* 전자부품연구원 디지털미디어센터 연구원 (radio95@hanmail.net),
** 포항공과대학교 컴퓨터공학과 멀티미디어 통신시스템 연구실 (hwangjun@postech.ac.kr),
*** 전자부품연구원 디지털미디어센터 선임연구원 (minkw@keti.re.kr)
논문번호 : KICS2005-03-090, 접수일자 : 2005년 3월 3일

계산의 복잡도 증가로 인한 문제 때문에 현재까지 널리 사용되지 못하고 있다. 또한 완벽한 IP 멀티캐스트를 구현하기 위해서는 인터넷상에 존재하는 대부분의 라우터들이 멀티캐스트 기능을 지원하도록 수정 혹은 교체해야만 한다. 이러한 이유로 IP 멀티캐스트는 현실적으로 사용하기 어렵다.

이처럼 현실성이 부족한 IP 멀티캐스트의 대안으로, IP계층 대신 응용계층에서 멀티캐스트 기능을 구현하는 오버레이 멀티캐스트가 최근 제안되어 활발한 연구가 이루어지고 있다. 오버레이 멀티캐스트에서는 멀티캐스트에 참여하는 몇몇 호스트가 멀티캐스트 라우터의 기능을 대신한다. 현존하는 라우터에 새로운 부가기능을 추가하지 않기 때문에 오버레이 멀티캐스트는 현재 인터넷에 바로 적용할 수 있는 이점을 가지고 있다. 또한 하드웨어의 빠른 발전으로 호스트의 연산처리속도가 급속히 증가하고 있고, 네트워크 환경 역시 대용량 데이터를 초고속으로 전송할 만큼 발전하고 있어서 오버레이 멀티캐스트의 발전 가능성은 충분하다. 일반적으로 오버레이 멀티캐스트 알고리즘은 제어신호/데이터의 토폴로지의 구성 형태에 따라 그물망 구조 우선 접근법(mesh-first approach), 트리 구조 우선 접근법(tree-first approach), 그리고 함축적 접근법(implicit approach)로 분류할 수 있다[3, 4]. 그물망 구조 우선 접근법에서 구성원들은 먼저 구성원 정보에 기초하여 그물망 구조를 분산 구성한다. 이렇게 구성된 그물망 구조에서는 구성원 간에 다중 경로가 존재하게 되는데, 각 구성원은 다른 구성원에 이르는 유일 경로를 각각 계산하여 데이터를 전달한다. 이러한 구성 방법은 제어신호가 실질적으로 주고 받아야 할 데이터 신호보다 많이 발생할 수 있기 때문에 소규모 멀티캐스트 그룹에 효과적이다. Narada[1]와 Bayeux[5]가 대표적인 예이다. 트리 구조 우선 접근법에서 트리는 구성원 목록을 기반으로 잠재적 부모 노드를 무작위로 선택하여 구성한다. 그리고 각 구성원은 이웃한 다른 구성원들을 검색하고 제어링크를 구성한다. Yoid[6], ALMI[7] 그리고 HMTP[8]은 트리 구조 우선 접근법에 속한다. 마지막으로 함축적 접근법은 트리 구조와 그물망 구조를 동시에 구성한다. 이 접근법은 멀티캐스트 멤버수가 매우 클 때 적당하다. NICE[9], CAN[10] 그리고 Scribe[11]은 함축적 접근법을 사용한다.

오버레이 멀티캐스트는 앞서 언급한 것처럼 많은 장점을 갖고 있으나, 해결해야 할 몇몇 중요한 문제가 존재한다. 첫 번째로 오버레이 멀티캐스트는 IP 멀티캐스트와 동등한 수행능력을 갖지 못한다. 즉, IP 멀티캐스트와 비교해서 오버레이 멀티캐스트는 물리적 링크 상에서 중복된 트래픽을 더 많이 발생시킨다. 물론 오버레이 멀티캐스트에서 발생하는 중복된 트래픽의 양은 유니캐스트와 비교하면 상당히 적다. 두 번째로 몇몇 호스트를 경유하여 데이터를 전송하여 발생하는 시간 지연 문제이다. 이는 실시간 멀티미디어 전송에 있어서 심각한 문제이다. 양방향 대화형 통신의 경우 원활한 대화가 가능하기 위해서는 100 ms 이하의 시간 지연이 요구되며, 스트리밍 서비스에서는 최소 수 초 내의 시간 지연이 요구된다. 그리고 동영상, 오디오와 같은 시간 제약적인 미디어 서비스에서 끊기지 않고 재생하기 위해서는 지터(jitter) 역시 매우 중요하다.

Ⅱ. 제안한 동영상 오버레이 멀티캐스트 시스템

동영상은 시간적 제약을 받기 때문에 동영상 전송에 있어서 시간 지연은 무엇보다 중요하다. 일반적으로 양방향 대화형 통신에서는 100 ms 이하의 시간 지연이 요구되고, 스트리밍 서비스에서는 수 초 이내의 시간 지연이 요구된다. 시간 지연과 지터가 증가할수록 사람이 느끼는 동영상의 품질은 심각하게 저하된다. 오버레이 멀티캐스트는 목적지로 데이터를 전송하기 위해서는 몇몇 호스트를 경유하기 때문에 IP 멀티캐스트와 비교하여 시간 지연이 커진다. 따라서 오버레이 멀티캐스트를 통하여 동영상과 같은 미디어를 실시간 전송하기 위해서 어플리케이션에서 요구하는 만큼 시간 지연을 최소화할 수 있는 오버레이 멀티캐스트 트리를 구성하여야 한다. 그리고 이러한 오버레이 멀티캐스트에 적응적인 동영상 부호기가 필요하다. 즉, 네트워크 상태 정보를 이용하여 동영상 부호기는 출력 비트율, 부호화 프레임율 등을 조절할 수 있어야 한다.

