가변 비트율 비디오 전송을 위한 효율적인 스무딩 알고리즘

이 면 재[†] · 이 준 용[††] · 박 도 순[†††]

요 약

스무딩은 가변 비트율의 비디오 데이터를 고정 비트율로 변환하는 전송 계획이다. 이를 위한 스무딩 알고리즘에는 CBA, MCBA, MVBA, PCRTT, e-PCRTT 등이 있으며, PCRTT 알고리즘을 개선한 e-PCRTT 알고리즘에서는 전송률 변화 횟수가 주어지고 런의 크기가 일정하다. 이는 불필요한 전송률의 변화를 필요로 하고, 또한 버퍼 크기가 작은 경우에는 QoS를 보장하지 못할 수도 있다. 본 논문에서는 이러한 e-PCRTT 알고리즘의 단점을 해결하기 위해 전송률 변화 횟수에 대한 제한이 없고, 런의 크기가 가변적인 스무딩 알고리즘을 제안한다. 제안 알고리즘의 성능은 e-PCRTT 알고리즘을 포함한 다른 알고리즘들과 전송률 변화 횟수, QoS를 유지하기 위한 버퍼 크기 등과 같은 다양한 평가 요소들로 비교하여 우수함을 보였다.

An Efficient Smoothing Algorithm for Video Transmission at Variable Bit Rate

Myoun-Jae Lee[†] · Junyong Lee[††] · Do-Soon Park[†††]

ABSTRACT

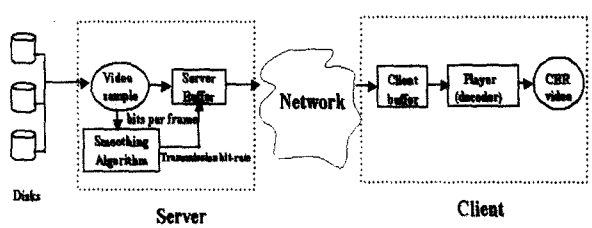
Smoothing is a transmission plan where variable rate video data is converted to a constant bit rate stream. Among them are CBA, MCBA, MVBA, e-PCRTT and others. E-PCRTT algorithm, which was improved from PCRTT, restricts the number of rate changes with fixed-size run. This causes unnecessary rate changes and may not guarantee QoS when buffer size is small. In this paper, a smoothing algorithm is proposed, where the number of rate changes are not limited and the size of run can be flexible, in order to overcome the shortcomings of e-PCRTT. Experiments demonstrated that the proposed algorithm outperformed other algorithms such as e-PCRTT. In order to show the performance, various evaluation factors were used such as the number of rate changes, buffer size to maintain QoS, and so on.

키워드 : 스무딩(Smoothing), 가변 비트율(VBR : Variable Bit Rate)

1. 서 론

비디오 데이터는 효율적인 저장 공간의 사용과 네트워크 전송을 위하여 압축되어 저장된다. 이러한 압축 방식에는 프레임을 구성하는 비트 수가 동일한 고정 비트율(CBR : Constant Bit Rate) 방법과 동일하지 않은 가변 비트율(VBR : Variable Bit Rate) 방법이 있다[1]. 특히 가변 비트율 방법으로 저장된 비디오 데이터는 프레임을 구성하는 비트 수의 차이가 심하므로 이것을 그대로 전송한다면 QoS (Quality of Service)를 보장하기 위해서 전송률을 급격히 증가시켜야 되는 버스트(Burst) 현상이 발생될 수 있다[2-5]. 스무딩 기법은 이러한 버스트 현상을 피하기 위해 가변 비트율로 저장된 비디오 데이터를 일련의 고정 비트율로 변환하는 전송 계획이다[6].

(그림 1)은 스무딩 기법을 위한 구조이다[7]. 클라이언트가 서버에게 비디오 스트림을 요청하면, 서버는 클라이언트에서 요청한 비디오 스트림을 검색하여 비디오 스트림을 구성하는 모든 프레임들의 정보, 즉 프레임 당 비트 수를 스무딩 알고리즘에게 전달한다. 그러면 스무딩 알고리즘은 프레임 당 비트 수와 클라이언트 버퍼 크기를 고려하여 클라이언트 버퍼에서 언더플로우와 오버플로우가 발생되지 않을 전송률, 즉 QoS가 보장되는 전송률을 설정하여 해당 비디오 스트림을 클라이언트에게 전송한다.



(그림 1) 스무딩 기법의 구조

† 준 회 원 : 한국산업기술대학교 교수
†† 정 회 원 : 홍익대학교 컴퓨터공학과 교수
††† 종신회원 : 홍익대학교 컴퓨터공학과 교수
논문접수 : 2004년 6월 25일, 심사완료 : 2004년 9월 10일

(그림 2)는 스무딩 기법의 원리[8, 9]를 나타내는데, X축은 시간 즉 프레임 번호이며 Y축은 바이트 수이다. V(t)는 언더플로우 경계선으로 프레임들의 누적된 바이트 수이고 식 (1)과 같이 표현되는데, t는 프레임 번호, f_i는 프레임 i의 바이트 수이다. 서버가 언더플로우 경계선보다 낮은 전송률로 프레임을 전송하면 클라이언트에서 언더플로우가 발생되어 QoS를 보장할 수 없다.

$$V(t) = \sum_{i=0}^{t} f_i \tag{1}$$

VB(t)는 오버플로우 경계선으로 언더플로우 경계선에 클라이언트의 버퍼 크기 b를 더한 값으로 식 (2)와 같이 표현된다. 서버에서 오버플로우 경계선보다 높은 전송률로 프레임을 전송하면 클라이언트에서 오버플로우가 발생되어 QoS를 보장할 수 없게 된다.

$$VB(t) = b + \sum_{i=0}^{t} f_i \tag{2}$$

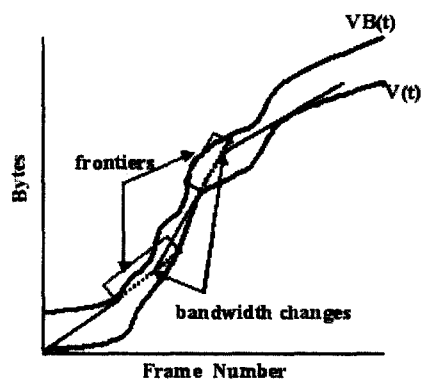
따라서, 서버에서 계산되는 전송률이 QoS를 만족하려면 식 (3)과 같이 언더플로우 경계선과 오버플로우 경계선 영역 내에 있어야 하며, c_i는 전송하려는 프레임이 i일때 서버에서 계산되는 전송률이다.

$$V(t) \le \sum_{i=0}^{t} c_i \le VB(t) \tag{3}$$

스무딩 알고리즘은 동일한 전송률로 전송할 수 있는 연속적인 프레임들을 검색해야 하며, 이때 이 전송률에 의해 언더플로우 경계선(오버플로우 경계선)을 만나는 경우에 이 지점부터 오버플로우(언더플로우)를 발생시키는 프레임까지의 구간을 연장 구간(frontiers)이라 하고, 연장 구간에 속한 프레임들 중에서 스무딩 알고리즘의 목적에 적합한 프레임을 검색하여 이 프레임에서 전송률을 변화시키며, 동일한 전송률로 전송하는 구간을 런(Run)이라 한다. 이러한 스무딩 기법의 원리를 바탕으로 QoS를 만족하면서 전송률 변화 횟수, 전송률 변화량, 그리고 버퍼 크기와 같은 특정 요소를 최적화하기 위한 목적으로 다양한 스무딩 알고리즘들이 연구되어 왔다. 스무딩 알고리즘들중에서 구현의 용이함과 네트워크의 전송률 예약에 대한 통신 비용을 줄이려는 목적을 갖는 CRTT(Constant Rate Transmission and Transport) 알고리즘[14]에서는 런의 개수가 1개이어서 클라이언트 버퍼 크기와 초기 지연 시간(Initial Delay)이 크고, 첨두 전송률의 이용률이 낮을 수 있다[14, 15]. 이러한 단점을 개선하여 전송률 변화 횟수 즉 런의 개수를 설정하므로써 CRTT 방법보다 버퍼 크기를 줄이려는 PCRTT-휴리스틱(Piecewise Constant Rate Transmission and Transport-Heuristic) 방법[14]은 언더플로우를 피하기 위해 모든 구간에 동일한 옵셋값을 적용하므로 버퍼 크기가 아직도 커질 수 있다[18-20]. e-PCRTT(enhanced-PCRTT) 알고리즘[19]은 버퍼 크기가 커질 수 있는 PCRTT-휴리스틱 방법의 단점을 개선하였지만, PCRTT-휴리스틱 방법에서와 같이 전송률 변화 횟수에 대한 제한이 있고 런의 크기가 일정하다. 그러므로 전송률 변화 횟수가 큰 경우에는 런의 크기가 작아지므로 연속된 런에서 전송률의 변화가 필요하지 않을 경우에도 강제적으로 변화를 시켜야 하는 경우가 발생될 수 있고, 전송률 변화 횟수가 작은 경우에는 런의 크기가 커지므로 주어지는 버퍼 크기가 작은 경우에 QoS를 보장하지 못할 수도 있다.

본 논문에서는 e-PCRTT 방법에서의 단점들을 해결하기 위해 런의 첫 프레임의 언더플로우 경계점과 오버플로우 경계점의 평균점을 시작점으로 하고 현재 검색중인 프레임의 언더플로우 경계점과 오버플로우 경계점의 평균점을 종점으로 하는 전송률을 설정하는데, 이 전송률이 QoS를 만족하는 경우에는 검색중인 프레임을 현재 런에 추가하고, QoS를 만족하지 않는 경우에는 현재 런을 완성하고 검색중인 프레임을 새로운 런에 배정하도록 하여 런의 크기가 가변적이고 전송률 변화 횟수에 대한 제한이 없는 스무딩 알고리즘을 제안한다.

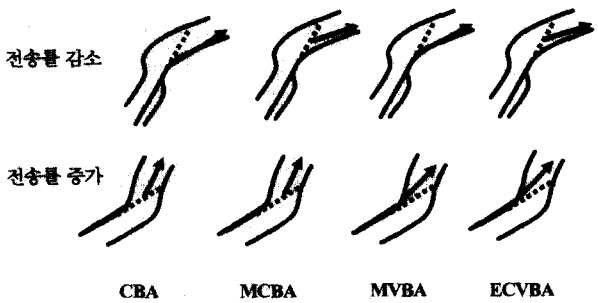
본 논문의 구성은 다음과 같다. 2장에서는 관련 연구를 살펴보고, 3장에서는 제안 알고리즘을 설명한다. 4장에서는 실험 결과를 설명하고, 5장에서는 결론 및 추후 연구 방향을 기술하였다.



