씬클라이언트 컴퓨팅에서 스트리밍 미디어의 QoS를 보장하는 지능형 미디어 플레이어

김 병 길[†] · 이 좌 형[††] · 정 인 범[†††]

요 약

한정된 자원을 보유하고 있는 씬클라이언트 환경 아래에서는 많은 연산량을 요구하는 MPEG 미디어의 복호화를 사용자에게 QoS를 보장되는 수준으로 동작시키기 어렵다. 이러한 문제점을 극복하기 위하여 미디어에 대한 복호화 연산은 중앙의 터미널 서버들의 자원을 이용하게 하고 씬클라이언트 쪽에서는 단지 화면 업데이트만 처리하는 방식들이 사용되어지고 있다. 그러나 제안된 기존의 방법들에서는 재생된 스트리밍 미디어의 화질이 열악한 형편이다. 더구나, 서버들에게 복호화의 전 과정을 부담시키므로 서버들이 적은 부하에도 쉽게 포화점에 도달하고 있다. 본 논문에서는 유선 및 무선 씬클라이언트 환경에서 화질의 열화가 발생되는 원인들을 규명한다. 분석된 기존 씬클라이언트 방법들의 문제점을 기반으로 미디어 화질의 질을 향상시키며 영상과 음성을 동기화를 맞추어 사용자들에게 QoS 가 보장되는 스트리밍 미디어 서비스를 제공하는 지능형 미디어 재생기를 제안한다.

키워드 : 씬클라이언트, 스트리밍 미디어, 비디오 화질, 음성 동기화, 미디어 재생기

An Intelligent Media Player for Guaranteeing QoS Streaming Media on Thin-Client Computing

Byeong Gil Kim[†] · Joa Hyoung Lee[††] · Inbum Jung[†††]

ABSTRACT

Due to the limited resources in thin-client and the large amount of computation for decoding MPEG media, it is not easy to support the QoS stream media to clients. To solve the problems, the terminal servers would be charged for decoding the MPEG media and thin-clients have a role to update only the changed areas in their screen. However, these previous approaches cause severely low video quality. In addition, since servers perform all procedures to decode MPEG media, they are easily saturated even under a small number of clients. In this paper, the sources of the low video quality are investigated in the previous thin-clients' solutions working in wireless and wired environments. From the detailed experiments, an intelligent media player is proposed to achieve the QoS streams by supporting both the enhanced video quality and the audio synchronized with video frames.

Key Words : Thin-Client, Streaming Media, Video Quality, Audio Synchronization, Media Player

1. 서 론

씬클라이언트 컴퓨팅 환경은 사용자가 모니터와 키보드만을 이용하여 터미널 에뮬레이터를 통해 서버에 접속해서 자신의 가상의 작업 공간을 만들어 각각 독립적으로 프로그램을 실행하는 방식을 말한다. 서버는 각각의 사용자들이 요청한 프로그램들을 수행하여 그 결과 화면을 클라이언트에게 전송한다. 이러한 씬클라이언트 컴퓨팅은 중앙 집중식 컴퓨팅 환경을 유지하면서 데스크 탑 컴퓨터에서 사용 가능한 작업 환경을 사용자들에게 지원해 줄 수 있다. 씬클라이언트 환경에서는 서버에서 모든 소프트웨어들을 중앙 관리하므로 추가적인 소프트웨어의 수정, 변경을 서버측에서의 처리하게 된다. 이런 환경은 소프트웨어 변화에 따른 사용자 각각에게 부담되는 추가적 비용 및 설치 시간을 줄여주므로 보다 효율적인 작업환경을 제공해줄 수 있다.

데스크탑 컴퓨팅 환경에서 그래픽 환경을 제공했을 때 씬클라이언트 컴퓨팅 또한 텍스트 기반에서 그래픽 기반 환경을 제공하였다. 최근 컴퓨팅 환경은 기존의 그래픽 컴퓨팅 환경에 멀티미디어 요소가 추가되는 컴퓨팅 환경으로 바뀌어가고 있다. 즉, 과거에는 찾고자 하는 정보를 텍스트 형태

※ 연구는 강원대학교 ITRC의 지원을 받아 수행하였음.
※ 본 연구는 학국과학재단 목적기초연구(R05-2003-000-12146-0)의 지원으로 수행되었음.
※ 2004년도 강원대학교 학술연구조성비로 연구하였음.
† 준 회 원 : 강원대학교 컴퓨터정보통신공학과 공학석사
†† 준 회 원 : 강원대학교 컴퓨터정보통신공학과 공학석사
††† 정 회 원 : 강원대학교 전기전자정보통신공학부 컴퓨터전공 교수
논문접수 : 2005년 3월 11일, 심사완료 : 2005년 7월 11일

의 웹페이지에서 찾을 수 있었다면 최근에는 필요로 하는 정보를 다양한 멀티미디어 컨텐츠들에서 얻을 수 있게 되었다. 이러한 변화된 컴퓨팅 환경에 맞추어 데스크탑 환경은 멀티미디어 컨텐츠에 대한 동작 환경을 제공하고 있지만 씬클라이언트들은 아직 그래픽 기반 환경에 머물러 있는 실정이다. 씬클라이언트는 대규모의 사용자들이 일을 하는 사무실 공간에서 효과적으로 시스템을 관리를 할 수 있는 장점을 제공하지만 멀티미디어 정보에 대한 요구사항을 충분히 만족시키지 못할 경우 클라이언트 플랫폼 시장에서 도태되는 결과를 초래할 것이다.

본 논문은 이미 데스크 탑 컴퓨팅 환경에서 제공되는 있는 멀티미디어 서비스를 씬클라이언트 환경에서도 가능하도록 하는 연구를 진행하였다. 현재까지 씬클라이언트 환경은 인터넷 검색이나 문서 작성과 같은 정적인 텍스트 기반의 작업환경을 제공하는 측면에서 연구가 진행되었다. 그러나 동영상과 같은 멀티미디어 데이터를 대규모의 씬클라이언트 사용자들에게 전송하기위해서는 아직 많은 연구가 필요하다. 특히 스트리밍 미디어는 사용자들에게 QoS가 보장되는 스트림을 전송할 경우에만 가치 있는 멀티미디어 서비스로 인정받는다. 전송되는 영상 화면 및 음성이 끊어짐 없이 매끄럽게 스트리밍 서비스되기 위해서는 사용되는 미디어의 고유 특성을 서버 및 클라이언트 양쪽에 반영하는 연구가 진행되어야한다.