일반적으로 오버레이 멀티캐스트 트리의 시간 지연은 각 호스트에서 조절할 수 있는 스트림 수와 관련된다. 그리고 스트림 수는 동영상 부호기의 출력 비트율에 따라 달라진다. 즉, 출력 비트율이 작아질수록 동영상의 품질은 저하되나 시간 지연은 줄어들고, 반대로 출력 비트율이 커질수록 동영상 품질은 향상되나 시간 지연은 증가한다. 따라서 주어진 시간 지연 조건하에서 최상의 동영상 품질을 얻기 위해서는 출력 비트율을 적절히 조절할 필요가 있다. 각 구성원의 유효 대역폭이 고정되어 있다는 가정 하에 스트림 수는 부호기의 출력 비트율을

조절하여 결정할 수 있다. 따라서 문제를 다음과 같이 수식화 할 수 있다.

문제 수식화: Determine quantization parameters q_i, $i=1,2,\cdots,m$ to minimize

$$\sum_{i=1}^{m} D_i(q_i) \quad (1)$$

subject to $$\frac{1}{N}\sum_{i=1}^{N} RTT_i(BW_{enc}) \le RTT_{\max} \quad (2)$$

satisfying $$\sum_{i=1}^{m} R_i(q_i)/T_m \le BW_{enc},$$

m은 초당 부호화되는 프레임 개수이고, $D_i(\cdot)$는 i_{th} 프레임의 왜곡이다. BW_{enc}는 고정 대역폭(양자화 매개변수에 관한 함수)이고, N은 오버레이 멀티캐스트에 참여하는 구성원의 수이다. $RTT_i(\cdot)$는 i_{th} 구성원의 RTT(round trip time) 값이고, T_m은 m 프레임의 시간 간격이다. 그리고 $RTT_{\max}$는 특정 어플리케이션에서 요구되는 최대 RTT이다. 일반적으로 식 (1)은 단조 증가 함수인 반면에 식 (2)의 좌항은 대역폭이 증가함에 따라 단조 감소한다. 오버레이 멀티캐스트 트리를 관찰, 조작하는 것은 H.263+ 부호기를 다루는 것보다 복잡한 문제이다. 따라서 위 문제를 다음과 같이 두 단계로 나누어 접근한다.

문제 1: 목표 비트율과 오버레이 멀티캐스트 트리 결정 Determine the maximum target bit rate BW_{enc} to satisfy

$$\frac{1}{N}\sum_{i=1}^{N} RTT_i(BW_{enc}) \le RTT_{\max}.$$

문제 2: 목표 비트율에 따른 H.263+ 부호화율 제어 Determine the quantization parameters q_i, $i=1,2,\cdots,m$ to minimize

$$\sum_{i=1}^{m} D_i(q_i)$$

subject to $$\sum_{i=1}^{m} R_i(q_i) \le BW_{enc} \cdot T_m.$$

2.1 고정 목표 비트율과 오버레이 멀티캐스트 트리 결정

오버레이 멀티캐스트 트리에서 평균 시간 지연은 각 구성원이 제공할 수 있는 스트림 수와 트리 구성에 따라 달라진다. 일반적으로 목표 비트율이 높으면 동영상 화질은 향상되지만 시간 지연은 늘어나게 된다. 따라서 동영상 화질과 시간 지연 사이에 적절한 균형점을 찾는 것이 필요하다. 이를 위해서 네트워크를 조작하는 것보다 동영상 부호기를 조작하는 것이 보다 쉽기 때문에 목표 비트율은 시간 지연 조건을 만족하도록 먼저 결정된다. 그리고 본 논문에서 동영상은 결정된 목표 비트율에 따라 부호화된다.

2.1.1 고정 목표 비트율 결정

오버레이 멀티캐스트 트리를 구성할 때 평균 시간 지연을 최소화하는 변형된 Dijkstra 알고리즘을 수행한다. 변형된 Dijkstra 알고리즘의 제어 변수는 각 호스트에서 조절 가능한 스트림 수이다. 그리고 이 스트림 수는 목표 비트율에 의해 결정된다. 목표 비트율 결정 과정은 다음과 같다.