(그림 2) 스무딩 기법의 원리

2. 관련 연구

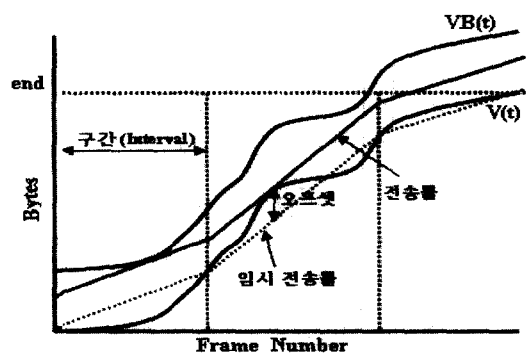
스무딩 기법은 가변 비트율로 저장된 비디오 스트림을 전송할 때 버스트를 줄이기 위해 일련의 고정 비트율로 변환하여 전송하는 방법으로, 연장 구간에 속하는 프레임 중에서 다음 런의 시작 프레임을 선택하는 방법은 알고리즘의 목적에 따라 다르다. (그림 3)은 CBA(Critical Bandwidth Allocation), MCBA (Minimum Changes Bandwidth Allocation), MVBA(Minimum Variability Bandwidth Allocation), ECVBA(Efficient Changes Variability Bandwidth Allocation) 알고리즘의 전송률 조절 방법을 나타낸다[18, 20, 27].



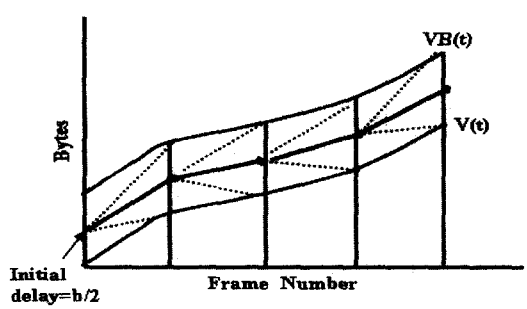
(그림 3) CBA, MCBA, MVBA, ECVBA의 전송률 조절 방법

CBA 알고리즘[10, 11, 21]에서는 현재 전송률에 의해 오버플로우가 발생된다면 연장 구간의 첫 프레임을 다음 런의 시작 프레임으로 설정하며, 현재 전송률에 의해 언더플로우가 발생되는 경우에는 연장 구간에 속한 프레임들 중에서 새로운 전송률로 언더플로우나 오버플로우가 발생되지 않고 가장 많은 프레임을 전송할 수 있는 프레임을 검색하여 이 프레임을 다음 런의 시작 프레임으로 설정하여 전송률 증가 횟수를 최소화한다[6, 19]. MCBA 알고리즘[6, 18]의 목적은 전송률 변화 횟수의 최소화인데, 이는 서버와 네트워크 사이의 전송률 조절에 관련된 통신 오버헤드를 감소시키고 서버에 저장된 비디오 데이터에 대한 접근 스케줄링을 용이하게 한다[18]. 이를 위해 다음 런의 전송률을 감소 또는 증가시켜야 할 때에 연장 구간에 있는 프레임 중에서 새로운 전송률에 의해 가장 많은 프레임을 전송할 수 있는 프레임을 검색하여 이 프레임을 다음 런의 시작 프레임으로 설정하지만, 알고리즘의 계산 시간이 크다[12]. MVBA 알고리즘[13, 19]은 전송률 변화량을 최소화하는 것이 목적이며, 시간 복잡도가 $O(n^2)$인 알고리즘[13]과 $O(n)$인 알고리즘[22]이 있다. 공유되는 네트워크의 자원들에게 이전 전송률보다 증가량이 큰 전송률을 요구하면 요구된 전송률을 예약하지 못할 수도 있다. 그래서 이전 전송률보다 전송률의 증감이 완만한 전송 계획을 세우는 것이 좋다[6, 18]. 이를 위해 MVBA 알고리즘은 다음 런의 전송률을 증가 또는 감소시켜야 할 때에 연장 구간의 첫 프레임을 다음 런의 시작 프레임으로 설정하는데 전송률 변화 횟수가 상대적으로 많다[12]. ECVBA 알고리즘[27]은 서버가 보다 많은 비디오 스트림을 서비스 할 수 있도록 다음 런의 전송률을 감소시켜야 할 경우에는 시작 프레임을 MCBA 알고리즘에서와 같이 설정하여 현재 전송률 보다 낮은 전송률로 보낼 수 있는 구간의 크기를 증가시키고, 다음 런의 전송률을 증가시켜야 할 경우에는 시작 프레임을 MVBA 알고리즘에서와 같이 설정하여 전송률 증가량을 최소화한다. Slice-and-Patch 알고리즘[28]은 비디오 프레임들을 첨두 전송률에 영향을 주는 프레임들과 그렇지 않는 프레임들로 분할하여 첨두 전송률에 영향을 주는 프레임들은 초기 지연 시간 동안에 전송하여 첨두 전송률을 최소화하지만 초기 지연 시간이 클 수 있고 프레임들의 재 조합에 대한 오버헤드가 발생한다.

(그림 4)는 PCRTT-휴리스틱 방법에서의 전송률 조절 방법[18, 23, 29]을 나타내는데, 전송률 변화 횟수가 주어지면 전체 비디오 스트림을 구성하는 프레임 수를 이 값으로 나눈 몫이 1개의 전송률로 보낼 수 있는 런 즉 구간의 크기가 된다. 이와같이 설정된 구간 크기로 비디오 스트림을 분할하여, 각 구간의 시작 프레임의 언더플로우 경계선을 시작점으로 하고 구간의 끝 프레임의 언더플로우 경계선을 종점으로 하는 연결선을 만드는데, 이 연결선이 해당 구간의 임시 전송률이다. 이와같이 모든 구간에서의 임시 전송률을 계산하여, 이 전송률에 의해 언더플로우가 가장 크게 발생되는 프레임이 속한 구간의 프레임들이 언더플로우를 피할 수 있는 최소의 오프셋 값을 구하여 이 오프셋 값을 모든 임시 전송률에 더해 각 구간의 전송률로 설정한다. 그러나 이 방법은 특정 구간에서만 언더플로우가 심하게 발생되더라도 모든 구간에 같은 옵셋 값을 더하므로 요구되는 버퍼 크기가 커질 수 있다[19]. PCRTT-휴리스틱 방법의 단점을 개선한 e-PCRTT 알고리즘[19, 29]은 (그림 5)와 같이 초기 지연 시간 동안 클라이언트 버퍼에 버퍼 크기의 1/2에 해당하는 바이트 수를 미리 저장하고 각 구간에서 언더플로우가 발생되지 않을 최소 전송률과 오버플로우가 발생되지 않을 최대 전송률의 평균을 해당 구간의 전송률로 설정한다. PCRTT-다이나믹(Dynamic) 방법[17]은 전송률 변화 횟수가 주어지면 이 값을 만족하면서 네트워크 또는 서버의 목적에 적합한 특정 요소, 즉 버퍼 크기, 첨두 전송률, 전송률 변화량 등을 최적화하는 구간의 크기를 찾으므로 구간의 크기가 가변적이지만, 알고리즘의 계산시간이 크다[17, 18, 20].



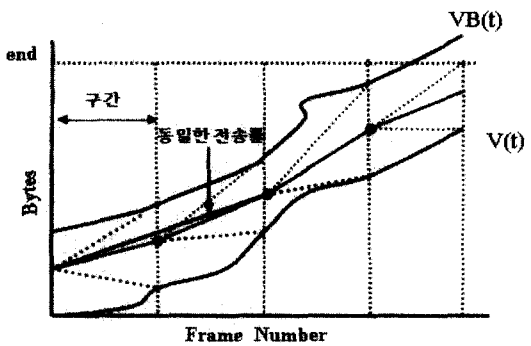
(그림 4) PCRTT 알고리즘의 전송률 조절 방법



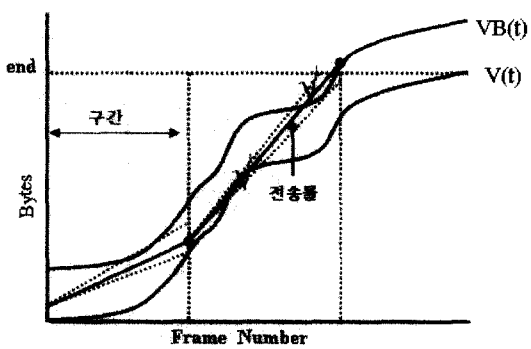
(그림 5) e-PCRTT 알고리즘의 전송률 조절 방법

3. 제안 알고리즘

전송률 변화 횟수가 주어져서 구간의 크기가 일정한 e-PCRTT 알고리즘은 주어지는 전송률 변화 횟수가 크면 구간의 크기가 작게 구성되는데, 이러한 경우에는 (그림 6)(a)에서와 같이 처음 2개 구간은 동일한 전송률로 설정될 수 있음에도 구간마다 전송률을 계산하기 때문에 전송률 변화 횟수가 많아질 수 있다. 또한 주어지는 전송률 변화 횟수가 적을 경우에는 구간의 크기가 커지므로 클라이언트 버퍼 크기가 작다면 QoS를 보장하지 못할 경우가 발생될 수 있다. 즉 (그림 6)(b)의 2번째 구간에서 언더플로우가 발생되지 않을 최소 전송률이 오버플로우 경계선을 초과하기 때문에 QoS를 보장할 수 없으며, QoS를 보장하기 위해서는 보다 큰 버퍼 크기가 요구될 수 있다.