한정된 자원을 보유하고 있는 씬클라이언트 환경 아래에서는 많은 연산량을 요구하는 MPEG 미디어의 복호화를 사용자에게 QoS를 보장되는 수준으로 동작시키기 어렵다. 이러한 문제점을 극복하기 위하여 미디어에 대한 복호화 연산은 중앙의 터미널 서버들의 자원을 이용하게하고 씬클라이언트 쪽에서는 단지 화면 업데이트만 처리하는 방식들이 사용되어지고 있다. 그러나 제안된 기존의 방법들에서는 재생된 스트리밍 미디어의 화질이 열악한 형편이다. 더구나, 서버들에게 복호화의 전 과정을 부담시키므로 서버들이 적은 부하에도 쉽게 포화점에 도달하고 있다. 본 논문에서는 유선 및 무선 씬클라이언트 환경에서 화질의 열화가 발생되는 원인들을 규명한다. 분석된 기존 씬클라이언트 방법들의 문제점을 기반으로 미디어 화질의 질을 향상시키며 영상과 음성을 동기화를 맞추어 사용자들에게 QoS 가 보장되는 스트리밍 미디어 서비스를 제공하는 지능형 미디어 재생기 (**iMedier** : intelligent **Medi**a Play**er**) 시스템을 제안한다.

iMedier는 서버와 사용자들 사이에 데이터 전송을 위하여 RMDP(Remote Multimedia Display Protocol) 프로토콜을 구현하였고, 변경된 영상화면에 대한 정보를 RGB가 아닌 YUV 포맷으로 구성처리 하므로 전송되는 비디오의 화질을 개선하였다. 또한 음성을 지원하기 위하여 비디오 프레임들과 복호된 음성 데이터들의 구간별 동기화 데이터를 생성하여 스트리밍 미디어와 같이 전송하여 사용자측에서 비디오와 음성 데이터들의 동기화가 가능하도록 하였다. iMedier에서는 복호된 음성과 영상의 동기화가 유지되므로 스트리밍 미디어의 재생이 데스크 탑의 로컬 재생과 같은 수준으로 제공되어 질수 있다. 본 논문에서는 무선 씬클라이언트 환경에서 iMedier를 구현하여 기존의 씬클라이언트 방식들보다 개선된 특성 및 성능을 평가한다.

본 논문의 구성은 다음과 같다. 2장은 기존에 제공되는 씬클라이언트 방법들에 대한 관련 연구를 기술한다. 3장에서는 본 논문에서 제안하는 iMedier 시스템에 대하여 설명한다. 4장은 본 논문에서 사용된 실험환경 및 성능 측정 기준에 대하여 설명한다. 5장에서는 iMedier의 성능 평가를 한다. 끝으로 6장에서 본 논문에 대한 결과 및 향후 연구에 대하여 논의한다.

2. 관련 연구

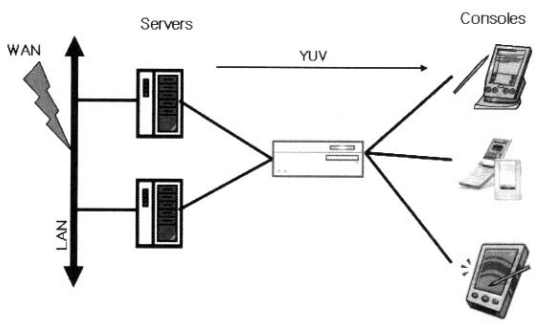
고성능의 집중된 자원을 네트워크를 통하여 그래픽 사용자 환경으로 사용할 수 있는 VNC 시스템은 이미 많은 연구가 이루어져 활성화가 되어 있으며 상업용으로 개발하여 이미 판매가 이루어지는 제품이 있다[1, 2]. 또한 씬클라이언트 컴퓨팅 환경을 기반으로 웹 브라우징을 하기 위한 성능 향상이나 이를 대역폭이 제한된 광 대역 네트워크 망에서 효율적으로 사용하기 위한 연구가 진행 중에 있으며 저사양의 씬클라이언트 시스템 기반에서 VNC를 이용했을 때 성능 제약으로 작용하는 원인을 분석하여 VNC의 전체 성능을 지배하는 인자들을 알아내고자 하는 연구도 있다[3].

최근에는 휴대폰이나 PDA, 태블릿 PC등 모바일 기기의 발전으로 무선 환경에서의 씬클라이언트컴퓨팅 연구가 시작되고 있는 추세이다[4]. 이러한 씬클라이언트 컴퓨팅 환경을 구성하는 소프트웨어로는 Citrix MetaFrame[5], Microsoft Terminal Services[6], AT&T Virtual Network Computing (VNC)[1] 그리고 Tarantella[7] 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. 또한 많은 관심의 대상이 되는 무선 환경의 모바일 기기에서 컴퓨팅이 가능하도록 하는 연구는 이루어지지 않고 있다.

3. 지능형 미디어 플레이어 시스템

3.1 시스템 구조

iMedier(intelligent Media Player)는 본 논문에서 제안하는 씬클라이언트 컴퓨팅 환경에서 QoS가 개선된 스트리밍 미디어의 재생을 가능하게 해주는 시스템의 이름이다. 전체적인 구조는 iMedier 서버와 iMedier 클라이언트로 구성되어지며 (그림 1)은 제안하는 iMedier의 전체적인 시스템 구조도의 모습을 보여준다. 서버는 하나 이상의 컴퓨터로 구



(그림 1) iMedier 시스템 구조도

성되어 질 수 있으며 각각의 서버마다 씬클라이언트 환경을 구성해 주는 iMedier 서버가 구동된다. iMedier 서버의 역할은 접속한 클라이언트마다 가상의 작업 공간을 할당하여 독립적으로 프로그램을 구동시킬 수 있도록 하며 클라이언트로부터 미디어 재생 요청이 있을 시 해당 미디어를 디코딩하여 영상 데이터를 클라이언트에게 전송하는 역할을 담당한다. 이용 가능한 클라이언트 시스템은 일반 데스크 탑 환경을 포함하여 무선 환경에서 모바일 단말을 이용하여 씬클라이언트 환경을 구축할 수 있다. iMedier 클라이언트는 iMedier 서버에 접속하기 위한 터미널 에뮬레이터 역할을 하며 서버로부터 받은 영상 데이터를 화면에 보여준다.

본 논문에서는 사용된 시스템 환경은 리눅스 기반으로 진행하였다. 연구에 사용된 미디어 재생기 소프트웨어는 리눅스 기반에서 미디어 재생에 최적화가 되어 있고 가장 널리 사용되고 있는 MPlayer(버전 0.90pre4)를 기반으로 하였고, 본 논문에서 제안된 기법들을 추가 및 수정하여 iMedier 시스템을 개발하였다.