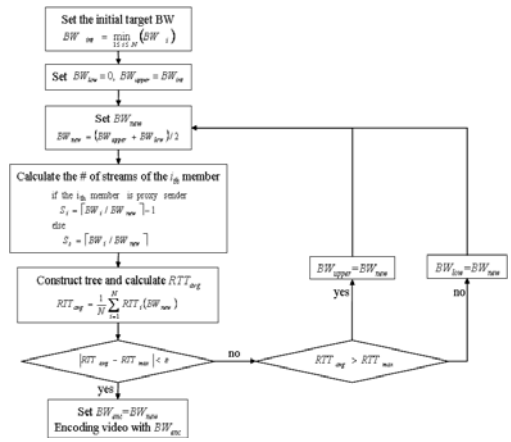


그림 1. 목표 비트율 결정 과정

Step 1: 초기 목표 대역폭은 구성원의 대역폭 중 최소 값으로 설정한다.

$$BW_{\text{int}} = \min_{1\le i\le N}(BW_i)$$

BW_{int}는 초기 목표 대역폭이고, BW_i는 i_{th} 구성원의 대역폭이다.

Step 2: BW_{low}와 BW_{upper}는 각각 0와 BW_{int}로 설정한다.

Step 3: $$BW_{new} = \left(BW_{upper} + BW_{low}\right)/2$$

Step 4: 스트림 수는 다음과 같이 구한다.

$$s_s = \left\lceil \frac{BW_i}{BW_{new}} \right\rceil, \quad s_i = \left\lceil \frac{BW_i}{BW_{new}} \right\rceil - 1$$

s_s는 센더(sender)의 스트림 수이고, s_i는 프록시 센더(proxy sender)의 스트림 수이다.

Step 5: 이전 Step 4에서 구한 스트림 수에 따라 변형된 Dijkstra 알고리즘으로 오버레이 멀티캐스트 트리를 구성한다. 오버레이 멀티캐스트 트리 구성에 관한 내용은 2.1.2에서 기술한다.

Step 6: 평균 RTT 값이 $|RTT_{avg} - RTT_{\max}| < \varepsilon$ (ε : 상수) 일 경우 $BW_{enc} = BW_{new}$로 설정하고 목표 비트율 결정 과정을 완료한다. 그리고 위 조건을 만족하지 않은 경우, 다음 조건에 따라 새로운 목표 비트율 결정 과정을 수행한다. 만약 $RTT_{avg} > RTT_{\max}$ 라면 $BW_{upper} = BW_{new}$로 설정한 후 Step 3로 돌아가고, 그렇지 않은 경우 $BW_{low} = BW_{new}$로 설정하고 Step3로 돌아간다.

2.1.2 변형된 Dijkstra 알고리즘

본 논문에서 오버레이 멀티캐스트는 다음과 같은 가정 하에 구성된다. 오버레이 멀티캐스트에 참여하는 호스트는 RTT 값에 따라 몇 개의 클러스터로 나누어지고, 클러스터 내의 호스트 중 하나는 멀티캐스트 라우터 기능을 담당하는 프록시 센더로 선택된다. 클러스터 내의 다른 호스트들은 프록시 센더로부터 데이터를 받는다. 즉, 센더에서 부호화된 동영상 데이터는 먼저 각 클러스터의 프록시 센더로 전송되고, 프록시 센더는 이 동영상 데이터를 다시 클러스터 내의 다른 호스트들로 재전송하게 된다. 이러한 가정 하에서 프록시 센더의 RTT 값은 다음과 같이 계산된다.

$$RTT^s_{p_i} = \begin{cases} RTT^s_{p_i} & : \text{프록시센더가 소스와 직접 연결된 경우} \\ RTT^s_{p_1} + RTT^{p_1}_{p_2} + \ldots + RTT^{p_{k-1}}_{p_k} + RTT^{p_k}_{p_i} & : \text{그 외의경우} \end{cases}$$

$RTT^s_{p_i}$는 센더와 프록시 센더 p_i 사이의 RTT 값이고, $RTT^{p_{k-1}}_{p_k}$는 두 프록시 센더 p_{k-1}와 p_k 사이의 RTT 값이다. 따라서 전체 시간 지연은 다음과 같이 구할 수 있다.

$$\sum_{i=1}^{n_p} m_i \cdot RTT^s_{p_i},$$

n_p는 클러스터의 개수이고, m_i는 i_{th} 클러스터 내의 호스트 개수이다. 그리고 i_{th} 구성원이 제공할 수 있는 스트림 수는 다음과 같다.

$$ST_i = \left\lceil \frac{BW_i}{BW_{enc}} \right\rceil - 1,$$

BW_i는 i_{th} 호스트의 유효 채널 대역폭이고, $\lceil x \rceil$는 x보다 작은 가장 큰 정수를 의미한다. 특정 호스트에 트래픽이 집중되지 않도록 스트림 수를 고려하고 평균 RTT를 최소화하는 오버레이 멀티캐스트 트리 구성을 위한 문제 접근은 다음과 같다.

문제 수식화 : Determine the number of clusters and trees to minimize

$$\frac{1}{N}\sum_{i=1}^{n_p} m_i \cdot RTT^s_{p_i} + \frac{1}{N}\sum_{i=1}^{n_p}\sum_{j=1}^{m_i-1} RTT^{p_i}_{j} \quad (3)$$

Subject to $s_s \le ST_s$ and $s_i \le ST_i$

여기서 N은 멀티캐스트에 참여하는 호스트의 개수 ($N = \sum_{i=1}^{n_p} m_i$)이고, s_s와 s_i는 각각 센더와 i_{th} 호스트의 스트림 개수이다. 그리고 ST_s와 ST_i는 각각 센더와 i_{th} 호스트에 의해 제공될 수 있는 최대 스트림 수이다. 식 (3)은 주어진 제한 조건에 따라 특정 호스트에 트래픽이 집중되는 것을 방지하고 대역폭의 효율을 높이도록 해준다. 이전 연구 [12]에서 앞서 언급한 트리 구성 최적화 문제에 대한 효과적인 방법으로 변형된 Dijkstra 알고리즘을 제안하였다. 이는 다음 그림 2의 순서도로 요약된다.

