씬클라이언트 컴퓨팅 환경에서 미디어 스트리밍의 화질 개선을 위한 연구

김병길° 이좌형 정인범
강원대학교 컴퓨터정보통신공학과
{bgkim°,jhlee}@snslab.kangwon.ac.kr, ibjung@kangwon.ac.kr

A Study on Improving the Video Quality of Media Streaming on Thin-Client Computing

Byeonggil Kim° Joahyung Lee, Inbum Jung
Dept. Computer Information & Telecommunication Engineering

요 약

서버기반 컴퓨팅을 가능하도록 해주는 씬클라이언트 컴퓨팅에 관한 연구와 상품화는 지속적으로 진행되어 오고 있다. 하지만 지금까지 개발되어 온 씬클라이언트는 단순히 텍스트 위주의 컴퓨팅 성능만을 고려하여 개발될 뿐 게임 등 멀티미디어에 대한 성능은 고려하지 않는다. 따라서 기존의 씬클라이언트 컴퓨팅 환경에서의 미디어 재생은 화질이 너무 열악하여 정상적으로 미디어 서비스를 즐기기에는 많이 부족한 실정이다. 본 논문은 리눅스를 기반으로 한 씬클라이언트 컴퓨팅 환경에서 미디어를 재생했을 때 기존 씬클라이언트 컴퓨팅에 비해 화질(Video quality)을 향상시키는 연구를 진행하였다. 제안된 방법을 씬클라이언트 컴퓨팅의 대표적인 기술인 VNC(Virtual Networking Computing)와 성능 비교한 결과 동일한 대역폭 사용으로 로컬 재생 수준의 화질을 보여주는 성능 향상이 가능하였다.

1. 서 론

씬클라이언트 컴퓨팅 시스템은 서버-클라이언트 구조로 하나의 서버에 다수의 사용자가 터미널 에뮬레이터를 통해 접속하여 서버에 가상의 작업 공간을 만들어 각각 독립적으로 프로그램을 실행할 수 있도록 하는 컴퓨팅 방식을 말한다. 이때 클라이언트 모니터에 디스플레이 될 데이터를 네트워크 상에서 주고받기 위하여 RDP(Remote Display Protocol) 프로토콜을 사용한다. RDP는 클라이언트에서 발생된 키보드와 마우스의 이벤트 정보를 서버에게 보내며 서버는 클라이언트로부터 받은 이벤트를 처리하여 업데이트 된 사용자 인터페이스의 스크린 정보를 클라이언트에게 전송한다. 이러한 RDP는 단지 텍스트 데이터만 클라이언트로 전송하는 과거의 더미 터미널 컴퓨팅 환경에서 그래픽 요소를 추가하여 전송하는 진보된 중앙 집중식 씬클라이언트 프로토콜이라고 할 수 있다.

오늘날 멀티미디어 기능은 컴퓨터 사용에 있어 필수적인 요소가 되었으며 씬클라이언트 컴퓨팅 환경 또한 이러한 기능의 지원은 필수적이라 하겠다. 하지만 현시점에서의 씬클라이언트 컴퓨팅 환경은 여전히 인터넷 검색이나 문서 작성과 같은 움직임이 정적인 텍스트 기반의 컴퓨팅 환경에 대해서만 연구가 진행 중에 있으며 이에 반해 멀티미디어처럼 많은 움직임이 존재하는 데이터를 클라이언트에 전송해서 화면의 끊어짐 없이 보여주도록 하는 연구는 아직 부족한 실정이다[7]. 따라서 미디어 재생이나 빠른 움직임을 포함하고 있는 플래쉬를 씬클라이언트 환경에서 동작시킨다면 정상적으로 재생되는 모습들을 보여주지 못하는 형편이다

본 논문에서는 씬클라이언트 컴퓨팅 환경에서 미디어 재생기를 실행하여 업데이트 되는 스크린 데이터를 실시간으로 전송했을 때 클라이언트 모니터에 보여 지는 미디어 재생 화면의 화질 열화 현상을 규명하고 이를 제거하기 위한 연구를 진행하였다. 본 논문에서 제안한 방법에 의해 화질이 개선된 연구 성능 결과를 증명하기 위해 AT&T에서 개발된 RealVNC 소프트웨어를 사용하여 성능 비교를 하였다.