3.2 원격 멀티미디어 재생 프로토콜

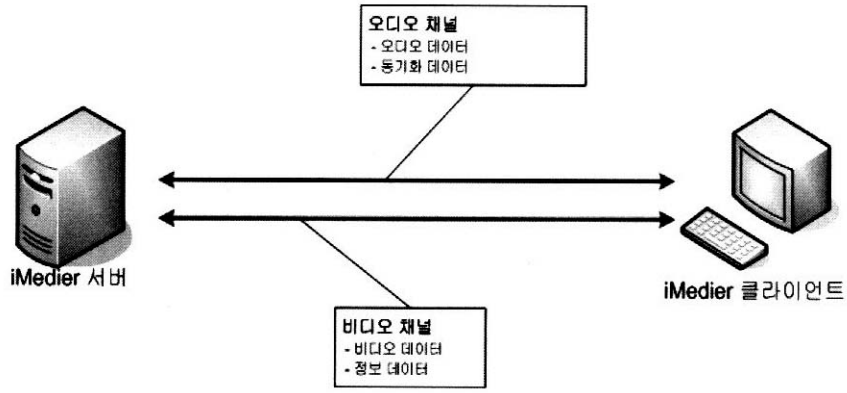
RMDP(Remote Multimedia Display Protocol)는 iMedier 시스템에서 사용하는 전송 프로토콜이다. RMDP는 서버에서 디코딩되어 화면에 보여 질 영상 화면과 음성 데이터를 클라이언트에게 전송해 주며 동기화를 위한 정보 및 영상 데이터의 각종 속성 정보들을 클라이언트에게 전송해 주는 역할을 담당한다. (그림 2)는 RMDP의 채널 관계를 보이고 있다. RMDP는 업데이트 데이터를 전송하기 위해 음성과 영상 채널을 연결하여 통신을 한다. 음성 채널은 오디오 데이터와 동기화 데이터를 전송하는 채널이며 영상 채널은 비디오 데이터와 미디어 정보 데이터를 전송하는 채널이다. 하나의 채널만을 이용할 경우 프로토콜 설계가 복잡해져 관리의 어려움이 있지만 속성이 다른 그룹으로 묶어 두 개의 채널을 사용함으로써 프로토콜 설계가 간단해 지는 장점이 있다.

3.2.1 iMedier 화면 업데이트 정책

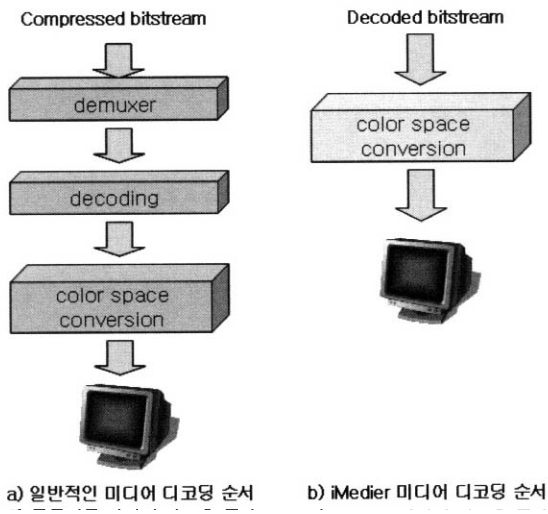
일반적으로 씬클라이언트 환경에서 터미널 서버는 생성된 업데이트 데이터를 클라이언트에게 전송하게 되는데 이때 클라이언트로 전송되는 데이터의 포맷은 RGB 형태의 데이터이다. RGB 데이터는 압축이 전혀 이뤄지지 않은 색차 공간 포맷 형태로 초당 전송되는 데이터의 양이 많다. 이러한 이유로 기존 씬클라이언트 환경에서는 RGB 형태의 데이터를 전송하기 때문에 네트워크 병목 현상을 일으키는 주된 요인이 된다. 이런 문제점들을 해결하기 위하여 iMedier 시스템에서는 색차 공간 포맷을 YUV 형태의 4:2:0 포맷으로 전송함으로써 대역폭 사용을 현저히 낮추었다. YUV 포맷은 밝기 신호인 Y와 색차 신호인 U, V를 분리한 색차 포맷으로 인간의 눈에 민감한 밝기 신호는 손실 없이 그대로 전송을 하고 이와 반대로 색에 둔감한 인간의 시각 특성을 고려하여 색차 신호 데이터를 1/4로 줄임으로써 결과적으로 전체 데이터의 절반을 감소시키도록 하였다.

일반적으로 MPEG 영상의 인코딩이나 디코딩 처리 시에는 RGB 포맷의 데이터를 YUV 포맷의 데이터로 변환을 한 후에 처리가 되며 완료시 화면에 보여주기 위해 다시 YUV에서 RGB 형태로 변환을 하여 그래픽 처리를 통해 화면에 보여주고있다. 이러한 MPEG 미디어의 특성을 이용하여 iMedier 시스템의 서버는 MPEG 미디어의 디코딩 처리를 YUV 형태까지만 진행한 후 추출된 YUV 데이터를 클라이언트로 전송한다. RGB 데이터로 변환하는 과정을 생략하므로 서버의 연산 처리량을 최소한으로 경감시켰다.

(그림 3)은 서버로부터 받은 미디어 스트림을 클라이언트에서 디코딩하기 위한 미디어 데이터 처리단계를 보여주고



(그림 2) RMDP 채널 관계



(그림 3) 미디어 디코딩 순서

있다. 왼쪽의 그림은 일반적인 컴퓨팅 환경에서의 미디어 비트스트림의 처리단계이며 오른쪽의 그림은 본 연구에서 제안하는 씬클라이언트 환경에서의 디코딩 단계를 보여주고 있다.

일반적으로 미디어 스트림을 디코딩하기 위해서는 서버에서 받은 패킷을 분리 작업하는 demuxing 단계를 거쳐 음성과 영상 각각 인코딩된 데이터를 디코더에서 디코딩 연산을 수행하여 최종적으로 화면에 보여줄 YUV 데이터를 얻어낸다. 이후에 RGB 데이터를 얻기 위해 색차 포맷 변환작업을 한 후 해당 영상 출력 라이브러리(X11, SDL, OSS, Frame Buffer 등)를 호출하여 영상 데이터를 화면에 보여준다.

이에 반해 제안하는 iMedier 시스템에서는 서버에서 디코딩 작업을 마친 YUV 형태의 데이터를 추출하여 클라이언트에게 전송하며 클라이언트는 수신 받은 데이터를 단지 색차 포맷 변환 작업만을 거친 후 해당 출력 라이브러리를 호출하여 영상 화면을 갱신하도록 하고 있다.