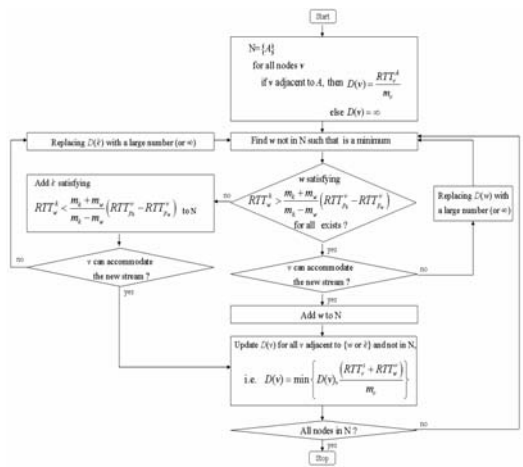


그림 2. 변형된 Dijkstra 알고리즘의 순서도

2.2 고정 대역폭에서 H.263+ 부호화율 제어 알고리즘

본 논문에서는 동영상 압축을 위해 H.263+을 사용한다. H.263+은 소프트웨어를 활용하여 동영상 코덱을 컴퓨터로 구현했을 때, 실시간 처리가 가능할 정도의 부호화/복호화 속도가 보장되기 때문이다. 부호화율 제어 알고리즘은 양자화 매개변수와 부호화 프레임율을 조절하여 출력 비트율과 동영상 품질을 제어한다. 앞서 언급했듯이 목표 비트율은 문제 접근을 수월하게 하기 위해 고정 비트율(CBR)을 사용한다. 동영상 트래픽은 장면 전환과 움직임 변화로 인해 일반적으로 가변 비트율(VBR) 특성을 갖는다. 따라서 고정 비트율 조건 하에서 동영상 품질은 저하될 것이다. 품질 저하를 최소화 하기 위해서 공간적, 시간적 품질 사이의 교환을 수행하는 부호화율 제어 알고리즘[13]을 사용한다. 부호화 프레임율을 능동적으로 제어하는 부호화율 제어 알고리즘은 갑작스런 움직임 변화와 시간에 따라 변하는 네트워크 환경에서 동영상의 끊김 현상 없이 일정한 범위 내에서 P 프레임의 화질을 유지한다.

2.2.1 프레임 계층에서의 율-왜곡 모델링

율-왜곡 모델링 기법은 보다 빠른 부호화율 제어 알고리즘 개발을 위해서 필수적으로 요구되는 방법으로 확률적 모델링 기법과 실험적 데이터베이스 기반 모델링 기법이 있다. 본 논문에서는 2차 부호화율 모델을 사용한 실험적 데이터베이스 기반 프레임 계층 율-왜곡 모델링 기법을 사용한다. 그리고 한 프레임 내에서 평균 양자화 매개변수 값에 대한 유사 왜곡 모델을 사용한다. 이는 다음 식 (4)와 같이 주어진다.

$$\begin{aligned} \mathcal{R}(\bar{q}_i) &= (a\,\bar{q}_{i-1} + b\,\bar{q}_{i-2}) \cdot MAD(f_{req}, f_{cur}), \\ \mathcal{D}(\bar{q}_i) &= a'\,\bar{q}_i + b', \end{aligned} \tag{4}$$

a, b, a', b'는 모델 계수이고, f_{req}는 이전 시간에서 재구성된 참조 프레임이다. f_{cur}는 현재 압축되지 않은 영상이고, $MAD(\cdot,\cdot)$는 두 프레임간의 절대값 차를 의미한다. $\bar{q}_i$는 i_{th} 프레임 내에서 모든 대블록(macroblock)들의 평균 QP(quantization parameter) 값이다. 모델 계수는 아래 식 (5)와 같이 이전 부호화 결과에 의해 얻는 율-왜곡 테이블을 이용하여 결정된다.

$$\begin{aligned} b &= \frac{N\left(\sum_{i=1}^{N} R_i\right) - \left(\sum_{i=1}^{N} R_i \bar{q}_i\right)\left(\sum_{i=1}^{N} \bar{q}_i^{-1}\right)}{N\left(\sum_{i=1}^{N} \bar{q}_i^{-2}\right) - \left(\sum_{i=1}^{N} \bar{q}_i^{-1}\right)^2}, \qquad a = \frac{\sum_{i=1}^{N}\left(R_i \bar{q}_i - b\bar{q}_i^{-1}\right)}{N}, \\ a' &= \frac{\left(\sum_{i=1}^{N} D_i\right)\left(\sum_{i=1}^{N} \bar{q}_i\right) - N\left(\sum_{i=1}^{N} D_i \bar{q}_i\right)}{\left(\sum_{i=1}^{N} \bar{q}_i\right)^2 - N\left(\sum_{i=1}^{N} \bar{q}_i^2\right)}, \qquad b' = \frac{\sum_{i=1}^{N} D_i - a\sum_{i=1}^{N} \bar{q}_i}{N}, \end{aligned} \tag{5}$$

N은 율-왜곡 테이블의 크기이고, D_i와 R_i는 각각 부호화된 i_{th} 프레임의 실제 왜곡 값과 비트율이다. 만약 모델에 의해 예측된 값과 율-왜곡 테이블 자료 값의 차가 임계 값보다 크다면, 테이블의 자료는 삭제되고 위와 같은 방법으로 모델 계수를 재계산한다.