본 논문의 구성은 다음과 같다. 2장에서는 본 논문과 관련된 연구에 대하여 설명한다. 3장에서는 제안하는 시스템에 대해 간략하게 설명한 후 4장에서는 시스템과 실험환경에 대해 설명한 후 측정 결과 그래프들을 보여줄 것이며 성능 측정에 대한 결과를 비교분석하여 설명한다. 5장에서는 본 논문의 결론을 맺고 향후 연구 계획을 설명한다.

2. 관련 연구

씬클라이언트 컴퓨팅 환경을 기반으로 웹 브라우징을 하기 위한 성능향상이나 이를 대역폭이 제한된 광 대역 네트워크 망에서 효율적으로 사용하기 위한 연구가 진행 중에 있다. 최근에는 휴대폰이나 PDA, 태블릿 PC등 모바일 기기의 발전으로 인하여 무선 환경에서의 씬클라이언트 컴퓨팅 연구가 시작되고 있는 추세이다[7][11]. 이러한 씬클라이언트 컴퓨팅 환경을 구성하는 소프트웨어로는 Citrix MetaFrame[8], Microsoft Terminal Services[9], AT&T Virtual Network Computing (VNC)[1] 그리고 Tarantella[10] 등이 있다. 이러한 것들은 독창적으로 개발되어 무료나 상업적으로 배포되거나 일부 소프트웨어는 하드웨어와 결합이 되어 하나의 완전한 씬클라이언트 시스템으로 판매가 이루어지고 있다. 하지만 이러한 소프트웨어들은 텍스트 기반 중심의 연구가 진행된 것들이기 때문에 컴퓨터를 이용하는 많은 사용자들에게 관심의 대상이 되는 미디어 재생 응용프로그램을 다루기에는 최적화가 되어 있지 않아 불편한 요소들이 많이 존재한다.

3. 제안하는 시스템

3.1 개발 환경

본 논문에서 사용한 개발 플랫폼과 서버 시스템의 사양은 [6]에서 제안한 시스템을 바탕으로 플랫폼은 리눅스를 채택하였으며 서버 사양은 AMD 듀얼 CPU의 준 서버급 시스템을 사용하였다. 개발에 사용된 소프트웨어는 리눅스 기반에서 미디어 재생에 최적화가 되어 있고 가장 널리 사용되고 있는 MPlayer(버전 0.90pre4)를 소스 수정하여 개발하였다.

* 본 연구는 강원대학교 ITRC의 지원으로 수행되었습니다.
* 본 연구는 한국과학재단 목적기초연구(R05-2003-000-12146-0)지원으로 수행되었습니다.

3.2 *iMedier*

iMedier(Intelligent Media Player)는 본 논문에서 제안하는 시스템의 이름이다. 전체적인 구조는 서버와 클라이언트 구조로써 구성요소로는 iMedier 서버와 iMedier 클라이언트로 구성되어지며 그림 1은 제안하는 iMedier의 전체적인 시스템 구조도의 모습을 보여준다.

3.3 시스템 구조

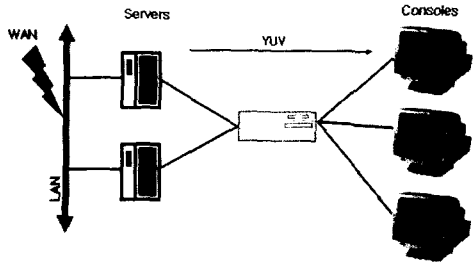

그림 1. iMedier 시스템 구조도

씬클라이언트 컴퓨팅 시스템은 서버에서 생성된 많은 양의 스크린 데이터를 클라이언트에게 전송하게 되는데 이때 전송되는 데이터의 포맷은 RGB 형태의 데이터이다. 하지만 RGB 데이터는 압축이 전혀 이뤄지지 않은 색차 공간 포맷 형태로 초당 전송되는 데이터의 양이 많기 때문에 네트워크 병목 현상을 일으키는 주된 요인이 된다[6]. 이에 반해 iMedier는 색차 공간 포맷을 YUV 형태의 4:2:0 포맷으로 전송함으로써 대역폭 사용을 현저히 낮추었다. 또한 영상 프레임의 공간적인 중복성을 제거하여 압축률을 높이기 위해 허프만 알고리즘을 사용하여 데이터를 압축하여 전송한다.