이러한 과정은 자원의 제약이 주어지는 모바일 환경에서 복잡한 코덱으로 인코딩 되어 있는 미디어 데이터를 디코딩 해야 하는 부담을 덜어주는 장점이 있다. 또한 디코딩 하는 코덱이 별도로 필요하지 않기 때문에 해당 기기에 코덱이 없어도 재생이 가능하다. 최근 모바일 기기에 디코더 칩을 장착하여 영상 재생이 가능하도록 하드웨어 구성이 이뤄지지만 제안하는 iMedier 시스템에서는 추가적인 하드웨어 비용을 들이지 않고도 재생이 가능한 장점이 있다.

3.2.2 데이터 압축

영상 미디어를 데이터 압축 없이 전송할 경우 많은 양의 데이터가 전송되어야 하므로 네트워크 대역폭에 제한을 받을 것이다. 이러한 이유로 표준 영상 포맷인 MPEG은 다양한 압축을 사용하여 전송이 되는 데이터의 크기를 줄이고자 노력한다. 일반적으로 영상 프레임은 초당 25프레임을 전송하기 때문에 프레임 간에는 공간적인 중복성이 존재를 하게 되는데 이를 이용하여 압축을 하는 다양한 압축 형태들이 연구되고 있다.

iMedier 시스템에서는 전송되는 데이터 크기를 줄이기 위해 공간적인 중복성 제거 알고리즘 중 가장 많이 사용되고 있는 허프만 알고리즘을 사용하여 전송되는 화면 갱신 데이터의 크기를 줄이고자 시도했다. iMedier 시스템은 화면 갱신 데이터를 클라이언트에게 전송하기 전에 두 번의 압축 과정을 수행한다. 첫째, 미디어 영상을 얻어내는 과정에서 YUV 4:2:0 포맷을 사용함으로써 데이터 크기를 절반 가까이 줄였다. 두번째로 네트워크 전송과정에서 허프만 알고리즘 사용하여 데이터 크기를 압축하여 전송하도록 하여 네트워크 대역폭사용을 감소하도록 하였다.

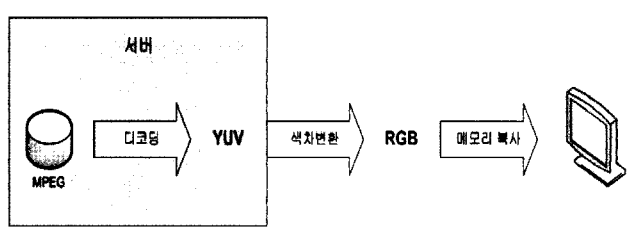
3.3 이동 단말기 환경

데스크탑 컴퓨터에서 iMedier 클라이언트의 영상 출력은 X 라이브러리를 이용한다. 하지만 대부분의 이동 단말기는 특정 데스크탑 환경을 이용하거나 프레임 버퍼를 통해 영상을 화면에 출력한다. X 라이브러리를 사용하여 영상을 출력하는 작업은 연산 작업이 많이 소요되므로 자원의 제약이 있는 모바일 기기에서는 비효율적이다. 하지만 프레임 버퍼는 어떠한 모바일 단말에서도 지원이 가능하므로 이동 단말기에 독립적이다. iMedier 클라이언트는 이동 단말기를 위해 서버로부터 받은 화면 갱신 데이터를 수신하여 갱신 내용을 프레임 버퍼에 기록하여 처리하도록 구현되었다.

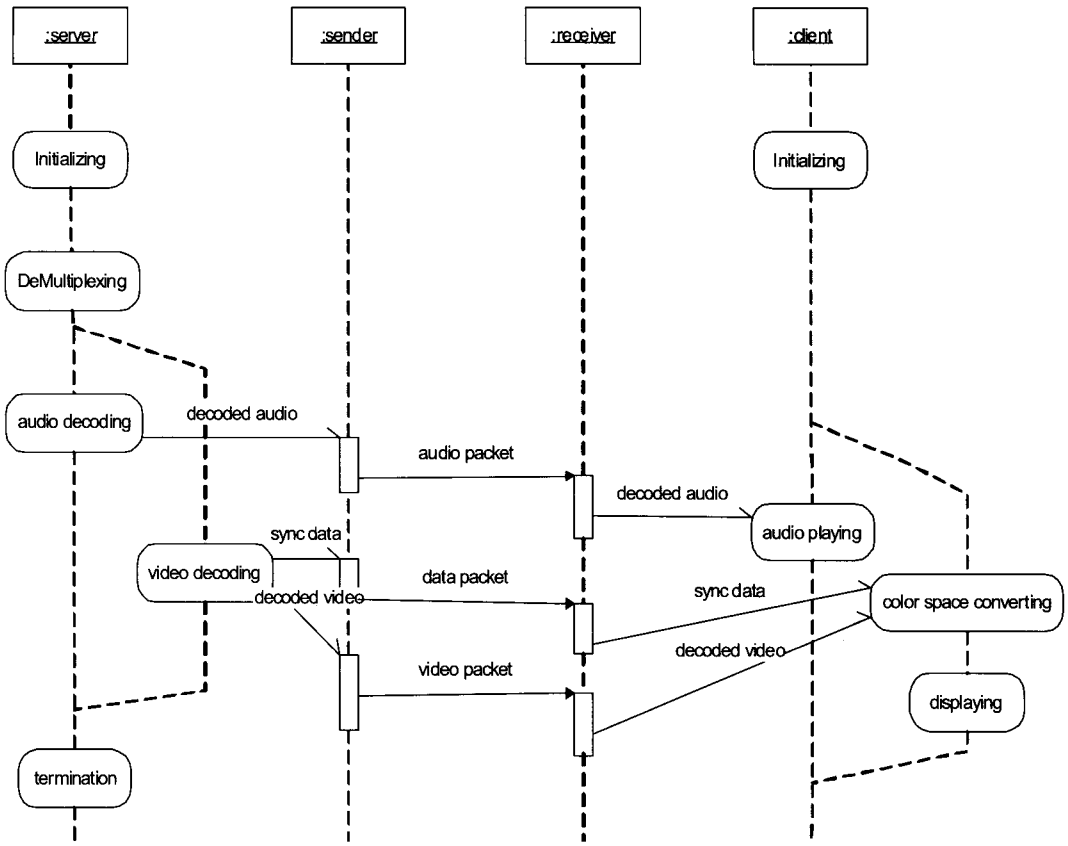
(그림 4)는 iMedier 서버가 디스크로부터 미디어 데이터를 읽어 iMedier 클라이언트의 화면에 재생되기까지의 과정을 나타낸다. 메모리 복사는 프레임 버퍼에 영상 데이터를 복사하는 과정을 말한다. 그림에 나타난 것처럼 프레임 버퍼는 디스플레이 영역에 화면에 보여줄 데이터를 직접 기록하는 기능만을 가지고 있으므로 서버로부터 받은 YUV 영상 데이터를 그대로 이용할

수는 없다. 따라서 YUV 영상 데이터를 수신 후 RGB 포맷으로 변환을 한 후에 프레임 버퍼 공간에 기록함으로써 갱신 내용을 화면에 출력하게 된다.