2.2.2 적응적 부호화 프레임율 제어

라그랑지 곱수는 목표 비트율에 따른 부호화율 제어의 최적 해를 얻기 위해 사용한다. 제안된 프레임 계층 부호화율 제어 알고리즘은 최적 과정과 라그랑지 곱수 적용 과정으로 구성된다.

▪ 율-왜곡 모델에 기반한 최적 과정

$$\begin{aligned} P_i(\bar{q}_i) &= \hat{D}_i(\bar{q}_i) + \lambda_i \cdot \max\{\hat{B}_i^{res}, 0\}, \\ \hat{B}_i^{res} &= \sum_{j=1}^{i-1} R_j + \hat{R}_i(\bar{q}_i) - BW_{enc} \cdot T_m, \end{aligned} \tag{6}$$

$P_i(\bar{q}_i)$와 λ_i는 각각 i_{th} 프레임에 대한 비용함수와 라그랑지 곱수이다. R_j는 j_{th} 프레임의 비트율이고, MC_k^i는 k_{th} 일시적 프레임 부분의 $(i-1)_{th}$ 프레임과 i_{th} 프레임간의 움직임 변화이다. 식 (6)에서 보듯이, 각 프레임에 대한 목표 비트량은 움직임 변화에 기반하여 결정된다. 빠른 움직임 변화를 포함하는 프레임은 느린 움직임 변화를 포함하는 프레임보다 좀 더 많은 비트량을 요구하기 때문이다. 그리고 i_{th} 프레임에 대한 최적 평균 QP 값은 다음과 같이 결정된다.

$$\bar{q}_i^{*} = \arg\min_{\bar{q}_i} P_i(\bar{q}_i). \tag{7}$$

▪ 라그랑지 곱수 적용 과정

라그랑지 곱수는 시스템 동작 중에 업데이트 되기 때문에 계산의 복잡도를 낮추기 위해 차선의 최적해만 고려한다. 따라서 본 논문에서는 라그랑지

곱수를 다음과 같이 구한다.

$$\lambda_{i+1} = \lambda_i + \Delta\lambda, \qquad \Delta\lambda = \frac{B_{used,i}}{B_{target,i}} - 1, \tag{8}$$

$$B_{used,i} = \sum_{j=1}^{i} R_j, \quad B_{target,i} = \sum_{j=1}^{i} BW_{enc} \cdot T_m$$

여기서 λ_i는 i_{th} 프레임의 라그랑지 곱수이다. 목표 비트율은 TMN8[14]의 대블록 계층 부호화율 제어 방법에 의해 한 프레임내의 대블록들에 할당된다. 화상 회의 시스템과 같은 경우 m은 짧은 대기 시간을 고려하여 1로 설정되고, T_m은 재생화면에서 급격한 변화 없이 부드럽게 동작이 이어지도록 일정한 범위 내에서 공간적 품질을 유지시켜주는 역할을 한다. 매우 작은 비트율에서 공간적, 시간적 품질 모두를 좋게 유지하는 것은 매우 어려운 일이다. 사람의 눈은 갑작스런 부호화 프레임율 변화에 매우 민감하다. 따라서 부호화 프레임율 제어는 동영상 내에서의 동작과 유효 채널 대역폭에 기반한 공간적, 시간적 품질 사이의 교환을 수행한다. 이러한 방법은 사람의 눈이 느끼는 경련적 동작과 같은 시간적 품질 저하를 감소시키는데 그 목적이 있다. 그리고 실시간 처리를 위해 부호화 시간 지연을 작게 해야 한다.전체적인 적응적 프레임율 제어 알고리즘은 다음과 같다.

Step 1: 예측 왜곡은 다음과 같이 구한다.

$$\hat{D} = a' \frac{a \cdot MAD(f_{i-1}, f_i) + \sqrt{(a \cdot MAD(f_{i-1}, f_i))^2 + 4b \cdot B(F_{i-1}) \cdot MAD(f_{i-1}, f_i)}}{2B(F_{i-1})} + b',$$

$$B(F_{i-1}) = \frac{F_{i-1}}{G} \cdot BW_{enc},$$

f_{i-1}는 이전 시간에 재구성된 참조 프레임이고, f_i는 현재 압축하지 않은 영상이다. F_i는 G fps로 카메라 촬영을 했다는 가정하에 현재 부호화 프레임 간격이고, R_i는 현재 유효 채널 대역폭이다.