(a) 불필요하게 전송률이 변화되는 경우



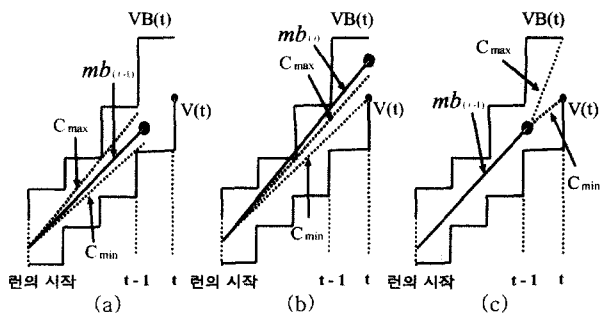
(b) QoS를 보장할 수 없는 경우

(그림 6) e-PCRTT에서 전송률 조절의 문제점

따라서, 적은 버퍼 크기로 비디오 스트림을 전송할 수 있는 e-PCRTT의 장점을 갖으면서, e-PCRTT에서 발생할 수 있는 문제들을 해결하기 위해 전송률 변화 횟수의 제한이 없고 구간의 크기가 가변적인 알고리즘을 제안한다. 즉, 제안 알고리즘에서는 런의 시작 프레임의 언더플로우 경계점과 오버플로우 경계점의 평균점을 시작점으로 하고 검색중인 프레임의 오버플로우 경계점과 언더플로우 경계점의 평균점을 종점으로 하는 연결선을 전송률로 설정하여, 이 전송률이 QoS를 만족하는 경우에는 검색중인 프레임을 현재 런에 추가하여 가능한 많은 프레임을 동일한 전송률로 보낼 수 있게 하므로써 불필요하게 전송률이 변화되지 않도록 하며, QoS를 보장하지 못하는 경우에는 현재 런을 완성하고 검색중인 프레임을 새로운 런의 시작 프레임으로 설정하므로써 버퍼 크기와 무관하게 항상 QoS를 보장할 수 있도록 한다.

(그림 7)은 제안된 전송률 조절 방법이다. C_{max}는 런의 시작 프레임부터 현재 검색하고 있는 프레임까지 QoS를 보장하는 전송률 중에서 최대 전송률이고 C_{min}은 최소 전송률이다. $mb_{(t-1)}$는 현재 검색중인 프레임 t-1에 의해 생성되는 전송률이다.

(그림 7)(a)에서는 $mb_{(t-1)}$가 QoS를 만족하는 C_{min}과 C_{max} 범위내에 있으므로 런의 시작 프레임부터 프레임 (t-1)까지의 프레임들은 동일한 런에 포함시킬 수 있으며, 다음 프레임 t를 동일한 런에 포함시킬 수 있는지를 검색한다. $mb_{(t)}$가 (그림 7)(b)와 같이 C_{min}과 C_{max} 범위를 벗어나 QoS를 보장할 수 없다면, 현재 런을 완성하고 새로운 런을 구성하여 QoS를 보장할 수 있도록 해야 한다. 즉, (그림 7)(c)와 같이 런의 시작 프레임부터 프레임 (t-1)까지의 전송률을 $mb_{(t-1)}$로 설정하여 현재 런을 완성하고, 프레임 (t-1)을 새로운 런의 시작 프레임으로 설정한다.



(그림 7) 제안 방법에서의 전송률 조절

이와 같은 원리를 사용한 알고리즘이 <표 1>인데, n은 비디오 스트림을 구성하는 프레임 개수이다. 단계 (2)는 런의 시작 프레임부터 현재 검색중인 프레임 t까지의 전송률인 $mb_{(t)}$를 계산하는 과정으로 q는 버퍼에 채워져 있는 바이트 수이고, ts는 런의 시작 프레임을 나타낸다. 단계 (3)에서 (7)까지는 프레임 t까지의 전송률 $mb_{(t)}$가 C_{max}보다 크거나 C_{min}보다 작게 된다면, 즉 QoS를 보장하는 범위를 벗어난다면 런의 시작 프레임부터 프레임 (t-1)까지의 전송률은 $mb_{(t-1)}$로 설정하고 다음 런의 시작 프레임은 프레임 (t-1)이 되는 과정이다. output(ts~t-1, $mb_{(t-1)}$)는 프레임 ts부터 프레임 (t-1) 까지의 전송률을 $mb_{(t-1)}$로 설정하는 함수

이다. (그림 8)은 제안 알고리즘에 의한 (그림 6)(b)의 전송률 조절 예로써, QoS를 보장하지 못하던 2번째 구간에서 제안 알고리즘에서는 전송률을 변화시켜 버퍼 크기가 작더라도 QoS를 보장할 수 있음을 보여준다.

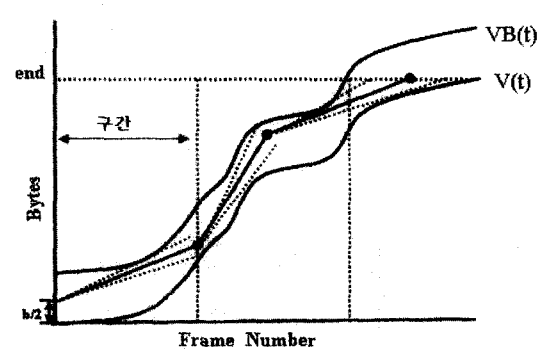
제안 알고리즘의 시간 복잡도는 다음 런의 시작 시점이 최대 전송률 C_{max} 또는 최소 전송률 C_{min} 을 벗어나기 이전 프레임이 되어 n에 비례하는 실행 횟수를 가지므로 $O(n)$이 된다. <표 2>는 제안 알고리즘과 기존 스무딩 알고리즘들의 시간 복잡도 비교이다. 제안 알고리즘의 시간 복잡도는 낮은 시간 복잡도를 갖는 CBA, PCRTT, e-PCRTT 알고리즘과 같다.

<표 1> 제안 알고리즘

```
PROCEDURE proposed algorithm ( )
    t = 1 , ts = 1, C_max = b
    C_min = size of 1'st frame, q = b / 2
    n = the number of last frame
    REPEAT
(1) set t = t + 1
(2) mb(t) = (((V(t) + VB(t)) / 2.0) - (V(ts) + q)) / (t - ts)
(3) IF (C_max < mb(t)) Or (C_min > mb(t)) then
(4)         mb(t-1) = compute mb in t - 1
(5)         output(ts ~ t - 1, mb(t-1))
(6)             ts = t - 1
(7) ENDIF
(8) compute C_max , C_min , q
(9) UNTIL t = n
END PROCEDURE
```

<표 2> 시간 복잡도 비교

알 고 리 즘	시간 복잡도
MVBA[3, 4]	$O(n^2) \sim O(n)$
CBA[13]	$O(n)$
MCBA[13]	$O(n^2 \log n) \sim O(n \log n)$
PCRTT[13]	$O(n)$
e-PCRTT[16]	$O(n)$
제안 알고리즘	$O(n)$



(그림 8) 제안 알고리즘에 의한 (그림 6)(b)의 전송률 조절 예

4. 실험 결과

C언어로 스무딩 알고리즘들을 구현하여 실험하였으며, <표 3>은 실험에 사용된 MPEG-2로 저장된 비디오 소스들 [24, 25]의 파라미터들이다. Length는 비디오 재생 시간, Ave Frame Size는 각 프레임들의 평균 바이트 수, Max Frame Size와 Min Frame Size는 프레임들의 바이트 수들 중에서 가장 큰 값과 작은 값이고, Total Size는 프레임들의 바이트 수의 합이다. 또한, Std Dev는 프레임들의 바이트 수에 대한 표준 편차로써 이 값이 클수록 프레임들 사이에 바이트 수의 변화가 심하다. 즉 1993 Final Four는 바이트 수의 변화가 가장 심한 소스이고, Seminar는 바이트 수의 변화가 아주 적은 소스이다.

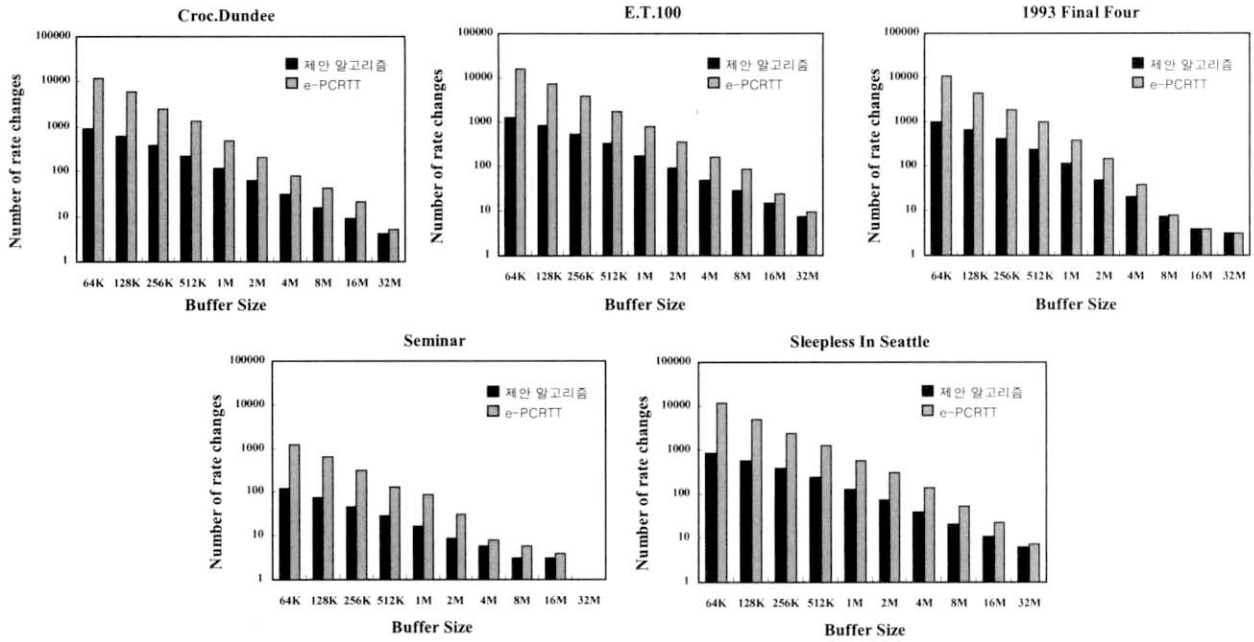
<표 3> MPEG 비디오 소스들의 파라미터들

Video Clip Name	Length (min)	Ave Frame Size(KB)	Max Frame Size(KB)	Min Frame Size(KB)	Total Size(KB)	Std Dev(KB)
Croc. Dundee	94	10.52	18.98	1.233	1,773,379	10.764
E.T.100	110	15.28	29.836	6.666	3,035,136	15.71
1993 Final Four	41	16.07	28.872	2.504	1,178,013	16.57
Seminar	63	8.4	10.791	7.012	953,385	8.422
Sleepless In Seattle	101	9.255	16.227	3.1318	1,679,428	9.5617