프레임 버퍼에 데이터를 기록하는 작업 즉 (그림 4)에서 메모리 복사 작업은 memcpy 시스템 콜을 이용하여 수행한다. 이 memcpy 시스템 콜은 초당 25 프레임율로 재생해야 하는 미디어 특성으로 빈번하게 호출이 되며 이러한 동작은 자원이 제약된 이동 단말에서 성능 감소를 가져올 수 있다. 이러한 단점을 고려하여 memcpy 시스템 콜의 횟수를 줄이기 위해 라인마다 호출되던 것을 한 화면에 대해 호출되도록 하여 한꺼번에 프레임버퍼에 기록을 하도록 변경하였다.



(그림 4) 이동 단말기에서 동작 순서



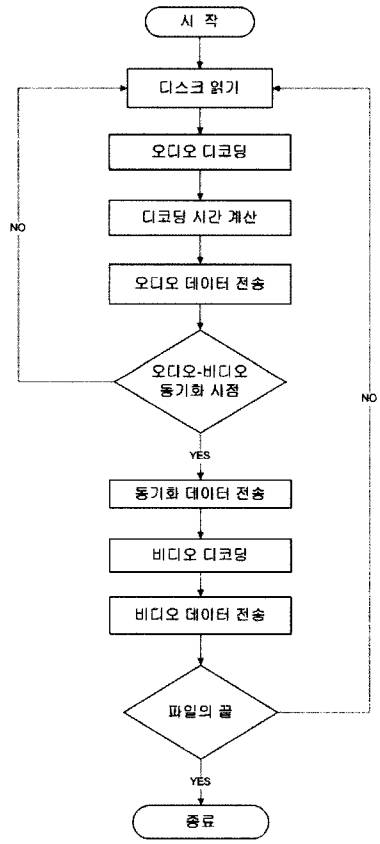
(그림 5) iMedier 동작 시퀀스 다이어그램

3.4 음성 동기화

iMedier 시스템은 멀티미디어 재생 시 동기화 된 음성 처리를 지원해 준다. 기존의 씬클라이언트 환경에서는 문서 기반 환경만을 고려하여 설계되었기 때문에 음성처리를 고려하지 않고 설계되었다. 최근 마이크로소프트사의 터미널 서비스에서는 음성 전송 기능을 추가했지만 음성 동기화를 제공해 주지 않기 때문에 미디어 재생 시 영상과 음성이 동기화 되지 않는 현상이 발생하고 있다. 하지만 iMedier 시스템은 영상과 음성의 동기화를 실시간으로 유지시키므로 스트리밍 미디어 재생을 테스크 탑 수준으로 완벽하게 지원하고 있다.

(그림 5)는 iMedier 시스템의 시퀀스 다이어그램이며 이해를 돕기 위해 객체 상태를 표시하였다. iMedier의 객체는 서버, 수신, 송신, 클라이언트 객체로 구성된다. 서버는 음성과 영상 채널 설정 등 초기화 과정을 마친 후 디스크로부터 영화 데이터를 읽어와 영상과 음성을 분리하는 역다중화 과정을 수행한다.

오디오 데이터는 디코딩을 수행 한 후 디코딩된 음성 데이터를 클라이언트에게 전송을 하며 동기화 시점에 따라 비디오 데이터를 디코딩하여 디코딩된 영상 데이터를 클라이언트에게 전송한다. 동기화 데이터는 영상 데이터를 보내기 전에 음성과 영상의 동기화를 유지하기 위해 전송하게 된다. 클라이언트 역시 영상과 음성 채널 설정 및 영상 정보에 따른 초기화 과정을 마친 후 서버로부터 음성 데이터를 먼저



(그림 6) 동기화 과정 순서도

수신한 후 오디오 장치에 의해 재생을 하게 되며 영상 재생은 RGB로 변환 작업을 한 후에 동기화 데이터에 따른 재생 시점에 영상 장치 보내져 재생을 한다.

미디어 데이터의 음성 표본화율과 영상의 프레임율는 차이가 크다. 즉, 음성의 표본화율은 11.025khz, 22.05kHz 또는 44.1kHz로 인데 반해 영상의 프레임율은 8~60 프레임으로 이루어져 있다. 다시 말해 영상은 초당 최대 60 프레임을 디코딩하고 재생을 하면 되지만 음성은 초당 최대 44.1k번을 디코딩하고 재생해야 한다. 음성에 비해 영상은 1초 동안 넓은 시간 간격을 두고 재생이 이루어지지만 음성은 좁은 시간간격으로 음성 재생을 해야 하므로 동기화 주체는 음성이 된다. 즉 음성 재생시 적절한 시점에 영상 프레임을 디코딩해서 재생을 한다면 동기화가 이루어지는 것이다.

(그림 6)은 iMedier 시스템에서 사용한 영상과 음성이 동기화 되는 과정을 순서도로 표시한 것이다. 디스크에서 읽은 오디오 데이터를 디코딩한 후에 해당 비디오 데이터를 디코딩 한다. 동기화를 위해 디코딩 시간을 계산 한 후 음성 데이터를 클라이언트로 전송을 한다. 만약 비디오 프레임을 디코딩 할 시점이라면 디스크에서 읽은 비디오 데이터를 디코딩 한 후 해당 데이터를 전송하게 되는데 이때 클라이언트 측에도 비디오 디코딩 시점이라는 사실을 알려주는 데이터를 전송한다.

4. 실험환경 및 성능 기준

본 논문에서는 제안된 iMedier의 성능 평가를 유선망 이외에 무선망을 사용하는 환경에서도 진행한다. 유선망은 클라이언트로 데스크 탑PC를 사용하였고 무선망에서는 이동 단말기를 사용하였다. 또한, iMedier 시스템과의 성능 비교를 위하여 VNC 프로토콜을 사용하는 씬클라이언트인 RealVNC의 성능도 측정한다.

4.1 실험 환경

씬클라이언트 컴퓨팅 환경을 구성하기 위해 서버 측에는 iMedier 서버를 작동시키고 클라이언트에서는 터미널 에뮬레이터 역할을 하는 iMedier 클라이언트를 작동시켜 서버에 접속하도록 하였다. iMedier 시스템 플랫폼은 리눅스를 기반[8]으로 하였으며 미디어의 음성 압축 포맷은 MPEG Layer-III을 사용하였으며 출력장치 또한 OSS 라이브러리를 사용하였다.