그림 2는 서버로부터 받은 미디어 비트스트림을 클라이언트에서 디코딩 하기 위한 미디어 데이터 처리단계를 보여주고 있다. 왼쪽의 그림은 일반적인 미디어 비트스트림의 처리단계이며 오른쪽의 그림은 제안한 iMedier에서의 디코딩 단계를 보여주고 있다. 일반적으로 미디어 스트림을 디코딩 하기 위해서는 서버에서 받은 패킷을 분리 작업하는 demuxing 단계를 거쳐 encoding 된 데이터를 디코더에서 decoding 연산을 수행하여 YUV 데이터를 얻어낸다. 이후에 RGB 데이터를 얻기 위해 색차 포맷 변환작업을 한 후 X 윈도우 라이브러리를 호출하여 영상을 화면에 보여준다. 이에 반해 iMedier 시스템에서는 서버에서 디코딩 작업을 마친 YUV 형태의 데이터를 클라이언트에게 전송하여 단지 색차 포맷 변환 작업만을 거친 후 X에게 보내 영상 화면에 출력하도록 하고 있다. 이러한 이유로 자원의 제약이 주어지는 임베디드 시스템 환경에서 미디어 데이터를 디코딩 해야 하는 부담이 적어지기 때문에 추가적으로 요구되는 디코더 칩이 iMedier 클라이언트 구조체계에서는 불필요하다는 장점이 있다

4. 성능 측정

씬클라이언트 환경에서 미디어를 상영했을 때 화질의 개선 정도를 평가하기 위해 iMedier 시스템에 대해 성능 측정을 하였으며 기존의 VNC 시스템과 성능 비교를 하였다.

4.1 시스템 및 실험 환경

씬클라이언트 컴퓨팅 환경을 구성하기 위해 서버 측에는 iMedier 서버를 작동시키고 클라이언트에서는 터미널 에뮬레이터 역할을 하는 iMedier 클라이언트를 작동시켜 서버에 접속하

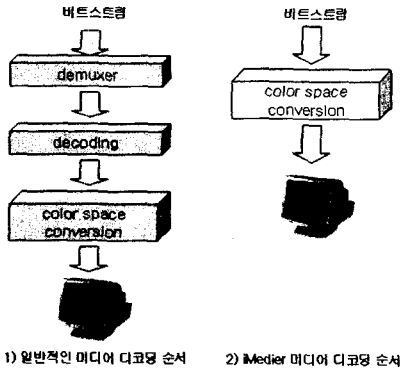

그림 2 미디어 디코딩 순서도

도록 하였다. 이때 상영되는 MPEG-4 미디어 정보는 표1과 같다. 측정에 사용된 미디어는 화면의 갱신 정도를 고려하여 움직임이 빈번하게 발생되는 뮤직비디오 클립을 사용하였으며 30fps의 고화질 영상을 디스플레이 하도록 하였다.

320 X 240	
30 fps	
251 s	

표 1 미디어 정보

4.2 비디오 화질 벤치마크

클라이언트에서 보여 지는 영상 화면의 화질을 측정하기 위해 1 frame/second(fps)와 30 fps 두 가지 재생비율에서 발생하는 패킷 트래픽을 모니터링 하였다. 비록 1 fps의 재생율에서 영화를 감상하는 사용자는 없지만 서버에서 클라이언트로 전송되는 완전한 데이터의 사이즈를 참조하기 위해 측정을 하였다. 마찬가지로 30 fps에 해당하는 정상재생을 하여 서버에서 클라이언트로 전송되는 패킷 트래픽을 모니터링 하여 전체 전송된 데이터 사이즈를 비교하였다. 비디오 화질은 1 fps의 프레임율에 대해 30 fps의 프레임율의 비율로 측정되어 질 수 있으며 그에 대한 수식은 아래와 같다[4].