Step 2: 부호화 프레임 간격 결정을 위해, 먼저 임계값을 다음과 같이 계산한다.

$$TH_{d_1} = (1 + c) \cdot D_{avg},$$
$$TH_{d_2} = (1 - c/2) \cdot D_{avg},$$

D_{avg}는 이전 부호화된 5개 프레임의 평균 왜곡이고, c는 상수이다. c가 큰 값을 갖는 경우 부호화 프레임율 제어는 동작하지 않고 대블록 계층 비트율 제어만 동작한다. 부호화 프레임 간격은 아래 식 (9)와 같이 구한다.

$$\triangle F_{i-1} = \lceil 0.3 \cdot F_{i-1} \rceil \cdot \tag{9}$$

그리고 목표 비트율은 다음과 같이 업데이트한다.

$$\tilde{B}_i = \begin{cases} (F_{i-1} + \Delta F_{i-1}) \cdot BW_{enc} / G & \text{if } \hat{D} > TH_{d1}, \\ (F_{i-1} - \Delta F_{i-1}) \cdot BW_{enc} / G & \text{if } \hat{D} < TH_{d2}, \\ F_{i-1} \cdot BW_{enc} / G & \text{otherwise}, \end{cases}$$

Step 3: 목표 비트율 $\tilde{B}_i$로 대블록 계층 비트율 제어를 수행하고, $\overline{QP}_i$, B_i, D_i 값을 반환한다.

Step 4: 율-왜곡 모델의 데이터 베이스와 계수를 업데이트하고, Step 1으로 돌아간다.

Ⅲ. 실험결과

그림 3은 제안한 동영상 오버레이 멀티캐스트 시스템의 개괄적인 블록 다이어그램이다. 서버 어플리케이션에서 USB 캠을 통해 촬영한 영상을 앞서 설명한 부호화율 제어 알고리즘이 포함된 H.263+로 부호화하여 각 호스트로 전송한다. 호스트의 클라이언트 어플리케이션에서는 수신된 동영상 데이터를 복호화하여 재생하고, 호스트가 프록시 센더로 동작할 경우 클러스터 내의 다른 호스트로 수신된 데이터를 재전송한다.

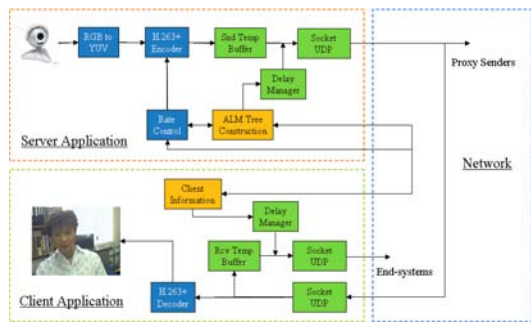


그림 3. 제안한 동영상 오버레이 멀티캐스트 시스템

인터넷을 통한 실시간 동영상 전송 실험을 위해 12대의 컴퓨터를 이용하여 오버레이 멀티캐스트를 구성하였다. 각 호스트간의 RTT 값은 표 1과 같다. 그리고 지연 관리자(delay manager)는 표 1에 근거

표 1. 각 호스트간 RTT 값

	Sender	1	2	3	4	5	6	7	8	9	10	11
Sender		91	101	62	65	95	96	106	106	66	67	70
1	91		57	152	120	7	5	59	64	157	158	124
2	101	57		157	153	63	56	5	8	157	159	158
3	62	153	157		82	158	158	162	160	8	6	85
4	65	120	153	82		121	125	158	160	90	88	5
5	95	7	63	158	121		8	65	71	161	162	124
6	96	5	56	158	125	8		58	63	161	162	129
7	106	59	5	162	158	65	58		6	162	164	163
8	106	64	8	160	160	71	63	6		160	163	164
9	66	157	157	8	90	161	161	162	160		4	93
10	67	158	159	6	88	162	162	164	163	4		91
11	70	124	158	85	5	124	129	163	164	93	91	

하여 동영상 패킷 전송 시간을 결정하여 시간 지연을 조절한다. 본 실험에서 최대 유효 시간 지연은 90 ms로 설정하였다.

각 호스트들은 (1, 5, 6), (2, 7, 8), (3, 9, 10), (4, 11) 이상 4개의 클러스터를 구성하고, 각 클러스터의 1, 2, 3, 4 호스트는 프록시 센터 역할을 담당한다. 주어진 시간 지연 조건을 만족하는 목표 대역폭은 3 단계를 거쳐 결정되었다. 첫번째 단계에서 초기 목표 대역폭은 128 Kbp로 설정되었고, 각 호스트의 스트림 수는 표 2와 같다. 그리고 센더와 프록시 센더 사이의 경로 결정을 위한 Dijkstra 테이블은 표 3과 같다.

첫번째 단계에서 목표 대역폭에 따른 시간 지연 값이 주어진 최대 유효 시간 지연보다 작기 때문에 두번째 단계에서는 목표 대역폭을 192 Kbps로 증가시켰다. 두번째 단계에서의 Dijkstra 테이블은 표 4와 같다.