4.1.1 시스템

<표 1>의 실험에 사용된 서버 및 클라이언트의 사양을 나타낸다. 클라이언트의 측정1은 데스크탑 환경을 나타내고 있고 측정2는 무선 단말의 환경을 보여주고 있다. 성능 측정을 위해 사용한 서버는 측정 1,2 모두 동일하며, 클라이언트 시스템은 측정 1에서 일반 데스크탑 컴퓨터를 사용하였고 측정 2에서는 이동 단말기를 사용하였다. 각각의 시스템 성능은 <표 1>과 같다.

〈표 1〉 시스템 사양

	서 버	클라이언트	
		측정 1	측정 2
CPU	애슬론 MP 2000	펜티엄 IV 1.8GHz	206MHz StrongARM
메모리	1G SDRAM	256MB SDRAM	128MB SDRAM
디스크	Seagate 36G (SCSI)	Seagate 40G (IDE)	32MB Flash ROM
네트워크	100Mbps	100Mbps	11Mbps
OS	Linux (커널 2.4.21)	Linux (커널 2.4.21)	ARM Linux (Linupy)

4.1.2 미디어

측정 1에서 사용된 영상 미디어 파일 정보는 <표 2>와 같다. 측정 2에서는 무선 환경의 성능 제약을 고려하여 다소 낮은 디코딩 성능을 요구하는 미디어 파일을 사용하였으며 해당 미디어 파일 정보는 <표 3>과 같다. 측정 2에서 음성 정보를 기록한 이유는 영상과 음성의 동기화를 이룬 상황에서 측정을 했기 때문이다. 사용된 미디어 파일은 화면의 갱신이 빈번하게 발생되는 뮤직비디오 클립을 이용하였다.

• 측정 1

〈표 2〉 미디어 정보 (데스크탑)

프레임 사이즈	320 X 240
프레임율	30 fps
재생시간	251 s

• 측정 2

〈표 3〉 미디어 정보 (이동단말)

영상	프레임 해상도	240*180
	컬러 수	16bit
	프레임 속도	8fps
	상영시간	90(s)
음성	표본화율	22050Hz
	채널	2bit
	양자화	16bit

4.2 화질 성능 측정 기준

미디어 서비스는 실시간성과 화질에 대한 QoS가 중요하다. 따라서 본 연구에서 이러한 QoS를 보장하기 위해 성능 평가를 하였다. 성능 평가 비교 대상은 AT&T에서 개발한 VNC(Virtual Network Computing) 프로그램인 RealVNC를 사용하였다.

본 연구에서 제안하는 iMedier는 영상의 디코딩은 서버에서 완전히 이루어지고 화면의 갱신은 클라이언트에서 이루어지므로 전송된 데이터의 손실은 곧 화질의 열화를 가져오기 때문에 Slow-Motion Benchmarking을 이용하였다. 따라서 영상 화면의 화질을 측정하기 위해서 1fps와 30 fps 두 가지 재생비율에서 발생하는 패킷 트래픽을 모니터링 한다. 비록 1 fps의 재생율에서 영화를 감상하는 사용자는 없지만 서버에서 클라이언트로 전송되는 완전한 데이터의 사이즈를 참조하기 위해 측정을 한다. 마찬가지로 30 fps에 해당하는 정상 재생을 하여 서버에서 클라이언트로 전송되는 패킷 트래픽을 모니터링 하여 전체 전송된 데이터 사이즈를 비교한다. 비디오 화질은 1 fps의 프레임율에 대해 30 fps의 프레임율의 비율로 측정되어 질 수 있으며 그에 대한 수식은 아래와 같다 [9]. 또한 실시간성 보장 여부를 확인하기 위해 전체 재생 시간을 기록하여 본래 재생 시간과 같은지를 비교한다.

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

수식 1 : 화질 측정

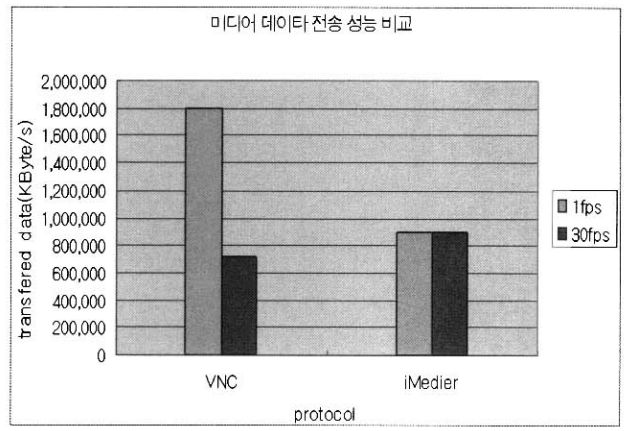
5. 성능 평가

5.1 데스크탑 환경

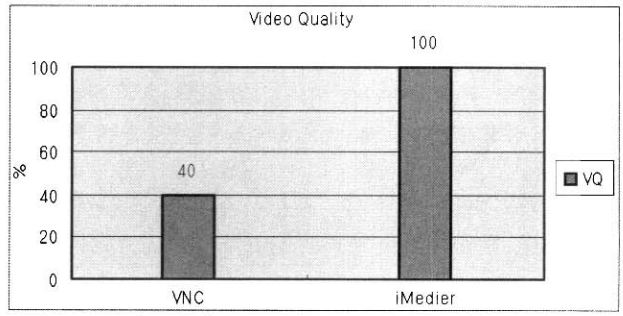
(그림 7)은 1 fps와 30 fps에 대해 미디어 재생 시간동안 클라이언트로 전송된 전체 데이터 크기를 프로토콜 별로 비교한 것이다. 그림에서 보듯이 VNC 에서는 1 fps에서 전체 1.8 GBytes의 데이터를 전송했지만 iMedier는 절반의 크기인 0.8 GBytes의 데이터만을 전송했다. 이러한 수치가 나온 이유는 VNC는 RGB 형태의 데이터를 전송 하지만 본 논문에서 제안한 iMedier는 4:2:0 포맷인 YUV 데이터를 전송하기 때문에 데이터 크기가 반으로 줄어든 것이다. 정상 재생인 30 fps에서도 VNC는 1 fps에 절반에 미치지도 않는 데이터를 전송했지만 iMedier는 거의 비슷한 크기의 데이터를 전송했다. 이러한 실험 결과를 위에서 언급한 수식 1에 적용하여 (그림 8)과 같은 화질 성능에 대한 결과 그래프를 얻을수 있다.