4.1 e-PCRTT와의 비교

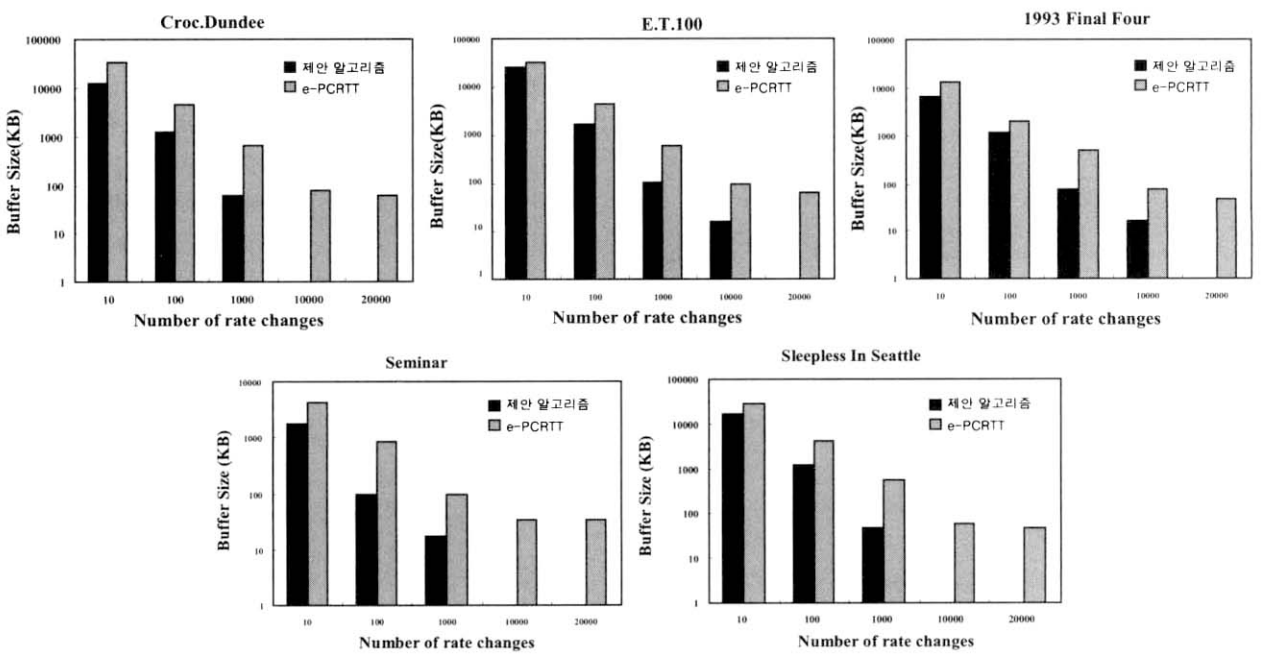
e-PCRTT 알고리즘과의 성능 평가를 위해 <표 3>의 비디오 데이터를 사용하여 전송률 변화 횟수, QoS를 유지하기 위해 요구되는 버퍼 크기, 첨두 전송률, 전송률 변화량, 버퍼 이용률, 그리고 첨두 전송률의 이용률을 비교하였다. e-PCRTT 알고리즘에서 사용되어야 하는 구간 크기는 실험에 사용되는 버퍼 크기에서 언더플로우 또는 오버플로우가 발생되지 않으면서 일정한 크기의 구간을 구성할 수 있는 최대 값으로 설정하여 제안 방법과 동일한 조건이 될 수 있도록 하였다.

(그림 9)는 각 버퍼 크기에 대한 전송률 변화 횟수를 비교한 결과이다. 두 방법에서의 전송률 변화 횟수는 버퍼 크기가 클수록 비슷해지는데, 이것은 버퍼 크기가 클수록 e-PCRTT 알고리즘에서 사용되는 구간의 크기가 커지기 때문이다. 특히, Seminar 데이터에서 버퍼 크기가 32M인 경우에는 주어진 버퍼 크기가 크고 프레임들간의 표준 편차가 적어서 구간의 개수가 1이 된다. 버퍼 크기가 적은 경우에는 e-PCRTT 방법에서의 구간의 개수가 많아져서 전송률 변화 횟수가 제안 방법보다 많아진다.

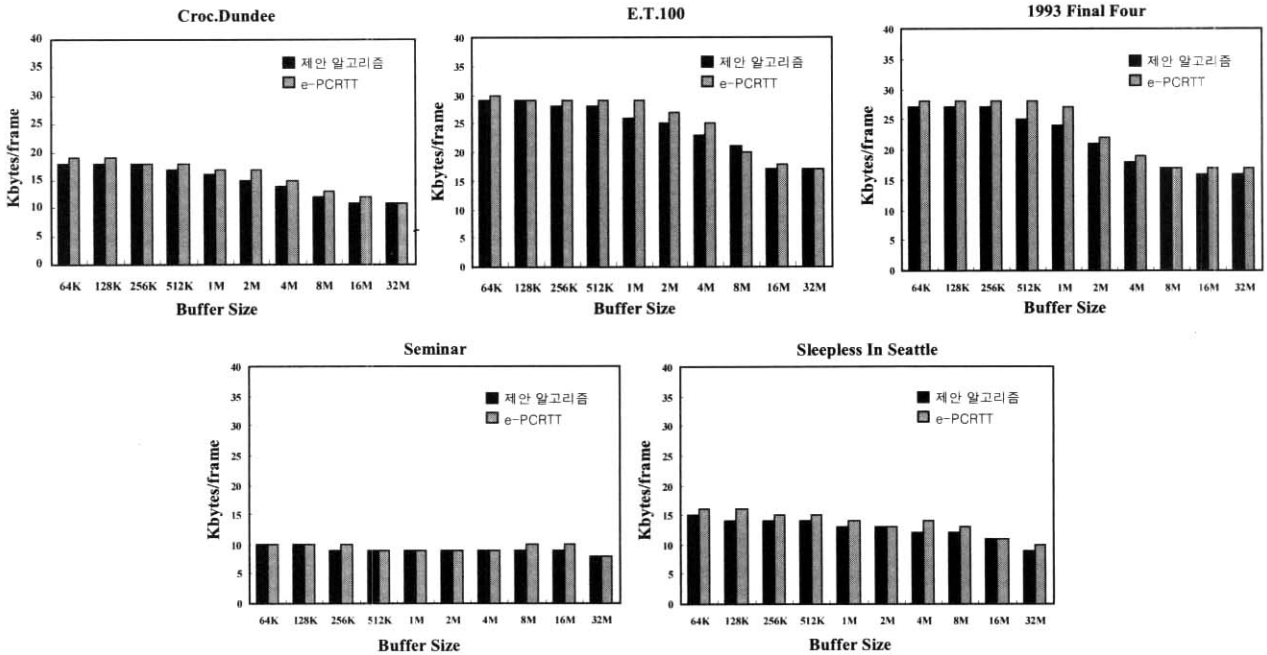


(그림 9) 전송률 변화 횟수 비교

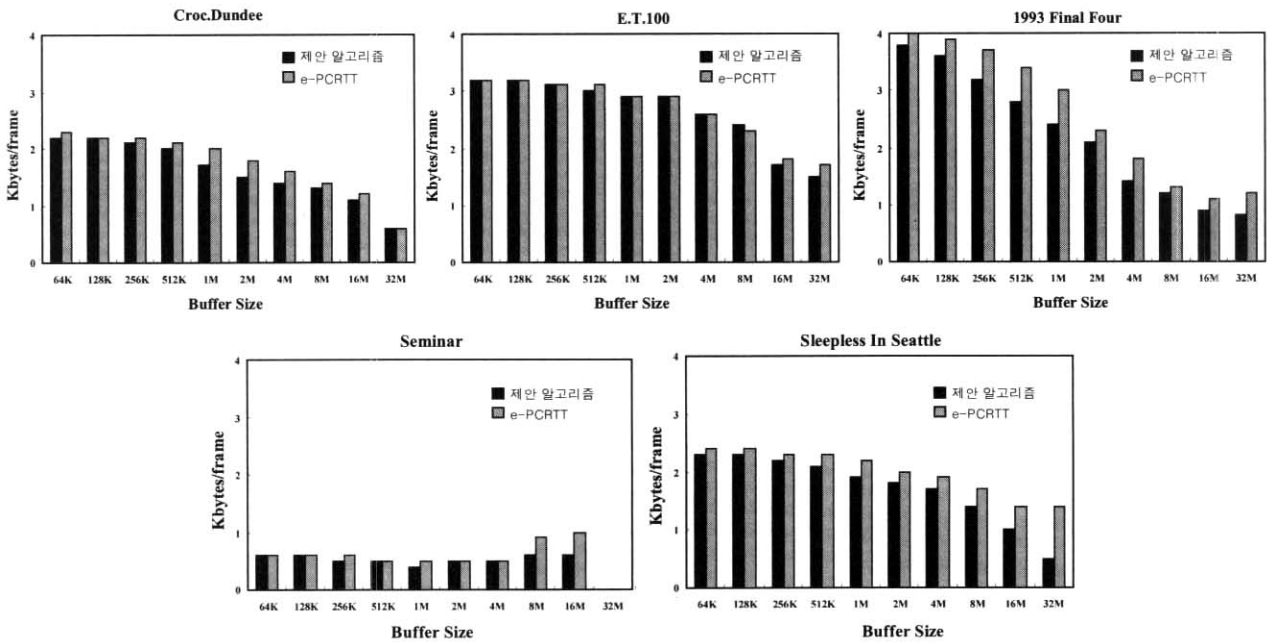
(그림 10)은 전송률 변화 횟수를 고정시킨 경우에 QoS를 유지하기 위해 요구되는 최소 버퍼 크기를 비교한 결과이다. 최소 버퍼 크기는 제안 알고리즘에서는 버퍼 크기를 16KB 크기로 증가시키면서 QoS를 유지하고 전송률 변화 횟수를 만족하는 첫 번째 버퍼 크기이고, e-PCRTT 알고리즘에서는 전송률 변화 횟수에 의해 구간 크기를 설정하고 버퍼 크기를 16KB 크기로 증가시키면서 QoS를 만족시키는 첫 번째 버퍼 크기이다. QoS를 유지하기 위해 요구되는 버퍼 크기는 e-PCRTT 방법이 크며, 이것은 e-PCRTT 알고리즘에서는 일정한 구간 크기로 인해 구간의 시작 프레임의 바이트 수와 끝 프레임의 바이트 수의 차이가 크거나 구간내에서 프레임들간의 바이트 수의 차이가 많은 경우에는 요구되는 버퍼 크기가 커질 수 있기 때문이다. 즉, 전송률 변화 횟수가 동일한 경우에 제안 알고리즘은 e-PCRTT 알고리즘보다 적은 버퍼 크기로 비디오 스트림을 전송할 수 있다. 특히, 제안 알고리즘에서는 여러 비디오 소스에서 전송률 변화 횟수가 20,000인 경우와 같이 1KB보다 작아지는 경우도 있다.