표 2. 호스트의 대역폭과 스트림 수

	Step 1 (128K)	Step 2 (192K)	Step 3 (160K)
Sender	**4**	2	3
1	6	4	5
2	6	4	5
3	14	9	11
4	3	1	2
5	4	2	3
6	11	7	9
7	2	1	1
8	15	10	12
9	4	2	3
10	7	5	6
11	4	5	3

표 3. 목표 대역폭이 128 Kbps 일때의 Dijkstra 테이블

Step	start N	D(1)p(1)	D(2)p(2)	D(3)p(3)	D(4)p(4)
0	S	30,S	34,S	**21,S**	33,S
1	S-3	**30,S**	34,S		33,S
2	S-3-1		34,S		**33,S**
3	S-3-1-4		**34,S**		
4	S-3-1-4-2				

표 4. 목표 대역폭이 190 Kbps 일때의 Dijkstra 테이블

Step	start N	D(1)p(1)	D(2)p(2)	D(3)p(3)	D(4)p(4)
0	S	30,S	[34,S]	**21,S**	[33,S]
1	S-3	**30,S**	73,3		72,3
2	S-3-1		**49,1**		72,3
3	S-3-1-2				**72,3**
4	S-3-1-2-4				

두번째 단계에서 구한 시간 지연은 최대 유효 시간 지연보다 큰 값을 갖기 때문에 세번째 단계에서 새로운 목표 대역폭 160 Kbps를 설정하고 시간 지연을 구하였다. 이때 Dijkstra 테이블은 표 5와 같으며, 최종적으로 구한 시간 지연은 표 6과 같이 최대 유효 시간 지연 값에 매우 근접한 89.75 ms이다. 따라서 H.263+ 부호화율 제어의 목표 대역폭은 160 Kbps가 되고, 촬영된 영상은 목표 대역폭에 맞게 부호화 되여 그림 4의 최종 오버레이 멀티캐스트 트리를 통해 각 호스트로 전송된다. 그리고 센더와 각 호스트는 그림 5와 같은 인터페이스를 통해 트리 구성 정보를 전달받고 동영상을 재생하게 된다.

표 5. 목표 대역폭이 160 Kbps 일때의 Dijkstra 테이블

Step	start N	D(1)p(1)	D(2)p(2)	D(3)p(3)	D(4)p(4)
0	S	30,S	[34,S]	**21,S**	[33,S]
1	S-3	**30,S**	73,3		33,S
2	S-3-1		49,1		**33,S**
3	S-3-1-4		**49,1**		
4	S-3-1-4-2				

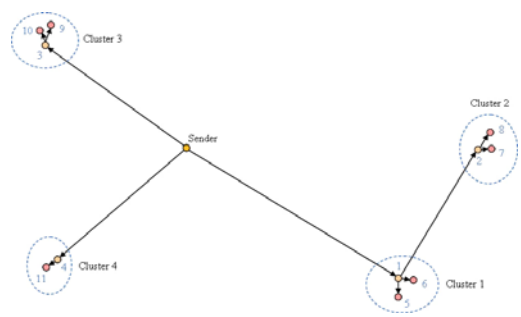


그림 4. 최종 오버레이 멀티캐스트 트리

표 6. 목표 대역폭 결정과정

	Step 1	Step 2	Step 3
target bit rate (bps)	128K	192K	**160K**
delay (ms)	78	102.92	**89.75**

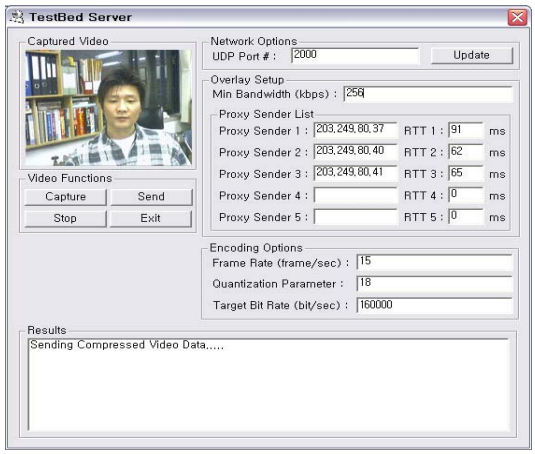


그림 5. 유저 인터페이스(센더측)

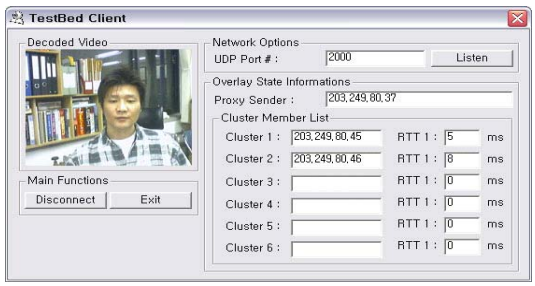


그림 6. 유저 인터페이스(호스트측)

그림 5는 제안된 알고리즘에 의해 촬영된 영상을 압축하고 오버레이 멀티캐스트 방식으로 각 호스트에 데이터를 전달하는 센터측 유저 인터페이스 이고, 그림 6은 클러스터 2의 프록시 센더인 호스트 2의 사용자 인터페이스이다. 그림에서 보듯이 센더 측에서 촬영된 영상과 호스트 측에서 복호화 되어 재생된 영상간의 화질 차이는 거의 없으며, 시간 지연 역시 최대 유효 시간 지연 내에 있기 때문에 주관적 판단으로 보았을 경우 시간 지연을 거의 느끼지 못하거나 무시할 수 있을 정도로 작음을 확인할 수 있다.