(그림 8)에서 VNC의 화질이 40% 밖에 되지 않는 이유는 정상 재생 (30fps)에서 나온 전체 데이터 사이즈가 1fps에서 측정된 전체 데이터 사이즈의 40% 밖에 되지 않기 때문이다. VNC 에서는 화면 갱신을 위해 사용하는 방식이 미디어 재생 특성을 고려하여 설계되지 않았기 때문이다. 즉 VNC 서버는 업데이트되는 화면 정보를 미디어 재생 비율(30fps)로 얻어 오도록 고려되지 않았으며 업데이트 화면 검출 영역이 화면 전체가 아닌 화면의 1/4 단위로 검출하도록 설계가 되어 있다.

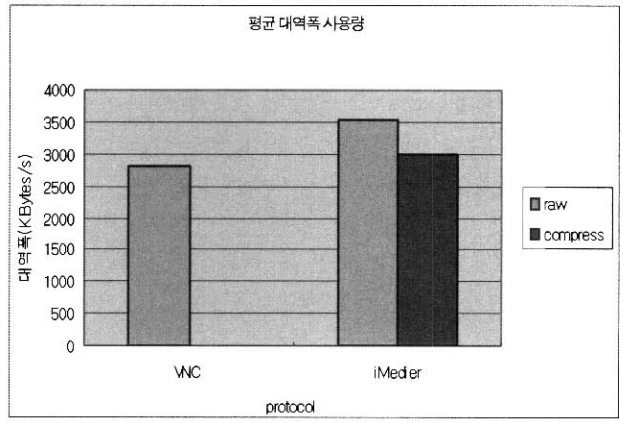
(그림 9)은 평균 대역폭 사용량을 비교한 그래프이다. 압축을 하지 않았을 때는 VNC보다 iMedier가 평균 25%의 데이터를 더 많이 전송하지만 VNC는 서버에서 미디어 재생



(그림 7) 전체 전송된 미디어 데이터 비교



(그림 8) 비디오 화질 성능 비율 비교



(그림 9) 평균 대역폭 사용량

시 생성된 모든 업데이트 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 전송하므로 그 차이를 줄일 수 있다. MPEG 영상의 프레임에는 공간적인 중복성이 존재를 하기 때문에 MPEG 시스템에서는 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. (그림 9)에서 보다시피 압축을 한 후

VNC와 비슷한 수준의 평균 대역폭 사용량을 보여주었다. 향후 최적화된 압축 기법을 적용하므로 더욱 줄어든 평균 대역폭 사용량을 얻을 수 있을 것이다.

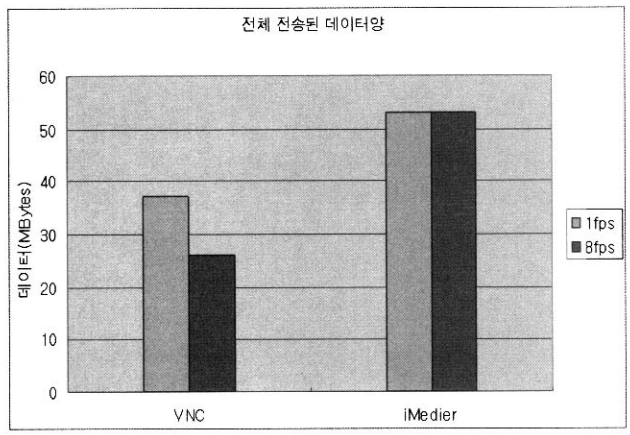
5.2 이동 단말기 환경

(그림 10)은 이동 단말기를 씬클라이언트로 사용 할 때 1 fps (느린 재생)와 8 fps (정상 재생)에 대해 미디어 재생 시간동안 클라이언트로 전송된 전체 데이터 크기를 비교한 것이다. 그림에서 보듯이 VNC 에서는 느린 재생에서 전체 37 MBytes의 데이터를 전송했지만 iMedier는 53 MBytes의 데이터를 전송했다. 데스크탑 환경에서 측정 수치와 비교해 보았을 때 정상적인 수치라면 느린 재생에서 iMedier는 VNC보다 1/2 수준의 전체 데이터 사이즈가 되어야 하지만 오히려 더 많은 데이터를 전송한 것을 볼 수 있다. 이러한 수치가 나온 이유는 VNC가 낮은 대역폭 상황에서는 클라이언트로 보내는 데이터를 네트워크 대역폭을 반영하여 고압축을 통해 전송 데이터의 크기를 최소화하였기 때문에 데스크 탑 환경과는 다르게 낮은 데이터 전송량을 보인 것이다. 이러한 결과는 오히려 미디어 재생의 품질을 낮추는 원인이 된다고 볼 수 있다. 즉, 자원이 제약된 이동 단말 환경은 압축을 해제할 만큼 충분한 자원이 있지는 못하기 때문이다.

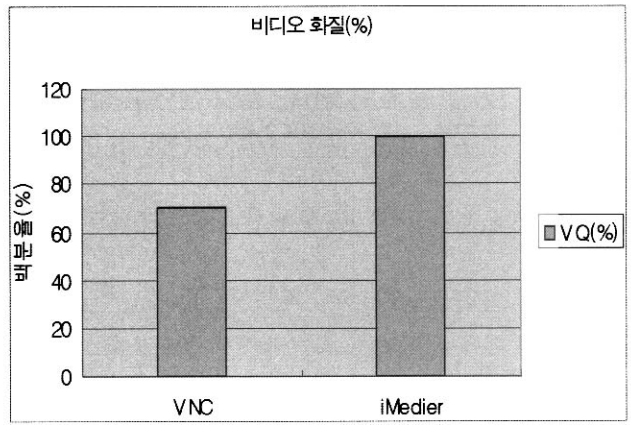
(그림 10)에서 보여주는 iMedier에서 전송된 데이터는 영상뿐만 아니라 음성과 동기화 데이터가 포함되어 있다. 전송된 음성 데이터의 사이즈는 비트율 X 재생 시간으로 계산 될 수 있다. 비트율은 표본화율 X 양자화 비트 X 채널 수 이므로 다음과 같다. 반면 VNC 시스템에서는 음성을 지원하기 않는다.