(그림 10) QoS를 유지하기 위한 버퍼 크기 비교



(그림 11) 첨두 전송률 비교



(그림 12) 전송률 변화량 비교

(그림 11)은 각 버퍼 크기에 대해 계산되는 전송률 중에서 가장 높은 전송률인 첨두 전송률의 비교이다. 제한된 전송률을 갖는 네트워크 자원 또는 서버는 1개의 비디오 스트림에 대한 첨두 전송률이 낮을수록 보다 많은 스트림을 제공할 수 있다[18]. e-PCRTT 알고리즘에서는 첨두 전송률이 구간을 구성하는 프레임들간의 표준 편차와 구간의 첫 프레임의 바이트 수와 끝 프레임의 바이트 수의 차이에 영향을 받기 때문에 제안 방법보다 일반적으로 높다. 그러나 E. T.100에서 버퍼 크기가 8M인 경우와 같이 제안 방법보다 낮은 경우는 구간 크기를 결정할 때에 가능한 최대 크기를 설정하여 전송률 변화 횟수를 최적화하였기 때문이다.

(그림 12)는 각 버퍼 크기에 대해 전송률 변화량을 비교한 결과인데, 전송률 변화량은 식 (4)[18]에 의해 계산되며, n은

전송률 변화 횟수, c_i는 i번째 런의 전송률, 그리고 stddev는 표준 편차이다.

$$\frac{stddev\{c_0, c_1, \cdots, c_{n-1}\}}{\frac{1}{n} \times \sum_{i=0}^{n-1} c_i} \quad (4)$$

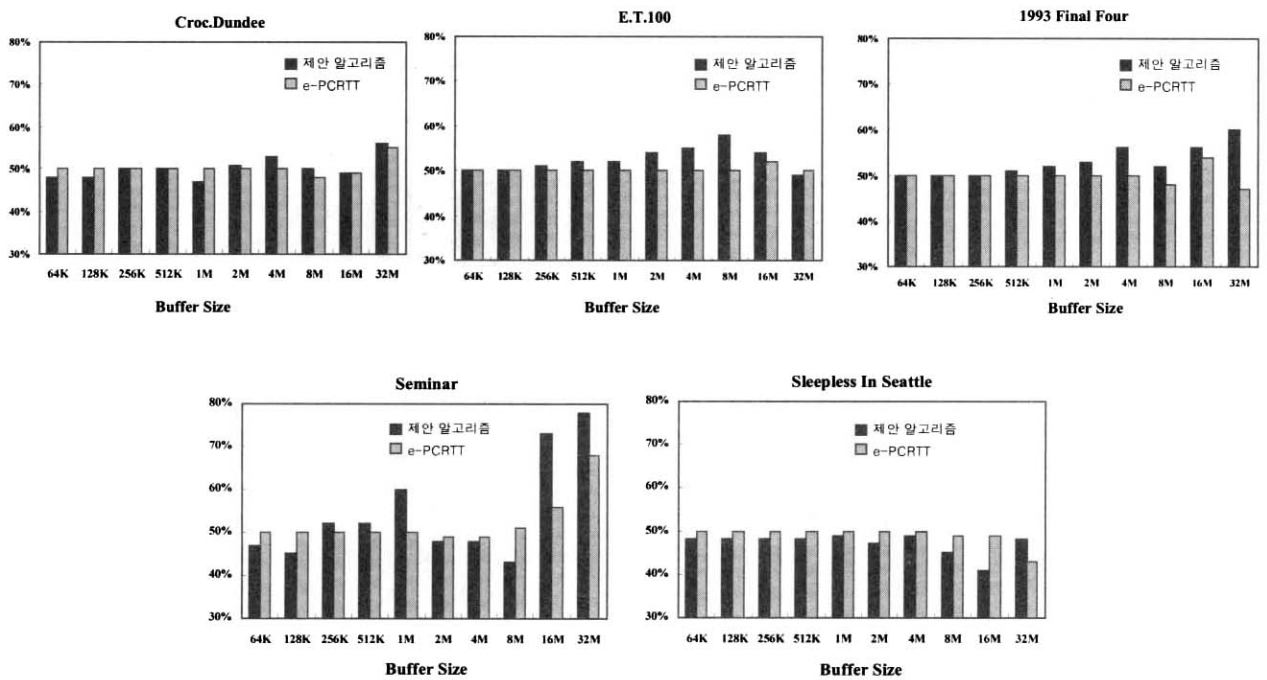
제안 알고리즘의 전송률 변화량이 낮은데, 이것은 e-PCRTT 알고리즘의 경우에 하나의 구간 내에서는 전송률을 변화시킬 수 없으므로 이전 런의 전송률에 비해 급격하게 전송률이 변화될 수 있기 때문이다. 또한, 두 방법에서의 전송률 변화량은 Seminar에서 가장 낮은데, 이것은 프레임들간의 표준 편차가 작기 때문에 이전 런의 전송률에 비해 새로운 런의 전송률을 크게 변화시킬 필요가 없기 때문이다. 특히, Seminar에서 버퍼 크기가 32M인 경우에 전송률 변화 횟수가 1이 되어 전송률 변화량은 0이 된다.

(그림 13)은 각 버퍼 크기에 대해 버퍼 이용률을 비교한 결과로써, 이 값은 모든 프레임들에 대해 서버에서 계산되는 전송률과 언더플로우 경계선의 차이에 대한 평균으로 계산되며 값이 낮을수록 많은 다른 비디오 스트림 또는 응용 프로그램들이 클라이언트 버퍼를 공유할 수 있다[18]. e-PCRTT 알고리즘에서는 구간의 전송률이 언더플로우가 발생되지 않을 최소 전송률과 오버플로우가 발생되지 않을 최대 전송률의 평균으로 계산되기 때문에, 실험에 사용된 대부분 비디오 소스의 경우에 버퍼 이용률이 약 50%로 일정하다. 그러나 Seminar 비디오 소스에서 일부 버퍼 크기인 경우에 최소 전송률이 최대 전송률에 근접하게 되어 버퍼 이용률이 높아지게 된다. 제안 알고리즘의 경우에도 대부분 비디오 소스에서 버퍼 이용률이 약 50%이지만, 일부 버퍼 크기에서 e-PCRTT 알고리즘보다 매우 높거나 낮게 되는데, 이는 전송률이 언더플로우 경계선 또는 오버플로우 경계선에 편중될 수 있기 때문이다.

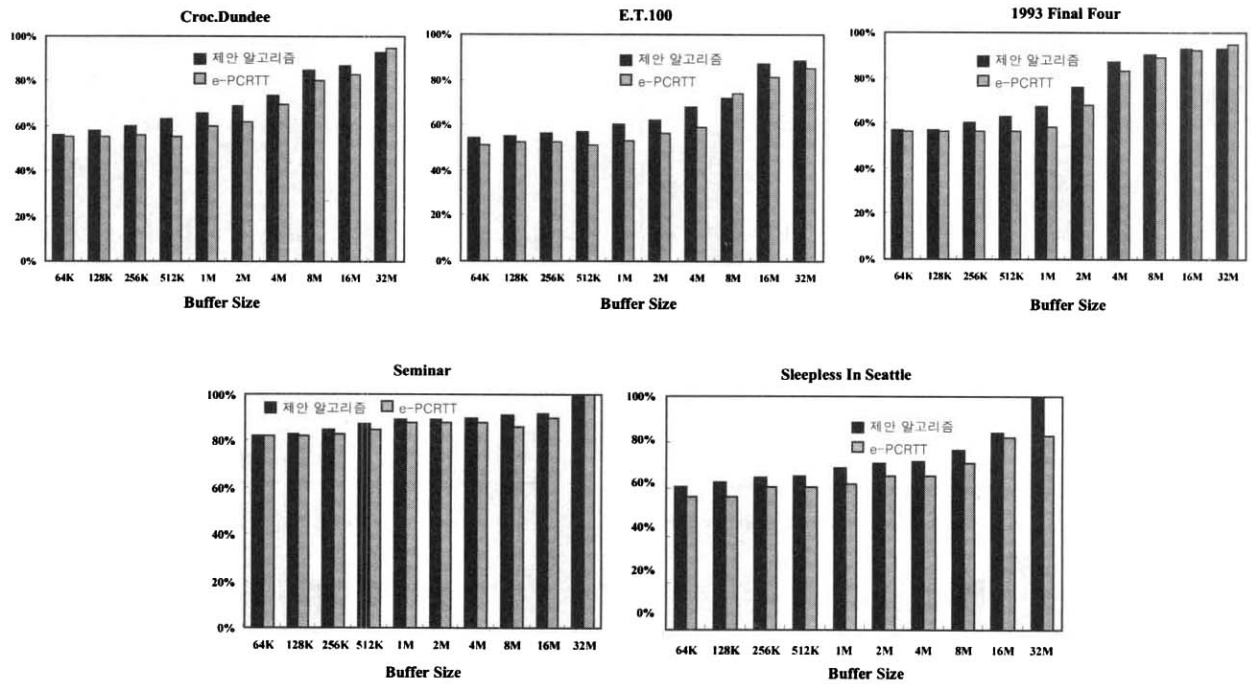
(그림 14)는 각 버퍼 크기에 대해 첨두 전송률의 이용률 비교이며 식 (5)[10]에 의해 계산되고, 이 값이 높을수록 첨두 전송률의 낭비가 없음을 의미한다.

$$\frac{\sum_{i=0}^{n-1}(c_i)}{\max\{c_i\} \times n} \quad (5)$$

제안 방법이 Croc.Dundee에서 버퍼 크기가 32M, E.T100에서 버퍼 크기가 8M, 그리고 1993 Final Four에서 버퍼 크기가 32M인 경우를 제외하고 e-PCRTT 방법보다 높은데, 이는 e-PCRTT 알고리즘에서는 구간내의 프레임들간의 바이트 수의 차이가 큰 경우가 발생되면 언더플로우가 발생되지 않을 최소 전송률이 큰 값을 갖게 되어 이 구간의 전송률이 첨두 전송률에 근접되기 때문이다.