$$VQ = \frac{\left(\frac{DataTransferred(30fps)/PlaybackTime(30fps)}{IdealFPS(30fps)}\right)}{\left(\frac{DataTransferred(1fps)/PlaybackTime(1fps)}{IdealFPS(1fps)}\right)}$$

수식 1 비디오 화질

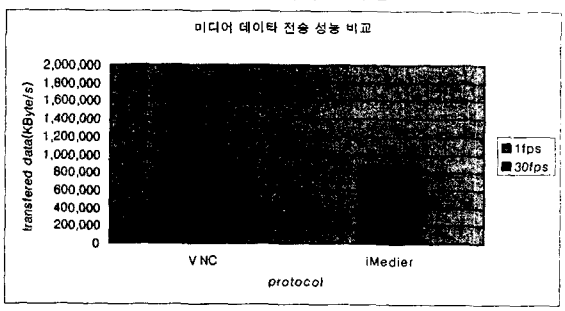

그림 3 전체 전송된 미디어 데이터 비교

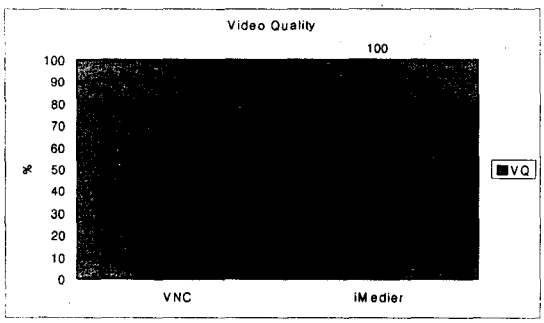

그림 4 비디오 화질 성능 비율 비교

그림 3은 1 fps와 30 fps에 대해 미디어 재생 시간동안 클라이언트로 전송된 전체 데이터 크기를 프로토콜 별로 비교한 것이다. 그림에서 보듯이 VNC 에서는 1 fps에서 전체 1.8 GBytes의 데이터를 전송했지만 iMedier는 절반의 크기인 0.8 GBytes의 데이터만을 전송했다. 이러한 수치가 나온 이유는 VNC는 RGB 형태의 데이터를 전송 하지만 본 논문에서 제안한 iMedier는 4:2:0 포맷인 YUV 데이터를 전송하기 때문에 데이터 크기가 반으로 줄어든 것이다. 정상 재생인 30 fps에서도 VNC는 1 fps에 절반에 미치지도 않는 데이터를 전송했지만 iMedier는 거의 비슷한 크기의 데이터를 전송했다. 이러한 실험 결과는 위에서 언급한 수식 1의 계산에 의해 그림 4와 같은 결과 그래프를 보여준다.

그림 4에서 VNC의 화질이 40% 밖에 되지 않는 이유는 정상 재생(30fps)에서 나온 전체 데이터 사이즈가 1fps에서 측정된 전체 데이터 사이즈의 40% 밖에 되지 않기 때문이며 VNC에서 스크린 업데이트를 위해 사용하는 방식이 미디어 재생 특성을 고려하여 설계되지 않았기 때문이다. 즉 VNC는 화면을 업데이트 하는 초당 횟수가 30 프레임 비율을 맞추지 못하며 한 화면에 대해 변경된 부분만 국소적으로 업데이트를 하기 때문이다.

그림 5는 평균 대역폭 사용량을 압축을 했을 때와 압축하지 않았을 때를 비교한 그래프이다. 압축을 하지 않았을 때는 VNC보다 iMedier가 평균 25%의 데이터를 더 많이 전송하지만 VNC는 서버에서 생성된 스크린 데이터를 미처 클라이언트에게 전송하지 못하고 버리기 때문에 낮은 대역폭 사용량을 보여준 것이다. 하지만 iMedier는 손실되는 데이터 없이 모든 데이터를 보냄에도 불구하고 대체적으로 낮은 대역폭 사용량을 보여주었으며 대역폭 사용량을 줄이기 위해 압축을 사용하여 전송하도록 하였다. MPEG 영상의 프레임에는 공간적인 중복성이 존재를 하기 때문에 일반적으로 MPEG 시스템에서는 그러한