$$22050\ Hz \times 16bit \times 2bit \times 90(s) = 7.6Mbyte$$

(그림 10)에서 VNC는 손실 없는 완전한 재생인 1 fps에서 37 Mbyte를 전송한 반면에 정상 재생인 8 fps에서는 26Mbyte를 전송했다. 이러한 결과는 데스크탑 컴퓨터와 마찬가지로 VNC가 스트리밍 미디어 재생을 위해 고려하지 않고 설계됐음을 판명할 수 있게 한다. 또한 VNC는 클라이언트의 화면을 갱신하기 위해 초당 영상 재생 프레임율 만큼 화면을 업데이트 하지 않고 화면 갱신 이벤트가 발생할 때만 갱신된 영상 데이터를 전송하도록 설계되어 있다. 이러한 이유로 정상 재생에서 전송된 전체 데이터의 사이즈가 적게 나온 것이며 미디어 재생 시 화질 열화가 발생하여 사용자에게 QoS를 보장하는 스트림을 제공할 수 없었다. 반면에 im미디어의 특성을 반영하기 설계되었기 때문에 정상 재생과 완전한 재생을 비교했을 때 전송한 전체 데이터 사이즈는 같게 나왔다. 이러한 결과는 손실되는 데이터 없이 프레임율에 맞춰 화면을 업데이트했기 때문이며 정상재생 시 모든 데이터를 수신했다는 것을 의미한다. 이러한 결과로부터 iMedier는 재생화질의 품질 측면에서 사용자들에게 QoS가 보장되는 스트림을 제공할 수가 있었다.



(그림 10) 전체 전송된 데이터



(그림 11) 비디오 화질

(그림 11)은 (그림 10)의 결과를 바탕으로 수식 1의 계산에 의해 측정된 비디오 화질에 대한 결과 그래프이다. VNC는 낮은 데이터 전송에도 불구하고 정상 재생에서 70%의 화질을 보여 주었지만 iMedier는 100%의 화질 우수성을 보여주었다.

6. 결론 및 향후연구

오늘날 컴퓨팅 환경의 추세는 개인이나 기업의 기호와 특성이 다양하므로 상호 공존한다. 하지만 사용자의 공통적인 관심사가 멀티미디어에 있음은 틀림없는 사실이다. 이러한 현실 속에 씬클라이언트 컴퓨팅은 여전히 텍스트나 그래픽 기반의 환경만을 제공해 준다. 만약 씬클라이언트 컴퓨팅 환경이 현재 상태로 머물러 있게 된다면 사용자의 관심 속에서 벗어나 도태하게 될 것이다. 본 논문은 이러한 현실을 직시하고 씬클라이언트 컴퓨팅 환경에 멀티미디어 기능을 추가하기 위한 연구를 진행하였다.

본 논문은 씬클라이언트 컴퓨팅 환경에서 미디어 스트리밍 서비스의 향상을 위해 화질 개선에 대한 연구를 진행하였다. 또한 멀티미디어의 필수 지원 항목인 음성 동기화에 대해서 연구하였다.

본 논문이 제안한 iMedier 시스템에 대한 성능 평가를 기존에 사용중인 VNC 시스템과 하였다. 실험 결과는 VNC 시스템은 음성을 지원하지 않을 뿐만 아니라 스트리밍 미디어 재생 화질이 열악한 결과를 나타내었다. 그 이유는 VNC 시스템은 MPEG 미디어의 특성을 고려하여 설계되지 않았기 때문인 것으로 판명하였다. 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.

특히 무선 환경에서 VNC는 전송 데이터의 압축을 통하여 낮은 네트워크 대역폭을 사용한다는 것을 확인하였고, 자원이 제한된 씬클라이언트 단말의 특성상 스트리밍 미디어의 QoS 에 부적절함을 확인하였다. 측정된 실험결과로 부터 VNC는 대역폭이 낮은 네트워크에서는 더욱 낮은 전송률을 보이도록 설계되었음을 알 수 있었다. 이런 VNC 시스템의 문제점은 링크 오류가 많고 대역폭의 기조가 심한 무선 환경에서는 정상적인 미디어 재생이 불가능하다는 것을 의미한다. 본 논문에서 제안한 iMedier 시스템은 음성 동기화 기법으로 영상과 음성이 동기화문제를 해결하였으며 또한, 미디어의 화질을 개선하여 유, 무선 씬클라이언트 사용자들에게 QoS 가 가능한 스트리밍 미디어를 제공할 수 있었다.

향후에는 iMedier에서 YUV 데이터의 압축기법의 최적화로 평균 네트워크 대역폭 사용량을 감소시켜 전체 시스템의 성능확장성을 향상시키는 연구를 하고자한다. 또한 다중 사용자 환경에 적합한 iMedier 시스템을 제한하고 이에 대한 성능 측정을 할 예정이다.

참 고 문 헌

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

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

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

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

[5] 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.

[6] 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.

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

[8] 김병길, 정인범, "멀티미디어 서비스를 위한 씬클라이언트 컴퓨팅의 성능평가 및 비교", 한국정보과학회 2003년 10월.

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

[10] R. W. Scheifler and J. Gettys, "The X Window System," ACM Transactions on Graphics, 1986.

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

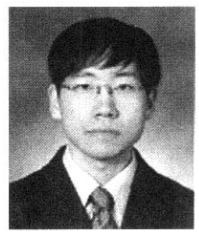
[12] Compaq Computer Corporation, "performance and Sizing of Compaq Servers with Microsoft Windows NT Server 4.0, Terminal Server Edition," Technology Brief, Houston, TX, June, 1998.

[13] J. Danskin, P. Hanrahan, "Higher Bandwidth X," Proceedings of the 1994 ACM Multimedia Conference.

[14] Alexander Ya-li Wong and Margo Seltzer, "Operating System Support for Multi-User Remote, Graphical Interaction," Proceedings of 2000 USENIX Annual Technical Conference.

[15] J. Nielsen, "Multimedia and Hypertext: The Internet and Beyond," Morgan Kaufmann, San Francisco, CA, Jan., 1995.

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



김 병 길

e-mail : bgkim@snslab.kangwon.ac.kr

2003년 강원대학교 정보통신공학과(학사)

2005년 강원대학교 컴퓨터정보통신공학과 (공학석사)

관심분야 : 멀티미디어 신호처리, 내장형 시스템, 유비쿼터스 컴퓨팅

이 좌 형

e-mail : jhlee@snslab.kangwon.ac.kr

2003년 강원대학교 정보통신공학과(학사)

2005년 강원대학교 컴퓨터정보통신공학과 (공학석사)

관심분야 : 운영체제, 병렬처리, 파일시스템, 멀티미디어 시스템

정 인 범

e-mail : ibjung@kangwon.ac.kr

1985년 고려대학교 전자공학과(학사)

1985년~1995년 (주)삼성전자 컴퓨터 시스템사업부 선임연구원

1992년~1994년 한국과학기술원 정보및통신공학과(공학석사)

1995년~2000년 8월 학국과학기술원 전산학과(박사)

2001년~현재 강원대학교 전기전자정보통신공학부 컴퓨터전공 교수

관심분야 : 다중처리기 구조, 운영체제