(그림 13) 버퍼 이용률 비교



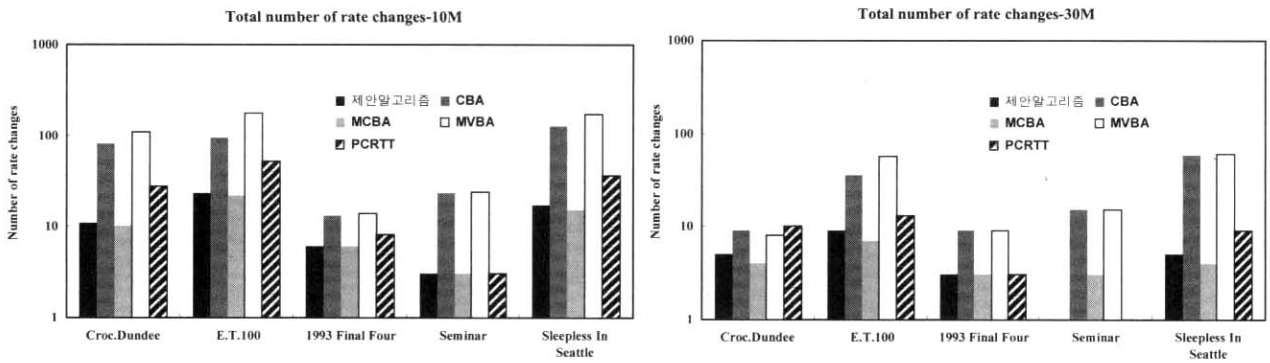
(그림 14) 첨두 전송률 이용률 비교

4.2 다른 스무딩 알고리즘들과의 비교

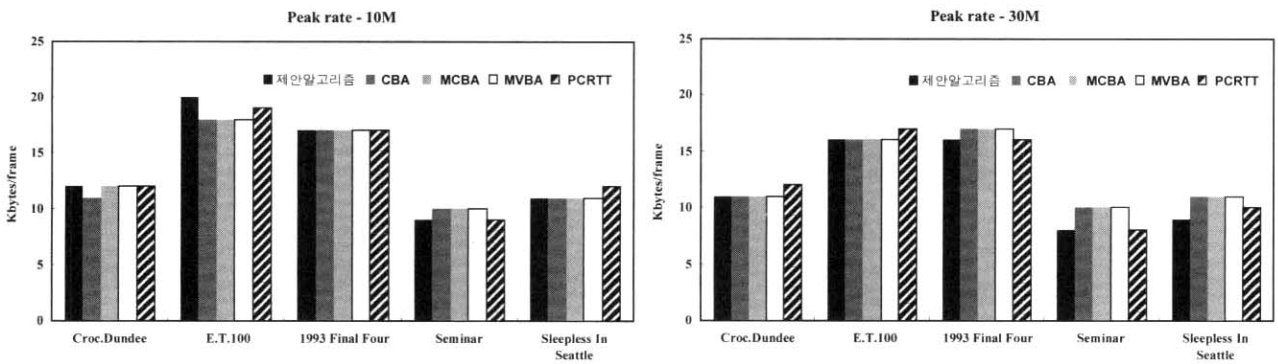
다른 스무딩 알고리즘들과의 성능 평가를 위해 <표 3>의 비디오 데이터로써 클라이언트 버퍼 크기를 10M, 30M [6, 21, 22]로 제한하여 전송률 변화 횟수, 첨두 전송률, 전송률 변화량, 그리고 첨두 전송률의 이용률을 비교하였다. 또한 네트워크 트래픽의 변동으로 인해 서버에서 계산된 전송률보다 클라이언트에서 프레임을 더 빨리 받거나 늦게 받아서 언더플로우 또는 오버플로우를 발생시킬 수 있는 프레임의 개수, 그리고 알고리즘의 계산 시간도 비교하였다. PCRTT-휴리스틱 방법에서의 구간 크기는 4.1절에서 설명된 e-PCRTT 방법에서의 구간 크기와 동일하게 설정하였다.

(그림 15)는 전송률 변화 횟수에 대한 비교 결과이다. 전송률 변화 횟수는 버퍼 크기가 30M인 경우가 10M인 경우보다 적은데, 이것은 버퍼 크기가 클수록 QoS를 만족하는 영역이 넓어져 동일한 전송률로 보낼수 있는 프레임들이 증가하기 때문이다. 제안 알고리즘의 전송률 변화 횟수는 전송률 변화 횟수의 최소화가 목적인 MCBA 알고리즘보다는 약간 많지만 다른 알고리즘들보다는 작다. 특히, MVBA 알고리즘은 연장 구간의 시작 프레임을 다음 런의 시작 프레임으로 설정하기 때문에 전송률 변화 횟수가 매우 많다. Seminar 비디오에서 버퍼 크기가 30M인 경우에 제안 알고리즘과 PCRTT 알고리즘의 전송률 변화 횟수는 1이다.

(그림 16)은 첨두 전송률에 대한 비교 결과인데, 버퍼 크기가 10M인 경우에는 프레임들의 바이트 수중에서 최대값(Max Frame Size)이 클수록 첨두 전송률이 높고, 버퍼 크기가 30M인 경우에는 프레임들의 바이트 수의 평균(Ave Frame Size)이 클수록 높다. 이것은 버퍼 크기가 작은 10M인 경우에는 바이트 수가 큰 프레임들에 대해 전송률을 완만하게 조절하지 못하지만, 버퍼 크기가 큰 30M인 경우에는 전송률을 완만하게 변화시킬 수 있기 때문이다. 특히, 제안 알고리즘의 첨두 전송률이 E.T.100 비디오에서 다른 알고리즘들보다 높다. 이는 검색되는 프레임의 언더플로우 경계점과 오버플로우 경계점의 평균으로 전송률을 조절하므로, 프레임들중에서 바이트 수의 최대 값이 큰 경우에 완만하게 전송률을 조절하지 못할 수 있기 때문이다. 또한, 첨두 전송률은 일반적으로 프레임들간의 표준 편차가 적어 완만하게 전송률을 변화시킬 수 있는 Seminar 비디오에서 낮고, 프레임들간의 표준 편차가 큰 편에 속해 급격하게 전송률을 변화시킬 수 있는 1993 Final Four 와 E. T. 100 비디오에서 높다. 물론 버퍼 크기가 10M인 경우보다 30M에서 낮은데, 이것은 버퍼 크기가 클수록 QoS를 만족하는 영역이 넓어져 완만하게 전송률을 변화시킬 가능성이 높아지기 때문이다.

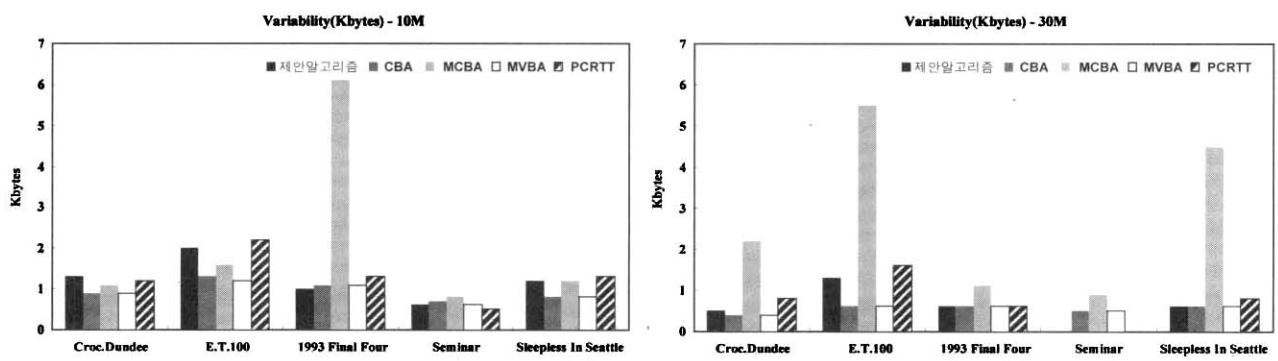


(그림 15) 전송률 변화 횟수 비교



(그림 16) 첨두 전송률 비교

(그림 17)은 전송률 변화량의 비교이다. 제안 알고리즘은 버퍼 크기가 30M인 경우 Seminar 비디오를 제외하고 전송률 변화량의 최소화가 목적인 MVBA 알고리즘보다 크다. CBA 알고리즘의 전송률 변화량은 전송률을 감소시켜야 될 경우에 MVBA 알고리즘과 전송률 조절 과정이 동일하기 때문에 비슷하지만, MCBA 알고리즘인 경우에는 전송률 변화 횟수를 최소화하기 위해 전송률 변화량을 고려하지 않고 연장 구간에서 새로운 전송률로 가장 많은 프레임을 보낼 수 있는 프레임을 검색하기 때문에 전송률이 급격하게 변화될 수 있어 높은 편에 속한다. 특히, MCBA 알고리즘에서 버퍼 크기가 10M인 경우에 Final Four, 버퍼 크기가 30M인 경우에 E.T.100과 Sleepless In Seattle에서 아주 높은데, 이것은 전송률을 감소시켜야 할 경우에 현재 런의 전송률과 큰 차이가 발생될 수 있는 프레임이 새로운 런의 시작 프레임으로 검색되기 때문이다. PCRTT 알고리즘의 전송률 변화량은 구간의 크기가 고정적이어서 구간의 시작 프레임의 바이트 수와 끝 프레임의 바이트 수의 차이가 크거나 구간을 구성하는 프레임들간의 바이트 수의 차이가 큰 경우가 발생되어 높다. 그러나 Seminar에서 버퍼 크기가 30M인 경우에 제안 알고리즘과 PCRTT 알고리즘의 전송률 변화량은 전송률 변화 횟수가 1이므로 0이 된다.