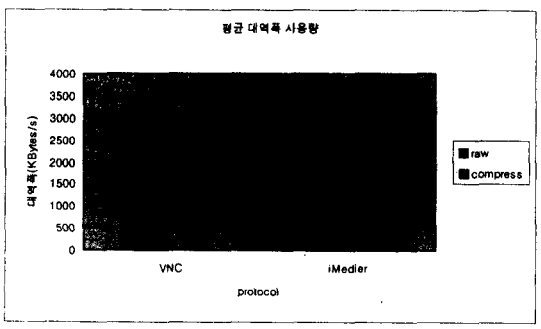

그림 5 평균 대역폭 사용량

중복성을 제거하는 압축 방법을 사용한다. iMedier 또한 공간적 중복성을 제거하기 위해 허프만 압축 알고리즘을 사용하여 데이터를 전송하도록 하였으며 CPU의 부하를 감소시키기 위해 압축률을 낮추어 최소한의 압축만 하도록 하였다. 연구 결과 그림 5에서 보다시피 압축을 한 후 VNC와 비슷한 수준의 평균 대역폭 사용량을 보여주었다.

5. 결론 및 향후 연구

본 논문에서는 씬클라이언트 컴퓨팅 환경에서 미디어 스트리밍 서비스의 향상을 위해 화질 개선에 대한 연구를 진행 하였다. 연구 결과에서 VNC가 미디어 재생 화질이 열악한 이유는 미디어의 특성을 고려하여 설계되지 않았기 때문인 것으로 규명하였으며 이에 반해 본 연구에서 제안된 iMedier는 미디어 특성을 고려한 플레이어이므로 로컬 재생과 같은 수준의 100% 화질 우수성을 보여주었다. 또한 제한된 네트워크 자원을 고려하여 전송되는 데이터의 크기를 줄임으로써 네트워크 대역폭의 사용을 최소화 하였다.

향후에는 다수의 사용자가 iMedier 서버에 접속 가능하도록 하기 위해 QoS를 고려한 병렬 디코더 서버를 제안할 계획이며 멀티미디어에 있어 음성 또한 없어서는 안 될 중요한 요소이기 때문에 음성처리에 대한 문제를 해결할 것이다. 또한 광 대역망이나 무선 환경에서 미디어 스트리밍 서비스를 위한 연구를 진행할 것이다.

6. 참고 문헌

[1] Tristan Richardson, Quentin Stafford-Fraser, Kenneth R. Wood, Andy Hopper, "Virtual Network Computing", IEEE, 1998

[2] S. Jae Yang, Jason Nieh, Matt Selsky, and Nikhil Tiwari, "The Performance of Remote Display Mechanisms for Thin-Client Computing", USENIX, 2002

[3] Brian K. Schmidt, Monica S. Lam, J. Duane Northcutt, "The interactive performance of SLIM: a stateless, thin-client architecture", ACM Symposium, December 1999

[4] S. J. Yang, J. Nieh, and N. Novik, "Measuring Thin-Client Performance Using Slow-Motion Benchmaking", USENIX, 2001

[5] Alexander Ya-li Wong, Margo Seltzer, "Operating System Support for Multi-User, Remote, Graphical Interaction", USENIX, 2000

[6] 김병길, 정인범, "멀티미디어 서비스를 위한 씬클라이언트 컴퓨팅의 성능평가 및 비교", 한국정보과학회 2003년10월

[7] S. Jae Yang, Jason Nieh, Shipa Krishnappa: Web Browsing Performance of Wireless Thin-client Computing, 2003

[8] T. W. Mathers and S. P. Genoway. Windows NT Thin Client Solutions: Implementing Terminal Server and Citrix MetaFrame. Macmillan Technical Publishing, Indianapolis, IN, Nov, 1998

[9] B. C. Cumberland, G. Carius, and A. Muir. Microsoft Wndows NT Server 4.0, Terminal Server Edition: Technical Reference, Microsoft Press, Redmond, WA, Aug. 1999.

[10] A. Tirumala and J. Ferguson. Iperf. http://dast.nlanr.net/Projects/Iperf

[11] Albert Lai, jason Nieh, "Limits of Wide-Area Thin-Client Computing", Proceedings of the ACM SIGMETRICS 2002