Ⅳ. 결 론

본 논문에서 오버레이 멀티캐스트 트리의 평균 시간 지연을 최소화함과 동시에 동영상의 왜곡을 최소화 하도록 하는 부호화율 제어 알고리즘을 결합하여 효율적인 동영상 오버레이 멀티캐스트 시스템을 제안하였다. 인터넷을 통한 오버레이 멀티캐스트 전송 실험에서 제안된 알고리즘에 의해 효율적인 실시간 동영상 전송이 이루어짐을 확인하였다.

인터넷 망에서 호스트는 유동적인 환경을 갖기 때문에 초기 설정 이후 자유로운 가입과 탈퇴가 보장되는 트리 유지 알고리즘으로의 확장 연구가 앞으로 필요하다. 또한 최근 활발한 연구가 진행되고 있는 H.264의 적용을 위해 실시간 전송이 가능하도록 인코딩 속도의 개선과 제안된 부호화율 제어 알고리즘의 적용이 요구되어진다.

참 고 문 헌

〔1〕 Y. Chu, S. Rao, S. Seshan and H. Zhang, "A case for end-system multicast," ACM SIGMETRICS, Santa Clara, June 2000.

〔2〕 S. Deering, "Host Extensions for IP Multicasting," RFC1112, August 1989.

〔3〕 S. Banerjee and B. Bhattacharjee, "A comparative study of application layer multicast protocol," at http://www.cs.wisc.edu/~suman/pubs.html.

〔4〕 Ayman El-Sayed, Vincent Roca and Laurent Mathyt, "A Survey of Proposals for an Alternative Group Communication Service," Network, IEEE, Volume: 17, Issue: 1, Jan. -Feb. 2003.

〔5〕 S. Zhuang et al., "Bayeux: An Architecture for Scalabel and Fult-Tolerant Wide-Area Data Dissemination," 11th Int'l. Wksp. Net. And Op. Sys. Support for Digital Audio and Video, June 2001.

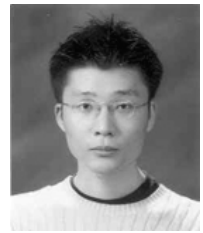
〔6〕 P. Francis, "Yoid: Extending the Multicast Internet Architecture," White paper http://www.aciri.org/yoid, 1999.

〔7〕 D. Pendarakis, D. Verma, S.Shi, M. Waldvogel, "ALMI: An Application Level Multicast Ingrastructure," Proc. Of the 3rd USNIX symposium on Internet Technologies and ystems, March 2001.

〔8〕 B. Zhang, S. Jamin, and L. Zhang, "Host multicast: A framework for delivering multicast to end users," In Proceedings of IEEE

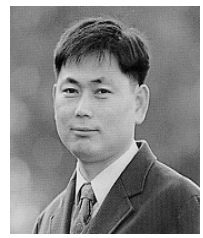
INFOCOM 2002, June 2002.
〔9〕 S. Banerjee, B. Bhattacharjee ,and C. Kommareddy, “Scalable Application Layer Multicast,” ACM SIGcomm ’02, Pittsburgh, PA, August 2002.
〔10〕 S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, “A scalable content-addressable network,” In proceedings of ACM Sigcomm, August 2001.
〔11〕 M. Castro, P. Druschel, A-M. Kermarrec, and A. Rowstron, “Scribe: A large-scale and decentralized application-level multicast infrastructure.” Selected Areas in Communications, IEEE Journal on, Volume: 20, Issue: 8, Oct. 2002.
〔12〕 Hwangjun Song and Dong Sup Lee, “Overlay multicast tree minimizing average time delay,” LNCS(Networking 2004), Vol. 3042, April 2004..
〔13〕 Hwangjun Song, Jongwon Kim and C. C. Jay Kuo, “Real-time encoding frame rate control for H.263+ video over the Internet,” Signal Processing: Image Communication, Vol. 15, Nos 1-2, pp. 127-148, Sept, 1999.
〔14〕 ITU-T, Video codec Test model, near-term, version 8(TMN8), H.263 AdHoc Group, Portland, June 1997.

강 호 종 (HoJong Kang) 준회원



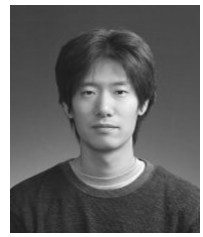
2003년 8월 홍익대학교 전파공학과(학사)
2005년 8월 홍익대학교 전파통신공학과(석사)
2005년 9월~현재 전자부품연구원 디지털미디어센터 연구원
<관심분야> 영상압축, 멀티캐스트

송 황 준 (HwangJun Song) 정회원



1990년 2월 서울대학교 제어계측공학과(학사)
1992년 2월 서울대학교 제어계측공학과(석사)
1999년 5월 Univ. of Southern California, EE-Systems(박사)
2000년~2005년 2월 홍익대학교 전자전기공학부
2005년 2월~현재 포항공과대학교 컴퓨터공학과
<관심분야> 영상통신시스템, 오버레이 멀티캐스트, Ad-Hoc

민 경 원 (KyungWon Min) 준회원



1994년 2월 홍익대학교 전자공학과(학사)
1996년 8월 홍익대학교 전자공학과(석사)
2005년 3월~현재 연세대학교 전기전자공학과(박사과정)
1996년 9월~현재 전자부품연구원 디지털미디어센터 선임연구원
<관심분야> MPEG 시스템, 3D 디지털 신호처리