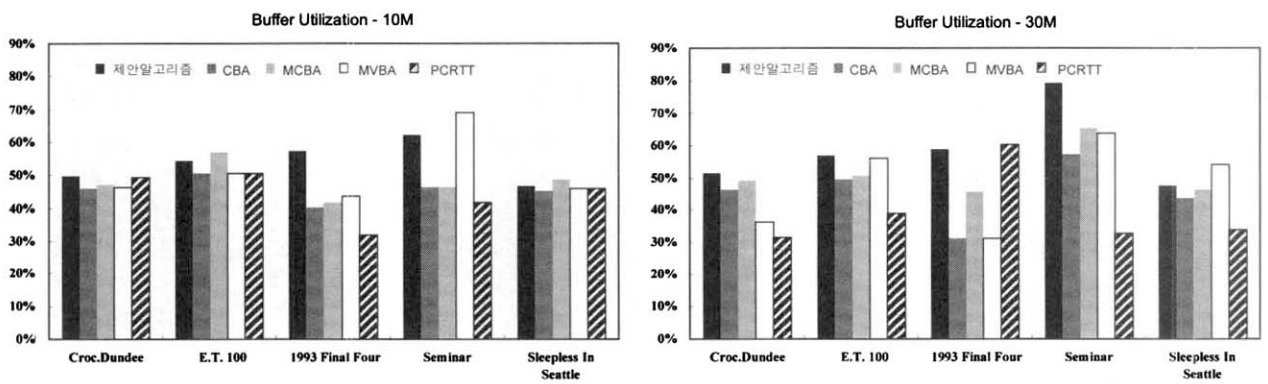
(그림 17) 전송률 변화량 비교

(그림 18)은 클라이언트 버퍼 이용률의 비교 결과인데, 이 값은 서버가 계산한 전송률이 언더플로우 경계선 또는 오버플로우 경계선과 인접되는 프레임 갯수에 영향을 받는다. 즉, 서버가 계산한 전송률이 언더플로우 경계선과 인접되는 프레임 수가 많으면 0%에 근접하고, 오버플로우 경계선과 인접되는 프레임 수가 많으면 100%에 근접한다. 특히, CBA, MCBA, MVBA 알고리즘의 버퍼 이용률은 다음 런의 시작 프레임의 위치에 종속되는데, 연장 구간의 시작 프레임 또는 끝 프레임을 다음 런의 시작 프레임으로 설정하는 경우에는 언더플로우 경계선과 오버플로우 경계선에 인접되는 프레임 수가 비슷하여 버퍼 이용률이 약 50%가 되지만, 그렇지 않은 경우에 언더플로우 경계선 또는 오버 플로우 경계선에 편중되는 프레임 개수가 많아져 50%보다 훨씬 크거나 작게 된다. 제안 알고리즘에서의 전송률 조절 방법은 전송률 변화 횟수가 적은 Seminar 비디오의 경우에 오버플로우 경계선에 인접되는 프레임 개수가 언더플로우 경계선에 인접되는 프레임 개수보다 많아져서 높게 나타났다. 이를 제외하면 제안 알고리즘의 버퍼 이용률도 약 50%이다.

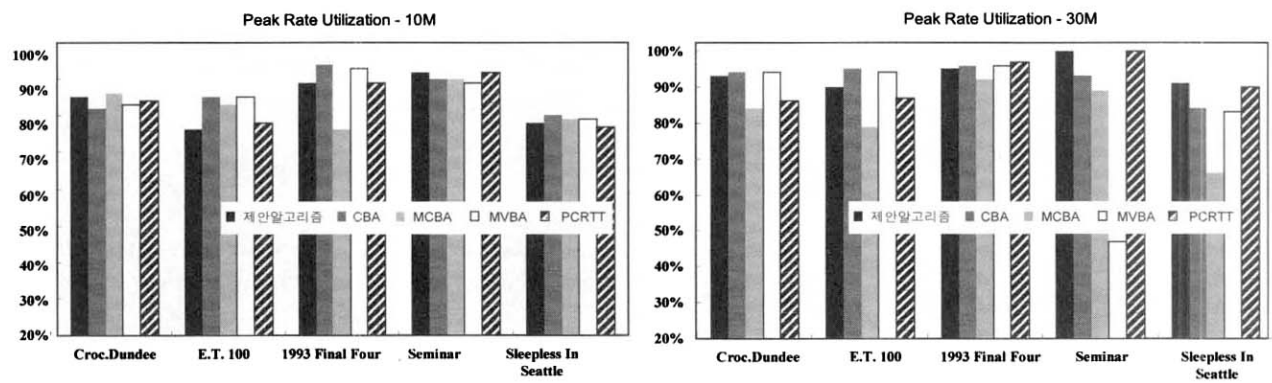
(그림 19)는 첨두 전송률의 이용률에 대한 비교 결과이다. 제안 알고리즘은 버퍼 크기가 10M인 경우에 E.T.100 비디오에서 낮다. 이는 첨두 전송률이 높은 것에 비해 이 전송률과 비슷한 전송률을 갖는 런의 수가 적기 때문이다. 이를 제외하면 제안 알고리즘은 높은 편에 속한다. 버퍼 크기가 30M에서 Seminar 비디오 소스인 경우에 제안 알고리즘과 PCRTT 알고리즘의 첨두 전송률 이용률이 100%인데, 이것은 구간의 개수가 1로 설정되었기 때문이다.

(그림 20)은 네트워크 트래픽의 변동으로 인해 서버에서 계산된 전송률보다 클라이언트에서 프레임을 빨리 받거나 늦게 받아서 QoS를 보장할 수 없는 프레임의 개수 비교로써, 이 값은 계산되는 전송률이 언더플로우 경계선 또는 오버플로우 경계선과 만나는 프레임의 개수로 계산하였다. 제안 알고리즘에서 전송률 설정 방법은 두 개의 경계선과 만나는 프레임이 없으므로 0이 된다. MVBA 알고리즘에서의 언더플로우 또는 오버플로우가 발생될 수 있는 프레임의 개수는 전송률 변화 횟수와 동일한데, 이것은 전송률이 언더플로우 경계선 또는 오버플로우 경계선에서 변화되기 때문이다.

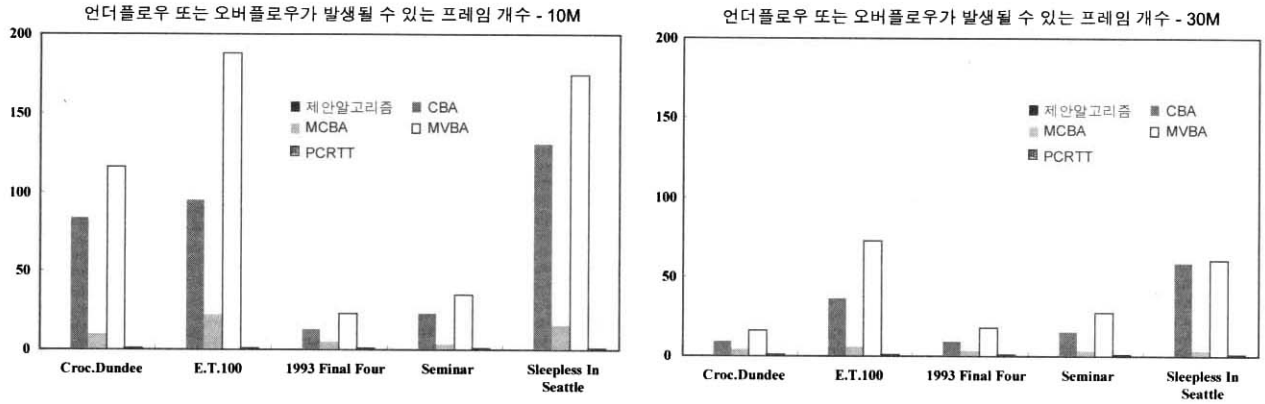
(그림 21)은 계산 시간 비교로써 단위는 msec이다. 전송률 계산에 소요되는 계산 시간은 비디오 서버가 다수의 클라이언트에게 비디오를 제공하고 디스크와 버퍼 등의 자원을 관리해야 하므로 작아야 한다[20, 30]. 제안 알고리즘의 계산 시간은 PCRTT, e-PCRTT 알고리즘과 비슷하고, MVBA, CBA, MCBA 보다는 작다. CBA, MCBA, MVBA 알고리즘들은 연장 구간에서 알고리즘의 목적에 적합한 프레임의 검색에 요구되는 검색 시간이 있고 다음 런의 시작 프레임으로 검색된 프레임부터 연장 구간의 끝 프레임에 속한 프레임을 새로운 전송률을 계산하기 위해 다시 검색해야 하기 때문이다.



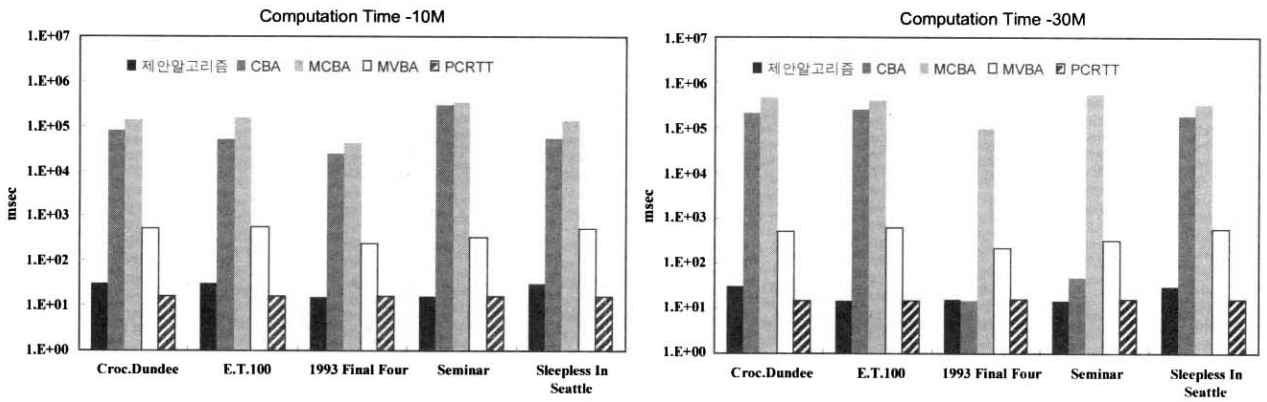
(그림 18) 버퍼 이용률 비교



(그림 19) 첨두 전송률 이용률 비교



(그림 20) 언더플로우 또는 오버플로우가 발생될 수 있는 프레임 개수 비교



(그림 21) 계산 시간 비교

5. 결론 및 추후 연구 방향

버퍼가 커지는 PCRTT-휴리스틱 알고리즘의 단점을 개선한 e-PCRTT 알고리즘은 전송률 변화 횟수가 주어져서 구간의 크기가 일정한 특징을 갖고 있다. 따라서, 전송률 변화 횟수가 많게 주어지면 구간의 크기가 작게 구성되어 불필요하게 전송률을 변화시켜야 할 경우가 발생될 수 있으며, 전송률 변화 횟수가 적게 주어지면 구간의 크기가 크게 구성되어 주어지는 버퍼 크기로 QoS를 보장하지 못할 수도 있다.

본 논문에서는 이러한 e-PCRTT 알고리즘에서의 문제들을 해결하기 위해 런의 시작 프레임의 언더플로우 경계점과 오버플로우 경계점의 평균점을 시작점으로 하고 검색중인 프레임의 언더플로우 경계점과 오버플로우 경계점의 평균점을 종점으로 연결하여 이를 전송률로 설정하며, 이 전송률이 QoS를 보장한다면 검색중인 프레임을 현재 런에 추가하고, QoS를 보장하지 않는다면 검색중인 프레임을 새로운 전송률을 갖는 런에 배정함으로써 런의 크기가 가변적이고 전송률 변화 횟수의 제한이 없는 스무딩 알고리즘을 제안하였다. 제안 방법의 성능은 전송률 변화 횟수, QoS를 유지하기에 요구되는 최소 버퍼 크기, 첨두 전송률, 전송률 변화량등의 평가 요소들을 사용하여 e-PCRTT 알고리즘과 비교하였으며, 대부분의 평가 요소에 의해 제안 알고리즘이 우수함을 보였다.

또한, 다른 스무딩 알고리즘들과 시간 복잡도, 전송률 변화 횟수, 첨두 전송률, 전송률 변화량, 버퍼 이용률등을 비교 분석하였다. 제안 알고리즘은 시간 복잡도에서는 다른 스무딩 알고리즘들 보다 같거나 낮으면서 전송률 변화 횟수, 계산 시간, 그리고 네트워크 트래픽의 변동에 대한 유연성에서 우수하다. 따라서, 제안 스무딩 알고리즘은 전송률 계산 시간이 적게 요구되어야 하는 서버 또는 서버와 네트워크간의 전송률 변화로 인한 통신 오버헤드를 감소시키고 서버에서 계산되는 전송률로 비디오 스트림을 전송하지 못할 가능성이 있는 네트워크에서 도움이 될 것이다. 추후에는 네트워크 트래픽의 변동이 발생되었을 경우를 고려하는 전송률 조절 알고리즘에 대해 연구 할 예정이다.

참 고 문 헌

[1] D. Le Gall, "MPEG : A video compression standard for multimedia applications," Communications of the ACM, Vol.34, pp.47-58, April, 1991.

[2] J. Zhang and J. Hui. "Applying traffic smoothing techniques for quality of service control in VBR video transmissions," Computer Communications, pp.375-389, April, 1998.

[3] J. Zhang and J. Y. Hui, "Traffic Characteristics and Smoothness Criteria in VBR Video Traffic Smoothing," in Proc. of the ICMC and Systems, June 3-6, Ottawa, Ontario, Canada, IEEE Computer Society, 1997.

[4] P. Thiran, et al., "Network calculus applied to optimal multimedia smoothing," in INFOCOM, 2001.

[5] Sanjay G, Rao, S. V. Raghan, "Fast Techniques for the Optimal Smoothing of Stored Video," Multimedia Systems, 7, pp.222-233, 1999.

[6] W. Feng, F. Jahanian, S. Sechrest, "An Optimal Bandwidth Allocation Strategy for the Delivery of Compressed Prerecoded Video," ACM/Springer-Verlag Multimedia Systems, Vol.5, No.5, pp.297-309, Sept., 1997.

[7] J. Kwak, M. Lee, H. Song and D. Park, "An Effective Smoothing Algorithm of VBR Video Stream over Internet," Proceedings of 29th KISS Spring Conference, Vol.1, pp. 436-438, April, 2002.

[8] W. Feng, "Rate-constrained bandwidth smoothing for the delivery of stored video," in SPIE Multimedia Networking and Computing, 1997.

[9] M. Lee, J. Kwak, H. Song, D. park, "An Efficient Transmission Rate Control Algorithm for MPEG VOD Services," Journal of the KCIES, Vol.3, No.8, August, 2002.

[10] W. Feng, S. Sechrest, "Critical Bandwidth Allocation for the Delivery of Compressed Prerecorded Video," Computer Communications, Vol.18, No.10, pp.709-717, Oct., 1995.

[11] W. Feng, et al., "Smoothing and buffering for delivery of prerecorded compressed video," in Proc. of ISET/SPIE Symp., on Multimedia Comp. and Networking, pp.234-242, Feb., 1995.

[12] Han-Chieh Chao, C. L. Hung, "Efficient Changes and Variability Bandwidth Allocation for VBR Media Streams," IEEE, 2001.

[13] J. D. Salehi, et al., "Supporting stored video : Reducing rate variability and end-to-end resource requirements through optimal smoothing," in Proc. of ACM SIGMETRICS, pp. 222-231, May, 1996.

[14] J. McManus and K. Ross, "Video on demand over ATM : Constant-rate Transmission and Transport," in proc. of ACM SIGMETRICS, pp.222-231, May, 1996.

[15] Ray-I chang, Meng-Chang Chen, Jan-Ming Ho and Ming-Tat Ko, "Schedulable Region for VBR Media Transmission with Optimal Resource Allocation and Utilization," infsci(1~2), pp.61-79, 2002.

[16] Ellen W. Zegura, Scott McFarland, "A Survey and New Result in Regoniated Service," Journal of High Speed Networks, Vol.6, No.3, 1997.

[17] Jean M. McManus, "A Dynamic Programming Methodology for Managing Prerecorded VBR Sources in Packet-Switched Networks," Telecommunication Systems, Vol.9, 1998.

[18] W. Feng and J. Rexford. "Performance evaluation of smoothing algorithms for transmitting prerecorded VBR video," IEEE Trans. on Multimedia, September, 1999.

[19] Ofer Hadar Reuven Cohen, "PCRTT Enhancement for Off-Line Video Smoothing," The Journal of Real Time Imaging, Vol.7, No.3, pp.301-314, June, 2001.

[20] Wu-chi Feng, "A Comparison of Bandwidth Smoothing Techniques for the Transmission of Prerecorded Compressed Video," in Proc. IEEE INFOCOM, pp.58-66, April, 1997.

[21] W. Feng, et al., "Optimal Buffering for the Delivery of Compressed Prerecorded Video," Proc. of the IASTED/ ISMM Intl Conference on Networks, January, 1995.

[22] J. D. Salehi, "Scheduling Network Processing on Multimedia and Multiprocessor Servers," Ph. D. dissertation, Univ., Massachusetts, Amherst, Sept., 1996.

[23] J. M. McManus et al., "Video-on-Demand Over ATM : Constant-Rate Transmission and Transport," in IEEE Journal on selected areas in comm., Vol.14, No.6, August, 1996.

[24] Wu-Chi Feng, "VIDEO-ON-DEMAND SERVICES : EFFICIENT TRANSPORTATION AND DECOMPOSITION OF VARIABLE BIT RATE VIDEO," PH. D, Computer Science and Engineering, 1996.

[25] http://www.cis.ohio-state.edu/~wuchi.

[26] K. R. Rao, Zoran S. Bojkovic, "Multimedia Communication Systems Technique, Standards and Networks," Prentice Hall, 2002.

[27] Han-Chelh Cha, C. L Hung and T. G Tsuel, "ECVBA Traffic Smoothing Scheme for VBR Media streams," International Journal of Network Management, pp.179-185, 2002.

[28] Kong Chun Wai and Jack Y. B. Lee, "Slice-and-Patch - An Algorithm to Support VBR Video Streaming in a Multicast-based Video-on-Demand System," Journal of Information Science and Engineering, Vol.19, No.3, May, 2003.

[29] O. Hadar S. Greenberg, "Statistical multiplexing and admission control policy for smoothed video streams using e-PCRTT algorithm," International Conference on Information Technology : Coding and Computing, March, 2000.

[30] D. Viswanathan, A. Nahrstedt, K. Dongyan Xu.Wichadakul, "Impact of CPU reservation on end-to-end media data transmission," IEEE International Conference on Performance, Computing and Communications, April, 2001.

이 면 재

e-mail : leemj@cs.hongik.ac.kr
1992년 홍익대학교 전자계산학과(학사)
1994년 홍익대학교 전자계산학과(석사)
1994년~1999년 정원엔 시스템 연구소 근무
2000년~2004년 용인송담대학 정보통신학과 겸임교수
2001년~2002년 코리아 홈넷 수석 연구원
2002년~2003년 아라마루 수석 연구원
2002년~현재 홍익대학교 전자계산학과 박사과정 수료
2004년~현재 한국산업기술대학교 겸임교수
관심분야 : 멀티미디어 통신, 멀티미디어 데이터 방송, 설계 자동화

이 준 용

e-mail : jlee@cs.hongik.ac.kr
1986년 서울대학교 컴퓨터 공학과(학사)
1988년 미국 University of Minnesota 대학원, Dept.of Computer Sci, & Eng(석사)
1996년 미국 University of Minnesota 대학원, Dept.of Computer Sci, & Eng(박사)
1996년~1997년 미국 IBM Staff 연구원
1997년~현재 홍익대학교 컴퓨터공학과 교수
관심분야 : VLSI Design, CAD, 컴퓨터 구조

박 도 순

e-mail : dspark@cs.hongik.ac.kr
1978년 서울대학교 전자공학(학사)
1980년 한국과학기술원 전산학과(석사)
1988년 고려대학교 수학과(박사)
1980년~1983년 국방과학연구소 연구원
1983년~현재 홍익대학교 컴퓨터공학과 교수
관심분야 : 컴퓨터 구조, 설계 자동화